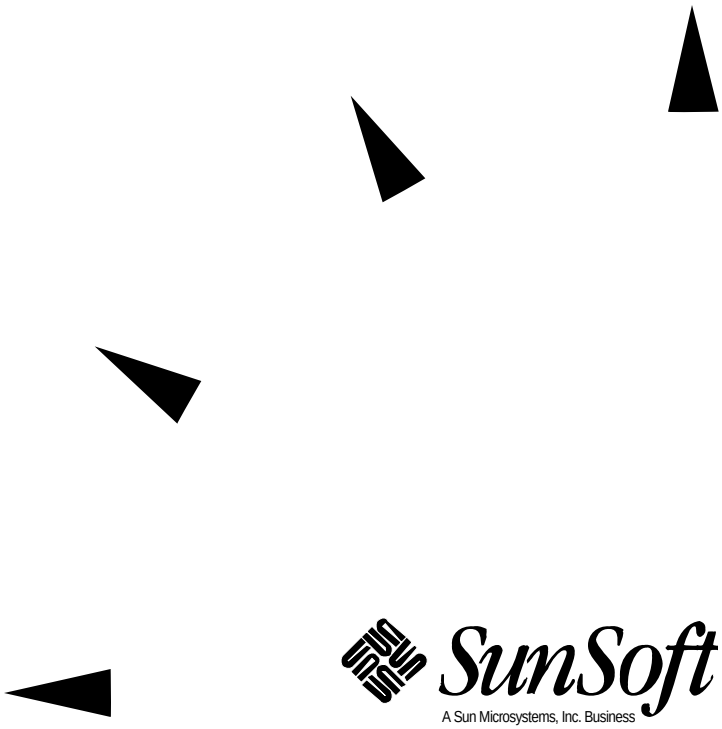
Administration Supplement for Solaris Platforms



2550 Garcia Avenue
Mountain View, CA 94043
U.S.A.

SunSoft
A Sun Microsystems, Inc. Business

Contents

Preface

The *Administration Supplement for Solaris Platforms* describes Solaris™ 2.x system administration tasks that differ on SPARC® and x86 systems. This book supplements the rest of SunSoft's system administration documentation set, which provides information about system administration tasks common to both hardware types.

When a task from the system administration documentation set contains hardware-specific information, you will be referred to this book.

See Table A-1 for a summary of differences between SPARC and x86 systems.

Who Should Use This Book

This book is intended for system administrators who need specific information about SPARC or x86 systems to perform Solaris 2.x system administration tasks.

The *Administration Supplement for Solaris Platforms* is also useful for system administrators who want to know how the Solaris 2.x environment is different between SPARC and x86 hardware.

Before You Read This Book

If you are an experienced SunOS™ 4.x system administrator, refer to the *Solaris 1.x to Solaris 2.x Transition Guide* for information about how to make the transition from administering SunOS 4.x (Solaris 1.x) systems to administering Solaris 2.x systems.

If you are reading this book because you want to know how Solaris 2.x system administration differs between SPARC and x86 hardware, see the next section called "How This Book Is Organized" for a brief description of each chapter.

The specific books in the SunSoft system administration documentation set that refer to this book are:

- *SPARC: Installing Solaris Software*
- *x86: Installing Solaris Software*
- *Common Administration Tasks*
- *File System Administration*
- *Peripherals Administration*

See these books for a full explanation of the system administration topics covered in this book.

For SPARC and x86 hardware configuration information, see the following:

- *SPARC Hardware Platform Guide*
- *x86 Device Configuration Guide*

How This Book Is Organized

This book is divided into two parts:

- Part I—SPARC

  Part I of this book provides information needed to perform system administration tasks on a SPARC system.

- Part II—x86

  Part II of this book provides information about performing system administration tasks on an x86 system.

Table P-1 provides a brief description of each chapter.

Table P-1 Chapter Descriptions

Chapter Number	Description
Part I	
Chapter 1	Information about accessing devices on a SPARC system, that includes a discussion of device names
Chapter 2	Task-specific instructions on booting a SPARC system using different scenarios
Chapter 3	Information about setting up disks on a SPARC system that includes an introduction to disk formatting and steps for setting up a system disk
Chapter 4	Task-specific information about managing file systems on a SPARC system
Part II	
Chapter 6	Information about accessing devices on an x86 system, that includes a discussion of device names
Chapter 7	Task-specific instructions on booting an x86 system using different scenarios
Chapter 8	Information about setting up disks on an x86 system that includes an introduction to disk formatting, and steps setting up a system disk
Chapter 9	Task-specific information about managing file systems on an x86 system

Device Terminology

This book uses the following terminology to describe areas of a disk connected to a Solaris 2.x system.

1. *Slice* refers to a specific area of a Solaris disk. For example, a disk connected to a Solaris 2.x SPARC system can contain up to eight slices. In previous SunOS releases, a specific area of a disk was referred to as a partition.
2. *Fdisk partition* on an x86 system refers to an area of a disk devoted to an entire operating system, such as the Solaris 2.x release. The Solaris `fdisk` partition may contain up to ten slices.

Conventions

- The key referred to as Return is labeled Enter on some keyboards.
- Functions that require you to hold down one key while pressing a second key or mouse button are shown with a hyphen between them (for example, Control-c or Control-d).
- Illustrations that depict screen images are representative; that is, the illustrations might not exactly match what you see on the screen.

What Typographic Changes and Symbols Mean

The following table describes the type changes and symbols used in this book.

Table P-2 Typographic Conventions

Typeface or Symbol	**Meaning**	**Example**
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `system% You have mail.`
`AaBbCc123`	What you type, contrasted with on-screen computer output	`system% su` `Password:`
AaBbCc123	Command-line placeholder: replace with a real name or value	To delete a file, type `rm` *filename*`.`
AaBbCc123	Book titles, new words or terms, or words to be emphasized	Read Chapter 6 in *User's Guide*. These are called *class* options. You *must* be root to do this.

Code samples are included in boxes and may display the following:

UNIX C shell prompt	`system%`
Superuser prompt, C shell	`system#`
UNIX Bourne and Korn shell prompt	`$`
Superuser prompt, Bourne and Korn shells	#

Part 1 — SPARC

Part 1 contains the following chapters:

- Chapter 1, “Accessing Devices on a SPARC System”
- Chapter 2, “Booting a SPARC System”
- Chapter 3, “Setting Up Disks on Your SPARC System”
- Chapter 4, “Managing File Systems on a SPARC System”

Accessing Devices on a SPARC System 1

This chapter describes the following topics:

- Device naming conventions
- Accessing disk and tape devices
- Accessing CD-ROM devices

Overview

This chapter describes the device naming conventions used when accessing peripheral devices such as CD-ROM devices, disks, and tapes.

If you are already familiar with the Solaris 2.x device naming conventions, use the following table to proceed directly to the task-oriented chapters.

1

SPARC *Device Naming Conventions in the Solaris 2.x Environment*

Device naming conventions in the Solaris 2.x environment are based on logical, not physical, names.

- A physical device name is derived from how a device is connected to the system at boot time.
- Logical device names represent the physical devices and are used to access devices when using administration commands.

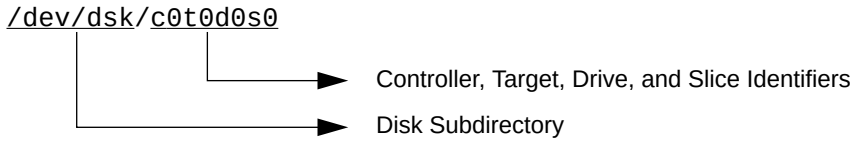
The device files representing the physical devices names are stored in the `/devices` directory with a symbolic link to the logical device names in the `/dev` directory.

The next subsections provide reference material for the various administrative tasks involving disks, tapes, and diskettes described in future chapters of this book and in other books.

SPARC *Accessing Disk Devices*

Many administration commands take arguments that refer to a particular disk or disk slice.

As shown below, you refer to a disk device by specifying the subdirectory to which it is symbolically linked (either `/dev/dsk` or `/dev/rdsk`), followed by a string identifying the particular controller, disk, and slice.



SPARC *Specifying the Disk Subdirectory*

Disk and file administration commands require the use of either a *raw* (or *character*) device interface, or a *block* device interface. The distinction is made by how data is read from the device.

Raw device interfaces transfer only small amounts of data at a time. Block device interfaces include a buffer from which large blocks of data are read at once.

Different commands require different interfaces.

- When a command requires the raw device interface, specify the subdirectory `/dev/rdsk`. (The “r” in `rdsk` stands for “raw.”)
- When a command requires the block device interface, specify the subdirectory `/dev/dsk`.
- When you’re not sure whether a command requires use of `/dev/dsk` or `/dev/rdsk`, check the reference man page for that command.

Table 1-1 shows which interface is required for a few commonly used disk and file-system commands.

Table 1-1 Device Interface Type Required by Some Frequently Used Commands

Command	Interface Type	Example of Use
`df`	Block	`df /dev/dsk/c0t3d0s6`
`fsck`	Raw	`fsck -p /dev/rdsk/c0t0d0s0`
`mount`	Block	`mount /dev/dsk/c1t0d0s7 /export/home/ziggy`
`newfs`	Raw	`newfs /dev/rdsk/c0t0d1s1`
`prtvtoc`	Raw	`prtvtoc /dev/rdsk/c0t0d0s2`

SPARC *Specifying the Slice*

The string you use to identify a specific slice on a specific disk depends on the controller type, either direct (Xylogics®) or bus-oriented (SCSI or IPI). The conventions for both types of controllers are explained in the following subsections.

Disks With Direct Controllers

To specify a slice on a disk with a direct controller, follow the naming convention shown in the figure below.

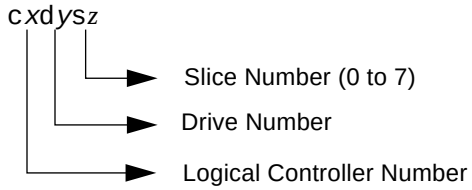


Figure 1-1 Naming Convention for Disks With Direct Controllers

To indicate the whole disk, specify slice 2 (`s2`).

If you have only one controller on your system, *x* will always be 0.

SPARC – Controller numbers are assigned automatically at system initialization. The numbers are strictly logical and imply no direct mapping to physical controllers.

Disks With Bus-Oriented Controllers

To specify a slice on a disk with a bus-oriented controller (SCSI, for instance), follow the naming convention shown in the figure below.

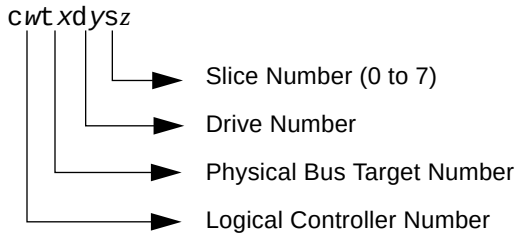


Figure 1-2 Naming Convention for Disks With Bus-Oriented Controllers

If you have only one controller on your system, *w* will always be 0.

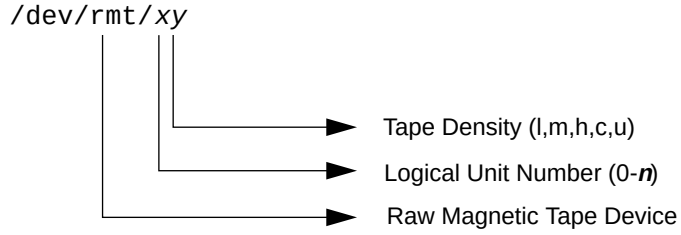
For SCSI controllers, *x* is the target address as set by the switch on the back of the unit, and *y* is the logical unit number (LUN) of the drive attached to the target. If the disk has an embedded controller, *y* is usually 0.

To indicate the whole disk, specify slice 2 (`s2`).

SPARC – Controller numbers are assigned automatically at system initialization. The numbers are strictly logical and imply no direct mapping to physical controllers.

SPARC *Accessing Tape Devices*

Logical tape device files are found in the `/dev/rmt/*` directory as symbolic links from the `/devices` directory.



The first tape device connected to the system is 0 (`/dev/rmt/0`), which may be one of the following types: QIC-11, QIC-24, QIC-150, or Exabyte®.

See *File System Administration* for more information about accessing tape devices.

1

SPARC *Identifying Disk Devices on Your System*

If you can't identify the type of disks connected to a system based on the order in which they were added to the system, use the following commands to discover the disk types.

Use the `ls -l` command to associate a logical device name with its physical device name:

```
# ls -l /dev/dsk/c0t1d0s0
lrwxrwxrwx   1 root     root          51 Nov 15 15:21
/dev/dsk/c0t1d0s0 ->
../../devices/sbus@1,f8000000/esp@0,800000/sd@1,0:a
#
```

In the above example, disk `0` (target `1`) is connected to the first SCSI host adapter (`esp@0...`), which is connected to the first SBus device (`sbus@1...`).

Use the `format` utility to identify the disks that are recognized on the system. The `format` output associates both the physical and logical device name to the disk's marketing name which appears in brackets `<>`. This is an easy way to identify which logical device names represent the disks connected to your system.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t1d0 <SUN1.3G cyl 1965 alt 2 hd 17 sec 80>
          /sbus@1,f8000000/esp@0,800000/sd@1,0
       1. c0t3d0 <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
          /sbus@1,f8000000/esp@0,800000/sd@3,0
Specify disk (enter its number):
```

See *Peripherals Administration* for information on using the `format` utility.

Booting a SPARC System 2

This chapter describes information needed to use the OpenBoot™ PROM monitor and step-by-step instructions for booting a SPARC system.

If you are already familiar with the PROM monitor and how to use it, use the following table to proceed directly to the task-oriented sections.

2

SPARC *The Boot PROM*

Each system has a PROM (programmable read-only memory) chip with a program called the *monitor*. The monitor controls the operation of the system before the kernel is available. When a system is turned on, the monitor runs a quick self-test procedure that checks things such as the hardware and memory on the system. If no errors are found, the system begins the automatic boot process.

SPARC – Some older systems may require PROM upgrades before they will work with the SunOS system software. Contact your local service provider for more information.

SPARC *The PROM Prompt*

When the system is halted, the PROM monitor prompt is displayed and the type of prompt depends on your system type. Older Sun systems like the Sun4/*nnn* series, use the greater than sign (>) as the PROM prompt. Newer Sun systems use `ok` as the PROM prompt but support the `>` prompt.

- **To switch from the > prompt to the `ok` prompt on newer Sun systems type the following command:**

```
> n
ok
```

SPARC How to Find the PROM Release for a System

- **Type `banner` from the `ok` PROM prompt and press Return.**
  Hardware configuration information, including the release number of the PROM, is displayed.

```
ok banner
SPARCstation 2, Type 4 Keyboard
ROM Rev. 2.2, 16 MB memory installed, Serial #426751
Ethernet address 8:0:20:e:fd:7c HostID 55411df8
```

♦ **Type** `kb` **from the** > **PROM prompt on Sun4/***nnn* **systems and press Return.**

```
> kb
Sun SPARCsystem 400.
ROM Rev 4.1.1, 64MB memory installed, Serial #4206788.
Ethernet address 8:0:20:10:37:c6, Host ID 244030BD.
```

If a system installed with Solaris 2.x software does not find the `/kernel/unix` boot program, you may need to change the `boot-from` setting in the PROM.

SPARC How to Change the `boot-device` PROM Setting

1. **Halt the system using the** `init 0` **command.**
   The PROM prompt is displayed.
2. **If the** > **PROM prompt is displayed, type** `n` **and press Return.**
   The `ok` PROM prompt is displayed.
3. **Type** `setenv boot-device disk`*n* `/kernel/unix` **and press Return.**
   The `boot-from` setting is changed.
4. **Type** `reset` **and press Return.**
   The `boot-device` setting is written to the PROM.

See the `monitor(1M)` manual page for a complete list of PROM commands.

SPARC How to Run the Self-Test Program

From the `ok` PROM prompt:

♦ **Type** `reset` **and press Return.**
The self-test program is run and the system is rebooted.

From the > PROM prompt on a Sun 4/*nnn* system:

♦ **Type** `k2` **and press Return**
The self-test program is run and the system is rebooted.

2

SPARC *Boot Scenarios*

Table 2-1 provides different boot scenarios for a SPARC system.

Table 2-1 Boot Scenarios

Boot Type	Boot Command	This Type Of Boot Is Needed When...
Booting to multiuser state	`ok` **`boot`** `>` **`b`**	Changing kernel configuration information by modifying the `/etc/system` file.
Booting to single-user state	`ok` **`boot -s`** `>` **`b -s`**	Performing file system maintenance, such as performing a backup or restoring system data. Repairing a system file such as `/etc/passwd`.
Booting interactively	`ok` **`boot -a`** `>` **`b -a`**	Repairing a system configuration file such as `/etc/system`.
Booting after adding new hardware	See example on page 21	Adding hardware to the system. (Also know as a *reconfiguration boot*.)
Booting from a local CD-ROM drive	See example on page 13	Installing a new release of the operating system (this includes upgrading to a new operating system release). Repairing a critical system file which is causing system boot failure.
Booting from the network	See example on page 13	Installing a new release of the operating system from an installation server (including upgrades). Repairing an important system file when a local CD-ROM drive is not available.
Booting `kadb`	`ok` **`boot kadb`** `>` **`b kadb`**	Booting the kernel debugger (`kadb`) to track down a system problem.
Forcing a crash dump and rebooting	See example on page 15	Recovering from a hung system and you want to force a crash dump.

SPARC – Booting brings the system to run level 3, multiuser level with resources shared, unless specified otherwise.

SPARC *Booting a System*

If a system is powered off, turning it on starts the multiuser boot sequence. The following procedures show how to boot to different states from the ok PROM prompt. Type n to display the ok prompt on a newer Sun system if the prompt is >, and then follow the appropriate steps.

SPARC How to Boot to Multiuser State (Run Level 3)

Booting to multiuser state is usually done after halting the system or performing some system hardware maintenance task. This is the default boot level where all resources are available and users can log into the system.

♦ **At the** ok **PROM prompt type** boot **and press Return.**
The automatic boot procedure starts, displaying a series of startup messages. The system is brought up in multiuser state.

SPARC How to Boot to Single-User State (Run Level S)

Booting to single-user state is usually done after performing some system maintenance task such as backing up the system. At this level only some file systems are mounted and users cannot log into the system.

1. **At the** ok **PROM prompt type** boot -s **and press Return.**
The system boots to single-user state and prompts you for the root password.

```
ok boot -s
INIT: SINGLE USER MODE
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance):   xxxxxxx
```

2. **Type the root password and press Return.**

SPARC – To bring the system up to multiuser state after the system maintenance task is performed, press Control-d.

2

SPARC How to Boot Interactively

You may want to boot interactively to make a temporary change to the system file or the kernel. Booting interactively lets you test your changes and recover easily if you have any problems.

1. **At the** `ok` **PROM prompt, type** `boot -a` **and press Return.**
   The boot program prompts you interactively.
2. **Press Return to use the default kernel (**`/kernel/unix`**) as prompted, or type the name of the kernel to use for booting and press Return.**
3. **Press Return to use the default** `/etc/system` **file as prompted, or type the name of the system file and press Return.**
4. **Press Return to use the default modules directory path as prompted, or type the path for the modules directory and press Return.**
5. **Press Return to use the default** `/etc/path_to_inst` **file as prompted, or type an alternate file and press Return.**
6. **Press Return to use the default root file system type as prompted:** `ufs` **for local disk booting or** `nfs` **for diskless clients.**
7. **Press Return to use the default physical name of the root device as prompted, or type the device name.**

SPARC *Example: How to Boot Interactively*

In this example, the default choices (shown in square brackets `[]`) are accepted by pressing Return:

```
ok boot -a
Enter filename [/kernel/unix]:
(Copyright notice)
Name of system file [/etc/system]:
Name of default directory for modules [/kernel /usr/kernel]:
Enter name of device instance number file [/etc/path_to_inst]:
root filesystem type [ufs]
Enter physical name of root device
[/sbus@1,f8000000/esp@0,800000/sd@0,0:a]:
Configuring network interfaces:  le0
Hostname: pluto
```

```
(fsck messages)
The system is coming up.  Please wait
(More messages)
pluto login:
```

SPARC How to Boot from Local CD Drive or the Network

Booting from the local CD drive or the network is required when you have to repair an important system file that is preventing the system from booting successfully. This situation requires booting from a local CD-ROM drive or the network to recover. Recovering from a invalid `/etc/passwd` file is used as an example.

Substitute the device name of the file system to be repaired for the *devicename* variable identified in the procedures below. Refer to Chapter 1, "Accessing Devices on a SPARC System," if you need help identifying your device names.

1. **Boot from the Solaris 2.x installation CD.**
   a. **Insert the Solaris 2.x installation CD into the CD caddy.**
   b. **Insert the CD caddy into the CD-ROM drive.**
   c. **Boot from the installation CD in single-user mode. Press Return.**

```
ok boot cdrom -sw
```

2. **Or, boot from the network if an installation server or remote CD drive is available.**

```
ok boot net
```

3. **Mount the `root` file system that has the invalid `passwd` file.**

```
# mount /dev/dsk/devicename /a
```

4. **Change to the newly mounted `etc` directory.**

```
# cd /a/etc
```

5. Set the terminal type.

```
# TERM=sun
# export TERM
```

6. Make the necessary change to the `/etc/passwd` file using an editor.

7. Change to the `root` directory.

8. Unmount the `/a` directory.

```
# umount /a
```

9. Reboot the system.

```
# init 6
```

SPARC How to Abort a Booting Process

Occasionally, you may need to abort the booting process. The specific abort key sequence depends on your keyboard type. For example, you can press Stop-A or L1-A. On terminals, press the Break key.

1. Type the abort key sequence for your system.
The monitor displays the `ok` PROM prompt.

```
ok
```

2. Type the `sync` command at the `ok` prompt to synchronize the disks.

```
> n
ok sync
```

3. **Or, type the `g0` command at the > prompt on a Sun 4/*nnn* system to synchronize the disks.**

```
> g0
```

4. **Once you see the `syncing file systems. . .` message, press the abort key sequence for your system again.**
5. **Type the appropriate boot command to restart the boot process.**

SPARC How to Force a Crash Dump and Reboot the System

Use the following procedure if your system hangs and you want to force a crash dump. This procedure assumes that you have enabled the savecore feature. See *Common Administration Tasks* for information on the savecore feature.

1. **Type the abort key sequence for your system.**
   The monitor displays the `ok` PROM prompt.

```
ok
```

2. **Type the `sync` command at the `ok` prompt to synchronize the disk and write the crash dump.**

```
> n
ok sync
```

3. **Or, type the `g0` command at the > prompt on a Sun 4/*nnn* system to synchronize the disk and write the crash dump.**

```
> g0
```

   After the crash dump is written to disk, the system will continue to reboot.

2

SPARC *Displaying Boot Information*

The `dmesg` command displays information that is provided during the boot process. The boot information is displayed as messages on the system console and provides helpful information for troubleshooting system boot problems.

This information is stored in the `/var/adm/messages` file.

An example of `dmesg` output is displayed below.

Boot date and time
SunOS release information
Copyright
Physical memory
Available memory
Ethernet address
System type
/kernel/unix probes devices

```
# dmesg
May 31 10:32
SunOS Release 5.4 Version[UNIX(R) System V Release 4.0]
Copyright (c) 1983-1994, Sun Microsystems, Inc.
mem = 28672K (0x1c00000)
avail mem = 26701824
Ethernet address = 8:0:20:9:5:d9
root nexus = Sun 4_65
sbus0 at root: obio 0xf8000000
dma0 at sbus0: SBus slot 0 0x400000
esp0 at sbus0: SBus slot 0 0x800000 SBus level 3 sparc ipl 3
sd1 at esp0: target 1 lun 0
sd1 is /sbus@1,f8000000/esp@0,800000/sd@1,0
    <SUN1.3G cyl 1965 alt 2 hd 17 sec 80>
sd3 at esp0: target 3 lun 0
sd3 is /sbus@1,f8000000/esp@0,800000/sd@3,0
    <SUN0207 cyl 1254 alt 2 hd 9 sec 36>
root on /sbus@1,f8000000/esp@0,800000/sd@1,0:a fstype ufs
zs0 at root: obio 0xf1000000 sparc ipl 12
zs0 is /zs@1,f1000000
zs1 at root: obio 0xf0000000 sparc ipl 12
zs1 is /zs@1,f0000000
cgsix0 at sbus0: SBus slot 2 0x0 SBus level 5 sparc ipl 7
cgsix0 is /sbus@1,f8000000/cgsix@2,0
cgsix0: screen 1152x900, single buffered, 1M mappable, rev 2
le0 at sbus0: SBus slot 0 0xc00000 SBus level 4 sparc ipl 5
le0 is /sbus@1,f8000000/le@0,c00000
le1 at sbus0: SBus slot 1 0xc00000 SBus level 4 sparc ipl 5
le1 is /sbus@1,f8000000/le@1,c00000
dump on /dev/dsk/c0t1d0s1 size 33308K
pseudo-device: vol0
vol0 is /pseudo/vol@0
fd0 at root: obio 0xf7200000 sparc ipl 11
```

Setting Up Disks on Your SPARC System 3

This chapter describes the disk slices found on a SPARC system running the Solaris 2.x environment, and the steps for setting up a system disk connected to a Solaris 2.*x* system.

If you are already familiar with the formatting utility and disk slice assignments, use this table to proceed directly to the task-oriented sections. Otherwise, continue to the next page.

If you are not familiar with hard disk concepts or terminology, or how and why disks are divided into slices, read *Peripherals Administration* before attempting any tasks.

3

SPARC *Disk Preparation Steps*

Before you can use a disk to store and retrieve data it must be formatted. Formatting can involve three separate processes:

- *Low-level Formatting* – Writing format information to the disk
- *Partitioning* – Dividing the disk into manageable areas
- *Surface analysis* – Compiling an up-to-date list of disk defects

Disks purchased from Sun are already formatted and provide a manufacturer's defect list that is loaded automatically. Using the `suninstall` utility to install Solaris 2.x will create the slices needed for system operation.

After installation, you may need to repartition a disk. See the *Peripherals Administration* guide for information on repartitioning a disk.

SPARC *The* `format` *Utility*

The Solaris operating system tool for maintaining disks is called the `format` utility. This name is something of a misnomer, however, because `format` enables you to analyze, partition, and label disks, as well as format them. Its repair option can even help you recover from file system difficulties brought on by a defective disk.

See the *Peripherals Administration* guide for reference information on the `format` utility.

SPARC *Special Areas of the Disk*

The beginning of the disk is reserved for an important use, storing the disk label or Volume Table of Contents (VTOC) that describes the entire disk.

The last two cylinders are set aside for diagnostic use, as well as for storing the disk defect list and a backup copy of the label.

SPARC *Dividing a Disk into Slices*

When you set up a disk's slices, you choose not only the size of each slice, but also which slice to use for a particular purpose. Your decisions about these matters depend on how you intend to use the system to which the disk is attached.

SPARC *Customary Slices*

Disks are divided into eight disk slices, each assigned a conventional use. These slices are numbered 0 through 7. Table 3-1 summarizes the contents of each disk slice.

Table 3-1 Customary Slice Disk Assignment

Slice	File System	Purpose
0	`root`	Holds the files and directories that make up the operating system.
1	swap	Provides virtual memory, or *swap space*. Swap space is used when a new program you need to run is too large to fit in a computer's memory at the same time as other programs that are already running. When this happens, the operating system "swaps" different programs from the computer's memory to the disk—and vice versa—as needed.
2	—	Used by the operating system to reference the entire disk. It is defined automatically by Sun's `format` and installation programs and should not be altered.
3	`/export`	Holds alternative versions of the operating system. These alternative versions are required by client machines whose architectures differ from that of the server.
4	`/export/swap`	Provides virtual memory space for the client rather than for the server.
5	`/opt`	Holds application software that is added to a system. If there is not enough room on the disk to put the `/opt` file system in slice 5, the `/opt` directory is put in slice 0.
6	`/usr`	Holds operating system commands—also known as *executables*— that are run by users. This slice also holds documentation, system programs (`init` and `syslogd`, for example) and library routines.
7	`/export/home` or `/home`	Holds files that are created by users.

The Solaris installation program provides slice size recommendations based on the software you select for installation.

3

SPARC *Instructions for Setting Up Disks*

This section includes instructions for performing tasks related to disks.

You will find an example of the screen input and output associated with the task with each set of instructions. Your screen output will vary based on your system configuration.

SPARC – You must be superuser to perform the following procedures.

SPARC *Adding a Secondary Disk*

Adding a secondary disk entails some or all of these procedures:

- Performing a reconfiguration boot
- Extracting a defect list
- Formatting a disk
- Dividing a disk into slices and labeling the disk

Performing a reconfiguration boot is covered on the following pages. This step is necessary when adding new hardware to the system so the appropriate device files are linked in the `/dev` directory.

The other procedures listed above are covered in the *Peripherals Administration* manual.

SPARC *Adding a System Disk*

If your system disk is damaged and you do not want to reinstall the Solaris 2.x environment, you can restore your system disk using the following steps:

- Physically connecting the new disk drive
- Booting the Solaris 2.x installation media from CD or the network to single-user mode
- Repartitioning the disk to create partitions for root and swap and restoring the `root` and `/usr` file systems

SPARC How to Perform a Reconfiguration Boot

1. **Load the new device driver, if necessary, following the instructions included with the hardware.**
2. **Create the `/reconfigure` file that will be read when the system is booted.**

```
# touch /reconfigure
```

The `/reconfigure` file will cause the SunOS software to check for the presence of any newly installed peripheral devices when you power on or boot your system later.

3. **Shut down the system.**

```
# /usr/sbin/shutdown -y -g30 -i0
```

SPARC – The 0 in `i0` is a zero.

In the example above, the command sends a message to all users who are logged in stating they have 30 seconds (`-g30`) before the system begins to shut down. The `ok` or > prompt is displayed after the operating environment is shut down.

4. **Turn off power to the system after the `ok` or > prompt is displayed.**
Refer to the hardware installation guide that accompanies your system for the location of the power switch.
5. **Turn off power to all external peripheral devices.**
For location of power switches on any peripheral devices, refer to the hardware installation guides that accompany your peripheral devices.
6. **Install the peripheral device.**
Refer to the hardware installation guides that accompany the peripheral devices for information on how to install and connect those devices.
7. **Turn on the power to all external peripherals.**
8. **Turn on the power to the system. The system will boot and you will be shown the login prompt.**

3

SPARC How to Add a System Disk

The following procedure assumes your system disk has become disabled and you need to replace it with a new one. The best way to solve this problem without reinstalling the Solaris environment is to replace the disk and restore your file systems from a backup medium. See *File System Administration* for information about restoring file systems to disk.

SPARC – This procedure displays screen output as an example and will vary from system to system.

The disk used as an example in this procedure is 200 Mbytes in size and is repartitioned into three slices:

- 50-Mbyte `root` file system
- 50-Mbyte swap area
- 100-Mbyte `/var` file system

1. **Connect the disk to the system and check the physical connections.**
   Make sure that any cables connecting the disk unit to the system are securely attached. See *SPARC Hardware Platform Guide* about hardware configuration requirements that should be checked before installation.

2. **Boot the Solaris 2.4 media from the CD-ROM drive or the network to single-user mode using one of the following commands**:
   ```
   boot cdrom -s
   boot net -s
   ```

3. **Enter the `format` utility to repartition the disk, if necessary, by typing `format` at the root prompt and pressing Return.**
   See *Peripherals Administration* for information on using the `format` utility to repartition a disk.

   a. **Enter the number of the disk that you want to repartition from the list displayed on your screen.**

   b. **Enter the partition menu (which lets you set up the slices) by typing `partition` at the `format>` prompt.**

   c. **Display the current partition (slice) table by typing `print` at the `partition>` prompt.**

d. Modify the current table by typing `modify` **at the** `partition>` **prompt. Type** `1` **at the** `Choose base` **prompt to set the disk to all Free Hog.**

```
Part         Tag    Flag   Cylinders   Size      Blocks
  0         root    wm     0             0       (0/0/0)
  1         swap    wu     0             0       (0/0/0)
  2       backup    wu     0 - 1253    198.39MB  (1254/0/0)
  3   unassigned    wm     0             0       (0/0/0)
  4   unassigned    wm     0             0       (0/0/0)
  5   unassigned    wm     0             0       (0/0/0)
  6          usr    wm     0             0       (0/0/0)
  7   unassigned    wm     0             0       (0/0/0)
Do you wish to continue creating a new partition
table based on above table[yes]? yes
```

e. Create a new partition table by typing `yes` **at the above prompt.**

f. Identify the Free Hog partition (slice) and the sizes of the root and swap slice when prompted.

```
Free Hog partition[6]? 5
Enter size of partition ‘0’ [0b, 0c, 0.00mb]: 50mb
Enter size of partition ‘1’ [0b, 0c, 0.00mb]: 50mb
Enter size of partition ‘3’ [0b, 0c, 0.00mb]:
Enter size of partition ‘4’ [0b, 0c, 0.00mb]:
Enter size of partition ‘6’ [0b, 0c, 0.00mb]:
Enter size of partition ‘7’ [0b, 0c, 0.00mb]:
```

The new partition table is displayed.

```
Part         Tag    Flag   Cylinders     Size      Blocks
  0         root    wm       0 - 316     50.15MB  (317/0/0)
  1         swap    wu     317 - 633     50.15MB  (0/0/0)
  2       backup    wu       0 - 1253    198.39MB  (1254/0/0)
  3   unassigned    wm       0             0       (0/0/0)
  4   unassigned    wm       0             0       (0/0/0)
  5   unassigned    wm     634 - 1253     98.09MB  (620/0/0)
  6          usr    wm       0             0       (0/0/0)
  7   unassigned    wm       0             0       (0/0/0)
```

g. Answer yes to the `Okay to make this the current partition table?` **prompt.**

h. Name the partition table using quotes.

```
Enter table name (remember quotes): "disk0"
```

i. Label the disk with the new partition table when you have finished allocating slices on the new disk.

```
partition> label
Ready to label disk, continue? yes
```

j. Quit the `partition` **menu.**

k. Quit the `format` **menu.**

4. Create the file systems for the newly partitioned disk using the `newfs` **command.**

```
# newfs /dev/rdsk/devicename
```

5. Run the `installboot` **command to install a boot block for the** `root` **file system.**

```
# installboot /usr/lib/fs/ufs/bootblk /dev/rdsk/devicename
```

The root slice must have a boot block. See “Installing a Boot Block on a ufs File System” on page 26 for more information on `installboot`.

See Chapter 4, “Managing File Systems on a SPARC System,” for step-by-step instructions on restoring the `root` or `/usr` file systems.

Managing File Systems on a SPARC System 4

This chapter describes how to install a boot block on a `ufs` file system and format diskettes for use with both `ufs` and DOS file systems.

Use this table to proceed directly to the task-oriented sections.

4

SPARC Creating File Systems

SPARC Installing a Boot Block on a `ufs` File System

When the system is booted, the primary boot program, `bootblk`, executes and loads the `/ufsboot` program into memory. The boot block is normally installed during system installation. However, you may need to use this procedure if the root file system becomes damaged because you cannot restore the boot block from tape.

SPARC How to Install a Boot Block on a `ufs` File System

1. **Choose the slice you want to use as the bootable file system.**
2. **If necessary, create the `ufs` file system.**
   See *File System Administration* for instructions on creating a new file system.
3. **Type `installboot /usr/lib/fs/ufs/bootblk /dev/rdsk/`*devicename* and press Return.**
   The contents of the file `/usr/lib/fs/ufs/bootblk` (the boot block) are installed in sectors 1-15 of the slice you specify. Sector 0 contains the disk label.

   If you install the boot block on an alternate file system that you want to use for booting, specify the corresponding disk or slice when booting:

```
ok boot disk1
```

SPARC Creating a File System on a Diskette

SPARC How to Format a Diskette for Use With a `ufs` File System

Use double-sided high-density 3.5-inch diskettes (diskettes are marked "DS, HD").

Caution – Reformatting destroys any files already on the diskette.

1. **Insert a 3.5-inch DS, HD diskette in the diskette drive.**

2. **Type** `fdformat` **and press Return.**
   The following messages are displayed:

```
Press return to start formatting floppy in
/vol/dev/rdiskette0/unlabeled.
```

SPARC – If there is a file system on the diskette and the diskette is mounted, you will get an error from `fdformat`. You must unmount the diskette by using `umount` and run `fdformat` again.

3. **Press Return.**
   While the diskette is being formatted, a series of dots (`...`) is displayed. When formatting is complete, the prompt is redisplayed.

```
pluto% fdformat
Press return to start formatting floppy in
/vol/dev/rdiskette0/unlabeled.
.....................................................
.................
pluto%
```

4. **Run the** `newfs` **command to create a** `ufs` **file system on the diskette.**

SPARC How to Format a Diskette That Can Be Read on a DOS System

Caution – Reformatting destroys any files already on the diskette.

1. **Insert a 3.5-inch DS, HD diskette in the diskette drive.**

2. **Type** `fdformat -d` **and press Return.**
   The following message is displayed.

```
Press return to start formatting floppy in
/vol/dev/rdiskette0/unlabeled.
```

3. **Press Return.**
   While the diskette is being formatted, a series of dots (. . .) is displayed. When formatting is complete, the prompt is redisplayed.

```
pluto% fdformat -d
Press return to start formatting floppy in
/vol/dev/rdiskette0/unlabeled.
.....................................................
.................
pluto%
```

SPARC *Restoring File Systems*

Restoring a damaged `root` or `/usr` file system from a backup tape is not like other types of restore operations because the programs you need to run are in the damaged file system. You must boot the system from the installation CD and reload the file system, or you must connect the drive to another system and reload the file system from there.

Substitute the device name of the file systems to be repaired for the *devicename* variable identified in the procedures below. Refer to Chapter 1, "Accessing Devices on a SPARC System," if you need help identifying your device names.

SPARC How to Restore the `root` and `/usr` File Systems

1. **Boot from the Solaris 2.x installation CD.**
   a. **Insert the Solaris 2.x installation CD into the CD caddy.**
   b. **Insert the CD caddy into the CD-ROM drive.**
   c. **Use the following command to boot from the installation CD in single-user mode. Press Return.**

```
ok boot cdrom -sw
```

2. **Use the `format` utility to repair and reallocate slices on the disk from the single-user mode # prompt.**
   a. **Locate and fix any bad blocks.**

See the *Peripherals Administration* guide for information on using `format` to repair a disk.

b. Divide the disk into slices with the `format` command.

3. Make a new file system with the `newfs` command and check the file system with the `fsck` command for each slice except swap.

```
# newfs /dev/rdsk/devicename
# fsck /dev/rdsk/devicename
```

4. Run the `installboot` command to install a boot block for the file system.

```
# installboot /usr/lib/fs/ufs/bootblk /dev/rdsk/devicename
```

The root slice must have a boot block. See "Installing a Boot Block on a ufs File System" on page 26 for more information on `installboot`.

5. Type `mount /dev/dsk/`*devicename* `/mnt` and press Return.
The file system is mounted on a temporary mount point.

SPARC – To mount the file system, you specify the block device directory (`/dev/dsk`), not the raw device directory.

---

6. Type `cd /mnt` and press Return.
You have changed to the mount-point directory.

7. Type `tapes` and press Return.
This command creates the tape device entries.

8. Write-protect the tapes for safety.

9. Type `ufsrestore rvf /dev/rmt/`*unit* and press Return.
The level 0 tape is restored.

10. Remove the tape and load the next level tape in the drive.
Always restore tapes starting with 0 and continuing from lowest to highest until you reach the highest level.

11. **Type** `ufsrestore rvf /dev/rmt/`*unit* **and press Return.**
    The next level tape is restored.

12. **Repeat steps 9 and 10 for each additional tape.**

13. **Type** `rm restoresymtable` **and press Return.**
    Removes the `restoresymtable` file that is created and used by `ufsrestore` to check point the restore.

14. **Type** `cd /` **and press Return.**

15. **Type** `umount /mnt` **and press Return.**
    The root file system is unmounted.

16. **Type** `fsck /dev/rdsk/`*devicename* **and press Return.**
    The restored file system is checked for consistency.

17. **Insert a new tape in the tape drive.**

18. **Type** `ufsdump 0uf /dev/rmt/`*unit* `/dev/rdsk/`*devicename* **and press Return.**
    A level 0 backup is performed. Always do an immediate backup of a newly created file system because `ufsrestore` repositions the files and changes the inode allocation.

19. **Repeat steps 5 through 19 for the** `/usr` **file system, if necessary.**

20. **Type** `init 6` **and press Return.**
    The system is rebooted.

Part 2 — x86

Part 2 contains the following chapters:

- Chapter 5, “Accessing Devices on an x86 System”
- Chapter 6, “Booting an x86 System”
- Chapter 7, “Setting Up Disks on Your x86 System”
- Chapter 8, “Managing File Systems on an x86 System”

Accessing Devices on an x86 System 5

This chapter includes the following topics:

- Device naming conventions
- Accessing disk and tape devices
- Accessing CD-ROM devices

This chapter describes the device naming conventions used when accessing peripheral devices such as CD-ROM devices, disks, and tapes.

If you are already familiar with the Solaris 2.x device naming conventions, use the following table to proceed directly to the task-oriented chapters.

5

x86 Device Naming Conventions in the Solaris 2.x Environment

Device naming conventions in the Solaris 2.x environment are based on logical, not physical, names.

- A physical device name is derived from how a device is connected to the system at boot time.
- Logical device names represent the physical devices and are used to access devices when using administration commands.

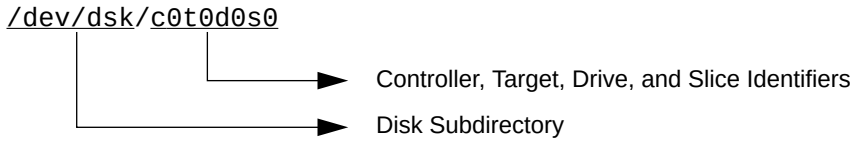
The device files representing the physical devices names are stored in the `/devices` directory with a symbolic link to the logical device names in the `/dev` directory.

The next subsections provide reference material for the various administrative tasks involving disks, tapes, and diskettes described in future chapters of this book and in other books.

x86 Accessing Disk Devices

Many administration commands take arguments that refer to a particular disk or disk slice.

As shown below, you refer to a disk device by specifying the subdirectory to which it is symbolically linked (either `/dev/dsk` or `/dev/rdsk`), followed by a string identifying the particular controller, disk, and slice.



x86 Specifying the Disk Subdirectory

Disk and file administration commands require the use of either a *raw* (or *character*) device interface, or a *block* device interface. The distinction is made by how data is read from the device.

Raw device interfaces transfer only small amounts of data at a time. Block device interfaces include a buffer from which large blocks of data are read at once.

Different commands require different interfaces.

- When a command requires the raw device interface, specify the `/dev/rdsk` subdirectory. (The “r” in `rdsk` stands for “raw.”)
- When a command requires the block device interface, specify the `/dev/dsk` subdirectory.
- When you’re not sure whether a command requires use of `/dev/dsk` or `/dev/rdsk`, check the reference man page for that command.

Table 5-1 shows which interface is required for a few commonly used disk and file-system commands.

Table 5-1 Device Interface Type Required by Some Frequently Used Commands

Command	Interface Type	Example of Use
`df`	Block	`df /dev/dsk/c0t3d0s6`
`fsck`	Raw	`fsck -p /dev/rdsk/c0t0d0s0`
`mount`	Block	`mount /dev/dsk/c1t0d0s7 /export/home/ziggy`
`newfs`	Raw	`newfs /dev/rdsk/c0t0d1s1`
`prtvtoc`	Raw	`prtvtoc /dev/rdsk/c0t0d0s2`

x86 *Specifying the Slice*

The string you use to identify a specific slice on a specific disk depends on the controller type, either direct (IDE) or bus-oriented (SCSI). The conventions for both types of controllers are explained in the following subsections.

Disks With IDE Controllers

To specify a slice on a disk with an IDE controller, follow the naming convention shown in the figure below.

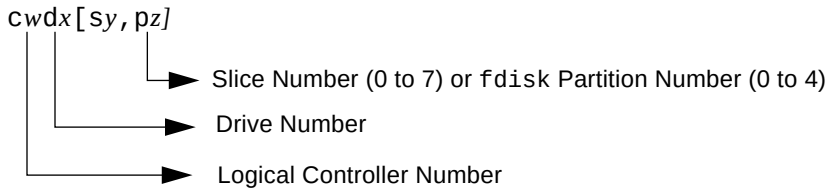


Figure 5-1 Naming Convention for Disks With IDE Controllers

To indicate the entire Solaris `fdisk` partition, specify slice 2 (`s2`).

If you have only one controller on your system, *w* will always be 0.

Disks With SCSI Controllers

To specify a slice on a disk with a SCSI controller, follow the naming convention shown in the figure below.

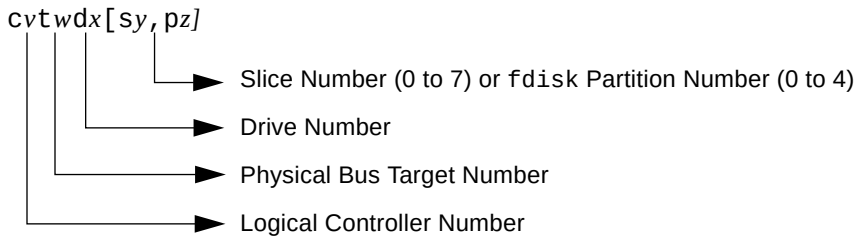


Figure 5-2 Naming Convention for Disks with SCSI Controllers

If you have only one controller on your system, *v* will always be 0.

For SCSI controllers, *w* is the target address as set by the switch on the back of the unit, and *x* is the logical unit number (LUN) of the drive attached to the target. If the disk has an embedded controller, *x* is usually 0.

To indicate the entire Solaris `fdisk` partition, specify slice 2 (`s2`).

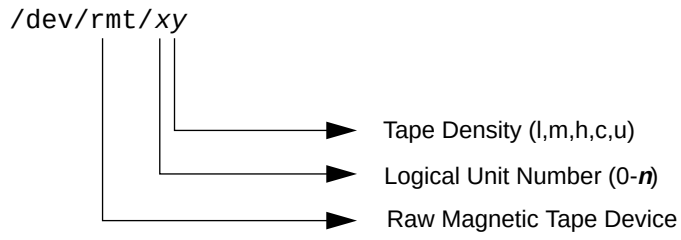
---

x86 – Controller numbers are assigned automatically at system initialization. The numbers are strictly logical and imply no direct mapping to physical controllers.

---

x86 *Accessing Tape Devices*

Logical tape device files are found in the `/dev/rmt` directory as symbolic links from the `/devices` directory.



The first tape device connected to the system is 0 (`/dev/rmt/0`), which may be one of the following types: QIC-11, QIC-24, QIC-150, or Exabyte.

See the *File System Administration* guide for more information about accessing tape devices.

x86 *Identifying Disk Devices on Your System*

If you can't identify the type of disks connected to a system based on the order in which they were added to the system, use the following commands to discover the disk types.

Use the `ls -l` command to associate a logical device name with its physical device name:

```
# ls -l /dev/rdsk/c0t0d0s0
lrwxrwxrwx   1 root     root          44 Feb  2 18:08
/dev/rdsk/c0t0d0s0 ->
../../devices/eisa/dpt@5c88,0/cmdk@0,0:a,raw
```

In the above example, disk `0`, target `0` (`cmdk@0,0`) is connected to the first DPT host adapter (`dpt@5...`), which is connected to the EISA device (`eisa`).

Use the `format` utility output to identify the disks that are recognized on the system. The `format` output displays a disk's logical and physical device name along with information about the disk's cylinders, heads, and sectors.

```
# format
Searching for disks...done
AVAILABLE DISK SELECTIONS:
       0. c0t0d0 <DEFAULT cyl 507 alt 2 hd 64 sec 32>
          /eisa/dpt@5c88,0/cmdk@0,0
       1. c0t3d0 <DEFAULT cyl 1852 alt 2 hd 15 sec 74>
          /eisa/dpt@5c88,0/cmdk@3,0
Specify disk (enter its number):
```

Use the `dmesg` command to associate the physical and logical device name to the disk's marketing name which appears in brackets **<>**. The size of the disk might also be displayed in this output after the Product identifier.

In the example output below, match the *Diskn* identifier with the target number in the logical device name (*cwt**x**dysz*) displayed in the `format` example above.

```
# dmesg | grep Disk
Disk0:<Vendor 'MAXTOR  ' Product 'LXT-535S        '>
Disk3:<Vendor 'SEAGATE ' Product 'ST11200N SUN1.05'>
```

Booting an x86 System 6

This chapter describes different boot scenarios and procedures for determining how to boot an x86 system.

If you are already familiar with boot subsystems and how to use them, use the following table to proceed directly to the task-oriented sections.

6

Overview

An x86 system has boot subsystems used for booting and controlling the system before the kernel is loaded. During the boot process, boot subsystem menus display different device and booting options. If the system receives no response after several time-out periods, it continues to boot automatically using default selections. You can stop the boot process when the boot subsystem menus are displayed or let it continue automatically.

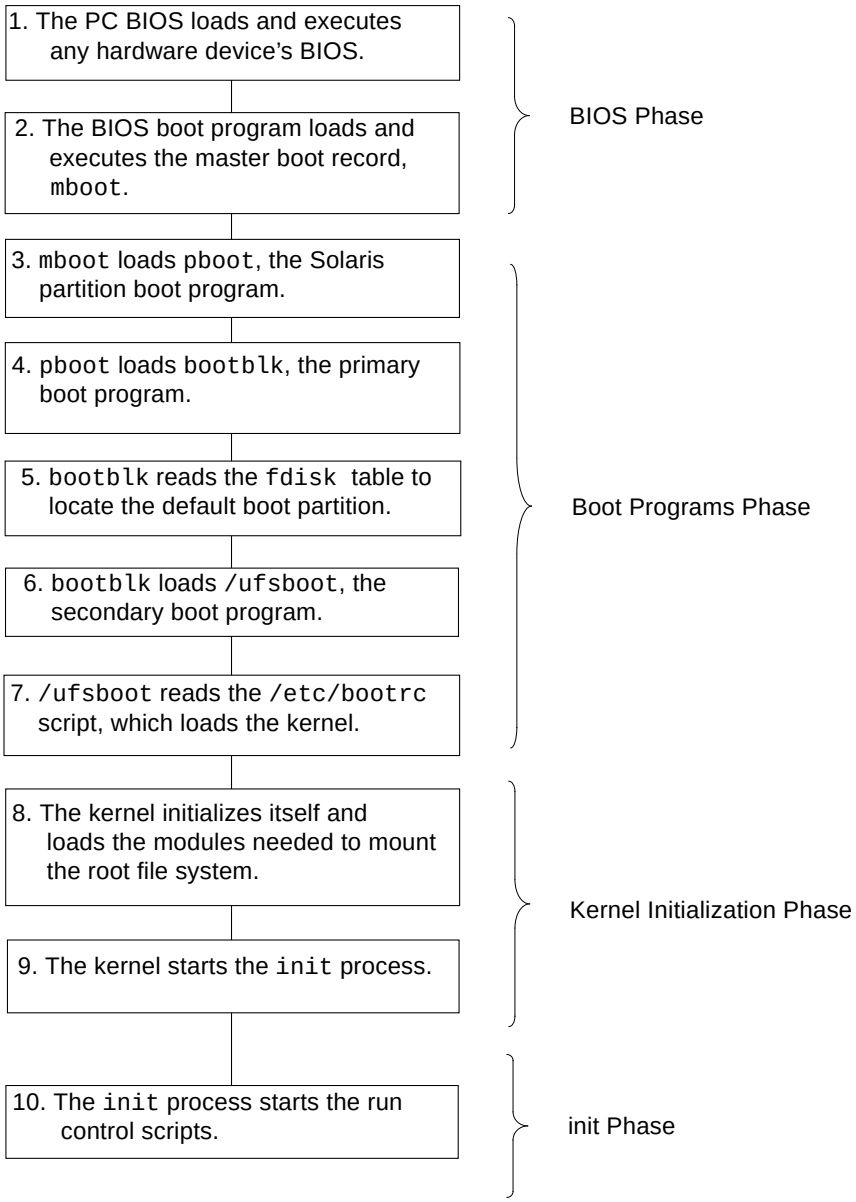
x86 *The PC BIOS*

Before the kernel is started, the system is controlled by the read-only-memory (ROM) Basic Input/Output System (BIOS), the firmware interface on a PC.

Hardware adapters can have an onboard BIOS that displays the physical characteristics of the device and can be used to access the device.

During the startup sequence, the PC BIOS checks for the presence of any adapter BIOS and if found, loads and executes each one. Each individual adapter's BIOS runs self-test diagnostics and displays device information.

x86 *The x86 Boot Process*



6

x86 The x86 Boot Process Details

The following boot process information describes the illustration on the previous page.

x86 BIOS Phase

1. When the system is powered on, the PC BIOS runs self-test diagnostics to verify the system's hardware and memory. The system begins to boot automatically if no errors are found. If errors are found, error messages are displayed describing recovery options.

   Additional hardware devices' BIOS are run at this time.

2. The BIOS boot program tries to read the first physical sector from the boot device—either a diskette or hard drive. This first disk sector on the boot device contains the master boot record (`mboot`), which is loaded and executed. If no `mboot` file is found, an error message is displayed.

x86 Boot Programs Phase

3. `mboot`, which contains disk information needed to find the active partition and the location of the Solaris boot program, `pboot`, loads and executes `pboot`.

4. `pboot` loads `bootblk`, the primary boot program, whose purpose is to load the secondary boot program located in the `ufs` file system.

5. If there is more than one bootable partition, `bootblk` reads the `fdisk` table to locate the default boot partition, and builds a menu of available partitions that is displayed to the operator. This step only occurs if there is more than one bootable partition present on the system.

6. `bootblk` finds and executes `ufsboot`, the primary boot program in the root file system slice.

7. `ufsboot` starts a command interpreter that executes the `/etc/bootrc` script, which provides a menu of choices for booting the system. The default action is to load and execute the kernel, `/kernel/unix`.

x86 *Kernel Initialization Phase*

8. The kernel initializes itself and begins loading modules, using `ufsboot` to read the files. When the kernel has loaded enough modules to mount the root file system, it unmaps the `/ufsboot` program and continues, using its own resources.
9. The kernel creates a user process and starts the `/sbin/init` process, which starts other processes by reading the `/etc/inittab` file.

x86 `init` *Phase*

10. The `/sbin/init` process starts the run control (`rc`) scripts, which execute a series of other scripts. These scripts (`sbin/rc*`) check and mount file systems, start various processes, and perform system maintenance tasks.

6

x86 *Boot Scenarios*

Table 6-1 provides different boot scenarios for an x86 system.

Table 6-1 Boot Scenarios

Boot Type	Boot Command	This Type Of Boot Is Needed When...
Booting to multiuser state	**b**	Turning off system power due to anticipated power outage. Changing kernel configuration information by modifying the `/etc/system` file.
Booting to single-user state	**b -s**	Performing file system maintenance, such as performing a backup or restoring system data. Repairing a system file such as `/etc/passwd`.
Booting interactively	**b -a**	Repairing a system configuration file such as `/etc/system`.
Booting after adding new hardware	See example on page 62	Adding hardware to the system. (Also known as a *reconfiguration boot*.)
Booting from a local CD drive	See example on page 50	Installing a new release of the operating system (this includes upgrading to a new operating system release). Repairing a critical system file which is causing system boot failure.
Booting from the network	See example on page 50	Installing a new release of the operating system from an installation server (including upgrades). Repairing a system file when a local CD-ROM player is not available.
Booting `kadb`	**b kadb**	Booting the kernel debugger (`kadb`) to track down a system problem.
Forcing a crash dump and rebooting	See example on page 52	Recovering from a hung system and you want to force a crash dump.

x86 – Booting brings the system to run level 3, multiuser level with resources shared, unless specified otherwise.

x86 *Boot Subsystems*

Booting an x86 system uses these subsystems:

- Primary boot subsystem
- Secondary boot subsystem

Additionally, booting from the network or a local CD-ROM drive uses this subsystem:

- Solaris boot diskette

The following section describes each of these boot subsystems and provides examples of each subsystem screen. Screen displays will vary based on system configurations.

x86 *Solaris Boot Diskette*

This diskette is also known as the multiple device boot (MDB) diskette.

```
Solaris 2.4 for x86                      Multiple Device Boot, vsn 2.1

                 Solaris/x86 Multiple Device Boot Menu

             Code   Device    Vendor     Model/Desc           Rev
             ======================================================

             10     DISK      MAXTOR     LXT-535S             8.75
             11     CD        SONY       CD-ROM CDV-8012      3.1d
             12     NET       SMC/WD     I/O=300 IRQ=5

             Enter the boot device code:

30
```

The Solaris boot diskette provides a menu of bootable devices such as disk, network, or CD-ROM. (The system probes currently connected devices and displays the devices in the MDB menu.)

6

x86 *Primary Boot Subsystem*

The primary boot subsystem menu displays a list of partitions to boot from if there is more than one bootable partition.

```
SunOS 5.4 for x86                        Primary Boot Subsystem, vsn 2.0

      Loading master boot record from specified device....

                  Current Disk Partition Information

            Part#    Status    Type         Start         Length
            ================================================================
              1               DOS16           32          61408
              2      Active    SOLARIS      61440         983040
              3
              4

            Please select the partition you wish to boot: 2
30
```

The default partition is automatically selected if you press Return or do not select an alternate partition from which to boot (after a 30-second time out).

x86 *Secondary Boot Subsystem*

The second boot subsystem menu displays available boot options.

```
Solaris 2.4 for x86                  Secondary Boot Subsystem, vsn 2.11

                     <<< Current Boot Parameters >>>
 Boot path: /eisa/dpt@5c88,0/cmdk@0,0:a
 Boot args: /kernel/unix

 Type    b [file-name] [boot-flags] <ENTER>   to boot with options
 or      i <ENTER>                            to enter boot interpreter
 or      <ENTER>                              to boot with default

                       <<< timeout in 5 seconds >>>

 Select (b)oot or (i)nterpreter: b
```

The system automatically boots to run level 3 if you don't select an option (after a five-second time out) from this menu. The other options enable you to specify boot options or enter the boot interpreter (see `boot(1M)`).

x86 *Booting a System*

The boot procedures provided in this section assume that the system has been shut down using the Solaris 2.x commands and is waiting at the `type any key to reboot` prompt or has been shut down by pressing the reset button.

x86 *Interrupting Automatic Time Outs*

You can prevent the system from automatically moving to the next boot subsystem menu (using the time out values) by typing a value on the boot subsystem menu without pressing Return.

To change the automatic time out value for the secondary boot subsystem, edit the `set boot_timeout` value in the `/etc/bootrc` file.

x86 – The following procedures use the reset button to restart the system. If your system does not have a reset button, use the on/off switch to restart the system. You might be able to press the Control-Alt-Del keys to interrupt system operation depending upon the state of the system.

x86 How to Boot to Multiuser State (Run Level 3)

Booting to multiuser state is usually done after halting the system or performing some system hardware maintenance task. This is the default boot level where all resources are available and users can log into the system.

1. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shut down.**
   The Primary Boot Subsystem menu is displayed after a few minutes.
2. **Select the Active Solaris partition (Part #2 in the example on page 46) as the boot device from the Primary Boot Subsystem menu. Press Return.**
   If you do not make a selection within 30 seconds, the default boot partition is selected automatically. The Secondary Boot Subsystem menu is displayed.
3. **Type** `b` **to boot the Solaris 2.x environment to run level 3. Press Return.**
   If you do not make a selection within 5 seconds, the system is automatically booted to run level 3.

x86 How to Boot to Single-User State (Run Level S)

Booting to single-user state is usually done prior to performing a system maintenance task such as backing up the system. At this level only some file systems are mounted and users cannot log into the system.

1. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**
   The Primary Boot Subsystem menu is displayed after a few minutes.
2. **Select the Active Solaris partition (Part #2 in the example on page 46) as the boot device from the Primary Boot Subsystem menu. Press Return.**
   If you do not make a selection within 30 seconds, the default boot partition is selected automatically.

The Secondary Boot Subsystem menu is displayed.

3. **Type** `b -s` **to boot the Solaris 2.x environment to run level s. Press Return.**
   If you do not make a selection within 5 seconds, the system is automatically booted to run level 3.
4. **Type the root password, if prompted.**
5. **Perform the maintenance task that needed the run level change to s.**
6. **Press** `Control-d` **to bring the system back to run level 3.**

x86 How to Boot Interactively

You may want to boot interactively to make a temporary change to the system file or the kernel. Booting interactively lets you test your changes and recover easily if you have any problems.

1. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**
   The Primary Boot Subsystem menu is displayed after a few minutes.
2. **Select the Active Solaris partition as the boot device from the Primary Boot Subsystem menu. Press Return.**
   If you do not make a selection within 30 seconds, the default boot partition is selected automatically.

   The Secondary Boot Subsystem menu is displayed.
3. **Type** `b -a` **at the** `Select (b)oot or (i)nterpreter:` **prompt. Press Return.**
4. **Press Return to use the default kernel (**`/kernel/unix`**) as prompted, or type the name of the kernel to use for booting and press Return.**
5. **Press Return to use the default** `/etc/system` **file as prompted, or type the name of an alternate system file and press Return.**
6. **Press Return to use the default modules directory path as prompted, or type alternate path for the modules directory and press Return.**

7. **Press Return to use the default root file system type as prompted:** `ufs` **for local disk booting or** `nfs` **for diskless clients.**

8. **Press Return to use the default physical name of the root device as prompted, or type the device name.**

x86 *Example: How to Boot Interactively*

In this example, the default choices (shown in square brackets `[]`) are accepted by pressing Return:

```
Select (b)oot or (i)nterpreter: b -a
(Copyright notice)
Enter filename [/kernel/unix]:
Name of system file [/etc/system]:
Name of default directory for modules [/kernel /usr/kernel]:
root filesystem type [ufs]
Enter physical name of root device
[/eisa/dpt@5c88,0/cmdk@0,0:a]:
Configuring network interfaces:  smc0
Hostname: venus
(fsck messages)
The system is coming up.  Please wait
(More messages)
venus login:
```

x86 How to Boot from the Local CD-ROM Drive or the Network

This procedure is used to recover from an invalid system file that is preventing the system from booting successfully. This situation requires booting from a local CD-ROM drive or the network to recover. Recovering from a invalid `/etc/passwd` file is used as an example.

Substitute the device name of the file system to be repaired for the *devicename* variable identified in the procedures below. Refer to Chapter 5, "Accessing Devices on an x86 System," if you need help identifying your device names.

1. **Boot from the Solaris 2.x installation CD (or the network) to single-user mode using steps a-f.**
   If you are booting from the network, skip steps a and b.

   a. **Insert the Solaris 2.x installation CD into the CD caddy.**

b. **Insert the CD caddy into the CD-ROM drive.**

c. **Insert the Solaris boot diskette into the primary diskette drive (DOS drive A).**

d. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**

The Multiple Device Boot Subsystem menu is displayed after a few minutes.

e. **Select the CD-ROM drive or net(work) as the boot device from the Multiple Device Boot menu.**
The Secondary Boot Subsystem menu is displayed.

f. **Type** `b -s` **at the** `Select the type of installation:` **prompt.**

After a few minutes, the single-user mode # prompt is displayed.

2. **Mount the root (/) file system that has the invalid** `passwd` **file.**

```
# mount /dev/dsk/devicename /a
```

3. **Change to the newly mounted** `etc` **directory.**

```
# cd /a/etc
```

4. **Set the terminal type.**

```
# TERM=AT386
# export TERM
```

5. **Make the necessary change to the** `/etc/passwd` **file using an editor.**

6. **Change to the** `root` **directory.**

7. **Unmount the** `/a` **directory.**

```
# umount /a
```

8. **Reboot the system.**

```
# init 6
```

x86 How to Abort a Booting Process

Occasionally, you may need to abort the booting process. The specific abort key sequence depends on your system type. For example, press the reset button to abort the system. If your system doesn't have a reset button, turn the power off and back on again.

x86 How to Force a Crash Dump and Reboot the System

Use the following procedure if your system hangs and you want to force a crash dump. This procedure assumes that you have enabled the savecore feature and you have booted kadb. See *Common Administration Tasks* for information on the savecore feature.

1. **Press Control-Alt-d.**
   The kadb> prompt is displayed.

```
kadb>
```

2. **Type** `0:c` **at the** kadb> **prompt.**

```
kadb> 0:c
```

3. **Type** `:c` **at the** kadb> **prompt.**

```
kadb> :c
```

   After the crash dump is written to disk, the system will continue to reboot.

x86 *Displaying Boot Information*

The `dmesg` command displays information that is provided during the boot process. The boot information displayed as messages on the system console provide helpful information for troubleshooting system boot problems.

This information is stored in the `/var/adm/messages` file.

An example of `dmesg` output is displayed below.

Boot date and time
SunOS release information
Copyright
Physical memory
Available memory
System type
/kernel/unix probes devices

Ethernet address

```
# dmesg
May 31 10:30
SunOS Release 5.4 Version [UNIX(R) System V Release 4.0]
Copyright (c) 1983-1994, Sun Microsystems, Inc.
mem = 15992K (0xf9e000)
avail mem = 12070912
root nexus = i86pc
eisa0 at root
EISA-device: dpt6
Disk0:<Vendor 'MAXTOR  ' Product 'LXT-535S        '>
cmdk0 at dpt6 target 0 lun 0
cmdk0 is /eisa/dpt@5c88,0/cmdk@0,0
Disk6:<Vendor 'SONY    ' Product 'CD-ROM CDU-8012 '>
cmdk6 at dpt6 target 6 lun 0
cmdk6 is /eisa/dpt@5c88,0/cmdk@6,0
EISA-device: asy0
asy0 is /eisa/asy@3f8,0
Ethernet address = 0:0:c0:68:14:5d
SMC WD8003/WD8013 driver: type=WD8013W   addr=00 00 c0 68 14 5d
EISA-device: smc0
smc0 is /eisa/smc@0,c0000
dump on /dev/dsk/c0t0d0s1 size 32756K
NOTICE: GIO_KEYMAP type 0
NOTICE: PIO_KEYMAP type 0
NOTICE: INSTALLING new map of type USL FORMAT
NOTICE: IN i8042_acquire
NOTICE: out i8042_acquire
NOTICE: rv was 1
NOTICE: IN i8042_release
NOTICE: about to enable keyboard
NOTICE: out i8042_release
Nov 30 17:19:31 sendmail[171]: network daemon starting
```

6

Setting Up Disks on Your x86 System 7

This chapter provides conceptual information and step-by-step procedures for setting up disk drives attached to your x86 system running the Solaris 2.x environment.

If you are already familiar with disk preparation steps and dividing a disk into slices, use the following table to proceed directly to the task-oriented sections. Otherwise, continue to the next page.

If you are not familiar with hard disk concepts or terminology, how and why disk slices are set up, or how file systems are arranged for different system configurations, read *Peripherals Administration* before attempting any tasks.

7

x86 Disk Preparation Steps

Before you can use a disk to store and retrieve data, it must be formatted. Formatting can involve four separate processes:

- *Low-level Formatting* – Writing format information to the disk
- *Creating a fdisk partition* – Choosing an area of the disk for Solaris
- *Partitioning* – Dividing the Solaris disk partition into manageable areas
- *Surface analysis* – Compiling an up-to-date list of disk defects

Disks purchased from Sun are already formatted and provide a manufacturer's defect list that is loaded automatically. Using the `suninstall` utility to install Solaris 2.x on an x86 system will create the `fdisk` partition and slices needed for system operation.

After installation, you may need to repartition a disk. See *Peripherals Administration* for information on repartitioning a disk.

x86 The Formatting Utilities

The primary method of formatting disks in the Solaris 2.x environment is the `format` utility. This name is something of a misnomer, however, because `format` enables you to analyze, partition, and label disks, as well as format them. Its repair option can even help you recover from file system difficulties brought on by a defective disk.

See *Peripherals Administration* for reference information on the `format` utility.

Note – Solaris 2.1 for x86 users, the `fdisk`(`1M`), `fmthard`(`1M`), `diskscan`(`1M`), and `addbadsec`(`1M`) commands are still available as a command line interface for setting up disks. See Appendix B, "x86: Additional Disk Commands," for instructions on using these commands to set up a disk.

---

x86 About `fdisk` Partitions

A *disk partition* is a portion of the disk reserved for use by an operating system. Different operating systems, for example, MS-DOS® and Solaris 2.x, can reside on the same disk in separate partitions. These partitions are sometimes referred to as `fdisk` partitions, because `fdisk` is the name of the program that is used to create them. During installation of the Solaris 2.x software on an x86 system,

a Solaris partition is created on the primary drive and is made active. Only one `fdisk` partition on a disk may be *active* at a time. The active partition is the one whose operating system will be booted by default at system start-up.

You determine the type and size of the `fdisk` partitions and which one is to be active. You may decide, for example, to make a Solaris partition the entire disk, or you may want to make it smaller to allow room for a DOS partition. You can also make a new `fdisk` partition on a disk without disturbing existing partitions (if there is enough room to create a new one).

---

x86 – Some Solaris systems do not support multiple operating systems on the same hard disk. Because of this restriction, some Solaris documentation and programs may refer to a *slice* as a *partition*. To avoid confusion, Solaris 2.x documentation tries to distinguish between `fdisk` partitions (which are supported only on Solaris for x86) and the divisions within the Solaris `fdisk` partition, which may be called slices or partitions.

---

x86 *Special Areas of the Disk*

The first sector of the disk is reserved for the *master boot record*. The master boot record contains the `fdisk` partition table.

The beginning of a Solaris partition is reserved for various boot programs and the disk label. Included in the disk label is the volume table of contents (VTOC). The VTOC describes how disk slices are arranged on the disk.

x86 *Dividing a Disk into Slices*

When you set up a disk's slices using the `format` utility, you choose not only the size of each `fdisk` partition, but also which `fdisk` partition will be active. These choices depend on how you intend to use the system to which the disk is attached.

x86 *The* `fdisk` *Partitions on a Disk*

The disk can be divided into a maximum of four `fdisk` partitions. One of these partitions must be a Solaris partition, and it must be made the active partition on the disk.

7

Use the DOS `format` command to initialize the DOS partition.

The `fdisk` command can be used to create a partition for DOS also, but DOS commands must be used to complete the creation of DOS partitions.

x86 *Locating* `fdisk` *Partitions on a Disk*

There are two rules to remember when deciding how to partition the disk.

1. Solaris `fdisk` partitions must begin on cylinder boundaries.
2. Solaris `fdisk` partitions must begin at cylinder 1, not cylinder 0, on the first disk because additional boot information, including the master boot record, is written in sector 0. To avoid overwriting the master boot record, begin a Solaris `fdisk` partition at cylinder 1.

x86 *Customary Slices*

Solaris `fdisk` partitions can be divided up into a maximum of ten slices. These slices are numbered 0 through 9, although 8 and 9 are not configurable by administrators. The slice number does not necessarily match the order in which the slice appears on the Solaris `fdisk` partition.

Table 7-1 on the next page summarizes the customary contents of the disk slices.

Table 7-1 Customery Disk Slice Assignments

Slice	File System	Purpose
0	`root`	Holds the files and directories that make up the operating system.
1	swap	Provides virtual memory, or *swap space*. Swap space is used when a new program you need to run is too large to fit in a computer's memory at the same time as other programs that are already running. When this happens, the operating system "swaps" different programs from the computer's memory to the disk—and vice versa—as needed.
2	—	Used by the operating system to reference the entire disk. It is defined automatically by Sun's `format` and installation programs and should not be altered.
3	`/export`	Holds alternative versions of the operating system. These alternative versions are required by client machines whose architectures differ from that of the server.
4	`/export/swap`	Provides virtual memory space for the client rather than for the server.
5	`/opt`	Holds application software that is added to a system. If there is not enough room on the disk to put the `/opt` file system in slice 5, the `/opt` directory is put in slice 0.
6	`/usr`	Holds operating system commands—also known as *executables*— that are run by users. This slice also holds documentation, system programs (`init` and `syslogd`, for example) and library routines.
7	`/export/home`	Holds files that are created by users.
8	—	Contains information necessary for Solaris to boot from the hard disk. It resides at the beginning of the Solaris partition (although the slice number itself does not indicate this), and is known as the boot slice.
9	—	Area reserved for alternate disk blocks, and is known as the alternate sector slice.

The Solaris installation program provides slice size recommendations based on the software you select for installation.

7

x86 Instructions for Setting Up Disks

This section includes instructions for performing tasks related to disks.

You will find an example of the screen input and output associated with the task with each set of instructions. Your screen output will vary based on your system configuration.

---

x86 – You must be superuser to perform the following procedures.

---

x86 Adding a Secondary Disk

To add a secondary disk, you may need to follow some or all of these procedures:

- Performing a reconfiguration boot
- Extracting a defect list
- Formatting a disk
- Dividing a disk into slices and labelling the disk

Performing a reconfiguration boot is covered on the following pages. This step is necessary when adding new hardware to the system so the appropriate device files are linked in the `/dev` directory.

The other procedures listed above are covered in *Peripherals Administration*.

x86 Adding a System Disk

If your system disk fails and the `root` file system is damaged and you do not want to reinstall the Solaris 2.x environment, you can restore your system disk using the following steps:

- Physically connecting the new disk drive
- Booting the Solaris 2.x installation media from CD-ROM or the network to single-user mode
- Creating the Solaris `fdisk` partition and dividing the disk into slices
- Creating the boot blocks on the Solaris `fdisk` partition
- Restoring the `root` (`/`) and `/usr` file systems

This procedure is covered on the pages to follow.

x86 *Maintaining Disk Drives*

The `format` utility can be used to perform disk diagnostics on SCSI or IDE disks, if needed.

An example of using the `format` utility is at the end of this chapter in the section called “How to Remap a Bad Block on an IDE Disk Drive.”

Examples of using the `format` utility to perform disk diagnostics on SCSI disks can be found in *Peripherals Administration*.

x86 *Installing Boot Blocks on a Solaris* `fdisk` *Partition*

If you want to make a disk bootable, you must install the boot blocks. Ordinarily, you do not need to install boot blocks. You may need to use this procedure if the root file system becomes damaged because you cannot restore the boot block from tape.

x86 How to Install a Boot Block on a Solaris `fdisk` Partition

1. **Identify the disk you want to use as a bootable disk.**
2. **Type the following command:**

```
# /usr/sbin/installboot /usr/lib/fs/ufs/pboot
/usr/lib/fs/ufs/bootblk /dev/rdsk/cn[tn]dns2
```

The contents of the `pboot` file (the partition boot block) are installed in the first sector of the Solaris slice. The contents of the `bootblk` file (the primary boot program) are installed after the disk label in the Solaris slice.

x86 How to Perform a Reconfiguration Boot

1. **Create a `/reconfigure` file that will be read when the system is booting.**

```
# touch /reconfigure
```

   The `/reconfigure` file will cause the SunOS software to check for the presence of any newly installed peripheral devices when you power on or boot your system later.

2. **Shut down the system.**

```
# /usr/sbin/shutdown -y -g30 -i0
```

x86 – The 0 in `i0` is a zero.

   In the example above, the command sends a message to all users who are logged in stating they have 30 seconds (`-g30`) before the system begins to shut down. The `type any key to continue` prompt is displayed after the system is shut down.

3. **Turn off power to the system after the `type any key to continue` prompt is displayed.**
   Refer to the hardware installation guide that accompanies your system for the location of the power switch.

4. **Turn off power to all external peripheral devices.**
   For location of power switches on any peripheral devices, refer to the hardware installation guides that accompany your peripheral devices.

5. **Install the peripheral device.**
   Refer to the hardware installation guides that accompany the peripheral devices for information on how to install and connect those devices.

6. **Turn on the power to all external peripherals.**

7. **Turn on the power to the system.**
   The system will boot and you will be shown the login prompt.

x86 How to Add a System Disk

The following procedure assumes your system disk has become disabled and you need to replace it with a new one. The best way to solve this problem without reinstalling the Solaris environment is to restore your file systems from a backup medium. See *File System Administration* for information about restoring file systems to disk.

---

x86 – This procedure displays screen output as an example and will vary from system to system.

---

1. **Connect the disk to the system and check the physical connections.**
   Make sure that any cables connecting the disk unit to the system are securely attached. See *x86 Device Configuration Guide* about hardware configuration requirements that should be checked before installation.

2. **Boot the Solaris 2.4 media from the CD-ROM drive or the network.**

   a. **Insert the Solaris boot diskette into the diskette drive.**

   b. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shutdown.**
   The Multiple Device Boot Subsystem menu is displayed after a few minutes.

   c. **Select the CD-ROM drive or net(work) as the boot device from the Multiple Device Boot menu.**
   The Secondary Boot Subsystem Menu is displayed.

   d. **Type** `b -s` **at the** `Select the type of installation:` **prompt.**

   After a few minutes, the single-user mode # prompt is displayed.

3. **Enter the format utility by typing** `format` **at the root prompt and pressing Return.**
   The `format` utility is used to create the Solaris `fdisk` partition and assign slices to the root and swap areas.

   a. **Enter the number of the disk that you want to partition from the list displayed on your screen.**

b. Enter the fdisk menu by typing `fdisk` at the `format>` prompt and select the interactive option. The following message is displayed.

```
The recommended default partitioning for your disk is:

  a 100% “SOLARIS System” partition.

To select this, please type “y”.  To partition your disk
differently, type “n” and the “fdisk” program will let you
select other partitions. y

fdisk> q
```

c. Type `y`(es) to create and activate a Solaris `fdisk` partition spanning the entire disk.

d. Exit the fdisk menu by typing `4` (Update disk configuration and exit) at the `Selection:` prompt. Type `q` (quit) at the `fdisk>` prompt.

---

x86 – In this next section, the `format` utility refers to a slice as a partition. The *partition menu* is used to divide a disk into slices. A *partition table* refers to a table of slice information.

---

e. **Enter the partition menu (which lets you set up the slices) by typing** `partition` **at the** `format>` **prompt.**

```
format> partition
PARTITION MENU:
        0      - change ‘0’ partition
        1      - change ‘1’ partition
        2      - change ‘2’ partition
        3      - change ‘3’ partition
        4      - change ‘4’ partition
        5      - change ‘5’ partition
        6      - change ‘6’ partition
        7      - change ‘7’ partition
        8      - change ‘8’ partition
        9      - change ‘9’ partition
        select - select a predefined table
        modify - modify a predefined partition table
        name   - name the current table
        print  - display the current table
        label  - write partition map and label to the disk
        quit
partition>
```

f. **Display the current partition (slice) table by typing** `print` **at the** `partition>` **prompt.**

g. **Modify the current table by typing** `modify` **at the** `partition>` **prompt. Type** `1` **at the** `Choose base` **prompt to set the Free Hog.**

The following information is displayed.

```
Part      Tag    Flag     Cylinders        Size       Blocks
  0 unassigned    wm       0               0         (0/0/0)
  1 unassigned    wm       0               0         (0/0/0)
  2     backup    wm       0 - 1463      411.75MB    (1464/0/0)
  3 unassigned    wm       0               0         (0/0/0)
  4 unassigned    wm       0               0         (0/0/0)
  5 unassigned    wm       0               0         (0/0/0)
  6 unassigned    wm       0               0         (0/0/0)
  7 unassigned    wm       0               0         (0/0/0)
  8       boot    wu       0                 .24MB    (1/0/0)
  9 alternates    wm       1-2               .48MB    (2/0/0)
Do you wish to continue creating a new partition table based on
above table[yes]?
```

h. Create a new partition table by typing `yes` **at the above prompt.**

i. Identify the Free Hog partition (slice) and the sizes of the root and swap slice when prompted.

```
Free Hog partition[6]?
Enter size of partition ‘0’ [0b, 0c, 0.00mb]: 100mb
Enter size of partition ‘1’ [0b, 0c, 0.00mb]: 100mb
Enter size of partition ‘3’ [0b, 0c, 0.00mb]:
Enter size of partition ‘4’ [0b, 0c, 0.00mb]:
Enter size of partition ‘5’ [0b, 0c, 0.00mb]:
Enter size of partition ‘7’ [0b, 0c, 0.00mb]: 200mb
Enter size of partition ‘9’ [0b, 0c, 0.00mb]: 2c
```

j. Answer yes to the `Okay to make this the current partition table?` **prompt.**

k. Name the partition table using quotes.

```
Enter table name (remember quotes): "disk0"
```

l. **Label the disk with the new partition table when you have finished allocating slices on the new disk.**

```
partition> label
Ready to label disk, continue? yes
```

m. **Exit the partition menu by typing** `quit` **at the** `partition>` **prompt.**

n. **Exit the format menu by typing** `quit` **at the** `format>` **prompt.**

4. **Create the file systems for the newly partitioned disk using the** `newfs` **command.**

```
# newfs /dev/rdsk/devicename
```

5. **Run the** `installboot` **command to install the boot blocks.**

```
# /usr/sbin/installboot /usr/lib/fs/ufs/pboot
/usr/lib/fs/ufs/bootblk /dev/rdsk/cw[tx]dys2
```

The root slice must have a boot block. See "How to Install a Boot Block on a Solaris fdisk Partition" on page 61 for more information on `installboot`.

See Chapter 8, "Managing File Systems on an x86 System," for step-by-step instructions on restoring the root (`/`) or `/usr` file systems.

x86 How to Remap a Bad Block on an IDE Disk Drive

IDE disk drives do not automatically map out bad blocks like other drives supported by Solaris software. Before installing Solaris on an IDE disk, you may want to perform a surface analysis on the disk.

1. **Use the** `fdisk` **program to create a Solaris partition on the disk, if necessary. (If a Solaris** `fdisk` **partition already exists, leave it alone.)**

2. **After you create a Solaris** `fdisk` **partition, you can use the** `format` **program to map out bad blocks. To start the** `format` **program, type** `format`**.**

3. **Specify the IDE disk drive on which you want to perform a surface analysis.**

---

x86 – IDE drives do not include a target number. The IDE drive naming convention is c*n*d*n*, where c*n* is the controller number and d*n* is the device number.

---

4. **At the** `format>` **prompt, type** `analyze`**.**
5. **At the** `analyze>` **prompt, type** `config`**. This will show you the current settings for a surface analysis. If you want to change any settings, type** `setup`**.**
6. **At the** `analyze>` **prompt, type** `read`**,** `write`**, or** `compare` **for the type of surface analysis to be performed. If** `format` **finds bad blocks, it will remap them.**
7. **At the** `analyze>` **prompt, type** `quit`**.**
8. **You may want to specify blocks to remap. If so, at the** `format>` **prompt, type** `repair`**.**
9. **Type** `quit` **to quit the** `format` **program.**

Managing File Systems on an x86 System 8

This chapter describes how to format diskettes for use with `ufs`, `pcfs`, and DOS file systems. Restoring `ufs` file systems is also described.

Use this table to proceed directly to the task-oriented sections.

8

x86 Additional x86 Disk-Based File Systems

x86 systems provides support for the following file system types in addition to the file systems listed in *File System Administration*.

- `pcfs` - The Solaris 2.x `pcfs` file system supports Primary and Extended DOS partitions on hard disks in addition to the diskette-based DOS file systems.
- `s5fs` - System V file system which supports pre-existing Intel® System V file systems with 512, 1024, or 2048 byte blocks.

x86 Formatting Diskette Devices

The `fdformat` command is used to format diskette devices. By default, the `fdformat` command formats the diskette to the configured capacity of the diskette drive.

Table 8-1 identifies the diskette drives available on x86 systems and the `fdformat` option used to achieve the resulting formatted capacities.

Table 8-1 Diskette Drive Information

A drive type and capacity of	**using a diskette type of**	**and these `fdformat` options**	**will result in a formatted capacity of...**
5.25 (360 Kbytes)	Double (DD)	Default (D)	360 Kbytes
5.25 (1.2 Mbytes)	High (HD)	Default (H)	1.2 Mbytes
	Double (DD)	D	360 Kbytes
3.5 (720 Kbytes)	Double (DD)	Default (D)	720 Kbytes
3.5 (1.44 Mbytes)	High (HD)	Default (H)	1.44 Mbytes
	Double (DD)	D	720 Kbytes
3.5 (2.88 Mbytes)	Extended (ED)	Default (E)	2.88 Mbytes
	High (HD)	H	1.44 Mbytes
	Double (DD)	D, L	720 Kbytes

See the `fdformat(1)` manual page for additional information on formatting diskettes. The following sections provide examples of using the `fdformat` command.

x86 Creating File Systems

x86 Creating a File System on a Diskette

x86 How to Format a Diskette for Use With a `ufs` File System

Use double-sided high-density 3.5-inch or 5.25-inch diskettes (diskettes are marked "*DS, HD*").

Caution – Reformatting destroys any files already on the diskette.

1. **Check the diskette to make sure that it is not write-protected.**
   A 3.5-inch diskette is write-protected if you can see through a square hole in the lower left corner of the diskette. A 5.25-inch diskette is write-protected if the notch near the upper right corner is covered. To protect 3.5-inch diskettes, turn the diskette over and push the plastic write-protect switch toward the top of the diskette.
2. **Insert a DS, HD diskette in the diskette drive.**
3. **Type** `fdformat /vol/dev/aliases/floppy`*n* **and press Return.**
   A message similar to the following is displayed:

```
Formatting 1.44 MB in /vol/dev/aliases/floppy0
Press return to start formatting floppy.
```

x86 – If there is a file system on the diskette and the diskette is mounted, you will get an error from `fdformat`. You must unmount the diskette using `umount` and run `fdformat` again or use the `fdformat` command with the `-U` option to umount any file system on the floppy and reformat it.

4. **Press Return.**
   While the diskette is being formatted, a series of dots (`. . .`) is displayed. Bad spots in a cylinder cause an "e" to be displayed. Do not use the diskette. When formatting is complete, the prompt is redisplayed.

Example

```
pluto% fdformat /vol/dev/aliases/floppy0
Formatting 1.44 MB in /vol/dev/aliases/floppy0
Press return to start formatting floppy.
.....................................................................
.................
pluto%
```

x86 How to Format a Diskette That Can Be Read on a DOS System

Caution – Reformatting destroys any files already on the diskette.

1. **Insert a DS, HD diskette in the diskette drive.**
2. **Type** `fdformat -t dos /vol/dev/aliases/floppy`*n* **and press Return.**
   The configuration and the message `Press return to start formatting floppy` are displayed.
3. **Press Return.**
   While the diskette is being formatted, a series of dots (. . .) is displayed. Bad spots in a cylinder cause an “e” to be displayed. Do not use the diskette. When formatting is complete, the prompt is redisplayed.

Example

```
pluto% fdformat -t dos /vol/dev/aliases/floppy0
Formatting 1.44 MB in /vol/dev/aliases/floppy0
Press return to start formatting floppy.
.....................................................................
.................
pluto%
```

x86 Creating a pcfs File System on a Disk Partition

You need to create `pcfs` or MS-DOS file systems only if you want to boot MS-DOS instead of Solaris on your system.

Solaris can define or allocate an `fdisk` partition for a Primary DOS and an Extended DOS partition only. The partition can be BIG-DOS (greater than 32 Mbytes in size) or have 12-bit or 16-bit FATs. The MS-DOS `fdisk` command must be used to define or allocate logical drives within the Extended DOS partition. The MS-DOS `format` command must be used to complete the file system creation on the Primary DOS partition and each logical drive.

Refer to your MS-DOS user's guide for details on using the DOS version of `fdisk` and `format`.

x86 *Mounting and Unmounting File Systems*

x86 How to Mount a `pcfs` File System From a Diskette

Use the following procedure to mount a `pcfs` file system from a diskette:

1. **Type the `volcheck` command and press Return.**
   The `pcfs` file system will be mounted automatically.

Use the following procedure to mount a `pcfs` file system from a diskette if you are not running Volume Manager.

1. **Become superuser.**
2. **Create the mount point for the file system, if necessary.**

```
# mkdir /pcfs
```

3. **Type the following `mount` command and press Return.**

```
# mount -F pcfs /dev/diskette /pcfs
```

x86 How to Mount a pcfs File System From a Hard Disk

To mount a `pcfs` file system from the hard disk, you must specify the block device name of the disk, the DOS logical drive, and the mount point for the file system.

- *devicename* – specifies the special block device file for the whole disk (for example, `/dev/dsk/c0t0d0p0`).
- *logical-drive* – specifies either the DOS logical drive letter (c through z) or a drive number 1 through 24. Drive c is equivalent to drive 1 and represents the Primary DOS partition on the drive; all other letters or numbers represent DOS logical drives within the Extended DOS partition.
- *mount-point* – specifies where the file system is mounted.

Note that the *devicename* and *logical-drive* must be separated by a colon.

Use the following procedure to mount a `pcfs` file system from a hard disk.

1. **Become superuser.**
2. **Type** `mount -F pcfs /dev/dsk/`*device-name*`:`*logical-drive mount-point* **and press Return.**

In this example, the logical drive in the Primary DOS partition is mounted on the `/pcfs/c` directory:

```
pluto% su
Password:
# mount -F pcfs /dev/dsk/c0t0d0p0:c /pcfs/c
```

In this example, the first logical drive in the Extended DOS partition on the disk is mounted on `/pcfs/d`:

```
# mount -F pcfs /dev/dsk/c0t0d0p0:2 /pcfs/d
```

x86 How to Mount an s5fs File System With Default Options

To mount an `s5fs` file system with the default options, you must specify the block device name of the partition from the `/dev/dsk` directory and the mount point for the file system.

- *device-name* – specifies the special block device file for the disk partition holding the file system (for example, `/dev/dsk/c0t3d0s7`).
- *mount-point* – specifies where the file system is mounted.

Use the following steps to mount an `s5fs` file system with default options.

1. **Become superuser.**
2. **Type** `mount -F s5fs /dev/dsk/`*devicename mount-point* **and press Return.**
   The file system is mounted.

In this example, `/dev/dsk/c0t3d0s7` is mounted on the `/files1` directory:

```
# mount -F s5fs /dev/dsk/c0t3d0s7 /files1
```

x86 How to Mount an `s5fs` File System With Additional Options

You can mount an `s5fs` file system with different mount options shown in Table 8-2. If you specify multiple options, separate them with commas (no spaces), for example, `-o rw,remount`.

Table 8-2 Additional Mount Options

Option	Description
f	Fake an entry in `/etc/mnttab`, but don't really mount any file systems.
n	Mount the file system without making an entry in `/etc/mnttab`.
nosuid	Disallow `setuid` execution. The default is to allow `setuid` execution.
rw	Read / write.
ro	Read only. If you do not specify this option, the default is read / write.
remount	Used with `rw` to remount a file system read / write.

See the `mount_s5fs(1M)` man page for a complete list of options.

Use the following step to mount an `s5fs` file system with additional options.

♦ **Type** `mount -o` *options* `/dev/dsk/`*device-name mount-point* **and press Return.**
The file system is mounted using the options you specify.

8

x86 *Restoring File Systems*

Restoring a damaged root (`/`) or `/usr` file system from a backup tape is not like other types of restore operations because the programs you need to run are in the damaged file system. You must boot the system from the installation CD-ROM and reload the file system or connect the drive to another system and reload the file system from there.

Substitute the device name of the file systems to be repaired for the *devicename* variable identified in the procedures below. Refer to Chapter 5, "Accessing Devices on an x86 System," if you need help identifying your device names.

x86 How to Restore the `root` and `/usr` File Systems

1. **Boot from the Solaris 2.x installation CD (or the network) to single-user mode using steps a-g.**
   If you are booting from the network, skip steps a and b.
   a. **Insert the Solaris 2.x installation CD into the CD caddy.**
   b. **Insert the CD caddy into the CD-ROM drive.**
   c. **Insert the Solaris boot diskette into the primary diskette drive (DOS drive A).**
   d. **Press any key to reboot the system if the system displays the** `type any key to reboot` **prompt. Or, use the reset button to restart the system if the system is shut down.**
      The Multiple Device Boot Subsystem menu is displayed after a few minutes.
   e. **Select the CD-ROM drive or net(work) as the boot device from the Multiple Device Boot menu. Press Return.**
      The Secondary Boot Subsystem menu is displayed.
   f. **Type** `b -s` **at the** `Select the type of installation:` **prompt and press Return.**
      After a few minutes, the single-user mode # prompt is displayed.
2. **Use the** `format` **utility to repair and reallocate slices on the disk.**
   a. **Locate and fix any bad blocks.**

See *Peripherals Administration* for information on using `format` to repair a disk.

b. Divide the disk into slices with the `format` command.

3. Make a new file system with the `newfs` command and check the file system with the `fsck` command for each slice except swap.

```
# newfs /dev/rdsk/devicename
# fsck /dev/rdsk/devicename
```

4. Type `mount /dev/dsk/`*devicename*`e /mnt` and press Return.
The file system is mounted on a temporary mount point.

---

x86 – To mount the file system, you specify the block device directory (`/dev/dsk`), not the raw device directory.

---

5. Type `cd /mnt` and press Return.
You have changed to the mount-point directory.

6. Type `tapes` and press Return.
This command creates the tape device entries.

7. Write-protect the tapes for safety.

8. Type `ufsrestore rvf /dev/rmt/`*unit* and press Return.
The level 0 tape is restored.

9. Remove the tape and load the next level tape in the drive.
Always restore tapes starting with 0 and continuing from lowest to highest until you reach the highest level.

10. Type `ufsrestore rvf /dev/rmt/`*unit* and press Return.
The next level tape is restored.

11. Repeat steps 8 and 10 for each additional tape.

12. Type `rm restoresymtable` and press Return.
Removes the `restoresymtable` file that is created and used by `ufsrestore` to check point the restore.

13. Type `cd /` and press Return.

14. **Type** `umount /mnt` **and press Return.**
    The root file system is unmounted.

15. **Type** `fsck /dev/rdsk/`*devicename* **and press Return.**
    The restored file system is checked for consistency.

16. **Insert a new tape in the tape drive.**

17. **Type** `ufsdump 0uf /dev/rmt/`*unit* `/dev/rdsk/`*devicename* **and press Return.**
    A level 0 backup is performed. Always do an immediate backup of a newly created file system because `ufsrestore` repositions the files and changes the inode allocation.

18. **Repeat steps 4 through 18 for the** `/usr` **file system, if necessary.**

19. **Type** `init 6` **and press Return.**
    The system is rebooted.

Summary of System Administration Differences

A

Table A-1 on the next page summarizes the differences between the SPARC and x86 system administration tasks.

A

Table A-1 SPARC and x86 System Administration Differences

Category	SPARC	x86
System operation before kernel is loaded	A programmable read-only memory (PROM) chip with a monitor program runs diagnostics and displays device information. It is also used to program default boot parameters and test the devices connected to the system.	The basic input/output system (BIOS) runs diagnostics and displays device information. A Solaris boot diskette with a program called Multiple Device Boot (MDB) is used to boot from non-default boot partitions, the network, or CD-ROM.
Booting the system	Commands and options at the PROM level.	Commands and options at the MDB, primary, and secondary boot subsystems level.

Boot programs	`bootblk` – the primary boot program, loads `ufsboot`
	`ufsboot` – the secondary boot program loads the kernel
		`mboot` – the master boot record, loads `pboot`
		`pboot` – the Solaris partition boot program, loads `bootblk`
		`bootblk` – the primary boot program, load `ufsboot`
		`ufsboot` – the secondary boot program, executes the `/etc/bootrc` script and loads the kernel

Reboot commands	The `shutdown`, `init 6`, or `reboot` commands can be used without additional operation intervention.	The `shutdown`, `init 6`, or `reboot` commands are used but requires operator intervention at the `type any key to continue` prompt.
Disk Controllers	SCSI, IPI, and Xylogics	SCSI and IDE
Disk slices and partitions	Maximum of eight slices, numbered 0-7.	Maximum of four `fdisk` partitions. The Solaris `fdisk` partition may contain up to ten slices, numbered 0-9, but only 0-7 can be used to store user data.
Diskette drives	Desktop systems usually contain one 3.5-inch diskette drive.	Systems may contain two diskette drives: a 3.5-inch and a 5.25 inch drive.

x86: Additional Disk Commands B

This appendix describes how to set up a disk connected to an x86 system using the following commands:

- `fdisk`
- `fmthard`
- `diskscan`
- `addbadsec`

B

x86 Setting Up Disks Using Additional Disk Commands

This section includes step-by-step instructions for performing tasks related to manually setting up disks using the disk command line interface.

You will find an example of the screen input and output associated with the task with each set of instructions. Your screen output will vary based on your system configuration.

x86 How to Add a Secondary Disk

The following steps are used to set up a disk using the command line interface:

- Physically connecting the disk device to the system
- Performing a reconfiguration boot
- Creating the Solaris `fdisk` partition with the `fdisk` command
- Dividing the disk into slices using the `fmthard` command
- Running surface analysis with the `diskscan` command
- Remapping bad sectors using the `addbadsec` command, if necessary

1. **Run the `fdisk` command to display the geometry of the second disk.**
   Use the `fdisk` command with the `-G` option to determine the number of sectors per cylinder according to the physical geometry of the disk:

```
# fdisk -G /dev/rdsk/c0t1d0p0
```

x86 – The raw device name in this example refers to target 1 on the SCSI bus, with device node, partition `p0`, representing the entire disk.

The following output is produced:

```
* Physical geometry for device /dev/rdsk/c0t1d0p0
* PCYL   NYCL  ACYL  BCYL  NHEAD  NSECT  SECSIZ
  1476   1464  12    0     9      64     512
```

The sectors per cylinder can be calculated by multiplying the number of heads (`NHEAD`) by the number of sectors per track (`NSECT`). In this example, `9 * 64 = 576` sectors/cylinder. Take note of the number of cylinders (`NCYL`) also, because it will be needed in future calculations.

2. **Run the `fdisk` command to create a partition table on the second disk.**
   By default, the `fdisk` command runs in interactive mode.

```
# fdisk /dev/rdsk/c0t1d0p0
```

   a. **Select option 1 to create a new partition.**

   b. **Specify 1 (a Solaris partition).**

   c. **Choose the “c” option to enter the partition in cylinders.**

   d. **Select the starting cylinder.**
      Select 1, for example, if you choose to make the Solaris partition start at the beginning of the disk. (Solaris partitions must not begin in cylinder 0.)

   e. **Select the partition size in cylinders.**
      This must be less than or equal to the `NYCL` number given in Step 1. In this example, assume that room was reserved for a DOS partition at the end of the disk and the Solaris partition was made a size of 1350 cylinders.

   f. **Enter ‘y’ to make the partition active.**

   g. **Enter 4 to write out the `fdisk` partition table.**
      This also writes out a 512-byte master boot record on the first sector on the disk.

      Based on the example, the `fdisk` partition table might now look like this:

```
          Total disk size is 1464 cylinders
          Cylinder size is 576 (512 byte) blocks
                                  Cylinders
Partition  Status  Type     Start   End   Length   %
=========  ======  ======== =====   ===   ======  ===
    1      Active  Solaris      1  1350     1350
```

3. **Run the `fmthard` command to get a copy of the default VTOC from the system disk.**
   The `fmthard` command is used to create slices out of the Solaris `fdisk` partition created in the previous step. A volume table of contents (VTOC), sometimes called a partition map, which is part of the disk label, is written on the disk near the beginning of the Solaris `fdisk` partition. To tell `fmthard` how to divide the Solaris `fdisk` partition, it is easier to:

   a. **Begin with a default partition map (like the one that exists on your system disk).**

   b. **Adjust it to suit your needs for the second disk.**

   c. **Run the `fmthard` command using the new modified partition map.**
      The `fmthard` command has a `-i` option that creates an ASCII representation of the VTOC on the disk. Since the second disk does not yet have a VTOC, request a copy of the one that is on the system disk (where Solaris is booted from).

Caution – Be sure to use the `-i` flag when specifying the device name for the primary (boot) disk. Failure to do so will cause the `fmthard` command to overwrite the VTOC on your primary disk, and you will have to reinstall Solaris.

```
# fmthard -i -n "" /dev/rdsk/c0t0d0s2 > tempfile
```

This time the device name specifies target 0 (first drive) on the SCSI bus, with device node, slice `s2`, representing the entire Solaris `fdisk` partition.

This produces output suitable to be used as a template for the `fmthard` command in the file called *tempfile.*

4. **Calculate the sizes of all slices needed for your hard disk.**
   For a secondary disk, you will need to identify slices for the alternate sectors slice (`ALTSCTR`), the slice representing the entire Solaris partition (`BACKUP`), and as many user file system slices as you want.

   In this example, assume that you want this disk to contain one `ALTSCTR` slice and one additional slice for `/export/home`. The `BACKUP` slice must also be identified.

x86 – All slices must begin on a cylinder boundary. You must determine the beginning sector and sector count for each slice.

a. Calculations for the BACKUP **slice**

Slice 2 (tag 5, type BACKUP) should begin with sector 0 and end with the last sector on the Solaris `fdisk` partition. In the case of the Seagate 1480 SCSI disk, a cylinder has 576 sectors, as calculated in Step 1. Since the choice was made to use 1350 cylinders for the entire Solaris partition, the size of the whole partition in sectors can be calculated as follows:

576 sectors / cylinder * 1350 cylinders = 777600 sectors

Because the beginning sector is 0 and the sector count is 776000, the ending sector is 777599.

b. Calculations for the ALTSCTR **slice**

In this example, locate the alternate sectors slice (tag 9) at the beginning of the Solaris `fdisk` partition. However, since you may want to make this disk a bootable disk sometime in the future, leave the first cylinder free, and begin at cylinder 1. Since a cylinder on a Seagate 1480 SCSI disk contains 576 sectors (NCYL), the last sector on that cylinder is 575 (the first being 0). The first sector on the ALTSCTR slice then is 576. Allow two cylinders for the ALTSCTR slice. The sector count is 2 * 576 = 1152. To calculate the last sector:

sector count (1152) + first sector (576) - 1 = 1727

The slice number we'll use for the ALTSCTR slice in this example is 8.

c. Calculations for the `/export/home` slice
Since the `ALTSCTR` slice ended at sector 1727, the `/export/home` slice (tag 8) should begin at 1728. The last sector is 777599. The difference between the two plus 1 = 775872. This is the sector count. The slice number we'll use for the `/export/home` slice is 7.

This concludes the calculation of the slices in the Solaris `fdisk` partition. Based on the calculations done so far, *tempfile* should be modified to look like the following:

```
*
* /dev/rdsk/c0t0d0s2 partition map
* Flags
*   1: unmountable
*  10: read-only
*
*
* Partition   Tag   Flag  First Sector  Sector Count
      2        5     00        0            777600
      7        8     00        1728         775872
      8        9     01        576          1152
```

5. Run the `fmthard` command to create the VTOC and copy the two default boot programs onto your hard disk.
Use the modified ASCII file that you set up in the previous step to tell `fmthard` how to slice up the Solaris `fdisk` partition.

```
# fmthard -s tempfile -n "name"  /dev/rdsk/c0t1d0s2
```

6. Run a surface analysis on the disk using the `diskscan` command.
Specifying the `-n` flag causes `diskscan` to suppress linefeeds in its progress reports. See `diskscan(1M)`. In this example, specify `s2` representing the entire Solaris partition on the second disk and redirect the output to a temporary file.

```
# diskscan -n /dev/rdsk/c0t1d0s2 2>/tmp/errors
```

7. **Run the** `addbadsec` **command (if needed).**
   The `diskscan` command creates a list of bad blocks. It might be necessary to edit out other error messages generated by `diskscan`. In addition, each transaction retry is announced to the system administrator in the form of a message on the console. The block numbers given are absolute, so the `p0` device node corresponding to the target device given to `diskscan` should be specified. In this example, assume that `diskscan` reported block numbers 24994 and 56553 as being defective.

```
# addbadsec -a “24994 56553” /dev/rdsk/c0t1d0p0
```

   Alternatively, if a number of bad blocks were collected in the file `/tmp/errors`, you can edit it to remove extraneous information and use it as input to `addbadsec` as shown below:

```
# addbadsec -f /tmp/errors /dev/rdsk/c0t1d0p0
```

Note – If the block that is defective is part of a file system structure or other special area of the disk, you may have to repartition the disk in order to use it.

8. **Run the** `newfs` **command to create a file system(s).**
   Create a file system on the new disk. (See `newfs(1M)`.) For example:

```
# newfs /dev/rdsk/c0t1d0s7
```

   This creates a file system of size 775872 sectors on slice 7, as shown in the modified partition table example on page 86.

9. **Make a mount directory for the file system.**
   For example:

```
# mkdir /export/home
```

10. **Mount the file system.**
    For example:

```
# mount -F ufs /dev/dsk/c0t1d0s7 /export/home
```

11. **Add an entry to the** `/etc/vfstab` **file if you want to have the file system automatically mounted. See** `vfstab(4)`**.**

Installing the JumpStart Software on a SPARC System

C

There may be times when you want to install the JumpStart software (Preinstall Boot Image) on a system. Some new systems have the JumpStart software already preinstalled, which enables you to power on the system and have it automatically install the Solaris software.

SPARC – When you power on a system, the system looks for the JumpStart software on the disk slice that it boots from by default.

SPARC How to Install the JumpStart Software on a System

The procedure to install the JumpStart software involves:

- Booting the system in single-user mode from the Solaris CD.
- Using the `re-preinstall(1M)` command.

Warning – This procedure overwrites existing data on the specified disk. If you want to keep the data, back up the files before proceeding. See *File System Administration* for more information on backing up data.

Follow this procedure to install the JumpStart software on a system:

1. **Identify the system's boot disk, if necessary. Use the output of the `mount` command to identify the device name of the root (/) file system. For example, `/dev/dsk/c0t3d0s0`.**
2. **Halt the system.**
3. **Boot the system in single-user mode from the Solaris CD.**
   See Chapter 2, "Booting a SPARC System"*for information on booting your system in single-user mode.*
4. **Use the `re-preinstall` command to install the JumpStart software on the system.**
   target_slice is the device name you found in step 1 (slice 0 is usually used).

```
# /usr/sbin/install.d/re-preinstall target_slice
```

For example, the following command would install the JumpStart software on the system's `c0t3d0s0` disk.

```
# /usr/sbin/install.d/re-preinstall c0t3d0s0
```

x86 External Cache Issues D

This appendix contains information regarding the performance of certain x86 systems running the Solaris software.

x86 External Cache Issues

Many x86 systems include an L2 external (or physical) cache. The external cache is a physical cache that represents a portion of physical memory. The L2 external cache improves CPU performance by reducing the time needed for data to be stored and retrieved by the CPU.

For some types of external caches, however, the cache does not accurately reflect the contents of the physical memory, due to a hardware problem.

The Solaris software provides a default mechanism that solves this problem by flushing the external cache in x86 systems. However, this mechanism actually reduces CPU performance on systems with functioning external cache designs (for example, systems using the Intel® Cache Controller chip set).

To improve CPU performance in x86 systems where the cache accurately reflects physical memory, you should turn off the Solaris cache flushing operation.

D

x86How to Turn Off Cache Flushing

1. **Edit the** `/etc/system` **file.**
2. **Add the following line or uncomment if it already exists:**

   ```
   set cacheflsh = 0
   ```

3. **Reboot the system.**

Index

C

D

E

F

H

I

K

L

M

N

O

P

R

S

T

U

W