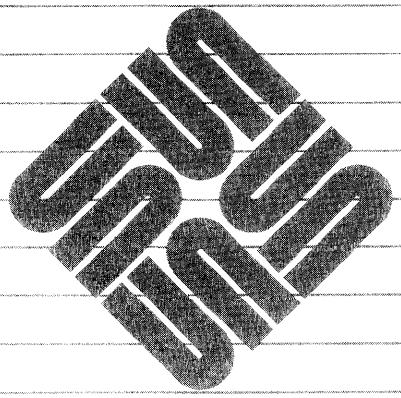
sun microsystems

Installing UNIX *on the* Sun Workstation®



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

Part Number: 800-1521-05
Revision: A of 19 September 1986

Credits and trademarks

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

Multibus is a trademark of Intel Corporation.

UNIX is a trademark of AT&T Bell Laboratories.

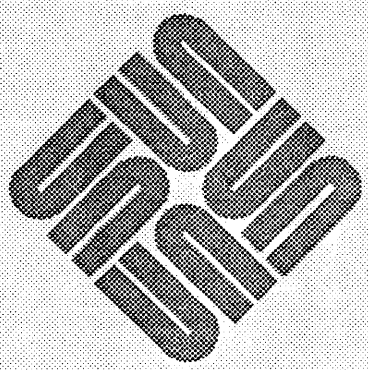
VMEbus is a trademark of VMEbus Manufacturers Group.

Contents

Figures

1

Introduction



1

Introduction

This manual is designed to help you install Sun's distribution of the UNIX† Operating System onto your Sun workstation for the very first time or as a system upgrade. The goal of this manual is to allow you to take your workstation from initial hardware power-up through the UNIX installation

A major portion of this document is devoted to *Setup*, the Sun release installation system. *Setup* provides a user friendly, intelligent installation editor that allows users to customize their configuration. *Setup* provides a SunWindows based user interface for those users with bit-mapped display devices, and a terminal based interface for those customers with terminals for console devices.

Before you begin, it is best to read through the major sections of this manual and familiarize yourself with the terminology in Chapter 9-Glossary. We use several conventions to make the directions clear and consistent. Some of these conventions are UNIX conventions, such as device names; others are simply Sun documentation conventions. The Glossary will help you understand these.

Be sure to install your workstation hardware according to the procedures in the Sun *Hardware Installation Manual*. This will be critical for the installation of Unix onto your workstation.

1.1. Conventions Used in This Document

Use of Fonts in Document

Various fonts are used in this manual to distinguish what you type from what the system types at you, and to identify those items for which either you or the system must substitute a variable. Our conventions are:

SMALL CAPS
: Names spelled with all capital letters (such as, UNIX and FORTRAN) are printed in small caps for readability.

`bold listing font`
: This font indicates things that you should type at your workstation.

† UNIX is a trademark of AT&T Bell Laboratories.

italic font This font is used as a place holder for words, numbers, or expressions that you define. Examples of these are: file names, function arguments, variables, and field names. Italics are also used in the conventional manner to emphasize important words and phrases.

`listing font`
This font indicates what the system types back at you. It also indicates literal values such as program names, function names, procedure names, variable names, field types, file listings, and session output.

Naming Scheme for Reference Manual Pages

References to commands and utilities from the *Commands Reference Manual for the Sun Workstation* and the *System Interface Manual for the Sun Workstation* use the notation:

passwd(1)

to indicate the *passwd* page in Section 1 of the manual pages. There are a total of eight sections: Sections 1, 6, 7, and 8 appear in the Commands Reference Manual; sections 2, 3, 4, and 5 appear in the *System Interface Manual.* Thus, *passwd*(5) means refer to the *passwd* manual page in the Section 5 pages of the *System Interface Manual.* The notation *spline*(1G) means that this is a Graphics command in the Section 1 pages — you'll also see the addended letter to indicate other subsections, like *title*(3M) for a page in the Math Library subsection of the Section 3 pages, or *title*(3N) for a page in the Network Library subsection.

2

Getting Started

2

Getting Started

This chapter is devoted to helping ready your workstation for using *Setup*. The various steps in this chapter must be completed before you can actually use *Setup*.

2.1. What is on the Distribution Tape?

The software needed to load the UNIX system is contained either on three half-inch magnetic tape reels, or on four quarter-inch tape cartridges. See *Appendix G* for a complete listing of the files on the distribution tape.

2.2. General Information

Read the Chapter 9-Glossary and familiarize yourself with the information in the following sections.

Abort Procedure

As you proceed through this manual, there may be a need to abort certain procedures. Listed below are "abort sequences" for Sun workstations. Familiarize yourself with the one that pertains to your workstation.

To return to the PROM monitor at any time during installation — either because you are asked to in the installation procedure or because you have botched things entirely — you can type what we call an **abort sequence** on your keyboard. The abort sequence usually consists of two keys typed in sequence; the first key is **held down** while the second key is typed. The keys vary with Sun Workstation model and keyboard type:

- If your Sun-1 keyboard has a 'SET-UP' key, the abort sequence is 'SET-UP a' (hold down the 'SET-UP' key while typing 'a').
- If your Sun-1 has an 'ERASE-EOF' key, the abort sequence is 'ERASE-EOF a' (hold down the 'ERASE-EOF' key while typing 'a').
- On a Sun-2 keyboard, type 'L1 a'. (hold down the 'L1' key in the upper-most left-hand corner while typing 'a').
- On a Sun-3 keyboard, type 'L1 a'. (This is the same as the Sun-2 keyboard.)
- On a standard terminal (if it is the console) the 'BREAK' key generates an abort.

UNIX Device Naming

UNIX has its own set of names for devices. These names are fairly arbitrary, but are often based on abbreviations for the controllers used to drive the devices. When UNIX boots up, it probes the system for the device and controller configuration and reports what it finds there. Since you will be using these names during installation and in most of your administrative dealings with the system it's a good idea to identify your system's devices at this point, and to remember their UNIX device names:

Table 2-1 *UNIX Device Names*

Name	*Device*
xy	Xylogics 440/450/451 SMD disk controller
sd	SCSI disk controller
mt	Nine-track 1/2 in. magnetic tape/tapemaster controller
ar	Archive 1/4 in. tape
st	SCSI 1/4 in. tape controller
xt	Nine track 1/2 in. mag. tape/Xylogics 472 tape controller
ec	3COM Ethernet controller
ie	Sun-2 Ethernet controller
le	Sun-3 Sun Lance Ethernet controller

In the walkthrough of the installation, when we ask you to type in your *tape*, *disk*, or *ethernet interface* name, you should type in the UNIX device name for the appropriate device.

2.3. Determining Network Information

NOTE

The procedures in this section apply only to systems with Ethernet. If you do not have Ethernet, skip to the next section, Loading the Bootstrap Program.

NOTE

If you have an existing network that you do not intend to upgrade, you might need to make it compatible with the new network software. In any case, complete the steps given below for any machines you are installing or upgrading, and when you are finished, read the section How to Make Current Networks Compatible With Older Networks *in the Sun System Administration Manual.*

Before beginning actual installation, you must know:

1) The full Internet address (network number followed by unique host number) for each workstation you are setting up — whether it is a server or a client,

2) The hardware Ethernet address of each client machine if you are installing a server/clients configuration of machines, and

3) The yellow pages domain name of the workstation or group of workstations you are installing.

These items will be requested later during installation, but must be obtained now.

For definitions of Internet address, domain name, and hardware Ethernet address, and for instructions on how to determine them, see the *Chapter 9-Glossary.*

Please remember that:

- You can use Sun Microsystems' default network number (192.9.200) if you have not been assigned a network number by ARPA, or if you are not connected to a higher level network.
- You must have both sets of Distribution Tapes if you are installing a heterogeneous server, i.e, you want this server to support both Sun-2 and Sun-3 hardware.
- You need a domain name if you are running the yellow pages, and there are duplicated hostnames or user id's (the domain name simply distinguishes network nodes). If you have only one */etc/passwd* file and one */etc/hosts* file for your organization (if there is a single unique user id and machine name space), the domain name is still necessary if you are running the yellow pages.
- The range of the Host Number is dependent on the class of network addressing. For a typical network that uses Class "C" addressing, the range is between 1-254.
- You can allow *Setup* to assign host numbers automatically if you are installing a new network, but you should assign host numbers yourself if you are adding workstations to an existing network to avoid accidental duplication of host numbers.
- Obtain each client's machine hardware Ethernet address by powering up the workstation, and checking the six-byte hexadecimal address displayed in the monitor power-up banner. When you turn the machine on, you'll see the Sun logo, and a message like the one in the example below. Abort immediately when the machine begins to auto-boot:

```
Self Test completed successfully.

Sun Workstation, model_type, keyboard_type
ROM Rev N, some_number_MBytes memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .

abort by typing the appropriate abort sequence here

Abort at some address
>
```

  You will need the entire six bytes of the displayed Ethernet address later; copy them down.

2.4. Loading the Bootstrap Program

This section covers the first steps of actual installation: loading the distribution tape on your tape drive, and using the PROM Monitor to load the bootstrap program from tape. The bootstrap program is used to load other programs from tape into memory.

In order to help you with these steps, we have put the basic commands for a simple walkthrough installation in the left hand margins of the following sections. However, these are only the basic commands. We recommend that you read this chapter in order to determine the type of system you are installing **before** using these commands since they may not meet your specific installation needs.

Step 1: Loading the Tape

Load the tape.

If you have any questions about your tape drive, see the subsystems chapter in the *Hardware Installation* manual for your machine.

Step 2: Loading the Bootstrap Program

1. Turn on the Sun Workstation which you are installing.

NOTE *If you are re-installing an existing system, you will need to halt the system. To do this become super-user and type the command:* `/etc/halt`.

Almost immediately, the PROM monitor displays its power-up banner, which looks something like the example below, and then the machine begins to auto-boot. Stop the auto-boot immediately by typing the appropriate abort sequence for your machine (abort sequences are described in Section 2.2 of this chapter). When you abort the auto-boot, you return control to the monitor, and it displays its prompt (>):

```
Self Test completed successfully.
Sun Workstation, model_type, keyboard_type
ROM Rev N, some_number_MBytes memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx
Auto-boot in progress . . .
abort by typing the abort sequence for your machine here
Abort at some_address
>
```

2. Now boot the general purpose bootstrap program from the tape by typing a **b** (for boot), followed by the two character device abbreviation for your tape drive type, followed by open and closed parentheses.

   In the following, please remember to substitute the proper device abbreviation for your tape controller for *tape*: **ar** for Archive 1/4'' Tape Controller, **st** for SCSI Tape Controller, **mt** for a 1/2'' tape controlled by a Tapemaster Controller, or **xt** for a 1/2'' tape controlled by a Xylogics 472 Tape Controller. For more information on device abbreviations or conventions used in these procedures, see the table at the beginning of this chapter under General Information, Unix Device Names.

> **b** *tape*()
> Boot: *tape* (0,0,0)

When you type the command, the monitor echoes it back to you, with the parameters filled in. A boot command looks like this:

```
>b tape()
Boot: tape (0,0,0)
Boot:
```

As shown in the example, the bootstrap program displays a **Boot:** prompt at the start of a line when it is ready to accept commands. If you have a 1/4" tape and the > prompt returns instead of **Boot**, the tape may need retensioning. Try entering:

```
>b tape(0,0,100)
Boot: tape(0,0,0)
Boot:
```

2.5. Using *diag* to Format and Label Disks

This section describes the third step of installation: loading the *diag* program from the distribution tape and using it to format and label your disk(s).

NOTE *If you are upgrading your system between major Sun releases, you may skip* `diag` *completely and proceed to* Section 2.6-The Mini UNIX System.

Using *diag* has two phases: in the first phase, *diag* prompts for information about the type of disk drive and controller, then 'configures' itself to work with that disk and controller. In the second phase, *diag* allows you to format and label your disk by executing a series of protocols as you type the commands *format* and *label*.

If you have multiple drives/controllers, complete both phases for your first drive on your first controller, and then loop back to begin the cycle for your next device. We give directions for this at the appropriate time.

Phase One: Specifying Hardware Configuration

Boot: *tape* (0,0,3)

1. First boot *diag* from your distribution tape. Type the following, (replacing *tape* with the appropriate device abbreviation for your controller):

```
Boot: tape (0,0,3)
Size: some_number+some_number+some_number bytes
[ varies with release ]
```

Read this section, respond appropriately.

2. When *diag* starts up, it displays a sign on message, then it begins prompting for hardware information. It asks what sort of disk controller(s) and disk drive(s) you have, then it uses this information to configure itself to work with your hardware.

   First, *diag* asks for the disk controller type:

```
Version sccs version_number and date
Disk Initialization and Diagnosis
```

```
When asked if you are sure, respond with 'y' or 'Y'

specify controller:
        0 - Interphase SMD-2180
        1 - Xylogics 440 (prom set 926)
        2 - Xylogics 450/451
        3 - Adaptec ACB 4000 - SCSI/ST506
        4 - Emulex MD21 - SCSI/ESDI
which one?  type the number for your controller type
```

3. Next, *diag* asks what address this controller occupies on the main system bus. Unless you have an unusual controller configuration, use the correct address from this table:

Table 2-2 *Default Addresses for Disk Controllers*

Controller Type	*Address (hex)*	
	1st Controller	2nd Controller
Xylogics 440/450/451	ee40	ee48
SCSI for Multibus System	80000	84000
SCSI for VMEbus System	200000	
SCSI for the Sun-3/50	140000	

`Specify controller address:`

`Device address:`

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

`Specify controller address on the mainbus (in hex):` *address*

`Device address:` *address you entered*

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

   `Which target?` *SCSI bus address*

5. Next, *diag* asks for the disk's unit number on this controller. The correct unit number is **0** for the first (or only) drive on this controller, and **1** for the second:

   `Which unit?` *unit number for the drive you're working with*

6. Now *diag* asks for the type of disk drive you are working with. It displays a menu of the different disks it knows about, and asks you to specify your disk type. Note that *diag* contains a different menu for each controller. It displays the correct menu for the controller you specified earlier.

If you do not know what type of disk you have, check behind the cover of your Sun pedestal or the back of a Sun shoebox for a label that states the disk drive configuration. If the label is not there, it should be apparent from the disk.

For instance, if you specified an Adaptec ACB 4000 SCSI, *Diag* displays the following:

```
Specify drive:
0 - Micropolis 1304
1 - Micropolis 1325
2 - Maxtor XT-1050
3 - Fujitsu-M2243AS
4 - Vertex V185
5 - Micropolis 1355
6 - Toshiba MK156F
```

`which one?` *type the number for your drive type*

`ncyl` *number* `acyl` *number* `nhead` *number* `nsect` *number* `interleave` *numb*
`status:`
`[` ...*status information*... `]`

After you select a disk drive from the menu, *diag* displays a table of physical data about that disk. This includes the number of cylinders, number of alternate cylinders, number of heads, and number of sectors per track. *diag* then displays drive-specific status information.

Now, *diag* knows everything it needs to know about the controller and the disk you are using. It displays its prompt:

```
diag>
```

7. If you are not sure you selected the correct disk type in step 6 above, use the *verify* command to read the label on the disk:

```
diag> verify
verify label
id: a disk type
    Partition a: starting cyl=0, # of blocks=#
    [ and so on ...]
```

If the drive type matches the disk drive type you specified in step 6, continue to phase 2. If not, type **diag** to re-start the initialization process, then go back to Step 2:

```
diag> diag
specify controller:
[ and so on ...]
```

You are now ready to begin the second phase of the *diag* operation: formatting and labeling the disk.

Phase Two: Formatting and Labeling the Disk

In this second phase, you use *diag* to format a disk, then to label it.

NOTE *The location of the hardcopy list depends on the workstation type. For example, it is usually taped to the front or top of the pedestal. If you don't see it or have misplaced it during unpacking, look for a second copy taped to the disk drive housing inside the pedestal (remove the two Phillips screws from the top rear of the enclosure, slide the top of the beige metal housing off, and check).* **Be sure to replace this second copy when you have used it.** *DO NOT OPEN THE SHOEBOX!!! The defect list is on the back of the outside cover. Also note that on some drives, the hardcopy list groups defects by head; the cylinder and bytes from index numbers appear in the CYL and BI columns. On others, the numbers appear in the CYL, H, and BYTE columns respectively.*

For servers, use the *partition* subcommand to select and possibly modify an alternate label for the disk, and then use the **label** subcommand to write this label on the disk. For standalone systems, simply use the **label** subcommand to write a default label to the disk.

NOTE *If you wish to change partition sizes, see Appendix F of this manual for the* **`partition`** *command. You can also use Setup to change all partitions except the root or to make the swap space larger.*

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

Formatting SCSI Disks

Note that all disks shipped by Sun are formatted for testing at the factory. Because of the possibility that additional surface damage may occur during transit, it is our policy to have the customer reformat the disk on site.

`diag> format`

1. Begin by typing **format** to call up *diag*'s SCSI formatting subprogram. It displays its prompt:

```
diag> format
SCSI format.
format>
```

If the disk is unformatted or the defect list has been corrupted, *diag* will print a message saying that it was unable to read the defect list off the disk. The SCSI format subprogram has various utilities which help you prepare and format a new disk. To see a list of its capabilities, type:

```
format> ?

SCSI format subcommands:
   f: format disk
   p: print defect lists
   a: add defect to physical list
   b: bias added defects
   d: delete defect from list
```

```
c: clear defect list
s: surface analysis
r: reassign logical block
t: translate logical block # to physical
q: quit format
```

2. Check to make sure that the disk defect list written on the disk is the same as the hardcopy list shipped with your drive. Enter the command **p** to display it on the screen:

NOTE *The defect list is read from the disk to RAM whenever you enter the format subsystem. To write the list from RAM to disk after making changes, you must use the subcommand* **format** *to format the disk. If you leave the format subsystem without formatting the drive, you loose any changes you made to the defect list. The only exception to this is when you use the subcommand* **reassign**. *This command immediately adds the defective sector to the defect list on the disk, so reformatting is unnecessary.*

```
format> p
Defect List-Physical Format
Defect  Cylinder  Head   Bytes from Index
1       34        1      9213
etc.
```

3. Make sure this list matches the hardcopy disk defect list.

   a) If they match, continue with step 4.

   b) If the hardcopy list shows defects that are not displayed on the screen list, use the *a* command to add to the list on the disk:

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

   c) If you cannot read the list from the disk, use the *a* command as shown above to type in the entire hardcopy list. The only exception to this is a brand new ESDI drive. Even though the defect list cannot be read off the disk, the controller can get at it. Simply format the drive; the controller will automatically use the built in defect list. After the format, diag will extract the defect list from the controller and store it on the alternate cylinders.

   d) After making any changes to the list on the disk, use the *p* command to display the changes on the screen. Check again to make sure the copy on the screen matches the hardcopy list.

4. When you have verified the defect list, format the disk with the format (f) command. After you type this command, the system displays a warning, then asks for confirmation:

```
format> format
DISK FORMAT, DESTROYS ALL DISK DATA!
are you sure? y
formatting.....
```

The formatting process takes several minutes or more. At the end, you should see a "done" message. For ESDI drives, you may also see a message about reassigning grown defects. This is ok. If you see any other message ("SCSI reset", for example), the formatting process did not succeed, and the defect list was not recorded on the disk. **You must format the disk again.**

NOTE *If your disk was previously formatted, you can proceed to* Labeling SCSI Disks.

5. When you have successfully formatted the disk, do a surface analysis using the **s** sub-command. This analyzes the entire disk, then displays a list of any defective sectors it found. For a new disk, you should do five surface analysis passes:

```
format> s
SURFACE ANALYSIS-DESTROYS ALL DISK DATA!
Are you sure? y
# of surface analysis passes (5 recommended)? 5
```

Five passes may take an hour or more to complete. When surface analysis is done, it displays a message to announce that fact.

NOTE *Surface analysis on ESDI disks does NOT destroy any disk data. Thus confirmation is not asked for. You can use surface analysis on live ESDI disks without worrying about destroying files.*

6. For ESDI disks, *diag* will automatically reassign any defective sectors it finds. It also adds these sectors to the defect lists immediately, so there is no need to format the disk again. If all the defective sectors were successfully reassigned, you can go onto the next step. If one of the reassigns failed for any reason, you should reassign that sector by hand then rerun the surface analysis to be sure the disk is now clean. For ST506 disks, *diag* simply reports the bad sectors it finds and adds them to the defect list in RAM. When it is done, it tells you to re-format the disk:

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

Before continuing, you must:

a. Reformat the disk (go back to step 4).

b. Continue this loop until the surface analysis reports that it found no bad sectors:

Once the surface analysis completes without finding any bad sectors, the format is successful. You may now continue to the next step, labeling the disk.

Labeling SCSI and ESDI Disks

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

NOTE *If you want to increase the size of the root partition, you will have to refer to Appendix F of this manual for the* **partition** *command.* **Setup** *cannot do this particular operation. However for all other partitioning, use* **Setup**.

1. To label your disk, type the **label** command to the *diag* prompt. (type the **q** to exit **format** and return to the *diag* program). *diag* then asks if you want to use the logical partition map that is 'built in' to the program, and then asks for confirmation. Default partitioning for Sun-supplied SCSI disks is shown in the following table:

Table 2-3 *Default Partition Sizes for SCSI Disk Subsystems*

SCSI Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Micropolis 1304	50	8.1	8.4	43.1	*unused*	*unused*	*unused*	26.5	*unused*
Micropolis 1325	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.6	*unused*
Maxtor XT-1050	50	8.1	8.4	44.4	*unused*	*unused*	*unused*	27.9	*unused*
Fujitsu M2243AS	86	8.1	17.1	70.8	*unused*	*unused*	*unused*	45.6	*unused*
Vertex V185	85	8.1	17.1	70.9	*unused*	*unused*	*unused*	45.5	*unused*
Micropolis 1355	170	8.1	17.1	141.7	*unused*	*unused*	*unused*	116.5	*unused*
Toshiba MK156F	170	8.1	17.1	141.8	*unused*	*unused*	*unused*	116.6	*unused*

```
diag> label
```

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

```
diag> label
label this disk...
OK to use logical partition map
```

2. After labeling the disk, *diag* automatically verifies the label it has just written. For example, the verify for a Micropolis 1325 might look like this:

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

```
verify label
id: <Micropolis 1325 cyl 1022 alt 2 hd 8 sec 17 >
        Partition a: starting cyl=0, # blocks=15912
    [ #'s vary with disk ]

        Partition b: starting cyl=117, # blocks=33456
        Partition c: starting cyl=0, # blocks=138448
        Partition g: starting cyl=363, # blocks=89080
diag>
```

3. This completes the formatting and labeling process for a single SCSI disk. If you have a second disk drive on your controller, or another controller and more drives, you have to return to the beginning of *diag*'s first phase by responding to the *diag* prompt with the command **diag**:

```
diag> diag
```

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

```
diag> q
```

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

```
diag> q
Boot:
```

Now continue with the procedures in Section 2.6-The Mini Unix System.

Formatting SMD Disks

```
diag> format
```

1. Begin by typing format to call up `diag's` SMD formatting subprogram (`diag` remembers again the controller type from its configuration phase):

```
diag> format
DISK FORMAT-- DESTROYS ALL DISK DATA!
Are you sure? y
Formatting
NNN
Verifying
NNN
diag>
```

This process takes a while.

As *format* formats each track, it displays the cylinder and track number (represented by NNN in the above display). Before *format* exits, it updates the defect list. Note that if it is interrupted, it does not update the list, and the process must be repeated.

Now, you may continue to the next step, labeling the disk.

Labeling SMD Disks

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

Default Partitioning: **label**

Default partitioning is adequate for most systems. The default tables include the "a" (root), "b" (swap), "c" (entire disk), and "g" (for */usr*) partitions.

NOTE *If you want to increase the size of the root partition, you will have to refer to Appendix F of this manual for the* **partition** *command.* **Setup** *cannot do this particular operation. However for all other partitioning, use* **Setup**.

Default values for these partitions (on standard SMD disk subsystems shipped by Sun) are:

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

Table 2-4 *Default Partition Sizes for SMD Disk Subsystems*

SMD Disk	*Raw*	*Partition Sizes (MBytes)*							
		"a"	"b"	"c"	"d"	"e"	"f"	"g"	"h"
Fujitsu 2322	168	8.1	17.1	134.5	*unused*	*unused*	*unused*	109.1	*unused*
Fujitsu 2351 Eagle	474	8.1	17.1	395.7	*unused*	*unused*	*unused*	369.8	*unused*
Fujitsu 2333	337	8.2	17.2	281.6	*unused*	*unused*	*unused*	256.3	*unused*
Fujitsu 2361 Eagle	690	8.2	17.2	576.3	*unused*	*unused*	*unused*	550.9	*unused*
CDC 9720	347	8.3	17.2	281.9	*unused*	*unused*	*unused*	256.3	*unused*

If this partitioning is suitable for your system, proceed as follows.

```
diag> label
```

1. Type the **label** command in response to *diag*'s prompt. When you give the command to *diag*, it asks if you want to use the logical partition map that is 'built in' to the program (the default map), and then asks for confirmation before proceeding:

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

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

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

```
verify label
id: <Fujitsu-M2322 cyl 821 alt 2 hd 10 sec 32 interleave 1>
        Partition a: starting cyl=0, # blocks=15884
    [ #'s vary with disk ]

        Partition b: starting cyl=50, # blocks=33440
        Partition c: starting cyl=0, # blocks=262720
        Partition g: starting cyl=155, # blocks=213120
diag>
```

```
diag> diag
```

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

```
diag> diag
```

Note that you must complete both phases of *diag* for each of your disks. Be careful every time you run *diag* to respond with correct values for the controller type, controller mainbus address, disk unit number, and disk type.

```
diag> q
```

4. If you have only one disk, or if you are done formatting and labeling all of your disks, you are ready to continue with the next phase of installation. Return to the bootstrap program by typing the **q** (quit) to *diag*:

```
diag> q
Boot:
```

2.6. The Mini UNIX System

After a disk is formatted and labeled, you can store information on it. To do this, you must first load and boot a minimal subset of the UNIX system called mini UNIX onto your disk. This contains the programs you will need to run *Setup* properly.

Loading the Mini UNIX System

Boot the standalone *copy* program from tape, and use it to copy the mini UNIX system from the distribution tape to your disk.

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

Remember to substitute the correct device abbreviation for *disk* and *tape* (device abbreviations are given at the beginning of this chapter). Also, note that the *copy* program prompts for the source (`From:`) and destination (`To:`) of the copy; there is very brief (approximately ten seconds) delay between the prompts as the *copy* program reads from the tape:

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

Copying in the mini UNIX system takes about seven minutes using a half-inch tape, and about nine minutes using a quarter-inch cartridge. At the end of the copy, the *copy* program returns control to the bootstrap program:

```
Copy completed   some_number bytes
Boot:
```

Booting the Mini UNIX System

1. Now that you have an operable mini UNIX system on disk, you can tell the the bootstrap program to boot mini UNIX from the disk. Because this boot is an unusual one, you must specify the –a (for ask me) option on the boot command, and also the –s (come up single user) option, as follows:

Boot: *disk* **(0,0,1)vmunix -as**

```
Boot: disk (0,0,1)vmunix -as
Size: some_number+some_number+some_number bytes
Sun UNIX 4.2  Release 3.2 (GENERIC) #1: Tue October 10 20:35:13
Copyright (c) 1986 by Sun Microsystems, Inc.
[ ...about twenty lines of configuration messages... ]
```

root device? *disk***0***

2. As the mini UNIX system comes up, it displays some messages about the configuration of the system on which it is running, and finally queries you, asking for its root file system. The root file system at this stage is "*disk* **0***", which has a special meaning to the mini UNIX system. If you have a Xylogics disk controller, your root device is **xy0***; if you have an Interphase controller, it is **ip0***; and if you have a SCSI disk controller, it is **sd0*** — the asterisk is part of the device name:

```
root device? disk0*
```

Depending upon your hardware, you may be asked to set the date at this point:

```
using number buffers containing number bytes of main memory
WARNING: clock gained xx days-- CHECK AND RESET THIS DATE!
Single user boot -- fsck not done
#
```

If so, continue with the following section.

2.7. Setting the Date

Systems that have Time of Day clocks should set the date and time at this point, so that when the real root and */usr* file systems are loaded, all the log files and such will start with the right dates.

1. When the mini UNIX system starts up it displays a # prompt. You can now set the correct system date using the *date*(1) command:

   # date *yymmddhhmm*[*.ss*]

   where *yy* is the last two digits of the year; *mm* specifies month; *dd* designates day of the month; *hh* is hour (on a 24-hour clock); the next *mm* is minutes elapsed; and the optional *.ss* specifies seconds.

For example, to set the date at 4:43 PM and 30 seconds on July 30, 1986 you would type:

```
[ This is only an example; don't enter this information. ]
# date 8607301643.30
```

The system would then echo back to you:

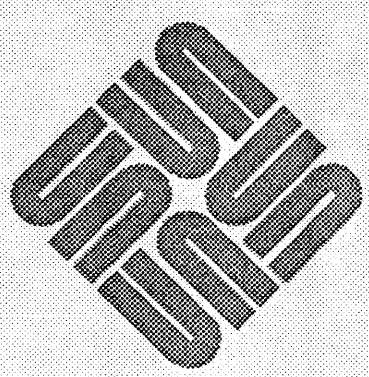
```
Mon October 13 16:43:31 PDT 1986
```

NOTE *All Sun-3's have Time of Day chips, but Sun-2/50 and Sun-2/160 may not. If you get the following: ' "WARNING" no tod clock -- CHECK AND RESET DATE!', you should reset the date after full UNIX is booted*

Now you are almost ready to begin *Setup*. Proceed now to the next chapter. You will read through the overview before proceeding to the commands that will enable you to use *Setup*.

3

Setup Overview



3

Setup Overview

3.1. Introduction

Setup will present you with a set of different "forms", each of which asks for information pertaining to a specific aspect of system configuration. You can edit the information on any form at any time; thus allowing you to correct mistakes or experiment with different configurations. When you have entered all the information needed by *Setup*, "pressing" a button will cause *Setup* to begin the installation.

Setup comes in two versions: one runs on cursor addressable terminals while the other takes advantage of the Sun window system and requires a Sun bit-mapped display. Both versions of *Setup* have the same functionality and gather the same information. Independent of the interface you choose, this chapter describes each of the forms used in *Setup* and explains what information is requested. The next two chapters explain how to use each interface.

Setup's electronic forms are very similar to "paper" forms or questionaires. They are composed of "items" that ask for a piece of information. There are 4 types of items: Text, Choice, Toggle, and Button.

- Text items are where you type in information such as a name ("fill in the blanks").
- Choice items present you with a list of choices and ask you to choose one element from the list (The equivilent of "Check the appropriate box" on a questionaire).
- Toggle items allow you to select a set of elements from a list ("Check all boxes that are appropriate"); it is called a Toggle item because you are able to toggle "on" and "off" elements in the list.
- Button items are used to invoke some action.

When *Setup* starts up, you will see 3 regions on the screen. The top region is the control panel. It consists of buttons that display the forms which you will fill out. Below the control panel is the message region which displays error messages and prompts. The rest of the screen displays the forms.

There are five forms in *Setup*, not all of which will need to be filled out for the typical configuration. The following is a brief discussion of each *Setup* form:

- Workstation Form: The information requested by this form describes the workstation being configured. For example on this form, you indicate if

your workstation is a file server or a standalone.

- Defaults form: This form is used to change the value, from the default, of a number of control parameters such as the units used in the display of disk partition sizes.

- Client form: If the workstation is a file server, this form allows you to enter the information about the diskless clients being served by the workstation.

- Software form: This form allows you to choose what optional software you would like installed on your system.

- Disk Form: The Disk Form allows you to change the hard partitions of a disk.

Here is the suggested order for filling out the forms:

Workstation Form
Defaults Form (optional)
Optional Software Form
Client Form (if configuring a server)
Disk Form (if default partitions are not acceptable)

If it does not make sense to fill in a form given the current configuration, *Setup* will not present it to you. Thus, in the simplest case of configuring a standalone workstation, only two forms will need to be filled in (Workstation and Optional Software). A file server whose clients use only the default ND partitioning need only fill in the Client form in addition to these. Only in the case that you require the maximum flexibility in configuring a file server must you fill in all five forms.

In the following sections, we will examine in some depth the *Setup* forms and each item of information they request. You should scan over them now to determine what information you need to gather, and then refer back to them while you use the interface that you select. For more rapid indexing, the form item labels are printed in the left margin.

3.2. Workstation form

The Workstation form is the first form you need to fill out. It requests information about the workstation that is being configured. If your workstation is on a network, you need to spend some time before using *Setup* determining some important information such as your network domain name and network number, whether your machine is tapeless, and whether or not you wish to use the Yellow Pages. Below is a list of the items you will see on the Workstation form.

Workstation Name

The name of your workstation here is typed in here. If you are on a network, be sure that no other machine in your domain has the same name as yours.

Workstation Type

Initially, *Setup* does not know if your workstation is a standalone or a file server. Cycle through this item and choose the appropriate type. If you choose file server, a new item will appear which asks you to indicate the cpu types you will be serving.

CPU-Types Served

This is where you select the CPU-types served by your workstation. This is a toggle item so you can select more than one CPU-type. File servers must always serve at least their own CPU-type.

Tape Device

This is where you select the type of tape drive you will be using to read the distribution tape.

Tape Location

This item gives you a choice between a local or remote tape drive. If you choose remote, two more items will appear below this item asking for the name and internet number of the machine with the tape drive. See Appendix B, *Installing UNIX on Tapeless Workstations*.

Server Name

This is where you type in the name of the workstation whose tape drive you will be using. This field only appears if you choose "remote" for tape location.

Server Internet Number

Here is where you type in the internet number of the workstation whose tape drive you will be using. This field only appears if you choose "remote" for tape location. Use only PERIODS between the numbers, any other character will not be read properly: 192.9.200.11.

Ethernet Interface

If your machine has only one Ethernet interface, *Setup* will select it for you. If you have more than one option, choose the the type of controller you wish to use. If "None" is not chosen, items will appear asking for your host number and Yellow Pages type.

Host Number

Automatic host numbering is done by default so *Setup* will assign you a number. You may disable this feature by visiting the Defaults form. The automatic host number can be changed to the actual host number.

Yellow Pages Type

This item asks you to indicate your relationship with the Yellow Pages. If you choose a YP type of "Master" or "Client", an item asking for the name of the YP "Domain" will appear. If the YP type is "Slave", the domain item will appear along with items asking for the name of the YP master and its internet number.

Domain

Type in the name of your Yellow Pages domain here.

Master Name

Type in the name of the machine which will act as your Yellow Pages master.

Master Internet #

Type in the internet number of the machine which will act as your Yellow Pages master. Use only PERIODS between the numbers, any other character will not be read properly: 192.9200.1.

3.3. Defaults form

The Defaults form is used to change a number of *Setup* variables from their initial default values. These parameters include both values that *Setup* will use in the install and those that only effect the interface.

Network Number

This text item displays the default Sun network address: "192.9.200". If you need to specify a different number, type in another value.

Auto Host Numbering

Setup will assign host numbers automatically for the workstation and any clients that may be added. If you wish to disable this feature, check the "off" option.

Begin Numbering at

If Auto Host Numbering is "Yes", this item specifies the first host number to be assigned. You may change it if you wish to any integer value greater than the default.

Display Units

This choice item selects the units that all memory sizes are displayed in. The default is Megabytes, but you may select Kilobytes, Cylinders, or Sectors as well.

Mail Configuration

This choice item specifies whether you wish the workstation to act as a Server or Client of the mail system.

Preserve Disk State

The default is no. By default, *Setup* will make a file system *(mkfs*(8)) on each partition of type "UNIX". If however, you have partitions containing data that is not affected by installing a new release (such as a partition containing source code), this item tells *Setup* to make a file system only on the partitions *Setup* touches. *Setup* will touch the root partition (/), /usr, and /pub. If more than one CPU-type is being served, there is a /usr and /pub partition for each CPU-type.

3.4. Client form

The Client form is divided into four regions. On the left half of the screen is information about clients; on the right half, "configuration cards". The Client form is also divided from top to bottom: the top displays the clients and configuration cards that *Setup* knows about, and the bottom half of the screen is where you "edit" clients or cards. By "edit", we mean create or change the properties of a client or card much like you can use a text editor to create a file or to change its contents.

Edit Client

The lower left corner of the screen is where you edit clients. The "Edit Client" item is a text item used to change the attributes of an existing client or to create a new client of the server. Type in the name of a client and press return; if the client does not exist, *Setup* will create a new one. If there is a default configuration card (see below), then *Setup* will use its values as the default for the newly created client. If the client already exists, the client's attributes will be displayed and may be modified. The properties of a client are its name, CPU type, location and size of its root partition ("/") and location and size of its swap partition, presence of a 3COM ethernet board, Ethernet address, and Host number.

Edit Card

The client cards on the right can be thought of as "templates" to be applied to clients. The properties of a card are just like a client except for the absense of Ethernet address, and Host number fields.

When *Setup* comes up for an initial installation, a set of read-only cards for typical client configurations is already provided. *Setup* will not allow you to modify these although you may open them to examine their contents. You can, however, create any number of your own client cards to suit your needs. When *Setup* is started in upgrade mode, these cards are once again available for use.

You can designate one client card to be the "default card" which will be applied each time a client is created, or you may apply any card to an existing client. When a card is applied, its values overwrite any existing values in a client unless the special value "Don't Apply" has been chosen.

Default Card

This is where you type in the name of an existing configuration card to make it the current default card. To clear the default, erase the name and hit RETURN .

Apply To

This is where you type in the name of an existing client. The configuration card which is currently open for editing will be applied to this client.

NOTE *The following form items apply to both clients and cards except where noted.*

Close

This button closes the current client or card; this means information about the client or card is no longer visible but not deleted from the system.

Delete This button deletes the client or card that is currently open for editing.

Name This item is filled in when you first start editing a client.

CPU type Here is where you choose the CPU type of the client. For configuration cards, you may also choose "DON'T APPLY" if you do not wish clients' CPU types to be affected.

Root Partition This is where you choose on which hard partition the client's root partition will go. Configuration cards have "FIRST FIT" chosen by default. This is true only for those hard partitions with type ND on the entire disk. Clients to whom this card is applied will have their root partitions placed on the first ND partition with sufficient space. You may designate a specific ND partition or select "DON'T APPLY" if you do not wish clients' root partitions to be affected.

Root Size Here is where you type in the size of the client's root partition. The current Display Units are assumed. You may change these on the Defaults form.

Swap Partition This is where you choose on which hard partition the client's swap partition will go. As with root partition above, "FIRST FIT" and "DON'T APPLY" are also choices for configuration card swap partitions.

Swap Size This is where you type in the size of the swap partition.

3com Board This is where you check yes if the client is using a 3com ethernet board.

Ethernet Address (Clients only) The format of the Ethernet Address is six colon-separated bytes of the form "AA:BB:CC:DD:EE:FF". You must type the colon between each byte.

Host Number (Clients only) This is the Internet host number of the client. If Auto Host Numbering is "Yes", then *Setup* will assign a host number to the client, otherwise you may type an explicit host number. See the Defaults form.

3.5. Optional Software form

This form is used to specify which optional software packages you wish to install. This is a simple form, and consists of a cyclic choice of the architectures served by the workstation and, for each architecture, a list of the optional software packages. You just check off which packages you would like. Note that if you are configuring a server of both 68010 and 68020 clients, you will need to repeat the process for each architecture.

Each package is listed with its size in the default units. As you select a software package to be installed, the size of the /usr disk partition is automatically adjusted. If there is no more space available, then a message will be displayed to that effect. At this point you should visit the Disk form to see how the partitioning is proceeding.

Buttons provide an accelerated means to CLEAR all selections, choose ALL packages, or pick just the COMMON CHOICES. If there is not enough space for all of the packages you request to be installed, *Setup* will select as many as possible and provide a message for the rest.

3.6. Disk Overview and Philosophy

This section describes the philosophy that *Setup* uses to configure disks. Many of the operations that *Setup* supports when changing disks are described below.

General Concepts

There are some general concepts that are fundamental to the way in which *Setup* (and Unix) use disks.

The basic unit of the disk is a *sector*. Sun disk sectors are 512 bytes. Sectors are then grouped into *tracks*. This is the maximum number of sectors that will fit on one "circle" of a disk platter. Tracks in the same location on all platters are grouped together as *cylinders*.

Each disk has a maximum of eight *hard partitions*, by convention named 'a' through 'h'. These hard partitions implement a virtual disk scheme to create separate sections of a physical disk. Disks do not need to use all of the available hard partitions.

Each hard partition has an *offset* and a *size*. Hard partition offsets must be in cylinders, but their sizes may be in either cylinders or sectors. For performance reasons, hard partition sizes are usually also in cylinders. Hard partitions that overlap one another can be created, but overlapping partitions cannot be used by Unix simultaneously.

The configuration of the hard partitions in encoded within the disk *label*. The label is written to special locations on the disk by the *diag*(8) program and can be changed by *Setup*. The label is read from the disk by Unix at boot time to determine the disk configuration. You can see the existing label by using *diag's* `verify` command or the command *dkinfo*(8) in UNIX

Creating or changing disk labels should be done with caution. If the disk label needs to be changed after Unix is installed, all of the information on the disk should be saved before modifying the label. Changing a hard partition's size or offset effectively destroys all of the information that was on that hard partition. See the *Sytem Administration* manual for more information.

The Network Disk (ND) facility in Sun Unix implements a mechanism to divide hard partitions into *soft partitions*. These soft partitions are used to provide diskless clients with Root and Swap partitions. In addition, there are ND public partitions that allow soft partitions to be shared read-only between diskless clients. Public partitions are used to share executable and common configuration files between workstations. ND configuration information is kept in the file /etc/nd.local. This file is analogous a hard partition label for a soft partition, providing soft partition offsets and sizes (for more information see *nd*(8)).

Setup dynamically changes the offsets and sizes of both hard and soft partitions depending on the configuration being installed. Users can also change the partition configurations from *Setup*. One of the actions of *Setup* during installation is to write a new disk label and create a new /etc/nd.local. The *Setup* disk screen presents a "picture" of the hard partitions and allows editing of the hard partitions. ND partitions are created and edited from the *Setup* client screen. (*Setup* has the limitation that soft partitions can only be created and edited in conjunction with a diskless client.)

Partition Usage in *Setup*

Setup has different uses for the hard partitions depending on what type of configuration is being installed. *Setup* requires that a workstation be configured as either standalone or as a file server. When choosing or changing between types, *Setup* changes the use of many of the hard partitions. Care should be taken to choose your type *before* spending time configuring your disks as this information may be lost if you change again later.

There are two common hard partitions that are used on all configurations:

- Root Partition — this is used for the Unix root file system (/) that contains the basic Unix files.
- Swap Partition — this is used for swap space by Unix. You must have a minimum of one swap partition per workstation.

Standalone machines have another hard partition used for the user file system (/usr). This partition contains additional files used by Unix.

Table 3-1 outlines the partitions used when configuring a standalone workstation.

Table 3-1 *Standalone Hard Partition Conventions*

Partition	Convention
a	/ (Root partition)
b	Swap
c	Entire Disk
d	unused
e	unused
f	unused
g	/usr
h	unused

Setup allows users to configure file servers to serve either or both Sun 68010 and 68020 architectures. For each architecture a file server is configured to serve, one hard partition is used for a public file system (/pub) and a second for a user file system (/usr). When a workstation is configured as a server, it gets its common files as if it were a diskless client. Therefore, *Setup* requires that a server must serve its own architecture.

Table 3-2 outlines the partitions used when configuring a file server for diskless clients. Table 3-2 shows the configuration for a server that is going to serve both 68010 and 68020 based clients.

Table 3-2 *Server Hard Partition Conventions*

Partition	Convention
a	/ (Root partition)
b	Swap
c	Entire Disk (ND Partitions)
d	Home Directories
e	/pub (68010)
f	/pub (68020)
g	/usr (68010)
h	/usr (68020)

All of the partitions used by *Setup* in the above tables are considered *special.* When *Setup* configures partitions as special, they are given the correct minimum sizes for their use (such as /usr). *Setup* prevents a user from setting these sizes below the minimum. However, it is up to the user to decide how large the partitions are to be. Described below are some unique restrictions and operations which apply to special partitions.

- The Root partition size or offset cannot be modified from *Setup*. If you need a larger Root partition, you will need to configure it from *diag* by creating a custom disk label (see Appendix F) before running *Setup*.
- The Swap partition offset cannot be modified from *Setup*. Its size may only be made larger than the initial size (See *Appendix F*).
- By convention, the 'c' partition is configured to span the entire disk. When *Setup* is installing a server, it also uses partition 'c' as the first ND partition. This means that the client partitions defined in `/etc/nd.local` will be located on some portion of the 'c' partition. Since the label will still have partition 'c' configured as the entire disk, this is how it will appear on the *Setup* disk screen even though *Setup* is using partition 'c' for an ND partition.

Setup Disk and Hard Partition Attributes

Setup allows hard partitions to be *overlapping*. This means that a label can be created so that hard partitions share locations on the disk. By default, *Setup* does not configure disks with overlapping partitions and they are not recommended.

Setup implements a facility called *floating* for disks. When a disk is floating, hard partition offsets are automatically changed so that hard partitions abut one another; one partition begins where the previous one ended. Floating does not affect partition 'c', which is the entire disk, but does affect the portion of the 'c' partition used for ND partitions. When a disk is floating, a user cannot directly change a hard partition's offset. By default, floating is turned on when *Setup* begins. Floating cannot be turned off on a disk that contains special partitions. Floating is not allowed on a disk that contains overlapping partitions.

When a disk is floating, *Setup* implements a facility to allow one hard partition to take up all of the available space on the disk not explicitly used by other hard partitions. This so-called *free space hog* is given all of the unused space, and will shrink in size as other hard partitions grow. A typical designation for the free hog is the hard partition on a server that contains the home directories.

Setup provides a facility to "move" hard partitions from one to another. This is not a general facility, but a means to move some special partitions (`/usr's` and `/pub's`) to another hard partition. This allows users to place various special partitions on different hard partitions or different disks. The destination partition in a move must be of type "free" (see below).

Setup has an attribute for each disk that allows a user to decide if sizes should be rounded to cylinder boundaries (hard partition *offsets* are always rounded to cylinder boundaries). By default, cylinder rounding is turned on. This means that when sizes are entered, they are automatically rounded up to the next cylinder boundary for that disk. For optimal performance, it is a good idea to round partitions to cylinder boundaries.

Setup Hard Partition Types

Setup allows users to create and "edit" hard partitions. There are 5 unique types of hard partitions that a user can configure. Special partitions (see above) are automatically set to the correct type by *Setup*.

- **Free** partitions are those that are unused by *Setup*. They can have an offset and a size, and will be configured into the label. When a partition's type is changed to free, its size is zeroed.
- **Swap** partitions are those used for swap space by Unix. The correct entry is generated for the file system table, but a user must still configure a kernel to know about the additional swap space (see Chapter 8, *Configuring the System Kernel*).
- **Unix** partitions are those used as Unix file systems. They have an additional attribute, a mount point, which is the name of the directory that the file system should be mounted on. During installation, *Setup* will create a Unix file system on these partitions, create the required directories to mount the file system, and generate the correct entry for the file system table (see *fstab*(5)).
- **ND** partitions are those used by *Setup* to provide ND partitions to diskless clients. When a partition's type is changed to ND, its size is set to zero. An ND partition's size can only be modified when soft partitions are assigned to it from the *Setup* client screen. In addition, an ND partition's type cannot be changed while it contains any soft partitions used by diskless clients.
- **Other** partitions are the same as free. They are a means for a user to create a partition that *Setup* will consider to be used for other purposes.

3.7. Disk form

The Disk form presents you with all the disks connected to the machine being configured and allows you to individually edit their hard partitions. There are two distinct levels of editing on this form: editing a disk, and editing a particular hard partition of that disk.

Edit Disk

Initially "None", this cyclic choice item opens a disk for edit. With a disk open, its hard partitions "a" through "h" are displayed. On the left, the type and size of each partition is listed. On the right is a graphic display of the relative positions and sizes of the partitions. The total size and amount of free space remaining on the disk are given in the upper left. In addition, there are several choice items on the upper right which alter disk parameters.

Round to Cylinders

If "Yes", all partition sizes entered will be rounded up to the next cylinder boundary.

Overlapping Allowed

Initially "No", meaning that *Setup* will check for overlapping partitions as a result of your actions. If you wish to allow partitions to overlap you may do so here by choosing "Yes".

Float

If set to "Yes", the partitions of the disk are automatically "floated" to avoid overlap when the size of another partition changes. This only applies if overlapping is not allowed.

Free Space Hog

This cyclic choice item identifies the partition that will consume any additional free space when the install is performed. The current Free Space Hog, if any, is highlighted in the partition display. This only applies if the disk partitions are being floated.

Edit Hard Partition

Initially "None", this cyclic choice item opens a hard partition of the disk for edit. Once open, the the parameters of a partition are displayed for editing.

Offset

This text item displays the partition's offset in the current display units. If you type in a new value, the current display units are assumed.

Size

This text item displays the partition's size in the current display units. If you type in a new value, the current display units are assumed.

Type

This cyclic choice item specifies the type of the hard partition from among "Free", "Root", "Swap", "ND", and "Unix".

Mount Point

If the partition is of type "Unix", this text item can be used to specify the file system mount point.

Move To

For partitions of type "UNIX" with special designation, such as /pub, the option to move this meaning to another partition is provided. Select the destination partition by cycling through the "To:" item. The new partition must be of type "FREE" and must already have enough space allocated or be able to float to the appropriate size to accomodate the move.

Move It This button commits the move operation once the destination has been selected.

3.8. Modes of Use

Initial Installation

Setup is used during initial installation of Sun Unix operating system distributions. This mode of *Setup* is usually used for new installations or when machines are being installed from the very start.

Prior to installing Sun Unix on any workstation, you should take a moment and decide how the disks are to be layed out. What partitions might be needed later for NFS partitions, how much space to allocate for future expansion, etc.

For machines that are being installed as file servers for diskless workstations, there is some addtional information that you should gather before beginning to use *Setup*. It is a good idea to take some time and plan out how you are going to distribute the available disk space between the file server and the diskless clients before beginning the installation. You should also obtain the Ethernet address of all of the clients before beginning, *Setup* will ask you for this information during the installation. You also might plan for expansion and allocate partitions for future client machines.

NOTE *If you wish to install UNIX on a tapeless workstation, see Appendix B of this manual.*

Machines that are file servers need to back up both clients and the servers files.

Demonstration

This mode of *Setup* can be used as a "dry run" or to demonstrate the user interface without affecting the installation.

3.9. Initiating *Setup*

You are now ready to begin using *Setup*. Type in the following next to your prompt:

```
# setup
```

```
# setup
```

Answer the following questions

You will now see the following questions appear in menu format on your screen. Your answers to these questions, will determine which of the interfaces *Setup* will be using to install Unix on your system. For all menus respond to the ">>" with the corresponding number of the menu item.

The first question asked is the mode of use of *Setup*.

```
Are you running Setup to:
        1) Install a major Sun Unix release
        2) Upgrade between major Sun Unix releases
        3) Demonstrate Setup
>>
```

The next question is to determine the type of interface to be used. Note that the cursor addressable interface can be used within a shelltool under SunWindows.

```
Will you be running Setup from a:
        1) Sun bit mapped display device
        2) cursor addressable terminal (TTY)
>>
```

If you have selected the TTY interface for cursor addressable terminals, *Setup* asks for the terminal type.

```
Select your terminal type:
        1) Televideo 925
        2) Wyse Model 50
        3) Sun Workstation
        4) Other
>>
```

If you select "Other", the name of the terminal must correspond to a name in the *termcap*(5) database.

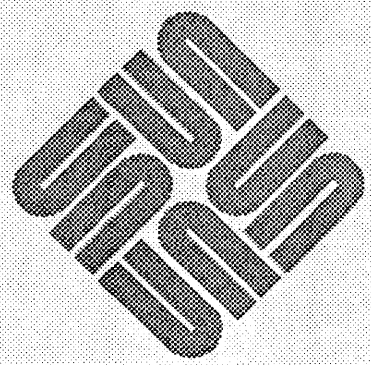
```
Enter the terminal type (your terminal type must be in /etc/te
        >>
```

Setup begins running the interface selected.

If you have selected the tty interface, proceed to Chapter 4 of this manual. If you have selected the bit-map interface, proceed to Chapter 5.

4

How To Use The Terminal Interface



4

How To Use The Terminal Interface

4.1. Introduction

This chapter describes the use of the terminal interface to *Setup*. The first section of this chapter discusses how to enter information by interacting with the *Setup* form items. The next sections describe how to move the cursor from item to item, and the last section has a number of images that showing terminal screens after entering example data.

Keyboard Basics

Before getting started, we briefly describe how to use a computer terminal keyboard; if you are familiar with computer keyboards you can skip over this paragraph. Since there are a number of items on the screen at one time, we use the terminal's cursor to indicate the current item; that is, the one we are currently interested in. In order to move the cursor from one item to another, you must press a "control key" such as "control-F" (a more detailed discussion of moving the cursor is given later on in the document). A control key sends a special character to the computer when you hold down the control key and a regular alphabetic key. You MUST hold down the control key while you press the alphabetic key just like you must hold down the shift key to get an upper case letter. The control key is found on the left side of the keyboard near the "A" key and is usually labeled "CTRL". In this document, we represent a control key like: [CTRL-F] which means to press the "F" key while holding down the control key.

4.2. Item Interaction

In order to enter information into *Setup*, you will move the cursor to an item (how to do this is described below) and then interact with the item. Each type of item has, on its left side, a label which describes what information the item is requesting. To the right of the label, is the value of the item. How you change the value of the item is described below.

Text Item Interaction

A text item has a label and a text field. The text field is the area where the text that is typed into the item is displayed. When the cursor is at a text item it appears at the last character in the text field. As you type, the characters appear in the text field and the cursor move to the left. If you type more characters than the text field allows, the characters will scroll to the left. The terminal bell will sound if you type more characters than the item has room for.

All alpha-numeric keys are legal and your character-, word- and line- erase characters work in a text field just like in the shell. (Note: the default value of character erase is [DEL] ; for word erase, it is [CTRL-W] and to erase the whole text

field, it is [CTRL-U]). You MUST press [RETURN] in order to send the string you typed to *Setup*. Here is a sample text item into which a user has typed "abcdefgh":

```
text item label: abcdefgh
```

Choice Item Interaction

Choice items present a list of choices; you can pick one value from the list to be your "choice". Here is an sample choice item:

```
choice item label:  [X] choice1, [ ] choice2, [ ] choice3
```

The choices that you can choose from in this example are "choice1", "choice2", and "choice3". The current choice is denoted by the "X" in the box next to the choice. To change the current choice, you first use the [Space Bar] to move the cursor forward to the next choice; your character delete key, [DEL] or [BACK SPACE] , moves the cursor backward to the previous choice. When the cursor indicates the choice you want, select it by pressing [X] and the box next to the choice will become marked with an "X".

Choice items can also appear in another form than the example given above. Here you can see a "cyclic" choice item:

```
choice item label:  [ ] choice2
```

Unlike the previous example of a choice item, only one choice is visible at a time in a "cyclic" choice item. When you press the [Space Bar] or your character delete key, a new choice will overwrite the previous choice. Note that in this example, "choice2" is not the current choice.

Toggle Item Interaction

Toggle items are like choice items in that they have a list of elements from which you can pick. However, unlike choice items, toggle items allow you to pick more that one choice. You are able to "toggle" the elements in the list on and off in order to indicate which ones you want. In this example, "toggle1" and "toggle3" have been chosen:

```
toggle item label:  <X> toggle1, < > toggle2, <X> toggle3
```

As with choice items, use the [Space Bar] or your character delete key to move the cursor from one element to another. Press [X] to change the state of an element: to turn it from off to on or from on to off.

Button Item Interaction

Button items cause *Setup* to perform an action. Buttons that appear in the control region of the Setup screen are used to display forms or to perform the installation. A button item looks like:

```
BUTTON LABEL
```

The cursor appears to the left of the first letter of a button. Hit the [X] key to "press" the button.

What the Parentheses Mean

In *Setup*, you will see some items surrounded by parentheses; for example: "`(Workstation Name:)`." This indicates that the item is in need of attention (*Setup* needs the information in order to do its job correctly) or the value you gave the item is illegal. In either case, you must enter the correct value to the item before pressing the "`EXECUTE SETUP`" button.

Parenthesis also appear around the button in the control region which represents the form that is currently displayed.

4.3. Moving the Cursor

Setup will move the cursor from the current item to the next item when you press [RETURN] if the value you give the current item is legal. However, there are times when you will want to go to a particular item in order to change its value. In addition to [RETURN] , two keys move the cursor from one item to another: [CTRL-F] moves the cursor forward to the next item and [CTRL-B] goes back to the previous item.

Table 4-1 *Control Key Reference*

Setup Terminal Interface Control Keys		
Cursor Movement	Next/Forward	Previous/Backward
Region	[CTRL-N]	[CTRL-P]
Item/Scrolling	[RETURN] or [CTRL-F]	[CTRL-B]
Choice/Toggle	[RETURN] or [Space Bar]	character delete key
Select Item	[X] or [x]	
Enter Text	[RETURN]	
Redraw	[CTRL-R]	

When *Setup* starts up, you will see 3 regions on the screen; see Figure 1. The top region is the "control panel". It consists of buttons that bring up other forms which you need to fill out. Below the control panel, is the message region which displays error messages and prompts. To move between regions, use [CTRL-N] to go to the next region and [CTRL-P] to go to the previous region.

Should the terminal screen need to refreshed, press [CTRL-R] to redraw the screen.

Scrolling

Messages appearing in the message region may be scrolled by placing the cursor in the message area and moving the cursor up and down via [CTRL-F] or [RETURN] and [CTRL-B] . The messages will scroll one line at a time.

4.4. Setup Forms

This section has pictures of each of the forms in *Setup* and gives advice on how to fill out each form using this interface.

Figure 4-1 *Initial Setup Screen*

```
(Workstation)        Clients              Software             Disks
 Defaults            Execute-Setup        Reboot               Quit
-------------------------------------------------------------------------------

-------------------------------------------------------------------------------
 Workstation Name:

(Workstation Type:)           [X] None, [ ] Standalone, [ ] File Server


 Tape Device:                 [X] 1/4" SCSI (st0)
 Tape Location:               [X] Local



 Ethernet Interface:          [X] Sun Intel (ie0)
     Host Number:             1
     YP type:                 [X] None
```

Figure 1 shows what your terminal screen will look like after you start *Setup* for the first time. The initial form displayed is the workstation form.

The Workstation Form

Figure 4-2 *The Workstation Form*

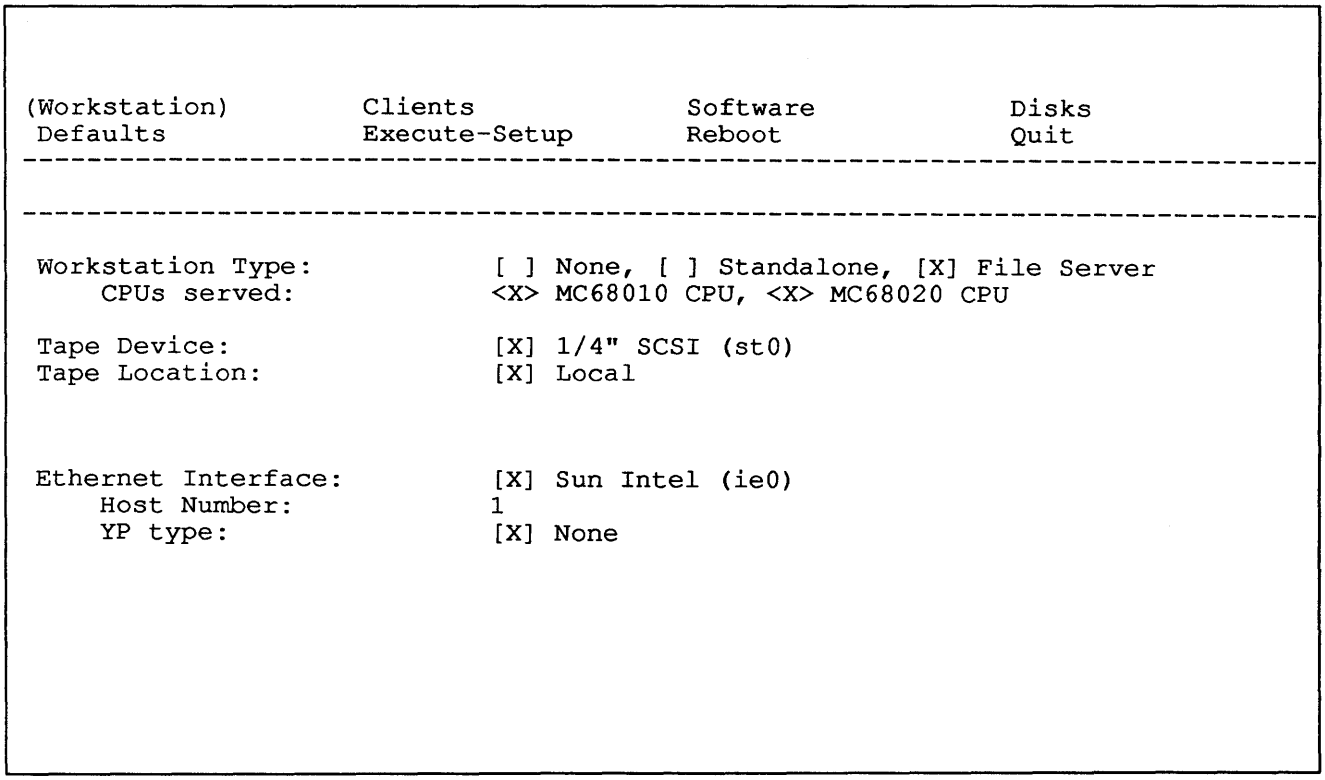
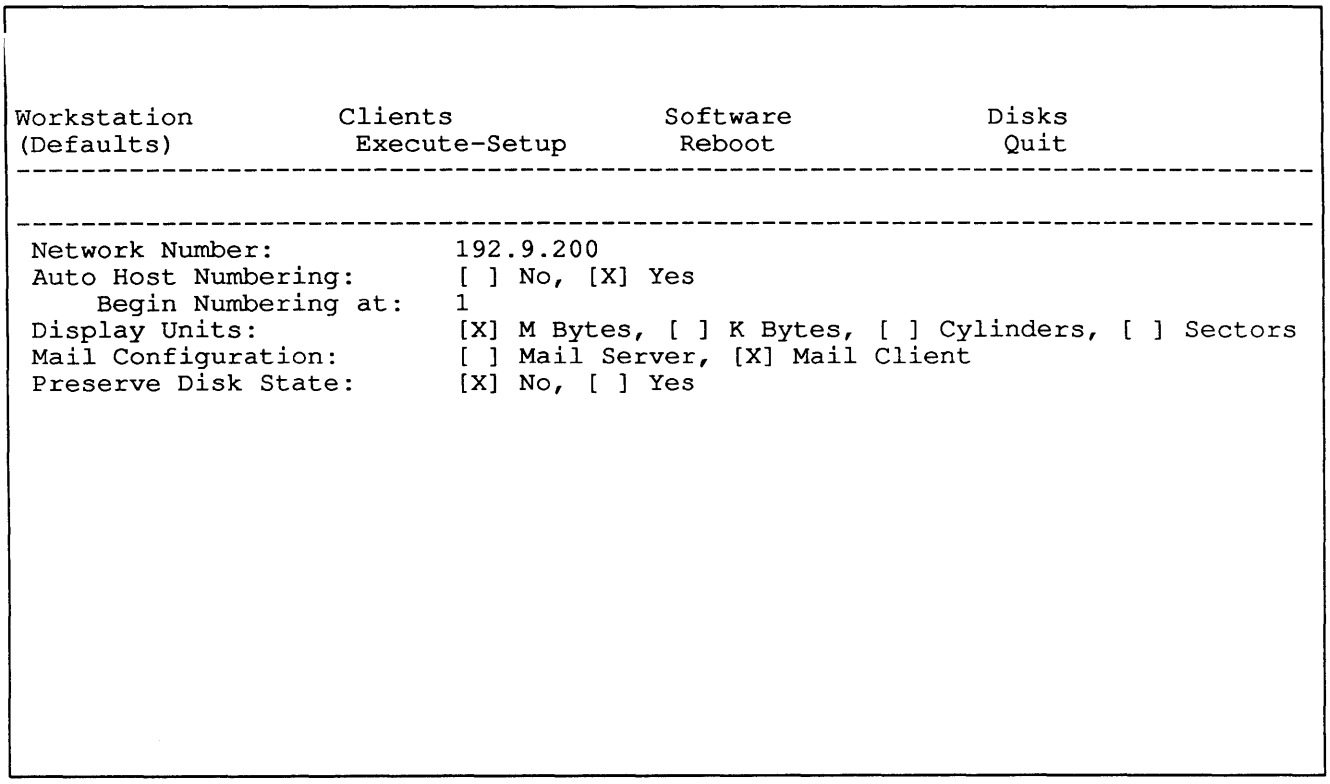


Figure 2 shows the Workstation form for a server. If you compare Figure 1 and Figure 2, you will see that Figure 2 has an extra item displayed below "Workstation Type". This item (the CPU types served by the workstation) only appears if you set your Workstation Type to be a File Server.

Defaults Form

Figure 4-3 *The Defaults Form*



The Defaults form presents you with a number of control parameters. Each item is a enumerated choice item. Simply select the settings you prefer by typing [X] while the cursor is placed at the appropriate choice.

Client Form

Figure 4-4 *Client Form*

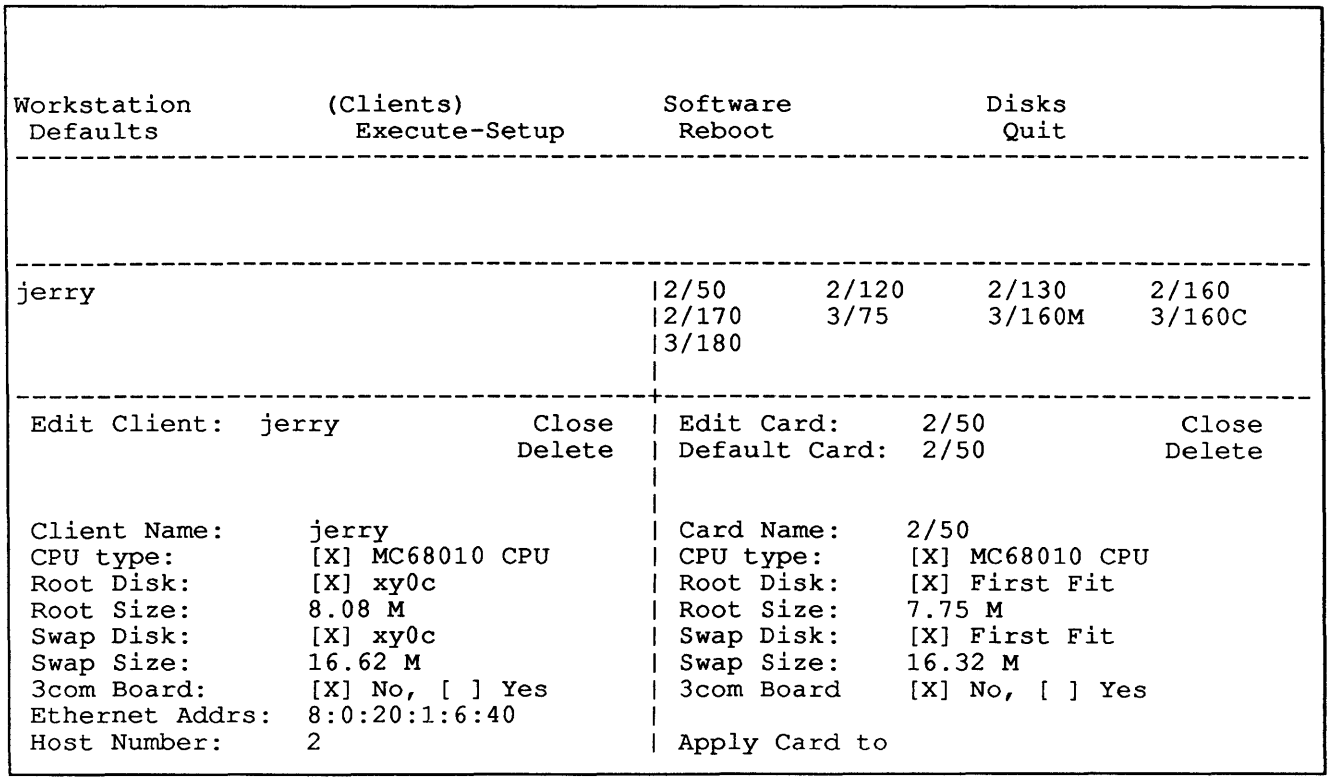


Figure 3 shows the Client form on which you enter information about the diskless clients of a server. There are 4 regions (in addition to the message region and the control region) that you might want to put the cursor in. The 4 regions are (from left to right, top to bottom), the list of clients, the list of configuration cards, the client editor and the configuration card editor. The cursor travels through these regions in that order.

Optional Software Form

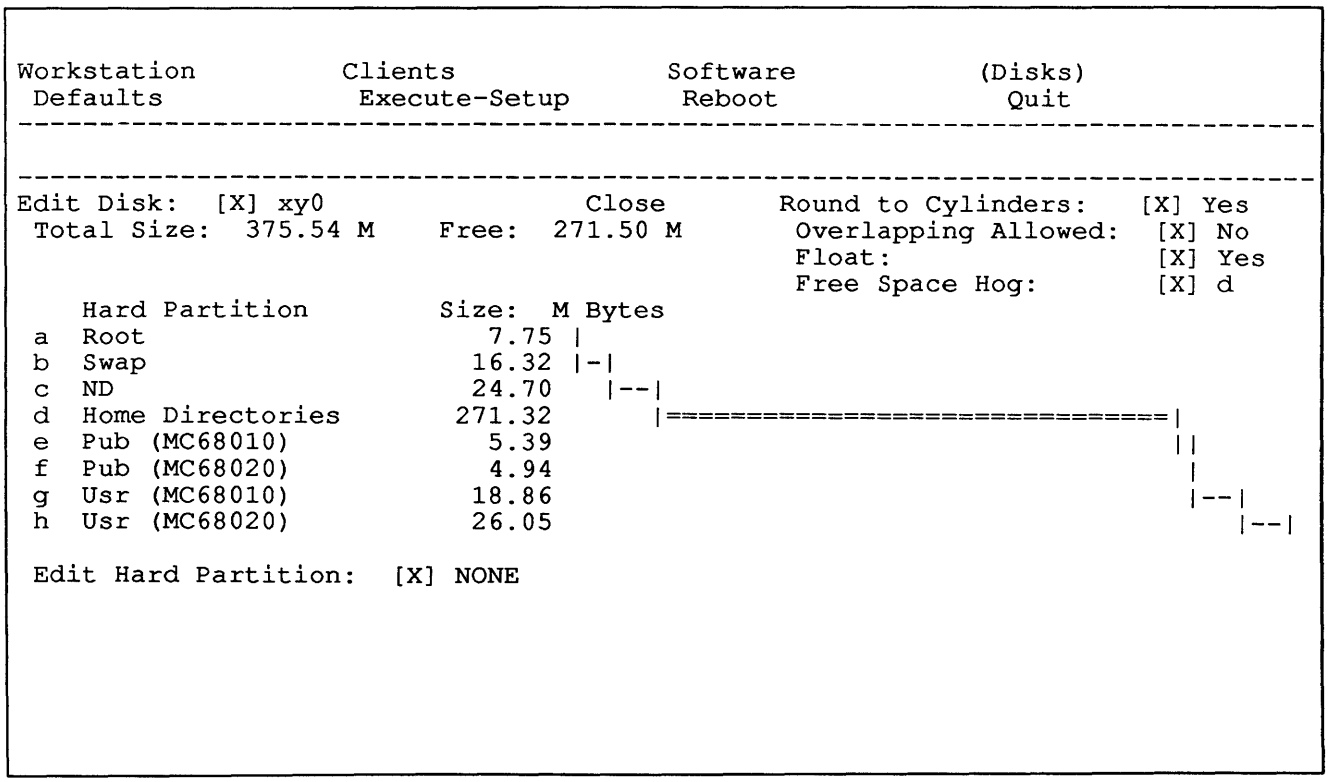
Figure 4-5 *The Optional Software Form*

```
Workstation          Clients             (Software)            Disks
 Defaults             Execute-Setup        Reboot               Quit
-------------------------------------------------------------------------------


-------------------------------------------------------------------------------
 Optional Software for:  [ ] MC68010 CPU, [X] MC68020 CPU
 Clear                    All                  Common Choices

< > Manual Pages (4.52 M)
< > Games (3.62 M)
< > Demonstration Programs (2.39 M)
< > Networking tools and programs (2.73 M)
<X> Debugging tools (1.36 M)
< > Text Processing tools (0.78 M)
<X> Setup tools (0.98 M)
<X> System V programs and libraries (0.98 M)
<X> Standalone Diagnostics (0.00 M)
< > Fortran Compiler & Libraries (1.10 M)
<X> User Level Diagnostics (1.76 M)
< > SunCore & CGI Libraries (2.98 M)
< > Pascal Interpreter & Compiler (1.11 M)
< > Profiled Libraries (0.94 M)
< > SunView and Demo Program source (0.46 M)
< > SunView User Programs (1.60 M)
< > SunView Programmers Files(2.43 M)
< > uucp programs (0.56 M)
< > Versatec Printer Software (6.53 M)
```

The Optional Software screen is shown in Figure 4. You will need to move the cursor from one software package to another and indicate which packages you would like by pressing [X] .

Disks Form

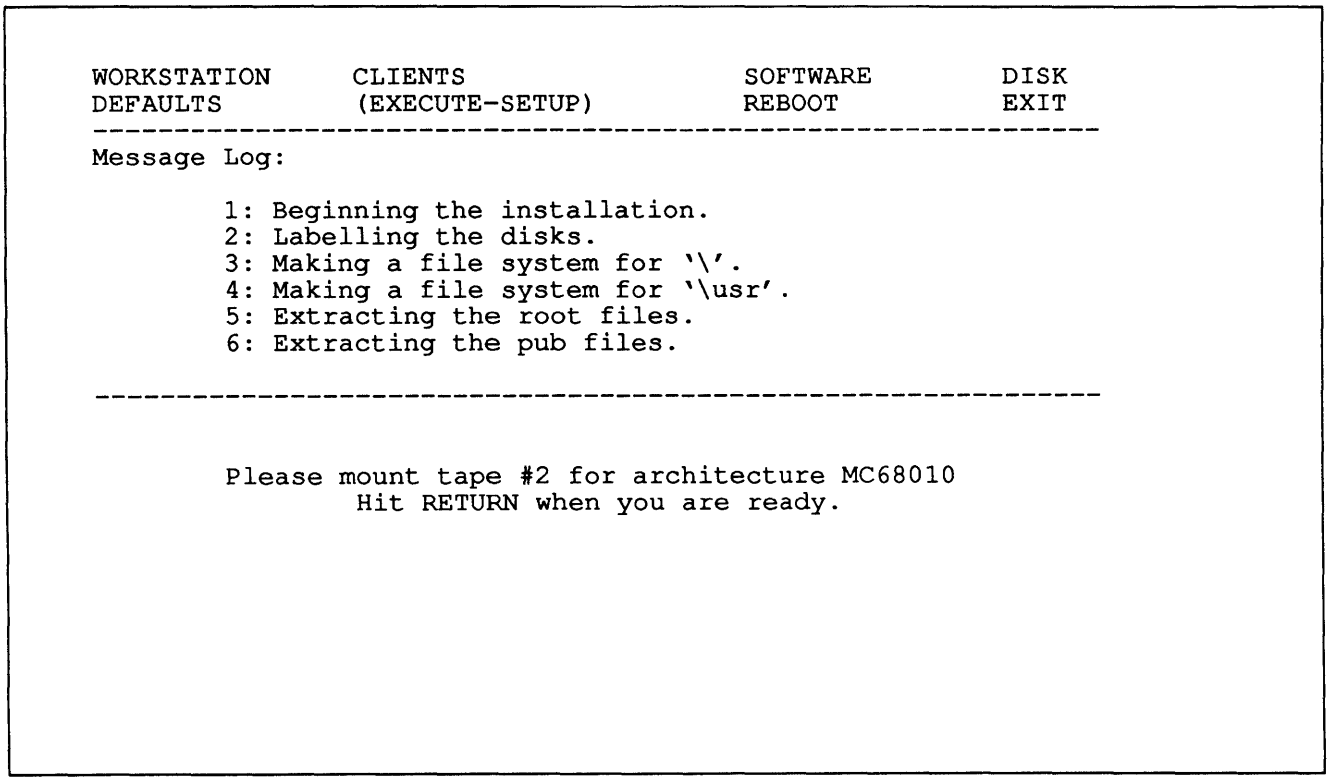
Figure 4-6 *The Disks Form*



The top part of the Disks form (Figure 5) is where you choose which disk you want to edit. To the right, you can tell *Setup* if you want it to do overlap checking or float the hard partitions. In the center of the screen is a display of the hard partitions of the disk which is open for edit. At the bottom of the screen is the area in which you edit a particular hard partition.

Executing *Setup*

Figure 4-7 *Executing Setup*

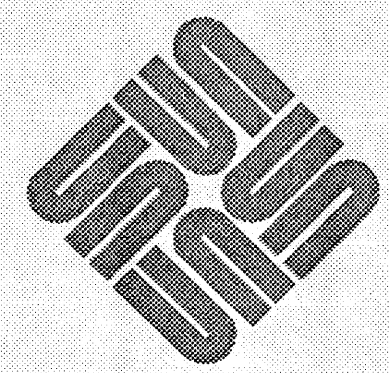


In the control region is a button labeled "EXECUTE SETUP", when you press this button, *Setup* will begin the process of installing Unix on your workstation. Figure 6 shows what the screen looks like after you have pressed this button. Note that the control region is no longer visible and the message area has become larger. In the area below, *Setup* will ask you to do something (like install a new tape). *Setup* will wait for you to preform its request; when you are done, press RETURN .

Chapter 6, *Executing Setup*, explains in more detail what happens during the course of Unix installation.

5

How To Use The Bit-Mapped Interface



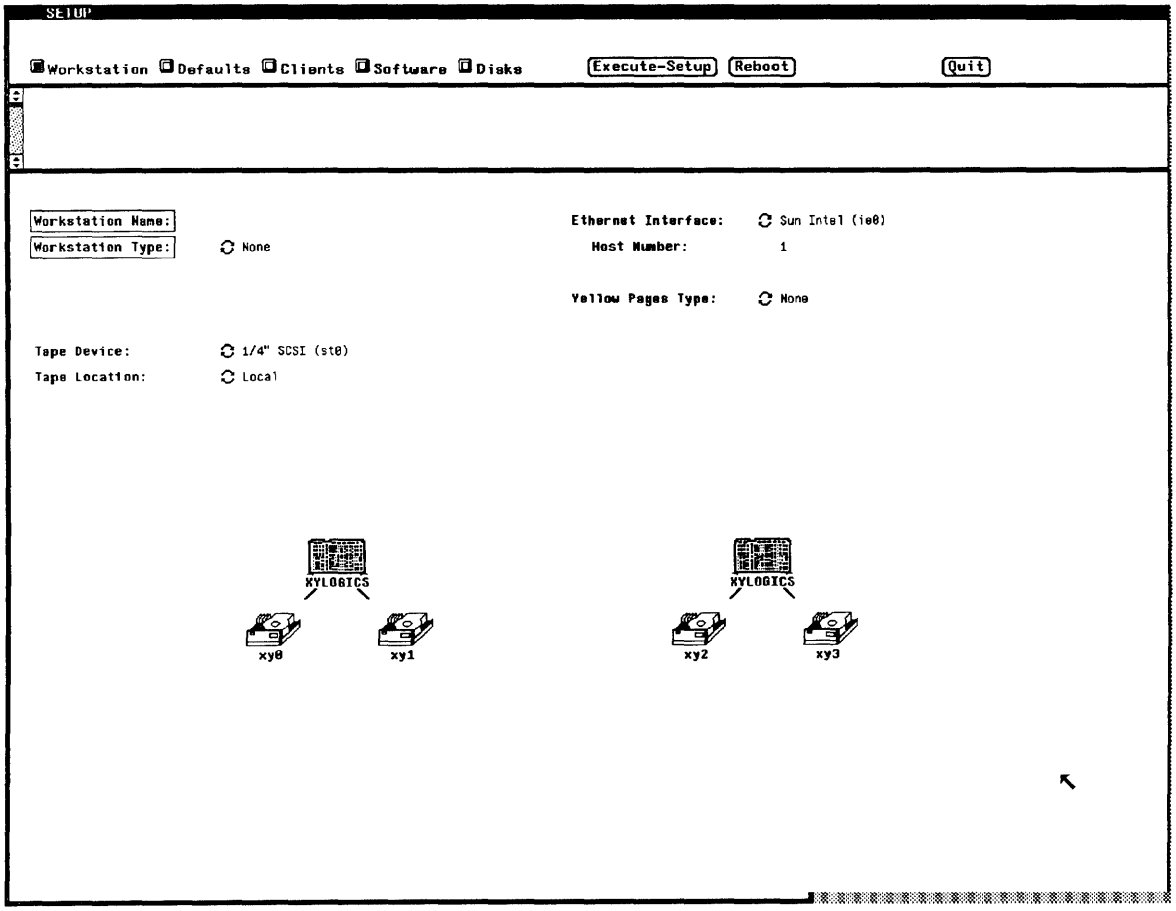
5

How To Use The Bit-Mapped Interface

5.1. Introduction

This section of the document explains how to use the mouse and keyboard to interact with the window version of the *Setup* editor.

Figure 5-1 *Setup Screen Layout*



5.2. *Setup* Screen Layout

After selecting the bit-mapped version of the *Setup* editor, the display will be divided into 3 regions (see Figure 5-1). The top region is the control region. It consists of buttons that switch between forms and initiate execution once the forms have all been filled in. Below it is a scrollable text area which displays error and console messages. The rest of the screen displays the current form.

5.3. Interacting With The *Setup* Editor

The bit-mapped interface to *Setup* allows you to use the mouse pointing device as well as the keyboard to input configuration data. You will therefore see two indications of *input focus* on the display.

Keyboard Input

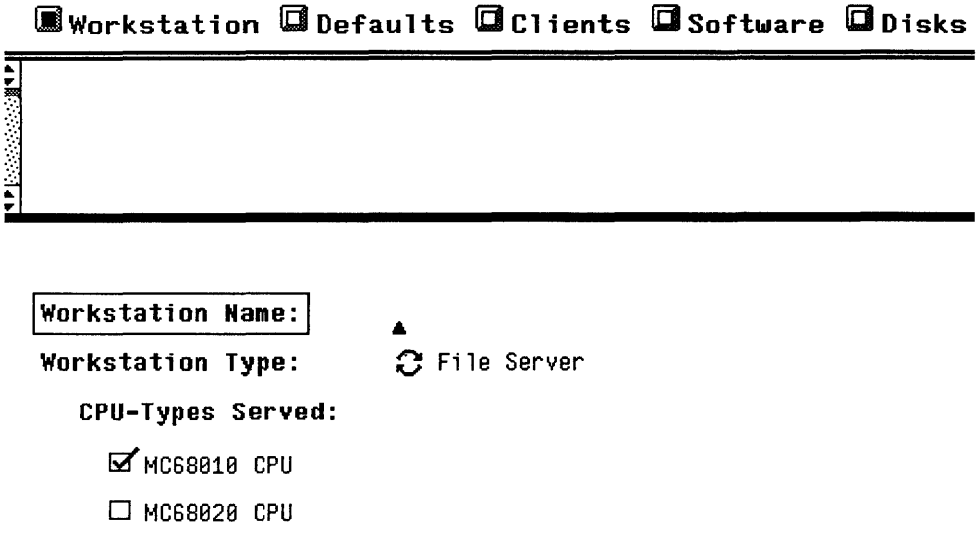
The focus of keyboard input is shown by the blinking *caret*. Figure 5-1 shows that it is initially positioned in the "Workstation Name" field. The caret indicates that any characters typed on the keyboard will be directed to this field. The caret is always placed in a text field.

Mouse Input

The mouse is a pointing device which allows quick access to any input item on the display. Move the mouse on its pad and note that the arrow-shaped *cursor* on the display tracks your movements. This arrow, or "mouse-cursor", indicates the input focus of the mouse. Three buttons on the mouse are used to provide input. They are referred to by position as the left, middle, and right mouse-buttons.

In general, use of the mouse involves simply pointing to the item of interest and pushing one of two buttons. The left mouse-button is used to *select* an item, and the right mouse-button will produce a pop-up menu where appropriate. To point to a specific input item, you must move the mouse to position the mouse-cursor over either the item's label, or its input field. We will now examine how to interact with each of the four input item types: text, choice, toggle, and button.

Figure 5-2 *Setup Input Item Types*



Text Item Interaction

As previously discussed, keyboard input is directed to whichever text item currently has the blinking "caret". *Setup* will endeavor to place the caret at the next text item that should be filled in, but you may locate it yourself. Do this by clicking the left mouse-button over the text item you want to type into. Thus, "selection" with respect to a text item places the caret at that item. When typing into a text item, the character-, word- and line- erase characters are [DEL] , [CTRL-W] , and [CTRL-U] by default. If there is already text in the input field, you must first type [CTRL-U] to erase it. If you have to type more letters than *Setup* has room for, the text will scroll to the left and a left arrow will appear. The field will invert if you have filled up *Setup*'s buffer area; you cannot type any more characters.

Note that you *must* type [RETURN] to enter the string. In Figure 5-2, "Workstation Name" labels the first example of a text item.

Choice Item Interaction

Choice items cycle through the possible options when selected with the left mouse-button. Holding down the [SHIFT] key while you select will cycle backwards through the choices. If you press and hold the right mouse-button over a choice item a menu will "pop-up". While still holding the right-button down, moving the cursor through the menu items inverts the choices. When you release the right button, the currently inverted choice is selected. Releasing the button while the cursor is outside the menu will cause no choice to be made. Menus are provided for "direct access" selection rather than cycling through the options with the left mouse-button. "CPU-Type" in Figure 5-2 is a choice item. The two semi-circular arrows are designed to quickly differentiate a cyclic choice item from a text item.

Toggle Item Interaction

Toggle items present a checklist of elements. You can "toggle" each item by selecting it with the left mouse-button. Toggle items, like choice items, have an associated pop-up menu. In Figure 5-2 "File Server" has been selected as the "Workstation Type" so a toggle item is displayed for "CPU-Types Served".

Button Item Interaction

To "press" a button, select it by pointing to it with the mouse cursor and clicking the left mouse-button. The button's image will invert to indicate that it has been selected, and remain shaded until the action it invokes is complete.

Scrolling

Messages appearing in the text region may be scrolled by moving the mouse-cursor into the scroll-bar on the left and clicking the left button to scroll up, and the right button to scroll down. When the left button is used, the line of text next to it is scrolled to the top of the window. The right button moves the top line of text down to the position of the cursor.

Navigation and Execution

The "buttons" on the control panel are used to randomly access the different *Setup* forms, and to execute the final configuration once the forms are completed. The form currently being displayed is indicated by the button with the reverse-video image.

Items Needing Attention

Input items which are in error or in need of attention are displayed with a box around their label. The *Setup* editor will guide you through the necessary fields by means of this highlighting. You are of course free to enter input in any field. However, these items are highlighted because without this information configuration of your system will not be possible. Before pressing the "EXECUTE-SETUP" button, all such items must be filled in or corrected.

For more information on interacting with Sun Windows, see *Beginner's Guide to Windows and Window-Based Tools*. This guide discusses the use of the keyboard, mouse, menus, scrollbars, and selections in greater detail.

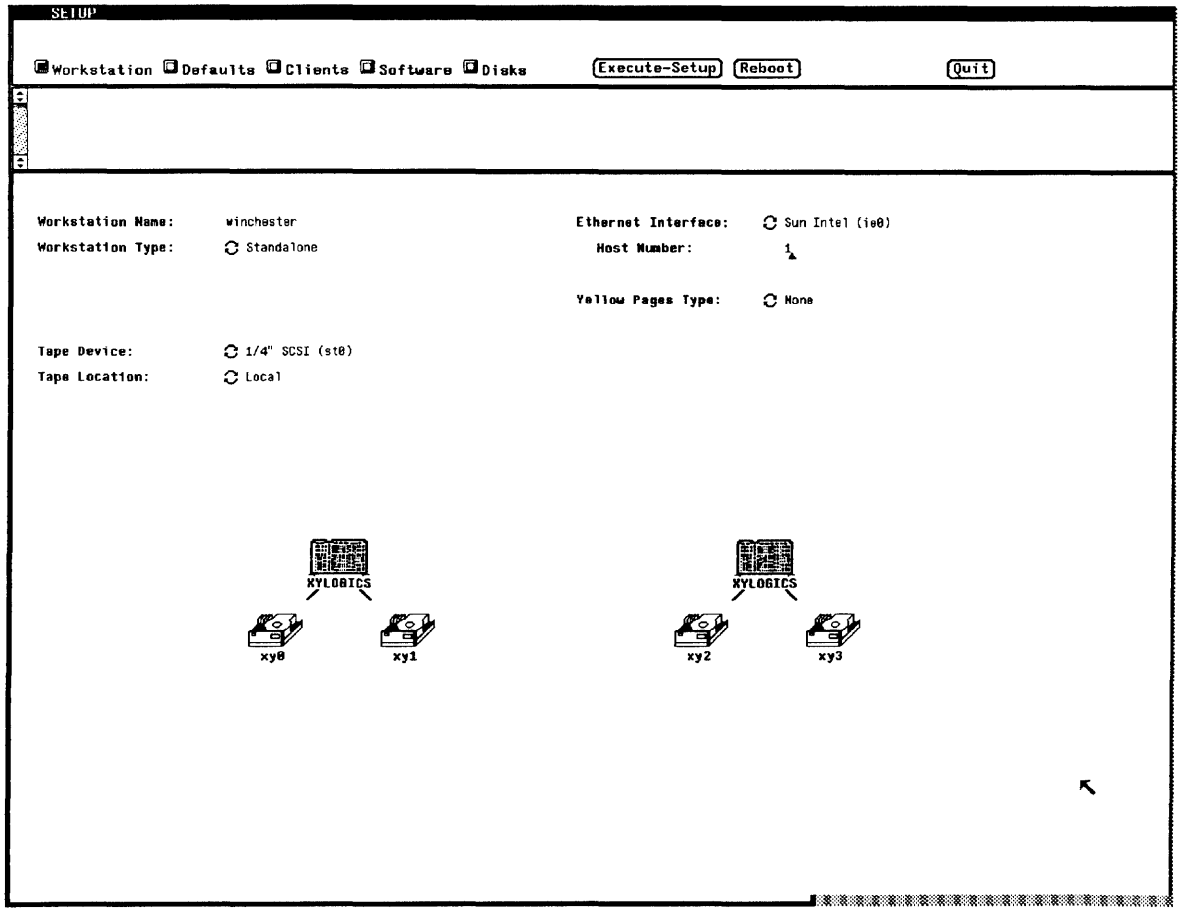
5.4. The *Setup* Forms

As was discussed in the *Setup* Overview chapter, editing your *Setup* configuration consists of filling out forms. The general focus of each has already been presented; here we will discuss each briefly and mention some "accelerated" means of input.

Workstation Form

The Workstation form requests information about the workstation that is being configured and its network connection, if any. The pictures of the controller board(s) and disk(s) indicate hardware attached to the machine being configured. The controller images indicate the grouping of the disks.

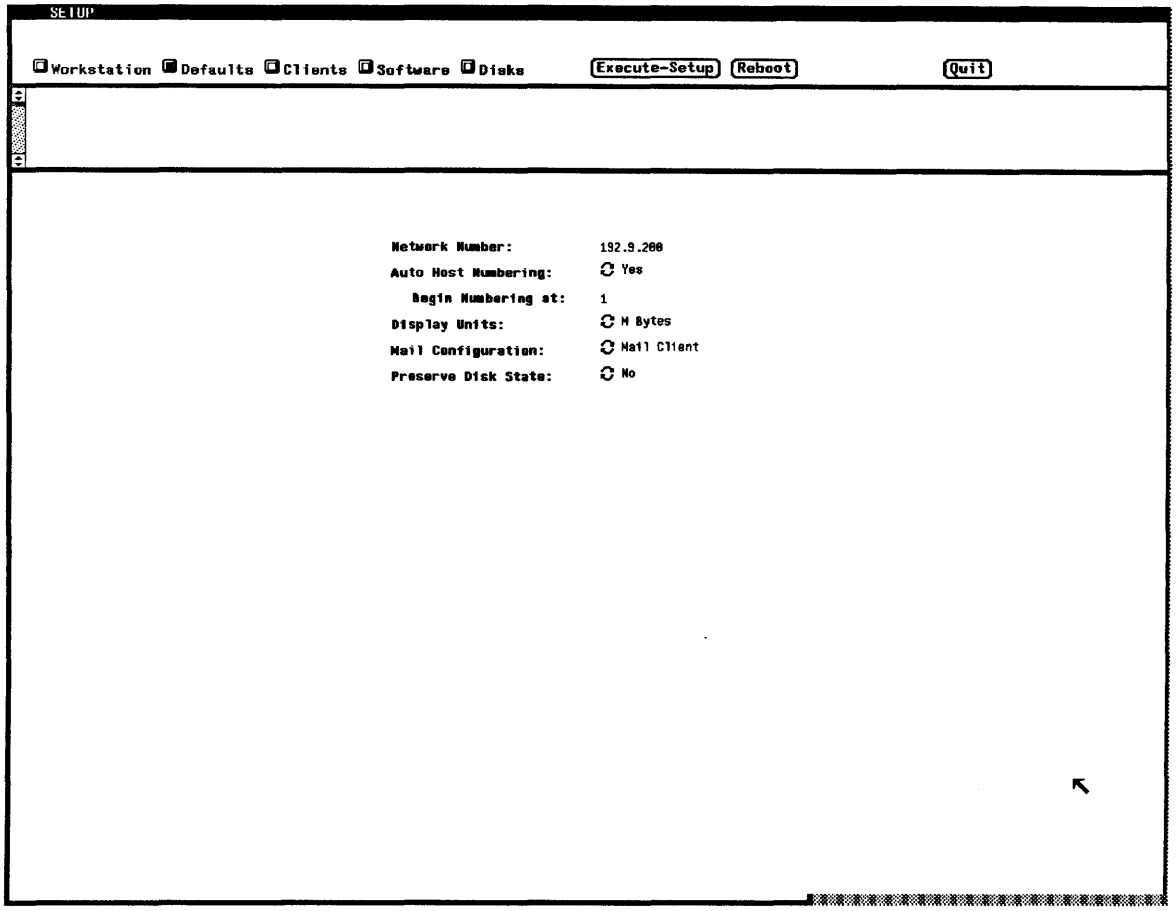
Figure 5-3 *The Workstation Form*



Defaults Form

The Defaults form presents you with a number of control parameters. Each item is a enumerated choice item. Simply check the settings you prefer by selecting them with the left mouse-button.

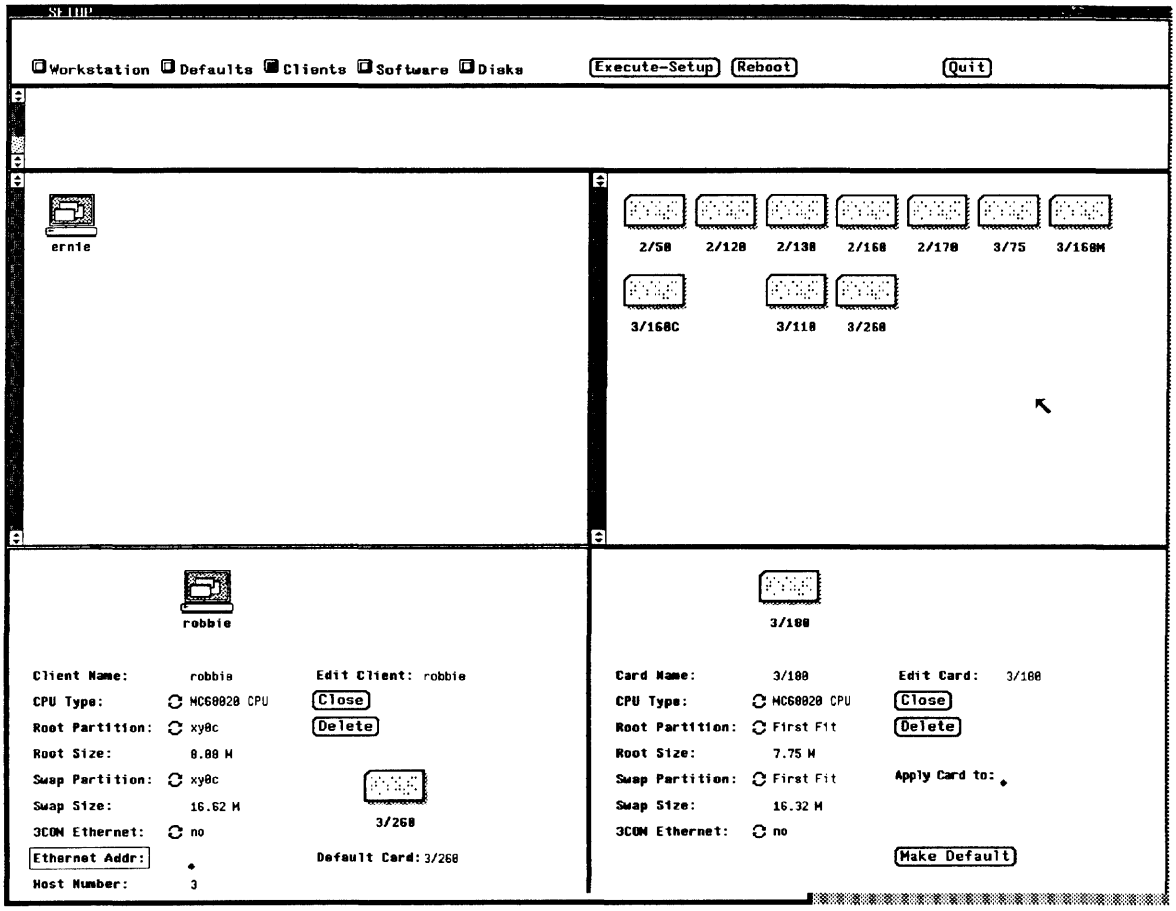
Figure 5-4 *The Defaults Form*



Client Form

The Client Form is used for entering information about the clients of a file server. The images of clients and configuration cards are actually buttons. They provide an accelerated means of opening a client or card for edit. These behave like any other button: position the mouse cursor over the image and select it with the left mouse button. Selecting an image in the top half of the screen will open that particular client or card for edit. This avoids having to type into the "Edit" field. Once open, clicking the image again is equivalent to hitting the "CLOSE" button.

Figure 5-5 *The Client Form*



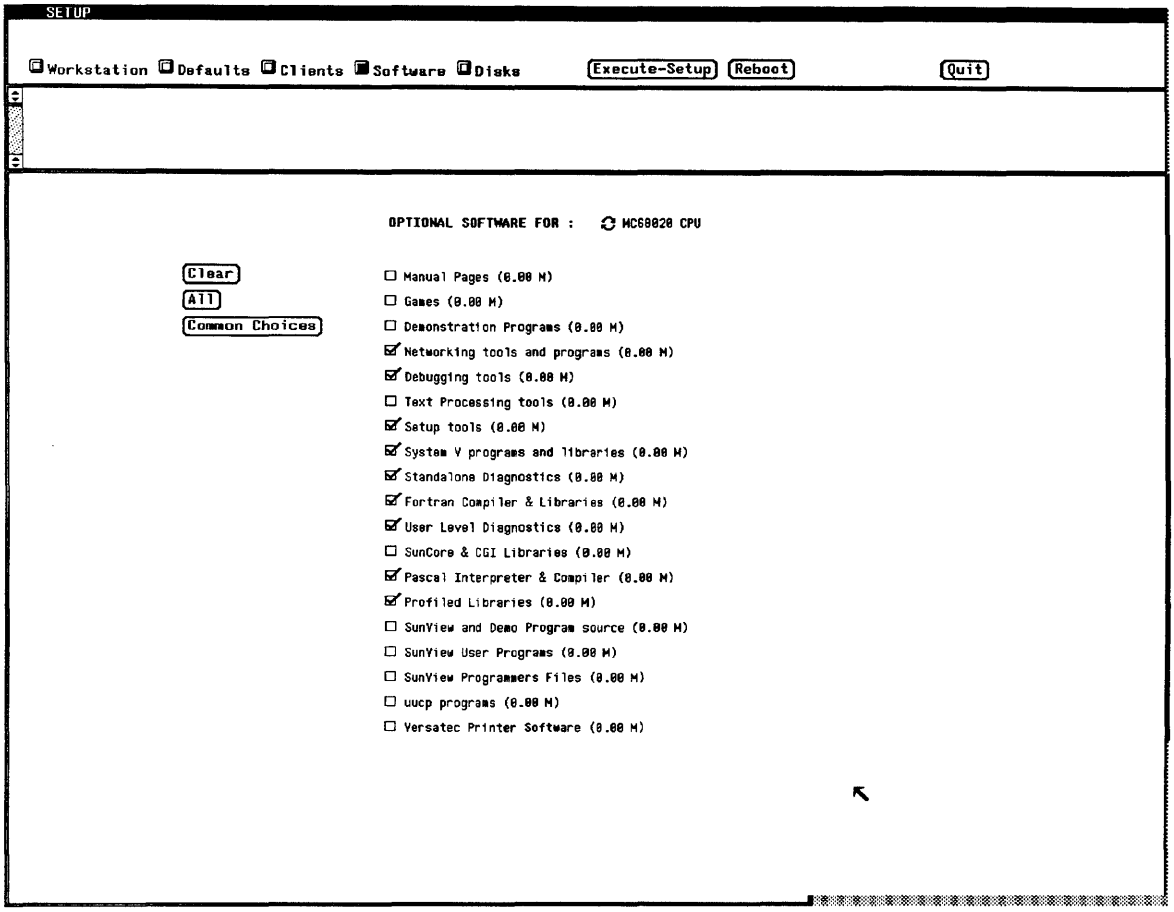
Optional Software Form

The Optional Software Form is used to specify which optional software packages you wish to install.

NOTE *Optional software Mbytes sizes vary with each release. These numbers are only examples. See your system for actual sizes.*

For each choice provided in the "OPTIONAL SOFTWARE FOR" item, select the software packages you wish to have installed. If you are a 68020 system, you will have to "click" the Optional Software item for the proper architecture.

Figure 5-6 *The Optional Software Form*



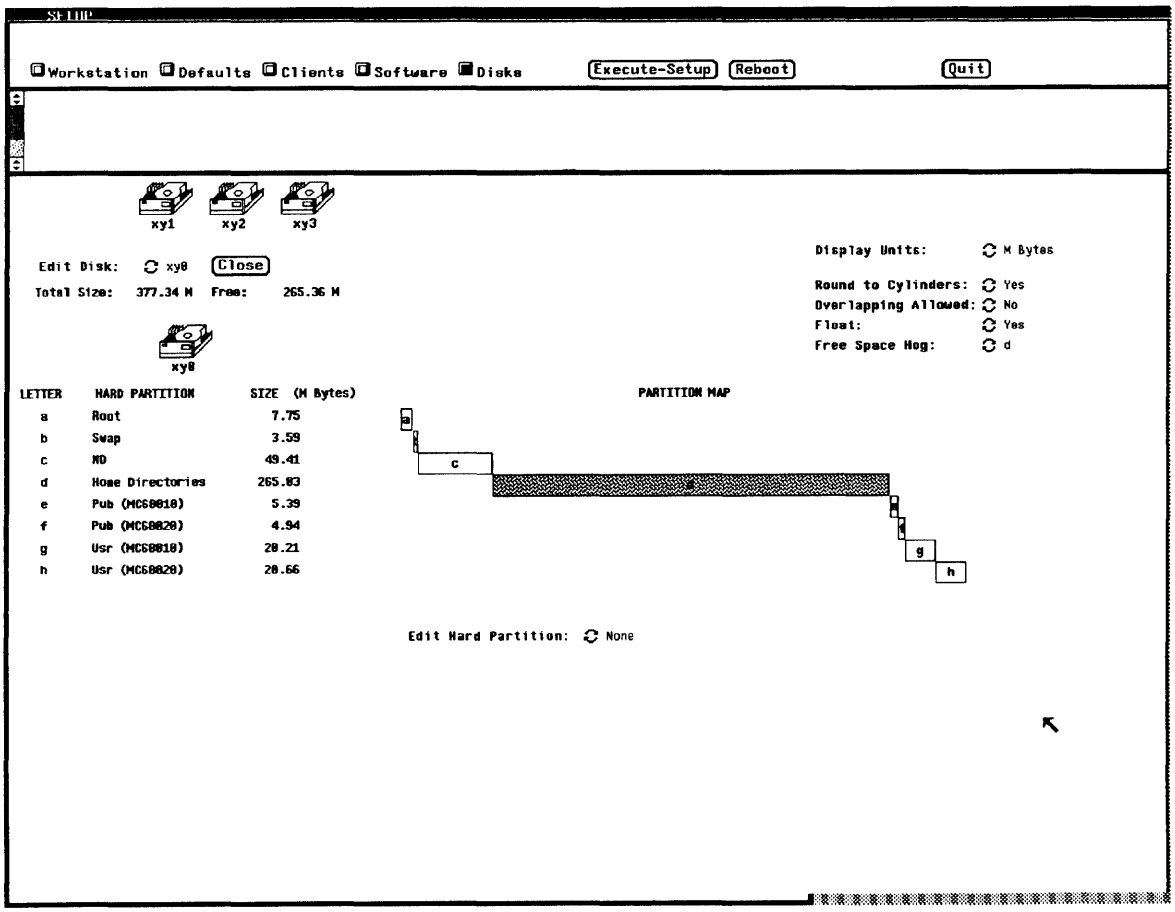
Disks Form

The Disks form presents you with all the disks connected to the machine being configured to allow you to individually edit their hard partitions. As with the Workstation and Client screens, the images of the disks and hard partitions are buttons. Selecting these with the mouse provides an accelerator for the "Edit" and "CLOSE" operations.

The current "Free Space Hog" partition, if any, will be shaded gray (partition "d" in Figure 5-7). The hard partition currently open for edit is shown inverted (partition "g" in Figure 5-7).

To the right of the "Size" of a partition is a *slider* to provide coarse settings of this attribute. Point in the slider with the mouse cursor and hold the left mouse-button down. The shaded portion of the slider will follow your movements and the "Size" field will also be updated. When you release the left button, the partition will be resized if possible.

Figure 5-7 *The Disks Form*



6

Walkthroughs

6

Walkthroughs

This chapter contains sample walkthroughs for a standalone machine and a heterogeneous server. You do not have to configure your systems by these examples. These are simply to help you design your own system configurations.

6.1. Standalone

Before beginning this the standalone walkthrough, you will need the following information:

name of your system (domain name)
Internet address
Yellow pages domain name

You have installed the system up to initializing *Setup*:

#
next to the prompt type:
setup

You will now see a series of questions, answer them like the example below:

```
Are you running Setup to:
                1) Install a major Sun Unix release
                2) Upgrade between major Sun Unix releases
                3) Demonstrate Setup
        >> 1


Will you be running Setup from a:
                1) Sun bit mapped display device
                2) cursor addressable terminal (TTY)
        >> 1
```

If you select "Other", the name of the terminal must correspond to a name in the *termcap*(5) database.

Setup begins running the interface selected.

The first screen: Workstation

Type in the name of your Workstation

Type in standalone after Workstation type

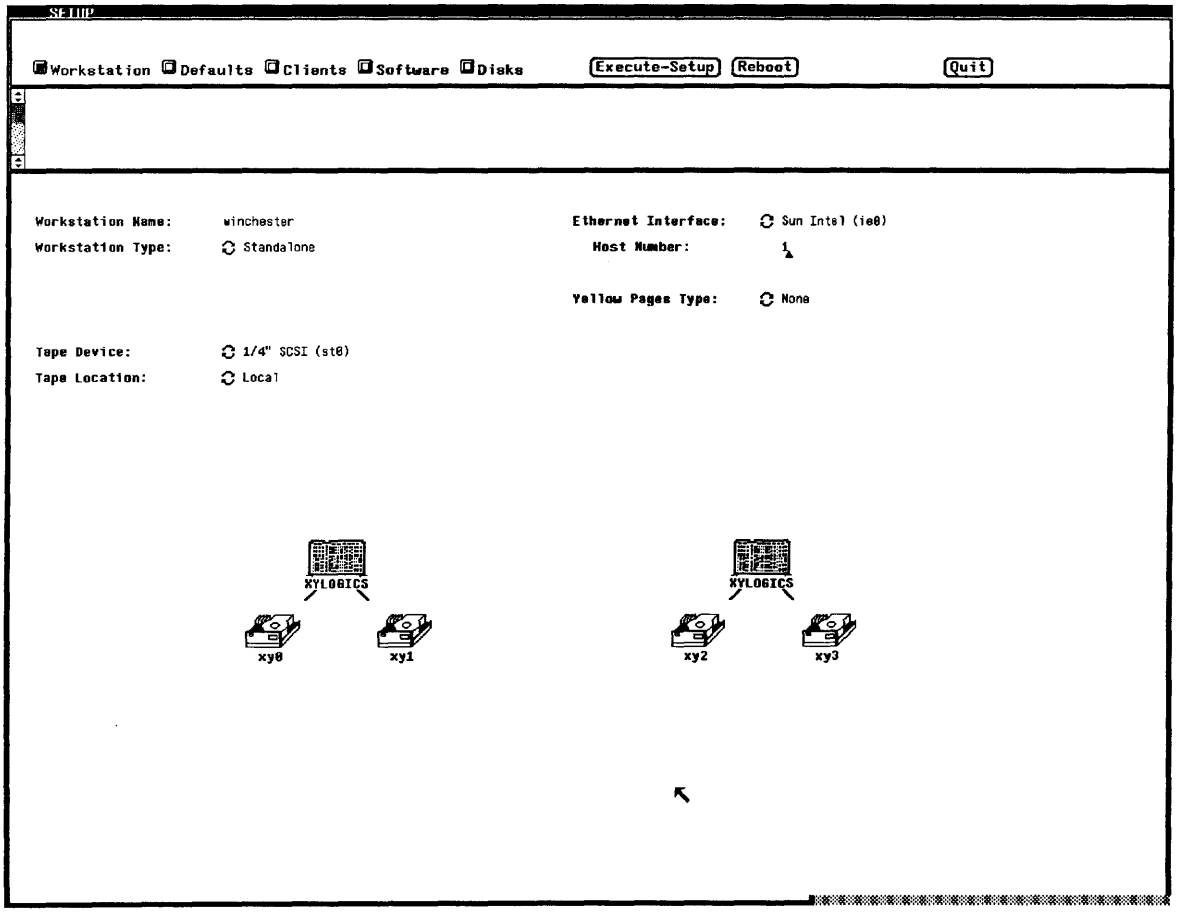
Pick correct CPU type

Pick Tape device

Pick local for tape location

Pick none for yellow pages

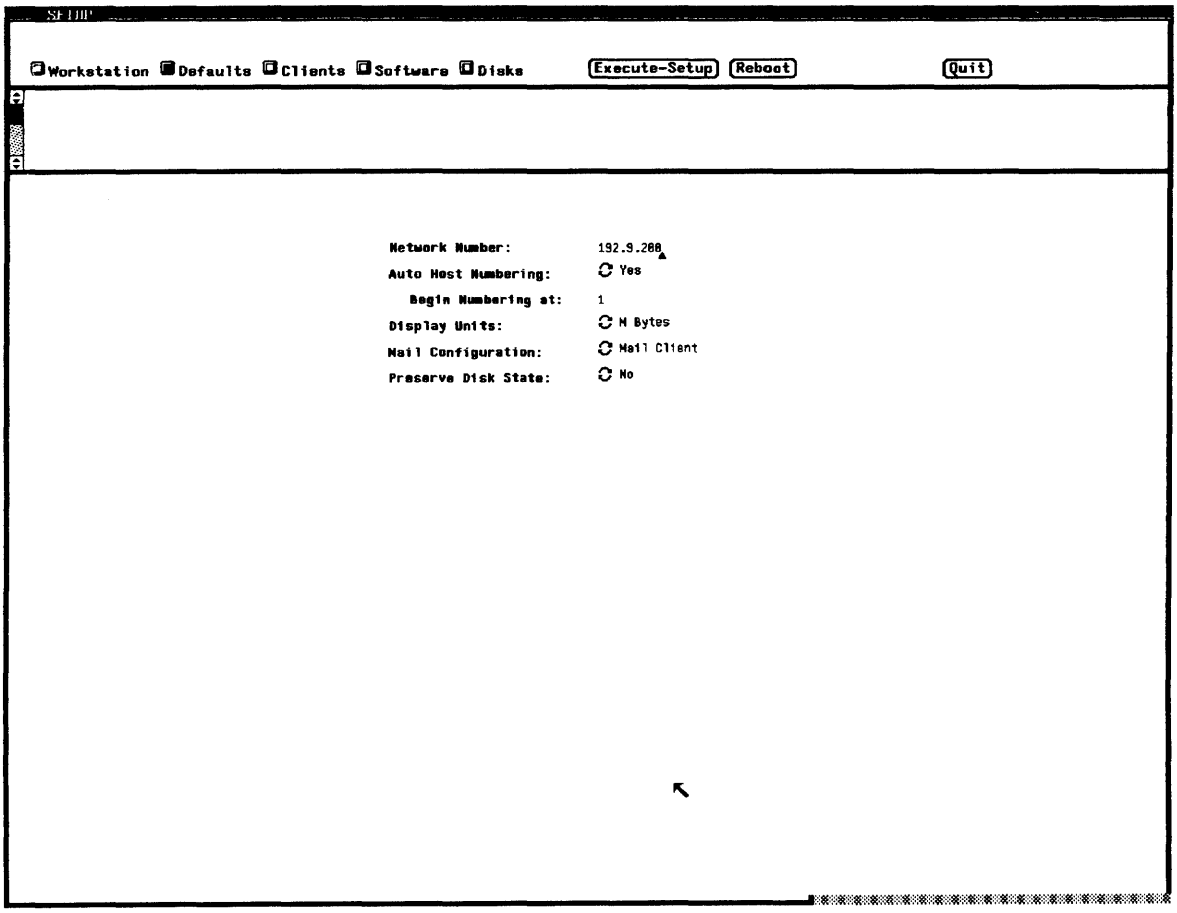
Figure 6-1 *Workstation Form:Standalone*



Default screen:

change the Network Number to reflect your system
any of the other choices are up to you and the way you want your system defaults.

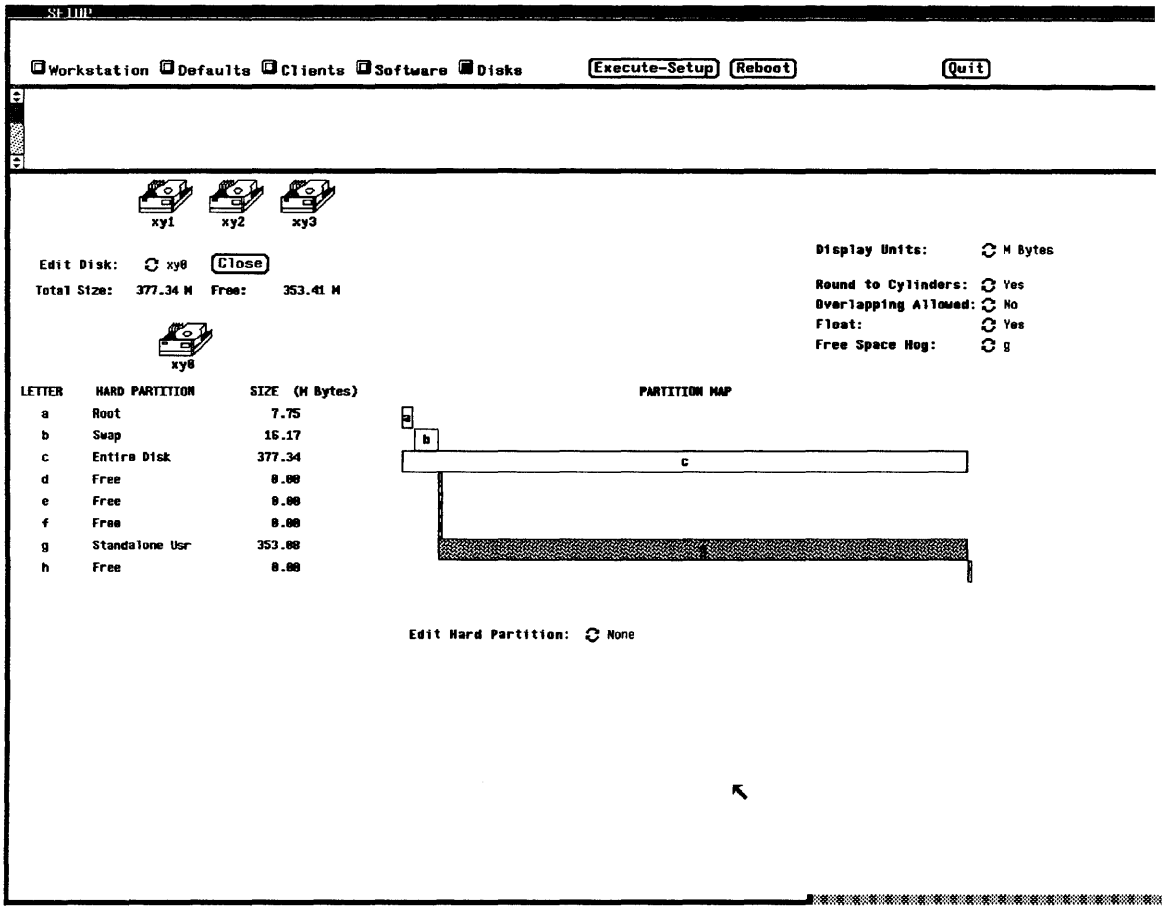
Figure 6-2 *Default Screen:Standalone*



Disk Screen:

When you select this screen, you will see icons at the top that indicate the type or types of disks you have on your system. You can rearrange your partitions by selecting the disk you want, the items below will reflect the disk you chose.

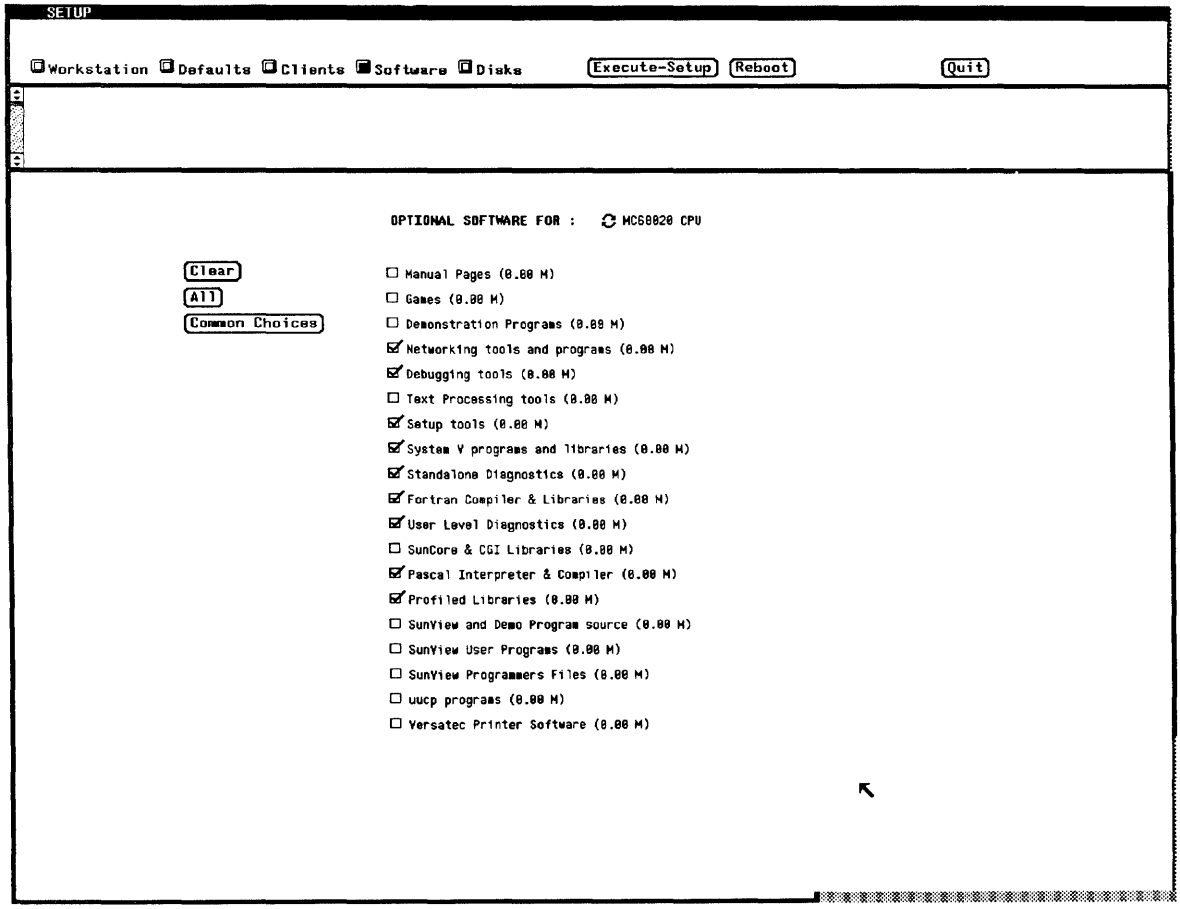
Figure 6-3 *Disk Screen:Standalone*



Software Screen:

You can choose the entire list of optional software, or you can pick only those that you want. If you choose common choice, it will default to a basic system.

Figure 6-4 *Software Screen:Standalone*



6.2. Heterogeneous Server

You will need the following information before beginning the hetrogeneous walkthrough:

the host name
internet address
hostnames of clients
clients' internet addresses
clients' ethernet addresses
yellow pages' domain name
yellow pages' master server name
network number

You have installed the system up to initializing *Setup*:

#

Next to the prompt type

```
# setup
```

You will now see a series of questions, answer them like the example below:

Are you running Setup to:
1) Install a major Sun Unix release
2) Upgrade between major Sun Unix releases
3) Demonstrate Setup
>> 1

Will you be running Setup from a:
1) Sun bit mapped display device
2) cursor addressable terminal (TTY)
>> 1

If you select "Other", the name of the terminal must correspond to a name in the *termcap*(5) database.

Setup begins running the interface selected.

The first screen: Workstation

Type in the name of the workstation

Workstation Type: server

CPU types server: check 68020 if your clients are Sun-3s

check 68010 if your clients are Sun-2s.

check both if you have clients from both architectures.

Tape Device: select the tape device(s) for your machine.

Tape location: select whether local or....

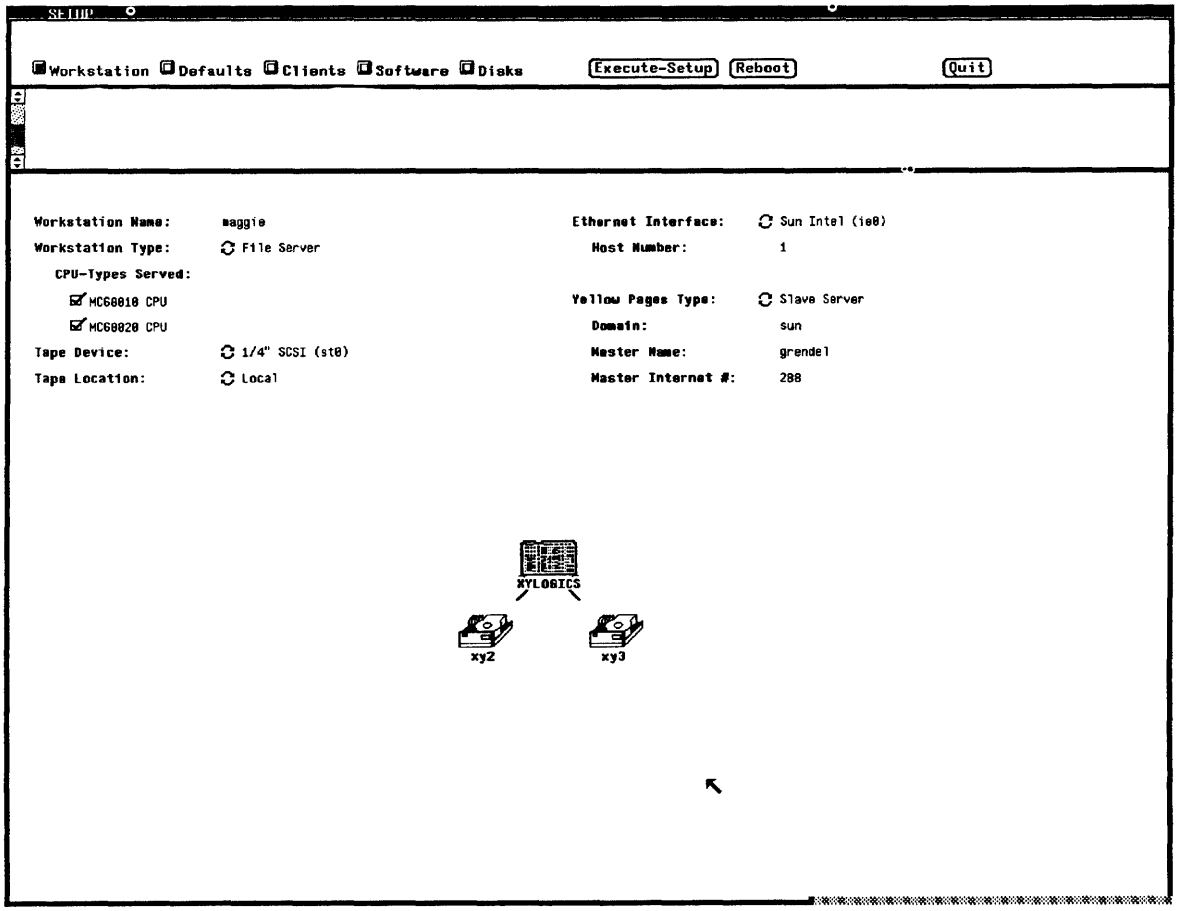
Ethernet interface: select type of interface.

Host Number: Fill in your host number.

Yellow Pages: select your yellow pages type.

Fill in Domain Name, Master Name, Master Internet #.

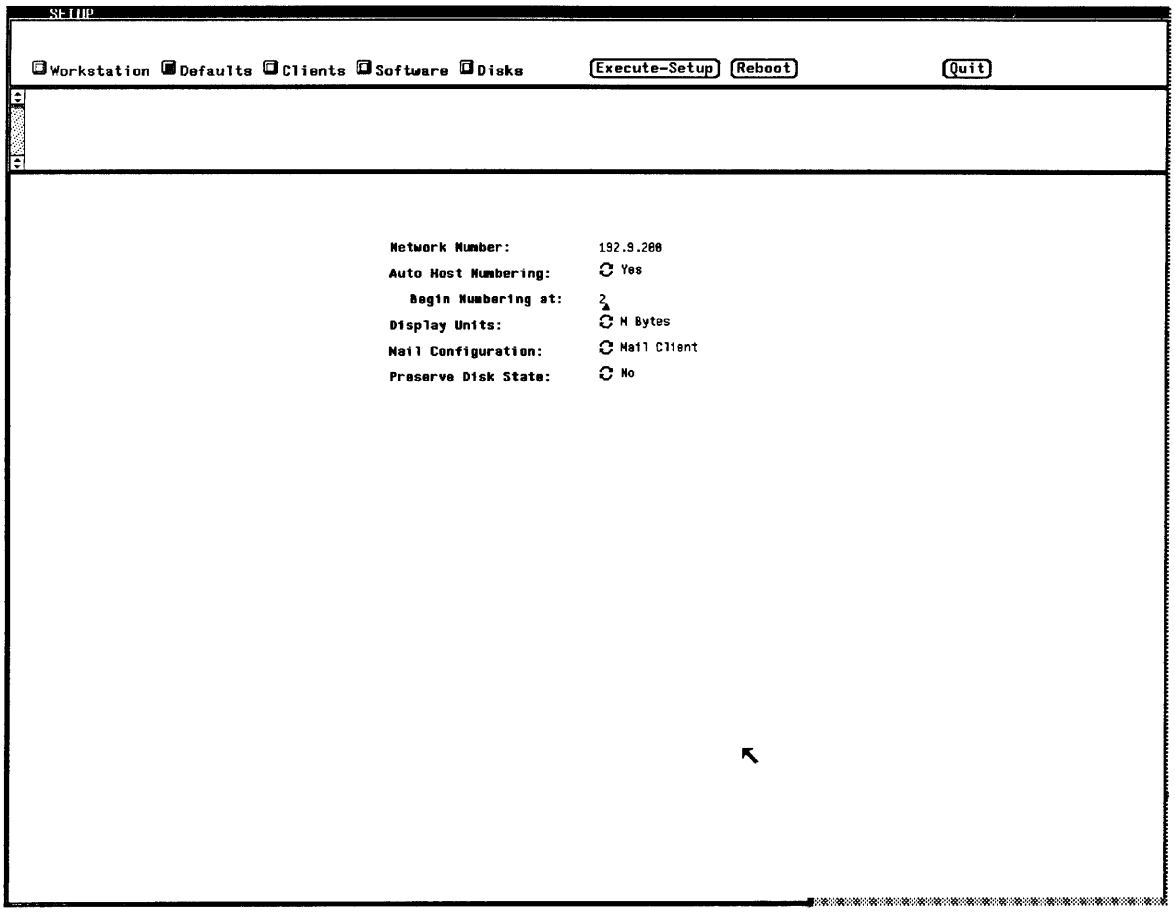
Figure 6-5 *Workstation Screen:Heterogeneous*



Default screen:

Change the Network Number to reflect your system any of the other choices are up to you and the way you want your system defaults.

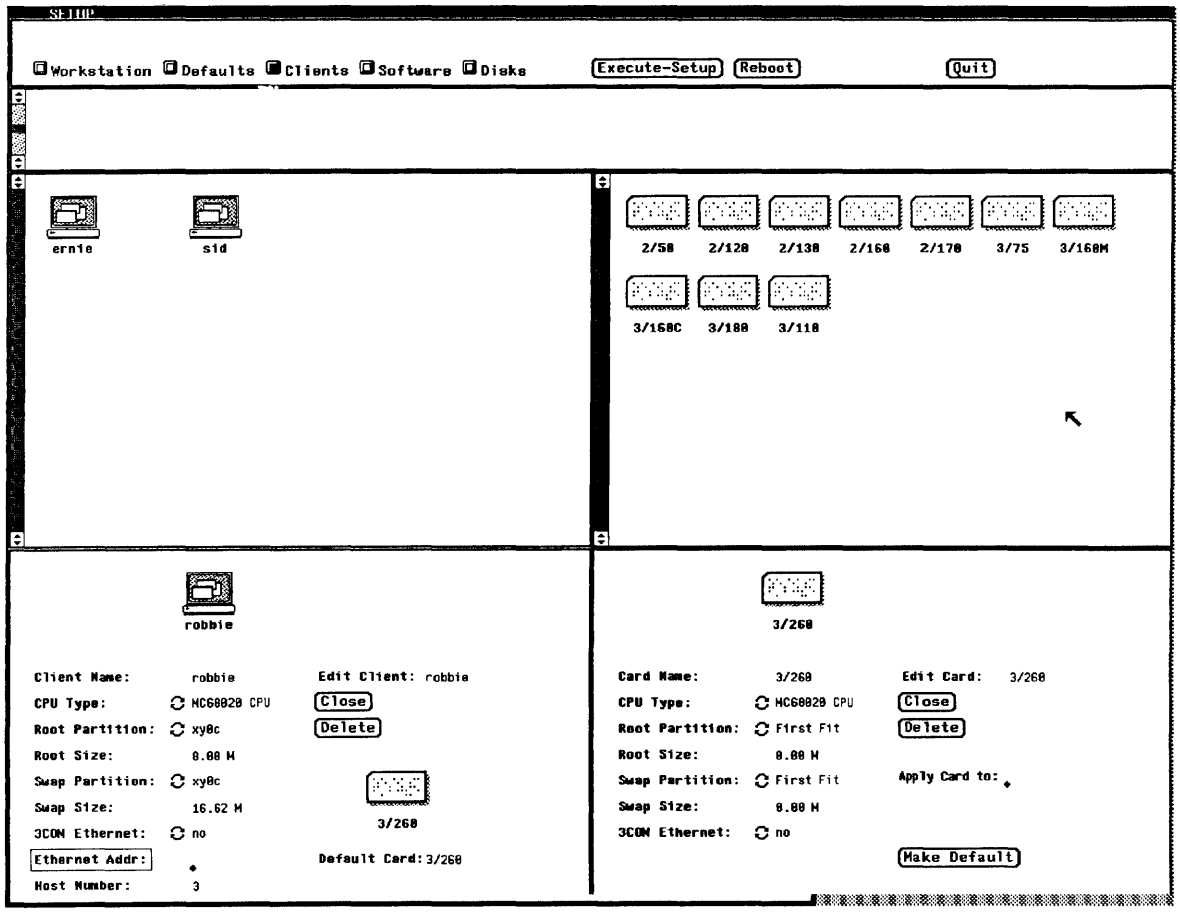
Figure 6-6 *Default Screen:Heterogeneous*



Clients Screen

The following Client Screen shows a system with three clients. You can edit the cards when your clients do not fit the default parameters of your system (eg.:root and swap partition sizes, default location of the swap and root partitions, etc.).

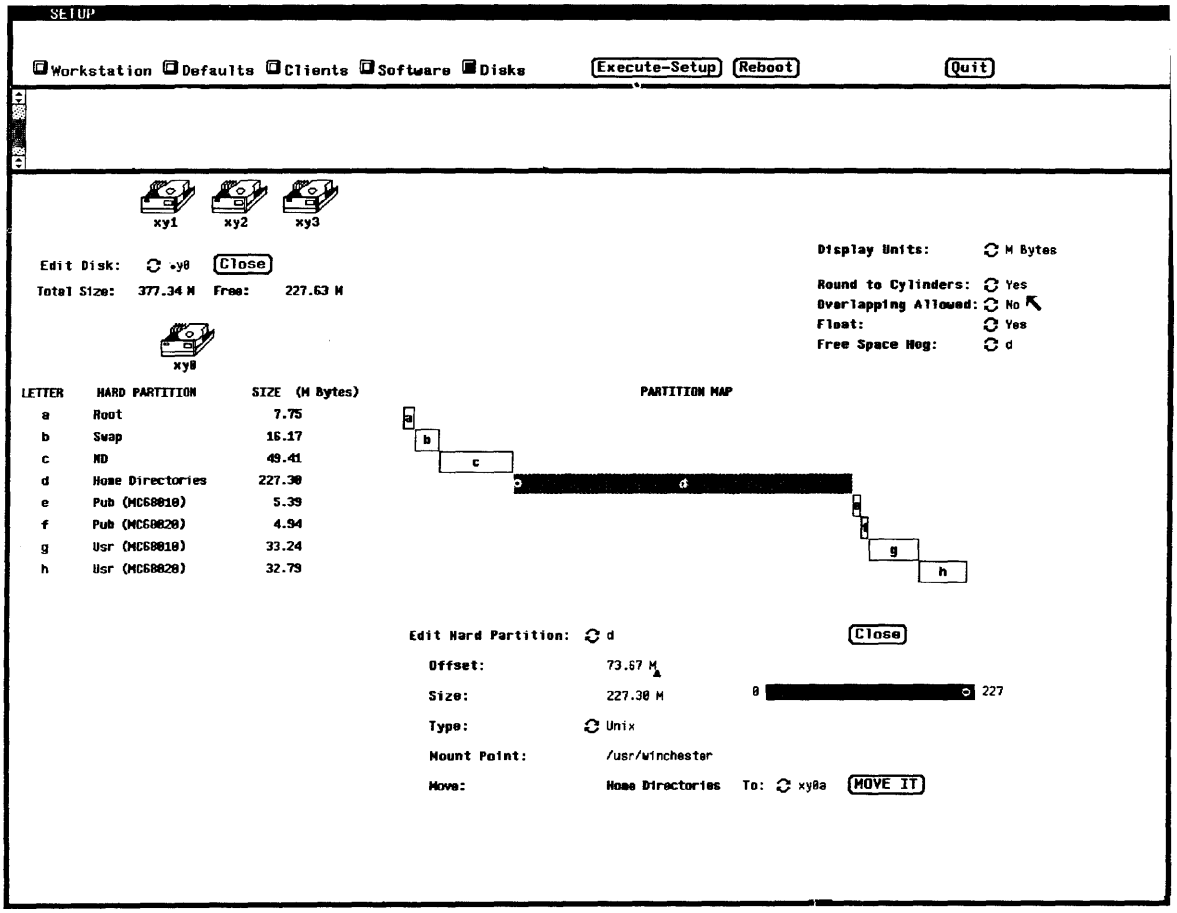
Figure 6-7 *Client Screen*



Disk Screen:

When you select this screen, you will see icons at the top that indicate the type or types of disks you have on your system. You can rearrange your partitions by selecting the disk you want, the items below will reflect the disk you chose.

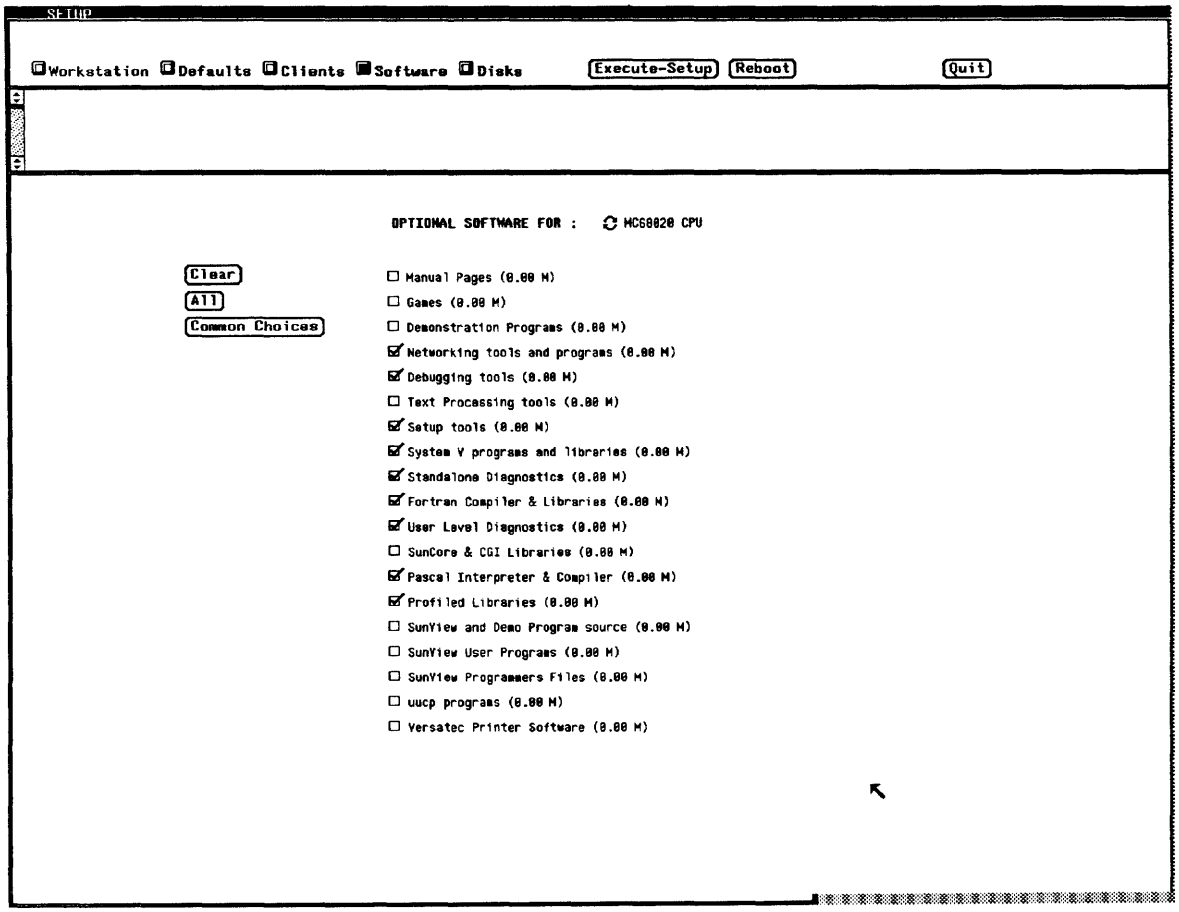
Figure 6-8 *Disk Screen: Heterogeneous*



Software Screen:

You can choose the entire list of optional software, or you can pick only those that you want. If you choose common choice, it will default to a basic system.

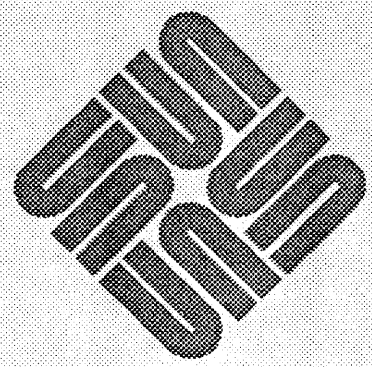
Figure 6-9 *Software Screen:Heterogeneous*



Now proceed to the next chapter *Executing Setup*.

7

Executing *Setup*



7

Executing *Setup*

You have now filled out all the information on either the tty or bit-map "forms" and you have just hit EXECUTE-SETUP. What happens now?

Clicking the EXECUTE-SETUP button causes *Setup* to check if any information is missing and, if not, it begins the installation process. If some piece of information is missing, *Setup* prints a message and returns you to the tty or bit-mapped interface. You should fill in the missing information and click EXECUTE-SETUP when you're ready to try again.

During the installation process, *Setup* prints messages to inform you where it is in the process. From time to time, it will also prompt you to insert or mount another tape. It will not begin reading from the tape until you have confirmed to *Setup* that the tape is ready to be used.

The installation process can be broken into the following steps:

- Labelling disks and making file systems. A new label is written on each disk to define the size and location of the hard partitions. A file system is made on each each hard partition that is to be a Unix file system. It takes a few minutes to make all the file systems.
- Updating system files. *Setup* prints a message as the files */etc/hosts*, */etc/nd.local*, and */etc/ethers* are updated. For a standalone workstation, only the */etc/hosts* file is updated.
- Extracting required files from the tapes. The required files come in three groups — the root files, the 'pub' files, and the */usr* files. The root and 'pub' files are extracted first and then, for a file server, the clients are initialized. Finally the */usr* files are extracted. This is typically the most time consuming portion of the installation. Extracting the root and 'pub' files takes 5 to 10 minutes each. Extracting the */usr* files takes about 30 minutes using a 1/4 inch tape on a 68010 workstation.

NOTE *If you are installing a heterogeneous server, Setup extracts files from the tape for each architecture.*

- Initializing clients (for a file server only). For each client, *Setup* makes a file system on its root ND partition and initializes the root partition. The first client's root partition is initialized from the tape and each subsequent client's root partition is initialized via a disk to disk copy from the first client. On the average it takes about 5 minutes per client.

- Extracting optional files from the tapes. Each optional software group that was chosen is extracted from the tapes. The amount of time spent here depends on which software groups were chosen and the size of the chosen groups.
- Yellow pages installation. If the workstation being installed is a yellow pages master or slave server, then the yellow pages must be installed. A master server is installed fairly quickly while it takes longer to install a slave server because many yellow pages data base files must be copied from the master server.
- Installation complete. The last message is the "Installation complete." message and indicates that *Setup* is finished.

NOTE *Sometimes an installation will fail due to an error reading the tape. This will usually be noted as a tar command failure; or a tar command called within a shell script fail. In these cases, you will get an error message and you will need to do the installation over again.*

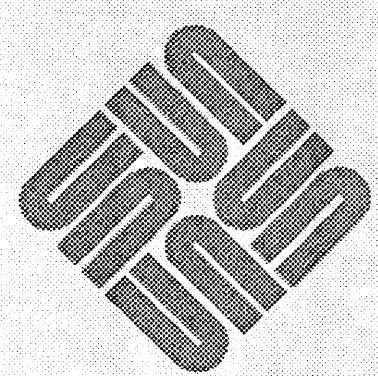
When *Setup* has acknowledged that the installation has completed successfully, review the message window then press the button labelled "REBOOT". UNIX will now be running on your fully configured machine. All that remains to be done is to make any changes to the Unix kernel that suit you particular installation.

If you do not want to reboot then press the button labelled "EXIT".

Proceed on to Chapter 8: *Configuring the System Kernel.*

8

Configuring the System Kernel



8

Configuring the System Kernel

Sun Microsystems' implementation of UNIX provides for a *configurable kernel.* Certain system parameters that were hardwired in previous implementations can now be changed. We strongly advise that you configure the system kernel on all new UNIX systems to meet your particular needs. This reduces the kernel size, giving a larger effective memory size to programs and improving system performance substantially. This is especially important if you intend to run the Sun Window System.

This chapter presents a very terse explanation and walkthrough of kernel configuration. We begin with a summary of the configuration process, explain the format of the configuration file used by the *config* utility (see *config*(8)) to build your system configuration, and then present a walkthrough of the procedure.

This material should be adequate for users who have some experience with UNIX kernel configuration, and for those who have a standard system configuration. However, if there is anything you do not adequately understand, please read the sections on kernel configuration in the Sun *System Administration Manual.* Those sections go into more detail on the kernel configuration process, describe the layout of the kernel code, and explain the format of the configuration file in much greater detail.

Your configurable system also allows for adding new device drivers; all the kernel object files required to build a new system are present. For procedures, see the *Writing Device Drivers* manual.

8.1. Kernel Configuration Introduction

Building a new system is a semi-automatic process; most of the fine detail is handled by a configuration-build utility called */etc/config*. */etc/config* uses five files as input; these files are in the */usr/sys/conf* directory as the system is shipped:

makefile.sun[23]	generic makefile for Sun-2 and Sun-3 systems,
files	lists files required to build the basic kernel,
files.sun[23]	files for a Sun-2 or Sun-3 specific kernel,
devices.sun[23]	name to major device mapping for Sun-2 and Sun-3 kernels, and
SYSTEM_NAME	describes characteristics of a specific system named *SYSTEM_NAME*. This is the only file you have to

'worry' about in this entire process; you must create it and tailor it to your system specifications.

When you run */etc/config*, it uses these files to generate the files needed to compile and link your kernel:

../SYSTEM_NAME/mbglue.s
: contains short assembly language routines used for vectored interrupts,

../SYSTEM_NAME/ioconf.c
: contains a description of I/O devices attached to the system,

../SYSTEM_NAME/makefile
: for building the system, and

../SYSTEM_NAME/device_name.h
: a set of header files (various *device_name*'s) containing devices which can be compiled into the system.

Next, you use the generated makefile to create the dependency graph for the new system, build the system, and, finally, install the new kernel.

If you are installing a server, you should also make and install a kernel for your clients.

8.2. Producing a System Configuration File

Your contribution to this process is to create a file *SYSTEM_NAME* which contains a description of the kernel you want */etc/config* to produce. It uses this information to create a kernel named *SYSTEM_NAME*. We have tried to make this phase as painless as possible in a few ways.

Rather than creating the file from scratch, you can copy and edit one of the several template files provided with the distribution. These templates are located in the */usr/sys/conf* directory, and their names are, by convention, written in upper case as in the example:

GENERIC contains a line for every Sun-2 or Sun-3 supported device.
Your system is shipped with a GENERIC kernel.

Copies of these files are included as Appendix A of this manual. Rather than starting from GENERIC, we suggest you choose one of the tailored files which most closely approximates your system configuration for editing.

To understand the format of the configuration file, begin by taking a look at the 'mother' template, GENERIC. The beginning of GENERIC looks something like this:

```
#
# GENERIC SUN-3
#
machine         "sun3"
cpu             "SUN3_160"
cpu             "SUN3_50"
cpu             "SUN3_260"
cpu             "SUN3_110"
ident           GENERIC
timezone        8 dst
maxusers        4
options         INET
options         SYSACCT
options         QUOTA
options         NFS
options         NIT

config          vmunix          swap generic

[ and so on ]
```

You can see by the line groupings that the configuration file has three different 'types' of entries:

- Lines which describe general things about the system (parameters global to the kernel image which this configuration file generates),
- A line which describes things specific to each kernel image generated, and
- Lines which describe the devices on the system, and what those devices are connected to.

The next three subsections cover these three types of lines.

General System Description Lines

The first six general description lines in the configuration file are mandatory for every Sun system. They are:

machine *type*
This system is to run on the machine type specified. Only **one** machine type can appear in the configuration file. The legal *type*s for a Sun system are "**sun2**" and "**sun3**" (note that the double quotes are part of the designation).

cpu *type*
This system is to run on the cpu type specified. More than one cpu type can appear in the config file. Legal *types* for a Sun-2 machine are "**SUN2_120**" (generic for any Multibus based Sun-2 machine), "**SUN2_50**" (generic for any VMEbus based Sun-2 machine). The legal *types* for a Sun-3 machine are "**SUN3_160**" (Sun-3/160C, Sun-3/160M and Sun-3/75), "**SUN3_50**" (Sun-3/50), "**SUN3_260**" (Sun-3/260) this includes all Sun-3/200 machines, and "**SUN3_110**" (Sun-3/110).

ident *name*

Gives the system identifier — a name for the machine or machines that run this kernel. *name* must be enclosed in double quotes if it contains both letters and numbers (for example, "SDST120"), or you will get a syntax error when you run */etc/config*. Also, note that if *name* is **GENERIC**, you can specify the unique *config_clause* **swap generic** in the **config** line described below. If you use any other string for *name*, and you also include an **options GENERIC** line, you can still use the **swap generic** line. However, if you use any other string for *name* and omit the **options GENERIC** line, *config* will set up the *makefile* as if you had.

timezone *number* **[dst]**

Specifies the timezone you are in, measured in the number of hours west of GMT. 5 is EST, 8 is PST. Negative numbers indicate hours east of GMT. If you specify **dst**, the system will convert to and from daylight savings time when appropriate. An optional integer or floating point number may be used to specify a particular daylight savings time correction algorithm; the default value is 1, indicating the United States. Other values are: 2 (Australian style), 3 (Western European), 4 (Middle European), and 5 (Eastern European). See *gettimeofday*(2) and *ctime*(3) for more information.

maxusers *number*

The maximum expected number of simultaneously active users on this system is *number*. This number is used to size several system data structures.

options *optlist*

Compile the listed options into the system. Options in this list are separated by commas. A line of the form "options FUNNY,HAHA" yields –DFUNNY –DHAHA to the C compiler. An option may be given a value, by following its name with "=" then the value enclosed in (double) quotes. None of the standard options use such a value.

In addition, options can be used to bring in additional files if the option is listed in the *files* files. All options should be listed in upper case. In this case, no corresponding *option*.h will be created as it would be using the corresponding *pseudo-device* method.

Specific System Description Lines

The next 'type' of line in the system configuration file is a single line specifying the name and location of a bootable kernel image. Multiple bootable images may be specified using multiple lines of this type. The line has the syntax:

config *kernelname config_clauses*

where

kernelname

Is the name of the loaded kernel image. *kernelname* is usually **vmunix**.

config_clauses

are one or more specifications indicating where the root file system is located, how many paging (or swap) devices there are and where they go. A *config_clause* may be one or more of the following:

root [on] *root_device*
Specifies location of the root file system.

swap [on] *swap_device* [and *swap-device*]
Specifies location of swapping and paging areas.

dumps [on] *dump_device*
Specifies device where you would like crash dumps to be taken.

The "on" in the syntax of each clause is optional, and the "and" clause in the **swap** clause may be repeated zero or more times. Multiple *config_clauses* are separated by white space. For example, the *config* line for a system with root on its first SMD disk (partition 'a'), and paging on the 'b' partition of this disk might be:

```
config vmunix root on xy0a swap on xy0b
```

Note also that the device names supplied in the clauses may be fully specified — as a device, unit, and file system partition — or underspecified. If underspecified, the *config* program uses built-in rules to select default unit numbers and file system partitions. For example, the swap area need not be specified at all if the root device is specified, because the default is to place swap in the "b" partition of the same disk where the root file system is located. Thus, our example line could have been simply:

```
config vmunix root xy0
```

Device Description Lines

Each device attached to a machine must be specified so that the system generated will know to probe for it during the autoconfiguration process carried out at boot time. The final type of entry in the configuration file tells the system what devices to look for and use, and how these devices are connected together.

Each device description line has the following format:

dev_type *dev_name* **at** *connect_dev* *more_info*

where

dev_type
Specifies the device type. *dev_type* may be one of the following:

controller
In general, a disk or tape controller.

disk or **tape**
Devices connected to a controller.

device
Something 'attached' to the main system bus, like an Ethernet controller.

pseudo-device
A software subsystem or driver treated like a device driver, but without any associated hardware. Current examples are the pseudo-tty driver and various network subsystems.

dev_name
The standard UNIX device name and unit number (if the device is not a **pseudo-device**) of the device you are specifying. For example, **xyc0** is the *dev_name* for the first Xylogics controller in a system; **ar0** names the first quarter-inch tape controller.

con_dev
is what the thing you are specifying is connected to. There are now several different bus types. They are:

virtual	Virtual preset
obmem	On board memory
obio	On board io
mbio	Multibus io (Sun-2 only)
mbmem	Multibus Memory (Sun-2 only)
vme16d16 (vme16)	16 bit VMEbus/ 16 bit data
vme24d16 (vme24)	24 bit VMEbus/ 16 bit data
vme32d16	32 bit VMEbus/ 16 bit data (Sun-3 only)
vme16d32	16 bit VMEbus/ 32 bit data (Sun-3 only)
vme24d32	24 bit VMEbus/ 32 bit data (Sun-3 only)
vme32d32 (vme32)	32 bit VMEbus/ 32 bit data (Sun-3 only)

more_info
is a sequence of the following:

csr *addr*
Specifies the address of the csr (command and status registers) for a device. The csr addresses specified for the device are now always the addresses within the bus type specified.

The csr address must be specified for all controllers, and for all devices connected to a main system bus.

drive *number*
For a disk or tape, specifies which drive this is.

flags *number*
These flags are made available to the device driver, and are usually read at system initialization time.

priority *level*
For devices which interrupt, specifies the interrupt level at which the device operates.

vector *intr number* [*intr number . . .*]
For devices which use vectored interrupts on VMEbus systems, *intr* specifies the vectored interrupt routine and *number* the corresponding vector to be used (64-254).

A **?** may be substituted for a number in two places and the system will figure out what to fill in for the **?** when it boots. You can put question marks on a *con_dev* (for example, at xyc?), or on a drive number (for example, drive ?). This allows redundancy, as a single system can be built which will boot on different hardware configurations.

The next few pages, contain an annotated copy of the `Sun-3GENERIC` file to help you identify the lines you need to include in your own system configuration file. The comments explain the device and pseudo-device lines, and may also refer you to the reference manual entry which covers the device in question. If the comments say the line is **mandatory**, the line *must* be included in every system configuration file, either exactly as it stands, or, if commentary indicates variables, with the variables adjusted to fit your system.

NOTE *For the specific GENERIC files for a Sun-2 or Sun-3 machine, see* Appendix C *of this document.*

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

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

Configuration Line	*Comments*	*Description*
#		
# GENERIC SUN-3		
#		
machine "sun3"	**mandatory**	Identifies the specific machine
cpu "SUN3_160"	**mandatory***	Identifies the specific CPU type (Sun-3/160, Sun-3/180, or Sun-3/75)
cpu "SUN3_50"	**mandatory***	Identifies the specific CPU type
cpu "SUN3_260"	**mandatory***	Identifies the specific CPU type (Sun-3/260 or Sun-3/280)
cpu "SUN3_110"	**mandatory***	Identifies the specific CPU type (Sun-3/110)
ident GENERIC	**mandatory**	See *General* and *Specific System Description Lines* for information. Finally, if *SYS_NAME* contains both alpha and numeric characters alpha and numeric characters (as in, for example, SDST120), you must enclose the name in double quotes ("SDST120") or you will get a syntax error when you run */etc/config*.
timezone 8 dst	**mandatory**	Specifies your timezone. Adjust value accordingly. Can also use half hour designations.
maxusers 4	**mandatory**	Number may vary. For most systems, "4" is the proper value for maxusers. See the section *General System Description Lines* for information.
options INET	**mandatory**	Controls inclusion of Internet code — see *inet* (4). You must also include the "pseudo-device loop" lines below.
options SYSACCT	*optional*	Controls inclusion of code to do process accounting — see *acct* (2) and *acct* (5).
options QUOTA	*optional*	Controls the disk quota checking system.
options NFS	*optional*	Inclusion of NFS code.
options NIT	*optional*	Inclusion of network interface tap code
options IPCMESSAGE	*optional*	Controls inclusion of code for SystemV IPC Message Facility.
options IPCSEMAPHORE	*optional*	Controls inclusion of code for SystemV IPC Semaphore Facility.
options IPCSHMEM	*optional*	Controls inclusion of code for SystemV IPC Shared-Memory Facility.
config vmunix swap generic	**mandatory**	Specify kernel name and configuration clauses. Please see *Specific System Description Lines* for information.
pseudo-device pty	*optional*	Pseudo-tty's. Needed for network or window system.
pseudo-device bk	*optional*	Berknet line discipline for high speed tty input — see *bk* (4).
pseudo-device ether	*optional*	ARP code. Must include if using Ethernet — see *arp* (4).
pseudo-device loop	**mandatory**	Software loop back network device driver — see *lo* (4). Must include with 'options INET'.
pseudo-device nd	*optional*	Network disk. Necessary for servers and diskless clients, and for machines serving as remote hosts for remote installation — see *nd* (4).
pseudo-device win128	*optional*	Window system. Number indicates maximum windows. If you include this line, you must also include the "pseudo-device dtop", "ms", and "kb" lines just below.
pseudo-device dtop4	*optional*	Maximum number of screens ('desktops'). Required for window system.
pseudo-device ms3	*optional*	Maximum number of mice. Required for window system — see *ms* (4).

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

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

connections for machine type type 1 (SUN3_160)

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

connections for machine type type 2 (SUN3_50)

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

connections for machine type 3 (SUN3_260)

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

connections for machine type type 4 (SUN3_110)

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

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

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

disk:

```
xy0 at xyc0 drive 0
```

hangs off of controller:

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

which hangs off bus type:

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

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

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

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

```
controller      xyc0 at vme16d16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller      xyc1 at vme16d16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk            xy0 at xyc0 drive 0
disk            xy1 at xyc0 drive 1
disk            xy2 at xyc1 drive 0
disk            xy3 at xyc1 drive 1
controller      sc0 at vme24d16 ? csr 0x200000 priority 2 vector scintr 0x40
disk            sd0 at sc0 drive 0 flags 0
disk            sd1 at sc0 drive 1 flags 0
tape            st0 at sc0 drive 32 flags 1
disk            sd2 at sc0 drive 8 flags 0
tape            st1 at sc0 drive 40 flags 1
#disk           sf0 at sc0 drive 8 flags 2
controller      si0 at vme24d16 ? csr 0x200000 priority 2 vector siintr 0x40
controller      si0 at obio ? csr 0x140000 priority 2
disk            sd0 at si0 drive 0 flags 0
disk            sd1 at si0 drive 1 flags 0
tape            st0 at si0 drive 32 flags 1
disk            sd2 at si0 drive 8 flags 0
tape            st1 at si0 drive 40 flags 1
#disk           sf0 at si0 drive 8 flags 2
device          zs0 at obio ? csr 0x20000 flags 3 priority 3
device          zs1 at obio ? csr 0x00000 flags 0x103 priority 3
device          mti0 at vme16d16 ? csr 0x620 flags 0xffff priority 4 vector mtiintr 0x8
```

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

8.3. Kernel Configuration Procedures

Now we begin the actual walkthrough.

The walkthrough assumes you're familiar with the information presented above; assumes you have chosen a system configuration file to serve as your template for creating your own specific system configuration file (see *Appendix C*); and, finally, assumes that the essential kernel source and object files are located in the */usr/sys* directory (this is how the system is shipped).

NOTE *If you wish to include IPC System parameters for System V in your kernel, see "Tuning IPC System Parameters" in the System V Enhancement Overview manual (800-1541-01).*

NOTE *You should only build kernels for a given machine type with the appropriate distribution (sun2 (68010) in /usr.MC68010/sys and sun3 (68020) in /usr.MC68020/sys).*

There are two paths: one for servers and one for standalone systems. The only difference is that the server's path includes procedures for configuring a special kernel for clients. Please use the appropriate procedures for your system.

Kernel Configuration for Standalone Systems

1. Choose a name for your configuration of the kernel — in the following we will use *SYS_NAME* to indicate this name. Note that by convention this name is all upper case.
2. Change directory to the */usr/sys/conf* directory, and make a copy of the model configuration file you are using as a template (*TEMPLATE_NAME*) for

your own specific configuration file.

This copy is the basis for your own specific configuration file; it is named *SYS_NAME* and you will edit it to reflect your system specifications (so it must be 'writable'):

```
tutorial# cd /usr/sys/conf
tutorial# cp TEMPLATE_NAME SYS_NAME
tutorial# chmod +w SYS_NAME
```

3. Edit your new system configuration file to reflect your system specifications. This part of the procedure takes some care and thought. We suggest you:

   - Carefully look over the annotated copy of the GENERIC configuration file provided above to make sure you include all the mandatory general system description lines and all lines which are mandatory for Sun systems;
   - Re-read the *Specific System Description Lines* section to be sure you have used the correct "**config . . .**' line;
   - Look through the annotated GENERIC, again, to make sure you have included all lines for all the devices and pseudo-devices in your configuration (and only these lines).

4. When you have completed editing your configuration file, run the */etc/config* program (from */usr/sys/conf*) to generate the files needed to compile and link your kernel:

   ```
   tutorial# /etc/config SYS_NAME
   Doing a "make depend"
   ```

   While *config* is running watch for any errors. Never use a kernel which *config* has complained about; the results are unpredictable.

   A successful run of *config* on your kernel configuration file generates a number of files in the kernel configuration directory (*../SYS_NAME*). These files are described in the introductory section above; unless you are curious about how the kernel's autoconfiguration scheme works, you should never have to look at any of them.

5. Now, change directory to your kernel configuration directory (*../SYS_NAME*). Then, use *make*(1) to build the new kernel:

   ```
   tutorial# cd ../SYS_NAME

   tutorial# make

   [ lots of output ]
   ```

   Note: if you have specified multiple bootable kernel images in your system configuration file with multiple **config** lines, and you wish to make only one of those multiple kernel images, type **make** *imagename* (where *imagename* is the *kernelname* specified on the **config** line of the configuration file) instead of simply **make** for the last command. Using **make** without arguments generates all kernels specified in the configuration file.

6. Now you can install your new kernel and try it out.

   First move the original kernel to another (safe) place, then copy the new kernel to the place of the original, and finally boot the system up with this new kernel. In the example, we assume you used **vmunix** for your *kernel-name*; substitute your own kernel image name for **vmunix** if you used a different one:

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

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

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

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

Kernel Configuration for Servers

The following procedures assume you want to create two kernels: one for the server itself, and one for the all the clients. The clients' kernel configuration file includes the entire set of devices used by all the client systems. The client kernel image is installed in */pub*.

NOTE *REMEMBER, you should only build kernels for a given machine type with the appropriate distribution (sun2 (68010) in /usr.MC68010/sys and sun3 (68020) in /usr.MC68020/sys). The kernel image installed in /pub must reflect the same architecture (/pub.MC68010 or /pub.MC68020).*

1. Choose a name for your server's configuration of the kernel — in the following we will use *SERVER_NAME* to indicate this name. Note that by convention this name is all upper case.
2. Change directory to the */usr/sys/conf* directory, and make a copy of the model configuration file you are using as a template (*TEMPLATE_NAME*) for the server's own specific configuration file.

This copy is the basis for your own specific configuration file; it is named *SERVER_NAME* and you will edit it to reflect your system specifications (so it must be 'writable'):

```
tutorial# cd /usr/sys/conf
tutorial# cp TEMPLATE_NAME SERVER_NAME
tutorial# chmod +w SERVER_NAME
```

3. Edit your new system configuration file to reflect your system specifications. This part of the procedure takes some care and thought. We suggest you:
   - Carefully look over the annotated copy of the GENERIC configuration file provided above to make sure you include all the mandatory general system description lines and all lines which are mandatory for Sun systems;
   - Re-read the *Specific System Description Lines* section to be sure you have used the correct "**config . . .**' line;
   - Look through the annotated GENERIC, again, to make sure you have included all lines for all the devices and pseudo-devices in your configuration (and only these lines).
4. When you have completed editing your configuration file, run the */etc/config* program (from */usr/sys/conf*) to generate the files needed to compile and link your kernel:

   ```
   tutorial# /etc/config SERVER_NAME
   Doing a "make depend"
   ```

   While *config* is running watch for any errors. Never use a kernel which *config* has complained about; the results are unpredictable.

   A successful run of *config* on your kernel configuration file generates a number of files in the kernel configuration directory (*../SERVER_NAME*). These files are described in the introductory section above; unless you are curious about how the kernel's autoconfiguration scheme works, you should never have to look at any of them.
5. Now, change directory to your kernel configuration directory (*../SERVER_NAME*). Then, use *make*(1) to build the new kernel:

   ```
   tutorial# cd ../SERVER_NAME

   tutorial# make

   [ lots of output ]
   ```

   Note: if you have specified multiple bootable kernel images in your system configuration file with multiple **config** lines, and you wish to make only one of those multiple kernel images, type **make** *imagename* (where *imagename* is the *kernelname* specified on the **config** line of the configuration file) instead of simply **make** for the last command. Using **make** without arguments generates all kernels specified in the configuration file.
6. Now prepare a kernel for your clients in the same way. When editing the configuration file (called *CLIENT_KERNEL_NAME* in the following), remember

to include the entire set of devices used by all the machines:

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

7. Now you can install both new kernels and try them out.

   - To install the server's kernel, first move the original kernel to another (safe) place, then copy the server's new kernel to the place of the original, and finally boot the server up with this new kernel. In the example, we assume you used vmunix for your *kernelname*; substitute your own kernel image name for vmunix if you used a different one:

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

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

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

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

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

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

booted up on the original, you can go about trying to fix the faulty kernel. For example, on the server:

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

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

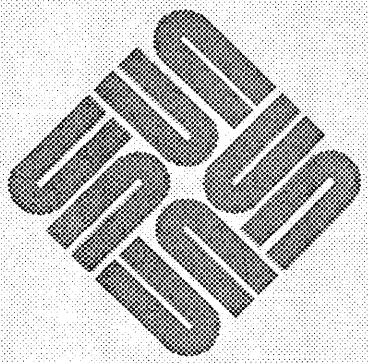
On the server:

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

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

9

Glossary



9

Glossary

This chapter is a dictionary of terms that will help you to use Setup effectively.

9.1. Terminology and Context

The following subsections describe terms used in the installation procedures, and provide necessary background information.

Machine Types

Standalone
: A standalone workstation has a complete root file sytem on its disk and does not require another machine to boot UNIX. It must have its own disk; it may or may not be attached to an Ethernet, and it may or may not have a local tape drive. But it does not rely on any other machines for storage of any sort. **A standalone machine without a local tape drive requires special installation procedures.** These are in Appendix B: *Installing UNIX on Tapeless Workstations.*

Server
: A server workstation on a local network provides resources like services and disk storage for other machines, which are called "clients". For installation purposes, the term "server" means "network disk server" (or "nd server" — see *nd*(4P)), that is, a machine which provides disk storage for its clients. Normally, a server uses both the *nd* and the NFS ("Network File System" — see *nfs*(4P)) protocols to exchange files with its clients. **For installation purposes, a server must have a local tape drive.**

Diskless Client
: A client workstation on a local network relies on a server for disk storage. **To 'install' a diskless client workstation, complete installation on its server and then simply power it on.** The only things you need to know about the client for server installation are its name, hardware Ethernet address (see *Networking Terminology*, below), and Internet address (see below).

Diskful Client
: A diskful client workstation on a local network relies on a server for resources, such as files, but has its own disk storage. A diskful client can actually boot the UNIX system up single user, but requires a remote machine to run multi-user. In other words, some of its files are local, and others are remote. The remote files can be obtained from any machine running NFS; the machine need not be an nd server.

At this writing, a straightforward installation path for such a machine does not exist; however, the NFS System Administration material, included with your documentation, provides notes. Basically, you can follow the path for a standalone machine until you get to running *setup*; then you must edit files by hand, and so on. See the NFS material for procedures. and *Appendix H* of this manual.

Networking Terminology

If you are installing several machines linked by a local network, part of installation includes basic network configuration.

Hardware configuration must be completed first: each machine must have Ethernet controller hardware, and be 'plugged in' via a transceiver to a common Ethernet cable. For Ethernet hardware configuration instructions see the *Hardware Installation* manual for your machine model.

Before proceeding with software installation, you must obtain basic information about the system's place in the network. Some of the information items, like machine name, are arbitrary, and others are determined by hardware.

For additional information on networking and tuning your configuration, see the Sun *System Administration Manual.*

Before proceeding with the installation, obtain, decide on, and write down all the items in the following list:

Hostname
: (Also called "machine name"). Names the workstation. Although hostnames may contain up to 32 alpha/numeric characters, it is best to keep them brief. All alpha characters in a hostname **must be lower case.** Avoid the use of special characters. Assign each machine in your network a hostname.

Ethernet Address
: Refers to the address which is permanently assigned to each workstation, and is used by the Ethernet software to decide which packets to deliver to that machine. The Ethernet address resides in the ID PROM on the Sun CPU Board. This address is a 6-byte hexadecimal value with each byte separated by a colon. A typical Ethernet address is "8:0:20:0:14:76".

To find a machine's Ethernet address, power it on. You will see the Sun logo, and a message like:

```
Self Test completed successfully.

Sun Workstation, model_type, keyboard_type
ROM Rev N, some_number_MB memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .
```

abort using the appropriate abort sequence here

```
Abort at some address
>
```

Network Number
: The network number is an arbitrary value used to uniquely identify local networks across an interconnected network system. If you expect to be

connected to the DARPA Internet, you must contact DARPA to get a network number which is unique on the Internet. If you do not intend to connect to a wider area network, you may use Sun's default network number: **192.9.200**. This number MUST be written with periods between the numbers.

Host Number

The host number of each host uniquely identifies the host machine to the all other machines on the local net. The host number **must be between 1 and 254** (since 8 bits are allocated for host numbers on Class C networks), and it **must be unique**.

You may elect to have host numbers assigned to all nodes 'automatically', or you may assign them 'manually'. If you elect to have then assigned automatically, the *setup* program, which you run later during the installation, handles it. When you run *setup* (see the chapter *Using Setup to Configure Your System*), it asks if you want host numbers assigned automatically. If you answer yes, your server is given host number "1", and your clients are given numbers "2", "3", etc. This works well if you are installing a new network with no previously existing nodes attached to it. **However, if you are adding workstations to an existing network you should assign the host numbers yourself when using *setup*,** to ensure that each host number is unique.

Internet Address

A machine's Internet address consists of two parts: the network number followed by the host number. For example, "192.9.200.45" is a typical Internet address consisting of Sun's default network number, "192.9.200", followed by the host number "45". See *inet*(3) man page for classification of networks.

Domain Name

The domain name identifies a group of workstations on a single local network that share the same */etc/passwd* and */etc/hosts* files. Note that you only *need* a domain name if you want to link two local networks, and hostnames on the two nets are not unique — two hosts with the same machine name are distinguished by their domain names. You *must* have a domain name if you plan to use the yellow pages. Without one the yellow pages will not work properly. Usually, one organization will have one */etc/hosts* file for the entire organization; the domain name is then unnecessary. Note also that this domain name has absolutely nothing to do with the domain name used by the *sendmail*(8) program for mail routing on the ARPA Internet.

Remote Host

A remote host is any machine on the network that provides a service.

Target Machine

A target machine is the machine you are installing.

Setup Terminology

Workstation Form
: This form describes the workstation being configured.

Client Form
: This form allows you to enter the information about the clients being served by a workstation that is a file server.

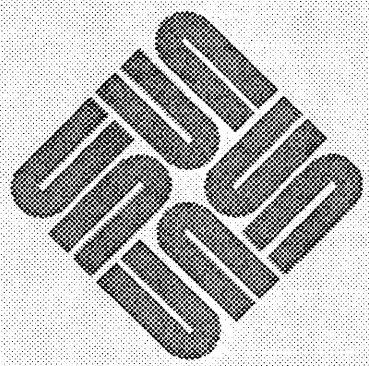
Optional Software Form
: This form allows you to choose what optional software.

Disk Form
: The Disk Form allows you to edit the hard partitions of a disk.

Defaults Form
: This form is used to change the value, from the default, of a number control parameters such as the units used in the display of disk sizes.

A

Files Affected by *Setup*



A

Files Affected by *Setup*

This section describes the files that *Setup* affects during an installation.

A.1. Installing a standalone machine

When a standalone machine is installed, *Setup* creates the following files:

/etc/hosts	Contains the hostname and Internet address of the machine being installed. (See Glossary) If the machine being installed does not have a local tape, then the hostname and Internet number of the tape server are also included.
/etc/fstab	Contains lines for the workstation's root and */usr* partitions. If any additional UNIX or swap partitions were created with *Setup's* Disk screen, lines for those partitions are also included.
/etc/exports	Contains one line for each of the UNIX partitions. *Setup* installs a standalone workstation such that all of its UNIX partitions are exported.
*/dev/**	Device files are created for each of the disks that *Setup* finds and for the tape device that is used during the installation.
/etc/rc.local	The domainname is set in this file. This is done only if the yellow pages are being used.
/etc/ypbind	If the yellow pages are not being used then this file is moved to */etc/ypbind.orig* so that the yellow pages will not be enabled when the machine is booted.
/etc/rc.boot	The hostname is set in this file.

A.2. Installing a file server

When installing a file server, *Setup* affects files on both the server and the clients.

The files affected on the file server are:

/etc/hosts	Contains the hostname and Internet address of the file server each of the clients. If the file server does not have a local tape, then the hostname and Internet number of the tape server are also included.

/etc/nd.local Contains the definitions of the public partition(s) and the root and swap partitions for each client.

/etc/ethers Contains the hostname and ethernet address of the file server and each of the clients.

/etc/fstab Contains lines for the file server's root, */usr* and */pub* partitions and the partition used for client's home directories. The */usr* and */pub* partitions are architecture-dependent and in the case of a heterogeneous file server there is one */usr* and */pub* partition per architecture. If any additional UNIX or swap partitions were created with *Setup's* Disk screen, lines for those partitions are also included.

/etc/exports Contains one line for each of the UNIX partitions. *Setup* installs a file server such that all of its UNIX partitions are exported.

*/tftpboot/** In this directory, a symbolic link is made for each client to the appropriate *tftp* boot program. There are four boot programs that expect the client to be either a Sun-2 (MC68010) or Sun-3 (MC68020) booting from either public partition zero or one. The name of the symbolic link is the client's Internet address translated into hexadecimal (using all upper case letters). For example, the file name for a client whose Internet address is 192.9.200.10 is C009C80A. If this client is a Sun-3 booting off of public partition one then the symbolic link is to file *ndboot.sun3.pub1*.

/usr, /pub, /private These files are all symbolic links to architecturally-dependent directories. For example, if the file server is a MC68020, then */usr* is linked to */usr.MC68020*.

*/dev/** Device files are created for each of the disks that *Setup* finds, for the tape device that is used during the installation, and for the clients' root partitions (*ndl**).

/etc/rc.local The domainname is set in this file. This is done only if the yellow pages are being used.

/etc/ypbind If the yellow pages are not being used then this file is moved to */etc/ypbind.orig* so that the yellow pages will not be enabled when the machine is booted.

/etc/rc.boot The hostname is set in this file. Also, the line to mount */pub* is changed to mount /pub.*arch* where *arch* identifies the architecture of the file server.

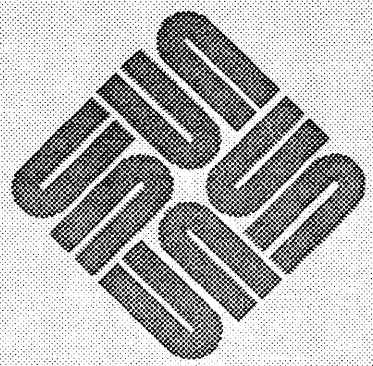
A.3. Files affected on a client

When installing a file server, the files affected on a client are:

/etc/hosts	Contains the hostname and Internet address of the file server and each of the clients.
/etc/fstab	Contains lines for the client's root partition and the server's public partition. It also contains lines for the client to import his */usr* directory and his home directory area from the file server. The home directory area's name has the form: **/usr/***servername*. If the server's hostname is **turing**, then the home directory area is */usr/turing*.
*/dev/**	Device files are created for the client's root and swap partitions (*nd**), and the public partition (*ndp**).
/etc/servers	The *rpc.mountd* line is commented out as clients are not expected to export file systems.
/etc/rc.local	The domainname is set in this file. This is done only if the yellow pages are being used.
/etc/ypbind	If the yellow pages are not being used then this file is moved to */etc/ypbind.orig* so that the yellow pages will not be enabled when the machine is booted.
/etc/rc.boot	The hostname is set in this file.

B

Installing UNIX on Tapeless Workstations



B

Installing UNIX on Tapeless Workstations

This chapter describes how to install UNIX on a standalone workstation or a file server which does not have a resident tape drive. To do this, you use the tape drive on another, fully installed machine. This machine is your tape server — which we call your **remote host** — and perform the installation on your **target machine** across the Ethernet.

NOTE

1. The remote host MUST be running Release 3.2. The remote host machine must either be configured as a server or a standalone machine; it may not be a client. If the machine is configured as a standalone, you must make it 'look' like a network disk server for remote installation. To do this, there is one primary requirement: the machine's kernel must have been generated from a system configuration file which includes the device description lines:

```
pseudo-device   ether
pseudo-device   nd
```

2. The remote host must have at least 5 megabyte of free disk space. If the remote host is a file server, this space must be in the */pub* partition and if the remote host is a standalone machine, this space must be in the */usr* partition.

B.1. Overview of the Installation Procedure

A "remote installation" is very similar to standard UNIX installation; steps are:

1. Complete UNIX installation on your remote host. See chapters 1 through 7 of this document for procedures. As noted above, the remote host must either be configured as a server or a standalone machine, and must have the "pseudo-device nd" and "pseudo-device ether" lines included in its system configuration file.
2. If your remote host is configured as a standalone system, you must enable it as a server and turn its */usr* file system into a public network disk. If your remote host is configured as a server, this step is unnecessary.
3. Make the ethernet addresses and Internet addresses for both the remote host and the target machine available to the networking software.
4. Start the reverse-ARP daemon.

5. Edit */.rhosts*.
6. Determine the unit number of the public partition (*pub#*).
7. Link to the proper *tftpboot* file.
8. Determine the network information necessary for installation.
9. Load the mini UNIX system onto the public partition of your remote host from the distribution tape.
10. Boot *diag* over the network; run *diag* to format (if necessary) and label your disk.
11. Boot the standalone *copy* program over the network. Run *copy* to copy a mini UNIX system over the network into the swap area on your disk.
12. Boot the mini UNIX system.
13. Run the *Setup* program to install UNIX from the tape and initialize the network files.
14. Boot the full UNIX system.

B.2. Configuring the Remote Host as a Network Disk

NOTE *Follow the procedures in this next section only if you are using a standalone machine rather than a server as your remote host. You can proceed to the next section if your remote host is already configured as a server.*

B.3. Configuring the Remote Host as a Network Disk Server

If the remote host machine is not configured as a network disk server, you need to turn the host's */usr* file system into a public network disk so that your target machine can access the files necessary for remote booting. Do the following on the remote host:

1. Edit the */etc/nd.local* file and add the following two lines at the end of the file.

```
user 0 0 /dev/usr_partition 0 -1 -1
son
```

usr_partition refers to the disk controller abbreviation, unit number and partition letter of the */usr* partition. The disk controller abbreviations are xy for a Xylogics disk controller, sd for a SCSI disk controller, and ip for an Interphase disk controller. On most standalone machines the */usr* partition is on disk unit zero and on partition *g*. For example, for a standalone machine with a Xylogics disk controller the commands are:

```
user 0 0 /dev/xy0g 0 -1 -1
son
```

2. Enable the network disk server by typing the following (the argument to MAKEDEV below is three alpha characters "ndl" followed by 'zero'

numeric):

```
host# cd /dev
host# MAKEDEV nd10
host# /etc/nd - < /etc/nd.local
```

The device description lines mentioned above must be included in the host machine's system configuration file for this to work.

B.4. Configuring the Remote Host

Now you must configure the remote host to allow it to communicate with the target machine.

1. You must insure that the ethernet addresses and Internet addresses for both the target machine and the remote host are known to the networking software.

   If you are using the yellow pages, this information must be added to the yellow pages databases on the yellow pages master server.

   If you are not using the yellow pages, the Internet addresses are added to the file */etc/hosts* and the ethernet addresses are added to the file */etc/ethers*.

   For example, if you are not using the yellow pages and you have the following configurations:

	Host name	Internet Addr	Ethernet Addr
Remote host	tapeserver	192.9.200.100	8:0:20:0:0:1
Target Machine	tapeless	192.9.200.101	8:0:20:0:0:2

   make sure */etc/hosts* contains these lines:

```
192.9.200.100 tapeserver
192.9.200.101 tapeless
```

   make sure */etc/ethers* contains these lines:

```
8:0:20:0:0:1 tapeserver
8:0:20:0:0:2 tapeless
```

2. Start the reverse-ARP daemon. Its arguments are the name of the ethernet interface and the host name. If you have a Sun ethernet board, the ethernet interface is *ie0*. If you have a 3COM ethernet board, the ethernet interface is *ec0*. For example, if the remote host's name is *tapeserver* and it has a Sun ethernet board the command is

```
host#  /usr/etc/rarpd ie0 tapeserver
```

3. Add the target machine's host name to the file */.rhosts* file. This enables the target machine to execute remote commands on the remote host machine.

4. You must determine the unit number of the public partition that is being used for remote booting. This number will be used in the next step and will be referred to as *pub#*.

   If the remote host is a standalone machine, the *pub#* is unit zero.

If the remote host is a file server, then you must look in the */etc/nd.local* file to find the unit number. If the file server is serving only one architecture, look for a line beginning with **user 0** and the unit number will be the number immediately following **user 0**. In the following example the unit number is one

```
user 0 1 /dev/xy0f 0 -1 -1
```

If the file server is serving multiple architectures, then there will be more than one line beginning with **user 0**. It is best to use the public partition that serves the same architecture as the target machine. Use */etc/mount* to determine the device name associated with each */pub.** file system and then look for a line beginning with **user 0** that is on the appropriate device name. The unit number will be the number immediately following **user 0**. For example if the output of the */etc/mount* command is

```
host# /etc/mount
/dev/xy0e on /pub.MC68010 type 4.2 (rw)
/dev/xy0f on /pub.MC68020 type 4.2 (rw)
```

and the */etc/nd.local* file contains the following lines

```
user 0 0 /dev/xy0e 0 -1 -1
user 0 1 /dev/xy0f 0 -1 -1
```

then the unit number for the MC68010 (Sun2) public partition is 0 and the unit number MC68020 (Sun3) public partition is 1.

5. A link to the proper tftpbooting file in the directory */tftpboot* must be created. If the architectures of the target machine and the remote host differ, the remote host will not have the proper tftpbooting files. For example, if the remote host is a Sun2 and the target machine is a Sun3, then the remote host will have tftpbooting files for Sun2s only. In this case, you must obtain Sun3 tftpbooting files from another Sun3 or from the Sun3 installation tapes. If there is another Sun3 on the network that is running Release 3.0, issue the following command:

```
host#  rcp machine:/tftpboot/ndboot.\* /tftpboot
```

If there is not another Sun3, then the files must be obtained from the installation tapes (see *Appendix G*). Position the tape to the *pub files* and use the *tar* command to extract the *./tftpboot* directory. Remember that to position a tape to file *n*, you must forward space the tape *n*-1 files. For example, if the *pub files* is the eighth tape file, issue the following commands:

```
host# cd /
host# mt -f /dev/nrtape0 rew
host# mt -f /dev/nrtape0 fsf 7
host# tar xfvbp /dev/nrtape0 126 ./tftpboot
```

Now the link must be created. The name of the link is the target machine's Internet address converted to hexadecimal and using all capital letters. *adb*(1) can be used to convert the Internet address to hexadecimal. For example, if the Internet address is 192.9.200.100:

```
host% adb
0t192=X
        c0
0t9=X
        9
0t200=X
        c8
0t100=X
        64
^D
host%
```

Therefore, the file name for Internet address 192.9.200.100 is *C009C864*. The link must be to a file that is appropriate for the architecture of the target machine. If the target machine is a Sun3, the link should be made to *ndboot.sun3.pub0* or *ndboot.sun3.pub1* and for a Sun2, the link should be made to *ndboot.sun2.pub0* or *ndboot.sun2.pub1*. If *pub#* is zero, use the file ending with *pub0* and if *pub#* is one, use the file ending with *pub1*. For example, if the target machine is a Sun3 and *pub#* is zero, the link command would be:

```
host#  cd /tftpboot
host#  ln -s ndboot.sun3.pub0 C009C864
```

6. The files required for remote booting must be installed on the public partition that is being used for booting. There are several scenarios here. The remote host may be either a file server or a standalone machine and the architectures of the target machine and remote host may or may not be the same. In the following sequences of commands */pubarch* refers to the directory name of the public partition that is being used for booting, (typically */pub.MC68010* or */pub.MC68020*). *machine* refers to another machine of the same architecture as the target machine. If there is not another machine on the network with the same architecture as the target machine, the files must be obtained from the installation tapes. The previous subsection explains how to obtain files from the installation tapes. *pub_partition* refers to the disk controller abbreviation, disk unit number and partition letter of the partition that is being used for booting (for example, **xy0d**). */bootfile* refers to the file **/pub.boot** if the target machine is a MC68010 (Sun2) and the file **/boot** if the target machine is a MC68020 (Sun3).

   If the remote host is a file server and the architecture being served by the public partition is the same as the target machine's architecture, then all the files are in place and you may skip the rest of this subsection.

   If the remote host is a file server and the architectures differ, then the files are in place but they are for the wrong architecture. The files must be moved and replaced by the correct ones. The correct files can be obtained from another machine of the appropriate architecture on the network; or they may be obtained from the installation tapes. The example shows how to obtain them from another machine:

```
host#  cd /pubarch
host#  mv stand stand.orig
host#  mv boot boot.orig
host#  mkdir stand
host#  rcp machine:/stand/diag stand
host#  rcp machine:/stand/copy stand
host#  rcp machine:/bootfile /pubarch/boot
host#  rcp machine:/usr/mdec/bootnd /tmp
host#  cd /usr/mdec
host#  installboot /tmp/bootnd /dev/pub_partition
host#  sync
host#
```

If the remote host is a standalone machine and its architecture is the same as the target machine's architecture, issue the following commands.

```
host#  cp /stand/diag /usr/stand
host#  cp /stand/copy /usr/stand
host#  ln -s /usr /pub
host#  cp /bootfile /pub/boot
host#  cd /usr/mdec
host#  installboot bootnd /dev/pub_partition
host#  sync
host#
```

If the remote host is a standalone machine and its architecture differs from the target machine's architecture, you must obtain files that are appropriate for the target machine. The correct files can be obtained from another machine of the appropriate architecture on the network; or they may be obtained from the installation tapes. The example shows how to obtain them from another machine:

```
host#  cd /usr
host#  mv stand stand.orig
host#  mkdir stand
host#  rcp machine:/stand/diag stand
host#  rcp machine:/stand/copy stand
host#  ln -s /usr /pub
host#  rcp machine:/bootfile /pub/boot
host#  rcp machine:/usr/mdec/bootnd /tmp
host#  cd /usr/mdec
host#  installboot /tmp/bootnd /dev/pub_partition
host#  sync
host#
```

B.5. Determining Network Information

For later phases of remote installation, you need to know the remote host's host number and the target's hardware Ethernet address. You must obtain this information now.

1. To determine the remote host's hexadecimal host number, find its entry in its own */etc/hosts* file. As you remember, entries consist of a machine's full Internet address (network number followed by host number) and name, for

example:

```
192.9.200.48    augustus
192.9.200.50    julius
192.9.200.52    claudius
```

Here, julius' Internet address is 192.9.200.50; its network number is 192.9.200, and its host number (in decimal) is 50. Be sure to remember the Internet address because you will need it during *Setup*. If you are using the yellow pages, you can find the Internet address with `ypmatch julius hosts`.

Since host numbers in */etc/hosts* are in decimal, and you need the remote host's host number in hexadecimal, you will need to convert. You can use *adb* for this if you wish:

```
host% adb
0t host_number_in_decimal = X
host_number_in_hex
^D
host%
```

2. To obtain the hardware Ethernet address of the target, power up the target workstation. You will see the PROM Monitor's power-up banner — which includes the hardware Ethernet address — and then the machine will start to auto boot. Stop the auto boot immediately by typing the appropriate abort sequence (if you don't know the abort sequence for the target machine, please see Chapter 2, *Abort Procedure*):

```
Self Test completed successfully.

Sun Workstation, [model type], [keyboard type]
ROM Rev N, some_number_MBytes memory installed
Serial #some_number, Ethernet address xx:xx:xx:xx:xx:xx

Auto-boot in progress . . .

[ abort by typing the appropriate abort sequence here ]

Abort at some address
>
```

Copy down the displayed Ethernet address.

B.6. Loading the Mini UNIX System on the Remote Host

Next, you copy the mini UNIX file system from the distribution tape to the remote host's public partition.

1. Load distribution tape 1. If you have any questions about loading the tape, see the previous chapter, *Loading the Bootstrap Program.*

2. Type the following on the remote host. Remember to replace *tape* with **mt** for the nine-track tape, **ar** for the Archive quarter-inch tape, or **st** for the SCSI tape controller. Replace *pub* with **pub** if the remote host is a standalone machine and either **pub.MC68010** or **pub.MC68020** if the remote host is a file server. If you are using a nine-track half-inch tape, use **20** for *blk_factor* ('bs=20b'); use **126** for a 1/4-inch tapes ('bs=126b'):

```
host# mt -f /dev/nrtape0 rew
host# mt -f /dev/nrtape0 fsf 5
host# dd if=/dev/nrtape0 of=/pub/minifs bs=blk_factorb
host# sync
host#
```

This takes about seven minutes using a 1/2'' tape, and fifteen minutes using a 1/4'' cartridge.

B.7. Using *diag* to Format and Label the Target Machine's Disk

Now, you start to work on your target machine, and install UNIX from the remote host. The first step is to format your target's disk(s) with the *diag* utility.

Procedures for using *diag* in this remote installation are identical to those for a standard UNIX installation with one exception: during a standard installation you boot *diag* from the distribution tape; here, you boot from your remote host.

Boot *diag* from the remote host with the following boot command. Remember to replace *e_interface* with the appropriate abbreviation for your Ethernet controller (**ec** for the 3COM Ethernet Controller, **ie** for the Sun-2 Ethernet Controller, or **le** for the Sun3/50); replace *host_number* with the remote host's hexadecimal host number (obtained earlier). If you have more than one Ethernet Controller Board in your system, and you are booting from the second, third, etc., replace the first "**0**" in the command with the controller's address on the Multibus (in hex).

```
> b e_interface (0,host_number,0) stand/diag
```

When you type this, the monitor boots *diag* from the network disk server. When *diag* starts up, it displays a sign on message:

```
Version sccs version_number and date
Disk Initialization and Diagnosis

When asked if you are sure, respond with 'y' or 'Y'
```

B.8. Loading the Mini UNIX System

When you're done formatting and labeling your disk(s), you're ready to load the mini UNIX system from the remote host to your disk. To do this, you use the standalone *copy* program which you boot from the remote host. Proceed as follows.

1. Boot the standalone *copy* program from the remote host with the following commands. Replace *e_interface* with the proper device abbreviation for your Ethernet controller, and replace *host_number* with the remote host's host number (in hex). The new *Setup* only works from the first Ethernet board. Also, if you are not booting from the first Ethernet Controller Board in your system, use the board's Multibus address (in hexadecimal) rather than "0" in the boot command:

```
> b e_interface (0,host_number,0) stand/copy
Boot: e_interface (0,host_number,0) stand/copy
Load: e_interface (0,host_number,0) boot

Boot: e_interface (0,host_number,0) stand/copy
[ ...messages displaying sizes of copy program... ]
```

2. The *copy* program prompts you for the source (`From:`) and destination (`To:`) of the copy. When you respond to its queries, remember to substitute the correct device abbreviations for *e_interface* and *disk*:

```
Standalone Copy
From: e_interface(0,host_number,0)minifs
To: disk(0,0,1)
```

NOTE *If trying to boot a Sun-3 machine from a Sun-2 server, the partition number has to be* **1**.

Copying in the mini UNIX system takes about seven minutes using a half-inch tape, and about fifteen minutes using a quarter-inch cartridge. This process loads the mini UNIX file system into the swap area on the disk. It takes about six minutes to do the copy over the Net. When it completes, the *copy* program returns control to the monitor:

```
Copy completed - 524880 bytes
Standalone Copy
From:
>
```

Now abort your system.

B.9. Booting the Mini UNIX System

The next step is to boot UNIX in single-user state and specify the location of its root file system.

1. First, bring in the main boot program:

```
> b e_interface(0,host_number,0)boot -a
```

2. Now you can tell the the bootstrap program to boot the mini UNIX system from your own disk. Because this boot is an unusual one, you must specify the **–a** (for ask me) option on the boot command, and also the **–s** (come up single user) option, as follows:

```
Boot: disk(0,0,1)vmunix -as
Size: 366592+61440+98828 bytes  [ numbers vary with system level ]
Sun UNIX 4.2 Release 3.0 (GENERIC) #145: Mon Feb 17 20:35:
Copyright (c) 1985 by Sun Microsystems, Inc.
[ ...about a dozen lines of configuration messages... ]
root device?
```

3. As the mini UNIX system comes up, it displays some messages about the configuration of the system on which it is running, and finally queries you, asking for its root file system. The root file system at this stage is "*disk* **0***", which has a special meaning to the mini UNIX system. Since this notation looks ambiguous, let me specify: if you have a Xylogics disk controller, your root device is **xy0*** ; and if you have a SCSI disk controller, it is **sd0*** — the asterisk is part of the device name:

NOTE *If you have an Interphase controller, it is* **ip0***.

```
root device? disk0*
```

Depending upon your hardware, you may be asked to set the date at this point:

```
using number buffers containing number bytes of main memory
WARNING: no tod clock -- CHECK AND RESET THE DATE!
Single user boot - - fsck not done
```

If so, continue with the section *Setting the Date* in Chapter 2, and return to this point when you've finished that subsection; otherwise, continue with the next section.

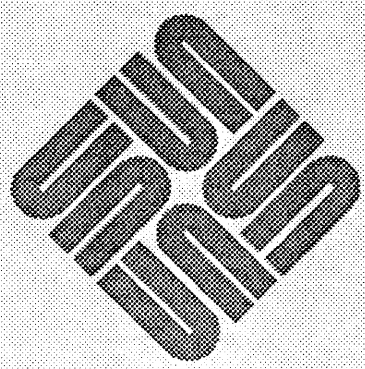
At this point, proceed to chapters 3-5 of this manual to invoke the *Setup* program.

If your remote host is a file server and if earlier you obtained booting files from another machine or the installation tapes, you should now reinstall the original files. In the following sequence of commands, */pubarch* refers to the directory name that was being used for booting (typically */pub.MC68010* or */pub.MC68020*). *pub_partition* refers to the disk controller abbreviation, disk unit number and partition letter of the partition that was being used for booting (for example, xy0d).

```
host# cd /pubarch
host# rm -rf stand
host# mv stand.orig stand
host# rm boot
host# mv boot.orig boot
host# cd /usr/mdec
host# installboot bootnd /dev/pub_partition
host# sync
host#
```

C

Sample Configuration File for the Sun-2 and Sun-3 Machines



C

Sample Configuration File for the Sun-2 and Sun-3 Machines

The following pages present sample configuration files for the basic Sun-2 and Sun-3 machines.

C.1. Sample Configuration File for a Sun-2 machine configuration

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

config      vmunix      swap generic

pseudo-device   pty
pseudo-device   bk
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win128
pseudo-device   dtop4
pseudo-device   ms3
pseudo-device   kb3

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

```
controller  mbio 1 at nexus ?   # Multibus io

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

controller  ipc0 at mbio ? csr 0x40 priority 2
controller  ipc1 at mbio ? csr 0x44 priority 2
disk        ip0 at ipc0 drive 0
disk        ip1 at ipc0 drive 1
disk        ip2 at ipc1 drive 0
disk        ip3 at ipc1 drive 1
controller  xyc0 at mbio ? csr 0xee40 priority 2
controller  xyc0 at vme16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller  xyc1 at mbio ? csr 0xee48 priority 2
controller  xyc1 at vme16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk        xy0 at xyc0 drive 0
disk        xy1 at xyc0 drive 1
disk        xy2 at xyc1 drive 0
disk        xy3 at xyc1 drive 1
controller  sc0 at mbmem ? csr 0x80000 priority 2
controller  sc0 at vme24 ? csr 0x200000 priority 2 vector scintr 0x40
disk        sd0 at sc0 drive 0 flags 0
disk        sd1 at sc0 drive 1 flags 0
tape        st0 at sc0 drive 32 flags 1
disk        sd2 at sc0 drive 8 flags 0
tape        st1 at sc0 drive 40 flags 1
controller  sc1 at mbmem ? csr 0x84000 priority 2
disk        sd2 at sc1 drive 0 flags 0
disk        sd3 at sc1 drive 1 flags 0
tape        st1 at sc1 drive 32 flags 1
#disk       sf1 at sc1 drive 8 flags 2
device      sky0 at mbio ? csr 0x2000 priority 2
device      sky0 at vme16 ? csr 0x8000 priority 2 vector skyintr 0xb0
device      zs0 at obio 1 csr 0x2000 flags 3 priority 3
device      zs0 at obio 2 csr 0x7f2000 flags 3 priority 3
device      zs1 at obmem 1 csr 0x780000 flags 0x103 priority 3
device      zs1 at obio 2 csr 0x7f1800 flags 0x103 priority 3
device      zs2 at mbmem ? csr 0x80800 flags 3 priority 3
device      zs3 at mbmem ? csr 0x81000 flags 3 priority 3
device      zs4 at mbmem ? csr 0x84800 flags 3 priority 3
device      zs5 at mbmem ? csr 0x85000 flags 3 priority 3
device      mti0 at mbio ? csr 0x620 flags 0xffff priority 4
device      mti1 at mbio ? csr 0x640 flags 0xffff priority 4
device      mti2 at mbio ? csr 0x660 flags 0xffff priority 4
device      mti3 at mbio ? csr 0x680 flags 0xffff priority 4
device      mti0 at vme16 ? csr 0x620 flags 0xffff priority 4 vector mtiintr 0x88
device      mti1 at vme16 ? csr 0x640 flags 0xffff priority 4 vector mtiintr 0x89
device      mti2 at vme16 ? csr 0x660 flags 0xffff priority 4 vector mtiintr 0x8a
device      mti3 at vme16 ? csr 0x680 flags 0xffff priority 4 vector mtiintr 0x8b
```

```
device      ie0 at obio 2 csr 0x7f0800 priority 3
device      ie0 at mbmem ? csr 0x88000 priority 3
device      ie1 at mbmem ? csr 0x8c000 flags 2 priority 3
device      ie1 at vme24 ? csr 0xe88000 priority 3 vector ieintr 0x75
device      ec0 at mbmem ? csr 0xe0000 priority 3
device      ec1 at mbmem ? csr 0xe2000 priority 3
controller  tm0 at mbio ? csr 0xa0 priority 3
controller  tm0 at vme16 ? csr 0xa0 priority 3 vector tmintr 0x60
controller  tm1 at mbio ? csr 0xa2 priority 3
controller  tm1 at vme16 ? csr 0xa2 priority 3 vector tmintr 0x61
tape        mt0 at tm0 drive 0 flags 1
tape        mt1 at tm1 drive 0 flags 1
controller  xtc0 at mbio ? csr 0xee60 priority 3
controller  xtc0 at vme16 ? csr 0xee60 priority 3 vector xtintr 0x64
controller  xtc1 at mbio ? csr 0xee68 priority 3
controller  xtc1 at vme16 ? csr 0xee68 priority 3 vector xtintr 0x65
tape        xt0 at xtc0 drive 0 flags 1
tape        xt1 at xtc1 drive 0 flags 1
device      ar0 at mbio ? csr 0x200 priority 3
device      ar1 at mbio ? csr 0x208 priority 3
device      gpone0 at vme24 ? csr 0x210000
device      cgtwo0 at vme24 ? csr 0x400000
device      cgone0 at mbmem ? csr 0xec000 priority 3
device      bwtwo0 at obmem 1 csr 0x700000 priority 4
device      bwtwo0 at obio 2 csr 0x0 priority 4
device      bwone0 at mbmem ? csr 0xc0000 priority 3
device      vp0 at mbio ? csr 0x400 priority 2
device      vpc0 at mbio ? csr 0x480 priority 2
device      vpc0 at vme16 ? csr 0x480 priority 2 vector vpcintr 0x80
device      vpc1 at mbio ? csr 0x500 priority 2
device      vpc1 at vme16 ? csr 0x500 priority 2 vector vpcintr 0x81
device      pi0 at obio 1 csr 0x1800
device      des0 at obio 1 csr 0x1000
device      des0 at obio 2 csr 0x7f1000
device      tod0 at obio 1 csr 0x3800
device      tod0 at vme24 ? csr 0x200800
```

C.2. Configuration File for the Sun-3 Machine

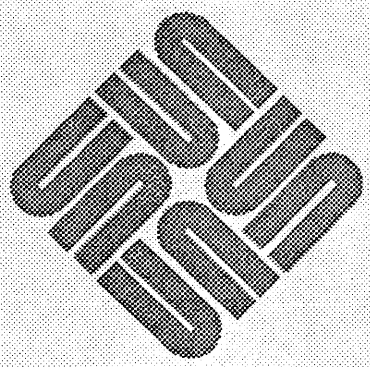
```
#
# GENERIC SUN3
#
machine     "sun3"
cpu         "SUN3_160"  # (Sun-3/160 or Sun-3/75 cpu)
cpu         "SUN3_50"
cpu         "SUN3_260"  # (Sun-3/280)
cpu         "SUN3_110"  # (Sun-3/130)
ident       GENERIC
timezone    8 dst
maxusers    4
options     INET
```

```
options     SYSACCT
options     QUOTA
options     NFS
options     NIT
options     IPCMESSAGE      # SystemV IPC Message Facility
options     IPCSEMAPHORE    # SystemV IPC Semaphore Facility
options     IPCSHMEM        # SystemV IPC Shared-Memory Facility

config      vmunix      swap generic

pseudo-device   pty
pseudo-device   bk
pseudo-device   ether
pseudo-device   loop
pseudo-device   nd
pseudo-device   win128
pseudo-device   dtop4
pseudo-device   ms3
pseudo-device   kb3

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

# connections for machine type 2 (SUN3_50)
controller  virtual 2 at nexus ?
controller  obmem 2 at nexus ?
controller  obio 2 at nexus ?

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

# connections for machine type 4 (SUN3_110)
controller  virtual 4 at nexus ?
controller  obmem 4 at nexus ?
controller  obio 4 at nexus ?
controller  vme16d16 4 at nexus ?
controller  vme24d16 4 at nexus ?
```

```
controller  vme32d16 4 at nexus ?
controller  vme16d32 4 at nexus ?
controller  vme24d32 4 at nexus ?
controller  vme32d32 4 at nexus ?

controller  xyc0 at vme16d16 ? csr 0xee40 priority 2 vector xyintr 0x48
controller  xyc1 at vme16d16 ? csr 0xee48 priority 2 vector xyintr 0x49
disk        xy0 at xyc0 drive 0
disk        xy1 at xyc0 drive 1
disk        xy2 at xyc1 drive 0
disk        xy3 at xyc1 drive 1
controller  sc0 at vme24d16 ? csr 0x200000 priority 2 vector scintr 0x40
disk        sd0 at sc0 drive 0 flags 0
disk        sd1 at sc0 drive 1 flags 0
tape        st0 at sc0 drive 32 flags 1
disk        sd2 at sc0 drive 8 flags 0
tape        st1 at sc0 drive 40 flags 1
controller  si0 at vme24d16 ? csr 0x200000 priority 2 vector siintr 0x40
controller  si0 at obio ? csr 0x140000 priority 2
disk        sd0 at si0 drive 0 flags 0
disk        sd1 at si0 drive 1 flags 0
tape        st0 at si0 drive 32 flags 1
disk        sd2 at si0 drive 8 flags 0
tape        st1 at si0 drive 40 flags 1
#disk       sf0 at si0 drive 8 flags 2
device      zs0 at obio ? csr 0x20000 flags 3 priority 3
device      zs1 at obio ? csr 0x00000 flags 0x103 priority 3
device      mti0 at vme16d16 ? csr 0x620 flags 0xffff priority 4 vector mtiintr 0x8
device      mti1 at vme16d16 ? csr 0x640 flags 0xffff priority 4 vector mtiintr 0x8
device      mti2 at vme16d16 ? csr 0x660 flags 0xffff priority 4 vector mtiintr 0x8
device      mti3 at vme16d16 ? csr 0x680 flags 0xffff priority 4 vector mtiintr 0x8
device      ie0 at obio ? csr 0xc0000 priority 3
device      ie1 at vme24d16 ? csr 0xe88000 priority 3 vector ieintr 0x75
device      le0 at obio ? csr 0x120000 priority 3
controller  tm0 at vme16d16 ? csr 0xa0 priority 3 vector tmintr 0x60
controller  tm1 at vme16d16 ? csr 0xa2 priority 3 vector tmintr 0x61
tape        mt0 at tm0 drive 0 flags 1
tape        mt1 at tm1 drive 0 flags 1
controller  xtc0 at vme16d16 ? csr 0xee60 priority 3 vector xtintr 0x64
controller  xtc1 at vme16d16 ? csr 0xee68 priority 3 vector xtintr 0x65
tape        xt0 at xtc0 drive 0 flags 1
tape        xt1 at xtc1 drive 0 flags 1
device      gpone0 at vme24d16 ? csr 0x210000
device      cgtwo0 at vme24d16 ? csr 0x400000
device      cgfour0 at obmem 4 csr 0xff000000 priority 4
device      bwtwo0 at obmem 1 csr 0xff000000 priority 4
device      bwtwo0 at obmem 2 csr 0x100000 priority 4
device      bwtwo0 at obmem 3 csr 0xff000000 priority 4
device          bwtwo0 at obmem 4 csr 0xff000000
device      vpc0 at vme16d16 ? csr 0x480 priority 2 vector vpcintr 0x80
device      vpc1 at vme16d16 ? csr 0x500 priority 2 vector vpcintr 0x81
device      des0 at obio ? csr 0x1c0000
device      fpa0 at virtual ? csr 0xe0000000
```

D

The PROM Monitor and the EEPROM



D

The PROM Monitor and the EEPROM

D.1. The EEPROM

This text covers the parameters that are used to program the EEPROM to cause the system to:

- Recognize the specified device as the primary terminal or console;
- Display the Sun logo or a custom banner during power-up
- Boot from a specified device.
- Change the console display size.
- Change the watchdog reset function.
- Vary the quantity of memory tested during self-test.
- Store a system configuration record on EEPROM.

D.2. The PROM Monitor

The boot PROM stores a program known as the "monitor", which contains self-tests and controls system operation during boot-up until the UNIX kernel takes over. The monitor program is invoked when you use one of the abort procedures defined in Section 2.2, and is identified with the > prompt.

This subsection provides information on commands entered following the monitor prompt to perform a variety of tasks, such as booting from alternate devices, changing the console output to a serial port, and so on. Commands that are self explanatory on the Monitor Help Menu are not included in this discussion; commands that require arguments and further understanding are explained following the Help menu example. Access the monitor as previously described, then enter **h** to view a "Help" table of monitor commands that looks something like this:

Table D-1 *Monitor Help Menu*

Cmd	*Parameters*	*Description*
a	*[digit]*	Open CPU Addr Reg (0-7)
b	*[dev (cont,unit,part or file)]*	Bootload a file and start it
c	*[addr]*	Continue program at this address
d	*[digit]*	Open CPU Data Reg (0-7)
e	*[address]*	Open address as 16-bit word
f	*beg_addr end_addr pattn [size]*	Fill memory with pattern
g	*[address]*	Go to given address
h		Display this help table

Table D-1 *Monitor Help Menu— Continued*

Cmd	*Parameters*	*Description*
i	*[addr]*	Open Cache Data
j	*[addr]*	Open Cache Tags
k	*[number]*	Reset (0) CPU, (1) mmu,(2) system
l	*[addr]*	Open address as a 32-bit long word
m	*[addr]*	Open the Segment Map
n	*[i/e/d]*	Cache Invalidate/Enable/Disable
o	*[addr]*	Open Addr as an 8-bit byte
p	*[addr]*	Open Page Map
q	*[addr]*	Open the EEPROM
r		Open CPU regs (i.e. PC,SR)
s	*[digit]*	Set/Query Function Code (0-7)
t	*[y/n/c]*	Trace: Yes/No/Continue
u	*[arg]*	Select Console Device
v	*beg_addr end_addr [size]*	Display Memory
w	*[addr][string]*	Vector
x		Extended Diag Tests
y	*[c cxt] [s cxt sg_addr] [p cxt pg_addr]*	Flush Cntxt/Seg/Page
z	*[addr]*	Set Breakpoint

Monitor Command Syntax

If no argument or parameter is entered after the **a, c, d, e, f, g, l, m, o, q, or r** commands (shown above), the action is performed on a default location. For instance, if you enter the **q** command alone, the first address assigned to the EEPROM is displayed.

Entering [RETURN] immediately following the display of the contents of a location causes the contents of the next location to be displayed. A space and then [RETURN] returns the monitor prompt.

The next few paragraphs provide additional information on parameter choices that affect the function of the commands listed in the Help Menu. Note that any changes made with these commands (with the exception of the q command for EEPROM programming) are valid only until the system is powered down. Parameters changed with the q command take effect when you power cycle or do a K2 reset to the system, and remain in effect until you use the q command again to change them.

NOTE *You may use either upper or lower case to enter these commands and values.*

Monitor Command **b**

If *diag* sw=0 (Normal) and it is not defined in EEPROM, typing a 'b' only will cause a probe in this order: xy, sd, ie or le. The **b** command must be followed with one of these abbreviations:

xy — Xylogics 450/451 disk
sd — SCSI disk
ie — Intel Ethernet
le — Lance Ethernet
st — SCSI tape
xt — Xylogics 472 Tape
mt — Tapemaster 9-Track Tape

The *mt, xt, sd, st and xy* arguments are followed with parentheses that optionally enclose the controller number, unit number, and partition or file number:

```
>b sd(0,0,0)
```

The default values 0,0,0 are used if you do not enter values inside the parentheses. You may also add a filename after the parentheses, to boot only the specified file.

You may boot from SCSI tape without entering anything between the parentheses, to receive a boot prompt, after which you enter the location of the file you have selected.

```
>b st()
boot: st(0,0,0)
boot: st(0,0,1)
```

Monitor Command **k**

The **k** command argument is a number that specifies the desired reset level:

0 = CPU instruction reset
1 = CPU instruction and MMU (Segment and Page Maps) Reset
2 = CPU instruction, MMU and memory reset
b = display banner message

NOTE *In systems with a Version 1.4 or earlier Boot PROM, the* **k2** *command does not inititalize the Ethernet control register or the SCC chip on the CPU board. For this reason, press the User Reset button on the CPU board edge before you use this command. The reset button invokes a watchdog reset, which reinitializes the register and the SCC. (Note that EEPROM programming may cause the system to reboot following a watchdog reset.)*

The Version 1.5 PROM corrects this problem.

Monitor Command **q**

The **q** command opens the EEPROM to allow examination or modification of configuration parameters. If you do not enter an address following the command, the content of the first address assigned to the EEPROM is presented. (EEPROM addresses are off-set, rather than complete addresses.)

```
>q[RETURN]
>EEPROM 000:some value in hex?
```

The EEPROM Programming subsection lists the addresses and values that must be entered in order to change various aspects of workstation configuration, such as Sun Logo display during boot-up, the default boot device, and so on.

To change the value of a specific EEPROM address, simply enter **q**, space, the address, and [RETURN]. When the program displays the contents of that address enter the new value, space and [RETURN]:

```
>q 1f[RETURN]
>EEPROM 01f: 10? 11 [SPACE] [RETURN]
```

If you want to modify sequential locations, simply press [RETURN] immediately following the value you just entered, and the value present in the next location will be displayed.

To exit from the modify mode, press the space bar and then [RETURN].

To verify that the change was made, re-enter the address of the location that was modified, and the program will display the contents.

Monitor Command **s**

The **s** command sets or queries the address space to be used by subsequent memory access commands. The processor function codes decode the address spaces. Argument choices represent the function codes:

0 = Reserved
1 = Device Space (User Data)
2 = Device Space (User Program)
3 = Control Space
4 = Reserved
5 = Device Space (Supervisor Data)
6 = Device Space (Supervisor Program)
7 = CPU Space

If you do not enter a function code number, the current setting (either 1 or 5) is displayed, and entry of the monitor **o** command, for example, would cause the program to look for the specified address in either user or supervisor data space. Conversely, if you reset the function code number, you could query registers in the space represented by the number entered. For example, entering

```
>s 3
```

would allow you to read the bus error or system enable register, which are located in Control Space.

Monitor Command **t**

A **y** argument to the **t** command causes the monitor to display one instruction at a time. The **c** argument causes a continuous display of program instructions, which can be momentarily stopped with [CTRL] s and restarted with [CTRL] q. This command is primarily used for development purposes.

Monitor Command **u**

The **u** command requires an argument to specify from which device(s) you want the system to expect input or which device(s) will display output.

Table D-2 *UART Definitions*

Name	*Definition*
uki	current input source
uso	current output source
ua9600	serial port A baud rate
ub9600	serial port B baud rate
uuEEC800	virtual Sun-2 address of UART
uuFE02000	virtual Sun-3 address of UART
ue	echo mode

If you do not enter an argument after the u command, the program will display the current settings. If no serial port is specified when changing baud rates, the baud rate of the current input device is changed. The default serial port baud rate is 9600 unless specified by EEPROM. Also Serial Port B is 1200 Baud if the diagnostic switch = 1.

Upon normal power-up (diag switch is in NORM position), the default console input device is the Sun keyboard, unless the EEPROM has specified another default input device. If the keyboard is unavailable, the system looks to serial port A for for input.

The default console output device is the Sun monitor (subject to change through EEPROM programming). If the workstation is a Sun-3/160C and a color board is unavailable, the program will look for a monochrome monitor as an output device.

You may alter the existing I/O settings while you are in the monitor mode, using the commands listed below; however, the default settings will be reinstated when the system is power cycled or K2 reset is used.

Monitor Commands to Change I/O:

- Enter **u a** or **u b** to select serial port A or B as the input and output device.
- Enter **u aio** or **u bio** to select serial port A or B as the input and output device.
- Enter **u ai** or **u bi** to select serial port A or B for input only.
- Enter **u ao** or **u bo** to select serial port A or B for output only.
- Enter **u k** to select the keyboard for input.
- Enter **u ki** to select the keyboard for input.
- Enter **u s** to select the screen for output.
- Enter **u so** to select the screen for output.
- Enter **u ks, sk** to select the keyboard for input and the screen for output.
- Enter **u a***baud rate* or **u b***baud rate* to set the serial port speed.
- Enter **u e** to cause the output to echo the input.
- Enter **u ne** to cause the output **not** to echo the input.

- Enter **u** *address* to set the serial port virtual address.

Monitor command **v**

This command allows you to view the hexadecimal values, as well as the ASCII equivalent, present in the addresses you enter. You must enter the complete address, as opposed to an offset address. (EEPROM addresses, for example, are offset.) You specify the format with a second argument, selected from these:

b for byte format
w for word (16-bit) format
l for long word (32-bit) format

For example, to view the contents of locations 0x00 to 0xff in 16-bit format, enter:

```
>v 0 ff w[RETURN]
```

If you do not specify a format, the display will be in long-word format.

To suspend scrolling of the display, press [RETURN], then press it again to restart the display. To quit out of the viewing process and display the monitor prompt, press the [SPACE] bar.

Monitor Command **w**

The **w** command should be used for programming purposes to cause the specified arguments to be passed to your routine. This is used largely for development purposes.

Monitor Command **x**

The **x** command presents the menu of Extended Diagnostic Tests that can also be accessed during diagnostic boot-up. The Sun-3 Diagnostic Manual provides more information on those tests.

D.3. EEPROM Programming

The CPU board has an electrically erasable, programmable memory (EEPROM) chip that can be accessed through the boot PROM monitor in order to change system configuration parameters. See *eeprom*(8s).

These procedures are for systems that contain a Version 1.4 or later Boot PROM.

This heading covers entry of EEPROM parameters that control these functions:

- vary the quantity of memory tested during self-test;
- change the console display size parameters;

NOTE *The monochrome console is for the Sun-3/75, Sun-3/160, Sun-3/180, Sun-3/260 and Sun-3/280 only. This also requires a corresponding shunt and PAL change to the CPU board.*

- select the action that follows a watchdog reset;
- boot from a specified device with diagnostics switch on NORM, or poll the devices;
- recognize the specified device as the primary terminal or console;
- display the Sun logo or a custom banner during power-up;
- turn the keyboard "click" on or off;
- boot a selected program from a specific device with diagnostics switch on DIAG;
- Store a system configuration record on EEPROM.

This table shows the EEPROM address locations for the parameters listed above.

Table D-3 *EEPROM Address Space Allocation*

Location	*Function*
0x14	MBytes of installed memory
0x15	MBytes memory tested
0x16	Display screen size
0x17	Watchdog reset function
0x18 to 1D	Choose polling or specify boot device
0x1F	Set console (primary terminal) I/O
0x20	Choose boot display banner
0x21	Turn keyboard "click" on or off
0x22 to 0x26	Select diagnostic boot device
0x28 to 0x4f	Store diagnostic boot path
0x50	Number of columns
0x51	Number of rows
0x58	SCC Port A Baud Rate
0x59-0x5A	Baud Rate in hex
0x5B	Port A DTR/RTS select
0x60	SCC Port B
0x61-0x62	Baud rate in hex
0x63	Port B DTR/RTS
0x68 to 0xb7	Store custom banner
0xbc to 0x18B	Store system configuration

NOTE *You may need to refer to Chapter 2 for information on disk or tape partitions, file numbers, and so on.*

These paragraphs will describe how to open the offset addresses shown in the ASCII Hex Table at the end of this appendix in order to read and modify the contents. You will first gracefully power-down the system then power-up and enter the monitor program after self-tests are successful: Simultaneously press the **L1** key (upper left-hand corner of the keyboard) and the **A** key. You have now aborted your present session and accessed the monitor program. The monitor displays as its characteristic prompt the "greater than" symbol:

```
>
```

NOTE *You may use either upper or lower case characters to enter the* **q** *command or any of the hexadecimal values needed for EEPROM programming.*

To access the EEPROM now, type the letter **q**, followed by the EEPROM address that you want to examine or modify.

```
>q 1f
```

The program will now display the contents of address 0x1f in the EEPROM, which is the value that determines which device is considered the system console:

```
>EEPROM 01F: some value?
```

To update the contents of this location, enter the new value immediately following the displayed value, and then a carriage return. You have now written the new value to address 0x1f — and automatically incremented to the next address. In this example, you are replacing the value 0x00 with the value 0x10:

```
>EEPROM 01F: 00? 10 [RETURN]
>EEPROM 020: some value?
```

Now, to stop incrementing EEPROM addresses and receive the monitor prompt again, press [SPACE] and [RETURN]. Now you can open a new EEPROM address, or verify the change you just made:

```
>EEPROM 020: some value? [SPACE] [RETURN]
>q 1f [RETURN]
>EEPROM 1F: 10? [SPACE] [RETURN]
>
```

To see the effect of your EEPROM programming, you must reinitialize the system. The next few headings show you how to change the parameters listed at the beginning of this section.

Setting Quantity of Memory Tested

The quantity of memory present in the system is recorded at EEPROM location 0x14, and can be viewed when that location is opened(a valid number is the number of megabytes of memory in the system):

```
>q 14[RETURN]
EEPROM 014:some value?[RETURN]
```

The quantity of memory to be tested during boot-up with the diagnostic switch in NORM position is recorded in location 0x15, and should agree with the value in location 0x14, if all of memory is to be tested. To view the contents of address 0x15, press [RETURN] immediately after the quantity of memory is displayed, and the prompt will look like this:

```
EEPROM 015:some value?
```

To change the value of location 0x15, simply enter the new value after the question mark, press [SPACE] and return to the monitor prompt.

Changing Monitor Display Size

The contents of EEPROM address 0x16 help determine whether the video monitor display size will be the standard 900x1152, or 1024x1024 pixels.

Table D-4 *0x16 EEPROM Address Contents*

Address	*Description*
0x00	*=900x1152 standard*
0x12	*=1024x1024 display*
0x13	*=1600x1280 display*

The contents of address 0x16 will read 0x00 for the standard display, and 0x12 for the 1024x1024 display and 0x13 1600x1280 display. To change to the 1024x1024 display, enter:

```
>q 16[RETURN]
EEPROM 016:00?12[SPACE][RETURN]
>
```

To complete the display change, PALs on the CPU board must be changed. Contact your Sun sales representative for more information on this option.

Changing Results of a Watchdog Reset

EEPROM address 0x17 determines whether a watchdog reset will result in a system boot, or whether it will simply invoke the PROM monitor.

Write 0x12 in location 0x17 to select boot-up after a watchdog reset; write 0x00 to that location to invoke the monitor after a watchdog reset.

Selecting the Boot Device

Ordinarily, when the system boots with the diagnostic switch in NORM position, and EEPROM address 0x18 contains the value 0x00, the monitor program polls boot devices in this order:

- It first checks to see if a XY disk controller is present, and if so, attempts to boot from it. If it is unable to locate that one, it will look for the SD disk controller next.
- If the polling sequence does not find a disk controller, the Ethernet controller on the CPU board is polled and becomes the boot device.

Boot Device Codes:
xy = Xylogics 440/450/451 disk
sd = SCSI disk
ie = Ethernet
st = SCSI tape
mt = Tape Master 9-Track
xt = Xylogics 472 Tape

To delete the polling sequence and change the normal boot device, you must open the EEPROM to address location 0x18 and change the contents from 0x00 (which causes polling) to 0x12 (to delete polling). You must then fill the next two address locations with a two-byte ASCII string that represents the new boot device. (Device codes are shown in the left margin.) Obtain hex equivlent of ASCII characters from the table at the end of this appendix. Finally, you must open the next three addresses and write values that represent the controller, unit and partition or file number.

For example, this read/write sequence would make Ethernet the default boot device:

ASCII equivalent of hex values shown here:

69 = i
65 = e

```
>q 18[RETURN]
EEPROM 018:some value? 12[RETURN]
EEPROM 019:some value? 69[RETURN]
EEPROM 01A:some value? 65[RETURN]
EEPROM 01B:some value? 0[RETURN]
EEPROM 01C:some value? 0[RETURN]
EEPROM 01D:some value? 0[SPACE][RETURN]
>
```

The sequence shown above represents the entry: ie(0,0,0).

Setting Up Console I/O

To open the EEPROM address that determines which device will display the boot-up dialogue and act as console or the primary terminal — the device over which you and the processor will communicate — enter:

```
>q 1f[RETURN]
```

After the program displays the contents of EEPROM address 0x1f, enter the hexadecimal value assigned to the device you want to act as the console, as indicated below:

00 = monochrome monitor
10 = serial port A
11 = serial port B
12 = color monitor/grayscale monitor

To test this set-up, press the space bar and [RETURN], then reboot with the diagnostic switch in NORM position. If you selected one of the serial ports, the boot-up informative messages should appear on the screen belonging to a terminal that is connected to the indicated port.

If you selected the color monitor, a color board must be present in the pedestal cardcage, with the appropriate cables connected.

Changing the Banner

To change the Sun banner that appears during boot-up, enter

```
>q 20[RETURN]
```

which will open the EEPROM address that pertains to that display. The monitor will echo the address you entered and display 0x00 for Sun Logo display, which you must change to the hex value 12, to delete the Sun banner display during boot-up:

```
>EEPROM 020:00 12 [SPACE] [RETURN]
```

You have just instructed the monitor program not to display the banner that includes the Sun Logo. To create a new banner, you must now write the hex value for ASCII "space" characters (0x20) or the hex equivalent of a new ASCII string into locations 0x68 to 0xb7.

NOTE *If you do not fill locations 0x68 to 0xb7 (80 characters) with appropriate hex numbers after you have changed the contents of address 0x20 to 0x12, you may see garbled information in place of the Sun banner when you boot-up.*

Put zero at the end of the banner 0xb7.

To create a custom banner, after you have written the value 12 into address 0x20, press the space bar and then return and access address 0x68:

```
>q 68[RETURN]
>EEPROM 068:some value?
>
```

Then begin entering the hexadecimal code for the new banner immediately after each address is displayed. There is an ASCII/Hex conversion chart at the end of this appendix to aid you in selecting values that represent the characters you want displayed in place of the Sun banner.

You can use the monitor `v` (view) command to view the characters you have just written, but you must enter the complete address where the logo begins, rather than the offset address 0x68. The complete address for this EEPROM location begins at 0xfe04000.

NOTE *The banner you have created will be displayed the next time you power-up. If you entered all blanks (except for the carriage return and line feed values), there will be no banner upon power up.*

Turning Keyboard Click On or Off

The value in EEPROM location 0x21 determines whether or not the Sun-3 keyboard will make a "clicking" noise when you type. Use the procedures described at the beginning of this EEPROM section to access location 0x21 and view or change the contents:

To turn the keyboard click OFF:

```
>q 21
EEPROM 021:012 00
```

To turn the keyboard click ON:

```
>q 21
EEPROM 021:000 12
```

Selecting the Diagnostic Boot Device

If you switch the diagnostic switch to the DIAG position and then boot without specifying either `vmunix` or a boot device and path that indicates where the selected diagnostic program is located, the system may display an error message and drop into monitor mode, displaying the ">" prompt.

You can program the EEPROM so that, when the diagnostic switch is in DIAG position, and you enter

```
>b
```

with no arguments, the monitor program will boot the desired program, this may

be a diagnostic or VMUNIX. To do this, you must first enter parameters that specify whether to boot from Ethernet, disk or tape. Then you must enter the path that tells the boot PROM where it can find the selected program in your file systems (see the heading, *Setting the Diagnostic Boot Path*).

If you want to invoke the monitor program during a diagnostic boot, you simply enter zeroes in addresses 0x22 through 0x4f.

The EEPROM addresses 0x22 and 0x23 contain the hexadecimal ASCII values that represent the selected boot device. For example, locations 0x22 and 0x23 might contain the values 0x73 and 0x64, respectively — the hex equivalent of ASCII "sd", for SCSI disk.

The next three bytes (0x24 through 0x26) contain the hexadecimal values that literally represent the controller, unit and partition or file numbers specific to the chosen device. In other words, these bytes represent the part of the boot command that is enclosed in parentheses:

```
>b sd (0,0,0)
```

To set the "diagnostic boot device", obtain the monitor prompt as described at the beginning of this section, and access EEPROM address 0x22:

```
>q 22
EEPROM 022:00
```

A hexadecimal value will be displayed as the contents of address 0x22, as shown above.

To specify a diagnostic boot device, simply enter the hex equivalent of the first letter that represents the new device — you can use the ASCII/Hex Conversion Chart at the end of this section — press [RETURN] and then, when the contents of location 0x23 are displayed, enter the value for the second letter than represents the device. (Entering **b ?** after the > prompt will give you a list of possible boot device choices.)

NOTE *To ensure that diagnostics are* not *invoked after self-test, and that the system drops into monitor mode during a boot-up with the diagnostic switch in DIAG position, enter zeroes instead of hex values.*

If you have entered values to specify a boot device, press [RETURN] again and write the controller, unit and partition or file numbers into addresses locations 0x24, 0x25 and 0x26, respectively. You do not need to use ASCII characters for these values. Simply enter one hexadecimal value in each location, and press [RETURN].

If you have specified the monitor mode, enter zeroes in all three locations.

Now, if you have specified a boot device, you are ready to use up to forty bytes, beginning with EEPROM address 0x28, to represent your complete diagnostic boot path. The values you enter represent any of the following:

1. the location that you have selected to store the selected diagnostic program,
2. the acronym *vmunix*, which means "virtual memory UNIX ",

3. or, zeroes to invoke the monitor program after self-test, during a diagnostic boot.

Setting the Diagnostic Boot Path

You have just programmed the EEPROM to tell the boot PROM monitor program what device, controller, unit and partition or file number to boot from during a diagnostic boot-up.

Now you must use the same method to tell the monitor one of the following:

1. Where in your file systems the selected diagnostic program is located;
2. that you want UNIX to boot after self-test when the diagnostic switch is in DIAG position and you have not selected Extended Tests;
3. or that you want the monitor program to be invoked after self-test when you do not select Extended Tests.

These 40 bytes represent a character buffer for a user specified diagnostic path (i.e. /stand/diag). They will be ASCII characters represented by Hex values with 0x00 being the terminator of the string.

Table D-5 *Address [0x028-0x04F]*

ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
ASCII	ASCII	ASCII	ASCII	ASCII	ASCII	ASCII
ASCII	ASCII	ASCII	ASCII	0x00		

For example, if you want to tell the monitor that the selected diagnostic program is located in the `/usr/stand` directory, you would first open EEPROM address 0x000028:

```
>q 28 [RETURN]
```

The program would then display the contents of location 0x000028, after which you would begin entering the hexadecimal equivalent of your diagnostic path. (The ASCI Hex Table at the end of this appendix can help you with ASCII/Hex conversions.) You are allowed forty bytes, or the equivalent of forty characters for your path name, and the last value you enter must be 0x00.

To enter `/usr/stand`, for example:

```
EEPROM 028: some value 2f[RETURN] (for /)
EEPROM 029: some value 75[RETURN] (for u)
EEPROM 02A: some value 73[RETURN] (for s)
EEPROM 02B: some value 72[RETURN] (for r)
EEPROM 02C: some value 2f[RETURN] (for /)
EEPROM 02D: some value 73[RETURN] (for s)
EEPROM 02E: some value 74[RETURN] (for t)
EEPROM 02F: some value 61[RETURN] (for a)
EEPROM 030: some value 6e[RETURN] (for n)
EEPROM 031: some value 64[RETURN] (for d)
EEPROM 032: some value 00[RETURN] (to end)
```

To select UNIX, use the ASCI Hex Table to determine the hexadecimal equivalent for each character in the acronym *vmunix*, and enter those values in the addresses beginning with 0x28. All other locations in the forty-byte block should be zeroes.

If you are specifying that the monitor program is invoked during a diagnostic boot, ensure that locations 028 through 0x4f contain zeroes.

Recording System Configuration

EEPROM locations 0xbc through 18B contain values that represent the types of factory installed printed circuit boards present in the system as well as information on the internal configuration of those boards and external devices that they control.

This subsection describes the type of information that can be encoded and stored in these address locations. If a board is not discussed here, the only byte used at this time is the first byte of the block assigned to that board, which identifies it.

There are 208 bytes used to represent the hardware configuration of the system. The system configuration is divided into 12 slot configuration blocks of 16 bytes, and 1 sentinel block of 16 bytes to denote the end of the configuration table. The first 16 bytes will represent the board specific information in slot #1 of the system. The second 16 bytes will represent slot #2. This process will continue through slot #12 which is the maximum number of slots available in the current system. All empty slots will be identified with a Board Type of zero (0x00). The sentinel block (0xFF) resides in the 13th configuration block which is non-existent today, but this table can be expanded in the future to support card cages with more slots available. The sentinal block would be the 7th configuration for a 6 slot package; 4th configuration for 3 slot; and so forth. The layout of the system configuration is illustrated in the following table (B/S = Board Specific). The configuration blocks for the current boards are detailed at the end of this appendix.

Table D-6

Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
Board Type	B/S	B/S	B/S	B/S	B/S	B/S	B/S
B/S	B/S	B/S	B/S	B/S	B/S	B/S	B/S
0xFF	N/U	N/U	N/U	N/U	N/U	N/U	N/U
N/U	N/U	N/U	N/U	N/U	N/U	N/U	N/U

If the configuration information is entered to match the board slot assignments, then blocks representing empty slots must contain all zeroes.

The first byte of each eight-byte block refers to the pc board type. Hexadecimal values 0x1 through 0xF identify the various boards that may be installed in the card cage. Values 0x80 through 0xfe are to be used to identify boards other than those installed by Sun. Values 0x10 through 0x7f are reserved for future expansion. The remaining 15 bytes contain information that varies, dependent on the board type.

The block following the last slot available in the card cage should contain the value 0xff in the first byte, to indicate that this is the end of the configuration.

The value in the first byte of each block is decoded as follows:

Table D-7 *Board Definitions*

Type	Board Definition
0x00	None (Empty slot)
0x01	CPU
0x02	Memory
0x03	Color
0x04	Frame Buffer
0x05	FPA
0x06	SMD Disk Controller
0x07	Tape Controller
0x08	Ethernet Controller
0x09	MTI/ALM
0x0A	Graphics Processor (GP)
0x0B	SCP Controller
0x0C	SCSI Controller
0x0D	Integrated Personal Computer
0x0E	Graphics Board (GB)
0x0F	3/75 SCSI with Memory
0x10-0x7F	Reserved Sun Hardware
0x80-0xFE	Reserved Non-Sun Hardware
0xFF	Sentinal Block

The starting addresses for the 13 blocks are:

Block #	*Starting Address*
1	bc
2	cc
3	dc
4	ec
5	fc
6	10c
7	11c
8	12c
9	13c
10	14c
11	15c
12	16c
13	17C

The next few paragraphs explain the meaning of other values found in the the eight bytes within the blocks assigned to various boards.

CPU Board Block

If the CPU board block is located at 0xbc, and you use the `q` command to open that location, the contents might look like this:

```
>q bc[RETURN]
EEPROM 0BC: 01?
```

In the example above, the value 0x01 represents the CPU board.

If you immediately press [RETURN], the display presents the contents of the next byte, which represents the quantity of memory (in Megabytes) present on the CPU board (not the whole system), like this:

```
EEPROM 0BD: 04?
```

To change this value, simply enter the new value after the question mark, then [RETURN], to view the contents of then next byte.

The next byte, 0xbe, contains a binary code that represents options present on the board. The two options are:

bit 0 = 68881 FPP chip
bit 1 = DCP (encryption processor) chip

Option	*Value*
None	00
FPP only	01
DCP only	02
Both	03

If the option is present on the board, the value of its assigned bit is 1; otherwise the value is zero. If the board has a 68881 installed, but not a DCP, the value at address 0xbe would be 1. If both chips are on the board, the value would be 0x03, which is the binary sum of a "1" in both bit 0 and bit 1. And, of course, if neither option is present on the board, the value would be 0.

The remaining bytes in this block, at locations 0xbf through 0xc3, are unused at this time, and should contain zeroes.

Memory Expansion Board

Memory Expansion board information begins with the value 0x02. If the location assigned to the memory board is 0xc4 (the first address in the second configuration block), then the first byte should read:

```
EEPROM 0C4: 02?
```

The next byte represents the quantity of memory installed on this board only: 02=2MB, 04=4MB, or 08=8MB.

```
EEPROM 0C5: 04?
```

The remaining bytes in this 8-byte block should contain zeroes.

Color Board

The first byte of the color board block contains the value 0x03. The second byte contains the value 0x02 for a Color-2 board, or 0x03 for for a Color-3 board. The remaining six bytes should be filled with zeroes.

SMD Disk Controller Board

The first byte in the SMD Controller block contains the value 0x06. The second byte contains the value 0x01 to represent a Xylogics 450 or 451 board.

The third byte contains either 0x00 or 0x01 to represent the controller board number.

The fourth byte represents the number of disks controlled by the board.

Tape Controller Board

The first byte of the Tape Controller Board block contains 0x07.

Values in the second byte are decoded as follows:

0x01= Xylogics 472 board
0x02= Ciprico 1000 Tapemaster

The third byte contains either 0x00 or 0x01, depending on the controller board number. The fourth byte represents the number of tape drives controlled by the board.

Asynchronous Line Multiplexer

The first byte of the ALM board block contains 0x09.

The second byte contains a hex value that represents the number of terminals connected to the board.

SCSI Board

The first byte in the SCSI board block contains the value 0x0c.

The next byte identifies a SCSI2 or a SCSI3 board.

0x2 = SCSI2
0x3 = SCSI3

The third byte represents the number of tape drives that are present.

The fourth byte contains the number of disk drives present in the system.

IPC	The first bytes containes the value 0x0d. The next byte contains the number of Mbytes installed on the IPC.

ASCII	*Hex*	*ASCII*	*Hex*	*ASCII*	*Hex*	*ASCII*	*Hex*
nl (line feed)	0A	6	36	N	4E	f	66
cr (return)	0D	7	37	O	4F	g	67
sp (space)	20	8	38	P	50	h	68
!	21	9	39	Q	51	i	69
"	22	:	3A	R	52	j	6A
#	23	;	3B	S	53	k	6B
$	24	<	3C	T	54	l	6C
%	25	=	3D	U	55	m	6D
&	26	>	3E	V	56	n	6E
'	27	?	3F	W	57	o	6F
(	28	@	40	X	58	p	70
)	29	A	41	Y	59	q	71
*	2A	B	42	Z	5A	r	72
+	2B	C	43	[	5B	s	73
,	2C	D	44	\	5C	t	74
-	2D	E	45	]	5D	u	75
.	2E	F	46	^	5E	v	76
/	2F	G	47	_	5F	w	77
0	30	H	48	`	60	x	78
1	31	I	49	a	61	y	79
2	32	J	4A	b	62	z	7A
3	33	K	4B	c	63	{	7B
4	34	L	4C	d	64	|	7C
5	35	M	4D	e	65	}	7D
						~	7E
						del	7F

The following are the tables for the 13 board types:

Table D-8 **No Board**

0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00
0x00

Table D-9 **CPU Board**

0x01
Mb of Memory (hex)
Installed Options: Bit 0 = 1 (68881) Bit 1 = 1 (DCP/DES) Bit 2 = 0 Bit 3 = 0 Bit 4 = 0 Bit 5 = 0 Bit 6 = 0 Bit 7 = 0
Kb of Cache (hex)
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-10 **Memory Board**

0x02
Mb of Memory (hex)
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-11 **Color Board**

0x03
Type (2 or 3) 0x2 = Sun2 0x3 = Sun3
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-12 **Frame Buffer Board**

0x04
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-13 **FPA Board**

0x05
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-14 **SMD Disk Board**

0x06
Manufacturer 1 = Xylogics 450 2 = Xylogics 451
Controller #
# of Disks
Drive #0 Capacity 0 = No Disk 1 = 8" 130 Mb (450/451) 2 = 8" 280 Mb (451) 3 = 10.5" 380 Mb (450/451) 4 = 10.5" 575 Mb (451)
Drive #1 Capacity
Drive #2 Capacity
Drive #3 Capacity
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-15 **1/2 Inch Tape Controller**

0x07
Manufacturer 1 = Xylogics (472) 2 = Ciprico (TM1000)
Controller #
# of Tape units
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-16 **Ethernet Board**

0x08
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-17 **MTI/ALM Board**

0x09
Terminals
Manufacturer 1 = Systech
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-18 **GP Board**

0x0A
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-19 **SCP Board**

0x0B
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-20 **SCSI Board**

0x0C
Type (2 or 3) 0x2 = Sun2 0x3 = Sun3
# of Tapes
# of Disks
Tape Controller 0x1 = Sysgen 0x2 = MT02
Disk Controller 0x1 = MD21 0x2 = Adaptec
Drive #0 Capacity 0x1 = 5.25" 71 Mb 0x2 = 5.25" 141 Mb
Drive #1 Capacity
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-21 **IPC Board**

0x0D
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-22 **GB Board**

0x0E
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-23 **3/75 SCSI Board**

0x0F
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved
Reserved

Table D-24 **Sentinal Block**

0xFF
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used
Not Used

Table D-25 *Default Addresses*

Address	Function	3/50	3/75/140/160/180	3/110	3/260/280
0x14	Installed Memory	0x04	0x04	0x04	0x08
0x15	Tested Memory	0x04	0x04	0x04	0x08
0x16	Monitor Screen Size	0x00	0x00	0x00	0x13
0x17	Watchdog Action	0x00	0x00	0x00	0x00
0x18	UNIX Boot Path	0x00	0x00	0x00	0x00
0x19-0x1D	Alt Boot Path	U/D	U/D	U/D	U/D
0x1E	Reserved	N/A	N/A	N/A	N/A
0x1F	Primary Terminal	0x12	0x12	0x00	0x12
0x20	Sun Logo	0x00	0x00	0x00	0x00
0x21	Keyboard Click	0x00	0x00	0x00	0x00
0x22	Diag Boot Device	0x00	0x00	0x00	0x00
0x23-0x4F	Diag Boot Path	U/D	U/D	U/D	U/D
0x50	High Res Columns	N/A	N/A	N/A	0x50
0x51	High Res Rows	N/A	N/A	N/A	0x30
0x52-0x57	Reserved	N/A	N/A	N/A	N/A
0x58	SCC A Baud Rate	0x00	0x00	0x00	0x00
0x59-0x5A	Alt Baud Rate	U/D	U/D	U/D	U/D
0x5B	SCC A DTR/RTS	0x00	0x00	0x00	0x00
0x5C-0x5F	Reserved	N/A	N/A	N/A	N/A
0x60	SCC B Baud Rate	0x00	0x00	0x00	0x00
0x61-0x62	Alt Baud Rate	U/D	U/D	U/D	U/D
0x63	SCC B DTR/RTS	0x00	0x00	0x00	0x00
0x64-0x67	Reserved	N/A	N/A	N/A	N/A
0x68-0xB7	Custom Banner	U/D	U/D	U/D	U/D
0xB8	Test Pattern	0xAA	0xAA	0xAA	0xAA
0xB9	Test Pattern	0x55	0x55	0x55	0x55

Notes:

1. Factory default for each location is zero unless otherwise specified.

2. N/A stands for Not Applicable.

3. U/D stands for User Defined. These fields do not need to be modified unless the user wishes to customize these parameters.

4. Installed/Tested memory values are for basic product offerings. Valid entries for the product's total memory capacity (basic + options) are: 0x6 (6 MB), 0x8 (8 MB), 0xA (10 MB), 0xC (12 MB), 0xE (14 MB), 0x10 (16 MB), 0x18 (24 MB) and 0x20 (32 MB).

E

Insert Pages for 3.0 Commands Reference Manual

E

Insert Pages for 3.0 Commands Reference Manual

NAME

setup – Sun UNIX installation program

SYNOPSIS

setup

DESCRIPTION

setup is the program supplied by Sun to install major Sun Unix releases such as 2.0 or 3.0. *setup* allows a system administrator to install major Sun Unix release on new hardware, upgrade between major releases, and add additional hardware to existing machines.

setup provide both a tty interface for cursor addressable terminals and a SunWindows system interface for use on bit mapped displays. The >I "Setup Reference Manual" contains a detailed description of the use of *setup*.

Initially, *setup* asks the following questions in a menu format before entering the tty or SunWindows interface. For all menus respond to the >> prompt with the corresponding number of the menu item you choose.

The first question asked is the mode of use of *setup*.

```
Are you running setup:
      1) to install on a new system
      2) re-entrantly
      3) to upgrade an existing system
      4) in demonstration mode
>>
```

The next question is to determine the type of interface to be used. Note that the cursor addressable interface can be used within a *shelltool*(1) under SunWindows.

```
Will you be running setup from:
      1) a Sun bit mapped display device
      2) a cursor addressable terminal
>>
```

If you have selected the tty interface for cursor addressable terminals, *setup* asks for the terminal type.

```
Select your terminal type:
      1) Televideo 925
      2) Wyse Model 50
      3) Sun Workstation
      4) Other
>>
```

If you select "Other", the name of the terminal must correspond to a name in the *termcap*(5) database.

```
Enter the terminal type (your terminal type must be in /etc/termcap):
>>
```

setup begins running the interface for the terminal-type you have selected.

FILES

/etc/hosts
/etc/nd.local
/etc/ethers
/etc/rc.local
/etc/rc.boot
/etc/setup.info
/usr/lib/sendmail.cf

BUGS

setup will not run on tty devices that do not support cursor addressing and are not registered in the *termcap*(5) data base.

NAME

eeprom – Sun-3 EEPROM display and load utility

SYNOPSIS

eeprom [**–i**] [**–**] [**–f** filename] [field[=value]] ...
eeprom [**–i**] [**–c**] [**–f** filename]

DESCRIPTION

eeprom displays or changes the values of fields in the EEPROM. It processes fields in the order given. When processing a *field* accompanied by a *value*, *eeprom* makes the indicated alteration to the EEPROM; otherwise it displays the *field*'s value. When given no field specifiers, *eeprom* displays the values of all EEPROM fields. A '–' flag specifies that fields and values are to be read from stdin (one *field* or *field=value* per line).

eeprom verifies the EEPROM checksums and complains if they are incorrect; if the **–i** flag is specified, erroneous checksums are ignored. If the **–c** flag is specified, all incorrect checksums are recomputed and corrected in the EEPROM.

OPTIONS

–i ignore bad checksums.

–f *filename* use *filename* as the EEPROM device.

–c correct bad checksums.

– read field names and values from stdin.

The field names and their possible values are:

hwupdate	a valid date (including "today" and "now")
memsize	8 bit integer (megabytes of memory on machine)
memtest	8 bit integer (megabytes of memory to test)
scrsize	"1024x1024", "1152x900", "1600x1280", or "1440x1440"
watchdog_reboot	"true" or "false"
default_boot	"true" or "false"
bootdev	%c%c(%x,%x,%x)
kbdtype	8 bit integer (0 for all Sun keyboards)
keyclick	"true" or "false"
console	"b&w" or "ttya" or "ttyb" or "color"
custom_logo	"true" or "false"
banner	banner string
diagdev	%c%c(%x,%x,%x) - diagnostic boot device
diagpath	diagnostic boot path
ttya_no_rtsdtr	"true" or "false"
ttyb_no_rtsdtr	"true" or "false"
columns	number of columns on screen (8-bit integer)
rows	number of rows on screen (8-bit integer)

SEE ALSO

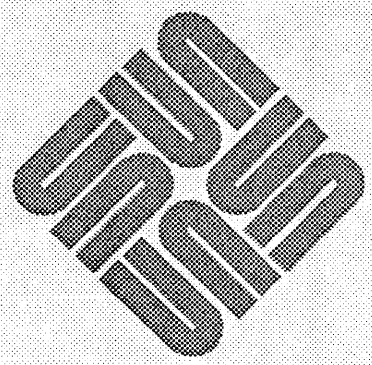
<mon/eeprom.h>

FILES

/dev/eeprom

F

Allocating a Larger Root Partition



F

Allocating a Larger Root Partition

This appendix explains how to use *diag* to allocate a larger root partition. *Setup* is capable of changing the sizes of all the other partitions and should be used instead of *diag* in all other cases.

Lets take as an example an 84-megabyte Fujitsu SCSI disk. The default label for this disk allocates a 7.9-megabyte root partition (partition A), a 16-megabyte swap partition (partition B), and the rest of the disk to partition G. The following procedure shows how to change the root partition to contain 20-megabytes.

First a brief discussion of disk "geometry". A disk is made up of *sectors* (512-bytes on all Sun disks), *heads* and *cylinders*. A group of sectors make up a head and a group of heads make up a cylinder. Therefore, the number of sectors multiplied by the number of heads equals the number of sectors per cylinder. The 84-megabyte Fujitsu disk has 11 heads per cylinder and 17 sectors per head for a total of 187 sectors per cylinder.

Hard partitions always begin on a cylinder boundary and have a size given as a number of sectors. To avoid wasting disk space, it is wise to have the number of sectors equal a multiple of the number of sectors per cylinder.

To allocate a 20-megabyte root partition you must first determine how many sectors are in 20-megabytes. A sector is 512-bytes (1/2 kilobyte) and there are roughly 1000 kilobytes per megabyte. Therefore, 20-megabytes is roughly equal to 40000 sectors.

40000 sectors = 20 * 1000 * 2

This number should then be rounded up to the next cylinder boundary (multiple of 187) yielding 40018 sectors spread over 214 cylinders.

213.9 = 40000/187
40018 = 214 * 187

Using the same alogrithm, the 16-megabyte swap partition is 32164 sectors spread over 172 cylinders.

The remaining disk space will be allocated to the G partition. The total number of sectors on the disk (140624) is the number of cylinders times the number of heads times the number of sectors.

140624 = 752 * 11 * 17

Therefore, the G partition will be 68442 sectors spread over 366 cylinders. The

following table summarizes the new partition map.

Partition	Starting Cylinder	Num of Sectors	Num of Cylinders
a	0	40018	214
b	214	32164	172
c	0	140624	752
g	386	68442	366

>From within *diag*, the **partition** command is used to change the partition map. Type the **partition** command in response to the *diag* prompt. *Diag* then displays a menu of partition tables and asks you to choose one:

```
diag> partition
Select partition table:
        0 - Micropolis 1304
        1 - Micropolis 1325
        2 - Maxtor XT-1050
        3 - Fujitsu M2243AS
        4 - Vertex V185
        5 - Other
Which one? 5
```

Choose the **Other** menu as you are creating your own partition map.

Diag then asks for a name for the partition map you are creating. The exact text of the name is not important, but use some descriptive phrase like "Large Root".

Diag then asks you partition by partition if you want to change the given partition and, if so, it requests the new values. For our example, the dialogue looks like:

```
Partition a: starting cyl=0, # blocks=0
Change this partition? y
Starting cylinder? 0
# of blocks? 214/0/0
Partition b: starting cyl=0, # blocks=0
Change this partition? y
Starting cylinder? 214
# of blocks? 172/0/0
Partition c: starting cyl=0, # blocks=0
Change this partition? y
Starting cylinder? 0
# of blocks? 752/0/0
Partition d: starting cyl=0, # blocks=0
Change this partition? n
Partition e: starting cyl=0, # blocks=0
Change this partition? n
Partition f: starting cyl=0, # blocks=0
Change this partition? n
Partition g: starting cyl=0, # blocks=0
Change this partition? y
Starting cylinder? 386
# of blocks? 366/0/0
Partition h: starting cyl=0, # blocks=0
Change this partition? n
```

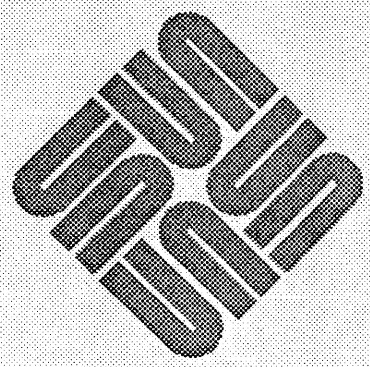
When you are done modifying the partition map, you are asked to verify the new label you have created:

```
Verify partition table 'Large Root'
    Partition a: starting cyl=0, # blocks = 40018
    [ and so on ]
    OK to use this partition table? y or n
```

When you have created a partition map you like and have verified it, you are reminded to "Use the **label** command to write out the partition table", then you see the *diag* prompt again. Use the **label** command as described in Chapter 2 (you will be given the alternate label you have just created).

G

Distribution Tape



G

Distribution Tape

The following table describes the distribution tapes' contents, and the order in which the tape files appear.

NOTE *For all tapes but the first one, the first tape file is a Copyright file, the second is a Table of Contents file and the last is another Copyright file. These tape files have been omitted in the table below in order to make the table clearer.*

NOTE *Note that the boot command follows the convention of numbering the first file on the tape as file #0. If you must load these files directly from the tape, decrement the numbers in the table by 1.*

Table G-1 *Contents of 1/4 Distribution Tapes*

1/4'' Tape File Number		*Contents*
Tape 1	1	A general purpose boot program which knows how to boot from the various devices that can be attached to the Sun Workstation. You boot this program from PROM monitor.
	2	Table of Contents.
	3	Copyright.
	4	A copy of the *diag* program. *diag* is used during installation to format and label disks.
	5	A standalone *copy* program which can copy from specified sources to specified destinations.
	6	An image of a mini version of UNIX, which contains enough information run *Setup* to install full UNIX.

Table G-1 *Contents of 1/4 Distribution Tapes— Continued*

1/4" Tape File Number	*Contents*
7	The complete root file system for the UNIX operating system [*tar*(1) format].
8	Pub files [*tar*(1) format].
9	Client Image [*tar*(1) format].
10	copyright
Tape 2 1	copyright
2	table of contents
3	Sys [*tar*(1) format].
4	Networking tools and programs [*tar*(1) format].
5	Debugging tools [*tar*(1) format].
6	`/usr files` [*tar*(1) format].
7	copyright
Tape 3 1	copyright
2	table of contents
3	SunWindows User Programs [*tar*(1) format].
4	SunWindows Programmers Programs [*tar*(1) format].
5	SunWindows and Demo Program source [*tar*(1) format].
6	Text Processing tools [*tar*(1) format].
7	Setup tools [*tar*(1) format].
8	Standalone Diagnostics [*tar*(1) format].
9	Fortran Compiler & Libraries [*tar*(1) format].
10	User Level Diagnostics [*tar*(1) format].
11	Suncore & CGI Libraries [*tar*(1) format].
12	Pascal Interpreter & Compiler [*tar*(1) format].
13	Profiled Libraries [*tar*(1) format].

Table G-1 *Contents of 1/4 Distribution Tapes— Continued*

1/4'' Tape File Number	*Contents*
14	uucp programs [*tar*(1) format].
15	copyright
Tape 4 1	copyright
2	table of contents
3	System V programs and libraries
4	Manual Pages [*tar*(1) format].
5	Demonstration Programs [*tar*(1) format].
6	Games [*tar*(1) format].
7	Versatec Printer Software [*tar*(1) format].
8	copyright

Table G-2 *Contents of Half—Inch Distribution Tapes*

1/2'' Tape File Number	*Contents*
Tape 1 1	A general purpose boot program which knows how to boot from the various devices that can be attached to the Sun Workstation. You boot this program from PROM monitor.
2	Table of Contents.
3	Copyright.
4	A copy of the *diag* program. *diag* is used during installation to format and label disks.
5	A standalone *copy* program which can copy from specified sources to specified destinations.
6	An image of a mini version of UNIX, which contains enough information run *Setup* to install full UNIX.

Table G-2 *Contents of Half—Inch Distribution Tapes— Continued*

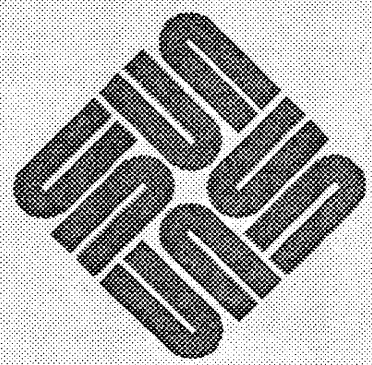
1/2" Tape File Number	*Contents*
7	The complete root file system for the UNIX operating system [*tar*(1) format].
8	Pub files [*tar*(1) format].
9	Client Image [*tar*(1) format].
10	Sys [*tar*(1) format].
11	Networking tools and programs [*tar*(1) format].
12	Debugging tools [*tar*(1) format].
13	`/usr files` [*tar*(1) format].
14	copyright
Tape 2 1	copyright
2	table of contents
3	SunWindows User Programs [*tar*(1) format].
4	SunWindows Programmers Programs [*tar*(1) format].
5	SunWindows and Demo Program source [*tar*(1) format].
6	Text Processing tools [*tar*(1) format].
7	Setup tools [*tar*(1) format].
8	Standalone Diagnostics [*tar*(1) format].
9	Fortran Compiler & Libraries [*tar*(1) format].
10	User Level Diagnostics [*tar*(1) format].
11	Suncore & CGI Libraries [*tar*(1) format].
12	Pascal Interpreter & Compiler [*tar*(1) format].
13	Profiled Libraries [*tar*(1) format].
14	uucp programs [*tar*(1) format].
15	copyright
Tape 3 1	copyright

Table G-2 *Contents of Half—Inch Distribution Tapes— Continued*

1/2'' Tape File Number	*Contents*
2	table of contents
3	System V programs and libraries [*tar*(1) format].
4	Manual Pages [*tar*(1) format].
5	Demonstration Programs [*tar*(1) format].
6	Games [*tar*(1) format].
7	Versatec Printer Software [*tar*(1) format].
8	copyright

H

Saving Disk Space with NFS



H

Saving Disk Space with NFS

H.1. Diskful Clients

Sun Workstations with their own disk can be made into clients of an NFS server, freeing up about 15 or 20 megabytes of disk space. The idea is simple. You just mount `/usr` from a file server on the network, then remove your own `/usr` files, freeing up all that disk space for your own use. This process transforms a *diskful* workstation into a *diskful client*. The details are below. But first, a few comments.

- The client and server must run the same version of the operating system (for example Sun's UNIX release 3.0).
- If the client is a Sun-3, it is not advisable to use a Sun-2 as a server, because performance will suffer.
- If you are converting a system that has already been in use, be sure to back up any files you want to keep. In particular, make sure you have made full backups of affected filesystems with the dump(8) command. If you will be moving around home directories, use `tar`(1) to write and read the tapes.

H.2. Finding a Server

The first step is to identify a file server on your network. If you have any diskless machines on your net, you already have a file server. Otherwise, you have to designate an NFS file server. To get the best performance, pick a machine that:

- runs the same version of UNIX as the client (essential),
- is a Sun-3 (or a Sun-2 with all Sun-2 clients),
- has a Xylogics SMD disk controller, and
- is not doubling as a timesharing machine.

There are two methods for setting up a client and server. The simplest method is to mount only `/usr/bin`, `/usr/etc`, and `/usr/ucb` from the NFS server. The other method is to mount all of `/usr` from the NFS server. The first method is easier and more flexible, but the second saves more disk space. The second method has the additional disadvantage that `/usr/tmp` ends up on the root partition, which is typically space-restricted.

H.3. Diskful Client, First Method

This method is generally the best. For example, if you are running a program that expects files to be in `/usr/wps`, you can easily create this directory because `/usr` is on your disk, not on the server. This flexibility is not without some compromise — it does not save as much disk space as the method discussed in the next section. For instance, the `/usr/lib`, `/usr/man`, `/usr/dict`, `/usr/hosts`, `/usr/local`, `/usr/adm`, `/usr/spool`, `/usr/preserve`, and `/usr/tmp` directory hierarchies are still on local disk. You may also want to mount these directories to free up more disk space, but the following instructions do not walk through the additional steps required. Remote mounting `/usr/lib` and `/usr/man` will generally save you the most space of all the directory hierarchies listed above.

As shown in the box below, become super-user and add the following entries to the client machine's `/etc/fstab` file. In the example, the NFS server's name is `server`, and the client's name is `client`:

```
server:/usr/bin /usr/bin nfs ro,hard 0 0
server:/usr/etc /usr/etc nfs ro,hard 0 0
server:/usr/ucb /usr/ucb nfs ro,hard 0 0
```

Next issue the following commands:

```
client% su
Password: enter root password
client# cd /usr
client# mv etc etc.orig
client# mv bin bin.orig
client# mv ucb ucb.orig
client# mkdir etc bin ucb
```

Now reboot the client machine; a good way to do this is with `/etc/reboot`. When you have verified that everything works properly, you can gain about 15 megabytes of free space by becoming super-user again and typing:

```
client# cd /usr
client# rm -rf bin.orig ucb.orig etc.orig
```

H.4. Diskful Client, Second Method

This method saves more disk space than the first, because you are able to remove almost all of `/usr` from your disk — everything except home directories, mail, and any personalized files. This results in a savings of about 20 megabytes. However, if you need to create directories in `/usr` (for example `/usr/wps`), you won't be able to do it as a regular user, since the server will almost surely have the `/usr` directory write-protected except for the super-user. If it were not write-protected, additions users made would appear on all machines mounting `/usr` from this NFS server.

Setting Up the Server

Before proceeding, it is a good idea to make a full backup of the `/usr` filesystem using `/etc/dump`, just in case you later find something missing:

```
server# dump 0f /usr
```

If your system administrator has done backups recently, this step may not be necessary.

As shown in the box below, get your system administrator (or someone who has root privilege on the server) to log in to the NFS server machine, become super-user, and create the directories `/private` and `/private/usr`. Next copy the `/usr/adm`, `/usr/preserve`, `/usr/spool`, `/usr/tmp`, `/usr/lib/news`, and `/usr/lib/uucp` directories to `/private`. The `tar` command provides a convenient mechanism for doing this. In the following examples, the NFS server is named `server`:

```
server% su
Password: enter root password
server# mkdir /private /private/usr
server# cd /usr
server# tar cf - adm preserve spool tmp lib/{news,uucp} | \
    (cd /private/usr; tar xfp -)
```

Remove all the former directories, and in their places, make symbolic links:

```
server# cd /usr
server# rm -rf adm preserve spool tmp
server# cd /usr/lib
server# rm -rf news uucp
server# ln -s /private/usr/adm /usr/adm
server# ln -s /private/usr/preserve /usr/preserve
server# ln -s /private/usr/spool /usr/spool
server# ln -s /private/usr/tmp /usr/tmp
server# ln -s /private/usr/lib/news /usr/lib/news
server# ln -s /private/usr/lib/uucp /usr/lib/uucp
```

Now move some system-specific files, and make symbolic links back to them. Symbolic links are required because these files are often different from one machine to another:

```
server# cd /private/usr/lib
server# mv /usr/lib/aliases .
server# mv /usr/lib/aliases.dir .
server# mv /usr/lib/aliases.pag .
server# mv /usr/lib/crontab .
server# mv /usr/lib/sendmail.cf .
server# cd /usr/lib
server# ln -s /private/usr/lib/aliases aliases
server# ln -s /private/usr/lib/aliases.dir aliases.dir
server# ln -s /private/usr/lib/aliases.pag aliases.pag
server# ln -s /private/usr/lib/crontab crontab
server# ln -s /private/usr/lib/sendmail.cf sendmail.cf
```

Finally, make `/usr` available for NFS mounting by adding the following line to the `/etc/exports` file (creating this file if it does not already exist):

```
/usr
```

Now reboot the file server machine; the `/etc/reboot` command is a convenient way to do this.

Setting Up the Client

Before proceeding with the client, it is a good idea to make a full backup of the `/usr` filesystem using `/etc/dump`, just in case you later find something missing:

```
client# dump 0f /dev/rst0 /usr
```

If you have done recent backups, this step may not be necessary.

To set up a client, become super-user and create the directories `/private`, `/private/usr`, and `/private/usr/lib`. Then you want to save your mail and personalized files with the following commands:

```
client# mkdir /private /private/usr /private/usr/lib
client# cd /usr
client# tar cf - adm preserve spool tmp | \
    (cd /private/usr; tar xfp -)
```

Now you need to move some files from `/usr/lib` (namely `aliases`, `aliases.dir`, `aliases.pag`, `crontab`, and `sendmail.cf`) to the corresponding location in `/private`:

```
client# cd /private/usr/lib
client# mv /usr/lib/aliases .
client# mv /usr/lib/aliases.dir .
client# mv /usr/lib/aliases.pag .
client# mv /usr/lib/crontab .
client# mv /usr/lib/sendmail.cf .
```

Now add the following line to your `/etc/fstab` file (assuming `server` is

your file server):

```
server:/usr /usr nfs ro,hard 0 0
```

You need to create a new mount point in your root directory for files you need to save from your *old* `/usr` filesystem, such as your home directory and the files it contains. In this example the new mount point is `/local`. First you need to incorporate this into your `/etc/fstab` file by changing the line

```
/dev/sd0g /usr 4.2 rw 1 2
```

to this:

```
/dev/sd0g /local 4.2 rw 1 2
```

After writing the `/etc/fstab` file to disk, you need to create the `/local` directory:

```
client# mkdir /local
```

Finally, you must update the home directory fields in `/etc/passwd` to indicate the new directory organization. Entries such as

```
you:5WdrqmZRu9ats:1508:10:Your Name:/usr/you:/bin/csh
guest:DADyzWS9nIozw:1718:40:& User:/usr/guest:/bin/csh
```

must become:

```
you:5WdrqmZRu9ats:1508:10:Your Name:/local/you:/bin/csh
guest:DADyzWS9nIozw:1718:40:& User:/local/guest:/bin/csh
```

Now reboot your machine; a convenient way is to use `/etc/reboot`. When the system comes up, files that used to be in `/usr` will be in `/local`. Become super-user, and delete all the files in `/local` that you don't need. This gives you an idea of some directory hierarchies you might remove:

```
client# cd /local
client# rm -rf bin dict etc games hosts include lib pub \
    sccs stand ucb
```

You should have about 20 megabytes more disk space than before. A hint: even after you have verified that everything works properly, retain the backup tapes for a few weeks just in case.

Because `/usr/tmp` becomes a symbolic link to `/private/usr/tmp`, `/tmp` and `/usr/tmp` are now on the same filesystem (the root partition). If you have applications that use both these directories heavily, you may find that you occasionally run out of disk space in `/private`. If you want, you can make `/private/usr/tmp` a symbolic link to someplace in your *old* `/usr` filesystem, now called `/local`. This will eliminate space competition between `/tmp` and `/usr/tmp`.

Index

T

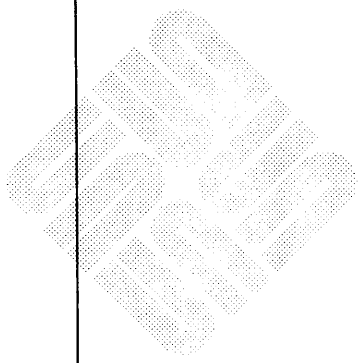
U

W

X

Revision History

Revision	Date	Comments
01 β	15 May 1986	First release of this UNIX Installation Manual.
02 β2	25 June 1986	Second Beta
03	15 August 1986	Beta Draft



Notes