TANDY®

TECHNICAL INFORMATION SERIES

SYSTEM V XENIX INFORMATION

0220 TECHNICAL SUPPORT SERVICES

XENIX™ System V/286 Information Manual

MS™-DOS Operating System:

XENIX™ System V/286 Operating System:

"vsh" Utility:

Revision History

Field(V7)
Ø4/Ø2/1984
kjb

Field(Sys3)
Ø2/17/1986
vrs

Field(SysV)
1Ø/29/1986
vrs

Revision History

Field(V7)
Ø4/Ø2/1984
kjb

Field(Sys3)
Ø2/17/1986
vrs

Field(SysV)
1Ø/29/1986
vrs

This manual is divided into several sections. The first section will deal primarily with what you need to know in order to transfer the XENIX Base System from the floppy diskettes provided to a customer's system.

Section Two will discuss unusual installation procedures, such as those systems which may be partitioned between MS-DOS and Xenix.

Section Three will discuss the hardware aspects of the system. Proper jumpering, termination, and mandatory modifications to the system will be explored.

Section Four deals with some software and control aspects of Xenix. The importance of the file **/usr/adm/messages**, media error map regeneration, setting of terminal options, and initialization of printers and terminals are discussed. The editor is briefly described, the hardware and software aspects of the multi-terminal interface are delved into, and file systems are explored.

Section Five describes the installation procedures to be used for secondary hard drives.

Section Six is devoted to "hints and kinks" about the system. Solutions to some common and not-so-common problems are addressed.

Section Seven contains advanced system information. This information is presented primarily for reference; it is not recommended that you try the material presented here unless you are very familiar with Xenix.

Section Eight addresses methods of system backup. This information is, again, primarily presented for reference, although the odds are good that at one time or another you will have cause to use it.

The appendices contain a variety of things ranging from error messages to command syntax summaries.

****

Before proceeding with the installation of the Xenix Base System, it is MANDATORY that you refer to Section Three and verify that ALL hardware modificatiions have been properly implemented.

****

****

Some Important Things You Need to Know

BEFORE STARTING

****

1) When running Xenix, the hard disks are numbered hdØx and hdlx, where x is the partition number and hdØ is the primary hard disk.

2) Any disks used in the system (with special exceptions) must not be write protected. Approximately every 3Ø seconds Xenix will access drives containing mounted file systems to update files and directories. This will occur even when the system is idle.

3) Xenix utilizes the Media Error Map which comes with the hard drive. Since this map is usually not easily accessible when the system is running, copy the contents of the map for each drive in the system before beginning to initialize Xenix.

   If the Media Error Map is missing for a given drive, the bubble may need to be checked with hard disk reliability diagnostics. Refer to Section Four for information about how to reconstruct an error map.

4) All commands except those specifically stipulated must be entered in **lower case** only.

5) If the hard drive to be initialized has been previously used for other data, operating systems, etc., ensure that all needed programs and data files have been saved off, as the initialization process will wipe all information.

6) **root** is the superuser. When logged into the system under this name, the user has unlimited access to all user, system, data, and program files. In addition, only the superuser may add or delete users on the system. He has complete control. When you log in on a customer installation as root, exercise extreme caution because you could inadvertently obliterate something... two months worth of payroll, a 6Ø,ØØØ name mailing list, the company president's password, the.....

.....

The Superuser has unlimited access to the system.
Be careful not to accidentally nebulize any
customer data when logged in as root.
The life you save could be your own!

.....

****

DO NOT ASSUME ANYTHING!

(Really. I kid you not.)

****

Table of Contents

****

SECTION ONE

Normal System Installation --
"Things just aren't easy anymore..."

****

A great many of you reading this manual will never have worked on Xenix systems before the release of System V for the Tandy 3ØØØ. In some ways, you will have some advantages over your co-workers with prior Xenix experience, because you won't have any preconceived notions to overcome. Things have gotten a lot more complicated ...!

One of the most alarming things for a service technician first being greeted by a System V installation is the fact that there are so many diskettes. System V can actually be thought of as three software systems -- it's kind of the equivalent of the Version 7 (Xenix 1.x.x) and System 3 (Xenix 3.x.x) Core and Development Systems. Admittedly, these are sold as the Base, Software Development, and Text Processing Packages, but if the customer buys the whole system, this is one large bunch of diskettes we're talking about.

The next big difference is that this package runs on a Tandy 3ØØØ, which is an Intel architecture (and a single-processor machine). This makes life rather different in terms of troubleshooting and even basic installation of a Xenix system. For those of you who have experience with MS-DOS hardware, and no experience with Xenix, and those of you who are familiar with Xenix on 68ØØØ systems, but have no inkling of what's happening on Intel hardware, hold on to your hats. Hopefully this manual will guide everyone through the cracks and allow you to figure on what is going on.

Here goes....

We'll open the show with a not-so brief synopsis of "How to Do a Normal System Installation Without Really Killing Yourself .... Well, Maybe."

.....

Xenix Software

or

"What Are All These Disks For, Anyway?"

.....

The complete Xenix System V operating system is a set of three more-or-less separate software systems. They can be broken down as follows:

- The Timesharing System (includes Help Pages)
- The Software Development System
- The Text Processing System

The Timesharing System (or the Base System) is essentially equivalent to the Core System in Xenix 1.x.x or 3.x.x. It contains the programs you need to create user accounts, manage file systems, and perform system maintenance tasks like backups. It also includes on-line documentation in the form of Help Pages which duplicate certain portions of the documentation which comes with the software (boy, we are talking **reams** of paper, too!... but I digress...). As a bonus, it also includes Xenix Deskmate, which serves as an introduction to Xenix for the truly novice user, and also provides some no-frills applications.

The Software Development System is comprised of the programs you need to create, compile, link, and debug assembly and high-level (read as "C") language programs. Included are an assembler, C language compiler with support libraries, an editor, a debugger, and various other pieces of support software.

The Text Processing System contains the Xenix utilities to create, edit, and typeset documents. I might note here that these are not the type of word-processing programs that users expecting something like Scripsit or Microsoft Word are going to be pleased with... these are not "what you see is what you get" utilities, although they are very powerful if used correctly.

Each of these systems is installed separately. The major constraint is that you **must** install the Base System before you can proceed with installing the Help Pages, Software Development, or Text Processing Systems. Additionally, the **hardware requirements** to install Xenix System V on a Tandy 3ØØØ are:

- one hard disk (minimum 15 meg)
- one high-capacity floppy drive.

The complete operating system **has to have** at least 15 meg of hard disk. You can't shoehorn it into a 1Ø meg, so spare yourself some pain and don't try it.

Incidentally, in these instructions, be sure to follow the directions exactly as they are given. You can correct any errors you make with the <BACKSPACE> or <CTL><H> keys, but this only applies before you press <ENTER>. If there is a space in the command line, it belongs there -- don't leave it out!

.....

All commands should be entered in lower case unless otherwise stated.

.....

Before you can install Xenix on a Tandy 3000, you have to run the "setup" program which comes on the Utilities disk packaged with the 3000. This program will accomplish two things:

- It will tell the 3000 what to expect in the way of memory, floppy disks, hard disks, and video.
- It will run a format procedure on the hard drive.

You must run the Setup program any time you add or remove:

- memory
- floppy drives
- hard drive (internal or external)
- video display cards

Also, since the CMOS RAM which stores all of the above information keeps track of the "permanent" system date and time, you may run Setup if you have a need to change either one of those, too. (Xenix does let you change the CMOS clock as well as the system clock, but it may be more convenient for you to change the time via Setup.)

So, onward....

(1) Insert the Utilities diskette into floppy drive A (the upper drive) with the label facing up. Don't close the door latch yet.

(2) Power the computer up. (For you non-Intel types, the power switch is in the right rear of the computer).

(3) Close the drive door.

(4) You should get some rendition of the following prompt screen:

```
BIOS ROM version Ø1.ØØ.ØØ (** this may be different! **)
Compatibility Software (C) 1985
Phoenix Software Associates Ltd.,
All Rights Reserved
Licensed to Tandy Corp.

ØØ512k Base Memory, xxxxxk Expansion ++
Time-of-day clock stopped
Invalid configuration information please run SETUP program
Strike the F1 key to continue
```

++ The amount of expansion and base memory will depend on the memory configuration of the machine.

This screen will vary, depending on whether or not Setup has been run on this machine before. If you have run Setup previously, you will probably go right into the Setup Menu, so bear with me!

(5) Press the <F1> function key. You should see the following message:

```
Phoenix Software Asc. ltd
Configuration Setup Program Ver 1.1
(C) Copyright 1985

This program is used to store system configuration
information into battery backed memory in your
computer. It is necessary to run this program when
any memory, disk drives, or monitors are added to
or removed from your system, or set the battery-
maintained time or date.
ERRORS FOUND -
INCORRECT CONFIGURATION INFO
MEMORY SIZE MISCOMPARE

Press <enter> to continue ...
```

(6) The next two screens you will see describe how to set the system date and time. Follow the instructions as they appear on the screen.

(7) The last screen is for the hardware configuration. If the system has never been set up, answer "no" (<N>) at each prompt, and select the correct response from the choices offered. You need to know the following information to complete the configuration:

- The floppy disk drive types in drives A and B
- The hard disk(s) type(s).
- The system base memory (either 512k or 640k)
- The expansion memory (if any)
- The primary video card

.....

"How do I know What Drive I Have?"

A List of Various Hard Drive Types

.....

Here is a summary of the various drive types that Setup will expect to see. If your drive doesn't exactly match one of these types, use the one which is closest -- we can fix up the differences when we actually get around to formatting the drive.

Drive Type	Description	Write PreComp	Landing Zone
1	306 cyl, 4 heads	128	305
2	615 cyl, 4 heads	300	615
3	615 cyl, 6 heads	300	615
4	940 cyl, 8 heads	512	940
5	940 cyl, 6 heads	512	940
6	615 cyl, 4 heads	no	615
7	462 cyl, 8 heads	256	511
8	733 cyl, 5 heads	no	733
9	900 cyl, 15 heads	no	901
10	820 cyl, 3 heads	no	820
11	855 cyl, 5 heads	no	855
12	855 cyl, 7 heads	no	855
13	306 cyl, 8 heads	128	319
14	733 cyl, 7 heads	no	733
15	undefined		

In any case, to finish step (7), answer all the configuration questions. When you have done this, you will get the following prompt at the bottom of the screen:

```
Are these options correct?
(Reply Y or N then <enter>)
```

(9) If you made any incorrect selections, press <N> and <ENTER> to repeat the setup procedure. If your selections are correct, press <Y>. Then, press <ENTER> and the date, time and hardware configuration will be recorded in the CMOS RAM.

(10) The computer must now be rebooted under the new hardware configuration. Reset the computer in one of the two following ways: press the <CTL><ALT><DEL> keys simultaneously (it helps to be a concert pianist with this option!) or the red RESET button on the front panel of the CPU.

(11) The copyright page appears on the screen again. Press the <F1> function key to display the main menu of the utility diskette. It should look something like this:

<1> Format diskette
<2> Copy diskette
<3> Prepare system for moving
<4> Setup
<5> Format hard disk

<9> End utilities

(12) We want to format the hard disk, so type:

<5> <ENTER>

(13) The following prompt appears:

Which hard drive do you want to format? (C/D)

We want to format the primary, so type **<C>**. After you make your selection, the following warning message is displayed on the screen:

All data on drive X will be DESTROYED!!
Do you want to continue? (Y/N)

They are serious about this, so if the customer was running MS-DOS on this drive and wants to save any of that data, you'd better bail out here. We'll assume (dangerously) that you want to proceed, so type **<Y> <ENTER>**.

(14) The formatting program then displays information on the drive type and the number of heads and cylinders of the hard disk. If you need to change something here, type **<N> <ENTER>** at the

Is this correct? (Y/N)

prompt. Otherwise, type **<Y> <ENTER>** to proceed with the format.

(15) The next prompt is:

```
Do you want to flag defective tracks?
(Y/N)
```

Enter any bad tracks here by typing **<Y> <ENTER>** and then entering any bad tracks when prompted. If there were no bad tracks, type **<N> <ENTER>**.

Note: This formatter wants you to enter bad tracks in "head, cyl" pairs. This can be a little confusing if you are used to the 68ØØØ-based Xenix's **diskutil** utility.

(16) The hard disk will now begin the format procedure. You must let this process proceed to completion. Stopping it in the middle may make this drive very difficult to get a format onto later.

(17) Once the format is completed, you will be returned to the main menu. Type **<9> <ENTER>** to exit the Utilities Menu. You will be prompted to ready the system for its next activity. At this point, remove the Utilities diskette and insert the **Xenix Installation** diskette into drive A and press **<ENTER>** when ready.

```
.........................................

                Hot Dig!!

            Finally, Some Xenix!
              (Well, sort of.)
.........................................
```

(18) Drive A will make some official-sounding bumping and grinding noises, and eventually the following screen ought to appear:

```
SYSV    XENIX Boot

Enter:  hd program
        fd program
        dos
        cf [-c conf_file] [device program]

Press Enter for default:        fd /xenix.fd
:
```

Do not press <ENTER> (yet). If you do not see this prompt, hit **<RESET>** and try again.

(19) The next thing we have to do is create a Xenix-compatible bad track table. To do this, we run a program called, strangely enough, **badtrack**. To do this, type:

fd /etc/badtrack <ENTER>

(20) The screen shows:

Loading

then:

Loaded, press Enter to start

(21) Press **<ENTER>**. The **badtrack** program scans the hard disk for bad tracks. The screen will say:

drive Ø has **xxx** cylinders and **x** heads
scanning drive Ø
cylinder **n**

where **n** is the current cylinder number. When **badtrack** has completed its scan of the disk, it displays:

Map for drive Ø is:

followed by a list of all bad tracks found. Next, **badtrack** asks you if you want to add any additional bad tracks.

Enter addtional bad tracks for drive **x**
Enter cylinder [Ø-xxx]
(press Enter to terminate):

Check the media error map which came with the hard drive and compare it to the error map displayed by **badtrack**. If there are any additional bad tracks on the error map which have not been locked out, enter them at this time. Enter the cylinder first. The machine will then prompt:

Enter track [Ø-x]

.....

This is a little confusing -- when they say "track" here they mean "head", so enter the head number.

.....

Continue this process until you have locked out all of the bad tracks. When you have finished, enter just a newline to terminate the process. **Badtrack** will then display the complete media error map.

.....

Badtrack allows you to lock out
up to **244 bad tracks.**

(Lots, eh?)

.....

(22) When **badtrack** completes execution, the screen will go back to the Xenix boot prompt.

****

Now that the format and bad track
mapping is complete,
the system installation can start.

****

Those of you who are familiar with the 68ØØØ Xenix versions may remember that they used a program called **hdinit** to initialize the hard disk. In System V, you have to call it by name, but the program we will use goes by the same title. **Hdinit** does several things for you:

- Copy the boot track to the hard disk.
- Sets the size of the swap area.
- Creates a **partition table** for the hard disk.
  -- this concept is new, and I'll talk about it here in a little bit.
- Creates a file system on the hard disk.
- Copy the contents of the Installation Disk to the hard disk.
- Performs a system shutdown after the above is completed.

This probably looks pretty familiar, except for the bit about the "partition table". What is this? Sounds kind of odd, and it is. Among other things, partitioning the disk allows you to have more than one operating system resident on a given hard disk drive. For example, if you wanted to run both MS-DOS and Xenix on the same hard drive, partitioning would make it possible for you to do so. I will cover this type of non-standard installation in the next chapter.

A more common use in System V is that it allows you to break the file system up into pieces which are easily manipulated. For example, the default installation creates a partion called "/dev/usr" which is where all the user files and directories are stored... and this can be treated just like a hard drive in that it has to be "mounted" and "umounted" .... but I'm getting way ahead of myself here! (Sorry about that!) Where were we? Oh, yeah. Initialization, right?

(23) At the "Xenix Boot" prompt, press **<ENTER>**. The screen shows:

```
fd /xenix.fd
Loading
Loaded, press Enter to start
```

(24) Press **<ENTER>**. Xenix displays some startup messages and the prompt:

```
No single-user login present
Entering System Maintenance Mode
#
```

(25) Once you see the root prompt -- # -- type:

hdinit <ENTER>

The floppy drive will make some official sounding noises, and display some information about what you are getting yourself into, followed by:

```
Press ENTER to continue or DEL to abort the installation:
```

(26) We want to continue the process, so press **<ENTER>**. When you do, the screen displays some information about choosing a swap space size and prompts you with:

```
Do you want to use the default swap space size? (y,n)
```

The default is adequate for most purposes, especially if the system has 3 meg or less of memory installed. On systems which have more than 3 meg of memory, you may find yourself in a position where the maximum amount of memory which any one user process can utilize exceeds the amount of swap space on the system. This is undesirable, but we'll discuss the reasons (and what to do about it) later. For now, type:

<y> <ENTER>

(27) Next, **hdinit** displays information which describes various ways in which you can partition your hard disk. This breaks down into three options:

-1- Xenix is going to be the only operating system on the hard disk.

-2- You want to reserve space for another operating system (like MS-DOS), in addition to Xenix. This operating system will not be the default "boot" operating system, since Xenix will be installed starting at track Ø.

-3- You want to add Xenix partitions to the current partition table on the hard drive. This option lets you install Xenix starting at someplace other than track Ø.

We will choose option 1, although I will mention the other two choices in the next chapter. The system prompts:

```
If you choose option 1 or 2, all current files on the
fixed disk will be destroyed.  You must back up all
user files you wish to save before continuing to
install XENIX on the fixed disk.

Enter option number:
```

(28) Type:

<1> <ENTER>

The system displays:

```
The root partition is now being created.
```

The system will display further information along these lines.

(29) The system will then ask you to deal with the following:

```
Enter the name of your local time zone by choosing
any one of the following time zones or create a
custom time zone by choosing Option 7 (Other):

        1.  Greenwich
        2.  Atlantic
        3.  Eastern
        4.  Central
        5.  Mountain
        6.  Pacific
        7.  Other

Enter option number:
```

Select the appropriate time zone, and type the corresponding option number followed by an "<ENTER>".

(30) **Hdinit** then proceeds to build file systems on the partitions you have created and copy a skeletal Xexix system to the hard disk. It displays information telling it what it is doing so you won't panic and think it is scraping the oxide off the hard drive. When it gets finished, it displays the following message and halts the system:

```
        Now do the following things:

1.  Wait for the shutdown message.

2.  Remove the Installation Diskette.

3.  Press ENTER to reboot XENIX from the fixed
    disk.

4.  Follow the instructions of the automated
    installation process.

** Normal System Shutdown **

Press ENTER to reboot:
 ........................................

        DO NOT Touch the System Until
            You See The Message:

        ** Normal System Shutdown **
 ........................................
```

Remove the Installation floppy disk from the floppy drive and press **<ENTER>**. This boots Xenix from the hard drive, and the Xenix boot prompt will appear.

Note: Xenix automatically boots if you do not press <ENTER> within a few minutes. If this should accidentally happen, Xenix shuts itself down. Press <ENTER> to reboot and try again. Automatic rebooting does not work at this stage of the installation process.

(31) At the boot prompt, press **<ENTER>**. The screen should show:

```
:
hd /xenix
Loading
```

Some other startup information is displayed which gives the status of optional equipment such as disk cartridge (cd), tape cartride (mt), serial adapter (sa), and multi-terminal interface (mc). Then the following prompt should appear:

```
No single-user login present
Entering System Maintenance Mode
```

(32) Finally (!), we are going to install the Xenix programs and files from the remainder of the distribution set onto the hard drive. After some other information about this process appears on the screen, Xenix prompts:

```
Press ENTER to continue:
```

Before you hit <ENTER>, make sure that you have all the diskettes for the Base System around, and that they are in the correct order.

When you are ready to go, type **<ENTER>**.

.....

The program Xenix uses to install its programs during initialization is called **firsttime**.

It calls another program named **xinstall** which deals with installing the individual software systems.

.....

(33) After **xinstall** does an initial check to make sure that the file system is there, it displays some information on inserting floppy diskettes followed by:

```
First diskette?  [y,n]
```

(34) Insert the first distribution floppy into the primary floppy drive. (Not the Installation Floppy, the next one!) Type **<y><ENTER>**.

The **xinstall** program displays the names of the files as it copies them from the distribution disk to the hard drive. After it has finished with the first diskette, **xinstall** will prompt you with either:

```
Next diskette? [y,n]
```

or

```
tar: please mount new volume
```

Remove the diskette and insert the next diskette into the floppy drive. Make sure that you have them in the correct order! If you got the first prompt, type **<y><ENTER>** when you are ready to continue; otherwise, just press **<ENTER>**.

(35) Continue inserting the diskettes which are part of the Base System Installation until you have used the last disk in this group. Then, type **<n><ENTER>** at the "Next diskette?" prompt. **Xinstall** then sets the correct permissions for the files you just installed.

(36) Xenix now displays:

```
XENIX System V/286 BASE SYSTEM INSTALLATION COMPLETE
Do you wish to install the Software Development
Package now? (y/n)?
```

We will assume for the sake of argument that this system is going to have all of the packages available for installation. In other circumstances, you may need to answer "no" to the above, but we are going to treat it as a "yes". So, type **<y><ENTER>**. Make sure you have all of the Software Development System diskettes available in order, and install them as instructed by the screen prompts in the same fashion as the Base System. When there are no more diskettes in the group, type **<n><ENTER>**.

(37) Xenix prompts:

```
XENIX System V/286 SOFTWARE DEVELOPMENT PACKAGE
INSTALLATION COMPLETE
Do you wish to install the Text Processing
Package now (y/n)?
```

Type <y><ENTER> and repeat step (36) for the Text Processing Diskettes. After all the diskettes in the group have been installed, type <n><ENTER> at the "Next Diskette?" prompt. The screen will show:

```
XENIX System V/286 TEXT PROCESSING PACKAGE
INSTALLATION COMPLETE
Do you wish to install the On-Line Help Pages
Package now (y/n)?
```

(38) Installing the On-Line Help Pages is a good idea, but it takes a while because you will be creating a cross-referenced keyword file which takes about 25 minutes. So, type <y><ENTER> and insert the diskettes in order as described above.

(39) After the keyword file has been created, and the Help pages installed, Xenix will display some additional information and proceeds to shut the system down.

(40) When Xenix displays the "XENIX Boot" prompt, press <ENTER>. The display will show:

```
** Normal System Shutdown **

Press Enter to reboot:
```

(41) Press <ENTER> to reboot the system. Xenix starts up and displays the boot screen again. Press <ENTER>. Xenix displays the startup screen and the following prompt:

```
Type Ctrl-d to proceed with normal startup,
(or give root password for system maintenance):


.........................................

            Congratulations!
          (Pant, pant, pant...)

 You have finished the system installation.
         (Unbelievable, isn't it?)
.........................................
```

At this point, you are ready to try this System V stuff out.

Type **<CTRL><d>**.

This tells the system to engage a normal startup procedure. One of the things that may seem a bit odd to those of you familiar with the earlier Xenix systems is that instead of prompting **you** for the correct system time, Xenix pulls it from the battery-backed internal clock. This is one good reason to be certain that you entered the correct time in the setup procedure at the very beginning of this arduous process!

Some information will be displayed about the status of various optional peripherals, and beneath that you should see a login prompt. Type in **root** and press **<ENTER>**.

Console Login: **root <ENTER>**

You will get a **#** prompt. At this point, you are logged in as "root" on this system, and we can check a few things out. (Carefully!)

****

Checking out the system

****

If you are not already logged in as root, do so. If you are the technician who did the installation, the system will not prompt you for a password. (This is because we have not assigned root a password yet, for those of you who are curious.) If it is an up-and-running customer system, you will need to ask the customer for his root password.

Once you have logged in, you will see a message stating "Welcome to Xenix System V/286" (or something similiar), followed by "Terminal type is ansi" and some seconds later the root prompt "**#**" will be displayed. Time to dive in.

A few simple commands which will tell you a little about what is going on in the system are given here. Be aware that a lot of the output is going to look different here than it did in the earlier Xenix versions! When in doubt at any point, you can always type **help <ENTER>**. The **help** utility, if given no arguments, is menu driven in System V, so you don't even have to know the name of the command you want help for if all else fails.

1) **l <ENTER>** (l = lower case L)
   This command produces a long version directory listing (just as it did in Version 7 and System 3).

.....

<CTRL><S> Hold video output (X-OFF)
<CTRL><Q> Restart video output (X-ON)

You may use these at any time
to pause information arriving on the screen.
.....

2) **ps -elf <ENTER>** (You **MUST** use the minus sign!)
   This is the command which gives you the current status of processes on the system. Syntax for this looks very similiar (in fact, identical) to that for the **ps** command on System 3. If you leave the minus sign out, or use the older Version 7 syntax, or use an argument that **ps** doesn't understand, it will give you a nasty little message which instructs you on correct usage.

3) **who am i <ENTER>**
   This tells you who you logged in as (in case you forget).

4) **<CTRL><D>**
   This logs you off the system.

To shut the system down, log in again as root. When the root prompt "#" appears, type:

shutdown <ENTER>

The system will ask you "how many minutes to shutdown?". Enter a **0** (that's a "zero", folks!) for an immediate shutdown. This is pretty much the same semi-graceful technique one uses to shut both Version 7 and System III systems down; once you get the **** Normal System Shutdown **** on the screen it is safe to boot the system up on a diagnostic or whatever else you want to do to it.

For those of you who are feeling adventurous, the **visual shell** (usually abbreviated "**vsh**") provides a (somewhat) more friendly interface to Xenix for the new and/or inexperienced user. It isn't a bad way to introduce yourself to the rudiments of Xenix, and it is very well documented in The Visual Shell User's Guide if you are interested.

.....

Never turn the system off without
first performing shutdown
(or a reasonable facsimile).

.....

To install additional hard drives, you will have to run the **setup** program on the Utilities disk, run a variant of the **/etc/badtrack** program found on the Xenix Installation disk, partition the secondary, and make a filesystem on any partitions on the secondary you plan to use with Xenix. In addition, you will have to make certain that the new file system(s) has a place to live. We will discuss this further in Chapter 5.

****

How to Install Deskmate
(And Other Applications!)

****

Now, the installation procedures we looked at for Xenix are great, but what happens if you want to install any additional application packages? We'll take Xenix Deskmate as an example, since that comes with the Base Package. As it turns out, it's really pretty easy, especially compared to what you've just been through!

(1) Log in as root. At the root prompt (#), type:

installap <ENTER>

(2) The system will prompt you with a menu:

```
Installation Menu
1.  to install
q.  to quit

Please Select:
```

We want to do an installation, so type:

1 <ENTER>

(3) Xenix responds:

Insert diskette in Drive Ø and press <Return>

Install the Xenix Deskmate Installation Diskette in floppy drive Ø and press **<ENTER>** when you are ready to proceed.

(4) Once you press <ENTER>, the system will access the floppy drive and print an initial "Tandy 3ØØØ DeskMate Installation" greeting screen. After this screen, you should see something like this:

Disk Configuration

#	Name	Free Space Available (blocks)
1.	/	3238
2.	/usr1	19672
3.	/usr	6326
4.	Quit	

> Enter selection (1-4):

This screen is asking you where you wish to install Deskmate. Not all application installations will ask you things like this, but the Deskmate installation is fairly representative. What you will see when you install Deskmate will strongly depend on the configuration of the system you are working on; this screen is just an example.

(5) On my example system, we want to put Deskmate on /usr, so type:

3 <ENTER>

(6) A certain amount of bumping about will be performed by the disk drive, accompanied by messages like, "Installing DeskMate programs...." When the process is complete, you should see:

DeskMate has been successfully installed

Installation complete - Remove the diskette, them press <Return>

(7) At this point, you will return to the Installation Menu outlined in step (2). Type q **<ENTER>** to quit and return to the root prompt.

****

Non-Standard Installation Procedures

(They're kind of wierd,
but what's a technician to do?)

****

Every once in a while you may find yourself in a position to do what might loosely be classed as a "nonstandard" installation. By "non-standard," I mean installations where Xenix will not be the only thing on the hard drive. In this chapter, I will try to address the two basic variants on this theme.

If you will think back to what we just covered in Chapter 1 for a moment, you will recall that the program **hdinit** was used to initialize Xenix on your hard drive. **Hdinit** gave you three options for using the primary hard drive:

-1- Xenix is going to be the only operating system on the hard disk.

-2- You want to reserve space for another operating system (like MS-DOS), in addititon to Xenix. This operating system will not be the default "boot" operating system, since Xenix will be installed starting at track Ø.

-3- You want to add Xenix partitions to the current partition table on the hard drive. This option lets you install Xenix starting at someplace other than track Ø.

Option (1) was covered in Chapter 1. Options (2) and (3) are what we will address here. In this chapter, we will assume (perhaps dangerously) that there is a Xenix compatible format already on the hard disk, and that **/etc/badtrack** has been run so that there is a bad track table present for Xenix to refer to.

Let's take a look at Option (2) first.

Reserving a Seat for MS-DOS

(1) Insert the Xenix Installation Diskette in floppy drive A and press **<RESET>** to bring up the "Xenix Boot" prompt:

```
SYSV    XENIX Boot

Enter:          hd program
                fd program
                dos
                cf [-c conf_file] [device program]

Press Enter for default:        fd /xenix.fd
:
```

(2) At the prompt, press **<ENTER>**. The screen shows:

```
fd /xenix.fd
Loading
Loaded, Press Enter to start
```

(3) Press **<ENTER>**. Xenix displays some startup information and the prompt:

```
No single-user login present
Entering System Maintenance Mode
#
```

(4) At the "#" prompt, type:

hdinit <ENTER>

The **hdinit** program loads and displays a few vaguely informative items regarding the installation process, followed by:

```
Press ENTER to continue or DEL to abort the installation:
```

(5) Press **<ENTER>** to continue. The system will then ask you:

```
Do you want to use the default swap area size? (y,n)
```

The default is adequate for most purposes, **unless** you have installed more than 3 megabytes of memory. This possibility will be covered at the end of this chapter. For now, we'll type:

<y><ENTER>

(6) The next thing up will be a screen describing the three options we talked about at the beginning of this chapter. We want to choose option 2, so we will type:

<2><ENTER>

Note: This option will wipe everything on the disk, so you will have to reinstall any of the (presumably) MS-DOS files on the DOS partition once we have created it. If you haven't got backups of the stuff on the disk, then stop now!

(7) You will now be asked how much room you want to reserve on the hard disk for the partition reserved for the other operating system. Valid answers are in the range from 1 to 45ØØ, measured in 1k blocks. (Note: If you want any room at all for user files under Xenix on the primary hard drive, you would be well advised to stay away from the upper end of that range). Type the number you select followed by <ENTER>.

(8) From here on out, the rest of the installation procedure will be just like the one we did in Chapter 1.

.....

So You Want to Put MS-DOS in
The Driver's Seat?

Here's how.

.....

So you've got a customer who's a real diehard MS-DOS user? He is really insistent that his MS-DOS system be resident (and the default operating system)? Not to worry -- if he has 2 empty or undefined partitions available, one of 9ØØØ contiguous blocks, and another of at least 6ØØØ contiguous blocks, he can install Xenix as a secondary operating system.

There are two important formulae that you will need to know to accomplish this. One allows you to calculate how many cylinders will be needed for a given partion:

Cylinders = (K-bytes + 9) / (Heads * Sectors / 2)

I will note here that the **standard number of sectors on our systems is 17.**

The other equation will allow you to calculate the number of 1K blocks available:

Blocks = (Cylinders * Heads * Sectors / 2) - 1ØØ

This will help you to determine if there is enough empty space on the hard drive to actually install Xenix.

.....

If there are not two empty partitions of 9ØØØ
and at least 6ØØØ contiguous blocks,
you will not be able to install Xenix.

.....

Assuming that the customer has enough empty space to be able to do the install, here's how you do it.

(1) Perform steps (1) through (5) in the section we just covered.

(2) When you get to the screen describing the various partition options, type:

<3><ENTER>.

(3) The **hdinit** program will display a table showing the size and type of the partitions already existing on the fixed disk. Next, it displays the prompt:

Enter the root partition number [1-n]:

Choose a partition number for the root file system. Type this number followed by **<ENTER>**.

(4) The screen displays the prompt:

Enter the user partition number [1-n]:

Choose a partition number for the Xenix user file system. Type this number followed by **<ENTER>**.

(5) The screen shows:

Enter the starting root cylinder number [Ø-n]:

Whatever else you do here, **make sure** that the number you choose does not conflict with a partition which you wish to preserve! Remember, the root partition **must have at least 9ØØØ contiguous blocks.** Otherwise, this isn't going to work. Anyway, type the starting root cylinder number and press **<ENTER>**.

(6) The screen redisplays the contents of the partition table and the prompt:

Enter the starting user cylinder number [Ø-n]:

Type the user cylinder number and press **<ENTER>**. You must have **at least 6ØØØ contiguous blocks** for the user file system.

(7) You are prompted:

```
Enter the size of the user partition in K blocks
          (at least 6000 blocks):
```

Give the size of the user partition (in 1K blocks) followed by an **<ENTER>**.

(8) The **hdinit** program will display the partition table again and ask:

```
Is the information correct? (y,n):
```

If you got everything in there the way you intended, type **<y><ENTER>**. If you goofed, or if you want to take a step back and regroup, type **<n><ENTER>** and you can try again.

(9) The rest of the installation will proceed as it did in Chapter 1.

.....

Installing Portions of the Development System after the System Was Initialized....

or

What to Do If the Customer Had Second Thoughts.

.....

Some customers may not have installed (for whatever reasons) the complete system when they first installed Xenix on their computer. They may later decide to install the portion (or portions) which they left out. Hopefully, you will never have to deal with this eventuality (not that it's difficult) but if you get asked, here's how you do it.

Before You Begin:

Make sure that you have all the appropriate diskettes (in the correct order). Log in as root, and halt the system using the **shutdown** command discussed in Chapter 1. Start up the system normally by pressing **<ENTER>** at the "Xenix Boot" prompt, and enter System Maintenance Mode by typing the root password (if there is one) at the following prompt:

```
Type Ctrl-d to proceed with normal startup,
(or give root password for System Maintenance):
```

Type the root password, followed by **<ENTER>**. Then, follow the instructions below for the package(s) you wish to install.

Installing the Software Development System:

(1) At the root prompt "#", in System Maintenance Mode, type:

xinstall soft <ENTER>

(2) The **xinstall** program displays some information as it checks the **/usr** file system and then prompts:

First Diskette? [y,n]

(3) Insert the first diskette of the Software Development System distribution set into floppy drive A, and type:

<y><ENTER>

(4) Official sounds will emerge from the floppy drive. Eventually, the screen will prompt:

Next Diskette? [y,n]

(5) Remove the disk from Drive A and insert the next diskette. Type:

<y><ENTER>

(6) Continue inserting and removing diskettes when prompted, until all diskettes have been installed.

(7) When all the diskettes have been installed, type:

<n><ENTER>

at the "Next Diskette?" prompt. The screen will then show:

Installation complete.

Installing the Text Processing System:

(1) At the root prompt "#", in System Maintenance Mode, type:

xinstall text <ENTER>

(2) Perform steps (2) through (7) of the Software Development Installation.

Installing the Help Pages:

(1) At the root prompt "#", in System Maintenance Mode, type:

xinstall help <ENTER>

(2) Perform steps (2) through (7) of the Software Development Installation.

.....

Everything You Wanted to Know
About Swap
But Were Afraid to Ask....

.....

I told you in Chapter 1 that there were circumstances in which using the default space allotted for the swapper was sometimes not enough. Well, here's where we'll talk about that.

The first thing that I need to mention is a little bit about the way things have changed from the older Xenix versions to the System V release. First off, the default swap space is huge (relatively speaking); instead of 1 meg of swap space as our default, we now have 3 meg. Seems a little odd? Seems even stranger when you notice the fact that System V is seemingly always swapping its brains out. It is not uncommon to put 1Ø seconds of time on the swap process before the system even completes the task of coming up multi-user. This would be enough to give any person used to Version 7 and System 3 (the 68ØØØ Xenix releases) heart failure, but it is perfectly normal operation in System V on the 3ØØØ. Here's why.

Xenix (any Xenix) has a limit on how large (meaning "how much memory") any one process will be. In Xenix System V, this number is dynamic. **There is a one-to-one relationship between the size of the swap area and the maximum amount of memory available to any one user process.** This is a change from the situation which was present in Version 7 and System 3, where the maximum size of any given user process was a constant.

What does this mean? Well, if you have more than 3 Meg of RAM installed, System V will give you the following message:

> Maxmem was reduced based on the size of the swap area.
> Refer to the system documentaion for information on
> the relationship between memory size and swap area.

When this message appears, it means that Xenix can only allocate up to 3 meg for any given user process (even though you have more than 3 meg of RAM present). This way, you avoid the problem of trying to stuff a 4 meg in-memory process into 3 meg of swap. This is just as bad as it sounds, and will probably lead to a "panic: out of swap" type error faster than you can say "oops." However, if you should <u>have</u> a need for a 4 meg process, and can only have a 3 meg process, things aren't so good either. So what should you do?

Easy. Remember back in Chapter 1 where it asks you if you want to use the default swap size? Instead of answering yes, tell it "no" and then give it a number in the range which it offers you. This will allow you to have the benefits of running processes which are as gargantuan as your system configuration will allow.

This does, however, have unexpected pitfalls. Let's imagine a worst case scenario where you have, say, four users all running very large databases (program + data = 4 meg), and you have a system with 4 meg of RAM and 4 meg of swap. What's going to happen? Well, you will have one process in memory (all 4 meg of it) and one process in swap (which takes up that 4 meg, too) and that leaves -- Ta Da!! -- two 4 meg processes left over!! This is, in the parlance, not good. In fact, you may get really lucky and see an "out of swap" and an "out of memory" error at the same time. Hopefully, though, this sort of thing will be rare -- I mean, you can choke a good-sized horse with the data you can get into 4 meg of RAM, so presumably this problem won't happen often. If you suspect something along these lines, get in touch with your local Xenix guru and ask his opinion. This is not a problem which you should be forced to solve!

****

SECTION THREE

Care and Feeding
of
Xenix Support Hardware

(Yes, it may bite the hand that feeds it!)

****

Computer Hardware

1) Card Order:

Card order is not quite as important in this computer as it was in the 68ØØØ-based Xenix systems. Certain cards do have somewhat standardized locations, however, and they are:

Parallel/Serial Card -- if installed, this card is installed in the 8-bit slot closest to the power supply. This takes advantage of its short length.

Hard/Floppy Controller Card -- this card goes into the 16-bit slot closest to the power supply, next to the Parallel/Serial Card if it is installed. This is due to cabling constraints... if you put it elsewhere, the cables won't reach.

Video Card -- typically goes in the 8-bit slot farthest from the power supply. If viewed from the front of the 3ØØØ, this places it in the leftmost card slot. Mainly, this is to keep it out of the way.

2) Technical Bulletins to check for:

3ØØØ:1	Provide a possible solution to intermittent operation
3ØØØ:2	Color Graphics Adapter -- eliminate garbage on screen in 8Ø column color mode
3ØØØ:5	Reduce reflected clock signal when using STB video boards
3ØØØ:7	Deluxe Text Adapter -- eliminate video noise introduction into reset line
3ØØØ:8	Gate Array Main Logic Board -- to correct oscillator circuit
3ØØØ:9	Discrete Main Logic Board -- to correct invalid interrupt vectors
3ØØØ:1Ø	2 Meg Expansion Ram Board -- to correct parity errors and provide the means to decode memory starting at a 512K boundary

3) Proper board jumpering:

Reference Notes and Jumpers.

Hard Drive Hardware

1) Technical Bulletins to check for:

3ØØØ:4	To explain motor speed alignment on Tandy 3ØØØ hard drives
3ØØØ:6	Installation procedure for internal 35 meg hard drive
HD:25	Motor speed alignment and drive logic board PCB differences for Tandon full height bubble assemblies
HD:29	Identify plated and oxide media and logic boards on Tandon full height bubble assemblies
HD:31	Proper connection of index connector on Tandon full height bubble assemblies
HD:41	7Ø meg drive information

2) Proper board jumpering:

Reference Notes and Jumpers.

3) Termination and Drive Select:

The first law is that you should never expect any hard drive to be shipped correctly jumpered or terminated. Fortunately, dealing with this is fairly simple.

.....

You should never expect any hard drive
to be shipped correctly jumpered
or terminated.

.....

Drive select jumpering is as follows:

The **first** drive (primary) is **DS1**.
The **second** drive is **DS2**.

Termination is almost that simple. If you have only one hard drive in the system, it should be terminated. If you have **two internal** or **two external** hard drives, terminate the last drive on the control cable. In other words, pull the terminator out of the primary (drive C or drive Ø) and make sure that the secondary (drive D or drive 1) is terminated.

If you have **one internal and one external** hard drive, terminate both drives.

Miscellaneous Other Hardware Considerations

1) 1Ø and 2Ø Meg Disk Cartridge System:

Technical Bulletins to check for:

I/O:1ØØ	Controller ROM replacement to enable the use of a secondary cartridge drive
I/O:1Ø4	To remedy possibly unstable or overly low output voltages which may cause erratic malfunctions of the disk cartridge drives
I/O:1Ø7	Avoid conflict with the Tape Cartridge System in the Xenix environment

Additionally, the disk cartridge system is extremely sensitive to cabling. Make certain that the cable between the interface board and the disk cartridge is securely seated, and screwed down where applicable.

NOTE: Both the 1Ø and 2Ø meg disk cartridge systems use the 1Ø meg disk cartridge interface (25-3Ø22) for Xenix operation as of the current release (Ø1.ØØ.ØØ). For correct dip switch settings for this board, refer to Notes and Jumpers.

2) Streaming Tape Drive:

There is a required PAL change on the tape cartridge interface board to enable it to work in a Xenix environment. Check and make sure this modification is in place!

I/O:1Ø8	PAL change to enable Tape Cartridge System to work in the Xenix environment

****

SECTION FOUR

Media Error Map Regeneration,
Multi-terminal PCB's and Terminals,
The Editor, and
Other Pearls of Useful Wisdom.

****

Let's say that you're working on a system with a long history of intermittent problems, and that the customer (although a nice guy otherwise) never thought that those little error messages that came up meant anything. Furthermore, he doesn't remember what they said, when they happened, or even how often they occurred. Before you start bashing your head against the nearest vertical surface in frustration, take heart. Xenix takes care of this (up to a point) with an error log of its own. The name of this useful file is:

/usr/adm/messages

and it can be a real boon to the technician under a lot of circumstances.

This file contains a record of most console error messages, although you will find that they consist primarily of disk-related errors and initial boot messages. There is a reason for this; the file is only updated every ten minutes and if the error is serious enough to crash the system then the odds are good that said error will not be recorded in the history table. However, most disk-related errors will not (at least, not immediately) crash the system, so they will usually wind up in /usr/adm/messages. You might be saying at this point, "What good is it going to do if it doesn't record all the errors?" Actually, this system works out pretty well. It turns out that a great many of the common Xenix errors/problems are disk-related, so they will usually be logged. Too, it seems to be a strange but true fact that most people are much more inclined to write error messages down when they do crash the system, so you may get a record of some of the non-logged errors that way.

In any case, you can save yourself a lot of strain if you take a moment or two to read this file. Frequently a recurring error will point you in the direction of the problem subsystem. To read this file, type:

cat /usr/adm/messages <ENTER>

and the file will be displayed on the CRT. To hold the output so you can read it, you may use the **<CTRL><S>** and **<CTRL><Q>** key sequences. If the file is lengthy or if you prefer working from hardcopy, you may print the file out on a system printer by typing either

cat /usr/adm/messages | lpr <ENTER>

or

lpr /usr/adm/messages <ENTER>

This will send the file to the default system printer. Be aware that it may be a few moments before the printer activates. This is normal.

.....

If you take the trouble to read
/usr/adm/messages
before you go tearing into a system,
you will often save yourself a lot
of detective work.

.....

To see an example of what reading this file can do for you, let's suppose that you had a situation where the customer complained of intermittent secondary hard drive problems. The /usr/adm/messages file contained, among other things, the follwing:

```
Jun  9 16:41
...
  4-Feb-86   /1:ABCDEF-OK /2:Not Installed

Reserved memory = 2k
Kernel memory = 2Ø6k
Buffers = 4Øk
User memory = 264k

Jun 1Ø 12:Ø3
HARD DRIVE 1:  hard err 4Ø59 on rd of cyl 214 head 4 sect 14
HARD DRIVE 1:  hard err 4Ø59 on rd of cyl 214 head 4 sect 14

Jun 12 17:47
HARD DRIVE 1:  hard err 4Ø59 on rd of cyl 214 head 4 sect 14
HARD DRIVE 1:  hard err 4Ø59 on rd of cyl 214 head 4 sect 14
```

Given the above, it would be a fairly safe assumption that cylinder 214, head 4 should be added to the media error map for this particular drive 1, and that you should go back through the format and installation process described in Chapter 5 to lock out this additional flaw... **BEFORE** doing this, of course, assuring yourself that the customer has at least one and preferably two backups of the current information on hard drive 1.

Unfortunately, System V doesn't have the nice little utility possessed by System 3 which allows you to easily see what the currently locked out tracks on a given drive are. If you have a full Development System installed on the system, you can see what has been locked out on a given drive... but it's not particularly easy! I will show you how to do this later, but for now an easier way to check what tracks are locked out is to use the Xenix Tools Disk.

"Very nice," you say, "but so what?" Well, what this does is make your life considerably easier if you should ever have to regenerate a hard drive media error map, which brings us rather nicely to the next subject of discussion.

Media Error Map Regeneration

At some point in time, you may find it necessary to regenerate the Media Error Map for a particular hard drive. Possibly the map has been lost by the customer, or the map does not match the bubble. The following procedure will allow you to generate a "quasi" error map, although you should bear in mind that **this process will not be 1ØØ% accurate.** If at all possible, find the real map!

This procedure is valid for all hard drives currently available from Radio Shack which might be used with a Tandy 3ØØØ.

1) If the drive is currently formatted under System V, use the Xenix Tools Disk to determine the tracks which are currently locked out on the drive.

2) Select the appropriate Diagnostic Program (refer to your hard disk diagnostic literature) that will allow you to sequentially read and verify all tracks.

3) Save all data on the hard drive if the customer desires and has not already done so. **If in doubt... CHECK!**

4) **DO NOT format the hard drive.**

5) Load and run the diagnostic selected in step 2. You do not want to do any writes, only sequential reads. **For this procedure to be effective, the diagnostic must run for a substantial period of time.** The longer you can let it run, the better; if possible allow it to run overnight.

6) Select the history table option under the diagnostic and all errors will be displayed. From this information (and the information provided by the Xenix Tools Disk and by /usr/adm/messages) you should be able to gain a fairly good idea where the flaws are on the media. Remember, Xenix is only interested in the cylinder and head number.

TERMINALS:

Setting Terminal Options:

A newly installed Xenix system does not automatically power up with the terminals enabled. The baud rates must be set to match that of the data terminals and the serial channels must be enabled at least once.

For example:

> You have just installed the Xenix Core System. From this point forward whenever you boot Xenix, the serial channels will not be enabled (i.e. your Xenix system will not talk to any terminals).
>
> However, if you configure the serial channels once, enable them, and leave them turned on, from then on whenever you boot the system the serial ports WILL BE ENABLED. Xenix remembers how you last left the system.

Unfortunately, setting the baud rates with Xenix is not the easiest thing in the world, and it has gotten a bit more complicated with the advent of System V. Mind you, it's not terribly difficult... just not easy the first couple of times. Consider the following chart:

```
1Cconsole
16tty00
05tty01
||  |
||  |_________ Port Name
||
||___________ Speed (I'll explain more about this in a moment)
|
|____________ Status (set by enable/disable)
                    1 = enabled
                    0 = disabled
```

As far as choosing the speed at which the port is prepared to talk to the terminal... well, this is where it gets complicated, ladies and gentlemen, for we are about to see yet another feature of this system which has become user configurable.

Back in the good ol' days of Version 7 and System 3, these baud rates were "wired in" to the software. All we had to deal with then was choosing the right one. Now, we get to cope with a file named **/etc/gettydefs** which allows you to configure your port parameters to your needs.

Let's take a look at a sample of what you might see in /etc/gettydefs:

```
C# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY #Console login:  #C

7# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY TAB3 #login:  #7

6# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY #login:  #6

5# B4800 PARENB CS7 OPOST ONLCR # B4800 SANE IXANY #login:  #5

4# B2400 PARENB CS7 OPOST ONLCR # B2400 SANE IXANY #login:  #4

3# B1200 CS8 OPOST ONLCR # B1200 SANE IXANY #login:  #2

2# B300 CS7 OPOST ONLCR # B300 SANE IXANY #login:  #3

1# B300 CS8 OPOST ONLCR CR1# B1200 SANE IXANY #login:  #1
```

Let's examine the boldfaced line at the top of the file. We will highlight one field at a time, and talk about what it does. Field separators, incidentally, are the symbol #.

(1) C# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY #Console login: #C

The C is a line label which identifies this line to the program which uses this file. This program is called /etc/getty, and this program is what actually sets up the "conditions" or parameters which will be used to communicate with any terminals on a given port. Getty reads the /etc/ttys file, a portion of which we examined earlier:

```
1Cconsole
16tty00
05tty01
...
```

The C (the speed flag) is what /etc/gettydefs is defining

and checks to see what the speed flag for a given channel is. This flag is the line label which getty will look for in /etc/gettydefs.

(2) C# **B96ØØ PARENB CS7 OPOST ONLCR** # B96ØØ SANE IXANY #Console login: #C

These are the initial flags which tell **getty** what to do if no terminal type is specified. In this case, they mean:

B96ØØ	96ØØ baud
PARENB	parity enable
CS7	7 bit word length
OPOST	post-processes output
ONLCR	maps newline to carriage-return/newline on output

Ordinarily, the only thing which is absolutely required is the baud rate; if left to its own devices, **getty** can set the rest of the parameters up in a reasonable fashion. These initial flags remain in effect until **getty** executes **login**, which doesn't happen until you see the "Password: " prompt or, if you don't have a password, until just before you see the "message of the day".

(3) C# B96ØØ PARENB CS7 OPOST ONLCR # **B96ØØ SANE IXANY** #Console login: #C

These are the **final flags.** Essentially, they set the same things as the "initial flags," and are set just prior to **login** execution. In this instance, the flags mean:

B96ØØ	96ØØ baud
SANE	A composite flag, which sets most of the other flags to some reasonable value
IXANY	Enables any character to restart output once it has been stopped.

(4) C# B96ØØ PARENB CS7 OPOST ONLCR # B96ØØ SANE IXANY **#Console login:** #C

This is the **login prompt.** Unlike the other fields we have discussed where white space is ignored, this field allows spaces, tabs, and newlines and will print them as such. This is what will greet users when they wish to login.

(5) C# B96ØØ PARENB CS7 OPOST ONLCR # B96ØØ SANE IXANY #Console login: #C

The last field identifies the **next label** for **getty** to try if the current one is not successful. This enables groups of entries to be set up so that, for example, a dialup line might try 3ØØ, 12ØØ, and 24ØØ baud before cycling back to 3ØØ baud to start the cycle again.

Ok, so where does all of that get us? The information:

```
1Cconsole
16ttyØØ
Ø5ttyØ1
```

is contained in a file called /etc/ttys. If you were to cat (short for concatenate) this file to the screen, you could see at a glance just how your communication channels were configured.

Those of you who are familiar with the Xenix versions for the 68ØØØ-based machines will notice that the console setting looks different than it did. This is again due to the /etc/gettydefs file we discussed above; if you didn't like describing the console setting with a "C" you could call it something else... but we won't do that, we will work with the more-or-less standard version of /etc/gettydefs that I showed you. In any case, "1Cconsole" means that the console is enabled (1Cconsole), it is communicating at 96ØØ baud (1Cconsole), and that the device name is console (1Cconsole).

"16ttyØØ" decodes to enabled (16ttyØØ), 96ØØ baud (16ttyØØ), and a device name of ttyØØ (16ttyØØ).

"Ø5ttyØ1" decodes to disabled (Ø5ttyØ1), when enabled it will communicated at 48ØØ baud (Ø5ttyØ1), and a device name of ttyØ1 (Ø5ttyØ1).

To enable a channel, enter the following command:

enable ttyØn <ENTER>

where n is the channel number. For example, to enable ttyØØ (usually, the serial port on the parallel/serial adapter) type:

enable ttyØØ <ENTER>.

.....

Normally, the serial port on the parallel/serial adapter is **ttyØØ.** The computer's keyboard is **console.**

.....

Conversely, to disable a channel enter:

disable ttyØn <ENTER>

where n is the channel number. **disable ttyØØ <ENTER>** would disable the serial port on the parallel/serial adapter.

.....

Unless You're Specifically Looking For Trouble DO NOT DISABLE THE CONSOLE (or change its baud rate).

.....

****

OK Guys, haul out the stale Twinkies.
(RC Cola is good if you have it....)
You Are Now About To Learn

The Editor.

(lots of caffine, smokes if you do,
extra-strength pain reliever optional.)

****

In order to change the baud rates of the serial channels, you must modify the file **/etc/ttys** to correspond to what you want. In order to make these changes, the file must be edited. This means that you must know how to use the Xenix editor. This editor is called **ed.**

ed is a fairly versatile, if somewhat dim-witted program designed to edit files. He will do exactly (and I do mean "exactly") what you tell him to with a minimum (really!) of cockpit error checking. This means that you should be careful. Understand what you're doing **before** you do it. **Never Assume Anything!** Read this section carefully before doing anything with **ed** or you may find yourself in (to say the least) an embarrassing situation. One way to minimize your chances of catastrophe is to make a "safety" copy of the file you intend to edit before you start! e.g.

cp /etc/ttys /etc/ttys.old <ENTER>

Also, to (very nearly) guarantee that you have permission to rewrite the file after changes have been made, you should be logged in as root, and be proceeding with the same EXTREME CAUTION you would apply to handling an armed grenade. A special note of warning to those of you who might be familiar with the MS-DOS editor, "EDLIN" -- don't take it for granted that the commands which **ed** uses will match those you are familiar with! That is a particularly vicious way to crash and burn.

.....

ed is a versatile (albeit somewhat stupid) program
designed to edit files. He will do exactly what you
tell him with a minimum of cockpit error checking.

.....

To invoke ed and edit the /etc/ttys file, type:

ed /etc/ttys <ENTER>

After a moment of thinking about it, the system will respond with a number roughly corresponding to the number of bytes in the file. Unlike the System 3 ed, you will not be prompted.... you will simply see the cursor, and ed will wait for you to give more input. To see the file in its entirety, type:

1,$p <ENTER>

This command instructs the editor to print the file beginning with line one to its end. The $ signifies the last line number, and the p tells ed to print these lines on the screen. As an example, you should now see something like the following on your screen:

```
# ed /etc/ttys
5Ø
1,$p
1Cconsole
Ø5ttyØØ
Ø6ttyØ1
Ø2ttyØ2
Ø3ttyØ3
Ø6ttyØ4
Ø3ttyØ5
Ø3ttyØ6
Ø5ttyØ7
Ø5ttyØ8
Ø5ttyØ9
```

You will have noticed that the System 5 /etc/ttys file looks a little different than the ones in Version 7 and System 3. System 5 for the 3ØØØ comes configured for eleven (yes, boys and girls, you read it right!) terminals instead of the three or six you would see in the older Xenix releases for the 68ØØØ.

Let's assume (dangerously) that our sample customer has a DT-1, a DT-1ØØ, and a Tandy 1ØØØ which he plans to use as a terminal. The DT-1 is to be run from a remote location over phone lines. It will have a modem, but because of situations beyond the customer's control, it may at times be a 12ØØ baud modem and at other times it may be a 3ØØ baud device. The DT-1ØØ and the Tandy 1ØØØ will be in the office and will run at a baud rate of 96ØØ.

.....

Even though ttyØ1 is shown in /etc/ttys,
it is not usable unless the kernel is reconfigured.

.....

We will configure tty00 to search between 300 and 1200 baud, and tty02 and tty03 for 9600 baud. Even though tty01 is shown in /etc/ttys, it is not usable unless the kernel is reconfigured. The first multiterminal board will use ports tty02, tty03, tty04, and tty05. For this example, we will configure the first two ports on the first multiterminal board (those are **tty02** and **tty03**) and the port on the parallel/serial interface board (that's **tty00**). **It should be noted here that ttys can be configured to search a range of baud rates and select the one that matches the incoming data.** Reference the discussion of baud rates earlier in this section.

What we need to do, then, is to change the /etc/ttys file as shown below:

FROM	TO
1Cconsole	1Cconsole
05tty00	03tty00
06tty01	06tty01
02tty02	06tty02
03tty03	06tty03
06tty04	06tty04
03tty05	03tty05
03tty06	03tty06
05tty07	05tty07
05tty08	05tty08
05tty09	05tty09

NOTE: It is not wise to edit the /etc/ttys file if the file shows a "1" in the first positions for the ttys implying that one or more terminals are enabled. It is wiser to quit, disable the ttys, then edit the file. Don't expect to be able to change the first character (enable or disable flag e.g. 0 or 1) and have this work. Use the **enable** or **disable** command at the system prompt instead. This forces Xenix to read the file.

In **ed**, every line has a number. **Ed** is a line oriented text editor and as such we must give him every line number we wish to change. The tricky part arises because the line numbers are not displayed so you must count them yourself. In addition to this, we must also tell ed what to do to that line.

Type the following command:

2p <ENTER>

You should see displayed on the screen the second line of the file, which in our example is:

05tty00

It's always a good idea to print the line you are about to change before you change it... just to make sure that you are going to change the correct line.

With the next command we will change the second line. Type in:

2s/Ø5/Ø3/p <ENTER>

You should now see on your screen:

Ø3ttyØØ

The above command instructed ed to get line 2 (2s/Ø5/Ø3/p), search it for the first occurrence of Ø5 (2s/Ø5/Ø3/p), change the Ø5 to a Ø3 (2s/Ø5/Ø3/p), and finally to print the line as changed (2s/Ø5/Ø3/p).

The / character is used as a delimiter between arguments of the command. If you omit one, ed will probably do something a bit strange and most likely not at all what you intended.

Now, we need to change the entries for ttyØ2 and ttyØ3, too, so first we find where they are by typing:

1,$p <ENTER>

You will see:

```
1Cconsole
Ø3ttyØØ _____________ Note that the change for ttyØØ shows up!
Ø6ttyØ1
Ø2ttyØ2 _____________
Ø3ttyØ3 _____________|_____ We need to change these lines.
Ø6ttyØ4
Ø3ttyØ5
Ø3ttyØ6
Ø5ttyØ7
Ø5ttyØ8
Ø5ttyØ9
```

To make sure we know where ttyØ2 and ttyØ3 are, type:

4p <ENTER>

You should see:

Ø2ttyØ2

Now, type:

4s/Ø2/Ø6/p <ENTER>

and you should see:

```
Ø6ttyØ2
```

Now, since ttyØ2 was on line 4, ttyØ3 will be on line 5.... but let's check it by typing:

5p <ENTER>

The following should appear:

```
Ø3ttyØ3
```

Now we can change this by typing

5s/Ø3/Ø6/p <ENTER>

and confirmation of the change will show up on your screen as the following:

```
Ø6ttyØ3
```

To check that all of our changes have been correctly performed, let's print the file out one more time. This is a good way to prevent yourself from making what might turn out to be an embarrassing mistake or two. Type:

1,$p <ENTER>

and hopefully, you will see this:

```
1Cconsole
Ø3ttyØØ
Ø6ttyØ1
Ø6ttyØ2 ______ Here are our changes.
Ø6ttyØ3
Ø6ttyØ4
Ø3ttyØ5
Ø3ttyØ6
Ø5ttyØ7
Ø5ttyØ8
Ø5ttyØ9
```

If the changes are correct, then the next step is to write the file out with the w command:

w <ENTER>

Again, the system will respond with a number to signify that the file has now been written.

To leave the editor, type **q**:

q <ENTER>

The root prompt **#** will be displayed.

The above information about the editor should be sufficient to enable you to alter the **/etc/ttys** file if you need to. However, for more detailed information about the editor in general you should refer to the Xenix System V/286 Operating System User's Guide, Appendix A. In addition, the full System 5 release has a help file on **ed** which also contains useful information. To use it (provided the help pages have been installed; see Section 1 for more information on the installation procedure), type:

help ed <ENTER>

and the on-line manual page will appear on the CRT.

Multi-Terminal Boards:

HARDWARE:

The multi-terminal interface board (25-4Ø31) will allow the user to add up to four additional asynchronous serial ports to his computer with each PCB. Up to two multi-terminal boards are supported by System 5 as it is shipped; the 3ØØØ hardware is capable of supporting up to three. If two multiterminal boards are used, the user will have a maximum of 1Ø ports, consisting of the console, the serial channel of the parallel/serial adapter, and eight ports on the two multi-terminal interface cards.

There are, naturally, several important points to be aware of when installing or servicing a Xenix system with multi-terminal interfaces. These are outlined below.

1) Switch Settings:

There are two dip switches on the multi-terminal interface board that must be set prior to use. SW1 determines the base I/O address for the board, which in turn specifies the ports it will handle. SW2 determines the host interrupt request line.

Board 1

SW1, position 4 in the on condition (closed). All other positions are in the off (open) condition.
SW2, position 5 in the on condition (closed). All other positions are in the off (open) condition.
This board will provide ports tty Ø2, tty Ø3, tty Ø4, and tty Ø5, at base I/O address 1ØØh and using IRQ1Ø.

Board 2

SW1, position 3 in the on condition (closed). All other positions are in the off (open) condition.
SW2, position 6 in the on condition (closed). All other positions are in the off (open) condition.
This board will provide ports tty Ø6, tty Ø7, tty Ø8, and tty Ø9, at base I/O address 1Ø4h and using IRQ11.

Board 3 (*)

SW1, position 2 in the on condition (closed). All other positions are in the off (open) condition.
SW2, position 7 in the on condition (closed). All other positions are in the off (open) condition.
This board will provide 4 ports at base I/O address 1Ø8h and using IRQ12.

2) Jumpers:

One of the first things you'll notice about this board is the fact that there is one whale of a lot of jumpers. This looks complicated, but it really boils down to one question: "Do you want these ports to look like terminals or look like modems?" From the point of view of a Xenix hard-wire terminal, it means, "Do you or don't you wish to use a null modem?"

For most people, the answer will be a resounding "no!" In order for this board to talk to a terminal then, you will need to jumper it as though it were a modem. Conversely, if you wanted to talk to a modem, you would need to jumper the board as though it were a terminal. There are names for these two styles of configuration. Things which emulate modems are said to be set for Data Communications Equipment (DCE) mode operation; if a port is set up in this manner on a host computer, then it can be hard-wired directly to a terminal (which is configured for DTE -- Data Terminal Emulation -- in normal operation) without using a null modem. This is the mode we will normally want to use with this board, and that is the way they are shipped.

For DCE (Data Communications Equipment) mode

Baud Rate Clocks:	A1-A2, B1-B2, L1-L2, M1-M2
Channel 1, J3 (tty Ø2 or tty Ø6):	F1-F2, F3-F4, F5-F6, F7-F8, E1-E3, E2-E4, D2-D3, D5-D6
Channel 2, J4 (tty Ø3 or tty Ø7):	K1-K2, K3-K4, K5-K6, K7-K8, H1-H3, H2-H4, G2-G3, G5-G6
Channel 3, J5 (tty Ø4 or tty Ø8):	V1-V2, V3-V4, V5-V6, V7-V8, T1-T3, T2-T4, P2-P3, P5-P6
Channel 4, J6 (tty Ø5 or tty Ø9):	Z1-Z2, Z3-Z4, Z5-Z6, Z7-Z8, X1-X3, X2-X4, W2-W3, W5-W6

* Note: The Tandy release of System V, as sold, will support only two multiterminal boards. Information for the third is provided for reference purposes only.

If you should have someone who does want to use null modems, they will need their board(s) configured for DTE, and here's how that's done:

For DTE (Data Terminal Emulation) mode:

Baud Rate Clocks:	A1-A2, B1-B2, L1-L2, M1-M2
Channel 1, J3 (tty02 or tty06):	F1-F5, F3-F7, F4-F8, E3-E4, E5-E6, D1-D2, D4-D5
Channel 2, J4 (tty03 or tty07):	K1-K5, K3-K7, K4-K8, H3-H4, H5-H6, G1-G2, G4-G5
Channel 3, J5 (tty04 or tty08):	V1-V5, V3-V7, V4-V8, T3-T4, T5-T6, P1-P2, P4-P5
Channel 4, J6 (tty05 or tty09):	Z1-Z5, Z3-Z7, Z4-Z8, X3-X4, X5-X6, W1-W2, W4-W5

SOFTWARE:

In contrast to the hardware, the software considerations for this system are from our standpoint fairly simple. I should note at this point that I'm only going to deal with what might be called a "standard" system.... "standard" being the way it is supplied to the customer. One of the things which makes System 5 for the 3000 a potentially difficult troubleshooting problem is the possibility that the customer may have (correctly or incorrectly) reconfigured the system.

However... the system is sold with the ability to support up to **two multi-user boards** as well as the serial/parallel board and the console. Therefore, the modifications which have to be performed are less intensive than those in the 68000 Xenix systems. What we will need to do is set the baud rate for each port to be added to the system by editing the **/etc/ttys** file (which we discussed earlier) and set up the default terminal type for the new ports by editing the **/etc/ttytype** file. We will take a quick look at each of these files to see what I'm talking about. If you should need to update these files, an editor will have to be used. Remember **ed**? If not, flip back a few pages and re-read the discussion of how to edit the **/etc/ttys** file. Once you feel confident enough to edit these files, come back to this point in the chapter and continue.

1) Enter the command:

cat /etc/ttys <ENTER>

You should see something like this:

```
1Cconsole
ØXttyØØ  ________________  Serial channel on parallel/serial adapter
ØXttyØ1  ________________  Not used in standard release
ØXttyØ2  ____________
ØXttyØ3  ____________|
ØXttyØ4  ____________|
ØXttyØ5  ____________|____  First multiterminal board
ØXttyØ6  ____________
ØXttyØ7  ____________|
ØXttyØ8  ____________|
ØXttyØ9  ____________|____  Second multiterminal board
||  |
||  |_____________________  Port name
||________________________  Set the baud rate here
|_________________________  Ø or 1 (enabled = 1, disabled = Ø)
```

Adjust the baud rates according to the needs of the system.

2) The second file to edit is the **/etc/ttytype** file. This file contains information relating to the terminal type of the device connected to a particular TTY port.

Enter the following command:

cat /etc/ttytype <ENTER>

You will probably see something much like this:

```
ansi console
dumb ttyØØ  ________________  serial port on parallel/serial adapter
dumb ttyØ1  ________________  Not used in standard release
dumb ttyØ2  ____________
dumb ttyØ3  ____________|
dumb ttyØ4  ____________|
dumb ttyØ5  ____________|_____  first multi-terminal board
dumb ttyØ6  ____________
dumb ttyØ7  ____________|
dumb ttyØ8  ____________|
dumb ttyØ9  ____________|_____  second multi-terminal board
 |    |
 |    |_____________________  port name
 |__________________________  This field will depend on the terminal
                              type attached to the port.
```

****

"Funny, that's not what the visual
shell's supposed to look like...."
or
How To Tell Xenix What Type of
Terminal You Are Using.

****

So, your customer just went tearing down to his friendly neighborhood Computer Center and, while he was laying out the bucks for his new multi-user Xenix software, decided he'd pick up a terminal too.... so he produces some more cold hard cash (or warm soft plastic) and purchases a brand spanking new DT1ØØ. Faster than a speeding bullet, he zooms back to the office to rip open the box with the insane zest of a toy poodle destroying a pair of slippers. Packing material flies everywhere, including -- Guess What??! -- the owner's manual. Well, really... you're only supposed to read the manual when the thing breaks, right?

Spending no time in further reflection, our stalwart customer frantically installs his Xenix software. He then boldly connects his new prize to the system, strides back to the console and configures the baud rate of the channel with the editor, and then types the fateful "enable ttyØØ". With mental trumpets blasting a fanfare, he leaps to the new aquisition as though he were Luke Skywalker out to impress the Princess and without a wasted movement logs in. He decides to try it out on the visual shell.... lots of neat inverse video and things.... he types "vsh", a pregnant pause ensues, and then, "Funny, that's not what the visual shell looked like in the pictures...."

Not too surprising, except perhaps to the customer, and although the screen does look rather funny, he's not laughing about it.

What Went Wrong:

Remember reading just a page or two ago about a file named **/etc/ttytype**? Unless the customer remembered to change this file, the port in question is probably still configured to talk to a **dumb** terminal. Actually, this would in all likelihood not make the screen look all that odd... but it wouldn't do any inverse video or the rest of the neat things one expects from a DT1ØØ. In any case, with the port configured as **dumb**, Xenix is going to configure its output to be compatible with (just like it sounds) the dumbest terminals imaginable. A more problematic situation would be one where the customer had, say, a DT1 attached to that port, and had configured it as an **adds25**. Speaking "adds25" to a DT1ØØ will make for a funny-looking screen indeed.

Actually this is something of an oversimplification. In point of fact, Xenix just doesn't care what terminal type you have, but some programs, typically those which are screen oriented (like visual editor, shells, spreadsheets, and the like) will check this table (/etc/ttytype) and in conjunction with another file, **/etc/termcap**, configure their output to look proper on the terminal in question. The Xenix utility **more** is another example of a program that will check these two files.

By working together with **/etc/ttytype** and **/etc/termcap** these programs will configure THEIR output to match that of the terminal. Codes like reverse video and cursor positioning can, and generally do, change from terminal type to terminal type.

Now, what does this mean to this rather befuddled customer? The visual shell thinks he's using a dumb terminal when he's really using a DT1ØØ. The codes are different so his screen isn't going to look the way it should.

How Does He Fix It?

He determines the tty number that he has the DT1ØØ hooked to and then edits the **/etc/ttytype** file to say **dt1ØØ** instead of **dumb**. Remember, he needs to disable the port in question before you edit /etc/ttytype, and enable it afterwards!

We decided that in the example the customer is using tty**ØØ**. Let's look at the file **/etc/ttytype.**

```
ansi console
dumb ttyØØ
dumb ttyØ1
dumb ttyØ2
dumb ttyØ3
etc. ....
```

The file needs to be edited to look like:

```
ansi console
dt1ØØ ttyØØ
dumb ttyØ1
dumb ttyØ2
dumb ttyØ3
etc. ....
```

Now, whenever anyone logs in on tty**ØØ**, Xenix and the application packages will know to configure their output, when necessary, to correctly "speak dt1ØØ".

There's Another Small Problem... (although it's unlikely to affect us)

The file **/etc/termcap** is a table that contains all the control codes, escape sequences, and various other attributes like number of columns that a particular terminal will understand. If there is no entry in the **/etc/termcap** file for the dtlØØ, the terminal output will still look funny, and nobody will be laughing.

In short, the file **/etc/ttytype** really "points" to an entry in the **/etc/termcap** file for Xenix, or more properly put, the application/utility program to look up before sending any data out the port.

The reason I said that this problem is unlikely to affect us is that System 5 is supplied with **/etc/termcap** entries for almost every imaginable terminal. Unless the customer has some truly unusual equipment to work with, it is unlikely that he will have any problems of this type.

A detailed explanation of the termcap file is more than this manual is intended for. However, the **termcap** entry in the "help" files is fairly clear, and if you are really curious I would refer you to it.

An Easy Way to Check for the Problem (if you think you have it)

Assuming that you have set everything up in a manner which you think should be correct, and the output still looks funny, try the following command:

TERM=[your terminal type here]; export TERM

Now try the visual shell, or **more** and see if the output looks correct. If it does, the **/etc/ttytype** file has not been modified or has been modified incorrectly.

And Finally, The Exception....

The **root** user. For every user on the system there is a **.profile** file that is loaded in. Without getting too descriptive, special system variables may be set here and these variables will take precedence over variables set in different system tables. The **root .profile** may set the ttytype variable to ansi regardless of the tty in use. In this case the customer may change the **.profile** file or use the above command (TERM=.....) to set the terminal type. In this case, the terminal type will be correct only until the user logs out.

****

Installing Printers,
Accessing Printers,
and
"That isn't what I meant ... can I change it now?"

****

One of the many (!) things which sets this version of Xenix apart from all the others which have been released by Tandy is that it **does not** come with the printer ready to use. Actually, I'd probably better expand on this a wee bit. The **devices** (the special files which the system uses to talk to the printer -- or printers) are there, but the programs which **spool** the individual print jobs have not been activated. What this means is that to print anything out on the system as it is sold, you would have go directly to the raw device... which makes it difficult from the standpoint of a multi-user system.

So, in order to be able to use a command like:

cat /usr/adm/messages | lpr <ENTER>

and have it work, you have to install the printer in software as well as in hardware.

First, I'll give you a little more background on the way Xenix deals with printing. To begin with, the Xenix lineprinter spooling system is a whole collection of commands that help the user to install, monitor and control the lineprinters (yes, that means you can have more that one, Virginia!) serving the system. When you issue a request to print a file using the **lp** or **lpr** command (they're two names for the same thing), the lineprinter system will respond with a **request ID**. This consists of the name of the printer the file will be printed on and a unique number identifying the print job. This **request ID** lets you do things like determine the status of the print request or cancel it altogether if you wish. There are also options to **lpr** which let you control the printer output. However, I won't go into those in any great detail here. If you would like to know more, either turn to the command summary in the back of this manual, or refer to the online documentation on the Xenix system itself.

.....

The Xenix lineprinter spooling system
is a collection of commands
that help you to install, monitor,
and control the lineprinters on your system.

.....

Just to keep things straight, let me define a few terms before I go much further. When I talk about a device in this discussion, I'll mean "the target for the output of the **lpr** command". It can be a hard-wired printer, a terminal that is sometimes used as a printer, or a regular file. A device can be represented by a full Xenix pathname, just like anything else on the system. A printer, on the other hand, is the name which the system administrator assigns to represent a device. This name can be up to 14 characters long. It follows that at different times, a printer may be associated with different devices. Additionally, you can group printers into classes; a class is an ordered list of printers. The destination for a print request can be a printer or a class.

Confusing, eh? Well, let's make a little table:

device	-- target for output of the **lpr** command. May be a printer, terminal, or a file.
printer	-- name assigned to represent a device.
class	-- an ordered list of printers.
destination	-- where the print request is routed; may be a printer or a class.

Ok? Now you are armed with the terminology, and can venture forth into the brave new world of printer initialization.

.....

Printer Initialization for the Faint of Heart

.....

Inasmuch as there are quite a few steps involved in printer initialization, the nice system programmers provided a program which automates this process to a great degree. This program is called **lpinit**.

Among other things, **lpinit** does the following:

- -- stops the line printer daemon **lpsched**
- -- runs a utility **lpadmin** which configures the line printer system
- -- **enables** the new printer
- -- tells the system to start accepting print requests for that printer
- -- restarts the printer daemon

You can begin to see why they wrote **lpinit**! This is a fairly complex process, and with **lpinit**, all you need to know is a few basic facts about the printer and where you putting it in the system. **Lpinit** will take care of the rest.

Before we try a test run, we need to have these few basic facts I mentioned above ready, so let's examine what we'll need:

(1) **What port is the printer to be attached to?**

Lpinit gives you the choice of one of the following:

(a) Serial/Parallel adapter, Parallel Port
(b) Alternate Serial/Parallel adapter, Parallel Port
(c) Monochrome and Printer adapter

The default is the serial/parallel adapter, parallel port, which is what we will use for the example.

(2) **Does the printer need special handling for codes like line feed, tab, and form feed?**

The options are yes and no; the default is no. We will use the yes option, since many Tandy printers will want this handling. As a note, if the printer has dip switches which will allow it to treat line feeds as line feeds, and carriage return as carriage return, and can additionally handle tabs and form-feeds, you will want to answer "no". Some of our printers are capable of this mode, but many are not.

(3) **What do we wish to call this printer?**

The printer name (what we called the printer in our terminology above) may be up to 14 alphanumeric characters or underscores. The default value is "linepr"; this is what we will use.

For informational purposes, here are some examples of acceptable and unacceptable printer names:

Acceptable	Unacceptable
bozo_nose	bozo*noze -- asterisk is not allowed
1231pr	antidisestablishment -- too long!!

(4) **What is the pathname of the printer interface program?**

The default printer interface program is named /usr/spool/lp/model/dumb, and will be used in our example. Printer interface programs can be shell scripts, C programs (or any other executable program), or the model program itself can be copied and modified for use on the system.

Ok, now we're ready to do the installation.

(1) Log in as root. At the root prompt (#), type:

```
/etc/lpinit <ENTER>
```

(2) **Lpinit** displays the following message on your screen:

```
The printer is attached to a

1. Serial/Parallel Adapter, Parallel Port.

2. Alternate Serial/Parallel Adapter, Parallel Port.

3. Monochrome and Printer Adapter.

Enter one of the options above
(default = 1)        :
```

(3) We are going to use the default value, so press **<ENTER>**. Next, **lpinit** displays the following message:

```
Do you need special handling for Tandy Carriage Returns and Line Feeds? [y|n]
[Refer to the lpinit command documentation for further information]
(default = n)              :
```

We will want the special handling, so type:

```
y <ENTER>
```

(4) Lpinit next says:

```
Enter a name for the printer
(default = linepr)              :
```

Again, we will use the default, so press **<ENTER>**.

(4) The screen will now say:

```
Enter an interface program
(default = /usr/spool/lp/model/dumb)    :
```

We want the default, so press **<ENTER>** again.

(5) After you have provided the answers to these four questions, **lpinit** displays the following message:

```
scheduler stopped
```

This means that the lineprinter daemon, **lpsched**, has been momentarily stopped so that **lpinit** can add linepr to the system.

(6) After stopping the daemon, **lpinit** asks if this printer will be the system default printer:

```
Is this the default printer ? (y/n)
(default = y)          :
```

Since we are assuming that this is a newly installed system (and is therefore not ready to run any printers at all), we will want this to be the default printer. So, press **<ENTER>**.

(7) The primary hard drive will commence upon a course of bumping, grinding, and beeping, and eventually the new printer will be configured to start taking print requests. When the root prompt appears, the system will be ready to start taking requests on the new printer.

.....

Testing the New Printer

"Once more, dear friends, into the breach...."

.....

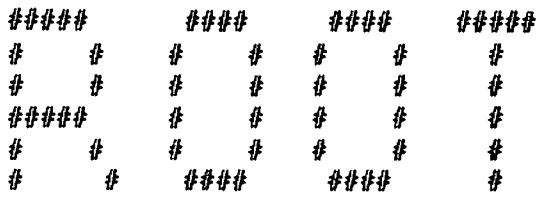
Now comes the fun part -- seeing if you have a working printer on the system. Presuming that all has gone well during the installation, all you should have to do is type:

lpr /etc/ttys <ENTER>

The system will respond with:

```
request id is linepr-1 (1 file)
```

Eventually, you should see a banner coming up on the printer:



```
User:  The Super User

Request id:  linepr-1          Printer:  _linepr

Wed Jul  2 14:59:Ø6  CDT  1986
```

and on the next sheet, your printout:

```
1Cconsole
16ttyØØ
Ø5ttyØ1
Ø6ttyØ2
Ø6ttyØ3
Ø6ttyØ4
Ø6ttyØ5
Ø6ttyØ6
Ø6ttyØ7
Ø6ttyØ8
Ø6ttyØ9
```

Your printout may not be exactly like this, but it should be close. If it is, then the system is ready for printer request.

This notion of a **request id** is new with the arrival of System 5. It is useful for a variety of reasons:

(1) Using the **lpstat** command, the user can determine the status of his print job (or jobs).
(2) Using the **cancel** command, a user can cancel his own print jobs without intervention by the system manager.

The latter is frequently very useful. If, for example, the user starts printing a data file of some five or six megs -- and then decides that wasn't such a hot idea -- he can cancel it with very little muss and fuss. To do this, he types:

cancel request-id <ENTER>

where request-id is the number given by the system as a reponse to the print request. If we wanted to cancel our printout of the /etc/ttys file in the example above, we would have said:

cancel linepr-1 <ENTER>

and the system would respond:

request "linepr-1" cancelled

and the printout would have stopped abruptly.

You should be aware that **cancel** does not generate a top-of-form at the end of the aborted printout, so that the printer will most probably need to have the paper re-adjusted before the next printout comes along.

.....

If you know the request-id of a print job,
you can use **cancel** to get rid of it.

.....

I have a lot more to tell you about printers in this system (believe me!) but since most of it is related to solving problems, I'll address it in Chapter 6.

****

Formatting Floppys and Cartridges,
Creating Small Additional File Systems,
and
Other Small Bits of Interesting Trivia.

****

Before you can use a floppy or a cartridge disk in Xenix, it must be formatted, just like anything else. (Anything but the streaming tape drive, that is!) A big difference between the 3ØØØ and the 6ØØØ versions of Xenix is that you perform this formatting in the 3ØØØ environment when the system is up and running Xenix. The reason for this is that the 3ØØØ is a one-processor machine; the 8Ø286 handles both I/O and computational services, unlike the 6ØØØ, which has a Z8ØA to do its I/O for it. This is one of the reasons why you had to run a Z8Ø-based program called "diskutil" on the 6ØØØ to format your floppy disks, and why you format in the Xenix environment on the 3ØØØ.

Furthermore, if you wish to **mount** the floppy or cartridge disk as an additional filesystem, you have to take an additional step and create a file system on it with a utility named **/etc/makefs**. This sounds kind of complicated, but the procedure is really much easier than it sounds. Follow these steps:

Formatting:

To format floppy and cartridge disks under System 5, you use the **format** command. The syntax is:

format [-qy] /dev/drive-type <ENTER>

where

-q is an option specifying no verify (fast, but not too good from a reliability standpoint. This option should rarely be used.)

-y is an option specifying to format with no questions asked, even if data exists on the disk.

drive-type may be one of the following:

Floppy Drives:

Drive Ø/Drive A:

	Device Name	Format
++	/dev/rfdØ48	9 sectors, double sided
++	/dev/rfdØ48ss8	8 sectors, single sided
++	/dev/rfdØ48ds8	8 sectors, double sided
++	/dev/rfdØ48ss9	9 sectors, single sided
++	/dev/rfdØ48ds9	9 sectors, double sided
	/dev/rfdØ96ds15	15 sectors, double sided
	/dev/rfdØ	15 sectors, double sided
	/dev/rfdØ96	15 sectors, double sided

++ At this point in time, these devices are not supported with the high capacity drive (as of the Ø1.ØØ.ØØ release). It is not known if they will be in the future.

Drive 1/Drive B:

	Device Name	Format
++	/dev/rfd148	9 sectors, double sided
++	/dev/rfd148ss8	8 sectors, single sided
++	/dev/rfd148ds8	8 sectors, double sided
++	/dev/rfd148ss9	9 sectors, single sided
++	/dev/rfd148ds9	9 sectors, double sided
	/dev/rfd196ds15	15 sectors, double sided
	/dev/rfd1	15 sectors, double sided
	/dev/rfd196	15 sectors, double sided

++ At this point in time, these devices are not supported with the high capacity drive (as of the Ø1.ØØ.ØØ release). It is not known if they will be in the future.

Cartridge Drives:

	Device Name	Format
++	/dev/rcdØ	Raw device
++	/dev/rcd1	Raw device

++ These devices are used for both the 1Ø and 2Ø meg disk cartridge systems. The internal 5.25 inch 2Ø meg disk cartridge is not supported in the Ø1.ØØ.ØØ release.

It should be noted that a **raw device** (i.e. one which does character I/O) must be used for the **format** command to work properly.

Creating the File System:

Once the floppy diskettes or disk cartridges have been formatted, you can create file systems on them. Follow the steps below:

NOTE: If all you ever plan to do with your cartridges or floppy disk is perform system backups, you can skip this information. If you plan to use either a floppy or cartridge as a **mountable** file system, you have to perform these steps.

Creating File Systems on Cartridges (both 1Ø and 2Ø meg):

1) At the root prompt, at the console, type:

/usr/bin/makefs <ENTER>

2) The screen will display the following menu:

```
Make a File System on...
    1  a floppy disk
    2  a hard disk
    3  a disk cartridge

    q  to quit


> Enter Number:
```

3) We want to build a file system on a disk cartridge, so type:

3 <ENTER>

4) The system will respond:

   On which cartridge Drive (Ø-1)?

   We'll assume that this is a one drive disk cartridge system, so type:

   Ø <ENTER>

5) The screen shows:

   Do you wish to format this cartridge?

   We are going to assume that you have formatted this cartridge, so type

   n <ENTER>

   If, however, you needed to format the cartridge, you could answer "yes" to this question and the system would format the cartridge before proceeding further.

6) Next, the screen will say:

   Insert formatted cartridge in drive Ø and press <RETURN>

   Insert the disk cartridge in the drive and press **<ENTER>**. **Makefs** will then make a file system on the cartridge.

7) When the file system is created, the screen will display:

   Press <ENTER> to continue

   The menu reappears on your screen. If you are finished creating file systems, press **q <ENTER>** and the root prompt will appear.

Creating File Systems on Floppy:

1) At the root prompt, type:

   /usr/bin/makefs <ENTER>

2) At the menu, select option 1. The system says:

   On which floppy Drive (Ø-1)?

3) Type **Ø <ENTER>**.

4) A menu will appear:

```
        Select floppy format ...

    1       High-density, double-sided
    2       40 track, 9 sector, single-sided
    3       40 track, 9 sector, double-sided


    Enter number of format:
```

5) Choose your poison, boys! For argument's sake, I'm going to arbitrarily assume we wish to make a file system on a high-density, double-sided disk. Going by this, type:

```
1 <ENTER>
```

6) Next, **makefs** asks:

```
Do you wish to format this disk?
```

We will assume you have previously formatted the disk, so type:

```
n <ENTER>
```

7) The screen should respond:

```
Insert formatted disk in Drive 0 and press <RETURN>
```

Insert the disk in Drive 0 (or Drive A, depending on your bias) and press **<ENTER>**. When the system is finished, it will say:

```
Press <ENTER> to continue
```

The main menu reappears. If you are finished making file systems, type q **<ENTER>** to exit and return to the root prompt.

```
.........................................

  You may create low-capacity file systems in
          the high-capacity drives,
  but you must format the low-capacity disks
          in a low-capacity drive,
  because the high-capacity drive will not format
     anything but the high-capacity diskettes.

                    (Whew!)
.........................................
```

Mounting File Systems:

In order to access files located within a **mount**able cartridge or floppy disk, (i.e. drives which have file systems), the drive in question must first be **mount**ed on the root file system. Although Xenix provides an empty directory for this purpose (specifically, **/mnt**) it is usually easier to remember where things are if you create a special empty directory within the root directory on which to mount your new file system. To do this, do the following:

cd / <ENTER>
mkdir cdØ <ENTER> (for the first cartridge drive)

The first command (**cd /**) moves you to the root directory. The second tells Xenix to create a directory entry called **cdØ**. If you also wanted to mount a second cartridge drive and the primary floppy disk, then you would want to use **mkdir** to make a directory for each of these file systems (calling them perhaps **cd1** and **fdØ**). In truth, you can call these directories pretty much anything you want, but cdØ, cd1, fdØ, and fd1 are fairly standard.

We're only halfway finished with this process, though... now that you know where you want to put your new file systems, you have to actually put them there. This has to be done for every secondary file system. Use the following commands as appropriate:

/etc/mount /dev/cdØ /cdØ <ENTER> (first disk cartridge)

/etc/mount /dev/cdØ /cd1 <ENTER> (second disk cartridge)

/etc/mount /dev/fdØxx /fdØ <ENTER> (first floppy disk ++)

/etc/mount /dev/fd1xx /fd1 <ENTER> (second floppy disk ++)

++ There are a variety of floppy devices; the xx signifies that you should choose the device according to what type of floppy disk you are using in it.

Unmounting:

If you need to remove or turn off a mounted secondary file system, you must first unmount it. The **shutdown** command automatically unmounts all drives so when utilizing that command it is not necessary to do any unmounts. The correct syntax to unmount a drive otherwise is:

/etc/umount /dev/cdØ <ENTER>

I really do mean you to type "/etc/umount"; that's the way the command is spelled.

If additional drives are on the system, substitute their device names as appropriate where you see "/dev/cdØ" in the above.

****

SECTION FIVE

Secondary Hard Drive Installation
...like primary installation, only more so.

****

To install a secondary hard drive on a System 5 system, you must do most of the things that you did when installing the system on the primary. The thing which makes it a bit more difficult is the fact that whereas most of the primary system installation procedure has been automated, the secondary installation is all up to the user. It isn't that difficult... it's just complicated, and leaving steps out will often lead to unpredictable (and surprising!) results.

You can divide the procedure up into six parts:

(1) Connect the hard drive to the system.
(2) Run **setup** to make the system aware of the new drive.
(3) Format the secondary.
(4) Create a bad track map for the secondary.
(5) Partition the secondary.
(6) Create file systems on the Xenix partitions created on the secondary.

A lot of stuff, but taken as a bunch of little steps, the entire process is really pretty simple. We'll dive right in at the top, and work our way through.

Connect The Hard Drive to the System:

This I'll leave to you, with the single exception that I will note that in this system you may have up to two hard drives -- no more. Installation (by which I mean, "physical connections") is going to depend on whether you have an internal or an external secondary drive.

Run Setup:

This is going to be very similar to what we did in Chapter One. This step tells the system, "Hey you! There's another hard drive here to talk to!" Otherwise, you get some nasty messages about "incorrect system configuration" upon powering the system up, and things will not proceed very well. Kind of a drag... therefore we run Setup again to prevent these problems.

To do this, haul out the **Utilities Diskette.**

(1) Insert the Utilities Diskette into Drive A. Don't close the door latch yet.

(2) Power the computer up, and close the drive door.

(3) After a certain amount of bumps, tweedles, and moans, you should get some rendition of the following:

```
BIOS ROM version Ø1.ØØ.ØØ (or Ø1.Ø1.ØØ)
Compatibility Software (C) 1985
Phoenix Software Associates Ltd.
All Rights Reserved
Licensed to Tandy Corp.

xxxxxk Base Memory, xxxxxk Expansion  ++
Invalid Configuration information please run SETUP program
Strike the F1 key to continue
```

++ The amount of base and expansion will depend on the memory configuration of the machine.

(4) Press the <F1> function key. You should see the following message:

```
Phoenix Software Asc. Ltd
Configuration Setup Program Ver 1.1
(C) Copyright 1985

This program is used to store system configuration
information into battery backed memory in your
computer.  It is necessary to run this program when
any memory, disk drives, or monitors are added to
or removed from your sytem, or set the battery-
maintained time or date.
ERRORS FOUND -
INCORRECT CONFIGURATION INFO

Press <enter> to continue ...
```

(5) The next two screens you will see describe how to set the system date and time. Follow the instructions as they appear on the screen. (If the system time needs updating, you might as well do it here!)

(6) The last screen is for the hardware configuration. If the system information is correct except for the secondary hard drive, answer "yes" (<Y>) at each prompt until you get to the one which inquires about the secondary.

(7) Eventually, you will get to the question asking about the secondary hard drive -- Drive D. On page 1.6 there is a table which summarizes the types of hard drives which **setup** is equipped to understand. Choose the drive type which most closely matches that which you have. If it isn't an exact match, don't

worry about it too much... we can fix up the differences when we get around to formatting the drive.

(8) When you have answered all the configuration questions, you will see the following prompt at the bottom of the screen:

```
Are these options correct?
(Reply Y or N then <enter>)
```

(9) If you made any incorrect selections, press <N> and <ENTER> to repeat the setup procedure. If your selections are correct, press <Y>, followed by <ENTER>. At this point, the new configuration information (including the information about the secondary hard drive) will be recorded in the CMOS RAM.

(10) The computer must now be rebooted under the new hardware configuration. Reset the computer in one of the two following ways: press the <CTL><ALT><DEL> keys simultaneously, or the red RESET button on the front panel of the CPU.

Format the secondary:

(11) The copyright page appears on the screen again. Press the <F1> function key to display the main menu of the utility diskette. It should look something like this:

```
<1>  Format diskette
<2>  Copy diskette
<3>  Prepare system for moving
<4>  Setup
<5>  Format hard disk

<9>  End utilities
```

(12) We want to format the hard disk, so type:

<5><ENTER>

(13) The following prompt appears:

```
Which hard drive do you want to format? (C/D)
```

This time we want to format the secondary, so type <D>. After you make your selection, the following warning message is displayed on the screen:

```
All data on drive X will be DESTROYED!!
Do you want to continue? (Y/N)
```

If there was any data on this drive you (or the customer!) wanted, here's your last opportunity to bail out of the format procedure and save it off. We'll assume that you know this to be a pristine, brand-new, out-of-the-box drive, however, so type **<Y><ENTER>**.

(14) The formatting program then displays information on the drive type and the number of heads and cylinders of the hard disk. If you need to change something here, type **<N><ENTER>** at the prompt:

```
Is this correct? (Y/N)
```

Otherwise, type **<Y><ENTER>** to proceed with the format.

(15) The next prompt is:

```
Do you want to flag defective tracks?
(Y/N)
```

Enter any bad tracks here by typing **<Y><ENTER>** and then entering any bad tracks when prompted. If there were no bad tracks, type **<N><ENTER>**.

Note: This formatter wants you to enter tracks in "head, cyl" pairs. This can be a little confusing, especially since the Xenix System V **badtrack** utility is going to ask for them in the reverse order....

(16) The hard disk should now begin the format procedure. You should let this process proceed to completion. Stopping it in the middle may make this drive very difficult to get a format onto later (a bit of brain-damage on the part of this formatter!).

(17) Once the format is completed, you should be returned to the main menu. Type **<9><ENTER>** to exit the Utilities Menu. You will be prompted to ready the system for its next activity. At this point, remove the Utilities Diskette and insert the **Xenix Installation** diskette into Drive A and press **<ENTER>** when ready.

.....

One of the differences you will notice
with this version of Xenix is that
you must have an Installation Floppy
to add a secondary hard drive to the system.

.....

Create a bad track map:

(18) Drive A will, after some suitable moans and groans, load up the bootstrap program, after which the following should appear on the screen:

```
SYSV    XENIX Boot

Enter:  hd program
        fd program
        dos
        cf [-c conf_file] [device program]

Press Enter for default:        fd /xenix.fd
:
```

(19) Type:

fd /etc/badtrack2 <ENTER>

This looks a little different, eh? Not to worry, **badtrack2** is the Xenix utility to create a Xenix-compatible bad track map for the secondary only.

(2Ø) The screen shows:

```
Loading
```

then:

```
Loaded, press Enter to start
```

(21) Press **<ENTER>**. The **badtrack2** programs scans the secondary hard drive for bad tracks. The screen will say:

```
drive 1 has xxx cylinders and x heads
scanning drive 1
cylinder n
```

where **n** is the current cylinder number. When **badtrack2** has completed its scan of the disk, it displays:

```
Map for drive 1 is:
```

followed by a list of all bad tracks found. Next, **badtrack2** asks you if you want to add any additional bad tracks.

```
Enter additional bad tracks for drive x
Enter cylinder [Ø-xxx]
```

(press Enter to terminate):

Check the media error map which came with the hard drive and compare it to the error map displayed by **badtrack2**. If there are any additional bad tracks on the error map which have not been locked out, enter them at this time. Enter the cylinder first. The machine will then prompt:

Enter track [Ø-x]

This is quite confusing, I realize -- when they say "track" here, they mean "head", so enter the head number.

Continue this process until you have locked out all of the bad tracks. When you have finished, enter just a newline to terminate the process. **Badtrack2** will then display the complete media error map.

(22) When **badtrack2** completes execution, the screen will go back to the Xenix boot prompt.

.....

Grab your calculators, gang!
Next, we partition the drive....

.....

Partitioning the secondary:

Ok, now in order for Xenix to use all this neat new space available to it, we have to do a little more massaging first. The only problem is that we have to do it all ourselves, since for the secondary drive this process is not automated in the least! Fortunately, this isn't all that difficult, it's just tedious. A calculator will help here, by the way, but pencil and paper will work fine, too.

First off, we have to know the number of **cylinders**, **heads**, and **sectors per track** the secondary hard drive has. Let us say for the sake of argument that the secondary we are working with is a 35 meg drive. This means that we have 512 cylinders, 8 heads, and 17 sectors per track. With this information, we can calculate the number of **blocks** on the drive using the following formula:

Blocks = (Cylinders * Heads * Sectors/2) - 1ØØ

In our example, this works out to:

Blocks = (512 * 8 * 17/2) - 1ØØ = **34716 blocks**

The next thing we have to decide is how to allocate these blocks on the secondary. You can create one Xenix user partition, or several, or you can even set aside space for a DOS partition.

We'll look here at two examples -- one being that the entire disk will be devoted to a Xenix partition, and the other being that part of the disk will be devoted to Xenix, and the remainder will be devoted to MS-DOS.

A note here: **you may have up to three partitions on the hard drive, not including the bad track table (which must always be present for Xenix). Also, you cannot create non-Xenix partitions while running Xenix.**

Well, having said all this, let's try it. Type:

hd /xenix <ENTER>

to boot off the hard drive. Type **<CTL><D>** to begin normal operation, and log in as root.

(1) When you see the root prompt (**#**), type:

/etc/fdisk /dev/rhd1Ø <ENTER>

The **fdisk** utility lets you divide the secondary disk (and, for that matter, the primary, too!) into logical partitions. The entire secondary disk is referred to as **/dev/rhd1Ø**. Once **fdisk** starts up, you should see an "*" as a prompt.

(2) The first thing we will want to do is take a look at the current partition information, so at the "*" prompt, type the following:

p <ENTER>

Fdisk will display the partition map of the drive. At this point, it should look something like this:

Partition	Type	Active	Start Cyl	End Cyl	Sectors
1	BTT(255)	N	Ø	Ø	1
2					
3					
4					

This table shows the Bad Track Table (BTT), created by **/etc/badtrack2**, on Partition 1. We have Partitions 2-4 available to use.

(3) First, let's create just one partition (i.e. we plan to devote the entire disk to Xenix use). To do this, we use the "create"

option. So, at the "*" prompt, type:

c <ENTER>

Fdisk prompts you with:

Enter partition number [1-4]:

Because the bad track table occupies partition 1, we can't use that. Therefore, we will choose partition 2. Type:

2 <ENTER>

The next prompt is:

Enter starting cylinder number [Ø-512]:

You can specify cylinder Ø as the starting cylinder of another partition, as **fdisk** will not overwrite the bad track table. Type:

Ø <ENTER>

The next prompt is:

Enter length in 1K blocks:

Remember earlier when we calculated the number of blocks we had available on the drive? Here's where we use it. Since we were going to devote the entire drive to Xenix, we will use that entire number of blocks, so type:

34716 <ENTER>

In our other example, we wanted to reserve part of the drive for MS-DOS. All this means is that we would reduce the above number by the amount you wished to reserve for MS-DOS. If, for example, you wanted a 1ØØØØ block MS-DOS partition, the number you would have used above would have been 24716. Incidentally, that is all there is to our second example!

Having done this, the program will now prompt you for the type of partition:

Enter type (2=system, 3=user)

We always want to create user partitions on a secondary, so type:

3 <ENTER>

(3) We are now ready to save out the new partition table to the hard drive. To do this, type:

w <ENTER>

To quit the **fdisk** program, type:

q <ENTER>

You will probably get a snooty little message along here commenting that "no partitions have been activated." Don't sweat it -- in this case, it's unimportant. The next step (the final step!) of making file systems will take care of that for us.

.....

Almost there....
Just gotta make the file system!

.....

Creating File Systems on the Secondary:

This is the final process in this somewhat arduous task in installing a secondary. It consists of a couple of stages: creating the file systems and giving them a place to be mounted on the root file system.

Before you can create file systems in System V, you have to create a link between the special device file that actually represents the partition and a symbolic name that you want to call the partition. (You can bypass this, but it's a lot easier if you don't....) In our example, we created a Xenix partition at partition 2 on the secondary. This is represented by the device name **/dev/hd12**. The 1 represents the device name, and the 2 represents the partition number. Additionally, we will need to create a directory for the new file system to be mounted on. To do this, follow these steps:

(1) Link the block device by typing the following command:

ln /dev/hd12 /dev/usr2 <ENTER>

/dev/usr2 will be the symbolic name we'll use for this partition.

(2) Link the character device by typing:

ln /dev/rhd12 /dev/ruser2 <ENTER>

(3) Create a directory upon which we can mount this partition:

mkdir /usr2 <ENTER>

/usr2 will be the directory on the root filesystem where we will eventually **mount** the secondary. First, though, we have to build a file system.

(4) Type:

makefs <ENTER>

The following menu will appear:

```
Make a File System on...
------------------------

  1  a floppy disk
  2  a hard disk
  3  a disk cartridge

  q  to quit
```

(5) We want to make a filesystem on a hard drive, so type:

2 <ENTER>

(6) **makefs** will prompt:

On which hard drive (Ø-1)

Type:

1 <ENTER>

to specify the second hard drive. At this point, **makefs** will display the partition table (remember that?) of the secondary, and asks:

On which XENIX partition (1-4 or <DEL> to abort)?

Since we want to make a filesystem on the second partition, we'll say:

2 <ENTER>

If you have more than one Xenix partition, you'll need to repeat this process for each of them.

(7) At this point, the screen will show a message something like this:

```
About to build a file system on partition X of drive X.
The file system will occupy XXXXX blocks.
Press <RETURN> to continue or <DEL> to abort:
```

We want to go ahead, so press **<ENTER>** to continue. The secondary drive should make lots of light-flashes and beeping noises, and eventually, you should be returned to the main menu.

(8) When you are back at the main menu, type:

q <ENTER>

to quit and return to the shell.

Mounting File Systems:

At this point, you're done! But you still can't access the secondary since it has not yet been made part of the root file system. In order to access files located within a secondary hard drive, or for that matter, a mountable floppy or cartridge disk, the drive in question must first be **mounted** on the root file system. That is why we made that empty directory a while ago. It is fairly standard practice to try to make directories to mount your additional drives on which reflect the name of the drive. We might have just as well named the destination directory "zeppo", but that doesn't make it easy to determine casually where your secondary hard drive is located!

You have to mount every secondary file system you want to access. To do this, type the following:

/etc/mount /dev/usr2 /usr2 <ENTER>

This command **mounts** the partition called **/dev/usr2** on the directory called **/usr2.** At this point, you can access anything on the secondary by looking in directory /usr2, although at this point you won't find anything except a directory called "lost+found" which is used by the system in cases where you are trying to recover from a crash.

Unmounting:

If you need to remove or turn off a mounted secondary file system, you must first unmount it. The **shutdown** command automatically unmounts all drives, so if you are using that command to shut a system down, it is not necessary to do any unmounts. The correct syntax to unmount a drive (like the one we mounted above) otherwise is:

/etc/umount /dev/usr2 <ENTER>

I really do mean for you to type "/etc/umount"; that's the way the command is spelled.

If you have additional drives mounted on the system, either floppy or cartridge, or additional secondary drive partitions, you must unmount them, too. The syntax is identical to what we used above, but instead of "/dev/usr2" you will substitute the appropriate device specification.

****

SECTION SIX

Hints and Kinks
or
How to overcome various system anomalies
in your spare time.

****

.....

"Space, the final frontier..."

.....

Periodically (just to keep you on your toes), the system you find yourself involved with may transport you into places where you are sure no technician has ever gone before. This new frontier may hit you with illogical hardware failures, dazzling displays of software prestidigitation totally unlike that which was intended, or, if things are really going well, both. When you find yourself in such a confrontation, take heart. There are frequently reasons, and sometimes even solutions, for such problems. Below is listed a cross-section of the more exotic software bugs, operational problems, cures, and fixes.

File Structure Cleaning:

Before Xenix can utilize or mount a device (primary or secondary hard drives and partitions thereof, floppy drives, and cartridge drives) that device must be "clean". By "clean" I mean that the file system on that particular device must have been shut down correctly. Look at it like this: the difference between a "clean" and an "unclean" file system is kind of like the difference between a file cabinet with every folder filed away in an orderly fashion, and one that someone tore through in a panic looking for their 1979 income tax records... the first one is obviously going to be the one where you'll actually be able to find something later (unless you're as disorderly as I am). Xenix has a neatness fetish, however, so before you can hope to find anything using it, you have to start with a tidy system.

Sometimes, for a plethora of reasons, a file system on a device may become corrupted, or unclean. When booting, Xenix will check the "root" and "usr" partitions on the primary hard drive to insure that those file systems are intact. If the root file system is not clean, the system will ask for permission to clean it. The usr file system will automatically be cleaned when the system goes multiuser. If the system is autobooting, it cleans the root file system without asking for permission; the reasoning here is that the computer is probably not attended by a human operator (which is likely to be why it autobooted) and so might as well go ahead with the cleaning process. The "gotcha" is that the system will not automatically clean the partition or partitions on the secondary hard drive (if installed) and this is what may become the source of some customer/technician confusion. Consider the following example:

A particular Xenix installation contains a 40 meg internal primary, divided into a root and usr partition, and a 70 meg external secondary, which is being treated as one large partition we'll call "/dev/usr2". Upon booting the system, this secondary partition is automatically mounted by way of the /etc/rc.user file. The customer, under normal operating conditions, should never need to manually mount the partition on the secondary hard drive.

Unfortunately, one day the gas company manages to sever the power cable into this facility while looking for a gas line, causing a temporary power outage and crashing the Xenix system in the bargain. When the lights come back on, the customer is confronted with a forbidding message asking if the file structure should be cleaned. He responds in the affirmative and several minutes later, when the cleaning has been completed, types "<CTL><D>" to bring the system up multi-user. The computer now informs him that it is cleaning /dev/usr. Finally, the system boots, and things return to normal... sort of.

Later, the customer tries to access the information which should be in /dev/usr2, mounted on /usr2. He finds an empty directory. He panics. His secretary panics. The customer calls you spitting sparks out both ears. "My system is doing something WIERD!!" he bellows. Plaster falls from the ceiling, the phone receiver cracks in your hand, and the neighborhood Xenix guru is on vacation in Tibet this week... congratulations! You have just entered that "Xenix Zone" you've heard about....

The problem, of course, is that only the partitions on the primary hard drive were cleaned, and not the ones on the secondary. Fortunately, this is an easy problem to solve. Follow the steps listed below:

1) **fsck /dev/**<u>**partition-name**</u> **<ENTER>**
   <u>partition-name</u> is the name(s) of the partition(s) being used for Xenix on the secondary hard drive.

2) mount the secondary partition(s) manually using the **mount** command.

The most common reason for a device to need cleaning is improper shutdown of the system. Simply hitting reset or the power switch without first performing the **shutdown** command will almost always corrupt the file systems. Removing **mounted** floppies or cartridges without first **umount**ing them is just as bad.

To clean a floppy diskette or a cartridge the procedure is much the same:

1) **fsck /dev/fdØ <ENTER>**
   fdØ is for a high-capacity floppy drive Ø
   fd1 is for a high-capacity floppy drive 1
   ... etc.
   cdØ is for cartridge drive Ø
   cd1 is for cartridge drive 1

2) Mount or install the floppy or cartridge as normal.

Various Line Printer Problems:

Underneath this heading resides two major categories of potential problems:

Line Printers That Will Not Print
and
Line Printers That Will Not Stop Printing

These problems are somewhat related, and can be quite frustrating if you've never dealt with them before. We will take the first problem first, and for the sake of argument assume that the hardware is flawless.

As I mention elsewhere, the line printer software in System 5 is considerably more complex than it was in our earlier versions of Xenix. To begin with, you have to "initialize" a printer before you can print on it, and even if the printer in question has been initialized, you have to worry about whether or not it has been **enabled**. Sounds a lot like a terminal, doesn't it? Some "printer not printing" problems can be as simple as the fact that the system printer hasn't been initialized yet. If the printer has been initialized and enabled, you might have a situation where the line printer scheduler, **lpsched**, is not running. Since **lpsched** is kind of the straw boss of the printing system, nothing will get done if he's not awake. Additionally, if the actual printing program **lpr** can't for some reason get to the files and/or directories in **/usr/spool/lp**, which is the "holding area" for spooled print requests, the whole printing process will come to a screaming halt.

.....

There are a lot of ways the printer
can refuse to print in System 5.

(Fortunately, most of them are easy to fix!)

.....

Anyway, if the printer isn't printing, consider the following steps:

1) Log in as root.

2) Type:

   lpstat -t <ENTER>

3) There are a variety of things possible to see returned by the above command. If you see:

   ```
   scheduler is not running
   no system default destination
   ```

   the odds are good that the system printer has not been initialized. In this case, refer to Section 4 of this manual, and try the sequence of steps outlined there to initialize the printer.

4) Another possible response to step (2) is:

   ```
   scheduler is not running
   system default destination: linepr
   device for linepr: /dev/lpTØ
   linepr accepting requests since Aug  8 1Ø:12
   printer linepr is idle.  enabled since Aug  8 1Ø:12
   ```

   The **boldface** is my doing... it's to point out what is probably wrong here. If you see something like this, the problem is likely the fact that the line printer scheduler **lpsched** is not running.

4a) To start **lpsched** running, type:

   /usr/lib/lpsched <ENTER>

4b) Check to see that the scheduler is now running by typing:

   lpstat -r <ENTER>

4c) If you see:

    scheduler is running

then try printing something out:

    lpr /etc/ttys <ENTER>

You should get a request id in response, something like:

    request id is linepr-2Ø (1 file)

and the printer should eventually print your job.

4d) If you see:

    scheduler is not running

in response to (4b), then try this:

    rm -f /usr/spool/SCHEDLOCK <ENTER>
    /usr/lib/lpsched <ENTER>
    lpstat -r <ENTER>

You should now see:

    scheduler is running

and should be able to print something out.

5) Another possible response to the "**lpstat -t <ENTER>**" command is this:

```
scheduler is running
system default destination: linepr
device for linepr: /dev/lpTØ
linepr accepting requests since Aug  8 1Ø:12
printer linepr disabled since Aug  9 1Ø:33 -
        reason unknown
```

Again, the **boldface** is mine... this indicates that the lineprinter is disabled. To correct this, type:

    enable linepr <ENTER>

If your printer is named something other than linepr, substitute your printer name for linepr in the above.

The system should respond with:

```
printer "linepr" now enabled
```

If you want to confirm this another way, try typing:

lpstat -p <ENTER>

The screen will show something like:

```
printer linepr is idle.  enabled since Aug  9 12:1Ø
```

6) If you get something like this when you do step (2):

```
scheduler is running
no system default destination
device for linepr: /dev/lpTØ
linepr accepting requests since Aug  8 1Ø:12
printer linepr is idle.  enabled since Aug  9 12:1Ø
```

your problem is that the printer system doesn't have a default destination. This means that lpr won't know where to send the output unless explicitly told, so if you rely on the default by typing something like "**lpr /etc/ttys <ENTER>**" you will get the response "lpr: no system default destination" and your printout will be aborted.

To fix this problem, you type:

/usr/lib/lpadmin -dlinepr <ENTER>

You can substitute the name of the printer on your system if it is different. To verify that the system has a default destination again, type:

lpstat -d <ENTER>

and you should see:

```
system default destination: linepr
```

7) Another possible source of difficulty will show up on the screen like this:

```
scheduler is running
system default destination: linepr
device for linepr: /dev/lpTØ
linepr not accepting requests since Aug  9 14:Ø1 -
        reason unknown
printer linepr is idle.  enabled since Aug  8 12:1Ø
```

What's this mean? Well, it means that even though the scheduler is running and the line printer "linepr" is enabled, the default system printer is not currently accepting requests. To fix this, type the following:

/usr/lib/accept linepr <ENTER>

The screen should respond with:

destination "linepr" now accepting requests

If you wish to check things out further, try:

lpstat -a <ENTER>

and in response you will see something like:

linepr accepting requests since Aug 11 11:Ø8

8) If after all of this, the printer **still** doesn't work, check to make sure that the files and directories in **/usr/spool/lp** are readable and writable (i.e. owned by) **lp**. In other words, when you type:

l /usr/spool/lp <ENTER>

you ought to see something like:

```
total 26
prw-------  1 lp    bin            Ø Aug 11 11:16 FIFO
-r--r--r--  1 lp    bin            2 Aug  8 1Ø:12 SCHEDLOCK
drwxr-xr-x  2 lp    bin           32 Jun 2Ø 1Ø:35 class
-rw-r--r--  1 lp    bin            7 Aug  8 13:43 default

etc.......
```

The **boldface** column is most important here. It indicates that **lp** is the owner of all these files and directories. If they were not owned by **lp**, you might run into some problems

when you attempted to print something out.

As a final note in all of that, there are printer log files available under System 5. They are roughly equivalent to **/usr/adm/messages** in that they log printer activity and errors. To get a pretty good idea of what may be going on in the system you are working on, you may wish to read these files. To do so, type:

more /usr/spool/lp/log <ENTER>

This is the record of activity since the print job scheduler, **lpsched**, was last started up. There is also a copy of the old log file (from the previous startup) which you may wish to look at. To read it, type:

more /usr/spool/lp/oldlog <ENTER>

If you cannot get the printer to work in Xenix, try it again under MS-DOS and the diagnostics. If it works there, try re-booting Xenix and testing again; if it doesn't work in MS-DOS, the odds are very good that you have a hardware problem.

NOTE: Daisy Wheel II's will **not** work with System 5 if connected directly to the 3ØØØ. They will work if you use a PTC-64 printer buffer between the printer and the 3ØØØ, however. This can be quite confusing, since the same printer and computer will work beautifully in an MS-DOS environment. The reason for this is that MS-DOS uses a polling method to talk to printers, whereas Xenix uses interrupts. Unfortunately, the DWII doesn't support the use of interrupts, so... it won't work by itself in the Xenix environment.

.....

NOTE:

Daisy Wheel II's will **not** work with System 5
if connected directly to a 3ØØØ.

They will work if you use a PTC-64 printer buffer
between the printer and the 3ØØØ.

.....

Once in a while a user will queue up a massive printer job that he really doesn't want. Fortunately, getting rid of it is easy in System 5. To abort the process:

1) Find out what the request-id number is for the process which is to be aborted. If it is not known, type:

   lpstat -o <ENTER>

   and you will get a listing of all the currently queued print requests which will look something like this:

```
linepr-22       root          1ØØ2345 Aug 11 11:55 on linepr
linepr-23       samw               325 Aug 11 11:59
linepr-24       ellie             1Ø89 Aug 11 12:Ø5
```

2) The job which is currently printing is the one we wish to abort. That job, denoted by the "on linepr" remark, is "linepr-22". To kill it, type:

   cancel linepr-22 <ENTER>

   and the print job will stop (after the printer empties its buffer).

That's all there is to it. The only thing to be wary of is the fact that the **cancel** command doesn't reset top-of-form, so the next printout may start in the middle of a page.

Failure to Reinstall an Application Package:

If, for some reason, you or the customer attempt to reinstall an application package onto the Xenix system... and it won't go in, there may be a reason. There is a file, **/etc/logbook**, which contains information on the application programs and their current version numbers. During the **install** procedure, this file is updated to reflect the current version number of the installed package. By **cat**ting this file, the user will see at a glance all of the current packages in place (or, at least, most of them -- some applications may not update this file), the date of installation, and their respective version numbers.

.....

During the install,
if the version number of the package in question
matches an entry in /etc/logbook,
the procedure will frequently abort.

.....

when you attempted to print something out.

As a final note in all of that, there are printer log files available under System 5. They are roughly equivalent to **/usr/adm/messages** in that they log printer activity and errors. To get a pretty good idea of what may be going on in the system you are working on, you may wish to read these files. To do so, type:

more /usr/spool/lp/log <ENTER>

This is the record of activity since the print job scheduler, **lpsched**, was last started up. There is also a copy of the old log file (from the previous startup) which you may wish to look at. To read it, type:

more /usr/spool/lp/oldlog <ENTER>

If you cannot get the printer to work in Xenix, try it again under MS-DOS and the diagnostics. If it works there, try re-booting Xenix and testing again; if it doesn't work in MS-DOS, the odds are very good that you have a hardware problem.

NOTE: Daisy Wheel II's will **not** work with System 5 if connected directly to the 3ØØØ. They will work if you use a PTC-64 printer buffer between the printer and the 3ØØØ, however. This can be quite confusing, since the same printer and computer will work beautifully in an MS-DOS environment. The reason for this is that MS-DOS uses a polling method to talk to printers, whereas Xenix uses interrupts. Unfortunately, the DWII doesn't support the use of interrupts, so... it won't work by itself in the Xenix environment.

.....

NOTE:

Daisy Wheel II's will **not** work with System 5
if connected directly to a 3ØØØ.

They will work if you use a PTC-64 printer buffer
between the printer and the 3ØØØ.

.....

Once in a while a user will queue up a massive printer job that he really doesn't want. Fortunately, getting rid of it is easy in System 5. To abort the process:

1) Find out what the request-id number is for the process which is to be aborted. If it is not known, type:

   lpstat -o <ENTER>

   and you will get a listing of all the currently queued print requests which will look something like this:

```
linepr-22       root          1ØØ2345 Aug 11 11:55 on linepr
linepr-23       samw               325 Aug 11 11:59
linepr-24       ellie             1Ø89 Aug 11 12:Ø5
```

2) The job which is currently printing is the one we wish to abort. That job, denoted by the "on linepr" remark, is "linepr-22". To kill it, type:

   cancel linepr-22 <ENTER>

   and the print job will stop (after the printer empties its buffer).

That's all there is to it. The only thing to be wary of is the fact that the **cancel** command doesn't reset top-of-form, so the next printout may start in the middle of a page.

Failure to Reinstall an Application Package:

If, for some reason, you or the customer attempt to reinstall an application package onto the Xenix system... and it won't go in, there may be a reason. There is a file, **/etc/logbook**, which contains information on the application programs and their current version numbers. During the **install** procedure, this file is updated to reflect the current version number of the installed package. By **cat**ting this file, the user will see at a glance all of the current packages in place (or, at least, most of them -- some applications may not update this file), the date of installation, and their respective version numbers.

.....

During the install,
if the version number of the package in question
matches an entry in /etc/logbook,
the procedure will frequently abort.

.....

when you attempted to print something out.

As a final note in all of that, there are printer log files available under System 5. They are roughly equivalent to **/usr/adm/messages** in that they log printer activity and errors. To get a pretty good idea of what may be going on in the system you are working on, you may wish to read these files. To do so, type:

more /usr/spool/lp/log <ENTER>

This is the record of activity since the print job scheduler, **lpsched**, was last started up. There is also a copy of the old log file (from the previous startup) which you may wish to look at. To read it, type:

more /usr/spool/lp/oldlog <ENTER>

If you cannot get the printer to work in Xenix, try it again under MS-DOS and the diagnostics. If it works there, try re-booting Xenix and testing again; if it doesn't work in MS-DOS, the odds are very good that you have a hardware problem.

NOTE: Daisy Wheel II's will **not** work with System 5 if connected directly to the 3ØØØ. They will work if you use a PTC-64 printer buffer between the printer and the 3ØØØ, however. This can be quite confusing, since the same printer and computer will work beautifully in an MS-DOS environment. The reason for this is that MS-DOS uses a polling method to talk to printers, whereas Xenix uses interrupts. Unfortunately, the DWII doesn't support the use of interrupts, so... it won't work by itself in the Xenix environment.

.....

NOTE:

Daisy Wheel II's will **not** work with System 5
if connected directly to a 3ØØØ.

They will work if you use a PTC-64 printer buffer
between the printer and the 3ØØØ.

.....

Once in a while a user will queue up a massive printer job that he really doesn't want. Fortunately, getting rid of it is easy in System 5. To abort the process:

1) Find out what the request-id number is for the process which is to be aborted. If it is not known, type:

   lpstat -o <ENTER>

   and you will get a listing of all the currently queued print requests which will look something like this:

```
linepr-22       root          1ØØ2345 Aug 11 11:55 on linepr
linepr-23       samw               325 Aug 11 11:59
linepr-24       ellie             1Ø89 Aug 11 12:Ø5
```

2) The job which is currently printing is the one we wish to abort. That job, denoted by the "on linepr" remark, is "linepr-22". To kill it, type:

   cancel linepr-22 <ENTER>

   and the print job will stop (after the printer empties its buffer).

That's all there is to it. The only thing to be wary of is the fact that the **cancel** command doesn't reset top-of-form, so the next printout may start in the middle of a page.

Failure to Reinstall an Application Package:

If, for some reason, you or the customer attempt to reinstall an application package onto the Xenix system... and it won't go in, there may be a reason. There is a file, **/etc/logbook**, which contains information on the application programs and their current version numbers. During the **install** procedure, this file is updated to reflect the current version number of the installed package. By **cat**ting this file, the user will see at a glance all of the current packages in place (or, at least, most of them -- some applications may not update this file), the date of installation, and their respective version numbers.

.....

During the install,
if the version number of the package in question
matches an entry in /etc/logbook,
the procedure will frequently abort.

.....

To get around this problem, it will be necessary to edit the **/etc/logbook** file. Use the editor to remove the complete line that contains the reference to the application package and its version number. Remember, the version number of the floppy package and the version number of the logbook entry must be the same for this problem to occur. (It may also occur if you attempt to install an older version of the application over a newer version already present on the system).

After the edit is complete and the modified file written with the **w** command, reinstall the package as usual.

Losing the Screen Display:

You may find yourself getting a call sometime from a panic-stricken customer who whimpers, "Scripsit dumped me out... and now I don't get anything on my screen! What do I do?" You start chewing your nails and worrying about video problems... but before you leap into monitor theory, try this. This problem is not uncommon if a program terminates abnormally. You may be able to restore the screen display as follows (provided, of course, that the hardware really is ok):

1) Type **<CTL><Q>**. This restarts transmission from the computer if you accidentally typed <CTL><S>.
2) Now, press **<ENTER>** a few times.

If the screen is still blank, continue with these steps.

3) Type **<CTL><J>**. You may get an error message on the screen; if you do, ignore it.
4) Now type:

stty sane <CTL><J>

Your screen display should return.

If this still doesn't do it, your best bet is to go to another terminal and kill the process attached to the terminal and log in again on the affected terminal.

That discussion brings us rather nicely to the next in the list of possible problems:

Runaway Processes and What to Do About Them:

A runaway process is a lot like a runaway horse: sometimes the only thing that will stop it is shooting it down. The equivalent of a gun in Xenix is the **kill -9** command.

.....

kill -9 terminates with extreme prejudice.
(Who needs ØØ7?)

.....

If you find yourself facing a process that you just can't stop, try the following:

1) Type **<CTL><C>**. Probably you've already tried that at this point, but one more time will rarely hurt.
2) If the process hasn't terminated, go to another terminal on the system and log in as root, and look at the processes currently running by typing:

   ps -elf <ENTER>

   Look for the process number of the runaway.
3) Try a subtle approach first. Type:

   kill -2 PID <ENTER>

   Wait a few seconds, then do a **ps -elf <ENTER>** again to see if the process has gone away. If it hasn't, type:

   kill -1 PID <ENTER>

   Wait a few more seconds, and see if the process has stopped. If not, type:

   kill -9 PID <ENTER>

   This will kill it... but it isn't very subtle, and some application packages get a little irate about abnormal termination, so you may have to do a little repair work like removing lock files to get things back to normal.

A final note on the subject of runaway processes: if you have something like this occur when you are running Xenix on a system which only has a console and no terminals, you may have to press the reset switch to get things to come back. This is obviously a last resort, inasmuch as it closes the system down in a decidedly incorrect fashion. You will at the very least have to clean the system before bringing it back up, and you may have other problems to contend with as well. Try to avoid this if possible!

Local Printing Problems:

Local printing is a nice feature offered by our versions of Xenix... it means that you can avoid having to run over to the console printer (or printers) every time you want to see a rough draft of a memo. However, there are a lot of things which will cause it not to work, or to work in a rather brain-damaged fashion. Here are a few of the things to look for when you've got a system with this difficulty.

1) First thing to check for is the modifications to allow successful local printing on both the DT1 and the DT1ØØ:

```
I/O:66  Allow local printing on DT1
I/O:76  Correct local printing problem on DT1ØØ
I/O:98  Correct DT1 holding printers in RESET
```

2) Make sure that Xenix is aware of the correct terminal type on the ports where local printing is to be used.

```
.........................................

  If you think screen output looks funny
when you give Xenix the wrong terminal type,
     wait until you try local printing!
.........................................
```

3) It is possible for Xenix to spew out information faster than some of our printers can keep up with. This causes funny looking results like dropped characters, misunderstood characters, and the like. There are a couple of ways around this problem:

   a) Slow the rate at which the terminal is talking to the Xenix system -- for example, if you are communicating at 96ØØ baud, try 48ØØ or even 24ØØ baud instead.

b) Add a "printer buffer" (like our PTC-64 printer controller) to the system between the terminal and its associated printer. What this will do is allow you to communicate at 96ØØ baud without overrunning the local printer... the printer buffer will feed information out to the printer at the rate it can cope with.

Terminals Not Clearing the Screen After Logoff:

At times you may hear complaints from customers about the fact that their terminals don't clear the screen when they log off... yet all the hardware passes all the checks you can think of. Well, not to worry. It turns out that terminals don't automatically clear the screen when the user logs out in System 5 and, unfortunately, there really isn't much we can do about this. As a possibly helpful note, there is a utility named **cls** which the customer may use to manually clear his screen before logging out. However, it will only work properly if the terminal type for that terminal is set correctly -- for the same reasons that Scripsit or the **vsh** will look funny if the terminal type is set wrong. Sending a DT1 "clear screen" command to a DT1ØØ won't accomplish much!

If the customer is really into Xenix, they can modify the entry in /etc/gettydefs for a given baud rate so that the screen will clear... but that particular operation is getting a bit beyond the scope of this manual.

.....

When **shutdown** Won't Work

or

In Times of Extreme Frustration,
How to Pull The Plug on Xenix
Gracefully.

.....

It may someday be your fate to deal with a system where **shutdown** just won't work. You enter the command, see the "broadcast message" displayed on the console, hear various whirrs, clicks, and moans from the hardware -- and then the system should shut down, right? Instead, nothing happens... and you think, "What now?"

Generally, when this occurs, hitting the <CTL><C> or <CTL><_> key combinations will return the root prompt. If it does, make sure all users are logged off and then type the following to gracefully halt the system:

sync;sync;/etc/haltsys <ENTER>

Although not as neat and tidy as **shutdown**, this will be more than sufficient to shut the system down cleanly and keep the filesystems intact for the next boot.

What if this doesn't work? If you can't get the root prompt back, truck on over to another still functioning terminal, log in as root, and perform the **sync;sync;/etc/haltsys** command. If this still doesn't work... all you can do is reset.

****

SECTION SEVEN

Advanced System Information

and

Restoring Blown Xenix File Systems

or

How to Really Mung Things Up

****

.....

Read This
BEFORE
Reading This Section

.....

This section is provided mainly as reference material for persons already familiar with the Xenix operating system. In this section, I will not walk you through step by step -- I am operating under the assumption that you will follow what I'm talking about. What I will give you are rather sketchy outlines of things which may need to be done under certain circumstances. The material in this section is optional and if you are not **very familiar** with this operating system, I would strongly suggest that you browse this section, but NOT implement any of it until you are well grounded in Xenix.

That having been said, I should also warn those who are familiar (or think they are) with Xenix that the information contained herein may help you out in some instances. It may also make matters worse. The standard operating procedure for these situations should be **extreme caution**. When in doubt, don't try it -- make sure you understand fully what you are doing before you do it. With this stuff, discretion is frequently the better part of valor.

A final note: the information in this section is not guaranteed to be accurate or to work with all versions and all hardware configurations all the time. Some of it you will simply not be able to do without access to the Software Development System programs. If it sounds like I'm trying to throw some healthy trepidation at you, you're quite right -- I am. Tread lightly.

All of the things I will discuss in this section can be accomplished in some fashion by using the Xenix Tools Disk. Under most circumstances, it is advisable to use the Tools Disk, since it has a fairly user-friendly interface and does most of the hard work for you. If you find yourself without a Tools Disk, however, this information may prove helpful.

.....

If you don't understand what I'm talking about,
don't even **think** of trying it!

.....

.....

Flaw Maps

Was it entered? Was it entered correctly?
How can you tell?

.....

Flaw maps are pretty important to Xenix. These maps tell the system where it probably can't write and hope to have a chance of reading the data back at some future time. Sure, **badtrack** will do a verify on all of the tracks it reads, but some flaws will just not be picked up automatically. A media retention problem will generally not be found by a media verification program like **badtrack**. However, difficulties of this nature must be included on the flaw map.

The easiest way to tell if the flaw map is correctly locked out is to use the Xenix Tools Disk. There is a program on this disk which will save the effort of decoding the media error map for yourself (floppy menu option "m", hard disk menu option "s"). This is far and away the preferable method of checking flaw maps.

If you have access to the Software Development System, and are curious about where this information is stored, read on... you will not be able to try this without the Software Development System.

Figures 1 and 2 show hexadecimal dumps of portions of a file called **/dev/hdØØ**. This is the primary hard drive, and the sections we are going to look at will tell us the number of cylinders, the number of heads, and what tracks have been locked out for this particular drive. If, for the sake of argument, you were interested in the secondary hard drive, the file you would look at would be **/dev/hd1Ø**. Naturally, all this stuff is going to be meaningless to the non-initiates, but hang in there -- you'll be initiated.

Log in as root and type the following command:

od -d /dev/hdØØ +13 | head <ENTER>

You'll only get ten lines, but that's plenty for our purposes. What you should see is something similar to what appears in Figure 1, although it obviously won't be identical.

Now, before we do anything else, type:

od -x /dev/hdØØ +1ØØØ | head <ENTER>

You will get ten more lines which will look similiar to Figure 2.

Now, let's check out Figures 1 and 2. Look pretty impressive, eh? The big question is, "What do they mean?" I'll give you a hint: Figure 1 contains the cylinder number/head number information, and Figure 2 contains the bad track map. Figure 3 corresponds to the flaw map for this particular drive. Take a look at these and see if you can see a correspondence... but don't hurt yourself in the process! It isn't obvious until the tricks are explained to you.

```
# od -d /dev/hdØØ +13 | head

ØØØØØ13 ØØ615 ØØØØ4 6528Ø ØØ255 ØØØØØ ØØØØØ ØØ615 ØØØ17
ØØØØØ33 ØØØØØ ØØØØØ ØØØØØ 36864 37ØØ8 47354 ØØØØØ 49294
                    etc....
```

Figure 1.

```
# od -x /dev/hdØØ +1ØØØ | head

ØØØ1ØØØ Ø4dØ Ø73e Ø963 Ø995 ØØØØ ØØØØ ØØØØ ØØØØ
ØØØ1Ø2Ø ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ
*
ØØØ176Ø ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ bdbd ØØØØ
                    etc....
```

Figure 2.

Flaws Entered Into Flaw Map on HDØ
(2Ø meg with 615 cylinders and 4 heads)

Cylinder	Head
3Ø8	Ø
463	2
6ØØ	3
613	1

Figure 3.

Figure it out? Well, first off, what is the command **od**? This command is an **octal dump** of whatever file you specify as an argument. For our purpose, we need to work first in decimal, then in hexadecimal, so that's what the arguments to **od** do for us. A **-x** forces a hexadecimal dump. A **-d** will force a decimal "short" (16 bit) dump. What we have done is two different dumps of /dev/hdØØ, the first with an offset of 13 octal and output in decimal, the second with an offset of 1ØØØ octal and output in hex.

Now, let's pick the dump in Figure 1 apart.

```
ØØØØØ13 ØØ615 ØØØØ4 65528Ø ØØ255 ØØØØØ ØØØØØ ØØ615 ØØØ17
ØØØØØ33 ØØØØØ ØØØØØ ØØØØØ 36864 37ØØ8 47354 ØØØØØ 49294
```

The two things that matter in the above are the items in boldface. You can ignore the rest.

The first group of 7 digits on the left of each line is an octal representation of the position in the file. 2Ø Octal is equivalent to 16 decimal. Each 5 digit decimal number will take two hex bytes to store resulting in a total of 16 bytes per line. You'll notice that the line numbers increment in this example by 2Ø octal, or 16 decimal, so it works out. You'll also notice that the dump starts at 13 octal, which is the offset we gave **od**, so that works out, too.

Now, the reason we care about this is that we have arrived at our first two interesting bits of information. The first one is the number of cylinders in the drive, and the second is the number of heads. The cylinder count is at offset ØØØØØ13 octal, and is an integer, which in this instance means that it takes up two bytes, or 16 bits. The head count is in the next two bytes, at offset ØØØØØ15 octal.

In other words, looking at the first line:

```
ØØØØØ13 ØØ615 ØØØØ4 65528Ø ØØ255 ØØØØØ ØØØØØ ØØ615 ØØØ17
         |     |
         |     |_____ number of heads is 4
         |
         |___________ number of cylinders is 615
```

Pretty easy, wasn't it? Well, don't get too complacent, because the next part is a bit more difficult.

The next dump we did is the actual flaw map. Each cylinder/track pair takes up 2 bytes (16 bits) of space. The least significant bits will be the head for a given error, and the minimum number of bits possible will be used to describe it. In other words, for a drive with 4 heads, the 2 least significant bits will be the head number. For a drive with 8 heads, the 3 least significant bits will be used for the head number, and so forth. The remainder of the number will be the cylinder on which the error occurs.

This all sounds rather confusing, so let's go ahead and take a look at the second dump and break some of the entries down:

```
ØØØ1ØØØ Ø4dØ Ø73e Ø963 Ø995 ØØØØ ØØØØ ØØØØ ØØØØ
ØØØ1Ø2Ø ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ
*
ØØØ176Ø ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ ØØØØ bdbd ØØØØ
```

The first thing I will have you note is the bdbd ØØØØ at offset ØØØ1774. Those 4 bytes are the marker for the end of the bad track map, and they will always be there even if your drive has no errors locked out at all.

Next, let's look at the first line of the dump:

```
ØØØ1ØØØ Ø4dØ Ø73e Ø963 Ø995 ØØØØ ØØØØ ØØØØ ØØØØ
```

Remember, I said that each bad track would take up two bytes in this map. Therefore, we have 4 bad tracks locked out... entries which are "ØØØØ" are simply empty spots for bad track mapping. Look at the first entry, Ø4dØ.

Now we need to do a little hex to binary conversion:

```
Hex:        Ø    4    d    Ø
Binary:     ØØØØ Ø1ØØ 11Ø1 ØØØØ
```

This drive has 4 heads, so in binary, we will need only 2 bits to describe the head for a bad track. They will be the **least significant** (i.e. ones farthest to the right) bits in our binary number.

```
Binary:     ØØØØ Ø1ØØ 11Ø1 ØØØØ
                           |____ ØØ binary = Ø decimal
```

To determine the cylinder number, remove the head bits from consideration, and for convenience, the leading (e.g. leftmost) zeros:

```
Binary:     ØØØØ Ø1ØØ 11Ø1 ØØØØ = 1ØØ11Ø1ØØ
```

and treat the whole thing as a binary representation of the cylinder number. To decode it, let's remember how binary to decimal conversions are done:

```
Decimal Value:  256  128   64   32   16    8    4    2    1
Binary "bit" :    1    Ø    Ø    1    1    Ø    1    Ø    Ø
```

Wherever there is a one, you will bring the decimal value down to be added up:

```
Decimal Value: 256   128    64    32    16     8     4     2     1
Binary "bit" :   1     Ø     Ø     1     1     Ø     1     Ø     Ø
               256 +   Ø +   Ø + 32  + 16  +   Ø +   4 +   Ø +   Ø  = 3Ø8 dec.
```

So, the first flaw works out to be at cylinder 3Ø8, head Ø, which matches the first entry in Figure 3.

Similarly, the next entry, Ø73e, will look like:

```
Hex:        Ø    7    3    e
Binary:     ØØØØ Ø111 ØØ11 111Ø
                              |____ head = 2

Binary:     ØØØØ Ø111 ØØ11 111Ø
                        |_________ cylinder = 463
```

The third entry, Ø963, works out to cylinder 6ØØ, head 3, and the fourth entry, Ø995, turns out to be cylinder 613, head 1.

After doing all this, I'm sure that you see why I said it was easier to use the Tools Disk! Still, if you ever find yourself stuck without one, this might be a nice thing to know. A note for those of you who might work on the 68ØØØ based Xenix systems: this is not the same layout for bad track mapping as that used by Version 7 and System 3 Xenix! Don't expect this to work for anything except our System 5 on the 3ØØØ. Bad track mapping schemes are very implementation dependent, and expecting them to be the same everywhere is a good way to really confuse yourself fast.

.....

The bad track mapping scheme is
different for System 5!

Don't confuse it with the 68ØØØ Xenix systems.

.....

.....

RESTORING A BLOWN BOOT TRACK ON
THE
PRIMARY HARD DRIVE

or

Desperate situations call for
desperate solutions (sometimes...)

.....

You've got a Xenix system with a serious problem on your hands. The computer side of the house (the 286 and its associated hardware) is o.k., but the primary hard drive is suspect. The controller board and its alignment are good, the bubble logic board is good, the power supply voltages are good, you don't see any gross read errors under your hard drive reliability diagnostic... but the boot track is blown. Normally, the thing to do in this situation would be to reformat, reinstall Xenix and have the customer restore all of his programs and data. One problem: the customer has nineteen months on data on the hard drive and no saves. Now, if we reformat, he loses all that data. He will not be very happy about that (can you blame him?); in fact, he's likely to become pretty upset. Is there anything we can do for this guy? Well... maybe.

.....

**... if the format is what's blown, you're gonna'
have a rat's chance of success.**

.....

Since for all practical purposes we've got a doomed system anyway, we can tinker a little bit and maybe get the system up long enough to save his data. **Be aware that this procedure may, or may not, work; you should explain this fact to the customer before any attempt is made to restore the boot track.** Actually, the initial attempt will be to restore just part of the boot track, and if the format is what's blown, you're gonna' have a rat's chance of success.

We'll first examine the possibility that this is a situation where you can use the Xenix Tools Disk. This implies that you have a copy of the Tools Disk with the **same version of Xenix as that on the hard drive.** If these criteria are met, follow these steps:

1) Boot the Xenix Tools Disk. When you reach the floppy menu, select menu option "n".
2) Shut the system down. Remove the Tools Disk and attempt to **boot off the hard drive. If the system boots, go immediately and do a save.** Don't wait; there is absolutely no guarantee that this will keep the system running for any length of time. Once a successful save has been performed, shut the system down and reformat and reinstall.

If you **can't** use a Tools Disk, try this:

Step One:

You'll need a floppy copy of a Xenix Installation Diskette, of the **same version Xenix that is resident on the hard drive.** With this in hand, install it in floppy drive A (or Ø) and boot the system on floppy.

At the prompt, press **<ENTER>** to boot into floppy Xenix.

Step Two:

At this point, you should have the root prompt (#). If you don't, then you did something wrong, and should start over. If you do, then enter these commands:

1) **/etc/mount /dev/hdØa /mnt <ENTER>**
This command will mount the root filesystem on the floppy drive. If it is successful, you should get a root prompt back. If it is unsuccessful, you will get an error message which may look something like this:

```
mount: structure needs cleaning
```

or

```
mount: bad super block
mount: Error Ø
```

If you get the latter error message, the chances of this procedure working are very small indeed.

If at this point, you have the **root prompt with no error message,** proceed to step (4). If you got an error, go to step (2).

2) **/bin/fsck -y /dev/hdØa <ENTER>**
This command will attempt to clean the root filesystem. It will do so without prompting you for permission to remove things which are damaged beyond repair; you will just need to sit back and let it work. This cleaning procedure should go through at least 5 and possibly 6 phases and should end with a message saying:

xxxx files xxxx blocks xxxx free **(message a.)**

where the "xxxx"'s are numbers which will vary from site to site. If you do not see something like this, but instead see:

UNKNOWN FILE SYSTEM VERSION xxxx **(message b.)**

you will not be able to finish this procedure successfully.

If you have **message a.** and a root prompt, proceed to step (3). If you got **message b.**, use this command to shut the floppy disk down and abort the procedure, because you will not be able to continue:

/bin/sync;/etc/haltsys <ENTER>

3) **/etc/mount /dev/hdØa /mnt <ENTER>**
This should mount the root filesystem on the floppy with no errors. You should see a root prompt for response.

4) **/etc/ib /dev/hdØØ a /etc/masterboot /etc/hdbootØ /etc/hdboot1 <ENTER>**
/bin/copy /etc/hdboot /mnt/boot <ENTER>
These commands create and copy fresh boot information back onto the primary hard drive.

5) **/etc/umount /dev/hdØa <ENTER>**
This unmounts the root filesystem on the hard drive.

6) **/bin/sync;/etc/haltsys <ENTER>**
This command will shut down the floppy-based Xenix system.

7) You should now see the **"Normal System Shutdown"** message.

8) Press **RESET** to boot from the primary hard drive.

If the copy was successful, Xenix should boot from the hard drive. If it does, **go immediately and do the save.** If it doesn't... well, you gave it your best shot. There is one more desperation maneuver you can try in this instance, although the chances of success are getting more remote. As well as refreshing the boot track, we can try recopying Xenix to the hard drive as well.

First, here's the procedure if you have a Tools Disk:

1) Boot the Xenix Tools Disk. When you reach the floppy menu, select menu option "o".
2) Shut the system down. Remove the Tools Disk and attempt to **boot off of the hard drive. If the system boots, go immediately and do a save.** After you've performed the save, shut the system down, reformat, and reinstall.

If you can't use or don't have a Tools Disk, try this:

Step One:

You'll need a floppy copy of a Xenix Installation Diskette, of the **same version Xenix that is resident on the hard drive.** With this in hand, install it in floppy drive A (or Ø) and boot the system on floppy.

At the prompt, press **<ENTER>** to boot into floppy Xenix.

Step Two:

At this point, you should have the root prompt (#). If you don't, then you did something wrong, and should start over. If you do, then enter these commands:

1) **/etc/mount /dev/hdØa /mnt <ENTER>**
   If at this point, you have the **root prompt with no error message,** proceed to step (4). If you got an error, go to step (2).

2) **/bin/fsck -y /dev/hdØa <ENTER>**
   This command will attempt to clean the root filesystem. This cleaning procedure should go through at least 5 and possibly 6 phases and should end with a message saying:

   xxxx files xxxx blocks xxxx free **(message a.)**

   where the "xxxx"'s are numbers which will vary from site to site. If you do not see something like this, but instead see:

   UNKNOWN FILE SYSTEM VERSION xxxx **(message b.)**

   you will not be able to finish this procedure successfully.

   If you have **message a.** and a root prompt, proceed to step (3). If you got **message b.**, use this command to shut the floppy disk down and abort the procedure:

   /bin/sync;/etc/haltsys <ENTER>

3) **/etc/mount /dev/hd0a /mnt <ENTER>**
This should mount the root filesystem on the floppy with no errors. You should see a root prompt for response.

4) **/etc/ib /dev/hd00 a /etc/masterboot /etc/hdboot0 /etc/hdboot1 <ENTER>**
/bin/copy /etc/hdboot /mnt/boot <ENTER>
/bin/copy /xenix /mnt/xenix <ENTER>
/bin/chown bin /mnt/xenix <ENTER>
These commands create and copy fresh boot information back onto the primary hard drive. Additionally, we have just copied a fresh kernal (**/xenix**) back onto the hard drive.

5) **/etc/umount /dev/hd0a <ENTER>**
This unmounts the root filesystem on the hard drive.

6) **/bin/sync;/etc/haltsys <ENTER>**
This command will shut down the floppy-based Xenix system.

7) You should now see the **"Normal System Shutdown"** message.

8) Press **RESET** to boot from the primary hard drive.

If the hard drive still doesn't boot, you have exhausted your possibilities. The only thing left to do now is reformat the drive and reinstall Xenix. If it does boot, perform your save immediately, then reformat and reinstall.

.....

WARNING!!

Even if you can save the customer's data
after reconstructing the boot information,
there is no guarantee that it will be reliable!

(But it's better than nothing at all....)
.....

****

Saving Data and Putting It Back
In One Piece
or
How to Win Friends and Influence
Irate Customers

****

Sooner or later, almost every one of you will encounter a situation where the system you are working on will have to be reformatted -- which implies that all the data on said system is going to be sent to that great bit bucket in the sky. Being the savvy technician that you are, you say to yourself, "Hmmm-mm... better call the customer and ensure that he has a full current save of his system!" So you place the call.

.....

If the world were a perfect place,
all customers would have current backups
of their systems...
but they usually don't.

.....

If the world were a perfect place, and if all system administrators were part of the subset of humanity that believes firmly in wearing both a belt and suspenders, the answer would of course be, "Yes, I've not only got one save, I've got **two**!" Unfortunately, things don't usually work out that way, and the answer one receives is frequently more along the lines of, "You want to reformat my system!?! I've got **SIX MONTHS** of accounting records (or word processing, or spreadsheet, ad nauseaum) on there!! My business will collapse!!!" Great wailing, pounding of chests, and weeping generally ensues, and you may find yourself needing to either back this sytem up, or instruct the customer in the in's and out's of the process.

The only problem with this situation is that you don't have the foggiest notion of where to begin... so, a little discussion of "how to do Xenix backups" seems in order.

First, A Little Background...

Despite all appearances to the contrary, there are two basic ways to back up a Xenix system. These two utilities are called **tar** and **dump** (also known as **backup** in System 5). **Tar** is frequently used in utility menus in application packages to backup data specific to that package. It is a good choice for situations where you only want to save certain parts of a system, and can also be used to back up an entire system. One of the idiosyncrasies of **tar** is that it cannot save devices. Take my word for it, you don't want to try it! I will note that System 5 is intelligent enough to only give you a bunch of nasty error messages if you try this, but this wasn't always the case in the history of Xenix.... you are best off just remembering not to do this.

Dump (or **backup**) is what the utility script **/etc/sysadmin** is built around. **Sysadmin** is very useful for doing full system backups, and incremental system backups which save only those files which have been changed since the last full system save. Currently, there is a small problem with using **sysadmin**, and it involves the fact that the floppy drivers do not currently allow the successful use of the swapper. Since we can't effectively swap, we can't

build a File Maintenance Diskette, which allows swift and effective **restores** of **dumped** data. Therefore, putting a **sysadmin** save back on the system can take a painfully long time. For most purposes, and certainly all that you should encounter, a **tar** save should be fine.

.....

In the first release of System 5,
putting a **sysadmin** save back on the system
can take a painfully long time.

Subsequent releases should fix this problem.
.....

Incidentally, the reason that restoring a **sysadmin** save without benefit of a File Maintenance Disk is so arduous is as follows: **sysadmin** restores only work quickly on inactive file systems, by this I mean file systems that you didn't boot off of initially. If you boot off of the File Maintenance Disk (a hypothetical one, since we don't have one in the first release due to the reasons I gave above), and restore to the primary hard drive from either an additional floppy drive or a cartridge drive, the process works very quickly. If you boot off the hard drive and try to restore the entire system, you are looking at possibly a 24 to 48 hour process -- not too good!

The upshot of all of this is that we will use **tar** (or related programs) to make saves for the most part. I will, however, explain how to do **sysadmin** saves so that you'll know for the future.

One more note before we start:

.....

In round numbers,

High-capacity diskettes hold 1 **meg** of data.
PC-type low-capacity diskettes hold 1/3 **meg** of data.
10 Meg Disk cartridges hold 10 **meg** of data.
20 Meg Disk cartridges hold 20 **meg** of data.
Streaming tapes hold 40 **meg** of data.

Bear this in mind as you prepare to do a save.
.....

Having said all that, let us start out by making a **tar** system save to floppy disk.

tar save procedure:

For systems with at least one high-capacity drive:

1) If the system is up and running, shut it down with either the

   shutdown Ø <ENTER>

   or

   sync;sync;/etc/haltsys <ENTER>

   command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

   Type CONTROL-d to proceed with normal startup, (or give root password for system maintenance):

   enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do backups, it is best to do so -- the process will go a lot faster and files are less likely to change while in the process of being saved.

2) Format as many diskettes as you will need for the backup before you start. Otherwise, you will have to stop in the middle of the procedure and start again from scratch, which is a drag!

3) When the root prompt "#" returns, type:

   mount /dev/usr /usr <ENTER>

   This save procedure is only going to save the basic root and usr filesystems. If your customer has additional file systems, you will probably need to check with him to determine how they are mounted so that you will be able to save them, too.

4) Insert a formatted high-density diskette into floppy drive Ø, and at the root prompt "#", type:

   tar -cvfbk /dev/rfdØ 16 1224 `ls | grep -v '^dev$'` <ENTER>

   I know that this is a long and complicated command line, but it must be typed in exactly this fashion to work correctly!

5) The system will start accessing the floppy drive. When you are instructed to, mount new volumes as necessary. Make sure you label the volumes sequentially! Although it is less of a problem with a **tar**-type save than with a **sysadmin** save, it is still a good habit to get into.

6) When the save is finished, the root prompt "#" will return. To unmount the **usr** filesystem, prior to bringing the system up in a multi-user state or shutting it down, type:

umount /dev/usr <ENTER>

7) At the root prompt, type **<CTL><D>** to log out, or, if you wish to shut the system down, type:

sync;sync;/etc/haltsys <ENTER>.

For a system with one or more cartridge drives:

1) If the system is up and running, shut it down with either the

shutdown 0 <ENTER>

or

sync;sync;/etc/haltsys <ENTER>

command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

Type CONTROL-d to proceed with normal startup, (or give root password for system maintenance):

enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do backups, it is best to do so -- the process will go a lot faster and files are less likely to change while in the process of being saved.

2) Format as many disk cartridges as you will need for the backup **before** you start. Otherwise, you will have to stop in the middle of the procedure and start again from scratch.

3) When the root prompt "#" returns, type:

mount /dev/usr /usr <ENTER>

This save procedure is only going to save the basic **root** and **usr** filesystems. If your customer has additional file systems, you will probably need to check with them to determine how they are mounted so that you will be able to save them, too.

4) Install a formatted cartridge in cartridge drive Ø. At the root prompt "#", type the following:

tar -cvfk /dev/cdØ 1Ø24Ø `ls | grep -v '^dev$'` <ENTER> (1Ø meg)
tar -cvfk /dev/cdØ 2Ø896 `ls | grep -v '^dev$'` <ENTER> (2Ø meg)

Make sure your fingers don't get tangled typing this, because it must be typed in exactly this fashion to work correctly!

5) The system will start accessing the cartridge drive. When you are instructed to, mount new volumes as necessary. Make sure you label the volumes sequentially! Although it is less of a problem with a **tar**-type save than with a **sysadmin** save, it is still a good habit to get into.

6) When the save is finished, the root prompt "#" will return. To unmount the **usr** filesystem, prior to bringing the system up in a multi-user state or shutting it down, type:

umount /dev/usr <ENTER>

7) At the root prompt, type **<CTL><D>** to log out, or, if you wish to shut the system down, type:

sync;sync;/etc/haltsys <ENTER>.

For systems with a streaming tape drive:

1) If the system is up and running, shut it down with either the

   shutdown 0 <ENTER>

   or

   sync;sync;/etc/haltsys <ENTER>

   command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

   Type CONTROL-d to proceed with normal startup, (or
   give root password for system maintenance):

   enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do backups, it is best to do so -- the process will go a lot faster and files are less likely to change while in the process of being saved.

2) When the root prompt "#" returns, type:

   mount /dev/usr /usr <ENTER>

   This save procedure is only going to save the basic **root** and **usr** filesystems. If your customer has additional file systems, you will probably need to check with them to determine how they are mounted so that you will be able to save them, too.

3) Load a tape into the drive, and make certain the drive is ready for use. At the root prompt "#", type:

   rewind <ENTER>

   This will rewind the tape to assure it is ready for the backup.

4) Type:

   wait <ENTER>

   This will assure we are ready to do the save.

5) When the root prompt returns, type:

tars -c / <ENTER>

This will save the entire contents of the root file system, and any other file systems which are currently mounted on the root file system (like /dev/usr!). (Incidentally, **tars** is the correct name of the archive utility for streaming tape... if you type "tar" instead, it will not work!)

6) When the save is finished, the root prompt "**#**" will return. To unmount the **usr** filesystem, prior to bringing the system up in a multi-user state or shutting it down, type:

umount /dev/usr <ENTER>

7) At the root prompt, type **<CTL><D>** to log out, or, if you wish to shut the system down, type:

sync;sync;/etc/haltsys <ENTER>.

For systems with a low-capacity drive:

1) If the system is up and running, shut it down with either the

shutdown 0 <ENTER>

or

sync;sync;/etc/haltsys <ENTER>

command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

Type CONTROL-d to proceed with normal startup, (or give root password for system maintenance):

enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do backups, it is best to do so -- the process will go a lot faster and files are less likely to change while in the process of being saved.

2) Format as many diskettes as you will need for the backup **before** you start. Otherwise, you will have to stop in the middle of the procedure and start again from scratch.

3) When the root prompt "#" returns, type:

mount /dev/usr /usr <ENTER>

This save procedure is only going to save the basic **root** and **usr** filesystems. If your customer has additional file systems, you will probably need to check with them to determine how they are mounted so that you will be able to save them, too.

4) Insert a formatted low-density diskette into floppy drive Ø, and at the root prompt "#", type:

tar -cvfbk /dev/rfdØ 16 36Ø `ls | grep -v '^dev$'` <ENTER>

I know that this is a long and complicated command line, but it must be typed in exactly this fashion to work correctly!

5) The system will start accessing the floppy drive. When you are instructed to, mount new volumes as necessary. Make sure you label the volumes sequentially! Althgouth it is less of a problem with a **tar**-type save than with a **sysadmin** save, it is still a good habit to get into.

6) When the save is finished, the root prompt "#" will return. To unmount the **usr** filesystem, prior to bringing the system up in a multi-user state or shutting it down, type:

umount /dev/usr <ENTER>

7) At the root prompt, type **<CTL><D>** to log out, or, if you wish to shut the system down, type:

sync;sync;/etc/haltsys <ENTER>.

These methods of backup will probably see you through most imaginable situations. Before we forge ahead into how to get this data back on the drive, however, I'm going to tell you a bit about how to do a **sysadmin** save. I told you earlier that with the first release of System 5 that **tar** was a better way to go... but that may not be true with subsequent releases. In any case, this is even easier than the ones we just did!

Again, a cautionary note: **sysadmin**, under any circumstances, should only be used for system backups when you have some method of restoring your data other than the primary floppy drive. This "other method" can be a streaming tape, a second floppy drive, or a cartridge drive. **If your only method of backup is one floppy drive, then sysadmin is not for you!**

/etc/sysadmin save procedure:

For systems with two floppy drives:

1) If the system is up and running, shut it down with either the

   shutdown ∅ <ENTER>

   or

   sync;sync;/etc/haltsys <ENTER>

   command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

   Type CONTROL-d to proceed with normal startup, (or
   give root password for system maintenance):

   enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do backups, it is best to do so -- the process will go a lot faster and files are less likely to change while in the process of being saved.

2) Format as many diskettes as you will need for the backup **before** you start. Otherwise, you will have to stop in the middle of the procedure and start again from scratch, which is a drag!

3) When you are done formatting diskettes, type:

   /etc/sysadmin <ENTER>

   The screen will say:

   File System Maintenance

```
Type    1  to do daily partial backup (root file system)
        2  to do daily partial backup (usr file system)
        3  to do a periodic full backup (root file system)
        4  to do a periodic full backup (usr file system)
        5  to get a backup listing
        6  to restore file(s)
        7  to do daily partial backup (other file system)
        8  to do periodic full backup (other file system)
        q  to quit

Enter Number:
```

4) We will be doing a full backup of the root and usr file systems. If your customer has other file systems as well, you will probably need to consult with them as to what will need to be additionally backed up. We will start with the root filesystem, so type:

    3 <ENTER>

The screen will say:

```
Write backup onto floppy (f), disk cartridge (d), or tape
cartridge (t)  ?
Enter choice (f, d, or t) :
```

We are doing a floppy backup, so enter:

    f <ENTER>

5) The next prompt will be:

```
Enter floppy drive number (Ø-1):
```

Choose the appropriate drive number and press <ENTER>. The system responds:

```
Insert first floppy disk in drive x, them press <RETURN>:
```

where x is the drive number you specified.

6) Insert the first high-capacity formatted disk in the drive you chose to use, and press <ENTER>. The system will start to access the floppy drive, and the screen should say something similiar to:

```
PERIODIC BACKUP - /dev/root filesystem
64+Ø records in
64+Ø records out
       date = Tue Jul 22 13:42:5Ø 1986
backup date = the epoch
backing up /dev/root to /dev/rfdØ96ds15
I
II
estimated xxxxK on x volume(s)
III
IV
Please insert new volume, then press ENTER:
```

where the xxxxK and x volumes(s) will vary with system configuration.

.....

Note: **sysadmin** on floppy is only set
up to use **high-capacity** diskettes.

This goes for both Drive Ø and Drive 1.

.....

7) As the system prompts you for additional diskettes, remove the last diskette used, insert the next, and press **<ENTER>** to continue. **Make certain that you label the diskettes in order!** Getting them out of order renders them virtually useless. When the backup has been completed, the screen will say something to the effect of:

```
level Ø backup on Tue Jul 22 13:42:5Ø 1986
DONE
4919K on 5 volumes(s)
Press <RETURN> to continue:
```

8) When you press **<ENTER>**, the main menu will reappear. Next, we will want to backup up the usr filesystem, so type:

4 <ENTER>

9) Follow the directions in steps (4)-(7) to complete the backup of the usr filesystem. When the main menu re-appears, type **<q><ENTER>** to exit **sysadmin**.

1Ø) At this point, you may type **<CTL><D>** to go multi-user or

sync;sync;/etc/haltsys <ENTER>

to shut the system down.

The **/etc/sysadmin** backup procedures for **cartridge** and **streaming tape** media are very similar to this we've just gone through, so I won't repeat myself. Essentially, the difference is in the choice of media, and if you get that right, and choose a drive which is in the system, you won't go far wrong.

.....

"Now that I've reformatted the drive,
how do I restore the system?"

or

Step 2 in the "save and restore" process.

.....

To put the data that you've saved back on the drive, go with the utility that you used to make the save. For example, your **tar** save disks won't be restorable using **sysadmin**, and vice-versa. On that note, turn to the next groups of instructions, match you system configuration, and find out how to restore your data.

.....

Relax....

If you've make it this far,
the rest is easy.

.....

tar restores:

For floppy system restores:

1) The first thing you will have to do is reinitialize and reinstall Xenix on this system. We went over that in Chapter 1, so I'm not going to belabor that further here. You should be certain that you are re-constructing the system as the customer had it originally! If you don't, the files may not go back to the correct locations.

   Once you have accomplished this, proceed to the next steps.

2) If the system is up and running, shut it down with either the

shutdown 0 <ENTER>

or

sync;sync;/etc/haltsys <ENTER>

command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

Type CONTROL-d to proceed with normal startup, (or
give root password for system maintenance):

enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do restores, it is best to do so -- the process will go a lot faster.

3) At the root prompt, type:

mount /dev/usr /usr <ENTER>

This is to ensure that all the /usr files will wind up in their correct locations.

4) When the root prompt returns, put the first disk of the save into the drive in which the save was performed, and type:

while : <ENTER>
do <ENTER>
tar -xvf /dev/fdx <ENTER>
echo -n "Next disk? " <ENTER>
read junk <ENTER>
done (wait to press **<ENTER>** here until you insert the disk!)

where x is the drive where the save was made. Make sure that there is a disk in the drive before you press the <ENTER> key after you type "done"; otherwise, Xenix will likely make nasty noises at you about "FLOPPY DRIVE not ready" or something like that. Nothing fatal, mind you, but awfully embarrassing if the customer happens to be watching!

What we have done here is write a little shell procedure which will prompt you with:

Next disk?

when it is finished with the current one. Insert the next disk in the series, and press **<ENTER>** to continue. Continue this process until all the disks have been read in.

5) When all the disks in the save have been restored to the system, press **<BREAK>** (or **<DELETE>**) to abort the shell procedure and return to the root prompt.

6) Type:

umount /dev/usr <ENTER>

to unmount the usr partition.

7) At this point, you may type **<CTL><D>** to go multi-user or

sync;sync;/etc/haltsys <ENTER>

to shut the system down.

For cartridge system restores:

1) The first thing you will have to do is reinitialize and reinstall Xenix on this system. We went over that in Chapter 1, so I'm not going to belabor that further here. You should be certain that you are re-constructing the system as the customer had it originally! If you don't, the files may not go back to the correct locations.

Once you have accomplished this, proceed to the next steps.

2) If the system is up and running, shut it down with either the

    shutdown 0 <ENTER>

    or

    sync;sync;/etc/haltsys <ENTER>

    command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

    Type CONTROL-d to proceed with normal startup, (or give root password for system maintenance):

    enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do restores, it is best to do so -- the process will go a lot faster.

3) At the root prompt, type:

    mount /dev/usr /usr <ENTER>

    This is to ensure that all the /usr files will wind up in their correct locations.

4) Insert the cartridge into the drive of your choice, and type:

    tar -xvf /dev/cdx <ENTER>

    where x is the drive you chose.

5) If there is more than one cartridge in the save set, repeat step (4) until all of the cartridges have been restored.

6) When all the cartridges have been restored, type:

    umount /dev/usr <ENTER>

    to unmount the usr partition.

7) At this point, you may type **<CTL><D>** to go multi-user or

    sync;sync;/etc/haltsys <ENTER>

    to shut the system down.

For streaming tape restores:

1) The first thing you will have to do is reinitialize and reinstall Xenix on this system. We went over that in Chapter 1, so I'm not going to belabor that further here. You should be certain that you are re-constructing the system as the customer had it originally! If you don't, the files may not go back to the correct locations.

   Once you have accomplished this, proceed to the next steps.

2) If the system is up and running, shut it down with either the

   shutdown 0 <ENTER>

   or

   sync;sync;/etc/haltsys <ENTER>

   command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

   Type CONTROL-d to proceed with normal startup, (or give root password for system maintenance):

   enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do restores, it is best to do so -- the process will go a lot faster.

3) At the root prompt, type:

   mount /dev/usr /usr <ENTER>

   This is to ensure that all the /usr files will wind up in their correct locations.

4) Insert the streaming tape cartridge into the tape drive and type:

   rewind <ENTER>

   followed by

   wait <ENTER>

   This will ensure that the tape has been properly positioned to restore the data.

5) When the root prompt returns, type:

   tars -x <ENTER>

   Then you can sit back and wait for the restore to finish. If there is more than one tape in your restore set, repeat steps (4) and (5) until all the tapes have been restored.

6) When finished, remove the last tape from the drive and type:

   umount /dev/usr <ENTER>

   to unmount the usr partition.

7) At this point, you may type **<CTL><D>** to go multi-user or

   sync;sync;/etc/haltsys <ENTER>

   to shut the system down.

Well, that covers the basics of saves and restores using **tar**. The next thing I'll talk about is how to do a **sysadmin** type restore by booting off of the hard drive. You should only use this sort of save with the first release of System 5 as a last resort.

/etc/sysadmin restores:

1) The first thing you will have to do is reinitialize and reinstall Xenix on this system. We went over that in Chapter 1, so I'm not going to belabor that further here. You should be certain that you are re-constructing the system as the customer had it originally! If you don't, the files may not go back to the correct locations.

   Once you have accomplished this, proceed to the next steps.

2) If the system is up and running, shut it down with either the

shutdown Ø <ENTER>

or

sync;sync;/etc/haltsys <ENTER>

command. Reboot the system, and enter System Maintenance Mode. To do this, when the screen says:

```
Type CONTROL-d to proceed with normal startup, (or
give root password for system maintenance):
```

enter the root password. This will put you into single-user (aka System Maintenance) mode. Whereas you don't exactly have to put the system into single user mode to do restores, it is best to do so -- the process will go a lot faster.

3) When the root prompt appears, type:

/etc/sysadmin <ENTER>

The File System Maintenance Menu will appear. Choose option (6) to restore files. The system will say:

```
RESTORE FILE(S)
Restore from floppy (f), disk cartridge (d), or tape
cartridge (t) ?
Enter choice (f, d, or t) :
```

Choose the type of media you used, and press **<ENTER>** after you type it.

4) The system will next ask you, where appropriate, which drive you wish to use. Give it your response, and press **<ENTER>**.

5) The next message displayed will be:

    Type Full Pathname of File or Directory to Restore.

   The first restore we will do is that of the root filesystem, so type:

    / <ENTER>

   The system will respond:

    Restoring . to /

    Insert first media type in drive x, then press <RETURN>:

   Insert the first volume of your root filesystem save set into the indicated drive. Follow the directions until the restore has been completed for the filesystem. This will in all probability take a very long time!

6) Eventually, the main menu for **sysadmin** should reappear. Again, choose option (6) to restore files, and answer the questions about which type of media and which drive are to be used. When prompted for the destination pathname, type:

    /usr <ENTER>

   When instructed to do so, insert the first volume of the usr filesystem save into the indicated drive. Follow the directions until the restore has been completed.

7) When the main menu reappears, you may either restore any additional file systems that the customer may have backups for, or press q **<ENTER>** to quit and return to the root prompt.

8) At this point, you may type **<CTL><D>** to go multi-user or

    sync;sync;/etc/haltsys <ENTER>

   to shut the system down.

****

Special Device Files

****

One of the things which can frequently be confusing about Xenix is the way in which it refers to its peripherals. Not only does it support a lot of them, but it usually has more than one way to address them! In System V on the Tandy 3000, things are much more confusing than they were with the older Xenix releases for the 6000, primarily because (a) there are so many types of floppy formats available and (b) disk partitioning is supported.

To relieve some of this confusion, and to provide some helpful information, I summarize the special device files which you may find on a System V Xenix system here. **Note: Depending on how the system is configured, you may or may not find all the files I list here.** Since System V allows the user to reconfigure the kernal, there is a certain flexibility in the drivers which will be available on any given system.

Floppy Drives:

Drive 0/Drive A:

Device Name	Format
/dev/fd048	9 sectors, double sided
/dev/fd048ss8	8 sectors, single sided
/dev/fd048ds8	8 sectors, double sided
/dev/fd048ss9	9 sectors, single sided
/dev/fd048ds9	9 sectors, double sided
/dev/f096ds15	15 sectors, double sided
/dev/fd0	15 sectors, double sided
/dev/fd096	15 sectors, double sided

Drive 1/Drive B:

Device Name	Format
/dev/fd148	9 sectors, double sided
/dev/fd148ss8	8 sectors, single sided
/dev/fd148ds8	8 sectors, double sided
/dev/fd148ss9	9 sectors, single sided
/dev/fd148ds9	9 sectors, double sided
/dev/fd196ds15	15 sectors, double sided
/dev/fd196	15 sectors, double sided

Hard Drives:

Hard Drive Ø:

Device Name	Format
/dev/root	User-defined system file system
/dev/usr	User-defined user file system
/dev/rhdØØ	Raw device (entire disk)
/dev/hdØØ	Block device (entire disk)
/dev/rhdØ1	Raw partition 1
/dev/hdØ1	Partition 1
/dev/rhdØ2	Raw partition 2
/dev/hdØ2	Partition 2
/dev/rhdØ3	Raw partition 3
/dev/hdØ3	Partition 3
/dev/rhdØ4	Raw partition 4
/dev/hdØ4	Partition 4
/dev/hdØa	Active partition
/dev/rhdØa	Raw active partition
/dev/hdØd	DOS partition
/dev/rhdØd	Raw DOS partition

Hard Drive 1:

Device Name	Format
/dev/rhd1Ø	Raw device (entire disk)
/dev/hd1Ø	Block device (entire disk)
/dev/rhd11	Raw partition 1
/dev/hd11	Partition 1
/dev/rhd12	Raw partition 2
/dev/hd12	Partition 2
/dev/rhd13	Raw partition 3
/dev/hd13	Partition 3
/dev/rhd14	Raw partition 4
/dev/hd14	Partition 4
/dev/hd1a	Active partition
/dev/rhd1a	Raw active partition
/dev/hd1d	DOS partition
/dev/rhd1d	Raw DOS partition

Note: Use **/dev/usr** and **/dev/root** whenever possible. The raw device names are used with the **badtrack** and **fdisk** commands.

Tape Drive:

Device Name	Format
/dev/mtØ	Block device, rewinds after use
/dev/cmtØ	Continuous block device
/dev/rmtØ	Raw device, rewinds after use
/dev/rcmtØ	Continuous raw device

Cartridge Drives (1Ø and 2Ø meg external):

Cartridge Drive Ø:

Device Name	Format
/dev/cdØ	Block device
/dev/rcdØ	Raw device
/dev/cdbtØ	Boot track (block)
/dev/rcdbtØ	Raw device boot track

Cartridge Drive 1:

Device Name	Format
/dev/cd1	Block device
/dev/rcd1	Raw device
/dev/cdbt1	Boot track (block)
/dev/rcdbt1	Raw device boot track

Block Sizes:

The block size of a disk is the number of blocks of storage space available on the disk. (A block is typically 1Ø24 bytes of storage). However, some commands, such as **tar**, use half-block sizes (512 bytes). Some commands require you to enter blocks or display sizes as half-blocks.

The following table lists the half-block sizes of floppy diskettes:

Diskette Format	Half-Block Size
8 sectors, single sided	16Ø
8 sectors, double sided	32Ø
9 sectors, single sided	18Ø
9 sectors, double sided	36Ø
15 sectors, double sided	12ØØ

A standard 2Ø megabyte fixed disk has 2Ø,ØØØ full blocks divided into up to four partitions. The number and size of each partition depend on the configuration set during installation.

Gap and Block Numbers:

The **mkfs** command uses gap and block numbers to describe how the blocks are to be arranged on a disk. The following table lists the gap and block numbers for the floppy diskettes and fixed disk used with the 3ØØØ:

Device	Gap	Blocks
Diskettes		
9 sectors per track	3	18
15 sectors per track	3	3Ø
Hard Disk	1	34

****

FILE GUIDE

****

In this appendix, we'll take a cursory look at frequently used files and directories on the Xenix system. There is a pretty fair number of things which cannot be removed if Xenix is to run correctly, and I'll try to point most of them out as I go.

Directory: **root (/)**

Purpose: Acts as the "root" directory of the "tree". Contains many subdirectories and files which cannot be deleted if the system is to operate properly -- they are:

/bin	Xenix commands directory
/dev	device special directory
/etc	additional program and data file directory
/lost+found	storage area for orphaned files recovered by the file system cleaning program **fsck**
/mnt	mount directory (reserved for mounted file systems)
/usr	user home directories
/tmp	temporary directory (reserved for temporary files created by programs)
/lib	C program library directory
/xenix	Xenix system object code

Directory: **/bin**

Purpose: Contains the commands which are most frequently used. All commands in this directory are required for correct system operation and should not be removed. Some things you would find are:

basename	echo	passwd	su
cp	expr	rm	sync
date	fsck	sh	tar
backup	login	sleep	restore
dumpdir	mv	stty	test

Directory: **/dev**

Purpose: Contains special files that control access to peripheral devices, such as disks, printers, and terminals. All files in this directory are required and must not be removed.

/dev/console	system console
/dev/lp	line printer
/dev/mem	physical memory
/dev/null	null device (the bit bucket)
/dev/fdx	floppy disk drives
/dev/hdxx	hard disk drives and their partitions
/dev/mtx	tape drives (currently only 1 supported)
/dev/rxx	unbuffered interface to corresponding device name
/dev/root	root file structure
/dev/swap	swap area
/dev/usr	user file structure
/dev/ttyx	terminals
/dev/tty	the terminal you are using
/dev/cdx	cartridge drives

Directory: **/etc**

Purpose: Contains miscellaneous system program and data files. All files are required, but you may modify some of them.

The following files must not be removed or modified:

/etc/mtab	mounted device table
/etc/mount	for mounting a file structure
/etc/mkfs	for creating a file structure
/etc/init	first process after boot

The following files may be modified but not removed:

/etc/passwd	password file
/etc/rc	bootup shell script (actually, you should pur local mods in /etc/rc.user)
/etc/ttys	terminal setup
/etc/termcap	terminal capability map
/etc/ttytype	terminal types on specified ports
/etc/gettydefs	contains baud rate and terminal definitions for **getty**
/etc/motd	message of the day
/etc/profile	contains commands executed by **login** for everyone

The following directory contains files that may be modified but not removed:

/etc/default	defaults for various programs

Directory: **/lib**

Purpose: Contains runtime library files for C and other language programs. The directory is required and must not be removed.

Directory: **/mnt**

Purpose: This is an empty directory reserved for mounting removable file systems.

Directory: **/tmp**

Purpose: Contains temporary files created by running programs. Normally, the files are present while the program is running. If the program terminates abnormally or prematurely, these files may be left in /tmp. You may remove any temporary file that does not belong to a running program.

Directory: **/usr**

Purpose: Contains the home directories of all users on the system. It also contains several other directories that provide additional Xenix commands and data files.

/usr/bin	contains more Xenix commands, which are less frequently used or are not essential for system operation
/usr/lib	contains more libraries and data files used by various Xenix commands
/usr/include	contains header files for compiling C programs
/usr/spool	contains directories for storing files to be printed, mailed, or passed through networks

/usr/tmp	more temporary files. You may remove any temp file which does not belong to a running program.
/usr/adm	contains data files associated with system administration and accounting.

File: **/etc/gettydefs**

Purpose: To designate baud rates and terminal characteristics

Typical Changes: Change baud rate, parity, and other terminal characteristics
Give next mode to try if user indicates need

Sample File:

```
C# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY #Console login: #C

7# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY TAB3 #login: #7

6# B9600 PARENB CS7 OPOST ONLCR # B9600 SANE IXANY #login: #6

5# B4800 PARENB CS7 OPOST ONLCR # B4800 SANE IXANY #login: #5

4# B2400 PARENB CS7 OPOST ONLCR # B2400 SANE IXANY #login: #4

3# B1200 CS8 OPOST ONLCR # B1200 SANE IXANY #login: #2

2# B300 CS7 OPOST ONLCR # B300 SANE IXANY #login: #3

1# B300 CS8 OPOST ONLCR CR1# B1200 SANE IXANY #login: #1
|                |                        |         |     |
a                b                        c         d     e
```

a — Line label which identifies the /etc/gettydefs entry to **getty**. This could be a letter or number. The label corresponds to the line mode field in **/etc/ttys**. **Init** passes the line mode as an argument to **getty**.

b — Initial flags which set the initial **ioctl** settings if a terminal type is not specified to **getty**. Normally only the speed flag ("B300" = 300 baud) is required in the initial flags. **Getty** automatically sets the terminal to raw input mode and takes care of most of the other flags. The initial flag settings remain in effect until **getty** executes **login**.

c Final flags which are set just prior to the execution of **login** by **getty**. The speed flag is again required. The composite flag SANE takes care of most of the other flags that need to be set to ensure that the terminal and the processor are communicating in a rational fashion.

d Login prompt

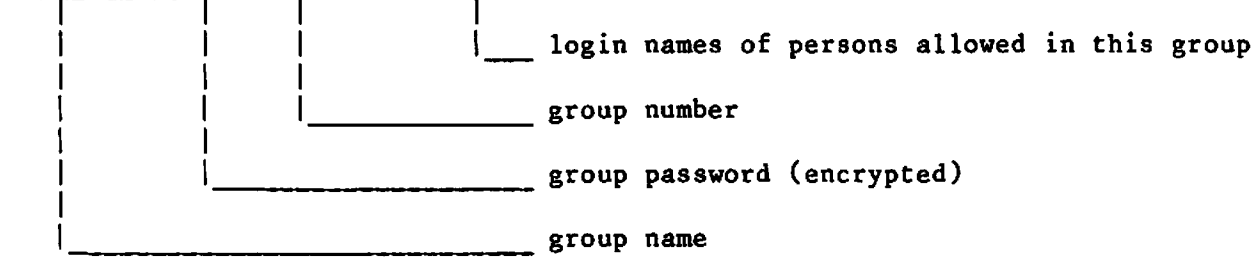
e Identifies the next label in /etc/gettydefs for **getty** to try if the current one is not successful. Groups of entries should form a closed set so that **getty** cycles back to the original entry if none of the entries is successful. In the above example, the entries labelled 2 and 3 form a closed 12ØØ baud - 3ØØ baud - 12ØØ baud set.

File: **/etc/group**

Purpose: To designate group 'names'

Sample file:

```
root:x:Ø:root
cron:x:1:cron
bin:x:3:bin,lp
uucp:x:4:uucp,uucpmgr
asg:x:6:asg
sysinfo:x:1Ø:uucp
network:x:12:network
group::5Ø:dennis,susan,james
payroll:sV23xdf:53:mark,tom,dave,fred
```



File: **/etc/logbook**

Purpose: To show what Tandy programs and updates have been installed on a customer's system

Typical changes: None. Do NOT change this file unless absolutely necessary.

Sample File:

```
Ø1.ØØ.ØØ  25-42Ø1  Timesharing System            Wed Apr  2 14:39:24 CST 1986
Ø1.ØØ.ØØ  25-42Ø2  Software Development System   Wed Apr  2 14:39:24 CST 1986
   |                      |                            |
   |                      |                            |__Date/time of
   |                      |                               installation
   |                      |
   |                      |___Product name and
   |                          catalog number
   |
   |___Product version
       number
```

File: **/etc/passwd**

Purpose: To define system users, their passwords, group membership, home directory and execution program.

Typical Changes: Change group number as necessary
Change password to nothing
Correct spelling errors in login name
Rename user directory
Redefine execution program

Sample File:

```
root:vFStfaD5tUmYD:Ø:Ø:The Super User:/:/bin/sh
cron:NOLOGIN:1:1:Cron Daemon for periodic tasks:/:
bin:NOLOGIN:3:3:The owner of system files:/:
uucp::4:4:Account for uucp program:/usr/spool/uucppublic:/usr/lib/uucp/uucico
uucpmgr::4:4:Account for uucp manager:/usr/lib/uucp:
asg:NOLOGIN:6:6:The Owner of Assignable Devices:/:
sysinfo:NOLOGIN:1Ø:1Ø:Access to System Information:/:
network:NOLOGIN:12:12:Account for mail program:/usr/spool/micnet:
lp:NOLOGIN:14:3:The lp Administrator:/usr/spool/lp:
bob::2Ø1:5Ø:bob davis:/usr/bob:/bin/sh
tom::2Ø2:5Ø:tom smythe:/usr/tom:/bin/sh
rick:k34j33nhz2:2Ø3:5Ø:roger hays:/usr/rick:runap
```

```
  a      b        c  d      e         f       g     h
```

a	login name
b	password if any (encrypted)
c	user id number
d	group number
e	full name (general comment field)
f	home directory
g	restrict access to one program
h	allow access to the command interpreter (shell)

File: **/etc/rc, /etc/rc.user**

Purpose: Execution files on system startup.

Typical Changes: Usually, items specific to the system (like required "mounts" of secondary file systems) are placed in /etc/rc.user, as are any modifications pertinent to applications. If you want to change the boot message, you might change /etc/rc.

Sample Files:

/etc/rc

```
autoboot=""
lflg=""
bootlog=""

while : ; do
        case $# in
        0)
                break
                ;;
        *)
                ;;
        esac
        case "$1" in
        -a)
                autoboot="TRUE"
                bootlog="/etc/bootlog"
                ;;
        -l)
                lflg="TRUE"
                ;;
        *)
                break
                ;;
        esac
        shift
done

PATH=/etc/:/bin:/usr/bin
echo >/dev/console 2>&1
/etc/umount /dev/usr
. /etc/profile
#
HZ=50; export HZ
#
```

```
# firsttime will use 'ed' to edit the next line to correct the timezone
# it will key on the token ---> FOOBAR <--- so don't delete this line.
TZ=CST6CDT
export TZ

cp /dev/null /etc/mnttab; chmod 644 /etc/mnttab
cp /dev/null /etc/utmp; chmod 644 /etc/utmp

if [ -c /dev/clock -a -x /etc/setclock ] ; then
        echo "Current System Time is: \c"
        date `/etc/setclock`
} 2>&1 | tee /dev/console >>${bootlog:-"/dev/null"}

elif [ "A$autoboot" != "ATRUE" ] ; then
        echo
        /etc/asktime </dev/console >/dev/console 2>&1
fi

echo 'root /' | setmnt

/etc/mount /dev/usr /usr
if test $? = 2; then
        echo "Cleaning /dev/usr:
        fsck -y -t /tmp/fdsk$$ /dev/usr
        /etc/mount /dev/usr /usr
fi 2>&1 | tee /dev/console >> ${bootlog:-"/dev/null"}

/etc/update

cp /dev/null /usr/adm/msgbuf
/etc/dmesg - >> /usr/adm/messages

/usr/lib/xmail_recover
/usr/lib/ex3.7preserve -

rm -f /usr/spool/lp/SCHEDLOCK; /usr/lib/lpsched
rm -r /usr/spool/lpd/*
rm -f /usr/spool/uucp/LCK..*

rm -f /tmp/* /usr/tmp/*

/etc/cron

mv /etc/wtmp /etc/owtmp
cp /dev/null /etc/wtmp; chmod 644 /etc/wtmp

if [ z"$1flg" = z"TRUE" ] ; then
        mv /xenix /xenix-
```

```
        mail -s "New /xenix Installed" root <<!

The IDD xenix you are running is more recent than the /xenix that
was present prior to this reboot.  A new /xenix has been configured by
/etc/rc, using your IDDs.  The old /xenix has been moved to /xenix-.

!
        /etc/xconf -o /xenix -c /lib/sys/config.sys \
                hd /lib/sys/xenix > /dev/console 2>&1 _____________________ h
fi

if [ -f /etc/logfile ]
then
        /bin/sh /etc/logfile
fi

if [ -f /etc/rc.user ] ____________________________________________________ i
then
        /bin/sh /etc/rc.user | tee /dev/console >> ${bootlog:-"/dev/null"}
fi

if [ "A$autoboot" = "ATRUE" ]
then
        mail -s "Automatic Boot Log" root < $bootlog
fi

rm -f $bootlog
```

/etc/rc.user

```
/etc/mount /dev/usr2 /usr2          _________
/etc/mount /dev/games /usr/games    _________|_________________________ j
```

a	parse command line options
b	set search path
c	time prompt
d	mount and clean if necessary /dev/usr
e	start addtional background processes
f	remove printer lock; start printer daemon
g	clear out temporary directories
h	reconfigure kernel (if requested)
i	execute instructions in /etc/rc.user if it exists
j	mount additional file systems

File: **/etc/systemid**

Purpose: To define system name during UUCP (Unix to Unix Communications Programs) and Micnet operations.

Typical Changes: During network operations each machine musr have a unique name.

Sample File:

systemv

File: **/etc/termcap**

Purpose: To define standard terminals and their control codes

Typical Changes: Add new terminal types

Sample File: **cat** the file **/etc/termcap** on your Xenix system.

File: **/etc/ttys**

Purpose: To define port status, characteristics, and name

Typical Changes: Change default terminal baud rate

Sample File:

```
1Cconsole
16tty00
05tty01
15tty02
05tty03
05tty04
05tty05
05tty06
05tty07
05tty08
05tty09
||  |
||  |_________port name
||____________baud rate
|_____________status (set by enable/disable command)
                 (0 = disabled, 1 = enabled)
```

Note: See **/etc/gettydefs** for baud rate information

File: **/etc/ttytype**

Purpose: To define default terminal type

Typical Changes: If a customer has other terminal types that he will be using exclusively

Sample File:

```
ansi console
dumb tty01
adds25 tty02
dtl00 tty03
dumb tty04
dumb tty05
dumb tty06
dumb tty07
dumb tty08
dumb tty09
  |    |
  |    |_________port name
  |______________default terminal type
```

File: **/etc/motd**

Purpose: Message for users upon login

Typical Changes: To notify users of important information
System to be down for repair
Reports due today
Posting will be done at 7:00 pm

Sample File:

```
  ****************************************
 **                                      **
**         Xenix System V/286           **
 **                                      **
  ****************************************
```

File: **/etc/default/backup**

Purpose: Default drive number and directory for "dumps"

Typical Changes: Change if drive Ø fails and you need to do a "dump"

Sample File:

```
tape=/dev/rfdØ
disk=/dev/rroot
```

File: **/etc/default/cron**

Purpose: To enable or disable cron loggings in /usr/lib/cronlog.

Typical Changes: Change to "CRONLOG=YES" if cron logging is to be enabled; this has the capacity to eat disk space in a profound manner so if it is to be used, free disk space must be closely monitored and the file truncated regularly.

Sample File:

```
CRONLOG=NO
```

File: **/etc/default/dos**

Purpose: To define abbreviations for special device files used in accessing MS-DOS files and directories.

Typical Changes: Change the styles of floppy disks which are defined.

Sample File:

```
A=/dev/fdØ48ds9
B=/dev/fd148ds9
C=/dev/hdØd
D=/dev/hd1d
X=/dev/fdØ96ds15
Y=/dev/fd196ds15
```

File: **/etc/default/dumpdir**

Purpose: To specify the default **dumpdir** device

Typical Changes: Change default device in case of failure of primary device

Sample File:

```
tape=/dev/rfdØ
```

File: **/etc/default/localprint**

Purpose: To specify default environment variables for local printing

Typical Changes: Change to match terminal type used on the system
Change forms length

Sample File:

```
TERM=dtlØØ
FORMS=
STRICT=YES
```

File: **/etc/default/login**

Purpose: To specify default environment variables for **login**

Typical Changes: Change upper limit on process file size

Sample File:

```
TIMEZONE=CST6CDT
ULIMIT=32767
```

File: **/etc/default/lpd**

Purpose: Contains banners on/off information

Typical Changes: Turn banners off

Sample File:

BANNERS=1 _____ (1=on, Ø=off)

File: **/etc/default/micnet**

Purpose: Command file for Micnet system

Typical Changes: Restrict or increase remote command privileges, change default search path

Sample File:

```
executeall
execpath=PATH=/bin:/usr/bin:/etc
```

File: **/etc/default/mkuser**

Purpose: Used by **mkuser** to define root directory of user's directory path

Typical Changes: Relocate position of user directory (for example, if all user directories were on a secondary drive partition, you might specify "/usr2" instead of "/usr").

Sample File:

```
HOME=/usr
```

File: **/etc/default/passwd**

Purpose: Contains variables used to invoke password aging and minimum length

Typical Changes: Make minimum length of password a number greater than zero, define the minimum and maximum number of weeks between password changes.

Sample File:

```
MINWEEKS=Ø
MAXWEEKS=999
PASSLENGTH=Ø
```

File: **/etc/default/restor**

Purpose: Contains default device used for **restore**

Typical Changes: Change if original restore default device fails

Sample File:

```
archive=/dev/rfdØ
```

File: **/etc/default/su**

Purpose: Contains information which allows the use of the **su** command to be logged on the console and/or in a log file

Typical Changes: Invoke accounting on **su**

```
#SULOG=/usr/adm/sulog                                    a
#CONSOLE=/dev/console                                    b
```

a — when pound sign (#) removed, invokes **su** accounting with destination file of /usr/adm/sulog. Destination file may be renamed.

b — when pound sign (#) removed, invokes monitoring of all **su** usage by echoing to system console.

Xenix Communications

DT1 Baud Rate Codes:

Note: You must also change the /etc/ttys file to the appropriate code.

Ø Ø Ø 1	75
Ø Ø 1 Ø	11Ø
Ø Ø 1 1	15Ø
Ø 1 Ø Ø	3ØØ
Ø 1 Ø 1	6ØØ
Ø 1 1 Ø	12ØØ
Ø 1 1 1	24ØØ
1 Ø Ø Ø	48ØØ
1 Ø Ø 1	96ØØ

Word Length: 8
Parity: None
Stop Bits: 1

****

XENIX ERRORS

A Compendium of Common and Uncommon
Xenix Error Messages
(And a Brief Explanation of Each)

****

This section describes the various system messages which may appear on the system console. These messages are frequently invaluable for a technician trying to diagnose a problem, so noting the exact wording of the message is crucial in attaching the correct meaning to the message. **This information may not be entirely accurate for all versions of System 5 Xenix**; however, it will probably at least point you in the right direction.

These messages fall into several categories:

Fatal
: Recovery is impossible, and Xenix ceases execution.

System inconsistency
: A contradictory situation (a "this can't happen" situation) exists in the kernel.

Abnormal
: A probably legitimate but extreme situation exists.

Hardware
: Indicates a hardware problem exists.

User error
: The user has caused the problem.

Fatal system messages begin with "panic:" and indicate hardware problems or kernel inconsistencies that are too severe for continued operation. After displaying a fatal message, the system will stop. Rebooting is required.

System Inconsistency messages indicate problems usually traceable to hardware malfunction, such as memory failure. These messages rarely occur since associated hardware problems are generally detected before such an inconsistency can occur.

Abnormal messages represent kernel operation problems, such as the overflow of critical tables. It takes extreme situations to bring these problems about, so they should never occur in normal system use.

User Errors are situations caused by the user trying to do something contradictory, for example, trying to write to a write-protected floppy.

System messages sometimes specify the device, **dev**, that caused the error. Each message gives a device specification of the form **nn/mm** where **nn** is the major number of the device, and **mm** is its minor number. The command:

```
ls -l /dev | grep nn | grep mm  <ENTER>
```

may be used to list the name of the device associated with the given major and

minor numbers.

Device Driver Messages

Device drivers display error messages in five basic formats, two of which are special and are related to specific drivers. Their formats are as follows:

```
device Drive n: specific error message
device Drive n: type error while Reading Cylinder cyl Head hd
        Sector sec  [Status: hhhh]
device Drive n: type error while Writing Cylinder cyl Head hd
        Sector sec  [Status: hhhh]
device drive n: type error while Reading Block blk [Status: hhhh]
device drive n: type error while Writing Block blk [Status: hhhh]
device drive n: type error during cmd from Block blk [Status: hhhh]
```

device Drive n: specific error message

This format is used for general errors from the driver which affect all operations. For example, the message "Cartridge Drive Ø: drive not ready" is displayed when the disk cartridge driver tried some I/O operation and the controller reported the drive busy or not ready.

The following messages are possible from the **disk cartridge driver**, the **hard disk driver**, the **floppy disk driver**, and the **tape driver**:

```
Further I/O aborted until device closed
Device closed; error cleared
Drive not ready
Hardware Failure [Status: hhhh]
Write protected
```

These are displayed **only** by the **disk cartridge driver**:

```
active disk changed
active drive not ready
active drive write protected
bad format
bad sector size
SCSI Bus Hung - Resetting
CMD Phase Violation
```

These are displayed **only** by the **tape driver**:

```
No tape in drive
Drive Hung
```

These messages are displayed **only** by the **hard disk driver**:

Bad signature
Can't read bad block map
Invalid partition table

Finally, this message is displayed **only** by the **floppy driver**:

Time Out

Further I/O aborted until device closed
Device closed; error cleared
These messages are seen when an unrecoverable error has occurred on a device and the driver is refusing to do any more I/O until the device is closed (or until Xenix shuts down). The first usually appears after the error messages describing the unrecoverable error, and the second after the device driver is closed. Hardware or User Error.

Hardware Failure [Status: hhhh]
This message is displayed when the user tries accessing a device, but when the system was booted the driver was unable to talk to its associated hardware. This could mean that the associated hardware was not installed, or that the hardware was not set up properly. (See the driver's manual page or help file for a description of what the status data is). Hardware.

Drive not ready
This indicates that when the drive was accessed, the hardware reported the drive was not ready when it should have been. This can happen, for example, if the drive is powered off when the user tries to access it. Hardware or User Error.

write protected
This error is displayed when a user tries to write to or mount something and the media is write-protected. User error.

active disk changed
This error occurs when a disk is changed while it was mounted. This message is very difficult to cause on cartridge drives inasmuch as the door is locked by Xenix and not released until the drive is unmounted. User error.

active drive not ready
This error occurs when the system tried some operation on a mounted drive, and the controller reported the drive busy or not ready. Hardware or User error.

active drive write protected

This message is seen when the system accesses a previously write-enabled cartridge disk which is mounted, and it is write-protected. This is a very difficult error to cause because of the locking mechanism which is engaged once a cartridge drive is mounted. User error.

bad format

This error occurs when the user tries to access a cartridge which is not formatted or is otherwise not readable. Hardware or User Error.

bad sector size

This is a very unusual error. It indicates that when the system looked at a cartridge to open it for I/O, it had an unusual (ie, non-standard) sector size. This error is unusual because there is only one sector size used on the disk cartridge system. Hardware.

Drive hung

This error indicates that the tape driver was unable to talk to its drive or read its status. Hardware.

No tape in drive

This message indicates that either there is no tape cartridge in the drive, or that the user removed it during drive operation. Hardware or User error.

Bad signature

The indicated fixed drive may not have been initialized properly. The drive should not be used until both **badtrack** and **fdisk** have been used to initialize the drive. Hardware or User error.

Can't read bad block map

An attempt to read bad block information from the specified fixed drive has failed. This may indicated a hardware malfunction. Hardware.

Time Out

An attempt to read from or write to the indicated device has failed. The drive door may be open; the media may be bad; or, in the case of a floppy diskette, the disk may have the wrong density for the requested read or write. Hardware or User error.

SCSI Bus Hung - Resetting

The SCSI bus, which connects the DCS (disk cartridge system) to the computer, was in a non-idle phase when it should have been idle. Bus Reset is asserted and the operation is retried. Hardware error.

CMD Phase Violation
The SCSI bus did not enter or exit the command phase when it was supposed to. Hardware error.

device Drive n: type error while Reading Cylinder cyl Head hd
Sector sec [Status: hhhh]
device Drive n: type error while Writing Cylinder cyl Head hd
Sector sec [Status: hhhh]
device drive n: type error while Reading Block blk [Status: hhhh]
device drive n: type error while Writing Block blk [Status: hhhh]

This format is used for errors which occur during I/O of some sort. The type of error, some hardware-related status, what type of I/O it occurred on (read or write), and where it occurred is displayed. The type of error can be one of the following:

command
hard
no-data
soft
unclassified

A hard error indicates that the driver retried the operation and was unable to complete the requested I/O.

No-data errors only happen on tape, and indicate that the tape drive was unable to find any recorded data during the requested operation.

A soft error indicates that an error occurred, the driver retried the operation and it succeeded. The data is valid. These are usually indications that the media is aging.

A command error is a rare error indicating that the system made an invalid request of some sort.

An unclassified error is an extremely rare error type indicating that the driver was unable to determine what the problem was.

The status displayed (hhhh) is data that might be useful from the controller or drive. For example, the data displayed for a hard drive are the contents of the error register and the control/status register from the controller. Let's say that you saw an error 4Ø5B. If you look these numbers up in the Western Digital WD1Ø1Ø book, or decode them for yourself as I will describe later, you will see that the 4Ø means "ID not found" and the 5B means "Ready, seek complete, Data Request, Command-in-Progress, Error." For the floppy drive, the data displayed are the four status registers, SRØ - SR3. For the cartridge drive, the data displayed are the sense key, class, code, and address valid bit. Lastly, where the error occurred is displayed. For

the floppy and hard disk, the cylinder, head, and sector are displayed. For the cartridge drive, the block number is displayed.

panic: cd major number
This indicates that the system was put together inproperly and the disk cartridge driver was unable to find its major number. System inconsistency, fatal.

System Messages

**** ABNORMAL System Shutdown ****
This message appears when errors occur during normal system shutdown. It is usually accompanied by other system messages. System inconsistency, fatal.

bad block on dev nn/mm
A nonexistent disk block was found on, or is being inserted in, the structure's free list. System inconsistency.

bad count on dev nn/mm
A structural inconsistency in the superblock of a file system. The system attempts a repair, but hits message will probably be followed by more complaints about this file system. System inconsistency.

Bad free count on dev nn/mm
A structural inconsistency in the superblock of a file system. The system attempts a repair, but this message will probably be followed by more complaints about this file system. System inconsistency.

Can't allocate message buffer
This message appears during Xenix startup if memory cannot be found for the message buffers used by the interprocess communication calls. This represents an internal system error unless there is an inadequate amount of memory present in the system. Xenix requires that your system have at least 512k memory for reasonable performance. Abnormal.

iaddress>2^24
This indicates an attempted reference to an illegal block number, one so large that it could only occur on a file system larger than 8 billion bytes. Abnormal.

Inode table overflow

Each open file requires an inode entry to be kept in memory. When this table overflows, the specific request (usually a call to **open** or **creat**) is refused. Although not fatal to the system, this event may damage the operation of various spoolers, daemons, the mailer, and other important utilities. Anomalous results and missing data files are a common result. The process will exit with the error "ENFILE". Abnormal.

interrupt from unknown device, vec=xxxx
panic: unknown interrupt

The CPU received an interrupt via a supposedly unused vector. Typically this event comes about when a hardware failure miscomputes the vector of a valid interrupt. Hardware, fatal.

Map overflow (map type indicatory), shutdown and reboot

The map indicated by the map type indicator has fragmented to such an extent that there are not enough map elements to keep track of all the resource pieces. The map is either the core map, the swap map, the message map used in interprocess communications, or the semaphore map. Abnormal.

Maxmem was reduced based on the size of the swap area.
Refer to the system documentation for information on
the relationship between memory size and swap size.

Maxmem represents the maximum amount of memory available to one user process. It is normally 75% of all available user memory. However, the swap area must be at least as big as **maxmem**, since the largest process allowable must be able to swap out as necessary. If, on startup, the kernel discovers that **maxmem** is larger than the swap area, it will reduce **maxmem** to approximately the size of the swap area. If you get this message when you start up your Xenix system, it indicates that you need to increase the size of the swap partition on your file system in order to make maximum use of the memory available on your machine. This requires backing up the disk, repartitioning the disk, and reloading all file systems on the disk. This should be done with extreme care. Abnormal.

no file

There are too many open files, the system has run out of entries in its "open file" table. The warnings given for the message "inode table overflow" apply here. Abnormal.

no space on dev nn/mm

This message means that the specified file system has run out of free blocks. Although not normally as serious, the warnings discussed for "inode table overflow" apply: often programs are written casually and ignore the error code returned when they tried to write to the disk; this results in missing data and "holes" in data files. The system administrator should keep close watch on the amount of free disk space and take steps to avoid this situation. Abnormal.

**** Normal System Shutdown ****

This message appears when the system has been shutdown properly. It indicates that the machine may now be rebooted or powered down.

Out of device descriptors, increase gdt size (NGDT) and relink XENIX

The number of global descriptor table (gdt) entries that are reserved for device drivers is inadequate to fill the requests of the drivers on the system. A device driver has requested a **gdt** entry and has been refused because the table is full. The value assigned to NGDT in **/usr/sys/h/machdep.h** must be increased, and the kernel must be relinked after removing the old configuration file, **/usr/sys/conf/c.c**. Abnormal.

Out of inodes on dev nn/mm

The indicated file system has run out of free inodes. The number of inodes available on a file system is determined when **mkfs** is run. The default number is quite generous; this message should be very rare. The only recourse is to remove some worthless files from that file system, or dump the entire sytem to a backup device, rerun **mkfs** with more inodes specified, and restore the files from backup. Abnormal.

out of text

When programs linked with the **ld -i** or **-n** switch are run, a table entry is made so that only one copy of the pure text will be in memory even if there are multiple copies of the program running. This message appears when this table is full. The system refuses to run the program which caused the overflow. Note that there is only one entry in this table for each different pure text program. Multiple copies of one program will not require multiple table entries. Each "sticky" program requires a permanent entry in this table; nonsticky pure text programs require an entry only when there is at least one copy being executed. The process will be killed. You can resubmit it at a later, less busy, time. Abnormal.

panic: bad 287 int

Attempted execution of a real mode 287 instruction. System inconsistency, fatal.

panic: blkdev
panic: devtab
An internal disk I/O request, already verified as valid, is discovered to be referring to a nonexistent disk. System inconsistency, fatal.

panic: iinit
The super-block of the root file system could not be read. This message occurs only at boot time and indicates that a device error occurred and may mean that the root or swap device need repair. Hardware, fatal.

panic: IO error in swap
A fatal I/O error occurred while reading or writing the swap area. Hardware, fatal.

panic: Kernel buffer crosses 64K boundary, change load address
Xenix cannot have any of its I/O buffers spanning a 64K address boundary. This is checked on system startup and can be corrected by specifying a different Xenix load address to the boot program when you boot Xenix. Abnormal, fatal.

panic: memory failure - parity error
A hardware memory failure trap has been taken. If the system contains ECC (Error Correcting Code) memory boards, the message will be slightly different and may indicate the board that caused the error. Xenix will only panic if the error cannot be corrected on the ECC memory board. System inconsistency, fatal.

panic: memory management failure
An error occurred during memroy management operations. That is, one of the memory management routines encountered an error when doing critical copying or allocating functions. This indicates an internal error and not an equipment malfunction. System inconsistency, fatal.

panic: no fs
The device specified in a command is not currently mounted. This error condition has occurred deep in the kernel system call processing and as such should never occur since the possibiblity has been checked in earlier system code. System inconsistency, fatal.

panic: no imt
A mounted file system does not have an entry in the mount table. This should not happen because the mount table is checked by previous code. System inconsistency, fatal.

panic: no procs
Each user is limited in the amount of simultaneous processes he can have; an attempt to create a new process when none is available or when the user's limit is exceeded is refused. That is an occasional event and produces no console messages; this panic occurs when the kernel has certified that a free process table entry is available and yet can't find one when it goes to get it. System inconsistency, fatal.

panic: Out of swap
There is insufficient space on the swap disk to hold a task. The system refuses to create tasks when it feels there is insufficient disk space. Repeated occurrence of this error indicates that the size of the swap partition should be increased.

panic: general protection trap
General protection trap has been taken in the kernel. System inconsistency, fatal.

panic: preadi
An error occurred when the system was attempting to read in a process' segments during a process switch. This represents an internal operating system failure. System inconsistency.

panic: segment not present
An attempt has been made to access an invalid segment. It may also indicate the segment-not-present trap has been taken in the kernel. System inconsistency, fatal.

panic: Timeout table overflow
The timeout table is full. Timeout requests are generated by device drivers; there should usually be room for one entry per system serial line plus ten more for other usages. There is no acceptable behavior that the kernel can provide if this situation arises, except to panic. System inconsistency, fatal.

panic: Trap in system
The CPU has generated an illegal instruction trap while executing kernel or device driver code. This message is preceded with an information dump describing the trap. System inconsistency, fatal.

panic: Invalid TSS
Internal tables have become corrupted. System inconsistency, fatal.

panic: unknown interrupt
The CPU received an interrupt via a supposedly unused vector. Typically this event comes about when a hardware failure miscomputes the vector of a valid interrupt. Hardware, fatal.

proc on q
The system attempts to queue a process already on the process ready-to-run queue. System inconsistency, fatal.

spurious kb interrupt
This message indicates a serious hardware malfunction associated with the system console. Hardware.

Trap violation number in USER (or SYSTEM)
This message precedes a "panic:" message and occurs when a process causes an exception or trap on the 286 that cannot be handled in software. The violation number is the vector number of the exception. A description of the trap vectors on the 286 can be found later in this appendix and also in the iAPX 286 Programmer's Reference Manual published by Intel.

A trap that occurs while the 286 was executing user code is said to have occurred in USER mode; one that occurs while executing kernel code has occurred in SYSTEM mode. The message is followed by a dump of registers. System inconsistency, fatal.

The following is a set of panic messages that may appear:

287 Exception
The 286 thinks it has received a trap from the 287 when there is no 287 on the system.

287 Segment Overrun
A trap type 9 (processor Extension Segment Overrun Exception) has occurred on a system that does not have a 287. This trap indicates that the 287 has overrun the limit of a segment while attempting to read or write the second or subsequent word of an operand.

If a system that does have a 287 generates this exception, it will be killed with a SIGSEGV (segment violation) signal.

bad 287 int
A trap type 7 (Processor Extension Not Available Exception) has occurred on a system that has a 287 chip present. If the system has no 287 on board, the process generating this exception will receive a SIGILL (illegal instruction) signal and will be killed.

Protection Exceptions, Reserved Vectors, and What They Mean:

Below you will find a list of Protected Mode Exception Vectors and some explanation of their meaning.

Vector #	Description
Ø	Divide Error Exception
1	Single Step Interrupt
2	NMI Interrupt
3	Breakpoint Interrupt
4	INTO Detected Overflow Exception
5	BOUND Range Exceeded Exception
6	Invalid Opcode Exception
7	Processor Extension Not Available Exception
8	Double Exception Detected
9	Processor Extension Segment Overrun Interrupt
1Ø	Invalid Task State Segment
11	Segment Not Present
12	Stack Segment Overrun or Not Present
13	General Protection

I won't go into massive detail on the explanation of these, but hopefully there will be enough here to give you something to go on. For more information, you can refer to the Intel documentation.

First, a little explanation of what these things are might be in order. "A protection violation will cause an exception, i.e., a non-maskable interrupt.... Protection exceptions can be classified into program errors or implicit requests for service. The latter include stack overflow and not-present faults. Examples of program errors include attempting to write into a read-only segment, or violating segment limits." That's a quote directly from the Intel manual. Now, to discuss what each exception means....

When the 286 execution unit detects an invalid opcode, **interrupt 6** is invoked. The invalid opcode is not detected until an attempt is made to execute it; this means that pre-fetching the opcode does not cause this exception. This exception will occur for all cases of an invalid operand. Examples include instructions with total length of more than 1Ø bytes, and intersegment jumps referencing a register operand. This situation can be a sign of memory problems, problems with the 286, or questionable software. You may also be fighting a problem like injected noise on a clock line, which may be orignating with an expansion board (like a graphics adapter) in the 3ØØØ.

If two separate protection violations occur during a single instruction, **interrupt 8** (double fault) occurs. If yet another protection violation occurs when this is being processed, the 286 will enter shutdown, at which point no further instructions or exceptions are processed. Either NMI or RESET can force the 286 out of shutdown, under the right circumstances; an NMI input can

bring the CPU out of shutdown if no errors occur while processing the NMI interrupt; otherwise, shutdown can only be exited via the RESET input. NMI causes the CPU to remain in protected mode (which is what Xenix runs in) and RESET causes it to exit protected mode.

Interrupt 9 signals that the processor extension (in our case, the 287) has overrun the limit of a segment while attempting to read/write the second or subsequent words of an operand. If you get this error when there is no 287 in the system, it is frequently a good sign of confusion on the part of the Xenix kernel or the 286.

Interrupt 10 is invoked if during a task switch (changing from one task to another in a multi-tasking environment like Xenix) the new Task State Segment pointed to by the task gate is invalid. There are a variety of conditions (from the standpoint of the 286) which can cause this. I would refer you to the Intel documentation for more information on this particular problem.

Interrupt 11 occurs when an attempt is made to load a not-present segment of memory or to use a control descriptor that is marked not-present.

Stack faults, such as overflow or underflow and non-present stack segments cause **interrupt 12**. **Interrupt 13** is caused by protection violations which are not covered in the preceeding paragraphs.

Interrupt 0 is the divide-error exception; it occurs during divide operations when the quotient is too large to be represented, or when the divisor is zero. Inasmuch as the operating system software should protect the CPU from errors of this type, if the software is reinstalled and the problem persists the memory and CPU should be suspected. **Interrupt 5** occurs when a value exceed the bound set for it. **Interrupt 1** is the single-step interrupt; inasmuch as this allows programs to single-step and therefore is primarily a development tool, this error should rarely if ever be seen.

****

A Quick Reference to the
Most Commonly Used
Xenix Commands

****

This is meant to be a quick guide to commonly used commands in Xenix. It is by no means (and I do mean that!) complete, but it should contain enough information to let you do just about everything that you would need to do on site or in the shop with a troublesome system.

When you try these commands, type **boldface text** exactly as I have it here. Plain text indicates a variable or an option.

accept allows print requests to a lineprinter or class of printers

/usr/lib/accept destinations
allows **lp** to accept requests for the named destinations

cancel cancel requests to the lineprinter

cancel [request-ID's] [printers]
cancels the requested jobs by specific ID numbers, or cancels the currently printing job on the requested printer

cat concatenate, or display, text on the screen

cat filename
displays the contents of the file on the screen

cd change directory

cd
returns you to your "home" directory

cd directory-name
changes your working directory to directory-name

chgrp change group

chgrp group-name filename(s)
changes the group of filename(s) to group-name.

Valid group names may be found as the first field in **/etc/group**.

chmod change mode

chmod who operation-code permissions filename
changes the permissions mode for file.

who

a	all (user, group, others)
g	group
o	others
u	user (login owner)

operation-code

+	add permission
-	remove permission
=	assign all permissions (read, write, and execute)

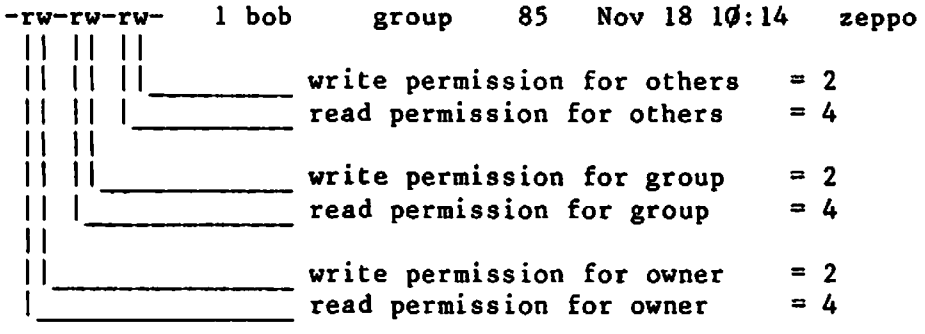
permissions

r	read
w	write
x	execute

chmod can also be used by using an octal number to determine what permissions you want set. For example:

```
chmod 666 zeppo
```

would set the following permissions for the file "zeppo":



If you add the results for each set together, you come up with the number that you gave **chmod**.

chown change owner

chgrp owner-name filename(s)
changes the group of filename(s) to owner-name.

Valid group names may be found as the first field in **/etc/passwd**.

copy copy groups of files

copy source-directory destination-directory
copy the contents of source-directory to destination-directory.

If files, directories, or special files do not exist at the destination, they are created with the same modes and flags as the source.

cp copy file

cp filename1 filename2
makes a copy of filename1 and names it filename2.

cp filename1 filename2 directoryname
copies 2 files into the specified directory.

date displays the current date and time on the screen

df disk space free

df
displays number of blocks and i-nodes free on all mounted file systems

df device-name
displays number of blocks and i-nodes free on device-name, whether it is mounted or unmounted.

doscat copies one or more MS-DOS files to the standard output

doscat filename
copies filename to the standard output.

doscat -r filename
copies filename to the standard output without doing a conversion from the MS-DOS CR-LF format to the Xenix newline format. This allows a true byte-for-byte display.

The filename argument has the form:

device:name

where device is a Xenix pathname for the special device file containing the MS-DOS disk, and name is a pathname to a file or directory on the MS-DOS disk. The two components are separated by a colon ":". For example, the argument:

/dev/fdØ:/src/file.asm

specifies the MS-DOS file, file.asm, in the /src directory on the disk in the device file, /dev/fdØ. Note that slashes and not backslashes are used as filename separators in MS-DOS pathnames. Arguments without a "device:" are assumed to be Xenix files.

For convenience, the default file, **/etc/default/dos**, can define MS-DOS drive names to be used in place of the special device file pathnames. For example, if the file contains the following line:

C=/dev/hdØd

then drive letter "C" may be used in place of the special device file name /dev/hdØd when referencing MS-DOS files.

doscp copies files between an MS-DOS disk and a Xenix file system.

doscp [-r] file1 file2
copies file1 to file2. The -r option is as that in the **doscat** command.

doscp [-r] file1 file2 directoryname
copies file1 and file2 to directoryname. The -r option is as that in the **doscat** command.

dosdir lists MS-DOS files in the standard MS-DOS style directory format

dosdir directoryname
lists the directory contents of directoryname.

dosls list MS-DOS directories and files in a Xenix style (see **ls**)

dosls directoryname
lists the directory contents of directoryname in Xenix format.

dosmkdir create a directory on an MS-DOS disk.

dosmkdir directoryname
creates directoryname on the MS-DOS disk specified by the "device:" portion of the directory name.

dosrm removes files from an MS-DOS disk.

dosrm file1
removes file1 from the MS-DOS disk specified by the "device:" portion of the filename.

dosrmdir remove an empty directory from an MS-DOS disk.

dosrmdir directoryname
removes directoryname from the MS-DOS disk specified by the "device:' portion of the directory name.

du disk usage

du directory
displays the number of blocks in use on the specified directory and in what files

find finds files or directories

find pathname-list **-name** 'expression' **-print**
will search for items matching 'expression' in the path described by pathname-list and prints the results to standard output

grep finds a pattern or patterns in a file

grep 'search-string' filename
finds and prints to standard output each line in filename with an occurrence of search-string in the line.

haltsys halts the system. (Should be used in conjunction with **sync**)

head prints the first few lines of a file

head filename
prints the first 10 lines of filename

install install application and upgrade software

kill terminate a process

kill -9 number
terminates with extreme prejudice the process with the process ID of number

l, ls, lc list, or display, the contents of a directory and information on files in that directory

l directory-name
lists in long format the contents of directory-name

ls directory-name
lists only the names of the contents of directory-name

lc directory-name
lists in columnar format the contents of **directory-name**

lpr, lp printer spooler

lpr filename
lp filename
prints the file filename on the default system printer

lpadmin configures the line printer spooling system

/usr/lib/lpadmin -d [destination]
makes destination the new system default printer

/usr/lib/lpadmin -x [destination]
removes destination from the line printer system. If destination is also the only member of a class, the class will be deleted, too. No other options are allowed with **-x**.

/usr/lib/lpadmin -p [printer] [options]
names a printer to which all of the following options refer. If printer does not exist, it will be created.

-c [class]	inserts printer into the specified class. Class will be created if it does not already exist.

-e [printer]	copies an existing printer's interface program as the new interface program for printer.
-h	indicates that the device associated with printer is hardwired. This option is assumed when creating a new printer unless the -l option is supplied.
-i [interface]	establishes a new interface program for printer. Interface is the pathname of the new program.
-l	indicates that the device associated with printer is a login terminal. The lineprinter scheduler, **lpsched**, disables all login terminals automatically each time it is started. Before reenabling printer, its current device should be established using **lpadmin**.
-r [class]	removes the printer from the specified class. If printer is the last member of the class, then the class will be removed.
-v [device]	associates a new device with printer. Device is the pathname of a file that is writable by the system manager, **lp**. Note that there is nothing to stop a system manager from associating the same device with more than one printer. If only the -p and -v options are supplied, then **lpadmin** may be used while the scheduler is running.

lpinit add new lineprinters to system

lpmove moves lineprinter requests

/usr/lib/lpmove requests destinations
moves requests to a new destination.

/usr/lib/lpmove dest1 dest2
move all jobs slated for dest1 to dest2. As a side effect, **lp** will reject all requests for dest1.

lpsched schedules requests taken by **lp** for printing

lpshut stops the line printer request scheduler

lpstat prints line printer status information

lpstat [options...]
If no options are given, **lpstat** will print the status of all requests made to **lp** by the user. Any arguments that are not options are assumed to be request ID's of the form returned by **lp**. **Lpstat** will then print the status of those requests. The following options are available:

-a[list]	prints acceptance status (with respect to **lp**) of destinations for requests. List is a list of printer and/or class names.
-c[list]	prints class names and their members. List is a list of class names.
-d	prints the system default destination for **lp**.
-o[list]	prints the status of output requests. List is a list of printers, classes, and/or requests.
-p[list]	prints the status of printers. List is a list of printer names.
-r	prints the status of the lineprinter scheduler, **lpsched**.
-s	prints a status summary, including the status of the lineprinter scheduler, the system default printer, a list of class names and their members, and a list of printers and their associated devices.
-t	prints all status information
-u[list]	prints status of output requests for users. List is a list of login names.
-v[list]	prints the names of printers and the pathnames of the devices associated with them. List is a list of printer names.

mkdir make directory

mkdir directory-name
creates directory-name

more display text on the screen, 1 screen at a time. Allows forward movement only (see **pg**)

more filename
displays the contents of filename on the screen, 1 screen at a time.

mv move or rename files and directories

mv filename1 filename2
changes the name of filename1 to filename2

mv filename1 directory-name
moves filename1 to the specified directory-name

mv directory-name new-directory-name
changes the name of directory-name to new-directory-name

od octal dump

od -c filename
displays filename with bytes displayed in ASCII

od -d filename
displays filename with bytes displayed as decimal numbers

passwd change your login password

pg displays text one page at a time; allows you to move forward and backward in a file (see **more**)

ps process status

ps -e
displays information about all active processes

pwd print working directory

quot summarize file system ownership

quot -f filesystem
displays the number of blocks and the number of files owned by each user in the specified filesystem

reject prevents print requests to a lineprinter or class of printers

/usr/lib/reject [-r[reason]] destinations
prevents print requests from being accepted by destinations. If the -r[reason] option is used, reason will be reported by **lp** when users direct requests to the named destinations. Reason is also reported by **lpstat**. If reason is more than one word, it must be enclosed in quotes.

rm remove files or directories

rm filename
removes the specified file

rmdir empty-directory-name
removes the specified empty directory

rmuser user-name
removes the specified user

su allows one to become (temporarily) root or another user

sync updates the superblock. Frequently used in conjunction with **haltsys**

tail displays the end of a file

tail filename
displays the last 1Ø lines of filename

tar tape archive. See Chapter 8 for detailed instructions on usage.

wall write to all users on the system

who who is on the system

vsh invoke the visual shell

For more information on these commands or for commands which aren't listed here, try consulting either the online **help** pages or the Commands Reference documentation.

SERVICE POLICY

Radio Shack's nationwide network of service facilities provides quick, convenient, and reliable repair services for all of its computer products, in most instances. Warranty service will be performed in accordance with Radio Shack's Limited Warranty. Non-warranty service will be provided at reasonable parts and labor costs.

Because of the sensitivity of computer equipment, and the problems which can result from improper servicing, the following limitations also apply to the services offered by Radio Shack:

1. If any of the warranty seals on any Radio Shack computer products are broken, Radio Shack reserves the right to refuse to service the equipment or to void any remaining warranty on the equipment.
2. If any Radio Shack computer equipment has been modified so that it is not within manufacturer's specifications, including, but not limited to, the installation of any non-Radio Shack parts, components, or replacement boards, then Radio Shack reserves the right to refuse to service the equipment, void any remaining warranty, remove and replace any non-Radio Shack part found in the equipment, and perform whatever modifications are necessary to return the equipment to original factory manufacturer's specifications.
3. The cost for the labor and parts required to return the Radio Shack computer equipment to original manufacturer's specifications will be charged to the customer in addition to the normal repair charge.

RADIO SHACK, A DIVISION OF TANDY CORPORATION
U.S.A.: FORT WORTH, TEXAS 76102
CANADA: BARRIE, ONTARIO L4M 4W5

TANDY CORPORATION

AUSTRALIA
280-316 VICTORIA ROAD
RYDALMERE, N.S.W. 2116

BELGIUM
PARC INDUSTRIEL DE NANINNE
5140 NANINNE

U.K.
BILSTON ROAD WEDNESBURY
WEST MIDLANDS WS10 7JN

PRINTED IN U.S.A.