TANDY

TECHNICAL
INFORMATION SERIES

System III XENIX INFORMATION

0220 TECHNICAL SUPPORT SERVICES

Revision History

Field(V7)
Ø4/Ø2/1984
kjb

Field(Sys3)
Ø2/17/1986
vrs

Field(SysV)
1Ø/29/1986
vrs

Field(Sys3,v2)
Ø1/25/1988
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 Core System from the floppy diskettes provided in the User's Manual to the customer's primary hard drive. Also, the Xenix 3.2 upgrade procedure is explained here.

Section Two will discuss the hardware aspects of the system. Proper jumpering, card order, and mandatory modifications to the system will be explored.

Section Three deals with some software and control aspects of Xenix. The importance of the file **/usr/adm/messages**, media error map regeneration, and setting terminal options are discussed. Two editors (**ed** and **vi**) are described briefly, the software and hardware aspects of the multi-terminal interface are delved into, and file systems are explored.

Section Four is devoted to "hints and kinks" about the system. Current DOS versions are discussed here.

Section Five 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 Six addresses the topics of backing up data and upgrading systems. Again, this information is presented mainly as reference, although the odds are fairly 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 Core System, it is MANDATORY that you refer to Section Two and verify that ALL hardware modifications have been properly implemented.

****

****

Some Important Things You Need to Know

BEFORE STARTING

****

1) When running TRS-XENIX on a 68ØØØ based machine, the hard disks are numbered hdØ - hd3, where Drive hdØ is the primary hard disk.

2) The hard drives must not be write protected and the floppy diskettes must have a write enable tab. Approximately every 3Ø seconds XENIX will access the drives to update its files and directories. This will occur even when the system is idle.

3) TRS-XENIX utilizes the Media Error Map located on the bottom of the hard drive. Copy the contents of this map for each drive in the system and then replace the map into its sleeve on the bottom of its drive. Do this for all hard drives before beginning to initialize XENIX.

   If the Media Error Map is missing from the bottom of the unit, remove the cover and check for the map inside. If it still cannot be located, the bubble may need be checked with HDREL. Refer to Section Three 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 used previously for other data, operating systems, and etc., ensure that all needed programs and data files have been saved off, as **diskutil** 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 Four -- Hints and Kinks, or How to Overcome Various System Anomalies in Your Spare Time

Section Five -- Advanced System Information and Restoring Blown Xenix File Systems, or, How to Really Mung Things Up

****

SECTION ONE

Installing the Core System Without Pain,
and
How To Upgrade It

****

The first thing you need to be able to do with Xenix is figure out how to get the system installed. Whether you are running Xenix 3.Ø, or Xenix 3.2, this process is going to remain much the same. In the case of a system already running Xenix, you may sometimes need to know how to upgrade it to a newer version... and that's something I'll talk about in this chapter, too.

For the most part, the process of installing System 3 (Xenix 3.x) is very similar to installing Version 7 (Xenix 1.3.x). There are a few cosmetic differences which may be a little confusing, and those are what we'll mention here.

***** NOTE *****: System 3 uses a different format than that used by Version 7, so it is an absolute necessity to format the hard drive with the System 3 Boot Disk. This applies to all upgrades to **any** 3.x Xenix! It also applies if you are going from Xenix 3.Ø to Xenix 3.1. Fortunately, Xenix 3.2 is not as difficult.

.....

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

.....

1) Power up the computer, hard drive/s, and all connected peripherals. (If you are using a cartridge system, you need to power it up before you power up the computer; leaving it connected and off may confuse things a bit). Hold the <BREAK> and <REPEAT> keys until the "Insert Diskette" messages appears on the screen.

   NOTE: Make certain you allow at least a half hour for the hard drive to arrive at a constant operating temperature. I'm not joking about this! If you don't let them warm up this long, you really are asking for trouble later on.

2) Insert the System 3 **Boot Disk** into floppy drive Ø and allow the system to boot. Make sure the hard drives aren't write protected.

3) When the boot message appears, type **diskutil** after ">" and press the <ENTER> key.

   Xenix Boot> **diskutil** <ENTER>

.....

If you got an error message saying "BUGHLT: NewPal", then you are missing hardware modifications that are necessary to run System 3.

.....

4) Answer the following prompts:

NOTE: Anything you see which refers to cartridge drives is only going to appear in Xenix 3.1 or later.

Diskutil: format or copy floppy diskettes
format hard disks
format or copy disk cartridges
....

Diskutil: Hard, floppy, or cartridge (h, f, or c)?
Type <h> to format a hard drive.
Hard disk unit number (Ø..3)?
Answer Ø to indicate the primary hard drive.

Destructive or Non-destructive format (d or n)?
Answer d to ensure proper verification. (Use the non-destructive option only under special circumstances -- more on that later).

Additionally, if there is not currently a Xenix-compatible format on the hard drive, this question about "destructive/non-destructive" probably won't be asked. Don't worry about it.

5) The next prompts will ask how many heads and cylinders are to be formatted.

Eight Meg Hard Drives:

How many cylinders (tracks)?
Answer 256.

How many heads?
Answer 4.

Twelve Meg Hard Drives:

How many cylinders (tracks)?
Answer **23Ø**.

How many heads?
Answer **6**.

Fifteen Meg Hard Drives:

How many cylinders (tracks)?
Answer **3Ø6**.

How many heads?
Answer **6**.

Thirty-Five Meg Hard Drives (Quantum):

How many cylinders (tracks)?
Answer **512**.

How many heads?
Answer **8**.

Seventy Meg Hard Drives (Micropolis):

How many cylinders (tracks)?
Answer **1Ø24**.

How many heads?
Answer **8**.

6) The next question will be:

Desired interleave factor (<ENTER> for default)?
Answer **<ENTER>**. (Xenix sets the optimum value automatically, so you really aren't going to help matters by not using the default).

For more information on what exactly this "interleave" is, please look at the end of this section....

7) In this step you will enter the flaws on the primary media error map. This is identical to the way you enter errors in the older versions of Xenix; enter the cylinder-head pairs one at a time and type **done <ENTER>** when you are finished.

8) After you type "done", the system will ask:

Full, partial, or no verify (f, p, n)?
Answer f. (There are times and places to use partial or no verify, but we'll cover those later).

.....

With **Drives Up to 15 Meg**, You May
Enter Up to **24 Flaws**.

With **35 and 7Ø Meg Drives**, You May
Enter Up to **96 Flaws**.

<BREAK> may be used to abort the process at any time should you make a mistake.

.....

9) The system should now display the message:

About to format hard disk drive Ø.
This will take about xx minutes.

Type <enter> to proceed or <break> to abort:

"xx" is approximately how long this process will take. Just like in Version 7 Xenix, you can bail out of this process at any time, but the drive will remain unusable until the process has run to completion. The cylinder number and the head number currently being formatted will be displayed.

1Ø) When the formatting is complete, the following will appear on the screen:

Hard disk successfully formatted.
Drive parameters and MEDIA ERROR MAP successfully written.
Your hard disk is ready for the XENIX initialization.

```
****************************************

    Now that the format is complete,
   the system is ready to be installed.

****************************************
```

Just like Version 7, System 3 uses a program called **hdinit** to initialize your hard drive. It does several neat things for you:

* Copy the boot track to the hard disk.
* Create a file system on the hard disk. (Note: System 3 uses a different file system structure than Version 7 -- beware!)
* Copy the contents of the Boot Disk to the hard disk.
* Performs a system shutdown after the initialization.

11) With the Boot Disk still installed in floppy drive Ø, press the RESET button while holding the <BREAK> and <REPEAT> keys. The boot prompt will appear. Press <ENTER>.

```
Xenix Boot> <ENTER>
```

12) The screen will now show:

```
System loaded . . .
Change root disk if desired;
type <enter> to proceed or <break> to abort
```

Press <ENTER>.

13) The system will ask you the following questions:

```
Do you wish to initialize your hard disk?
        Answer <y> for yes.

Has your hard disk been formatted with diskutil?
        Answer <y> for yes.

Is this a Model II? [y or n]
        If you are installing Xenix on an upgraded
        Model II, answer <y>, otherwise answer <n>.
```

.....

If you answer "Model II", your boot disk will continue to work on Model II's.

If you say "no", this boot disk will freak Shugart and TPI drives out, because the floppy drive rates will be set for thinline drives.

The simplest solution is to use two different sets of disks: one for thinline drives, and one for Model II's.

.....

Do you want the standard size swap area? [y or n]
Answer <y> for yes. (There are reasons to say no, but they'll be covered later).

14) At this point, the system will proceed to initialize itself. One of the major departures from Version 7 installations procedures is that the program will determine for itself how many heads and cylinders the drive has (by reading the media error map you entered during diskutil).

.....

DO NOT TOUCH THE SYSTEM UNTIL YOU SEE THE MESSAGE:

**** Normal System Shutdown ****

.....

NOTE: If for some reason you are repeating this procedure, you may be warned, while the filesystem is being created, that **"mkfs contains data"**. You will be asked whether to "overwrite". Answer <y> to finish installing the system.

****

Home Stretch....
After the ***Normal System Shutdown*** message,
RESET the computer and boot from the
hard drive.

****

15) This part, copying the remaining Xenix floppy diskettes onto the hard drive, is just like it was under Version 7. Reboot the system from the hard disk if you have not already done so.

16) As before, press <ENTER> after you see the "Xenix Boot>" prompt. The system will display information on inserting and removing the Install floppy diskettes.

17) You are prompted with the message:

First Floppy? [y or n]

Insert the **Install 1** floppy diskette into Drive Ø. Press **<y>** and then <ENTER>.

When the data is transferred to your hard drive, you will be prompted with:

Next Floppy? [y or n]

Insert **Install 2** and answer the prompt with **<y>** and <ENTER>. Repeat for Install 3, if you have one. In Xenix 3.1, you will; in Xenix 3.Ø and 3.2, you won't. A further note about Xenix 3.2 -- the Install disks are (usually) double-sided! Check this out if you have problems installing Xenix 3.2 on a Model II based system.

18) When the Install floppies have all been transferred to the hard drive, answer the prompt with **<n>** and <ENTER>.

You will see the message:

Setting up directories and permissions.

19) The system will ask you for information concerning your location and timezone.

.....

Answering the "timezone" question with "Yukon" will not make your customer very happy (unless you're in Alaska...)

.....

Congratulations... the primary hard drive now contains a complete XENIX core system and the installation is complete.... **UNLESS** you're using Xenix 3.1 or 3.2, in which case you may want to install the Help files. But I'll get to that in just a minute. Meanwhile, the screen should now say:

Installation complete.

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

This last is the normal XENIX boot prompt you may expect whenever you boot from the hard drive..

At this point, you are ready to try this System 3 stuff out.

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

This tells the system to engage a normal startup procedure. As usual, the system will prompt you to enter the correct date and time. Although you can bypass this with <ENTER>, it is recommended that you actually go ahead and give the correct information. (That way, if the system has a problem, and the error gets logged in the "/usr/adm/messages" file (a log of system errors), or if the system reboots as it can in Xenix 3.2 and sends mail to the system administrator, you'll stand a better chance of seeing if the error is related to a specific time of day or operation).

The screen will clear and in the upper left corner of the screen a login prompt will be displayed. Type in **root** and press **<ENTER>**.

login: root <ENTER>

Now, those of you using Xenix 3.1 or Xenix 3.2 who want to install the Help files, pay attention. The rest of you can kick back and relax for a while.

****

Installing Xenix Help Files
(Nice if you've got them).

****

As it turns out, installing the Xenix Help files teaches you how to do other applications installations as well. At the root prompt (which is a pound sign # just like it was in Version 7) type **install <ENTER>**. You will get the following menu:

Installation Menu
1. to install
q. to quit

Please select :

Type **1 <ENTER>** to select the Install option. The screen will say:

Insert diskette in Drive Ø and press <ENTER>

Insert the Help files diskette in floppy drive Ø, close the drive door, and press <ENTER>. Various grinding noises will emerge from your floppy drive, but eventually you should see:

Installation complete -- Remove the diskette, then press <ENTER>.

Remove the Help floppy diskette and press <ENTER>. The Installation Menu will reappear. Type **q <ENTER>** to quit.

Simple, eh? As it turns out, that's how most of the applications packages (things like Profile-16, Scripsit-16, Multiplan, the Development System, and the like) get installed, not to mention the Xenix Tools Diskette. In any case, now you can type

help [subject name]

and get help on that command.

By the way, if you want help files on a brand new Xenix 3.2 system, since the distribution does not come with a Help Pages Disk, you can install the Help Pages from Xenix 3.1. They install just the same way, and will work the same way. There may be some minor differences between the documentation for Xenix 3.1 and 3.2, but for the most part you will find them quite adequate!

***** NOTE *****: For those of you who aren't using Xenix 3.1.x, fear not. There are some help files available to you, too, and we'll cover getting to them fairly soon.

Well, those of you who weren't paying attention earlier should wake up now, since we're going to struggle into learning about System 3 checkout procedures.

****

Checking out the system
or
It still takes guts, gang!

****

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. 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 TRS-XENIX" (or something similar) and some seconds later the root prompt " # " will be displayed. Time to dive in.

A great many of the things you were familiar with in Version 7 Xenix will still work in System 3. Here are a few simple commands which will allow you to try the basics out on the system.

1) l **<ENTER>** (l = lower case L)
   This command produces the same long version directory listing that it did under Version 7 Xenix.

.....

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

These still work, and the <HOLD> key
still works at the console.

.....

2) **ps -elf <ENTER>** (You **HAVE** to use the minus sign!)
   This is the command which gives you the current process status on the system. Syntax for this one is different than that on Version 7; if you type "ps lax" by accident you will get a nasty little message which instructs you on usage.

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

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

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

shutdown

The system will ask you "how many minutes to shutdown"?. Enter a Ø for an immediate shutdown. This is pretty much the same semi-graceful technique one uses to shut Version 7 systems down; once you get the

**** Normal System Shutdown ****

message 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 **tsh** exists on System 3 (and it has Help available). It is a good way to introduce yourself to Xenix, and the Version 7 Xenix Information Bulletin covers the basics in it fairly well.

.....

Never turn the system off without
first performing shutdown,
or a reasonable fascimile thereof.

(Some things don't change...)

.....

To install additional hard drives, you will format them with **diskutil**. However, the name of the utility you will use to make a file system on the new drive has changed... it is now called **makefs**. There will be more discussion on this in Section 3.

.....

Some Notes On the Mysterious Subject
of
Interleave
and
The Various Types Thereof

.....

I told you earlier that I would try to explain what this thing "interleave" is. Well, here goes... although by the end of the explanation you may wish that you had never asked the question!

When formatting a hard drive using **diskutil**, you are asked for an interleave value. It turns out that **diskutil** uses something called **physical interleave**; this is to help get data to and from the hard drive as quickly as possible. The whole thing revolves around the fact that a hard drive rotates at a rate of speed which is higher than the rate at which we can read or write it... yet we don't want to waste any more time than we have to waiting for the next bit of data to come around once we are ready for it. What do we do now? I mean, if we lay data down in sectors one right after another, it goes by too quickly for us to read it sequentially, and we have to wait to read the sector(s) we missed until it comes around to us on the next revolution. This is obviously going to waste lots of time, so we get around the problem by numbering the sectors as we can pick them up. It goes like this:

Physical Sector Numbers:	1	2	3	4	5	6	7	8	9	1Ø	11	12	13
diskutil Sector Numbers:	1			2			3			4			5

See what's happening? **diskutil**'s numbering the sectors with spaces between them so that by the time the controller is ready to grab another sector's worth of data, the next sector is right there to be picked up. Actually, this is a bit of an oversimplification: to see how a track would really be numbered by **diskutil** see the next diagram... incidentally, Xenix configures hard drives for **17 sectors per track**.

Physical:	1	2	3	4	5	6	7	8	9	1Ø	11	12	13	14	15	16	17
diskutil:	1	7	13	2	8	14	3	9	15	4	1Ø	16	5	11	17	6	12

Ok, now the above discussion is based on a physical interleave of three -- which is coincidentally what Xenix 3.1 or 3.2 **diskutil** uses, and brings us down to one of the reasons that you <u>have</u> to reformat the hard drive when doing an upgrade from Xenix 3.Ø to the later versions. The 3.Ø **diskutil** uses an interleave of five. "So what," you say, being quick on the uptake, "this should only mean that your I/O will go kind of slowly compared to what it could be doing." True as far as it goes, but unfortunately there is more to

this story. It turns out that this physical interleave that **diskutil** uses also has a large effect on a logical interleave that **mkfs** uses. (**mkfs** is the utility that is used to create Xenix file systems). This logical interleave determines how the free blocks in a file system are referenced with regard to one another; if the wrong physical interleave is matched with a logical interleave, the drive will essentially become unreadable (except for the boot track).

See where this is going now? If you leave your Xenix 3.Ø **diskutil**ed disk as it sits and use the Xenix 3.1 (or 3.2) **mkfs** on the drive, the drive will never get past the information on the boot track. Instead, it will sit there and mumble to itself, and you will become extremely frustrated. The same thing in essence goes for the upgrade from Xenix 1.3.5 to Xenix 3.x, so the best thing to do in the case of an upgrade is reformat. Even if it's not necessary, it's a good idea in any event.

****

Upgrading to Xenix Ø3.Ø2.ØØ

(Not too hard, really... but a few gotchas!)

****

Let's say that you have a customer already running Xenix 3.1, and they just bought the Ø3.Ø2.ØØ Upgrade. Unfortunately, they haven't yet figured out just quite how to get it installed... and when you were out on the site to align their floppy drive, they asked you, "Oh, by the way, would you mind...?" Well, fear not. Most of this upgrade procedure is going to seem very familiar to you, since we've duplicated it in the first part of this chapter.

One thing, though... this is a **running** system, so we need to make certain that if anything goes wrong, we can backtrack and start over. To do that, you need to **ENSURE** that there's a backup!! (BEFORE you do the upgrade!)

.....

Does the customer have a **CURRENT** backup of the system?

If you verify that "yes, they do", go ahead with the upgrade.

If the answer's "no", do not pass GO, do not collect $2ØØ.
Instead, proceed directly to Section 6 and back the system up!

.....

Ninety-nine times out of a hundred, the installation will go perfectly, and the backup won't be needed. Unfortunately, it's that hundredth time that's the killer... so take the extra time and make the backup. Believe me, it's worth your while!

Ok, so now the system's backed up... let me take a brief step sideways for a moment and explain just what this upgrade is going to do. Simply put, it's going to replace almost all of the core programs normally associated with a Xenix installation. It's also going to replace a lot of associated files. Now, before you panic, let me tell you the files which the upgrade will leave alone:

/.profile	/etc/checklist	/etc/cshrc
/etc/default/backup	/etc/default/cron	/etc/default/dump
/etc/default/dumpdir	/etc/default/login	/etc/default/lpd
/etc/default/mkuser	/etc/default/passwd	/etc/default/restor
/etc/default/su	/etc/default/tar	/etc/group
/etc/passwd	/etc/ttys	/etc/ttytype
/usr/lib/crontab	/usr/lib/uucp/L-devices	

These files are frequently ones which are customized for a given system, which is why they are left alone. There are four more files which are commonly tailored to a system's needs which will be replaced, but are backed up:

/etc/default/tsh	/etc/motd	/etc/rc
/etc/termcap		

The backups of these files will appear in /etc/default/tsh-, /etc/motd-, /etc/rc-, and /etc/termcap-. If your customer has modified the old versions of these files, they may wish to do the same mods to the newer versions. There are some significant differences between the old and new versions, however, so for the most part it is a bad idea to just move the old files back! The recommended procedure is to duplicate the old file modifications in the new versions.

There is only one other minor detail to be contended with. You can only upgrade to 3.2 if the system is currently running a 3.1.x Xenix. In other words, if the customer is running 3.Ø, you need to upgrade them to 3.1 and then do the 3.2 Upgrade. (Fun, eh!?)

Now, back to the upgrade procedure...

1) Log in as root at the console. At the root prompt "#", type:

   who<ENTER>

   The system will respond with something like this.

   ```
   root     console Nov 17 11:14
   ```

   If anything other than a line for root comes up, i.e. somebody else is logged on the system, have them log out.

2) Next, type:

install<ENTER>

Your screen will show:

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

Please select :
```

Type 1 **<ENTER>** to select the Install option. The screen will say:

```
Insert diskette in Drive Ø and press <ENTER>
```

Insert the Upgrade Diskette in Drive Ø and press <ENTER>. You will see the following:

```
**********************************************
*                                            *
*       XENIX/68ØØØ    Version Ø3.Ø2.ØØ      *
*                System Update               *
*                                            *
**********************************************
```

If the system is not currently running a 3.1.x version of Xenix, you will see a message telling you to install Xenix 3.1 and then redo this upgrade. Also, if you don't have at least 22ØØ blocks free on the primary hard drive, you will be told to free up some space and try again. However, most of the time you will see this message:

```
**** Warning: THIS INSTALLATION WILL **HALT** THE SYSTEM UPON COMPLETION.
             If you are not ready to halt your system, type "no".
             To continue, press <ENTER>:
```

3) Press **<ENTER>** to continue the installation. The floppy drive will start to make noises, and the screen will say:

```
Backing up original copies of some system files.
This will take a few minutes.
```

The next thing you will see is some files being moved to the hard disk from the floppy.

4) The next prompt from the installation procedure is a familiar one explaining how to install the Install floppies. When you see this prompt:

    First Floppy? [y,n]

    place Install 1 in drive Ø and type:

    y<ENTER>

5) The system prompts you for the next disks with:

    Next Floppy? [y,n]

    Place Install 2 in drive Ø, and press **y<ENTER>** to continue. If you have more than two Install floppies (which is unlikely!) continue this process for the remaining disks.

6) When you have installed all of the Install disks, and see

    Next Floppy? [y,n]

    again, type:

    n<ENTER>

    to finish the installation.

7) Next, you will see:

    Setting up directories and permissions.

    Please reboot from the hard disk after the system shuts itself down.

    Halting system....

    **** Normal System Shutdown ****

8) After the "Normal System Shutdown" message appears, reboot off the hard drive. Once the system starts up, the screen shows:

    Is this a Model II? [y or n]

    Make the appropriate selection. The system will now ask you about daylight savings time and timezone again.

9) Once you have completed the timezone screen, the system will say:

```
Timezone changed
Dstflag changed
Timezone successfully updated
Reboot option is now OFF
Auto-boot option is now OFF
autoboot: Installing new boot track...
11+1 records in
11+1 records out

Installation complete.

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

1Ø) Type **<CTL><D>** for normal startup, enter the time when prompted, and you now have an upgraded Xenix 3.2 system.

Xenix 3.2 has quite a few new quirks and features, some of which may seem rather odd initially... but fear not, we'll talk about them later!

****

SECTION TWO

Care and Feeding
of
Xenix Support Hardware

(Yes, it will bite the hand which feeds it!)

****

Things have gotten rather more complicated with the advent of System 3 and the fast 68ØØØ hardware. There are also a couple more hard drives to contend with, and a new form of disk storage with the 1Ø meg disk cartridge system. As if this weren't enough, with the advent of Xenix 3.2 you can also use the Model II graphics board for console graphics and the 8 inch 2Ø meg disk cartridge system (or the original 1Ø meg!). All of this combines to make System 3 even more sensitive to hardware than Version 7 was... so hold on to your hats, there's lots of stuff to check.

When looking at the modifications for the various CPU's, I will note a basic division in what to look for. Essentially, there will be a breakdown between the systems running the older 6 MHz 68ØØØ hardware, and the newer 8 MHz 68ØØØ hardware.

So much for introductions -- here's what you need to look for!

Computer Hardware

Model II and 16:

1) Card Order (right to left, as viewed from rear):

Z-8Ø CPU
FDC
Arcnet (if installed)
Hard Disk Interface
Multi-terminal board (if installed)
Disk Cartridge Interface (if installed) *
VDG
Hi-Res Graphics (if installed)
Z-8Ø memory
68ØØØ cards

*** NOTE: The disk cartridge interface is a terminated interface. If it is installed, it must be the last interrupt driven device.**

2) Technical Bulletins to check for:

II:1	Correct floppy termination
II:2	Correct memory board jumpering
II:4	Zener diode on old style FDC board
II:5	Motorola video monitor board modification
II:8	Proper heat sinking on Motorola monitor bds
II:9	Old style FDC board modification and alignment
II:1Ø	Trace cut on early design FDC for use with Xenix
II:13	Early design FDC has C17 installed backwards
II:14	Correct retrace problem with RCA video bd
II:15	Proper Shugart stepper tightening
II:16	Boot Errors on Shugart Drive (LSI only)
II:18	Correct dual addressing problem on early VDG card
II:19	Correct late style FDC jumpering
II:2Ø	Boot Errors on Shugart Drive (Discrete only)
II:21	Wire jumper on early design FDC board and -Ø2 FDC chip
II:22	Correct Shugart sector sensor alignment
II:23	System lockup or wierd keyboard entries
II:24	Correct garbage on screen
II:25	Reset modification
II:26	Enhanced DMA modification
II:27	Correct side select signals
II:28	Cut and jump to correct head load termination
II:3Ø	Correct floppy drive termination
II:31	Missing jumpers on 64K RAM board and HD interface board

II:32	Missing jumpers on CPU board
II:33	Boot ROM checksums
II:35	Intermittent wire wrap jumpers
II:37	Motorola video board mod verification
II:38	Check for shorted Z4 on keybd
II:39	Correct intermittent TPI drive errors
II:41	Correct keyboard power on reset problem
II:42	Multiterminal board installation procedure
16:1	Correct FDC jumpering for 16A
16:3	Improve Z-8Ø clock on REV D CPU board
16:5	C41 on Z-8Ø CPU board installed backwards
16:6	Correct problems accessing drive 1
16:7	Correct thinline drive problems
16:8	Tandon motor speed modification
16:9	Correct thinline termination
16:1Ø	Incorrect power supply fuse
16:13	Correct intermittent speed errors
16:14	Correct early Tandon drive problems
16:15	Low 5 volts in card cage
16:25	Allow use of disk cartridge system
12/16B:12	Slow video RAM access
12/16B:26	Current limiting modification
12/16B:31	Improve VDG operation
INF:14	Correct failure to regulate due to damaged capacitor
I/O:41	Hi-res board mods
I/O:62	Reliability mods for multiterminal bd
I/O:1Ø1	DT-1ØØ multiterminal board mods
6ØØØ:4	Video board inverter incompatibility
6ØØØ:9	Video board mods
6ØØØ:12	Correct installation of R87 on belt drive Tandon
6ØØØ:13	Prevent "BUGCHK:SCSIFI" errors in Xenix

For machines running System 3 on the 6 MHz 68ØØØ boards:

16:18	Correct memory buffer problems on 6 MHz 68ØØØ memory PCB
16:23	Modifications to allow Xenix 3.Ø to run on 6 MHz boards
16:24	Reliability modifications to 6 MHz 68ØØØ CPU board
12/16B:19	Correct switch settings for 68ØØØ RAM bds
12/16B:34	Defective decoders causing Xenix errors

For machines running System 3 on the 8 MHz 68ØØØ boards:

16:21	8 MHz upgrade procedure
16:19	Correct ground connections on 8 MHz 68ØØØ CPU board
16:2Ø	Correct data setup to 8 MHz 68ØØØ memory board
6ØØØ:2	Correct timing error on 8 MHz 68ØØØ CPU board
6ØØØ:6	Reliability modifications to 8 MHz CPU board
6ØØØ:8	Reliability modifications to 8 MHz memory board

3) Proper Board Jumpering:

Reference Notes and Jumpers

Model 12, Model 16B, Model 16B+, and Tandy 6000:

1) Card Order (bottom to top when viewing rear of card cage):

Hard Disk Interface
Arcnet (if installed)
Multi-terminal card (if installed)
Disk Cartridge Interface (if installed) *
VDG
Hi-res board (if installed)
68000 memory
68000 CPU

* **NOTE: If the Disk Cartridge Interface is installed, it must be the last interrupt driven device in the chain.**

2) Technical Bulletins to check for:

12/16B:2	Increase reset drive on main logic board
12/16B:3	Power up problems - U81, main logic board
12/16B:5	Card Cage Interrupt modification (**verify that this one is correct**)
12/16B:6	DMA modification, main logic board
12/16B:8	Correct floppy drive termination
12/16B:9	Proper main logic board jumpering
12/16B:10	Cooling enhancement kit
12/16B:11	Change in SIO control signals
12/16B:12	Video Ram access timing change
12/16B:13	Change in type of tuning cap in FDC VCO circuit
12/16B:14	Add pullup to PSEL line for printer interface
12/16B:16	FDC alignment procedure
12/16B:17	Card cage guide change
12/16B:20	Motorola monitor brilliance adjustment
12/16B:21	Problems with Xenix and floppy drives
12/16B:23	Correct low 5 volt problem
12/16B:25	Missing components on AX-9432 controller board
12/16B:26	Power supply problems
12/16B:27	Internal hard drive supply may be jumpered for 230V
12/16B:29	RCA fan mount
12/16B:30	Direct drive Tandon boot errors mod
12/16B:31	Improve reliability of RCLOCK on VDG card
12/16B:32	Correct RFI between int. HD and floppy
12/16B:39	Faulty "AS" type chips on main logic board
12/16B:40	Correct card cage artwork errors.
6000:4	VDG inverter incompatibility
6000:7	Bad AS parts on int HDC

6000:9	Reliability mods for VDG
6000:11	Correct slow multiplexers
6000:12	Correct R87 installation on belt drive Tandons
6000:13	Correct "BUGCHK:SCSIFI" occurences in Xenix
6000:14	Reduce occurences of "Active Drive Not Ready"
6000:15	Correct power-on reset problems
INF:14	Correct failure to regulate due to damaged capacitor
INF:17	Correct low output voltages on HD supply
I/O:41	Hi-res mods
I/O:62	Multiterminal bd reliability mods
I/O:101	DT-100 multiterminal bd mods
16:7	Correct intermittent thinline errors
16:8	Correct motor speed circuit
16:10	Correct 24V AA11082 fusing
16:13	Intermittent speed errors
16:14	Correct early release Tandon problems

For machines running System 3 on the 6 MHz 68000 boards:

12/16B:19	68K memory board switch settings
12/16B:24	Correct memory buffer problems on 68k memory PCB
12/16B:34	Defective decoding chips, 68k CPU board
12/16B:41	Increase reliability of 68k CPU board
12/16B:44	Describe System 3 modifications

For machines running System 3 on the 8 MHz 68000 boards:

12/16B:38	Outline 8 MHz upgrade procedure
6000:1	Correct data setup time to memory board
6000:2	Correct timing error on 68000 CPU board
6000:3	Correct ground connections on 68000 CPU board
6000:6	Reliability modifications, 68k CPU
6000:8	Reliability modifications, 68k memory board

NOTE: **The above Technical Bulletins must be verified to ensure proper operation of a Xenix system.** Don't just look; verify these mods! **System 3 is even more sensitive to hardware than Version 7.**

68ØØØ Memory Boards

Everything that has been said about the touchiness of 68ØØØ memory boards is true in spades with the advent of the faster board set. As if the standard concerns weren't sufficient, you now get to worry about the possible permutations of CPU and memory boards as well. First things first.... these are the combinations which are supported (as of this writing) to run System 3.

6MHz CPU board (modified) with 128k/256k memory board(s)
6MHz CPU board (modified) with 512k/1meg memory board
8MHz CPU board with 512k/1meg memory

NOTE: The "dog-eared" or "long" 6MHz 68ØØØ is not supported for use with Xenix 3.x.

Correct switch settings and jumpers, a good +5 VDC (**at least +5.ØØ VDC while system is fully loaded, measured on the 68ØØØ memory board**), and correct ventilation are essential for the health and well-being of a Xenix system.

Hard Drive Hardware:

12/15/35/7Ø Megabyte Hard Disk Interface Card (AX-9367) And 12 Megabyte Hard Drives

The 12 meg hard disk interface PCB will need to be jumpered differently depending upon whether it is installed within a Model 12 or 16B/6ØØØ. This is due to a difference of internal memory configurations, as supplied from the factory, between the Model 12 and 16B/6ØØØ.

The Model 12, as shipped, will contain internal memory mapped at page 15 with a mirror image appearing at page 14. For this reason, it is necessary to remap the memory included on the 12 meg interface PCB to another location, customarily pages 8 and 9.

The Model 16B/6ØØØ is supplied with no internal memory mapped at pages 14 and 15; thus the interface card must supply this memory. Be aware that these differences apply not only to Xenix but to the 4.x TRSDOS operating systems as well.

Jumpering:

Model 12	AG - AL	selects pages 8 and 9
Model II/16/16B/6ØØØ	AK - AP	selects pages 14 and 15

Additional Jumpers (same for all computers):

A-B
A-B (cloverleaf next to CTC chip)
V-W

.....

*** * * NOTE: * * ***

All 12/15/35/7Ø meg interface boards
must be modified to comply with
Technical Bulletin HD:12.

.....

There is a modification to the hard drive interface board outlined in HD:12 which must be performed whether or not the interface board is being used with an 8X3ØØ controller board or a WD1Ø1Ø controller board. In addition to the interface modification, there is also a modification which needs to be made to the 8X3ØØ controller board. If you have a problem getting your interface board to talk to your controller board, refer to **Technical Bulletin HD:12** for a more in depth description of the modifications involved.

There exist several possiblities:

1) An unmodified interface will work (mostly) with an unmodified 8X3ØØ controller, and again mostly with a WD1Ø1Ø controller.

2) A modified interface will **NOT** function with an unmodified 8X3ØØ controller.

3) A modified 8X3ØØ controller will **NOT** function with an unmodified interface card.

Termination and Drive Select:

It is mandatory that proper hard drive termination and drive select jumpering be followed to the letter. Despite this fact, a great many people really don't understand how termination works.

.....

Termination is simple.

Always terminate the last drive on a cable.

.....

Always terminate the last drive on a cable. This means that if you have an internal WD1Ø1Ø controller, the primary and the secondary both are terminated (because they are on separate cable runs). Ok? If you have an external WD1Ø1Ø controller or an 8X3ØØ controller, then only the last secondary gets terminated because there is only one cable run.

If you don't terminate drives correctly, you can expect to see anything from various random system errors to smoke. **Don't expect any secondary hard drive to be shipped properly jumpered or terminated.**

The 8 Megabyte Hard Drive and Its Associated Interface Card

Did you read the previous section on 12/15/35/7Ø meg hard drives? If not, you should now, because the considerations for the 8 meg hard drive are very similar.

1) Possible Memory Conflicts:

As with the 5.25 inch drives, the 8 meg Interface card will need to be jumpered according to the machine in which it finds itself. The reasons are the same; possible memory contention at pages 14 and 15. The following is a good rule of thumb.

Model 12	AG - AL	selects pages 8 and 9
Model II/16/16B/6ØØØ	AK - AP	selects pages 14 and 15

Additional Interface Card Switch Settings:

S1	1, 3, 5, and 7 ON
S2	1, 3, 5, and 7 ON
S3	3 and 4 ON

Additional Interface Card Jumpers:

A - B
W - V

2) Drive Select Jumpering and Proper Termination:

The drive select jumpering and the placement of the resistor pack terminator are handled almost identically to the 12 meg. The only major difference is that **pin 6 of the terminating resistor pack MUST be lifted.** Refer to HD:5 for modification when pin 6 is lifted.

3) Potential 8 Meg Hard Drive Lockup Problems:

Unfortunately, one of the most unpleasant things about this drive is the tendency for 8 meg hard drives to be subject to disturbances created by RF fields. These problems show up in all kinds of freaky ways; however, the most common problem by far is the evil system lockup.

To minimize future agonies, you should therefore when working on any installation prcedure involving an 8 meg hard drive keep your eyes peeled for any potential sources of RF. Radio transmitters, x-ray machines, close proximity to telephone switching equipment and other computers may cause problems to system integrity.

.....

Don't blame Mr. Marconi for everything;
RF is not to be used as a catch-all
scapegoat for hardware problems.

.....

Be aware, though, that this potential problem and its cure is not all encompassing or to be used as a panacea for other hardware faults. Suspect RF interference only when all other methods of troubleshooting have failed. In other words, do the alignments, check for correct implementation of the applicable Technical Bulletins, and when all else fails consider RF as a possible culprit.

4) Magnetic Media Retention Problems:

This is as much a problem with System 3 as it was with Version 7, and the same basic truths about this ugly situation are true: it has high nuisance value, it can be rather difficult to diagnose, and it drives even the most pleasant customer off the deep end. Even though the System 3 (and Xenix 1.3.5) formatters are more rigorous, they will still miss many problems of this nature, since formatters only catch immediate problems and media retention difficulties may take weeks to manifest themselves. If you suspect this type of catastrophe, contact Technical Support for advice.

IMPORTANT HARD DRIVE TECHNICAL BULLETINS:

Technical Bulletins Associated with the AX-9282
Controller Board and AX-9367 Interface PCB
12 and 15 Meg External Primary Drives

HD:9	Loose cables on drive logic board
HD:1Ø	Resistor packs installed backwards on AX-9282
HD:12	Controller board circuit change (AX-9282)
HD:15	Capacitor change for VCO
HD:18	Missing jumpers on interface board (AX-9367)
HD:2Ø	Circuit change for the VCO (AX-9282)
HD:23	Write data pulse is not in center of timing window reducing data reliability (8X3ØØ AX-9282)
HD:24	Power supply current limiting modification
HD:28	Reduce occurrence of "Active Drive Not Ready"
HD:33	Correct error in artwork on "No Revision" boards
HD:37	Grounding differences in 35 meg primaries
INF:17	Voltage adjustment for power supplies

Technical Bulletins Associated With the AX-9432
Controller Board
Model 16B+ and 6ØØØHD with Internal 15 Meg

HD:16	Alignment procedure
HD:19	Improve interface signal integrity
HD:21	Acceptable WD1Ø1Ø and WD11ØØ combinations
HD:27	Correct drive select problems with external secondary on early revision rev PP2 boards
12/16B:25	To eliminate Boot Error HN and blown boot track on internal hard drives
6ØØØ:7	Bad AS parts on int. HDC
6ØØØ:14	Correct long reset pulse

Technical Bulletins Associated with the AX-9Ø36
Controller Board, AX-9ØØØ Shugart Bubble PCB
and the AX-9Ø35 Interface PCB
8 Meg External Hard Drives

HD:1	Correct switch settings and jumpers
HD:2	R3 is incorrect value
HD:3	Correct shipping procedure
HD:4	Modification of the controller board
HD:5	Write gate termination
HD:6	Filter on +5 volt line
HD:7	Mislabeled index connector
HD:8	Possible write protected drive after service
II:31	Missing jumpers on HD interface board

Technical Bulletins Associated with the AX-9454
External Controller Board (WD-1Ø1Ø)

HD:12	Proper termination of drive interface signals
HD:21	Acceptable WD1Ø1Ø and WD11ØØ combinations
HD:3Ø	Modification of wire connection point
HD:34	WAIT* modification
HD:35	Modification to allow secondary drives to power on
HD:36	PAL change for Western Digital controller

Miscellaneous Other Hard Drive Related Bulletins

HD:11	Differences between 5 and 12 meg bubble logic bds
HD:13	Decode Xenix hard errors
HD:25	Motor speed alignment and PCB differences
HD:26	Brake specifications
HD:29	Plated and oxide ID's
HD:31	Correct plated logic board connection
HD:32	Conversion of oxide to plated logic bd
HD:38	35 meg wiring points
HD:41	7Ø meg wiring points
HD:42	Using thinline 1Ø meg with 8X3ØØ controllers

Xenix is hard disk intensive. This means that the hard drives have to be operating correctly to insure the peaceful survival of a Xenix system. Make sure that you understand what you're doing when you check Xenix drives out... if you aren't sure, ask! You'll save yourself a lot of extra work.

Disk Cartridge System Hardware

1Ø and 2Ø meg external DCS systems and Interface:

Disk cartridge system support will vary with the version of Xenix which is used. Xenix 3.1.x will support **only** the 1Ø meg DCS; Xenix 3.2 will support either the 1Ø meg DCS or the 2Ø meg (external) DCS. Both systems use the same interface board.

There are a couple of important things to remember when you use a disk cartridge system:

(1) The interface is **terminated**! This means it must be **last** in the Z8Ø interrupt chain, if it is installed.

(2) Turn the DCS on **before** you turn the computer on. Otherwise, you may get some very strange behavior from the system.

Otherwise, these units are fairly easy to use. There are some important modifications to watch out for:

Disk Cartridge System Modifications:

6ØØØ:13	Prevent "BUGCHK:SCSIFI" errors
I/O:1ØØ	ROM replacement for secondary cartridge drive
I/O:1Ø4	Boschert power supply adjustment
I/O:11Ø	Astec power supply adjustment

****

SECTION THREE

Media Error Map Regeneration,
Multi-terminal PCB's and Terminals,
Two Editors,
On-Line Formatting, and
Other Pearls of Useful Wisdom.

****

So you've got a system with a history of intermittent errors, and the customer isn't exactly sure that they remember quite what the problems were? Fear not. Xenix takes care of that (up to a point) with a history table 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 just what errors you will find kind of depends on the version of Xenix you are working with. If the system has Xenix 3.Ø or 3.1.x, you will find that they consist primarily of disk-related errors and initial boot messages. There is a reason for this; in these versions of Xenix 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.

On the other hand, if you are running Xenix 3.2, you will find that just about **every** console error message, whether they crash the system or not, will show up in /usr/adm/messages. The reason for this is that there is a new device called /dev/recover which is devoted entirely to the idea of "recovering" error messages which may have happened when the system crashed. Obviously, this is a pretty powerful tool for a tech! With this version, unless the system is really in deep trouble, you should be able to see the exact text of errors as they appeared on the screen.

By reading this file, you can save yourself a lot of strain. Frequently a recurring error will point you in the direction of the problem subsystem. To read this file, type:

cat /usr/adm/messages

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

cat /usr/adm/messages ¦ lpr (where ¦ is produced by typing **<CTRL> <Ø>** on the console)

or

lpr /usr/adm/messages

This will send the file to the 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 following:

```
Jul 31 11:14
...
to the terms stated in the customer Non-Disclosure Agreement.

System 132k User 892k
Root 14356k Swap 1Ø28k

Jul 31 12:Ø1
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

Aug 1 Ø9:36
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 to diskutil and reformat the drive to lock out the 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.

.....

With **Xenix** 3.2, there another way to
"save" console error messages,
in addition to seeing them in /usr/adm/messages.

.....

Another option you have with Xenix 3.2, if you have the error on the console screen, is to make a "copy" of the console screen and print it out. **This will only work with Xenix 3.2!** Here's how you do it, using a temporary file called "/tmp/outfile" to hold the information until you print it:

cp /dev/screen /tmp/outfile<ENTER>
lpr /tmp/outfile<ENTER>

This is also quite helpful if a customer is struggling with a elusive problem at the console, and needs to be able to show it to you. Have them copy /dev/screen (which is one of the devices used to access the console CRT) into a file for either later perusal, or hardcopy. Sometimes the printout of the problem, or a file containing a copy, is a big help in grasping what the customer is talking about!

.....

Although they are few and far between,
there are a few nice surprises about
System 3.

One of them is
diskstat.

.....

One of the nice things that System 3 throws your way which simplifies matters is a utility called **diskstat.** This utility makes it very simple to check drives to see if all the tracks on the media error map which came with the drive were locked out. It also lists interesting information like the number of heads and cylinders the drive has, the size of the filesystem, the number of sectors per track, and so forth. To use it, all you do is type:

diskstat [drive number]

where [drive number] is Ø, 1, 2, or 3.

The default is drive Ø, which is in fact what it will look at if you just type "diskstat."

"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 100% accurate.** If at all possible, find the real map!

This procedure is valid for all hard drives currently available from Radio Shack.

1) If the drive is currently formatted under System 3, use the **diskstat** utility 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.** I mean it, gang... reformatting will make it much harder to find tracks which may have media retention problems.

5) Load and run the diagnostic selected in step 1. 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. It is unlikely that one pass, for example, is going to be very conclusive.

6) Select the history table option under the diagnostic and all errors will be displayed. From this information (and the information provided by **diskstat**) 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 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. Mind you, it's not terribly difficult... just not easy the first couple of times. Consider the following chart:

```
19console
13tty01
09tty02
||   |
||   |_____________ Port Name
||
||_________________ Speed
|                        0 = 300/1200/150/110 baud
|                        - = intended for online Model 33 teletype
|                        1 = Model 37 teletype (150 baud)
|                        2 = 9600 (for online terminals which require delays)
|                        3 = 1200/300 baud
|                        4 = LA36
|                        5 = 300/1200 baud
|                        6 = 2400 baud
|                        9 = 9600 baud (use with Radio Shack online terminals)
|
|___________________ Status (set by enable/disable)
                         1 = enabled
                         0 = disabled
```

As far as the speed setting goes, there are yet more choices. The ones listed above are special purpose; the next table gives a complete listing of the documented baud rates on Xenix (separate tables for Xenix 3.1 and 3.2):

Xenix 3.1.x Baud Rate Settings:

Special Purpose:

Ø = 3ØØ/12ØØ/15Ø/11Ø baud
- = intended for online Model 33 teletype
1 = Model 37 teletype (15Ø baud)
2 = 96ØØ baud (for online terminals which require delays like Tektronix 41Ø4)
3 = 12ØØ/3ØØ
4 = LA36
5 = 3ØØ/12ØØ
6 = 24ØØ
9 = 96ØØ

General Purpose:

a = 5Ø baud
b = 75 baud
c = 11Ø baud
d = 134.5 baud, usually with two stop bits
e = 15Ø baud
f = 2ØØ baud
g = 3ØØ baud
h = 6ØØ baud
i = 12ØØ baud
j = 18ØØ baud
k = 24ØØ baud
l = 48ØØ baud
m = 96ØØ baud
n = external baud rate "A", usually 192ØØ baud
o = external baud rate "B", often either 36ØØ or 72ØØ baud

Xenix 3.2 Baud Rate Settings:

Special Purpose:

Ø = 3ØØ/12ØØ/15Ø/1ØØ baud
- = Model 33 Teletype
1 = Model 37 Teletype (15Ø baud)
2 = 96ØØ baud with delays
3 = 12ØØ/3ØØ
4 = LA36
5 = 3ØØ/12ØØ
6 = 24ØØ baud
7 = 48ØØ baud
8 = 12ØØ baud
9 = 96ØØ baud

General Purpose:

a = 5Ø baud
b = 75 baud
c = 11Ø baud
d = 134.5 baud, usually with 2 stop bits
e = 15Ø baud
f = 2ØØ baud
g = 3ØØ baud
h = 6ØØ baud
i = 12ØØ baud
j = 18ØØ baud
k = 24ØØ baud
l = 48ØØ baud
m = 96ØØ baud
P = 3ØØ/12ØØ/24ØØ
Q = 12ØØ/24ØØ/3ØØ
R = 24ØØ/3ØØ/12ØØ

The information:

```
19console
13ttyØ1
Ø9ttyØ2
```

is contained within 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.

You will notice that the console setting looks different in System 3 than in Version 7. The baud rate for the console is now "9" instead of "h", which is good since "h" as a baud rate in System 3 is 6ØØ baud. In any case, "19console" means that the console is enabled (19console), it is communicating at 96ØØ baud (19console), and that the device name is console (19console).

"13ttyØ1" decodes to enabled (13ttyØ1), the system will check for 12ØØ or 3ØØ baud (13ttyØ1), and a device name of ttyØ1 (13ttyØ1).

"Ø9ttyØ2" decodes to disabled (Ø9ttyØ2), when enabled it will communicate at 96ØØ baud (Ø9ttyØ2), and a device name of ttyØ2 (Ø9ttyØ2).

To enable a channel, enter the following command:

enable ttyØn <ENTER>

where **n** is the channel number. For example, to enable channel 2 (otherwise known as serial port B) type: **enable ttyØ2 <ENTER>**.

.....

Serial Channel A is ttyØ1.
Serial Channel B is ttyØ2.
The computer's keyboard is console.

.....

Conversely, to disable a channel enter:

disable ttyØn <ENTER>

where **n** is the channel number. **disable ttyØ1** would disable serial port A.

.....

Unless You're Specifically Looking For Trouble

DO NOT DISABLE THE CONSOLE
(or change its baud rate).

.....

****

OK Guys, haul out the stale Twinkies.
You Are Now About To Learn (or relearn)

The Editors

(Yes, that's plural!)

(Lots of coffee, smokes if you do,
pain reliever of your choice 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 a Xenix editor. The "universal" Xenix editor is called **ed**. On Xenix 3.2, there is a more advanced, screen-oriented editor (available on 3.1.x if you have the Development System) called **vi**. We're going to learn the basics of both. Although you can get by knowing just **ed**, it's nice to learn **vi**, if only because it is a screen-oriented editor; it is much easier to work on large files in a screen-oriented environment than in a line-oriented one!

Let's take the "universal" one first...

ed is a fairly versatile, if somewhat moronic 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.

.....

ed is a versatile, if somewhat moronic 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 contained within the file, and an asterisk (*) as a prompt (a change from the Version 7 editor). 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
19console
Ø3ttyØ1
Ø3ttyØ2
Ø9ttyØ4
Ø9ttyØ5
Ø9ttyØ6
*
```

You will notice that the System 3 /etc/ttys file looks a little different than the Version 7 equivalent. System 3 comes pre-configured for six terminals instead of the three in the Version 7 file.

Let's assume (dangerously) that our sample customer has two DT-1 terminals. One DT-1 is to be run from a remote location over telephone 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 other DT-1 will be in the office and will run at a baud rate of 96ØØ.

We will configure ttyØ1 for 96ØØ baud and ttyØ2 to search for 3ØØ or 12ØØ baud. **It should be noted here that ttyØ1 and ttyØ2 can be configured to search a range of baud rates and select the one that matches the incoming data.** Reference the baud rate chart.

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

FROM	TO
19console	19console
Ø3ttyØ1	Ø9ttyØ1
Ø3ttyØ2	Ø3ttyØ2
Ø9ttyØ4	Ø9ttyØ4
Ø9ttyØ5	Ø9ttyØ5
Ø9ttyØ6	Ø9ttyØ6

NOTE: It is not wise to edit the /etc/ttys file if the file shows a "1" in the first position for the terminals to be changed. This implies that the terminal is enabled. It is wiser to quit, disable the terminals in question, then edit the file. Don't expect to be able to change the first character (enable or disable flag e.g. Ø or 1) and have this work. It won't! Use the **enable** or **disable** command at the system prompt instead. This forces the programs which deal with terminal communications to read the file, and correctly configure the ports in question.

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. (Actually, you can get **ed** to tell you what line you're on, but that is a command which is easy to forget. You can always remember how to count lines!)

Type the following command:

2p <ENTER>

You should see displayed on the screen the second line of the file. 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/Ø3/Ø9/p <ENTER>

You should now see on your screen:

Ø9ttyØ1

The above command instructed **ed** to get line 2 (2s/Ø3/Ø9/p), search it for the first occurrence of Ø3 (2s/Ø3/Ø9/p), change the Ø3 to a Ø9 (2s/Ø3/Ø9/p), and finally to print the line as changed (2s/Ø3/Ø9/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 had intended.

Now, to make sure that the changes are absolutely correct, list the entire file again:

1,$p <ENTER>

If it is correct, then you must write the file back 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 System Administrator's Guide to Xenix, chapter 11. In addition, Xenix 3.1 has a help file on **ed** which also contains useful information. To use it (provided it has 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.

In case you don't (somehow) have any of the above options available to you for further reference, here is a quick guide to **ed** commands:

Common ed Commands:

a	append text (use a . on a line by itself to exit this mode. If a line number is given before the **a**, text will be appended after the line.
c	change text (use a . on a line by itself to exit this mode). If a number is given before the **c**, the line specified will be where the change starts.
d	delete. If a line or a range of lines is given, like **1d** or **1,3d**, those lines will be deleted.
i	insert text (use a . on a line by itself to exit this mode). If a line number is given before the **i**, text will be inserted before the line.
s/searchstring/replacementstring/ **s/searchstring/replacementstring/g**	The first form of this command says to replace the first occurence of the search string with the replacement string. The second form of this command says to replace all occurences. Both forms may be used with a line number, or a range of lines.
=	Tells you what the current line number is.
w	Write the file out.
p	Print the current line. If used with a range of line numbers, prints the lines requested.
q	Quit. If you wish to quit without writing, enter this command twice.

One editor down, one to go -- now, let's talk about **vi**.

.....

Vi is kind of the Rolls-Royce of
Xenix editors.
...

Once you're used to **vi**, **ed** is going to
look like a '71 Pinto in comparison.
.....

Instead of being a line-oriented editor like **ed**, **vi** is a "screen-oriented" editor. What this means is that you are going to see a screenful of text at a time, instead of working with things a line at a time. Now, for a small file like /etc/ttys, this difference won't seem very impressive... but if you are trying to write anything fairly lengthy, you quickly appreciate the beauties of this approach. Take it from me, trying to wade through several hundred lines of text with **ed** is about as much fun as a having a root canal done with no anesthetic!

Vi is also a fairly intelligent editor, using terminal specific codes to do things like cursor positioning... which means that Xenix has to have a correct notion about what kind of terminal you have. Not to worry, though, that's easy to check! Also, if you should ever use **vi** on a terminal running at lower baud rates, you will notice that the "window" of text you start up with is smaller the lower the baud rate. The reason for this is so that redrawing the screen will (hopefully) take only a reasonable amount of time. Obviously, you can redraw more lines in the same amount of time at 12ØØ baud than, say, at 3ØØ.

Ok, let's get down to the problem at hand. If you are at the console, getting ready to edit the /etc/ttys file with **vi**, you need to verify that Xenix knows what type of terminal you've got. To check this out, type:

echo $TERM<ENTER>

By the way, the "TERM" is capitalized here! On the console, you should see the system respond with:

trs16

If you **don't** see that, type this:

TERM=trs16;export TERM<ENTER>

This tells Xenix the correct terminal type for a System 3 console. If you were using **vi** from a terminal, you could substitute your terminal type for "trs16" in the above command.

.....

To find out what terminal Xenix thinks you have,
type:

echo $TERM<ENTER>

To set your terminal type, use:

TERM=[terminal type goes here];export TERM<ENTER>

where the name of your terminal type
is substituted for "[terminal type goes here]".

.....

Now, to actually edit the file, type:

vi /etc/ttys<ENTER>

You will see something which looks like this:

```
19console
Ø3ttyØ1
Ø3ttyØ2
Ø9ttyØ4
Ø9ttyØ5
Ø9ttyØ6
~
~
~
~
~
"/etc/ttys" 6 lines, 5Ø characters
```

Your cursor will be under the "1" in the first line -- that's why there is a underline under the "1" above. Also, you'll have a slightly larger screen display... I've left out some of the empty lines (denoted by the "~" on a line by itself) to save space.

Before we make any changes, let me point out that vi has two major operating modes. One is called **command mode**, and that's the mode you are in when you start up. You can do things like move around, delete text, write the file out, and quit in command mode. The other mode is the **edit mode** (sometimes called **insert mode**) and this mode is the one which allows you to insert, append, or change text.

Let's move down to the line that needs to be changed. The cursor movement keys are:

Key	Movement
j or <DOWN ARROW>	down one line
k or <UP ARROW>	up one line
l or <RIGHT ARROW>	right one character
h or <LEFT ARROW>	left one character

We want to move down one line so that the cursor is at the front of the line which has the "tty01" entry. So, press j or **<DOWN ARROW>** once. You should now have:

```
19console
03tty01                 <--- see, now your cursor is down one line
03tty02
09tty04
09tty05
09tty06
~
~
~
~
~
"/etc/ttys" 6 lines, 50 characters
```

However, we don't want to change the first character in that line, but the second, so press the 1 key or the **<RIGHT ARROW>** once. That should move you like this:

```
19console
03tty01                 <--- the cursor has moved one position to the
03tty02                      right.
09tty04
09tty05
09tty06
~
~
~
~
~
"/etc/ttys" 6 lines, 50 characters
```

Now we're at the character we want to change. To change the character, we will have to go into the edit mode, and we will be substituting text. To do so, type:

```
s
```

When you press the s key, a "$" character will appear at your cursor position.

```
19console
Ø$ttyØ1
Ø3ttyØ2
Ø9ttyØ4
Ø9ttyØ5
Ø9ttyØ6
~
~
~
~
~
"/etc/ttys" 6 lines, 5Ø characters
```

Now, type a **9**, and press **<ESCAPE>**. This will change your ttyØ1 baud rate to a 9 (or 96ØØ), and get you out of the edit mode. You should now see:

```
19console
Ø9ttyØ1                 <--- your cursor is under the new "9"
Ø3ttyØ2
Ø9ttyØ4
Ø9ttyØ5
Ø9ttyØ6
~
~
~
~
~
"/etc/ttys" 6 lines, 5Ø characters
```

Now, all you have to do is write the file out, and quit. Do that by typing:

:wq<ENTER>

This will write your file out, and get you out of the editor.

This is only the barest minimum on how to use **vi**. Like **ed**, more help can be obtained by reading the manual page in the owner's documentation. However, for reference let me give you a quick list of **vi** commands:

Common vi Commands

Command	Function
j or <DOWN ARROW>	down one line
k or <UP ARROW>	up one line
l or <RIGHT ARROW>	right one line
h or <LEFT ARROW>	left one line
w	forward one word
b	back one word
Ø	beginning of line
$	end of line
nG	go to a specified line, where n is the line number
G	go to the last line
H	go to upper left screen corner
M	go to middle line on screen
L	go to bottom line on screen
<CTL><d>	down one half screen
<CTL><u>	up one half screen
<CTL><f>	down one screen
<CTL><b>	back one screen
dd	delete current line of text
dw	delete current word
x	delete current character
i	insert text starting at cursor (use <ESC> to exit)
a	append text after cursor (use <ESC> to exit)
s	substitute text, starting at cursor (use <ESC> to exit)
:1,$s/searchstring/replacestring/g	global search and replace
:wq	write file out and quit
:q!	quit without writing the file

Multi-Terminal Boards:

HARDWARE:

The multi-terminal interface board (AX-7981) will allow the user to add three additional serial ports to his computer with each PCB. Up to two multi-term boards are now supported under Xenix 3.Ø , 3.1.x, and 3.2, giving the user a maximum of 8 serial channels plus the console.

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) Card Order:

The multi-terminal board is interrupt driven and interfaces to the Z8Ø side of the machine. Due to Z8Ø interrupt chaining, the position of this board in the scheme of things becomes important. Refer to Section 2 for proper card order.

2) Switch Setting:

There is one switch on the multi-terminal board that must be set prior to use. This switch determines which ports a particular board will be responsible for.

Board 1

Switch position one in the on condition (closed). All other positions are in the off (open) condition.
This board will provide ports ttyØ4, ttyØ5, and ttyØ6.

Board 2

Switch position two in the on condition (closed). All other positions are in the off (open) condition.
This board will provide ports ttyØ7, ttyØ8, and ttyØ9.

3) Technical Bulletins:

Refer to the following Technical Bulletins:

I/O:62	To correct intermittent operation and to increase reliability.
I/O:1Ø1	Multiterminal board modifications for use with DT1ØØ's

SOFTWARE:

Xenix 3.Ø, 3.1.x, and 3.2 come equipped to deal with the ports on one multi-term board. If you wish to run two in a system, simply setting the switches and putting the board in the proper slot is not sufficient. **Xenix has to be told that the boards are there!**

To tell the system to look for the additional ports, follow the steps outlined below:

1) You must be logged in as **root** to effect these changes.

2) Change to the **/dev** directory:

cd /dev <ENTER>

3) Output a directory listing of the current TTY's to the screen:

1 tty* <ENTER> list only files beginning with "tty"

4) You will see listed **ttyØ1** and **ttyØ2**. For one interface board look for **ttyØ4, ttyØ5,** and **ttyØ6**. In System 3, these five should be present automatically. For the second interface board, look for **ttyØ7, ttyØ8,** and **ttyØ9**. If Ø4, Ø5 and Ø6 appear (one board) and Ø4, Ø5, Ø6, Ø7, Ø8, and Ø9 appear for two boards then skip to step 6; otherwise, continue with step 5.

5) You will need to 'make the nodes' for these tty's. Essentially this means telling Xenix that these devices are available and linking said devices to their drivers. Enter the following:

One Multi-Terminal Board:

```
mknod ttyØ4 c Ø 4 <ENTER>
mknod ttyØ5 c Ø 5 <ENTER>
mknod ttyØ6 c Ø 6 <ENTER>
chmod 666 ttyØ4 ttyØ5 ttyØ6 <ENTER>
```

NOTE: These ports should already be in the system; this is provided primarily for reference purposes.

Two Multi-Terminal Boards:

```
mknod ttyØ7 c Ø 7 <ENTER>
mknod ttyØ8 c Ø 8 <ENTER>
mknod ttyØ9 c Ø 9 <ENTER>
chmod 666 ttyØ7 ttyØ8 ttyØ9 <ENTER>
```

6) The 'nodes' being installed, two other files must be updated. These are the **/etc/ttys** file and the **/etc/ttytype** file.

To update these files, an editor will need 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. (If you've got **vi**, you can use that if you want). Once you feel confident enough to edit these files, come back to step 7.

7) Enter the command:

```
cat /etc/ttys <ENTER>
```

You should see **at least** the following in the file:

```
19console
ØXttyØ1
ØXttyØ2
```

With one multi-terminal interface the file must contain:

```
19console
ØXttyØ1
ØXttyØ2
ØXttyØ4  ------------\
ØXttyØ5               |---- Add these lines if not there.
ØXttyØ6  ------------/
```

With two multi-terminal interfaces, the file must contain:

```
19console
ØXttyØ1
ØXttyØ2
ØXttyØ4  ------------\
ØXttyØ5               :
ØXttyØ6               \_____ Add these lines if not there.
ØXttyØ7               /
ØXttyØ8               :
ØXttyØ9  ------------/
!!
!!_____________________________ Set the baud rate here.
!
!_____________________________ Ø or 1 (enabled = 1, disabled = Ø)
```

8) The second file to edit is the **/etc/ttytype** file. This file contains information as 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 the following:

```
trs16 console
adds25 ttyØ1
adds25 ttyØ2
adds25 ttyØ4
adds25 ttyØ5
adds25 ttyØ6
```

The file will need to be changed as follows:

```
trs16 console
adds25 ttyØ1
adds25 ttyØ2
adds25 ttyØ4 --------------\
adds25 ttyØ5                !----- add for first board if not present
adds25 ttyØ6 --------------/
adds25 ttyØ7 --------------\
adds25 ttyØ8                !----- add for second board
adds25 ttyØ9 --------------/
  !
  !_____________________________ This field may change depending upon
                                  the terminal type connected to its
                                  respective port.
```

NOTE: In System 3, for the last terminal (ttyØ6 or ttyØ9) to reinitialize correctly after logging out (i.e. clear the screen and put the login message back up), there needs to be a **blank line** at the end of both the /etc/ttys and the /etc/ttytype file.

****

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

****

So, your customer just hauled himself down to the neighborhood Computer Center and, while he was laying down the bucks for his new System 3 software, decided that he would buy a brand-new terminal to boot... so he put forth some more hard cold cash and purchased a shiny new DT1ØØ. Roaring back into his office with single-minded enthusiasm, he rips open the box with the insane zest of a toy poodle destroying a pair of slippers. A veritable blizzard of packing material results, 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 reflection, our stalwart customer goes boldly forth and connects his new prize to the system. Striding purposefully across the room, he logs into the console and configures the baud rate of the channel with the editor of his choice, and then enables the port. Leaping back to his newest acquisition as though his life was threatened by Darth Vader himself, or at least a Storm Trooper, he breathlessly logs in and invokes his favorite application package. A pregnant pause, then, "Funny, that's not what Profile's supposed to look like...."

Not really 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 an **adds25**. This means that whenever you log in on, say for example tty02, Xenix is going to configure its output to be compatible with an Adds-25 terminal, or one emulating an Adds-25 terminal.

Actually this is something of an oversimplification. In point of fact, Xenix just doesn't care what terminal type you have, but some application packages (Scripsit-16, Profile-16, Xview, etc.) 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. Another example of a Xenix utility which needs this information is **vi**.

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? Profile-16 thinks he's using an Adds-25 terminal when he's really using a DT100. 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 DT100 hooked to and then edits the **/etc/ttytype** file to say **dt100** instead of **adds25**.

We decided that in the example the customer is using tty02. Let's look at the file **/etc/ttytype**.

```
trs16 console
adds25 tty01
adds25 tty02
adds25 tty04
adds25 tty05
etc. ....
```

The file needs to be edited, again using the editor of choice, to look like:

```
trs16 console
adds25 tty01
dt100 tty02
adds25 tty04
adds25 tty05
etc. ....
```

Now, whenever anyone logs in on tty02, Xenix and the application packages will know to configure their output, when necessary, to correctly "speak dt100". (As a note, this won't take effect for the current login; you will need to log out and log back in for this change to be evident).

There's Another Small Problem...

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 dt100, 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.

System 3 Xenix has a DT100 entry already present in the **/etc/termcap** file. However, sometimes an older version of this file can get restored in an upgrade, or the entry needs to be modified for certain applications. In these cases, an "install" diskette with the correct DT100 termcap should be available from your local software support personnel to make the necessary changes to the **/etc/termcap** file. Some application programs, Profile-16, Scripsit-16, Unify, and MultiPlan as examples, will have their own termcap file. Their install diskettes, in conjunction with this one, will take care of them also.

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

An Easy Way To Check For The Problem

Assuming that the DT100 "Install" diskette has indeed been installed correctly (I'll explain what I mean by "correctly" in just a moment) but the output still looks funny, try the following command:

TERM=dt100; export TERM <ENTER>

This command should look familiar; we just used it to check if the terminal type was set correctly before entering **vi**. Now try Profile, or Scripsit, or **more** and see if the output looks correct. If it does, then 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 trs16 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.

****

A Confusing Compendium of
Application Packages
and
How To Install Them

****

Since we just finished a discussion of DT1ØØ's and why they might occasionally look funny, let's go on ahead and add to the confusion by discussing some of the commonly used applications software and their relationship with System 3 and terminals.

To begin with, let's do a quick review of how to install an applications package:

1) At the root prompt, type:

   install <ENTER>

2) The screen should show the following:

   Installation Menu
   1. to install
   q. to quit

   Please select :

3) Type 1 **<ENTER>** to select the Install option. The screen shows:

   Insert diskette in Drive Ø and press <ENTER>

4) Insert the first installation diskette in Drive Ø, close the drive door, and press **<ENTER>**. If the application is complete on one diskette, the screen shows:

```
Installation complete -- Remove the diskette, then
press <ENTER>
```

If the application is contained on more than 1 diskette, Xenix tells you when to insert the remaining diskettes and then displays the above message when the installation is complete.

5) After you press **<ENTER>**, the screen shows the Installation Menu again. If you want to install another application package, type 1 again; otherwise, type **q <ENTER>** to quit. You should get the root prompt **#** back.

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

install must be run by root at the console.
.........................................
```

Now, as if matters weren't already complicated enough, there are specific applications packages and versions of same which have interesting quirks waiting to grab you regarding DT1ØØ's.

Multiplan, Unify, and Scripsit-16 with DT1ØØ's

If you are using DT1ØØ's and the following versions of these applications, you must install a special DT1ØØ file after you install the application. (Nothing is ever easy!) The versions are:

Multiplan Version Ø1.Ø6.ØØ (Cat. No. 26-648Ø)
Unify Version Ø1.ØØ.ØØ (Cat. No. 26-6415)
Scripsit-16 Version Ø1.ØØ.ØØ and Ø1.ØØ.Ø1 (Cat. No. 26-6431)

After you install the application, at the root prompt, type:

dt1ØØ-install <ENTER>

The screen shows:

```
        DT1ØØ Terminal
     Installation Menu
        1.  to install
        q.  to quit

  Please select :
```

Now, type 1 **<ENTER>**. When prompted, insert the DT1ØØ installation diskette into Drive Ø and type **<ENTER>**.

All later versions of these particular applications packages do not require any modification.

Profile-16

With System 3, you must use Version Ø1.Ø1.ØØ (or later) of Profile-16 (Cat. No. 26-6412).

Accounting Applications (GL, AR, AP, etc.)

Apparently, the Version 7 and System 3 versions will both work on Xenix 3.Ø, 3.1.x, and 3.2, but beware of mixing versions... the programs work best if you use all new versions or all old versions.

Hopefully, you should never have to deal with the nuts and bolts of mismatched applications packages; it really is going somewhat above and beyond the call... but this should give you enough information to spot the obvious pitfalls.

****

Accessing Additional Hard Drives
With Xenix
and
Neat Things About Cartridge Drives

****

Before you can access any additional secondary hard drives with Xenix, several steps must be taken... and these steps pretty much equally apply to cartridge and floppy drives as well. Briefly, the drive(s) must have been formatted with **diskutil**, a file system must have been created on each additional hard drive (and, in some instances, on floppy/cartridge drives) with **/etc/makefs**, and the drive(s) must be **mounted**. This procedure is really much easier than it sounds, and the steps will be discussed below.

Formatting:

Formatting has gotten both easier and more complicated with the advent of Xenix 3.2. If you want to format either a floppy disk or a cartridge drive **while Xenix is running**, you can now do so using the **format** command. However, hard drives must still be formatted with **diskutil**. (You can still format your floppies and cartridges with **diskutil**, too).

To format anything under Xenix 3.Ø or 3.1.x, be it hard, floppy or cartridge drive, the program **diskutil** must be used. The system is booted from the floppy **Boot diskette** or the hard drive and at the boot prompt the program **diskutil** is called. This procedure is the same as when formatting the primary hard drive with the exception that the drive number will change to correspond to whichever secondary you may wish. You are directed to Section One of this document for formatting instructions.

Creating the File System:

Once all secondary hard drives are formatted and the system has been booted from the primary hard drive, you are ready to begin creating file systems. Follow the steps below:

NOTE: Since we're at this, I'll also cover here how to make a file system on both floppy and cartridge disks. If all you ever plan to do with floppy or cartridge drives is backup your hard drive(s), you can skip this information; if you plan to use either floppy or hard drives as **mount**able file systems, you have to perform the steps outlined below.

Creating Hard Disk Secondary File Systems:

1) At the root prompt, type:

   /etc/makefs <ENTER>

2) The screen shows the following menu:

Xenix 3.Ø and 3.1.x
(3.Ø has no disk cartridge choice!)

```
      Make a File System on . . .
      ---------------------------
Type  1 a single-sided floppy
      2 a double-sided floppy
      3 a hard disk
      4 a disk cartridge
      q to quit

 > Enter Number:
```

Xenix 3.2

```
Make a File System on...
------------------------
1  a single-sided floppy
2  a double-sided floppy
3  a hard disk
4  a disk cartridge

q  to quit

------------------------

> Select option:
```

3) Type 3 **<ENTER>**. The screen shows:

   On which hard drive (1-3)?

4) Type 1 **<ENTER>** for the first secondary drive (drive 1). The screen shows the following:

   Disk has xxx cylinders.
   Disk has x heads.
   Disk has xx tracks reserved for bad track mapping.
   Making a file system of xxxxx blocks on /dev/hd1.

5) When the file system is created, the screen shows:

   Press <ENTER> to continue.

6) Type **<ENTER>**, and the menu returns to the screen. At the menu, type q **<ENTER>** and the root prompt will be displayed.

Creating File Systems on Cartridge:

1) At the root prompt, type:

    /etc/makefs <ENTER>

2) At the menu, type **4 <ENTER>**. This is the option to make a file system on a disk cartridge. The screen will display:

    On which cartridge Drive (Ø or 1)?

3) Type **Ø <ENTER>**. The screen shows:

    Xenix 3.1.x:

    Insert formatted cartridge in Drive Ø and press
    <ENTER>

    Xenix 3.2:

    Do you wish to format a cartridge? (Y/N):

You can answer "Y" to the 3.2 format question if you need to format the cartridge. If not, answer "N". If you respond with a "Y", you will see:

    About to format a cartridge in Cartridge Drive Ø
    Warning! Any existing data will be destroyed.
    Press <BREAK> to prevent this, or insert a disk
    cartridge in Drive Ø and press ENTER to continue:

Place a cartridge in the drive, and press **<ENTER>**. The system will display status messages about formatting the cartridge and making the file system.

If you answered no, the screen will say:

    About to make a file system on Cartridge Drive Ø
    Warning! Any existing data will be destroyed.
    Press <BREAK> to prevent this, or insert a formatted disk
    cartridge in Drive Ø and press ENTER to continue:

Place a formatted disk cartridge in the drive, and press **<ENTER>**. The system will display status messages about formatting the cartridge and making the file system.

4) 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 reappear.

Creating File Systems on Floppy:

1) At the root prompt, type:

    /etc/makefs <ENTER>

2) At the menu, select either option 1 or 2, depending on whether you wish to make a file system on a single or a double sided disk. The screen will then show:

    On which floppy Drive (Ø - 3)?

3) Type **Ø <ENTER>**. The screen shows:

    Insert formatted disk in Drive Ø and press <ENTER>

4) When the file system is created, the screen shows:

    Press <ENTER> to continue

    The menu reappears on the screen. Type q **<ENTER>** to quit.

Mounting File Systems:

In order to access files located within a secondary hard drive, a mountable floppy disk, or a mountable cartridge disk (i.e. drives which have file systems), the drive in question must first be mounted on the root file system. Although Xenix provides empty directories for this purpose (with names like **/mnt**, **/mnt1**, **/mnt2**, and so on) 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 h1 <ENTER> (for the first secondary hard drive)

The first command (**cd /**) moves you to the root directory. The second tells Xenix to create a directory entry called **h1**. If you also wanted to mount a third hard drive and your first disk cartridge, then you would want to use **mkdir** to make a directory for each of these file systems (calling them perhaps **h2** and cdØ). In truth, you can call these directories pretty much anything you want, but h1, h2, h3, etc. 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/hd1 /h1 <ENTER> (first secondary hard drive)

/etc/mount /dev/hd2 /h2 <ENTER> (second secondary hard drive)

/etc/mount /dev/hd1 /h3 <ENTER> (third secondary hard drive)

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

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

Remember, you have to mount these devices onto empty directories that exist! If the directory isn't there (and /h1, /h2, /h3, /cdØ, and /cd1 usually don't) you will have to make it with the **mkdir** command.

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/hd1 <ENTER>

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

If additional hard drives are on the system, or cartridge or floppy drives, substitute for **hd1** in the above command from the following list as needed:

hd2, hd3, cdØ, cd1, fdØ, fd1, fd2, fd3

A Note About Disk Cartridges:

Xenix locks disk cartridge drives when they are in use. This is to prevent anyone from removing a disk while it is in motion. When the drive is no longer in use, Xenix unlocks the drive so the cartridge can be removed.

If the cartridge is being used as a mountable filesystem, Xenix locks the drive until the filesystem is unmounted. **NOTE: If you improperly shutdown the system, or if it crashes with the cartridge file system in place and mounted, Xenix does not unlock the drive.** The cartridge drive will remain locked until you press the reset switch and see the "Xenix Boot>" prompt. At that point, Xenix will unlock the drive for you.

Under Xenix 3.2, the system will automatically determine the size of the drive (whether it is a 10 meg or a 20 meg disk cartridge) and create the correctly sized file system accordingly.

Remember that Xenix 3.1.x can use only the 10 meg DCS!

The format Command (Xenix 3.2 only)

Xenix 3.2 allows you to do online formatting with the **format** command. What this means is that you can now format floppy and cartridge disks **while Xenix is running**. This is a big change from earlier versions! You can also, of course, still format diskettes using the stand-alone **diskutil** program.

To use this command to format cartridge or floppy disks, place the diskette to be formatted in the target drive first, and then type:

format <device> <ENTER>

where you substitute for **<device>** one of the following:

fd0
fd1
fd2
fd3
cd0
cd1

Xenix will determine if the floppy disk to be formatted is single or double sided, and the size of the disk cartridge. Any user can run **format**; in other words, you don't have to be root to format a disk. (You do, however, have to be root to use **makefs**).

.....

If you are formatting a bulk-erased floppy disk,
the **format** program will throw a hard error
and then format the disk.

Don't worry... that's normal.
The reason is that **format** first
tries to read the disk before formatting
(to see if it's going to mung anything!).

If there's data (i.e. a good format), you will be warned.
If there isn't, since Xenix can't read the disk,
there will be a hard error....
but **format** knows to go ahead.

.....

****

Remember....

NEVER BULK-ERASE A CARTRIDGE DISK!!!

****

****

SECTION FOUR

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 are 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, 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" filesystem 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 maybe be able to find something later. In Xenix, before you can hope to find anything, you have to start with a neat system.

Sometimes, for a plethora of reasons, a file system on a device may become corrupted, or unclean. When booting, Xenix will check the primary hard drive to insure that the file structure is intact and if not, the system will ask for permission to clean it. The "gotcha" is that the system will not automatically clean the secondaries (if installed) and this is what frequently causes some customer/technician confusion. Consider the following example.

> A particular Xenix installation contains two twelve meg hard drives, one primary and one secondary. Upon booting the system the secondary is automatically mounted by way of the /etc/rc.user file. The customer, under normal operating conditions, should never need to manually mount the secondary.
>
> Upon booting the system one day, 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, boots the system.

Later, the customer tries to access the information resident on the secondary only to find 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 is that only the primary hard drive was cleaned, not the secondary. Fortunately this is an easy problem to cure. Follow the steps listed below:

1) **fsck /dev/hd1 <ENTER>**
   hd1 is for the first secondary hard drive
   hd2 is for the second secondary
   hd3 is for the third secondary

2) mount the secondary 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 floppy drive Ø
   fd1 is for 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

Both are equally taxing if you haven't been up against them before. We will take the first problem first and for the sake of argument assume that the hardware is flawless. Normally all line printer jobs are spooled into a holding area as separate files to await their turn to be printed. This holding area is in the **/usr/spool/lpd** directory. If there is ever a printer fault the system will add another file to this directory called **lock** to inhibit spooling any further jobs to a dead line printer. (Actually, **lock** is also present when the system is printing normally, to tell any new copies of the line printer daemon that a daemon is already running, but that adds to the confusion...). Anyway, to check for this problem, consider the following steps:

1) Log in as root.
2) **cd /usr/spool/lpd <ENTER>**
3) **l <ENTER>**
4) Look for a file named **lock**. If it is there continue, otherwise you may have a hardware problem.
5) **ps -elf <ENTER>**
6) Look for a pathname in the column labeled "CMD" containing "lpd". Once you have found it, look to the left under the column labeled "PID" and note the number. This is the process ID.
7) **kill -9 PID <ENTER>**
   This command kills the line printer process.
8) **rm /usr/spool/lpd/lock <ENTER>**
   This command removes the lock file.
9) **/usr/lib/lpd <ENTER>**
   This command reinitializes the printer.
10) **cd / <ENTER>**
    l ¦ lpr <ENTER>
    Step 9 should print a listing of the root (/) directory on the line printer. If it doesn't there may be a hardware problem. Do a shutdown and boot with TRSDOS and try the printer again. If it still doesn't work... you **definitely** have a hardware problem.

Once in a while a user will queue up a massive printer job that he really doesn't want. To abort the process:

1) Take the printer off-line, and follow steps 5, 6, and 7 above.
2) **cd /usr/spool/lpd <ENTER>**
3) **pwd <ENTER>**
   Make sure the system displays **/usr/spool/lpd** before step 4.
4) **rm * <ENTER>**
   This command removes all files in the line printer queue.
5) **/usr/lib/lpd <ENTER>**

Now, these techniques will work for all System 3 Xenix systems, but there are a couple of new utilities for Xenix 3.2 which you may be able to use in some situations instead of these methods.

Xenix 3.2 Line Printer Management Utilities:

In the 3.2 version of Xenix, there are a couple of new utilities to help you manage the line printer system. What they do is tell you what's queued up to print, and allow you to remove spooled print jobs by user, job id number, or filename. These utilities are called **lpq** and **lprm**.

.....

lpq tells you what's in the print queue, and
lprm lets you remove jobs.

.....

The real beauty of this is that if a user spools up something and wants to get rid of it, he doesn't have to enlist the help of the root user.

Let's say that a user (named **mike**) queued up two print jobs that he later decided that he didn't really want to print. He could look at what was currently happening in the spooled printing system by typing:

lpq <ENTER>

He would then see something like this:

Name	Id	Chars	Filename
root	Ø4894	5Ø	/usr/spool/lpd/lfAØ4894
susan	Ø4896	7396	/usr/spool/lpd/lfAØ4896
mike	Ø4898	275	/usr/spool/lpd/lfAØ4898
mike	Ø4899	78	/usr/spool/lpd/lfAØ4899

The jobs are listed in the order in which they will print, so the job on top (owned by **root**) is the one which is currently printing.

As long as the job you want to get rid of hasn't started to print, you can get rid of it using the **lprm** command. There are three ways you can use **lprm** to get rid of print jobs:

(1) You can kill all jobs belonging to a user.

(2) You can kill jobs by their ID numbers. The ID number is the one which appears under the column labelled "Id" in the **lpq** output.

(3) You can kill jobs by their filenames. The filename to be used for this is the one which appears under the "Filename" column in the **lpq** output.

Well, our friend Mike notes the fact that his jobs aren't printing yet. He wants to kill all his print jobs, so he opts to use **lprm** to kill all the print jobs belonging to him. To do this, he types:

lprm mike<ENTER> <-- removes all print jobs belonging to this user

The system responds with:

```
removing Ø4898
removing Ø4899
```

In other words, **lprm** will tell you the ID of the print jobs it is removing.

Similarly, Mike could have opted to kill the jobs by their ID's, or by their filenames. To do that, he would have typed the following:

lprm Ø4898 Ø4899<ENTER> <-- removes these particular jobs by ID number

lprm /usr/spool/lpd/lfAØ4898 /usr/spool/lpd/lfAØ4899<ENTER>
→
¦__ removes these jobs by filename

These utilities are kind of neat, since they let users manage their printouts a little more easily. There are a couple of things to bear in mind, though:

- o **lprm** will only let users remove their own print jobs. To get rid of anyone else's, you must be root!
- o If the job has started to print, even though **lprm** will say that the job has been removed, it will continue to print! This is because the line printer daemon, **/usr/lib/lpd**, buffers the information to be printed for the current job in memory. To kill something which is currently printing, take the printer off line, log in as root, and follow the procedure for "Line Printers That Won't Stop Printing".

While on the subject of Xenix 3.2 line printer problems, let me address one more:

.....

Xenix 3.2 and Slow Daisy Wheel II Printing...

.....

Let's say you've got a customer who's just upgraded his system to Xenix 3.2, and everything is fine until he goes to print... then, he calls you with a worried note in his voice. "My printer seems awfully slow now," he says, "could you fix it, please?"

Upon questioning him further, you find out that he has a Daisy Wheel II. This means that you can fix it, but first, let me tell you what's going on.

Until Xenix 3.2, printing in Xenix was conducted in a "polled" mode. This means that the system would send something to the printer, then go back and check if the printer was still busy, and if not send it some more data. With Xenix 3.2, this mode is still supported, but the system doesn't poll the printer as often, so if the printer is capable only of "polled" printing instead of "interrupt-driven" printing (using the BUSY* and ACK* lines to let the system know the printer is ready), you end up printing slowly.

In the case of the DWII, there is a modification to allow the printer to function in the "interrupt-driven" mode. If you perform I/O:111 on this DWII, the performance of the printer should be back up to speed... (no pun intended!)

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, the date of installation, and their respective version numbers. (As a note, Tandy applications update the logbook file... but there's no guarantee that other vendors' products will!)

.....

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 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.

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

Another reason that an application installation might fail is that the Install diskette for the application might be "dirty". This is kind of similiar to the reason why secondary hard drives won't automatically mount after a crash... if the disk is shutdown improperly, you can't mount it.

"So what?", I hear you ask. Well, the fact is that "Install" disks are file systems, too. The **install** program mounts the floppy disk on the directory **/mnt**, and then looks for information on that disk that will tell it how to install that particular package. If the floppy disk filesystem is in an improperly shutdown state, **install** goes nowhere fast.

.....

If you get an error during **install** which says:

"mount: mount failed
mount: structure needs cleaning"

suspect a "dirty" Install disk.

.....

To fix this problem, the ideal solution is to get another copy of the application or Install diskette. This is because of the possibility that cleaning the disk may cause some loss of data... and on a small file system like a Install disk, you are almost certain to need everything on the disk.

However, if you can't get another disk, you need to clean the one you've got. Remember the **fsck** command? Here's where you get to use it again. Put the diskette to be cleaned in floppy drive Ø, and type:

fsck /dev/fdØ<ENTER>

After this has been completed, then you can retry the installation.

Another possible cause of a failure to reinstall an application may be the fact that something is in the way.

.....

If you get the following during **install**:

"mount: mount failed
mount: Device busy"

probably there is another filesystem already mounted on the **/mnt** directory!

.....

To check this out, type:

/etc/mount<ENTER>

The **mount** command, when used by itself, will tell you what filesystems are currently "mounted". If you see something like this:

```
/dev/root on / read/write on Tue Nov 17 17:15:28 1987
/dev/cdØ on /mnt read/write on Fri Nov 2Ø Ø8:54:44 1987
```

Your problem is that another filesystem is currently mounted on the **/mnt** directory, which the **install** program uses to mount diskettes. To fix this, you must unmount the device which is in the way. In our example here, the device to be reckoned with is **/dev/cdØ**. To fix this problem, type:

/etc/umount /dev/cdØ<ENTER>

If the root prompt **#** returns with no messages, then the drive should have been correctly unmounted. To check it, type:

/etc/mount<ENTER>

again. You will now see a response where the line involving **/mnt** is gone:

```
/dev/root on / read/write on Tue Nov 17 17:15:28 1987
```

Now, you can retry the application installation.

To park or not to park... some answers to the question.

System 3, like Version 7, provides a rather nice utility called **park**. **park** does just that -- upon system shutdown, it positions the head assembly over the innermost cylinder on the drive. It has changed a little bit since its inception; whereas the early **park** utility had to be asserted every time the system was rebooted, the current Xenix 3.1.x and 3.2 **park** command modifies the file **/xenix** so that it remembers which way **park** was toggled last. In other words,

park -on <ENTER>	turns the **park** utility on until root turns it off
park -off <ENTER>	turns the **park** utility off until root turns it on
park <ENTER>	tells you what the current state of **park** is.

park is a good thing to use, whether or not you are moving the hard drive around. It provides a little extra insurance against the possibility of glitching the boot track... and any help in that area is welcome. There is one instance where **park** is contraindicated: **park should not be used with 8 meg drives.**

.....

Don't use **park** with 8 meg drives....
They'll hurt themselves!

.....

Due to the way the **park** command works, it doesn't get along with 8 meg hard drives, so you should not use it. On a brighter note, **park** does work fine with 12, 15, 35, and 7Ø meg hard drives, both primary and secondary... bear in mind that when **park** is on, the last secondary on the chain is going to be the drive whose activity light is left on after you shut the system down. This is normal! Also, if you are running Xenix 3.2, when you shut the system down, you will see messages telling you which drive is being parked. This is just giving a little visual feedback to the user to let him know that "yes, I really **am** parking the drives!".

Losing the Screen Display:

You may find yourself getting a call sometime from a panic-stricken customer who whimpers, "Profile 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>** and **<REPEAT>** simultaneously. (If the keyboard doesn't have a <REPEAT> key, hit <ENTER> a few times instead).

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> <ENTER>

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.

All of this holds true for all System 3 Xenix versions, but there's an additional wrinkle with Xenix 3.2.... the console now has a "screen saver" function, which may cause the screen to go blank after a certain period of inactivity.

Xenix 3.2 Console Screen Saver:

In Xenix 3.2, the console now has a "screen saver" function, which may cause the screen to go blank after a certain period of inactivity. In other words, if there is no activity at the console for more than **2Ø minutes** (no data going to the screen, or no keyboard activity), the screen will "blank out".

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

   In Xenix 3.2, if the console is idle
   for more than 2Ø minutes, the screen
            will "blank out".
.........................................
```

Any key except the <SHIFT>, <CTL>, <CAPS>, <LOCK>, and <REPEAT> keys will turn the display back on. Also, sending any new data to the screen will turn it back on.

However, let's say you are in an application program, where you don't really want to send any data to the program when you turn the screen back on, which pressing a key will do. There is a special key sequence which will turn the screen back on, but **not** send anything to whatever program you are running. To turn the screen back without sending data, type:

<CTL><=> (turns console display back on when screen saver has turned it off)

This will turn your display back on without perhaps generating unexpected side effects! (Like maybe accidentally generating a 5Ø,ØØØ byte report....)

If you decide that you don't want to use the screen saver feature on the console at all, there's a way to turn it off, too. This command is a toggle, so using it twice has the effect of not doing anything at all. To turn the screen saver off, press:

<CTL><.> (toggles screen saver off. Pressing it again turns it back on.)

If you re-enable the screen-saver feature with the **<CTL><.>** command, the screen will go dark immediately, instead of waiting 2Ø minutes, to let you know immediately that you re-enabled the feature.

.....

To turn the console screen on without sending data,
press **<CTL><=>**.

To toggle the console screen saver feature off/on,
press **<CTL><.>**.

.....

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 processes with extreme prejudice.
(Who needs ØØ7?)

.....

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

1) Press **<BREAK>**. 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 id number of the runaway.

3) Try a subtle approach first. Type:

kill -2 PID <ENTER>

where "PID" is the process ID number you want to kill. 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!

The Dreaded 3.Ø and 3.1.x System Slowdown -- Some Possible Causes and Fixes:

Xenix versions 3.Ø and 3.1.x will sometimes seem to slow the pace of operations down to the point of being nearly unusable. This would be understandable if there were 8 users on the system all busily typing away in Scripsit, but sometimes it happens when there is nearly no one on the system. A couple of things to look for when this happens are:

1) Make sure that the users are logging in on separate user names. If you have 4 people all logged in at the same time as "clerk", for example, the system will run a lot slower than it would if the same 4 users were logged in as "tom", "dick", "jane", and "dumbo".

2) If a serial port is enabled (whether or not it is a multi-terminal board or serial A or B), it should have a terminal which is powered up connected to it... or, more exactly, it should have something connected to it which will allow it to establish communications.

.....

In any version of Xenix,
if a port is **enabled**,
the terminal connected to it should be **on**.

(You'll save yourself a lot of grief.)
.....

Paying attention to those two items will make a lot of "system slowdowns" go away.

On a related note, this phenomenon of powered-down terminals slowing the system down (called "thrashing") is dealt with a little differently in Xenix 3.2.

Automatic Terminal Disabling in Xenix 3.2:

Before I explain what Xenix 3.2 does when it encounters problems talking to a terminal, let me first give you a little idea of what happens to get that "login: " prompt up on the screen.

.....

Xenix uses three programs to establish terminal
communications and to allow users to log in.

They are called **init**, **getty**, and **login**.
.....

To begin with, there are three programs which essentially control the processes of terminal communications and logins. Those programs are named **/etc/init**, **/etc/getty**, and **/etc/login**. (There is also a /bin/login, but it is not directly involved in this process).

The **/etc/init** program is the one which starts the whole thing off, by opening a line to a terminal (provided the port is enabled). Once it has completed that task, it calls another program, **/etc/getty**, with a line mode (or the baud rate as defined in /etc/ttys) as an argument. If **/etc/getty** is successfully established, the "login: " prompt will appear on the screen.

When the user enters his login name, the **/etc/getty** program then calls **/etc/login**, which asks the user for his password (if he's got one) and starts up his command interpreter (usually a shell). When the user logs out, the **/etc/init** process wakes up, and starts the whole thing over again.

This process, with minor changes, holds true for all versions of System 3 Xenix. The differences between the earlier versions and Xenix 3.2 become apparent, however, when you have a situation where a line is "thrashing", or trying to initialize over and over again.

In Xenix 3.2, the **/etc/init** process tries to prevent line thrashing and the subsequent system slowdown by monitoring the line opened to a given terminal port. There are several common causes of this line thrashing phenomenon:

- Attaching a powered-down terminal to an enabled port. (See above under "System Slowdowns").
- Attaching a long cable with nothing at the other end to an enabled port. This cable can pick up noise, which may appear as characters to the terminal port.
- Enabling a port with nothing attached to it at all. (Rare, but it happens occasionally).
- Using an intelligent modem which either echoes commands or produces command responses. This can lead to a scenario where the **init** and **getty** programs talk to the modem, which talks to the programs, which talk to the modem... you see what I mean!

If the **/etc/init** program has to restart the **/etc/getty** program within 3Ø seconds of the last time it was started; **init** starts counting the times this problem occurs. If **/etc/getty** terminates 4Ø times without ever lasting more than 2 minutes, then the **/etc/init** program starts to insert 2Ø second delays between each startup of the **/etc/getty** program. If the **getty** program dies another 4Ø times without lasting more than 2 minutes at a time, then the **/etc/init** process will automatically disable the troubled port, and send mail to root about the problem which will look something like this:

```
From root Thu Jul 1Ø 13:15:Ø1 1987
Subject: Init: Thrashing problem on line ttyØ5.
This line will be disabled.
```

If the **getty** process runs more than two minutes, the **init** program assumes that the line is working ok, and resets its "problem" counter. Also, **init** will never disable the console (good news), although if you repeatedly enter enough <CTL><d>'s it will insert 2Ø second delays. Leave the console keyboard alone for two minutes, and the 2Ø second delays will be reset to normal.

Once a thrashing line has been disabled, it will not be automatically re-enabled! The root user will have to re-enable that line. If you re-enable a thrashing line without curing the problem which caused the thrashing, the line will stay up for at least 2Ø minutes. It may stay up longer than that, but you can only count on 2Ø minutes worth of time.

.....

Once **/etc/init** has disabled a thrashing terminal line, it will not be automatically re-enabled!

Enabling a port requires human intervention.

.....

Now, probably you are starting to ask yourself, "Just how can I tell what the **/etc/init** and the **/etc/getty** processes are doing on a system?" That's not too hard... in Xenix 3.2, you can use the **ps** command to examine the state of the processes controlling the terminal lines to see what's happening on a given system.

Let's say that we have a hypothetical system with a multiterminal board with two DT-1ØØ's connected to it, and an intelligent modem of some variety on the third port. The two built-in serial ports, ttyØ1 and ttyØ2, are connected to a DC2212 and a DT1, respectively. The problem you were called out to the site for was that the customer couldn't get the DT1 to work at all, one of the DT-1ØØ's quit working, and the line that the intelligent modem is on keeps being disabled. In other words:

ttyØ1	DC2212	works
ttyØ2	DT1	doesn't work
ttyØ4	DT1ØØ	doesn't work, but did once
ttyØ5	DT1ØØ	works
ttyØ6	intelligent modem	works, but gets disabled periodically

In order to see what's happening on the system, type the following:

ps -ef<ENTER>

In the case of the above system, we got something like this as a response:

```
   UID   PID  PPID  C    STIME TTY  TIME COMMAND
  root     Ø     Ø 19  Dec 31   ?  Ø:ØØ swapper
  root     1     Ø  Ø  Dec 31   ?  Ø:Ø9 /etc/init -n          (1)
  root  552Ø     1  Ø  Nov 2Ø  co  Ø:Ø6 -sh
  root    2Ø     1  Ø  Nov 17   ?  Ø:44 /etc/update
  root    31     1  Ø  Nov 17   ?  5:Ø8 /etc/cron
  root 1Ø575     1  Ø 15:23:5Ø  ?  Ø:ØØ /etc/init init 0 Ø4   (2)
  root 1Ø579     1  Ø 15:24:16  ?  Ø:ØØ /etc/init init 0 Ø1   (3)
  root 1Ø582     1  Ø 15:24:32  ?  Ø:ØØ /etc/init init 0 Ø2   (4)
  root 1Ø588     1  Ø 15:25:Ø5 Ø5  Ø:ØØ - 9                   (5)
  root 1Ø589     1  Ø 15:25:Ø5  ?  Ø:ØØ /etc/init init T Ø6   (6)
  root 1Ø591  552Ø 24 15:25:2Ø co  Ø:Ø2 ps -ef
```

I've highlighted the lines we're interested in **boldface**. The numbers in parenthesis at the end of each line are just reference numbers so we can discuss each line.

(1) The "**/etc/init -n**" is the master /etc/init process for the system. Nothing is wrong here.

(2) This entry is for tty04. You will note that it is not associated with a terminal yet (that's what the ? in the "TTY" column means). The interesting part is:

init 0 04

This says that the port is enabled, but there is no carrier detect (indicated by the capital "0"). The "04" at the end tells you what terminal the process is aimed at. Now, this entry is normal for a modem dial-up line, since modems don't usually assert carrier detect until they actually detect a carrier. However, for a hard-wire terminal (the DT-100 on tty04) that is turned on and should be working, it's an indication that something is wrong. Check the hardware and the cabling out!

(3) This entry is for the DC2212 on tty01. It's being used as a dial-up line, and the entry:

init 0 01

saying the port is enabled, but has no carrier detect, is perfectly normal for a modem waiting for a call.

(4) This is the DT1 on tty02 which has never worked. It, too, says, that it is enabled and has no carrier detect:

init 0 02

Since this is one of the built-in ports, and it has never worked, you might check to see that there is a null modem installed here! Remember, for terminals to be used on ports tty01 and tty02, you need null modems. Additionally, you might have other hardware or cable/setup problems.

(5) Here, you have an entry other than a question mark in the "TTY" column (a "Ø5") and you have the following in the command column:

- 9

This indicates that **/etc/getty** is running on ttyØ5, and is ready to communicate at 96ØØ baud. The "9" in the "- 9" is the baud rate entry for ttyØ5 in the /etc/ttys file. This terminal is sitting at the "login: " prompt (if it is turned on).

(6) This port, out of all of these, is the only one which is actively thrashing. This is indicated by:

init T Ø6

The "**T**" means that the line is thrashing, and that the **/etc/init** program is pausing for 2Ø seconds before starting **/etc/getty** again. Now, this port has an intelligent modem connected to it... which may mean that the modem and the **/etc/getty** program are mumbling to one another. If the modem echoes commands sent to it, and has a "forced carrier detect" option, this may result. Here, you may want to consult with the customer as to modem configuration.

As you can see, just a quick glance at the **ps** output can yield a lot of information about the system! Let me summarize the things you may see while doing this and what they mean (I'll use a sample terminal line of ttyØ4):

init O Ø4 — Line ttyØ4 is enabled, but has no carrier detect signal. This is normal, and typical of a dial-in line that is not in use. If this occurs on a direct-wired terminal that is turned on and should be working (but isn't), it's probably indicative of hardware or cable problems.

init N Ø4 — Line ttyØ4 is enabled, and is executing a 1 second pause before starting **getty**. This is normal, but isn't very common. The line is apparently not thrashing.

- 3, - 9, etc. — These indicate that **getty** is running on a line. The terminal that it is running on can be located in the "TTY" column in a "**ps -ef**" output. The baud rate (the letter or number after the "-") is the speed code from the /etc/ttys file. This is normal, and is typical of a terminal sitting at the "login: " prompt.

init T Ø4 — Line ttyØ4 is enabled, and is pausing for 2Ø seconds before starting **getty** because the line appears to be thrashing. This indicates trouble with the line or with what is connected to the line.

init F Ø4 — Line ttyØ4 is enabled, but the system was unable to open this line. The system will wait 1Ø seconds before trying again. This indicates trouble with the line which may include:

No hardware for the line (i.e. in the case of ttyØ4-ttyØ6, no multiterminal board installed).

Enabling the line while another program was making an outgoing call (like **uucp** or **cu**).

The driver for the port is hung.

If you get a problem indicated by the "**F**", wait a couple of minutes, disable the port, wait another couple of minutes, re-enable the port, and see if you still have the problem. If the problem is still there, then suspect hardware or a hung driver. In the case of a hung driver, try shutting the system down with the **shutdown** command and rebooting.

/etc/init -n — The master system **init** process.

Local Printing Problems:

Local printing is a nice feature offered by Xenix... it means that you can avoid having to run over to the console printer 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 9600 baud, try 4800 or even 2400 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 9600 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 saying plaintively that one terminal on their system just won't clear the screen when they log out... yet when you look at the hardware everything checks out fine. Typically the problem will be exhibited on tty06 on systems with one multi-terminal board, and on tty09 on systems with two multi-terminal boards. If this is the case, you need to check the files /etc/ttys and /etc/ttytype for the presence of a blank line at the end of the file. For the terminal initialization process to work correctly on the last terminal entry in those files, there must be a blank line at the end of said file. If there is no blank line there, use ed to add one... your problem should vanish once those two files have been modified.

****

DOS STUFF

How to tell which version you've got
and what they do... or don't!

****

Xenix Ø3.ØØ.ØØ -- was the first release of System 3 Xenix. There were a few items that didn't work quite like Version 7 that caused some confusion.... when printing, for example, you got an extra form feed which was a bit of a nuisance. The manual wasn't horribly helpful, and the DC2212 support was also a bit confused. The file system structure was radically different than that on Version 7 which necessitated a somewhat complicated upgrade procedure, not to mention the hardware modifications which are necessary to run System 3 on 6 MHz hardware.

Xenix Ø3.ØØ.Ø1 -- never officially released, this was an upgrade which came with the System 3 development system which only applied if you had Version Ø3.ØØ.ØØ installed.

Xenix Ø3.Ø1.ØØ -- second official release of System 3, this version has a file system structure which is yet again different, meaning another involved upgrade procedure. This version has increased the speed of disk I/O, comes with online HELP pages for the Bourne shell as well as the tsh, supports Hayes modems, and has the drivers necessary to run the disk cartridge system. In addition, the minor problems encountered in Xenix Ø3.ØØ.ØØ were corrected.

There is a printer bug with this version which crops up most frequently when printing long reports... it manifests itself as the printer either dropping offline or going not ready, or the spooler locking up. Typically, the problem is fixed by putting the printer back on line or shutting the system down and bringing it back up, causing the printer to continue with the job it stalled on.

Xenix Ø3.Ø1.Ø1 -- a patched version of 3.1.Ø, this was available on an upgrade disk through the software support personnel. The only major difference with this version was that the printer bug mentioned under 3.1.Ø had been fixed.

Xenix Ø3.Ø1.Ø2 -- this version was available as an upgrade disk from the warehouse. Again, this is a patched version of 3.1.Ø, and it has incorporated into it the printer patch present in 3.1.1, and additional changes which decrease the likelihood of seeing "Drive Not Ready" error messages (particularly on systems which have 35 meg hard drives). This version should not be used in lieu of doing the mandatory hard drive technical bulletins! Rather, it should be used in conjunction with the appropriate modifications on systems which display this problem even after the modifications, alignments, voltages, and so forth have been verified as being correct.

Xenix Ø3.Ø2.ØØ -- the most recent version of System 3, this version incorporates some fairly major changes. The file system structure and format is the same as that for Xenix 3.1.Ø. New features in this release include: support for the 1Ø meg DCS and the 2Ø meg (external) DCS, on-line formatting of floppy and cartridge disks, automatic reboot, console screen saver, system configuration utilities, support for the high resolution graphics board on the console, two additional command interpreters (**csh** and **csh1Ø**), a new editor (**vi**), a mail program (**xmail**), automatic disabling of thrashing serial channels, support of new Daylight Savings Time standards, line printer spooler maintenance commands, and a host of other new options.

There is one minor bug in this version, which will only be noticed if you are trying to use a 3ØØ baud modem for dialout with the built-in autodialers. The autodialers ignore speed flags and always try to dial out at 12ØØ baud. A patch is available for this problem, and I'll discuss it in the "Advanced System Information" section.

When **shutdown** Won't Work

or

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

Sometimes the **shutdown** command 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 seemingly plays dead. What now?

Generally, when this occurs, the **<BREAK>** key or a **<CTL><_>** will return the root prompt. If it does, make sure that 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 is happening is that the process status command **ps** is freaking out and never returning from talking to the little Xenix gremlins... since **shutdown** uses the **ps** command to invoke a search and destroy mission on some of the active processes, when this command flames out, the **shutdown** process will sit and wait forever for something to happen. If you can recover, it will be the **<BREAK>** or the **<CTL><_>** that will do it for you.

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.

As a footnote to the above discussion, if when performing the **ps** command the terminal you are using locks up, **do not try to shut the system down using the shutdown command.** It will fail! Use the **sync;sync;/etc/haltsys** command instead.

Automatic Reboot Options in Xenix 3.2:

Remember back when we were talking about Xenix versions? Under the Xenix 3.2 discussion, I mentioned something about automatic reboot. Here's where I explain that a little further.

.....

In Xenix 3.2, you have three options as to how your system will boot or reboot:

(1) Completely manual

(2) Only autoboots if reset (i.e. like a power failure or a normal shutdown followed by an operator reset)

(3) Reboots on crash (like a "panic:" message), on reset, or on a normal system shutdown.

.....

This is quite different than anything we've had before with Xenix for the 68ØØØ-based machines. Although automatic reboot is familiar in System 5 Xenix for Tandy 3ØØØ's, the option had never previously existed for 6ØØØ's.

The command you use (as root!) to turn on the various flavors of bootup in Xenix 3.2 is called **autoboot**. In order to turn automatic boot completely off, leaving the system in the state we are used to seeing on 6ØØØ's, type:

autoboot -n<ENTER>

The system will respond:

```
Reboot option is now OFF
Auto-boot option is now OFF
autoboot: Installing new boot track...
11+1 records in
11+1 records out
```

This is the condition that will be present when Xenix 3.2 is first installed on a system, either as an upgrade or a direct installation.

If you want to set autoboot so the only time an automatic boot will take place is after a hardware reset (either power off induced, or operator induced via the reset switch), type:

autoboot -a<ENTER>

The screen will say:

```
Reboot option is now OFF
Auto-boot option is now ON
autoboot: Installing new boot track...
11+1 records in
11+1 records out
```

This option will not reboot the system after a "panic: " or a normal shutdown; it only deals with automating the process once you've arrived at the "Xenix Boot> " prompt.

Finally, if you want to set both the autoboot and the reboot functions so that almost anything in the way of a system stoppage, whether normal or abnormal, will allow the system to reboot itself, type:

autoboot -ar<ENTER>

The screen will say:

```
Reboot option is now ON
Auto-boot option is now ON
autoboot: Installing new boot track...
11+1 records in
11+1 records out
```

Both the "autoboot -a" and "autoboot -ar" commands add a few comments to the boot process, as well as send root mail on the subject. When you come to the initial screen, you will now see:

```
[Automatic Reboot Enabled]

Xenix Boot>
```

Also, right before the system normally asks you for the time, you will see:

```
[Automatic reboot in progress]
Current system time is: Tue Nov 24 15:47:Ø7 CST 1987
```

The system automatically sets the time by itself during an automatic reboot.

Finally, the system will send mail to root documenting the reboot. To read root's mail, type:

xmail<ENTER>

You'll then see something like this:

```
xmail - Improved mail utility  v2.8(Ø5)  Type '?' for help.
You have new mail.
```

Once in **xmail**, you will have a prompt which says:

```
mail>
```

To get the new mail, type:

n<ENTER>

Xmail will tell you:

```
getting new mail...

1 new message:
u  1 root      Tue Nov 24 14:34  "Auto-boot log"          [197 bytes]
```

To read a message, type its number followed by <ENTER>. In other words, in this case, you would type:

1<ENTER>

The screen will show as you type this:

```
mail> t 1

Reading message 1:
From root Tue Nov 24 14:33:13 CST 1987
To: root
Subject: Auto-boot log



Nov 24 14:3Ø
**  Normal System Shutdown  **

[Z8Ø Control System Halted]
Current system time is: Tue Nov 24 14:3Ø:Ø8 CST 1987
```

This gives you the conditions which prompted the reboot/autoboot (in this case, a normal system shutdown at 14:3Ø on November 24) and the time which the system finished coming up multiuser.

If you want to print this mail out, type:

p<ENTER>

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

q<ENTER>

and you will be back at the root prompt.

****

SECTION FIVE

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 is 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 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 implementations all the time. Some of it you will simply not be able to do without access to the Development System software. If it sounds like I'm trying to throw some healthy trepidation at you, you're quite right -- I am. Tread lightly.

For those of you who are curious, I will be picking apart a bit of the Xenix Tools disk in this section and will tell you how to do some of the particularly useful things in their inconvenient form. This may prove helpful to those of you who are in a bind on a system without a double-sided floppy drive.

****

An Introductory Note About
The Xenix Tools Disk

****

This disk was first prepared for use on Xenix 1.3.5 systems. As such, the first version of the Tools Disk will **not** work properly as it stands on System 3 systems. In order to get it to work on Xenix 3.Ø, 3.1, or 3.2, you will need to regenerate the disk. This turns out to be fairly simple.

Log in as root on a system running the desired version, and use **install** to install the Tools on the hard disk. Then, log in as "tandy" with a password of "support" to bring up the hard disk menu. The "k" option of the menu can then be used to produce another Tools disk with the Xenix version which is resident on the hard drive. This isn't totally fool-proof, but it should allow you to use most of the tests available to you on the Tools diskette.

The second version of the Tools Disk was prepared on Xenix 3.1.Ø. If you are desperate, and for some reason **can't** regenerate, you may use this disk directly on Xenix 3.1.Ø, 3.1.1, and 3.1.2 systems, although you should make certain that you upgrade the system to the correct version afterwards should you use the Tools Disk to, say, reinstall the kernel and boot track. If you want to use this disk on a Xenix 3.2.Ø system, you should first regenerate the disk and make one with the 3.2.Ø version of Xenix on board.

The reasons for these precautions are:

(1) Xenix 1.3.5 cannot read System 3 filesystems. No way, no how.

(2) Putting the wrong boot track or kernel onto a system is a good way to make matters worse than they started out! A 1.3.5 boot track won't work on Xenix 3.2.Ø, and conversely a 3.1.Ø boot track won't make a 1.3.5 system very happy, either.

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, **diskutil** will do a verify on all of the tracks it formats (provided it was told to), but some flaws will just not be picked up by a formatter. A media retention problem generally will not be found by a formatter. However, difficulties of this nature usually will be included on the flaw map.

Remember the System 3 utility **diskstat**? That's a good way (available on all System 3 systems) to check to see that the flaw map has been correctly entered. It is, in fact, exactly the program being used to do this on the Xenix Tools Diskette (floppy menu option "k", hard disk menu option "p"). To use it, type:

diskstat drive number
optional, will default to "Ø"
may use Ø, 1, 2, or 3

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

Figure 1 shows a decimal dump of a file called **/dev/hdbtØ**. This is the boot device for the primary hard drive. It contains information on the size of your hard drive and what errors have been locked out, among other things. If, for argument's sake, you were interested in the flaw map on the first secondary, the file would be **/dev/hdbt1**. Naturally all this junk 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/hdbtØ ¦ more **<ENTER>**

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

Now, check out Figure 1. Looks impressive, eh? The big question is, "What does it mean?" Figure 2 is the flaw map that corresponds to the bubble that this file was dumped off of. Take a look at Figures 1 and 2 and see if you find a relationship between the two -- if you don't, not to worry, I'll explain it anyway.

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

ØØØØØØØ 11888 3Ø312 ØØØ16 ØØ512 ØØØØ8 ØØØ17 ØØ512 ØØ256
ØØØØØ2Ø ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ
*
ØØØ2ØØØ 11874 24932 ØØØ96 ØØØ18 ØØØ14 ØØØØ3 ØØØ23 ØØØØ1
ØØØ2Ø2Ø ØØØ5Ø ØØØØ5 ØØØ52 ØØØØ1 ØØØ54 ØØØØØ ØØØ54 ØØØØ1
ØØØ2Ø4Ø ØØ148 ØØØØ7 ØØ149 ØØØØ3 ØØ266 ØØØØØ ØØ291 ØØØØ7
ØØØ2Ø6Ø ØØ337 ØØØØ1 ØØ358 ØØØØ1 ØØ416 ØØØØ1 ØØ462 ØØØØ6
ØØØ21ØØ ØØ466 ØØØØØ ØØ49Ø ØØØØ5 ØØ493 ØØØØ5 ØØØ22 ØØØØØ
ØØØ212Ø ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ
*
```

Figure 1.

....

....

Flaws Entered Into Flaw Map on HDØ (35 meg hd)

Cylinder	Head
14	3
23	1
5Ø	5
52	1
54	Ø
54	1
148	7
149	3
266	Ø
291	7
337	1
358	1
416	1
462	6
466	Ø
49Ø	5
493	5
22	Ø

Figure 2.

....

Figure it out? Well, first off, what is the command **od -d**? This command is an **octal dump** of whatever file you specify as an argument. Frankly, I don't much enjoy working in octal, I much prefer working in hex or decimal. By adding an argument to the **od** command, we can tell it what radix we want to see the file dumped out in. A **-x** will force a hex dump. A **-d** will force a decimal "short" (16 bit) dump. So essentially what we did was to do a decimal dump of a file called /dev/hdbtØ.

Now let's pick this dump apart.

```
ØØØØØØØ 11888 3Ø312 ØØØ16 ØØ512 ØØØØ8 ØØØ17 ØØ512 ØØ256
ØØØØØ2Ø ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ
*
ØØØ2ØØØ 11874 24932 ØØØ96 ØØØ18 ØØØ14 ØØØØ3 ØØØ23 ØØØØ1
ØØØ2Ø2Ø ØØØ5Ø ØØØØ5 ØØØ52 ØØØØ1 ØØØ54 ØØØØØ ØØØ54 ØØØØ1
ØØØ2Ø4Ø ØØ148 ØØØØ7 ØØ149 ØØØØ3 ØØ266 ØØØØØ ØØ291 ØØØØ7
ØØØ2Ø6Ø ØØ337 ØØØØ1 ØØ358 ØØØØ1 ØØ416 ØØØØ1 ØØ462 ØØØØ6
ØØØ21ØØ ØØ466 ØØØØØ ØØ49Ø ØØØØ5 ØØ493 ØØØØ5 ØØØ22 ØØØØØ
ØØØ212Ø ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ ØØØØØ
*
```

The things that matter to us in the above mess are the lines with boldfacing and boldfacing/underlining. 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 five 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.

But we really don't care about the file position, although the first piece of interesting information -- the number of cylinders and heads (i.e. the drive size) will occur at relative byte ØØØØØØ6 Octal (remember Octal counts Ø, 1, 2, 3, 4, 5, 6, 7, 1Ø, 11, ...). If you're not into Octal, just go to line ØØØØØØØ (the first line) and count over to the fourth number (ØØ512) and start there. In this example, the cylinder/head counts are shown in boldface.

Look at the first line:

```
 ØØØØØØØ 11888 3Ø312 ØØØ16 ØØ512 ØØØØ8 ØØØ17 ØØ512 ØØ256
                           |      |
number of cylinders is 512 __|      |
                                    |
number of heads is 8 _______________|
```

The next interesting bit of information is the flaw map. From that file, we can determine the maximum number of errors the system can lock out on the drive, how many have been locked out, and what those flaws are. The maximum number of bad tracks starts at relative byte ØØØ2ØØ4 Octal, the number of tracks actually locked out at relative byte ØØØ2ØØ6 Octal, and the flaw map itself at relative byte ØØØ2Ø1Ø Octal. Let's take a look:

```
ØØØ2ØØØ 11874 24932 ØØØ96 ØØØ18 ØØØ14 ØØØØ3 ØØØ23 ØØØØ1
                    |     |     |     |     |     |
maximum bad tracks _|     |     |     |     |     |
                          |     |     |     |     |
number of tracks locked out _|    |     |     |     |
                                |     |     |     |
1st flaw entered is cyl 14, hd 3 _|___|     |     |
                                            |     |
2nd flaw entered is cyl 23, hd 1 _____________|___|
```

Your flaw map may be bigger, or smaller, depending upon the number of flaws that were entered. All you have to do is to compare each five digit decimal number pair with your hard drive flaw map. The flaws in the dump are shown as cylinder-head. When you start seeing a lot of zeros, or the second asterisk (*), you're done.

A couple of things to bear in mind are that for 8, 12, and 15 meg drives, the maximum number of bad tracks allowed is 24. For 35 and 7Ø meg drives, the maximum number of bad tracks allowed is 96.

Of course, you can find all this information just as correctly and rather more simply using **diskstat**, but this is a good thing to remember should you ever find yourself troubleshooting a system with outdated software. A further note: **diskstat** works fine on Version 7 systems, too.

.....

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 Z8Ø and 68ØØØ subsystems) 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 is good, you don't see any gross read errors under HDREL... but the boot track is blown. Normally, the thing to do in this situation would be to re-diskutil, reinstall the Xenix core and have the customer restore all of his programs and data. One problem: the customer has nineteen months of 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; 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 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 the floppy drives are double-sided, and 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 the following steps:

1) Boot the Xenix Tools Disk. When you reach the floppy menu, select menu option "1". This will copy the boot track information back to the hard drive.
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.** 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 Boot Diskette, of the **same version Xenix that is resident on the hard drive.** With this in hand, install it in drive Ø and boot the system on floppy.

At the prompt, press <ENTER> to boot into floppy Xenix. Answer "n" to "Do you want to initialize your hard disk?"; this should put you into a standard single-user shell.

Step Two:

Now that you're booted into Xenix, you should have the root prompt(#). If you don't, you did something wrong. If you do, then proceed with the following. Enter the following commands:

1) **dd if=/hdboot of=/dev/hdbtØ bs=1b seek=3 <ENTER>**
This command recopies the boot track information back onto the primary hard drive.

2) **sync;sync;/etc/haltsys <ENTER>**
This command will halt the floppy-based Xenix system.

3) You should now see the "**Normal System Shutdown**" message.

4) Press **RESET** to boot from the primary hard drive.

What if the above doesn't work? Well, nil desperandum -- I've got one more trick up my sleeve before we try a non-destructive format; we'll restore the boot track **and** the kernel (consisting of /xenix and /z8Øctl). As before, I'll first assume that you have a Tools Disk:

1) Boot the Xenix Tools Disk. When you reach the floppy menu, choose option "b", which will allow you to clean a filesystem. We want to clean the primary hard drive, so when prompted, type "hdØ" for the drive to be cleaned.
2) Once the drive has been cleaned, you will be returned to the floppy drive menu, where you should select menu option "m". This will copy the boot track and the kernel back to the hard disk.
3) 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.** 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 **don't** have a Tools Disk, you will need a version of the Xenix Core Boot Disk with the correct version of Xenix, and a File Maintenance Disk which also matches the version you are using. Then, follow these steps:

1) Boot off of the File Maintenance Diskette. Choose the menu option which allows you to "Check a File System". Follow the prompts, and clean the primary hard drive. When the cleaning is complete, use the "q" option to shut the File Maintenance Disk down.

2) Boot off the Core Boot Disk, and enter these commands:

3) **dd if=/hdboot of=/dev/hdbtØ bs=1b seek=3 <ENTER>**
This command recopies the boot track information back onto the primary hard drive.

4) **/etc/mount /dev/hdØ /mnt<ENTER>**
This command mounts the primary hard drive onto the boot floppy so that we can copy the /xenix and /z8Øctl files over.

5) **copy /xenix /mnt/xenix<ENTER>**
copy /z8Øctl /mnt/z8Øctl<ENTER>
These commands copy the /xenix and /z8Øctl files (the 68ØØØ and Z8Ø control programs) over to the hard drive.

6) **/etc/umount /dev/hdØ<ENTER>**
This unmounts the primary hard drive.

7) **sync;sync;/etc/haltsys** **<ENTER>**
This command will halt the floppy-based Xenix system.

8) You should now see the "**Normal System Shutdown**" message.

9) 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 odds of its success are virtually nil. Boot up on floppy, go into **diskutil** and try a non-destructive format with no verify. Assuming the format takes (large assumption, that!) you can proceed to try to reinstall the boot track. As I say, the odds are against you on this, but since you've nothing to lose anyway, it may be worth a shot.

A footnote to this is that if you try the non-destructive, no verify format option, make sure you are using the version of **diskutil** which is appropriate for the drive in question. Due to the different interleaves in Xenix 3.Ø and 3.1/3.2 (see Section 1 for more information), using the wrong version is worse than doing nothing at all.

.....

BEEFING UP THE SWAPPER

or

How to increase swap space on the primary
(is more really better?)

.....

Normally installed, the Xenix Core System will provide the user with 2Ø48 blocks of swap space on the primary. Knowing that a block is equal to 512 bytes, this works out to 1Ø24K or 1,Ø48,576 bytes or what we would call one megabyte of swap space. For most normal applications, this should be plenty. (Actually, I'm rounding the numbers a wee bit... on a 3.1 or 3.2 system, the default is 1Ø28k, but this is close enough to 1 meg of swap as to make no difference).

Sometimes, however, it may be necessary to increase the swap space on the hard drive to allow numerous huge application packages to run a tad faster. The reasons why or when we might want to increase this space is really beyond the scope of this paper; this is provided solely as reference material. It is easy, though, to tell when you **don't** need to increase swap space, and the way to make that determination is included here.

Why Swap?

Just what does this swapper thing do, anyway? Look at it like this. You've got two gallons of Kool-Aid -- one gallon of blue, and one gallon of red, and there is a one-gallon-sized cup to drink from (a gallon of Kool-Aid is a lot, but this is Texas). In order to drink some of the blue, you first have to pour a gallon into the cup; then you can swill it down... but don't drink yet; we need it all. Your buddy wants some red, and you, being such a nice person, decide to let him have his share first. In order to do this, you pour your blue Kool-aid back into the blue gallon container and then pour a gallon of red Kool-Aid into the cup. What have you just done besides being so nice? You swapped blue Kool-Aid for red. Ok, but how does this apply to Xenix?

The two one gallon containers of Kool-Aid represent, we'll say, two 512K programs and the one gallon cup will represent 512K of 68ØØØ RAM. There is no way you're going to get 512K of RAM to hold two 512K programs at the same time, unless you change the universe, but if you could why would you be fixing Xenix systems.... but enough philosophy, back to pertinent matters. To run the red program (drink it) you first have to get it into the cup, but the cup is full of the blue program and there is no room. How do you do it? Easy -- just pour the blue Kool-Aid (program) back into its container (part of the swap space on the hard drive) and then pour the red Kool-Aid from its container (part of the swap space) into the cup (RAM). Now you can drink the red Kool-Aid (run the red program).

The swap space is just a holding area on the hard drive for programs to be "swapped" into and out of RAM depending upon what we're running. This is necessary because Xenix is a multi-user environment and there will always be several programs in RAM belonging to different users at the same time. If the programs are too big to all fit into RAM, then we have to swap some of them out onto the hard drive. If the programs are really big, and the amount of RAM is fairly small, then the system may overflow the swap space. This is when we might want to increase the size of the swap area.

If there is a lot of time shown on the swapper, then you will probably want to increase the swap area. You can tell the swap time by doing a **ps -elf** and noting the amount of swap time shown under the **TIME** heading that corresponds to the process labelled **swapper**.

How To Do It

Xenix 3.x makes it easy for you to increase the swap space. However, it does require reformatting the drive with **diskutil** and reinstalling the operating system.

NOTE: If you do a save of what is currently on the system, and you are running Xenix 3.Ø, do a **tartype** save. If you use **sysadmin**, you will wind up with the same size swap space that you had before. Xenix 3.Ø File Maintenance Diskettes do not ask you for swap size; Xenix 3.1 and 3.2 do.

When you reinstall the core system, when it asks you for the amount of swap you want, give it the desired amount **in blocks**. (Remember, a block is 512 bytes in this system). For example, if you want 2 meg worth of swap space, type **4Ø96**. When it creates the file system on the primary, you will wind up with one meg less on the file system, but Xenix will make the necessary adjustments automatically.

.....

Fixing "Fast" Boot Disks,
or
How to Get Your Model II
Over Those "Hard Error FE1Ø on trk 5Ø"
Initialization Blues

(What's a nice disk like you doing
at a step rate like that?)

.....

Most of you probably at one time or another have gone forth to deal with a Model II based Xenix system, prepared to reinitialize the system and daring it to give you trouble... and suddenly, it does! Every single core boot disk you've got gives you the same sort of error: "Hard Error FE1Ø on trk 5Ø", or something very much like it. You think to yourself, "Shoot, must be alignments," so you align the FDC and floppy drive. You try again. You still can't get the system initialized. You sigh. You check the modifications, and they're all done. You try *again*. Still no joy in Mudville. You replace everything in the system. You try *AGAIN*. ARGGGGGHHHHHHHH!!!!

Sound familiar? Actually, it should, since we talked about this problem briefly in Section One, when we were initializing systems. Remember during the initialization process, when you are asked if the machine is a Model II? If you answer "yes" to that question, the floppy drive seek rate is left wherever it was when you were asked the question... on a fresh set of disks, that's at 1Ø milliseconds, which is a nice safe step rate for all eight inch floppy drives in the drive Ø position. (I know that CDC drives like 15 milliseconds, but then, they are never used as drive Ø).

On the other hand, if you answer the question with a "no", the system thinks, "Aha! Not a Model II! Must have thinline drives..." and sets the floppy drive seek rate to **3 milliseconds**. This is great for thinline drives, but causes great consternation in the ranks of Shugart and TPI drives. This change is permanent, too... unless you explicitly fix that floppy disk, it will now **always** cause the floppy drives to step at 3 milliseconds. This is why if you try to use this "fast" disk in a Model II you will get errors... it's because the poor Shugart or TPI just can't haul that head assembly around as fast as the Tandon drives do!

Fortunately, there is a fix to this problem, although as always in the case of an improperly shutdown floppy diskette you are better off getting another copy which is known to be clean. We will have to clean any diskettes you have treated in this manner, and you may lose some files in the process. Still, this procedure will help to recover some disks sufficiently to allow you to get a system installed.

To recover these disks, we will use the **/etc/drive** command, which allows you to set various parameters about your floppy drives such as their seek rate, whether they wait for the motor to spin up, and whether they can detect drive door open or not. The format for **/etc/drive** is:

/etc/drive [number] [parameters]

where

number = the drive number (Ø-3)
If you don't use a number, the requested parameters will be set for all drives.

parameters = **wait** or **nowait**
detect or **nodetect**
seek=[value] where **[value]** is one of the following:

Ø = 3 milliseconds
1 = 6 milliseconds
2 = 1Ø milliseconds
3 = 15 milliseconds

How to fix your floppy disks to work again on Model II's:

(1) Find a machine with at least two thinline drives.

(2) Using either a Tools Diskette of the appropriate version or a File Maintenance Diskette, clean the Core Boot Disk(s) you wish to fix.

(3) After cleaning the Core Boot Disk, place it in Drive Ø and boot it up. When you are asked "Do you wish to initialize your hard drive?", answer **no**.

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

/etc/drive seek=2<ENTER>

This will set the seek rate to 1Ø milliseconds for all drives on the system.

(5) When the root prompt returns, shut the diskette down by typing:

sync;/etc/haltsys<ENTER>

Now, your diskette should work on Model II systems. As I've mentioned before, perhaps the simplest way to avoid this sort of problem is to keep a separate set of disks reserved exclusively for initializing Model II systems.

.....

What is the mandatory /etc/drive setting
on thinline drives?

.....

Easy... wait. If this isn't set, the system
won't wait for the drive to spin up, and
errors will result.

.....

Aside from seekrate, there is one other setting which may cause occasional grief on systems where the /etc/drive command has been used to tailor parameters. If you should run into a system with thinline drives where the drives work -- sometimes -- and otherwise come up and say "not ready", examine the possibility that the wait parameter is not set.

To check this out, type:

/etc/drive<ENTER>

This will give you the parameters for the floppy drives on that system. For thinline drives, you should expect to see something like this:

```
drive Ø: detect, wait, seekrate=Ø
drive 1: detect, wait, seekrate=Ø
drive 2: detect, wait, seekrate=Ø
drive 3: detect, wait, seekrate=Ø
```

If instead, you should see:

```
drive Ø: detect, nowait, seekrate=Ø
drive 1: detect, nowait, seekrate=Ø
drive 2: detect, nowait, seekrate=Ø
drive 3: detect, nowait, seekrate=Ø
```

then you will have this problem. To fix it, type:

/etc/drive wait<ENTER>

and the problem will go away (provided no hardware problems existed)! As an additional note, you need to have root privileges to use the /etc/drive command.

Systems Which Lose (or Gain) Time

(Xenix 3.2 only!)

Ever have someone complain about their system gaining or losing time? It's a common phenomenon; the real time clock is driven off the Z8Ø, and tends to (generally) lose a little time... the more disk activity on a system, the more time lost, since the RTC is the last thing serviced.

Well, it used to be that there wasn't really much to be done about this. As of Xenix 3.2, there is something you can do about it; use the command **/etc/adj_clock**.

The **adj_clock** program lets the root user correct for small time gains and losses on his system. The format is:

adj_clock [+/-seconds /period[s|m|h|d]]

or

adj_clock -f fudge

If you want to find out what the current adjustment is on the system, use just the **adj_clock** command by itself:

adj_clock<ENTER>

You will see something like this:

```
The adjustment is 119Ø:
Currently adding 1 second every 19 minutes and 5Ø seconds
```

The value "119Ø" is the current adjustment. You can arrive at this by converting "19 minutes and 5Ø seconds" to seconds:

```
19 minutes * 6Ø seconds/minute = 114Ø seconds
                                + 5Ø seconds
                                ------------
                                 119Ø seconds
```

This "adjustment" is the value you would use with the "-f" option. Let's say, for some reason, that you are going to install another kernel onto the system, and this customer was aware that they had a value set for **adj_clock** that they were happy with. **Before** you put the new kernel in, you would check the value that **adj_clock** had set in the old program with:

adj_clock

The current value of the system's adjustment would be displayed:

```
The adjustment is 121Ø:
Currently adding 1 second every 2Ø minutes and 1Ø seconds
```

You install the new kernel, and check its current value. You find that it is different. To make the new kernel match the old in terms of time adjustment, you would type:

adj_clock -f 121Ø<ENTER>

The system would respond:

```
The new adjustment is 121Ø:
Now adding 1 second every 2Ø minutes and 1Ø seconds
```

This is one way of making time adjustments. The more commonly used method is for the system administrator (i.e. the root user!) to notice that the system is, say, losing about 5 seconds every 1Ø hours. If he wanted to correct for that, he could type:

adj_clock -5 /1Ø

The system will respond:

```
The new adjustment is 1Ø21:
Now adding 1 second every 17 minutes and 1 seconds
```

When you issue this command, you are saying "I am losing 5 seconds" (the -5) "every 1Ø hours" (the **/1Ø**). If you don't specify the units for the period of time you are correcting, **adj_clock** will assume hours. **Note:** there must be a space between the correction ("-5" in this case) and the period ("/1Ø"). Otherwise, you will get an error message.

The **adj_clock** program does have some limitations. They are:

```
Minimum correction:  +1 second every 18 hrs 12 min 15 sec
                     (Adjustment of 65535)
Maximum correction:  +1 second every second
                     (Adjustment of 1)
```

If you run into a system which is adding
one second every second,
and you want to get it back to original condition,
type:

adj_clock -f 119Ø<ENTER>

(This is how it comes out of the box!)

Adjusting your clock is bad enough, but Xenix 3.2 also allows you to do even more exciting things to your system...

Xenix 3.2 Kernel Reconfiguration...

(you need a lot of guts for this...)

One of the features which comes with Xenix 3.2 is the ability to reconfigure the operating system without the use of the development system. This allows the user to tailor certain features of the system such as the maximum number of processes per user, maximum memory per process, number of system buffers, and the like.

All of this power can be useful, since certain (non-Tandy) packages may require that these system parameters be changed. Without the configuration program **cfg**, you would have to have the Development System to affect these changes. However, this is a two-edged sword... you **can** end up with a system that doesn't boot any longer, too!

I am **not** going to tell you everything you ever wanted to know about kernel reconstruction (that could take years and millions of lives...) but I **will** tell you how to set a kernel back to its default values, and how to affect one of the most common changes, an increase in the amount of memory allowed per process.

Let's start with Part One of the configuration saga: how to set the kernel back to the defaults. This is useful when you have a system which comes up at boot time saying something like, "WARNING: bad configuration, resetting to defaults". What that message is trying to say is that whoever reconfigured this kernel specified that an huge amount of RAM be used for Xenix, and didn't leave enough RAM left over for the users. Xenix figures this out when it tries to boot, and ignores the settings in the kernel, using some default values which are more reasonable. This allows the system to boot and run, but obviously is not the most stable of arrangements. To keep this from occurring every time the system comes up, we can use the configuration program, **cfg**, to put the kernel back to its normal condition.

Actually, we will set it **almost** back to its normal condition. The command I'm going to show you will not specify changing the swap device back to its normal configuration... but this probably won't be a difficulty provided the system is running swap off the primary hard drive (which it almost always will). So, let's fix a kernel; log in as root, and at the prompt type the following:

(1) Log in as root, and save a copy of the customer's **/xenix** (which has a problem of some variety) so they can examine it later if they wish:

cp /xenix /xenix.old<ENTER>

This makes a copy of the **/xenix** program in a file called **/xenix.old**.

(2) Reconfigure **/xenix** back to somewhat more normal parameters:

cfg -d<ENTER>

The system will respond with:

```
/xenix is configured as follows:
    Buffers     AutoConfig
    Inodes      AutoConfig
    Files       AutoConfig
    Locks       AutoConfig
    Procs       AutoConfig
    Clists      AutoConfig
    Maxprocs    15
    Maxmem      256k
    Nodename    "technoid"      \_____ both of these will be
    Swapdev     AutoConfig      /      left alone by "cfg -d"
```

Note: "Swapdev" set to "AutoConfig" is the **normal** configuration.

(3) Shut the system down and reboot. The error message regarding configuration should no longer appear.

That wasn't so bad, eh? Now, to deal with Part 2... how to crank the maximum amount of memory per process up a bit. As an aside to this, I will mention that you need to consider that the minimum amount of swap to be set aside on a system should be **at least four times** the maximum memory allocated per process on the system. In a "default" system, the maximum memory per process (or "maxmem", as it is colloquially called) is 256k. Therefore, we would expect a "default" system to have at least one meg of swap space... which it does. If you are involved in helping raise the maxmem on a system, you should check the available amount of swap space and assure yourself that it will be large enough to handle the load. If it isn't, the system will need to be saved off and reinstalled, enlarging the swap space in the process. **Failure to do this may cause the system to crash with "panic: Out of swap" messages!**

Well, let's try an example: you have a customer who needs a maxmem of 384k on their system.

(1) Log in as root, and make a backup of the **/xenix** program:

cp /xenix /xenix.old<ENTER>

(2) Verify that sufficient swap space exists on the system to raise the maxmem value to 384k. You will need to have at least four times the amount of the proposed maxmem available in swap:

Needed: 4 * 384 = 1536k swap

Examine the **/usr/adm/messages** file to determine available swap:

tail /usr/adm/messages<ENTER>

In our case, the system responded:

```
Dec  8 Ø9:3Ø
[Z8Ø Control System   Version  3(12Ø)  23-Mar-87]

Tandy 68ØØØ/XENIX version 3.2(42)

Microsoft  XENIX  v3.Ø

System 77k+51k+32k=16Øk, User 864k
Root 13328k Swap 2Ø56k
                 ¦_________ This is sufficient for our needs.
```

If the swap space on this system had been, say, 1Ø28k, you would have had to save the system off and reinstall, raising the amount of swap space.

(3) Reconfigure the kernel to the larger amount of memory:

cfg maxmem=384<ENTER>

The system should respond:

```
/xenix is configured as follows:
    Buffers     AutoConfig
    Inodes      AutoConfig
    Files       AutoConfig
    Locks       AutoConfig
    Procs       AutoConfig
    Clists      AutoConfig
    Maxprocs    15
    Maxmem      384k
    Nodename    "technoid"
    Swapdev     AutoConfig
```

(4) Shut the system down and reboot.

The **cfg** program will let you do a lot of other things to **/xenix** as well, but that's a bit beyond the scope of this document; mostly, these will probably be all you'll need to be concerned with.

"Well," you say, "what do I do about my customer running Xenix 3.1 who needs their maxmem increased?" There is something which can be done for systems running earlier versions of Xenix, (in fact, two somethings), but it requires that the customer has a copy of the Xenix Development System. One thing is that the customer can acquire a copy of the Configuration Kit, which lets him adapt his system in much the same way as the Xenix 3.2 cfg program. The other thing only requires the use of the **adb** program, a debugging program which comes with the Development System.

.....

Increasing the Value of **Maxmem** On
Earlier Versions of Xenix....
(3.Ø, 3.1.x)

WARNING!
Don't do this unless you are certain
you know what you're doing!

.....

I'm serious when I say, "Don't do this unless you know what you are doing". It is all too easy to screw this procedure up, so if you aren't sure what I'm talking about, **don't try it!**

Having said that, what we are going to do here is use **adb** (which stands for "a debugger" -- original, right?) to adjust the value of maxmem in the **/xenix** program. The only thing which makes this tricky is **adb**; even as debuggers go, **adb** is arcane in its syntax, and not at all forgiving of operator errors. So, in order for this to go correctly, follow this procedure and you won't go far wrong.

Let's use our example of the customer who needs a maxmem value of 384k:

(1) Log in as root, and make a backup of the **/xenix** program:

cp /xenix /xenix.old<ENTER>

(2) Verify that sufficient swap space exists on the system to raise the maxmem value to 384k. You will need to have at least four times the amount of the proposed maxmem available in swap:

Needed: 4 * 384 = 1536k swap

Examine the **/usr/adm/messages** file to determine available swap:

tail /usr/adm/messages<ENTER>

In our case, the system responded:

```
Dec  8 Ø9:3Ø
[Z8Ø Control System   Version  3(12Ø)  23-Mar-87]

Tandy 68ØØØ/XENIX version 3.2(42)

Microsoft  XENIX  v3.Ø

System 77k+51k+32k=16Øk, User 864k
Root 13328k Swap 2Ø56k
             |_____ This is sufficient for our needs.
```

If the swap space on this system had been, say, 1Ø28k, you would have had to save the system off and reinstall, raising the amount of swap space.

(3) Use **adb** to change the maximum memory per process:

Type the following:

```
adb -w /xenix<ENTER>
_Maxmem?D<ENTER>
```

The system will respond with the current value of "_Maxmem", which is the name the kernel uses for maxmem. If you currently have a stock system, this is what you will see:

```
_Maxmem:        256
```

Next, we will change the value assigned to "_Maxmem". Type:

```
_Maxmem?W 384<ENTER>
```

The system will respond:

```
_Maxmem:        Øx1ØØ           =       Øx18Ø
```

Now, to verify that the change was made correctly, type:

```
_Maxmem?D<ENTER>
```

This should make the system display the new value:

```
_Maxmem:        384
```

To exit adb, press **<CTL><d>** together. The root prompt should reappear.

(4) Shut the system down and reboot.

****

If you can't boot a modified kernel
after any reconfiguration,
you can boot the system off your backup file
/xenix.old
by typing "/xenix.old" at the
"Xenix Boot> " prompt.

****

As long as we're talking about patching and configuring **/xenix**, let me bring up a fix to a small problem with the release version of Xenix 3.2.

.....

Xenix 3.2 and 3ØØ baud autodial modems ONLY!

Xenix 3.2(42) has a minor bug in the autodialer code.

If you try to use the kernel dialers to dial out for you at 3ØØ baud, even if you correctly specify the speed, they won't... they always dial at 12ØØ baud.

If you have a customer trying to autodial out at 3ØØ baud, who needs this fix, read on.

Otherwise, don't mess with this!

.....

This is a **very** specific problem fix, and it only applies under the following circumstances:

- The customer is running Xenix version 3.2(42). (To figure this out, type:

  dmesg<ENTER>

  This will put the most recent boot messages up on the screen:

  ```
  Dec  8 Ø9:3Ø
  [Z8Ø Control System   Version  3(12Ø)  23-Mar-87]

  Tandy 68ØØØ/XENIX version 3.2(42)

  Microsoft  XENIX  v3.Ø

  System 77k+51k+32k=16Øk, User 864k
  Root 14356k Swap 1Ø28k
  ```

  The line in **boldface** gives you the kernel version. If it says "version 3.2.(42a)", the fix has already been applied.

- The customer is trying to use the kernel autodialers to dial out at 3ØØ baud, and they aren't working.

If your case matches this one, here's the problem: there is a minor bug in the autodialer code which makes it always dial out at 12ØØ baud (even if you correctly specify that you want to dial out at 3ØØ baud). To fix this problem, we will use yet another program called **patch**.

To implement this fix, **verify** that the customer has version 3.2(42)!

(1) Log in as root, and make a backup of **/xenix**:

cp /xenix /xenix.old<ENTER>

(2) Enter the following commands. (The stuff in **boldface** is what you type; the remainder is how the system responds).

patch /xenix<ENTER>

byte offset (<ENTER> to exit)? **34b9<ENTER>**
34b9: 8Ø !.! **>d4<ENTER>**
34ba: ff !.! **>q<ENTER>**

byte offset (<ENTER> to exit)? **d4Øf<ENTER>**
d4Øf: 8Ø !.! **>eØ<ENTER>**
d41Ø: Ø2 !.! **>q<ENTER>**

byte offset (<ENTER> to exit)? **d4a1<ENTER>**
d4a1: Ø7 !.! **>Ø6<ENTER>**
d4a2: Ø2 !.! **>q<ENTER>**

byte offset (<ENTER> to exit)? **1Øfde<ENTER>**
1Øfde: 29 !)! **>61<ENTER>**
1Øfdf: Ød !.! **>29<ENTER>**
1ØfeØ: Øa !.! **>q<ENTER>**

byte offset (<ENTER> to exit)? **<ENTER>**

After you press <ENTER> to exit, the root prompt should appear.

(3) Shut the system down and reboot to have the changes take effect.

.....

Xenix 3.2 only

The **idle** Program...

useful for keeping idle users from
slowing the system down unnecessarily.

.....

If you ever run into a situation where a system is running slowly, and is plagued with users who log in and do nothing else -- for hours! -- consider suggesting the **idle** program as a way of keeping this sort of problem in hand. Or, in a different problem, if you have someone complaining that "I keep getting logged out every few minutes," check and see if **idle** is in use on the system.

What **idle** does is find users that have been logged in and idle for more than a specified period of time, and logs them out. Normally, this program is invoked by **cron**, the clock daemon. You can set the time allowed for a user to be idle, and several other useful parameters, in a file called **/etc/default/idle**.

To use **idle**, you will need to put a line in the file which tells **cron** what to do (**/usr/lib/crontab**) and also verify that the **idle** defaults are what you want them to be. Here is a sample **/etc/default/idle** file:

```
# Put the names of "non-work" programs in here
NULLFILE /etc/idle.pgms

# Where to record logouts
LOGFILE  /usr/adm/logouts

# Tunable Constants (minutes)

#   Normal Idle limit
C_LIMIT  5Ø

#   Dial-up Idle limit
CD_LIMIT 2Ø

#   "Appears-busy" Idle limit
CB_LIMIT 24Ø

#   Dial-up "Appears-busy" Idle limit
CDB_LIMIT 6Ø

#   How often the program is run (used to figure when to warn the user)
CHECK_INTERVAL 1Ø
```

I'll explain these entries one at a time:

NULLFILE: this is the name of a file where programs which are not considered in the conditions for being "busy" are listed. Usually, they are:

```
<defunct>
csh
csh1Ø
sh
```

LOGFILE: where to record users who were logged out by **idle**.

C_LIMIT: normal idle time limit.

CD_LIMIT: same as "C_LIMIT" except for dial-in lines. Note: dial-in lines in this sense are denoted by "ttyd"; since on a standard system all lines start with "ttyØ" this parameter is not particularly useful.

CB_LIMIT: the idle time limit for anyone running a program not in the "NULLFILE" list. For example, this limit would apply to anyone sitting idle in Profile.

CDB_LIMIT: same as "CB_LIMIT" except for dial-in lines.

CHECK_INTERVAL: tells **idle** how often it will be executed by **cron**. This value is used to allow **idle** to calculate when it needs to warn users.

Using the editor of your choice, make any necessary changes to the default file. Then, to finish this up, you will need to put a line in **/usr/lib/crontab** telling **cron** when you want **idle** executed. Before I tell you what that line will be, let's briefly review the layout of the **/usr/lib/crontab** file:

```
5,15,25,35,45,55 * *    *       *       /etc/dmesg - >> /usr/adm/messages
Ø,3Ø     *       *      *       *       /usr/lib/atrun
2Ø       1       *      *       *       /usr/bin/calendar -
¦        ¦       ¦      ¦       ¦               ¦
¦        ¦       ¦      ¦       ¦               ¦_______ command to execute
¦        ¦       ¦      ¦       ¦                        (use full pathname)
¦        ¦       ¦      ¦       ¦_____________________ day of the week
¦        ¦       ¦      ¦                              (Ø-6)
¦        ¦       ¦      ¦_____________________________ month of the year
¦        ¦       ¦                                     (1-12)
¦        ¦       ¦_____________________________________ day of the month
¦        ¦                                             (1-31)
¦        ¦_____________________________________________ hour  (Ø-23)
¦_____________________________________________________ minutes (Ø-59)
```

According to this, then, if we wanted to execute **idle** (full pathname: **/usr/lib/idle**) every 1Ø minutes, we should use an editor to add a line to **/usr/lib/crontab** which looks like this:

```
Ø,1Ø,2Ø,3Ø,4Ø,5Ø * *    *       *       /usr/lib/idle
```

Idle won't immediately blow users off the system; it will warn them first by displaying a message that if they don't type something within a given time period, they will be logged out. This allows users who are temporarily tied up elsewhere an opportunity to prevent themselves being logged out.

The reason I'm telling you about this is that it can be extremely (!) disconcerting to find yourself being logged out as a user if you aren't expecting it. Frequently, people may not even know what is running that is logging them off, and so may attribute their difficulties to hardware problems. A lot of times, they might be right, but if you can't find anything wrong with the system, take a look at **/usr/lib/crontab** and see if the system is using **/usr/lib/idle** to log idle users out. Bear in mind that being logged out by **idle** when you are running a program like **Scripsit** can leave lock files around which may make it difficult to run the application afterwards!

.....

Xenix 3.2 only!

/etc/inittab,

or

How to set up an **extremely** special-purpose login.

.....

Xenix 3.2 offers the option of being able to set up special-purpose logins on specific tty lines which bypass the normal process of entering a login name and a password and take the user directly into a program. For certain applications, this can be an extremely useful tool. Let's say, for example, that you had a dial-in line (with an autoanswering modem), and you wanted to be able to have branches of the company call in with sales figures... but you didn't want logins for all these people, you just wanted them to go directly into one Profile screen, enter data, and get out. Xenix 3.2 can let you do this via a file called **/etc/inittab**. This file when properly set up, can tell the **/etc/init** program how to use the **/etc/getty** program to bypass the normal login process and go directly to a program.

Confused? Don't worry, so was I when I first saw this... remember how the login process normally works:

```
/etc/init --> calls /etc/getty at a baud rate determined by /etc/ttys --
                                                                        !
              -- user enters login name <-- "login:" prompt appears <--
             !
              --> /etc/getty calls /etc/login --> "Password: " prompt --
                                                                       !
                                  user put into login shell or program  <--
```

Now, that's how it **normally** works. Here's what happens if we use **/etc/inittab** to set up a special purpose login:

```
/etc/init --> references /etc/inittab, sees special entry --
                                                           !
       -- calls /etc/getty with baud rate and user name <--
      !
       --> when CD seen, /etc/getty calls login program in /etc/passwd --
                                                                       !
            user is put into program without prompting for login name <--
```

Ok? In this special purpose entry, once Carrier Detect is seen at the port, the user is **automatically** placed into the login program specified in the password file.

Let's look at how to set a sample of this up. We'll configure a system where all branch offices dial up a modem on **tty01**, and go directly into a program called **/usr/bin/sales** (a completely made-up program!) as a user named **branch**. First of all, **disable tty01!** Next, we'll create the **/etc/inittab** entry for this login.

The format of the **/etc/inittab** file is four colon (:) separated fields:

```
0:01::/etc/getty branch
| | |          |
| | |          |_________ name of getty program with special user
| | |                     name
| | |_____________________ this field is always empty
| |_______________________ two character terminal ID
|_________________________ always a zero (0)
```

Use an editor to create a **/etc/inittab** file with the above line in it. The second field terminal identifier "**01**" corresponds to the two character number corresponding to the terminal line, which in this case is tty**01**. The fourth field has the name of the **/etc/getty** program and the name of the special user **branch** separated by a space.

Make a user name called **branch**. When it asks you for a password, do not give it one; logins of this type do not use them. When it asks you for shell type, tell it "1"; this will use the normal shell -- but we'll change that in a minute.

Once you've created this user, we will need to edit the **/etc/passwd** file entry for this user so that the login program is no longer "/bin/sh" but our special program **/usr/bin/sales**. In other words, if the **/etc/passwd** entry for **branch** looked like this:

```
branch::204:50:Branch Offices:/usr/branch:/bin/sh
                                          |____ login program name
```

we would need to change it to this:

```
branch::204:50:Branch Offices:/usr/branch:/usr/bin/sales
                                          |____ special login
                                                program
```

As a final step, verify that the entries for the terminal port in the **/etc/ttys** and **/etc/ttytype** files are correct for your situation, and re-enable the port. Now, when anyone calls up the auto-answer modem on tty01, they will automatically go into **/usr/bin/sales** as user **branch**.

When might this become a problem for us? Well, you go into the program automatically whenever Carrier Detect is asserted at the port. For a dial-up line, this usually means only when the modem answers a call. What happens if somebody does this on an **ordinary terminal line?** This line will (usually) have Carrier Detect asserted whenever the terminal is powered on and connected; therefore, it will always have someone logged in on it.

"So, why is this a problem?" I hear you ask. Simple... what happens if you want to disable that port? The **disable** program won't let you disable a port where a user is currently logged in. And, quitting the program won't help... all you'll see is a re-display of the **/etc/motd** file (if one exists) and back you go into the program....

The only way out of this scenario is to turn the terminal off or unplug it, and hope the system doesn't see a Carrier Detect at that port, which is somewhat dependent on the hardware. Another solution is to disable the port when the system is in **single-user** (or **system maintenance**) mode.

Moral of the story: if somebody has a problem similiar to this: "I can't log off my system, it just keeps dumping me back into the program!", try looking for a special-purpose login such as this one.

****

SECTION SIX

Saving Data and Upgrading Systems
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, "Hmm-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 Profile records (or Scripsit, or General Ledger, ad nauseaum) on there!! My business will collapse!!!" Great wailing, pounding of chests, and weeping generally ensues, and you the unfortunate technician on the spot may find yourself needing to either back this system up, or instruct the customer in the in's and out's of the process. (Yes, I know that in the perfect world the software support personnel are set up to do this, but this is the worst imaginable world we're examining here!)

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 3). **Tar** is used in the **tsh** "**save**" command, and is also frequently used in utility menus in application packages. 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 Xenix 3.1 and Xenix 3.2 are intelligent enough to only give you a bunch of nasty error messages if you try this, but this wasn't always the case... 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 on systems with more than one floppy drive, or, in the case of Xenix 3.1/3.2, on systems with at least one floppy and one cartridge drive. The reason for this 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 diskette, 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 long process -- not too good!

.....

Rule of Thumb:

For systems with more than one floppy, or a floppy and cartridge system, use **sysadmin** for full system saves.

For systems with only one floppy drive, use **tar** or **tsh "save"** for full system saves.

.....

Clear as mud, eh? The thing to keep in mind is that if the customer can boot off of floppy, and still has a backup device (i.e. cartridge/floppy drive) free, he can use **sysadmin** and get reasonable speed on system restores. Otherwise, he would be best advised to do a tar-type save.

All of which brings us to the real meat of the question: how do you actually perform a save? We'll start with a **sysadmin** save first, and look at situations where the customer has (a) two floppy drives and (b) a floppy and a hard drive.

/etc/sysadmin save procedure:

For systems with two (or more) floppy drives:

1) Format all the diskettes you are going to need **before** you start the system up. To format diskettes, use **diskutil**. If the customer is running Xenix 3.2, they may use **format** to format their diskettes. **Make sure that you have all the formatted diskettes you will possibly use before you start; otherwise you'll have to start over from scratch after formatting more diskettes!**

.....

In round numbers,

Double-sided diskettes hold **1 meg** of data,
Single-sided diskettes hold **1/2 meg** of data.

1Ø meg disk cartridges hold **1Ø meg** of data.
2Ø meg disk cartridges hold **2Ø meg** of data.
.....

2) If you used **diskutil** to format your diskettes, once you have finished formatting your backup media, reset the machine. At the "Xenix Boot>" prompt type **<ENTER>** to boot Xenix.

   If you used **format** to format your diskettes (Xenix 3.2 only), shut the system down and reset. Once at the "Xenix Boot>" prompt, press **<ENTER>** to boot Xenix.

3) At the point where the machine 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 System Maintenance (single-user) mode. It is best to do backups when the system is in System Maintenance Mode -- the process will go a lot faster and files are less likely to change while in the process of being saved.

4) At the " **#** " prompt, type

   /etc/sysadmin

5) The following menu will appear on your screen:

   File System Maintenance
   Type 1 to do daily backup,
   2 to backup all files,
   3 to get a backup listing,
   4 to restore a file,
   5 to restore a hard disk,
   6 to check a file system,
   q to quit

   > Enter Number:

6) Type 2 **<ENTER>** to back up the entire system. The screen shows:

```
FULL SYSTEM BACKUP
Backup which hard drive (Ø-3)?
```

5) In our case, we are assuming that this system only has one hard drive, so type **Ø <ENTER>**. The screen shows:

```
Backup to C)artridge or F)loppy drive?
```

6) Type **f <ENTER>**. The screen shows:

```
Backup to which floppy drive (Ø-3)?
```

7) Type **Ø <ENTER>**. Now the screen shows:

```
Type 1 for single sided, 2 for double sided:
```

8) Depending on what type of floppy disks you are using, type either 1 or 2 followed by an **<ENTER>**.

NOTE: You **cannot** change between single and double sided disks during the course of a given backup! Once you've decided, you're stuck (unless you bail out and start over).

The screen should now show:

```
Insert volume in drive Ø, then press <ENTER>
```

9) When the diskette is full, the screen shows:

```
Change volumes and press <ENTER> to continue.
```

1Ø) Insert a new formatted diskette. The cursor stays at the end of the prompt until you press <ENTER>.

NOTE: Make sure you label volume numbers in order! I know this sounds obvious, but if you get them out of order you'll have a real problem on your hands.

When the entire file system is backed up, the screen shows:

```
DONE
```

and gives you information on how much information you actually saved off.

11) The File Maintenance Menu should now be back on the screen. Type **q <ENTER>** to quit, and the root prompt will appear. To shut the system down, type

    sync;sync;/etc/haltsys <ENTER>

    and the screen will come back with

    ** Normal System Shutdown **

For systems with cartridge and floppy drives:

1) Format your cartridges using **diskutil**, if you are using Xenix 3.1.x. If you are running 3.2, you may use **format** instead, then shut the system down. Reset the system, and at the "Xenix Boot>" prompt type **<ENTER>**. Enter System Maintenance Mode at the prompt by entering the root password.

2) At the " # " prompt, type:

    /etc/sysadmin <ENTER>

3) At the File Maintenance Menu, type **2 <ENTER>** to do a full system backup.

4) The screen will show:

    FULL SYSTEM BACKUP
    Backup which hard drive (Ø-3)?

    Type **Ø <ENTER>**. The screen will then show:

    Backup to C)artridge or F)loppy drive?

5) Type **c <ENTER>**. The screen shows:

    Backup to which cartridge (Ø-1)?

6) Type **Ø <ENTER>**. Now the screen shows:

    Insert volume in drive Ø, then press <ENTER>

7) When the cartridge is full, the screen will show:

    Change volumes and press <ENTER> to continue.

8) Insert a new formatted cartridge. The cursor will stay at the end of the prompt until you hit <ENTER>.

9) When the backup is complete, the screen will say:

DONE

followed by some information on how much data was actually saved.

NOTE: If you are using a cartridge system, remember that Xenix locks the drives until it is finished accessing the cartridge!

Pretty simple, eh? It really only has one bad problem -- it's tedious, which is why system administrators frequently don't keep good backups.

Now, for those of you out there with only one floppy drive, this **sysadmin** stuff isn't going to work real well. The save will be fine; the only trouble is that the restore is going to take forever! For these situations, knowing how to do at least one type of tar-type save is going to be very useful, and that's what we'll look at next.

A Sample "tar" type Save:

For systems with only one floppy drive:

1) Format your floppy disks under **diskutil** (or **format**). Reset the system, and boot Xenix.

2) At the prompt, enter the root password to put the system into System Maintenance Mode.

3) At the root prompt " **#** " type **tsh <ENTER>**. This will put you into the "trs-shell". This shell is relatively user friendly and supports TRSDOS-like syntax. You will get a prompt that looks like:

tsh-#>

4) At the "**tsh-#>**" prompt, type one of the two following commands. The one you type will depend on whether you are using single or double sided floppy diskettes to do the save.

   Single-sided Floppies:

   save :Ø -ss -all

   Double-sided Floppies:

   save :Ø -ds -all

5) 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 "**tsh-#>**" prompt will return. Type **<CTL><D>** to return to the "**#**" prompt, and

   sync;sync;/etc/haltsys <ENTER>

   to shut the system down.

The "tsh" **save** command can also be used to save off individual files or directories; in fact, it is very good as a way to backup just small portions of a system.

.....

"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 "tsh" **save** disks won't be restorable using **sysadmin**, and vice-versa. On that note, turn to the next groups of instructions, match your system configuration, and find out how to restore your data.

.....

Relax....

If you've made it this far,
the rest is easy.
.....

/etc/sysadmin restores:

For systems with two or more floppy drives or a cartridge:

1) Find the diskette in your Xenix package labeled "File Maintenance Diskette." Insert the File Maintenance Diskette in Drive Ø. Press the reset switch, and press **<BREAK>** and **<REPEAT>** simultaneously until the system boots off floppy and you see the "Xenix Boot>" prompt on the screen.

2) Press **<ENTER>**. The screen shows:

```
System loaded
Change root if desired
type <enter> to proceed or <break> to abort
```

3) Press **<ENTER>** and the following menu will appear (after a certain amount of noise from the floppy drive):

```
                File System Maintenance
Type    1 to get a backup listing,
        2 to restore an entire backup,
        3 to restore a daily backup,
        4 to check a file system,
        q to quit
```

4) Type **1 <ENTER>** to verify that you are restoring the latest backup. Answer the prompts, and insert your first backup diskette/cartridge into the appropriate drive when asked to do so.

5) When the backup listing is complete, press **<ENTER>** to return to the File System Maintenance menu. Now type **2 <ENTER>**, and the screen shows:

```
RESTORE AN ENTIRE BACKUP SET TO A HARD DISK

Restore to which hard drive (Ø-3)?
```

6) Type **Ø <ENTER>**, and Xenix asks for information about your hard disk. (If you are running Xenix 3.Ø, this information will require that you know the cylinder/head count of the drive; otherwise, it will ask you just about swap size desired). If you have to give information on the size of the drive, use the same information you used when you formatted the hard disk! If you wish to increase the swap size, and you are asked about it, you may do so here (provided you will end up with enough room after the increase to do the save!) After you enter the information, your screen shows:

```
About to erase any data on drive Ø
Press ENTER to continue or BREAK to abort:
```

NOTE: Any existing information really will be overwritten at this point, so be sure that you've got a good backup or you could be acutely embarrassed!

7) Press **<ENTER>** if you really are ready to continue, and then answer the prompts as they appear on the screen. During the process of restoring, you may see the following message on your screen:

```
"/dev/rhdØ" contains data; press <ENTER> to continue
or <BREAK> to abort:
```

8) Press **<ENTER>** and continue restoring. When all the information on your backup diskettes/cartridges has been restored to your hard disk, your screen shows:

    End of Dump

9) The File System Maintenance menu reappears on your screen. Type **q <ENTER>** to quit. Wait until your screen shows

    [Z8Ø Control System Halted]

    You may then press the reset switch to boot Xenix.

tsh save restores:

For systems with one floppy drive:

1) Find the Boot Disk in your distribution set. Insert the disk in drive Ø, and press **<BREAK>** and **<REPEAT>** simultaneously until the "Xenix Boot>" prompt appears on your screen.

2) At the prompt, press **<ENTER>**. Various whirring and grinding noises will commence, and the screen will eventually show:

    System loaded . . .
    Change root disk if desired;
    type <enter> to proceed or <break> to abort

3) Press **<ENTER>**. Further noises will emerge, and after a while the screen will eventually show:

    Do you wish to initialize your hard disk?

    Since we wish to do so, type **y <ENTER>**. The program now asks:

    Has your hard disk been formatted with diskutil?

    Presumably, the answer is yes to this too, so type **y <ENTER>**. The screen shows the following message:

    Is this a Model II? [y or n]

4) Type the appropriate answer for the machine you are using. The screen shows:

```
Do you want the standard size swap area? [y or n]
```

We will assume for the moment that the answer is yes... it might not be, but we'll address that possibility elsewhere.

5) Whirs, clicks and groans emanate from the floppy drive, and the screen shows that initialization is proceeding. This process takes a few minutes, so be patient.... When initialization is complete, the screen shows:

```
Please reboot from the hard disk after the system
shuts itself down.

Halting system . . .
** Normal System Shutdown **

[Z8Ø Control System Halted]
```

6) Now, remove the Boot Disk from drive Ø. Reset the system, and at the "Xenix Boot>" prompt, press **<ENTER>** to boot Xenix.

7) Your screen will show a long message about distribution floppies, but this is where your tar save floppies will be used. You should eventually see a prompt which says:

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

8) Insert your first save diskette and type **y <ENTER>**. When the data is transferred to your hard disk, the screen shows:

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

9) Continue until your save disks have all been restored. At that point, at the "Next Floppy? [y,n]" prompt, type **n <ENTER>**. The system will prompt you for some additional information, and eventually come up with the standard boot message. You may now boot the system... you've completed the restoration process.

You may well ask at this point, "Well, this is great, but what am I supposed to do with this information?" Good question, actually... remember, we are responsible for ensuring whenever possible that saves of customer data exist before we blow it into non-existence. Knowing how to do saves and restores (as well as the other things addressed in this manual) will often save you a long wait for software support to arrive at your location to bail you out of a situation and may even help to placate an angry customer. Besides, it's rather closely related to the next subject I'll mention, which is the dreaded system upgrade.

What I'm going to cover here is how to upgrade from 1.3.5 or 3.Ø to 3.1. This is a fairly tedious process, since reformatting is involved as well as installing the new version. (The reason these upgrades are described here rather than in Section One is because Xenix 3.1 is no longer the most current version).

.....

Upgrading Xenix systems is a lot like doing backups... but the two processes are not identical.

.....

All the stuff about backups is going to come in handy now, because in a lot of ways a system upgrade is similar to a system backup. Still, there are differences -- instead of putting everything back on the system, just as it was, you are selectively restoring only certain things like data and custom files. This could be fairly complicated if you had to do it all by hand. Instead, the process is automated to a certain extent by the programs contained on the upgrade disk.

Xenix 1.3.5 to Xenix 3.1 Upgrade Procedure:

1) We will assume that the customer's hardware has been upgraded to support System 3... this means that they either have the 8 MHz 68ØØØ hardware, or their 6 MHz boards have had the necessary modifications performed.

2) Make sure that you have enough diskettes to back up all the hard drives in the system **twice**. Since this is a system upgrade, you need to do this complete backup twice to make sure you've covered all possibility of error.

3) Backup all your hard drives in the manner best suited to the system configuration (see the **Backup** instructions above).

.....

The above steps are in case of accident...
The next steps describe the upgrade.

.....

4) Format enough diskettes to back up all the drives in the system **once**.

5) Log in as **root** on the console. Use the **who** command to make sure that all users (other than root) are logged out.

6) Be sure that all secondary drives are mounted in their usual places on the system. To do this, type:

/etc/mount <ENTER>

This displays a list of mounted secondary drives and where they are mounted. **Write this information down**; we're going to need it later!

7) Type:

install <ENTER>

The Installation Menu will appear on the screen. (We looked at this in Section 1, remember?) Type:

1 <ENTER>

to begin the upgrade procedure.

8) The screen should say:

Insert diskette in Drive Ø and press <ENTER>

Insert the Xenix 3.1 Upgrade Disk in Drive Ø and press **<ENTER>**.

9) When the installation is complete, Xenix prompts you to remove the diskette and press **<ENTER>**. Your screen shows:

Constructing list of directories and permissions...
Creating list of files to save...
This is going to take a while...

Your hard drive(s) will probably start to make lots of official sounding noises. Don't worry -- that's normal.

10) Xenix will eventually ask you:

    Type 1 for single-sided, 2 for double-sided floppy disks:

    Enter the number which corresponds to the type of diskette you are using.

11) Xenix now prompts you to insert a diskette. Insert a formatted diskette in Drive Ø and press **<ENTER>**. Xenix then uses the **tar** command to start saving the files to floppy disk. As it fills each diskette, **tar** prompts for another formatted diskette. **Be sure to number the diskettes in the order they are made.**

    NOTE: If **tar** returns an error during this procedure, you had better stay on the safe side and start over, even though the system may try to convince you not to.

12) When the procedure is complete, Xenix warns you to ignore the message "umount: invalid argument." Remove your last diskette and press **<ENTER>**.

13) At the Installation Menu, type **q <ENTER>**.

14) At the root prompt, type **shutdown Ø <ENTER>** to shut the system down.

15) At this point, you will need to format and initialize your system under Xenix 3.1 as outlined in Section 1. **You have to reformat the drives!**

    .....

    Xenix 1.3.5 format and file system structure
    is **different** from Xenix 3.1....

    This means you **have** to reformat.
    (No one ever said it would be easy!)

    .....

16) Once you have your Xenix 3.1 core system installed on the primary hard drive, shut the system down and format any secondary drives according to the procedure outlined in Section 3.

17) Boot the system, and log in as root. You will need to use the **mkdir** (make directory) command to make the directories needed to mount the secondaries. **In order for the restore process to work, you must create the same directories as used previously for the secondary drives.**

18) Use **makefs** (Section 3 again) to create file systems on all the secondary drives. Use the **mount** command (also outlined in Section 3) to mount the secondary drives on the system in the same positions they were when the backup was made. For example, if **hard drive 3** was mounted on **/hd3** on the Xenix 1.3.5 system, it must also be mounted there for the upgrade to Xenix 3.1 to finish up properly.

.....

Almost finished...
just got to restore the data!

.....

19) At the root prompt, type:

/firsttime <ENTER>

2Ø) When prompted to insert the "distribution" diskettes, insert the first of the tar disks created earlier. Type **y <ENTER>**. Continue to insert the disks in the same sequence as they were made. When there are no more disks left, type **n <ENTER>** at the prompt.

21) Type the following command:

l /usr/bin/save <ENTER>

If Xenix lists the file, type:

rm /usr/bin/save <ENTER>
ln /usr/bin/SAVE /usr/bin/save <ENTER>

If Xenix says the file does not exist, no action is necessary.

Well, that's it... you've finished the upgrade to Xenix 3.1. The procedure for upgrading to Xenix 3.1 from Xenix 3.Ø is essentially identical, although it is pretty obvious that the hardware modifications should already be present since Xenix 3.Ø wouldn't have run without them. **It is important to note that when upgrading from 3.Ø to 3.1, you have to reformat the drive(s)!**

****

Differences Between Version 7 And System 3

And

Commands Which Are New With Xenix 3.2

****

The following is a list of some of the differences between good old Version 7 and its newer sibling System 3. This is not an exhaustive list! Undoubtedly there will be commands not listed here which are different, but hopefully this will cover most of the ones that you will encounter while working on a system.

cd -- change directory

They made this a little more idiot-proof; if you cd to a file instead of a directory, Xenix asks if you want to move to the next highest directory. In some quarters this is called the "do what I think, not what I say" option. For example, if "test.c" is a file and you try typing:

```
cd /usr/jim/test.c
```

The system prompts:

```
cd /usr/jim ?
Answer "y" or "n".
```

chdir -- change directory

No longer exists. Use **cd** instead.

chown -- change file owner

Under System 3, if you own a file, you can change its owner to someone else. You could only use this command as root under Version 7. Be careful when you're doing it, because once it's changed, you won't be able to change it back without the help of root or the new owner.

date -- current date

The format for entering the date has changed a bit... you can no longer enter seconds on boot (no great loss).

df -- disk free

The output format has changed a lot on this one; also, it runs a lot faster, and reports on all mounted file systems. If a customer wrote a shell script using output from this command, they will have to rewrite it due to the differences.

diskutil -- format disks

A non-destructive format option has been added which can sometimes be useful. You can also specify full, partial, or no verify during format, and interleave factor. **Under Xenix 3.1 (and later)**, there is also support for cartridge drives.

dump -- dump files to save media	This command (used in **/etc/sysadmin**) is also now called **backup**.
help -- manual pages	**Under Xenix** 3.1, help files (similar to the **man** command in Version 7) are available for optional use with the Core System.
login -- log in to system	You can no longer invoke this command while you are logged in... you have to logout first.
ls -l -- long form of directory listing	For those of you really into system details, this displays the setuid, setgid, and sticky bit permissions differently than it did on Version 7.
makefs -- make a file system	Use this menu-driven utility to make additional file systems. **mkfs** is still in the operating system, but it expects slightly different numbers than it used to and **makefs** is simpler to use. **Under Xenix** 3.1, there is support for cartridge drives.
mkuser -- create new users	This command has been enhanced to make assigning users to different groups easier (sometimes helpful for security reasons). There are additional prompts, so if a customer has written a shell script to automate this even further, they will need to allow for the additional prompts. As if this weren't enough, the default files given to users (like .profile) are now stored in /usr/lib/mkuser, not /usr/lib.
mount -- mount secondary file systems	You can now only use this command as root or in System Maintenance Mode. In addition, the output format has changed rather radically so if (again!) a customer is running a shell script depending on output from this command, he will have to rewrite it to account for the differences.

ps -- process status

There are hordes of new options, and the old ones don't mean the same thing. (Gotcha!) For example, the equivalent of Version 7's "ps lax" is now "ps -elf". You **have** to use the minus sign now. In addition, the output format has changed slightly. In case you have a customer who asks, the Version 7 and System 3 **ps** commands are <u>not</u> interchangable.

restore, restor -- restore save set

In Version 7, it was **restor**; in System 3, it is **restore**. Don't ask me why... I don't know either!

sh -- command interpreter

Again, for those of you into nit-picking detail (or who have customers who are), the command-line parsing is a little different. Most of the time, you'll never notice, but some shell scripts may be affected by it.

stty -- configure terminal parameters

Names for lots of the options have changed. Even worse, some things which were one-word options are now combinations of options (ugh). Here are a few samples for comparison:

<u>Version 7</u>	<u>System 3</u>
cbreak	time Ø min 1 brkint ignpar istrip ixon ixany opost isig -icanon
echo	echo echok echoe
-echo	-echo -echok -echoe
intrc 'c'	intr 'c'
quitc 'c'	quit 'c'
eofc 'c'	eof 'c'
brkc 'c'	eol 'c'

Note: 'c' in the above means 'character of your choice'.

tar -- tape archival

There is a new option that lets you use abbreviations to save data off on floppy. The option is a single digit:

Ø-3 specifies single-sided floppy drives Ø-3.

4-7 specifies double-sided floppy drives 4-7 where 4 is drive Ø, 5 is drive 1, and so on.

What this means is that you may use either one of the following syntaxes in System 3:

single-sided:

tar -cvfbk /dev/rfdØ 16 6Ø8 files
tar -cvØ files

double-sided:

tar -cvfbk /dev/rfdØ 16 1224 files
tar -cv4 /dev/rfdØ files

There are also certain files in System 3 which look different.

Files in System 3 which have some obvious differences are listed below. Again, this list is by no means exhaustive, but it will point to some of the more obvious pitfalls along your way.

/etc/passwd There are entries present for "uucp" and "sysinfo" which are new to the core system.

/etc/group Entries were added for "uucp" and "cron".

/etc/ttys The entry for the console has a "9" instead of an "h" for its baud rate.

/etc/rc There are loads of differences here, but the biggest ones are that if you wish to customize the /etc/rc file, you normally put the modifications into a file called /etc/rc.user. In addition, if the user adds any applications packages which require modification of the /etc/rc file, those mods are to be put in a file called /etc/rc.appl.

Commands Which Are New With Xenix 3.2

Here follows a listing and brief explanation of some of the commands which are new with the release of Xenix 3.2:

adj_clock -- adjust system clock — Allows the system administrator to adjust the system clock for small gains and losses of time.

autoboot -- set autoboot options — Allows the system administrator to set various autoboot options. This includes autoboot after reset, autoboot and reboot after crash (when possible), or normal manual boot.

cfg -- reconfigure kernel — Permits kernel reconfiguration by the system administrator without the Development System.

csh -- "C" shell — A command shell with C-language-like syntax, this is now released with the Core System. Previously was available only with the Development System.

csh1Ø -- "C" shell with Tenex-like extensions — A "C" shell with added features such as filename completion and substitution.

format -- on-line diskette and cartridge formatter — Allows diskettes and disk cartridges to be formatted while the system is running Xenix.

hello -- write to another user — Allows users to communicate with one another directly by writing to one another's terminals.

idle -- logs idle users off — Finds, warns, and ultimately logs off users who have been idle for more than a specific period of time.

init -- initialize terminals	Not a new command, but significantly different in Xenix 3.2, this now allows the use of a **/etc/inittab** file to set up special login arrangements.
lpq -- line printer queue status display	Displays the queue of spooled line printer tasks.
lprm -- remove queued line printer tasks	Allows users to remove spooled line printer jobs.
qps -- "quick" process status	A fast version of the **ps** program with no options.
vi -- screen oriented editor	A screen-based editor which lets the user display and edit text a "window" at a time instead of a line at a time.
w -- displays who and what is on a system	Prints a summary of who is currently on the system and what tasks they are performing.
xmail -- extended mail program	An improved version of **mail**, it allows the use of any system editor to edit mail. Also supports user-defined aliases, a local working mailbox, and an initialization file which can be set up to match a user's tastes.

****

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. If a command is present only in Xenix 3.2, I will note it so.

adj_clock **(Xenix 3.2 only)**
adjusts the system clock speed.

adj_clock +/-seconds /period[s|m|h|d]
tells the system to adjust for **seconds** gained (+) or lost (-) in **/period** (if unit not specified, system assumes hours).

autoboot **(Xenix 3.2 only)**
sets automatic reboot operation

autoboot -ar
turns on autoboot and reboot options

autoboot -a
turns on autoboot option only

autoboot -n
turns off autoboot and reboot options and allows normal boot

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

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 permissions
=	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":

```
-rw-rw-rw-  1 bob     group   85  Nov 18 1Ø:14  zeppo
 ¦¦ ¦¦ ¦¦
 ¦¦ ¦¦ ¦¦_____ write permission for others  = 2
 ¦¦ ¦¦ ¦_______ read permission for others   = 4
 ¦¦ ¦¦
 ¦¦ ¦¦_________ write permission for group   = 2
 ¦¦ ¦__________ read permission for group    = 4
 ¦¦
 ¦¦____________ write permission for owner   = 2
 ¦_____________ read permission for owner    = 4
```

If you add the results for each set together, you come up with the number that you gave **chmod**.

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.

csh **(Xenix 3.2 or Development System only)**
invokes a shell with C-language-like syntax

csh1Ø **(Xenix 3.2 only)**
invokes an enhanced **csh** with tenex-like file name and commmand completion

date displays the current date and time on the screen

df disk space free

df
displays the number of blocks free on your root file system

df /dev/hdx (where x = 1 to 3)
displays the number of blocks free on the specified disk

drive sets floppy drive parameters

/etc/drive nodetect wait seek=3
sets drive parameters so that any floppy drive can be used

/etc/drive detect wait seek=Ø
sets drive parameters to optimum for thinline drives

/etc/drive seek=2
sets drive parameters on Xenix Core Boot disk back so diskette can be used to initialize Model II Xenix system

du disk usage

du directory
displays the number of blocks in use on the specified directory and in what files

format **(Xenix 3.2 only)**
formats floppy or cartridge disks while Xenix is running

format fd*x*
formats the disk in floppy drive *x*, where *x* is between Ø and 3

format cd*x*
formats the disk in cartridge drive *x*, where *x* is between Ø and 1

hello **(Xenix 3.2 only)**
write to another user

hello username
copies lines from your terminal to that of *username*

kill terminate a process

kill number
terminates the process *number*

lc list, or display, the contents of a directory and information on files in that directory

lc directory-name
displays the contents of the specified directory

lpq **(Xenix 3.2 only)**
line printer spooler queue status display

lpq
prints the contents of the line printer queue

lpr printer spooler

lpr filename
prints the file on the printer

lprm **(Xenix 3.2 only)**
removes a currently non-printing entry from the printer queue

lprm id-number
removes the task id-number from the queue (provided it belongs to the user)

lprm username
removes all queued jobs belonging to username

lprm filename
removes the task filename from the queue

ls list, or display, the contents of a directory and information on files in that directory

ls directory-name
displays the contents of the directory

ls -l directory-name
displays the contents of the directory, including the names of all subdirectories and files, size, date created, and permissions.

mkdir make directory

mkdir directory-name
makes a directory

mknod builds special files

mknod /dev/tty09 c 0 9
builds a character device (indicated by c) with a major node of 0 and a minor node of 9 named /dev/tty09. This is the device for tty09 on II/12/16B/6000 Xenix systems.

more display text on the screen, 1 screen at a time

more filename
displays the contents of filename on the screen, 1 screen at a time.

mount mounts drives or displays mounted filesystem information

/etc/mount /dev/hd1 /mnt
mounts the first secondary hard drive (**/dev/hd1**) on the directory **/mnt**

/etc/mount
displays information regarding all currently mounted file systems

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

mv directory-name new-directory-name
changes the name of the specified directory

passwd change your login password

park park, or position, the read/write head assembly of the hard drive(s)

park -on
positions the heads on the last cylinder of the drive(s) upon shutdown

park -off
does not park the heads upon shutdown

ps process status

ps -e
displays information about all active processes

pwd print working directory

qps **(Xenix 3.2 only)**
"quick" process status report

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 file system

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

umount umounts filesystems

/etc/umount /dev/hd1
umounts the (presumably) currently mounted secondary hard drive **/dev/hd1**

w **(Xenix 3.2 only)**
displays who is currently on the system and what they are doing

wall write to all users on the system

who who is on the system

For commands which aren't listed here, try consulting either the online **help** pages or The System Administrator's Guide to Xenix.

****

Using the TRS-shell
or
"My friends call me 'tsh'..."

****

The TRS-shell (**tsh** -- pronounced "tee-shell") is a special command interpreter in Xenix that uses commands similiar to those found in TRSDOS. For those of you who want to get more familiar with Xenix, but don't know where to start, it presents a more familiar face. As you get more familiar with the system, you can wean yourself off and gradually start using the standard Xenix command interpreter (which incidentally is called the Bourne shell).

In order to use the **tsh**, you first have to invoke it. That turns out to be simple; at either a user prompt or a root prompt, type:

tsh <ENTER>

The screen will show one of the two following prompts:

```
tsh>          (if you logged in as a user)
tsh-#>        (if you logged in as root)
```

At this prompt, you can use any **tsh** or Bourne shell (**sh**) command.

Before you can log out of Xenix, you have to exit the **tsh**. To do this, type either:

exit <ENTER>

or

<CTL><D>

The screen will then show either the root prompt (**#**) or user prompt ($).

NOTE: If you want, you can even set things up so that your "login shell" (the command interpreter that interprets the commands you type in) is the **tsh**, but you keep a certain amount of flexibility if you don't.

A Brief Compendium of **tsh** Commands....

Despite all appearances to the contrary,
it's not the ol' Model II.

Here follows a (mostly) alphabetized list of the commands specific to the **tsh**. Anything in boldface is a required part of the command; things in normal print on the command line are optional.

again string ¦ history-number

Executes a previously executed command. You can specify the command by a search string or ("¦" in the command line means "or" -- a convenient shorthand) a history number. If you type **again** without any options, **tsh** executes the last command entered.

alias command command-line
unalias command

Maps a command name to another command line. This lets you give shorter or easier names to commands that you use often. If you omit command-line, **alias** displays the command line for that command. If you omit both command and command-line, **alias** displays the current lists of aliases. Use **unalias** to remove an alias.

auto command-line

Creates or removes an automatic command that executes when you enter **tsh**. The command-line is stored in the **.tshrc** file. This file executes when **tsh** starts up. If the file doesn't already exist, it is created.

To cancel the **auto** function, type:

auto <ENTER>

The automatic command line is removed from the **.tshrc** file.

cd directory (Xenix 3.2 tsh)
chdir directory

Changes the current working directory to the specified directory. If you do not specify a directory, your home directory becomes the current working directory.

cls

Clears the screen and places the cursor beside the **tsh** prompt.

copy file1 file2

Copies file1 to file2. If file2 is an existing file, the permissions and ownership do not change. If file2 is being created, the permissions and ownership are the same as the file being copied.

NOTE: The **copy** command will not copy a file onto itself.

copy file1 file2 ... directory

Copies multiple files into a directory. The filenames do not change.

NOTE: Suppose we have two files named **bozo** and **zeppo**, and a directory named **ouch**. The following examples may be instructive.

```
copy bozo clown  -- gives you a file named "clown" with the same
                    contents as bozo.
copy bozo zeppo ouch -- gives you two files in the directory "ouch"
                    named bozo and zeppo with the same contents as
                    their counterparts.
```

device

Displays a list of devices that **tsh** recognizes and their descriptions. You use the device names with the **dismount**, **mount**, **restore**, and **save** commands.

dir

Displays an alphabetic list of the contents of the current directory.

dir filename

Displays, in long format, the specified file. You can specify multiple files.

dir directory-name

Displays an alphabetic list of the contents of the specified directory. You can specify multiple directories

dir filename directory-name

Displays an alphabetic list of filenames and directory names; the list of filenames precedes the list of directory names. You can specify multiple filenames and directory names.

dir option filename
dir option directory-name

You can use the following options with the **dir** command.

-a Displays a list of all files. If you are logged in as root, Xenix assumes the -a option even if you do not specify it.

-b Displays invisible characters in octal notation.

-c Sorts or prints while creating files.

-d Displays only the directory name, not its contents, if you specify a directory name.

-g Displays the group ID instead of the owner ID in the listing.

-r Reverses the order of the sort, depending on the options you specify. For example, you may wish oldest date first or inverse alphabetic order.

-s Displays size in blocks, including indirect blocks, for each entry.

-t Sorts by time you modified the file or directory unless you specify the -c or -u options.

-u Sorts or prints according to last access time.

-A Displays a list of all files except for . and ..

NOTE: "." and ".." are current directory and parent directory respectively.

-R Displays the contents of all subdirectories for the specified directory.

dismount drivenumber -device

Tells Xenix that you are removing a mounted file system on the specified drive and device. You should execute the **dismount** command (if you are running only in the **tsh**) before removing a **mounted** disk.

display filename

Displays a full screen of text of the specified file. If the file contains more lines of text than the screen can show at one time, the following prompt appears at the bottom of the screen:

--more--

If you are reading a file, and not a pipe, a percent sign appears after the "more". This is the percentage of the file (in characters, not lines) read thus far.

To see the next line of text, press **<ENTER>**. To see the next full screen of text, press **<SPACEBAR>**.

Below are the options available for use with the **display** command. You use them in the following form:

display options filename

-n specifies the number of lines the screen displays, or the window size. If you do not specify **n**, Xenix uses the entire screen. (On a 24 line screen, the default is 22 lines).

-s "squeezes" extra blank lines from the screen, i.e. if you have inserted more than one blank line between lines of text that space is reduced to only one blank line. This option lets you display more useful information at a time.

-r displays control characters on the screen. For example, if the file contains **<CTL><C>**, the screen would show:

^C

If you don't specify the -r option, this won't happen.

-w waits to exit at the end of a file until you press **<SPACEBAR>**. If you do not specify **-w**, Xenix exits immediately.

+ linenumber displays the file, starting at the specified line number.

There are even more options available, but for more information, you should refer to **appendix a** of The System Administrator's Guide to Xenix.

do filename

Executes the commands stored in filename as if they had been entered from the keyboard. You may nest do commands as deep as necessary or as the system memory allows.

files directory-name

Lists in columns in alphabetical order all files in the specified directory. If you do not specify a directory, Xenix uses the current directory. You can specify multiple directories. When you do so, the screen shows the filenames first and then the directories.

The options available for use with this command are listed below. You use them in the following form:

files options directory-name

-a displays a list of all files. If you are logged in as root, all files are listed even if you do not specify the **-a** option.

-b displays invisible characters as /nnn (octal).

-c sorts and displays while a file is being created.

-d lists only the directory name and not its contents if you specified a directory.

-q lists group ID and not owner ID.

-r reverses the order of the sort, depending on the options you specify. For example, you can sort in reverse alphabetical order or by oldest date first.

-s lists size in blocks, including indirect blocks, for each entry.

-t sorts by time modified unless you specify the -c or -u option.

-u sorts and displays according to last access time instead of last modification.

-x displays in columnar format. Sorted material is displayed across rather than down.

-1 forces a one-entry-per-line output format.

-A displays a list of all files except those that have . and ..

-C forces multi-column output.

-R displays the contents of all subdirectories for the specified directory.

free filesystem

Displays the number of free blocks on the specified file system. If you do not specify a file system, the screen shows the amount of free space on all mounted file systems.

help

Displays an explanation of the help text and the notations used. To see a list of available subjects, type **help * <ENTER>**. To see the help text for a particular subject, type **help** subject **<ENTER>**.

To halt the display of the help text file, type **q**.

history

Displays a list of the previously executed commands. **history** assigns each command a number which can be used to re-execute it using the **again** command. **history** is set to display the last 2Ø commands. (See also **again** and **set**).

kill filename

Kills (removes) the specified file from a directory. The file is destroyed. You must have write permission for a directory to delete files from it. You do not, however, need read or write permission for the file to be able to delete it.

You may specify multiple filenames. If you do so, Xenix prompts you for each file before the file is removed. Type **y <ENTER>** in response to the prompt.

To delete a directory, type **kill -r** directory-name. This deletes the contents of the directory and the directory itself. If you attempt to delete a file that is actually a directory, the screen shows an error message.

NOTE: You cannot remove the file ".." from a directory.

lib

Displays a list of the **tsh** commands.

mount drive number-device
dismount drive number-device

Tells Xenix that a file system is now present on the specified drive and device. The file system's contents are made available in the **/mnt**drive-number directory. **device** specifies the type of device you are mounting (ss, ds, c1Ø). See **device** for more information. If you omit **device**, **tsh** assumes floppy diskette.

For example, if you place a floppy diskette containing a file system in floppy drive Ø, you can mount the disk by typing **mount Ø <ENTER>**. The contents of the disk will be accessable in the directory /mntØ. You tell Xenix that you are removing the diskette from the drive by typing **dismount Ø <ENTER>**.

The **mount** and **dismount** commands keep track of mounted devices. If you do not specify a drive number, the screen shows a table of currently mounted file systems.

NOTE: You can only mount disks that contain Xenix file systems. If you attempt to mount a diskette that contains an alien file system (for example, TRSDOS) you will at least get error messages and the possibility even exists that the system will crash.

move -i file1 file2
move -i file1 file2 ... directory-name

Copies file1 to file2. If file2 already exists, the existing permissions and ownership are retained. If it does not exist, the permissions and ownership of the source file become those of file2.

-i is optional. If you specify -i, Xenix lets you know if the destination file already exists. You can then choose to keep or overwrite the existing file.

You can copy multiple files into a directory with the second command line above. The filenames are not changed.

NOTE: You cannot copy a file onto itself with the move command.

print filename

Queues the file for printing on the printer. You can place multiple filenames on the command line.

Listed below are the options you can use with the print command. To use the options, follow this format:

print option(s) filename(s)

-T local	sends output to local printer regardless of how PRINTER was set.
-T spooler	sends output to the print spooler for printing.
-r	removes the file after it is queued.
-c	copies the file so that no changes are made before it is printed.
-m	reports by **mail** when printing is complete.
-n	does not report by **mail** when printing is complete. This is the default.

rename old.file new.filename

Changes old.file's name to new.filename.

restore drivenumber -d -device
restore drivenumber -device filename ...

Restores files that were saved on diskette/cartridge to where they were originally. With the **restore** command, you may use multiple filenames or directory names. If you specify a directory name, the directory and its subdirectories, files, and their contents are restored.

Device specifies the type of device from which you are restoring (ss, ds, c1Ø -- and in Xenix 3.2, also c2Ø). See **device** for more information.

-d is optional. If you specify -d, the screen shows a list of files on the diskette/cartridge.

save drive-number -device filename ¦ -all

Saves files or directories and their contents to the disk in the specified drive. You may use multiple filenames or directory names. If you specify a directory, the directory and its subdirectories, files, and their contents are saved.

Specify the -all option if you want to save the entire filesystem, including all mounted filesystems.

Device specifies the type of device to which you are saving (ss, ds, c1Ø -- and in Xenix 3.2, also c2Ø). See **device** for more information.

-ss tells Xenix that you are using single-sided diskettes. -ds specifies double-sided diskettes. -c1Ø specifies a 1Ø meg cartridge drive. -c2Ø, in Xenix 3.2, specifies a 2Ø meg disk cartridge drive. You must tell Xenix one of the above.

set

Displays a list of the internal option settings of **tsh**.

set option

Changes an internal option. The following options are available.

verbose on ! off If verbose is off, **tsh** does not inform you of a command's success or failure.

retry on ! off Enables or disables retry. If retry is on, the command line is converted to lower case if **tsh** is unable to execute a command. **tsh** then tries to execute the lower case command line.

prompt string Set the **tsh** prompt to string. For example, to set the prompt to:

```
cue>
```

type:

```
set prompt cue> <ENTER>
```

term terminal-type Sets the user's terminal type by setting the environment variable TERM.

history number Sets the number of commands that **history** remembers to number.

showhistory on ! off If showhistory is on, **tsh** displays each command and its history number as soon as you enter it.

showstatus on ! off If showstatus is on, **tsh** displays the exit code if a command fails.

showdir

Displays the pathname of the current working directory.

source file

Reads the command to execute from the specified file. **source** reads and executes the commands as if you had typed them from the keyboard. **source** differs from **do** in that the commands are executed from the current shell. **tsh** does not display the commands or record them in the history list.

time

Displays the current time and date.

version

Displays the name of the shell and its current version.

****

FILE GUIDE

****

In this appendix, I'll give you 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
/mnt	mount directory (reserved for mounted file systems)
/usr	user home directories
/tmp	temporary directory (reserved for temporary files created by programs)
/lib	directory containing system date utility
/diskutil	disk formatter program
/xenix	xenix system object code
/z8Øctl	z8Ø control program
/z8Ødiags	exists in 3.1 and 3.2, not in 3.Ø... additional z8Ø programming

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:

cp	login	sh	tar
date	mv	stty	test
echo	passwd	su	
fsck	rm	sync	

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/fd**x**	floppy disk drives
/dev/hd**x**	hard disk drives
/dev/cd**x**	cartridge drives (Xenix 3.1 and 3.2)
/dev/r**xx**	unbuffered interface to corresponding device name
/dev/root	root file structure
/dev/swap	swap area
/dev/tty**x**	terminals
/dev/recover	used to recover kernel messages after shutdowns (Xenix 3.2)

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 put local mods in /etc/rc.user)
/etc/ttys	terminal setup
/etc/termcap	terminal capability map
/etc/motd	message of the day

The following directory contains files that may be modified but not removed:

/etc/default	defaults for various programs

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, are not essential for system operation, or are more user friendly
/usr/lib	contains more libraries and data files used by various Xenix commands
/usr/spool	contains directories for storing files to be printed, and user's mail boxes
/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/group**

Purpose: To designate group 'names'

Sample File:

```
other:x:1:
sys:x:2:sys
bin:x:3:bin
uucp:x:4:
group::5Ø:
payroll:sV23xdf:53:mark,tom,dave,fred
```

- `payroll` — group name
- `sV23xdf` — group password (encrypted)
- `53` — group number
- `mark,tom,dave,fred` — login names of persons allowed in this group

File: **/etc/logbook**

Purpose: To show what Radio Shack programs and updates have been installed on a customer's system

Changes: None. Do NOT change this file unless absolutely necessary.

Sample File:

```
Ø1.Ø2.ØØ 7ØØ-2Ø52  TRS-XENIX Core System
Ø1.Ø3.ØØ 7ØØ-2Ø52  TRS-XENIX Core System
Ø1.Ø3.Ø2 7ØØ-2Ø52  TRS-XENIX Core System
Ø1.Ø3.Ø3 7ØØ-2Ø52  TRS-XENIX Core System
Ø1.Ø3.Ø4 7ØØ-2Ø52  TRS-XENIX Core System
Ø3.ØØ.ØØ 7ØØ-3Ø3Ø  XENIX Multi-User O/S
Ø2.Ø2.ØØ TERMCAP   XENIX Terminal capabilities    Wed Jul 18 17:ØØ:ØØ CDT 1985
Ø3.Ø1.ØØ 7ØØ-3Ø3Ø  XENIX Multi-User O/S           Mon May 15 17:ØØ:ØØ CDT 1985
Ø3.Ø1.ØØ HELP      On-Line Help Pages (Core)      Wed Jul 31 11:52:32 CDT 1985
Ø1.Ø1.ØØ 26Ø-6431  Scripsit 16                    Thu Sep 12 Ø9:39:12 CDT 1985
```

- `Ø1.Ø1.ØØ` — Product version number
- `26Ø-6431` — Product name and catalog number
- `Thu Sep 12 Ø9:39:12 CDT 1985` — Date/time of installation

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:1Ns25tlcfkHnQ:Ø:Ø:The Super User:/:/bin/sh
cron:NOLOGIN:1:1:Cron Daemon for periodic tasks:/:
daemon:NOLOGIN:1:1::/:
sys:NOLOGIN:2:2::/usr/sys:
bin:NOLOGIN:3:3:The owner of system files:/:
uucp:x:4:4:Account for uucp program:/usr/spool/uucppublic:/usr/lib/uucp/uucico
uucpmgr:x:4:4:uucp manager's account:/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:
who::18:18::/bin:who
bob::2Ø1:5Ø:bob davis:/usr/bob:/bin/sh
tom::2Ø2:5Ø:tom smith:/usr/tom:/bin/sh
rick:jk34j33nhz2:2Ø3:5Ø: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 (Accounts Payable)
h	allow access to the command interpreter (shell)

File: /etc/rc, /etc/rc.user, /etc/rc.appl

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; applications modifications are located in /etc/rc.appl. If you want to change the boot message, you might edit /etc/rc.

Sample Files:

Xenix 3.1.x /etc/rc

```
PATH=/etc:/bin:/usr/bin
echo "→L
Tandy 68ØØØ/XENIX version `cat /etc/version`
Microsoft XENIX-68ØØØ  V3.Ø

Copyright Microsoft Corporation, 1983.  All rights reserved.,
Licensed to TANDY CORPORATION.
Restricted rights: Use, duplication, and disclosure are subject
to the terms stated in the customer Non-Disclosure Agreement.

\"tsh\" and \"tx\" Copyright 1983 TANDY CORPORATION.  All rights reserved.
" > /dev/console
TZ=CST6CDT
export TZ                                                              a
HZ=3Ø
export HZ
/etc/verify -f y
/etc/verify -h y                                                       b
cp /dev/null /etc/mnttab; chmod 644 /etc/mnttab
/etc/setmnt <<!
root /
!
cp /dev/null /etc/utmp; chmod 644 /etc/utmp
/etc/asktime </dev/console >/dev/console 2>&1                          c
/etc/update                                                            d
> /usr/adm/msgbuf
/etc/dmesg - >> /usr/adm/messages
/usr/lib/ex3.7preserve -
/usr/lib/xmail_recover
if test -f /etc/rc.user
then
        sh /etc/rc.user < /dev/console > /dev/console 2>&1             e
fi
```

```
if test -f /etc/rc.appl
then
        sh /etc/rc.appl
fi
rm -f /usr/spool/lpd/lock; /usr/lib/lpd
rm -f /tmp/* /usr/tmp/*
/etc/cron
mv /usr/adm/wtmp /usr/adm/owtmp
cp /dev/null /usr/adm/wtmp; chmod 644 /usr/adm/wtmp
rm -f /usr/spool/micnet/remote/pids
rm -f /usr/spool/uucp/LCK.*
```

Xenix 3.2 /etc/rc

```
PATH=/etc:/bin:/usr/bin
echo "→L
Microsoft XENIX-68ØØØ  V3.Ø

Copyright Microsoft Corporation, 1983.  All rights reserved.
Licensed to TANDY CORPORATION.
Restricted rights:  Use, duplication, and disclosure are subject
to the terms stated in the customer Non-Disclosure Agreement.

\"help\", \"tsh\", \"tx\", \"xmail\", \"cfg\" and \"format\"
Copyright 1983, 1987 TANDY CORPORATION.  All rights reserved.
" > /dev/console

if [ "$1" = "-r" ]; then
        /bin/echo "[Automatic reboot in progress]" > /dev/console
fi

TZ=CST6CDT
export TZ
HZ=3Ø
export HZ

/etc/verify -f y
/etc/verify -h y

cp /dev/null /etc/mnttab; chmod 644 /etc/mnttab
/etc/setmnt <<!
root /
!

cp /dev/null /etc/utmp; chmod 644 /etc/utmp
```

```
if [ "$1" = "-r" ]; then
        date="`date`"
        /bin/echo "Current system time is: $date" > /dev/console

        /etc/dmesg -r | tee /tmp/boot.log >> /usr/adm/messages
        /bin/echo "Current system time is: $date" >> /tmp/boot.log
        /etc/fsck_recover -c >> /tmp/boot.log
else
        /etc/asktime </dev/console >/dev/console 2>&1
        /etc/dmesg -r >> /usr/adm/messages
fi                                                                    c

/etc/update                                                           d

> /usr/adm/msgbuf
/etc/dmesg - >> /usr/adm/messages

/usr/lib/xmail_recover

/usr/lib/ex3.7preserve -

if [ -f /etc/rc.user ]; then
            if [ "$1" = "-r" ]; then
                 sh /etc/rc.user $* | tee /dev/console >>/tmp/boot.log 2>&1
            else
                 sh /etc/rc.user $* </dev/console > /dev/console 2>&1
            fi
fi                                                                    e

if [ "$1" = "-r" ]; then
        xmail -s "Auto-boot log" root < /tmp/boot.log
fi                                                                    j

if [ -f /etc/rc.appl ]; then
        sh /etc/rc.appl $*
fi                                                                    f

rm -f /usr/spool/lpd/lock; /usr/lib/lpd                               g
rm -f /tmp/* /usr/tmp/*                                               h
/etc/cron                                                             d
mv /usr/adm/wtmp /usr/adm/owtmp
cp /dev/null /usr/adm/wtmp; chmod 644 /usr/adm/wtmp
rm -f /usr/spool/micnet/remote/pids
rm -f /usr/spool/uucp/LCK.*
```

/etc/rc.user

```
mount /dev/hd1 /h1 _______
mount /dev/hd2 /h2 _______!_____________________________________ i
```

a	System startup message
b	Turn floppy and hard drive verify on
c	Time prompt
d	Start additional background processes
e	Execute instructions in /etc/rc.user if it exists
f	Execute instructions in /etc/rc.appl if it exists
g	Remove printer lock; start printer daemon
h	Clear out temporary directories
i	Mount secondary filesystems
j	Send root mail regarding autoboot (Xenix 3.2 only)

File: **/etc/systemid**

Purpose: To define system name during UUCP (Unix to Unix Communications Programs) operations

Typical Changes: During UUCP each machine must have a unique name.

Sample File:

trs16

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:

```
19console
13tty01
19tty02
09tty04
09tty05
09tty06
||  |
||  |_________port name
||____________baud rate
|_____________status (set by enable/disable command)
                    (0 = disabled, 1 = enabled)
```

Note: See Section Three 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:

```
trs16 console
adds25 tty01
dt100 tty02
adds25 tty04
adds25 tty05
adds25 tty06
  |      |
  |      |_____________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:

```
*****************************************
*                                       *
*          Welcome to TRS-XENIX         *
*                                       *
*****************************************
```

File: **/etc/default/dump, /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/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/idle** (Xenix 3.2 only)

Purpose: Specifies defaults for the **/usr/lib/idle** program.

Typical Changes: Maximum allowed idle times

Sample File:

```
NULLFILE /etc/idle.pgms
LOGFILE  /usr/adm/logouts
C_LIMIT  5Ø
CD_LIMIT 2Ø
CB_LIMIT 24Ø
CDB_LIMIT 6Ø
CHECK_INTERVAL 1Ø
```

File: **/etc/default/login**

Purpose: Specifies default environmental variables for **login**

Typical Changes: Change default umask or paths

Sample File:

```
TIMEZONE=CST6CDT
HZ=3Ø
ALTSHELL=YES
UMASK=Ø22
PATH=:/bin:/usr/bin:/usr/local
SUPATH=:/bin:/usr/bin:/usr/local
```

File: **/etc/default/lpd**

Purpose: Contains banners on/off information, also printer mode information

Typical Changes: Turn banners off

Sample File:

```
BANNERS=1 ____________________  (1=on, Ø=off)
FILTER=NO
```

File: **/etc/default/micnet**

Purpose: Command file for Micnet system

Typical Changes: Restrict or increase remote command privileges

Sample File:

```
executeall
execpath=PATH=/bin:/usr/bin
```

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 hard disk, you might specify "/mnt1/usr" 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; sets default PATHs.

Typical Changes: Invoke accounting on **su**; change default paths.

Sample File:

```
#SULOG=/usr/adm/sulog ____________________________________ a
#CONSOLE=/dev/console ____________________________________ b
PATH=.:/bin:/usr/bin:/usr/local
SUPATH=.:/bin:/usr/bin:/etc:/usr/local
```

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.

File: **/etc/default/tar**

Purpose: Contains default settings for numeric arguments to **tar**

Typical Changes: If customer has a cartridge drive, the file may be edited and device, block, and archive size changed to reflect this.

Sample File:

```
#          device      block     size
archiveØ=/dev/rfdØ       16        6Ø8
archive1=/dev/rfd1       16        6Ø8
archive2=/dev/rfd2       16        6Ø8
archive3=/dev/rfd3       16        6Ø8
archive4=/dev/rfdØ       16        1224
archive5=/dev/rfd1       16        1224
archive6=/dev/rfd2       16        1224
archive7=/dev/rfd3       16        1224
archivef=/dev/null       1         Ø
```

File: **/etc/default/tsh**

Purpose: Contains device information for tsh "save" command

Special Notes: **If the user is running Xenix 3.2, and wishes to run a 2Ø meg DCS, they must change the following:**

```
       Original: c2Ø:  cd      2Ø      2Ø896   2Ø-Megabyte Cartridge Disk
     Change To: c2Ø:  cd      16      2Ø896   2Ø-Megabyte Cartridge Disk
```

Sample File:

Xenix 3.1:

```
# Option        Dev     Block   Output-Volume-size (in K)
c1Ø:            cd      16      976Ø
```

Xenix 3.2:

```
# Device information for tsh "save" command:
# Option        Dev     Block   Volume size(K)  Description
c1Ø:            cd      16      976Ø            1Ø-Megabyte Cartridge Disk
c2Ø:            cd      16      2Ø896           2Ø-Megabyte Cartridge Disk
```

File: **/usr/lib/crontab**

Purpose: To define times and dates for automatic program execution

Typical Changes: Clean out administrative files on a given date
Start posting procedures during weekend

Sample File:

```
5,15,25,35,45,55 *      *       *       /etc/dmesg - >>/usr/adm/messages
Ø,3Ø    *       *       *       *       /usr/lib/atrun
2Ø      1       *       *       *       /usr/bin/calendar -
¦       ¦       ¦       ¦       ¦               ¦
¦       ¦       ¦       ¦       ¦               ¦_________ command to execute
¦       ¦       ¦       ¦       ¦                          (use full pathname)
¦       ¦       ¦       ¦       ¦_____________________ day of the week
¦       ¦       ¦       ¦                              (Ø-6)
¦       ¦       ¦       ¦_____________________________ Month of the year
¦       ¦       ¦                                      (1-12)
¦       ¦       ¦_____________________________________ Day of the month
¦       ¦                                              (1-31)
¦       ¦_____________________________________________ Hour (Ø-23)
¦_____________________________________________________ Minutes (Ø-59)
```

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
(3.1.x and 3.2)

A Compendium of Common and Uncommon
Xenix Errors Messages
(And a Brief Explanation of Each)

****

Supervisor Traps and What They Mean:

Below you will find listed 68ØØØ Trap Errors and some explanation of their meaning.

Trap #	Assignment
2	Bus Error
3	Address Error
4	Illegal Instruction
5	Zero Divide
6	CHK Instruction
7	TRAPV Instruction
8	Privilege Violation
9	Trace
1Ø	Line 1Ø1Ø Emulator
11	Line 1111 Emulator
12-14	Unassigned, Reserved
15	Uninitialized Interrupt Vector
16-23	Unassigned, Reserved
24	Spurious Interrupt
25	Level 1 Interrupt Autovector
26	Level 2 Interrupt Autovector
27	Level 3 Interrupt Autovector
28	Level 4 Interrupt Autovector
29	Level 5 Interrupt Autovector
3Ø	Level 6 Interrupt Autovector
31	Level 7 Interrupt Autovector
32-47	Trap Instruction Autovectors
48-63	Unassigned, Reserved.

I can't promise that I'll explain all of the above; however, I should touch at least lightly on most of the frequently seen ones.

First, a little explanation of what Supervisor Traps are might be in order. "Traps are exceptions caused by instructions" -- that's a quote directly from the Motorola data. They are caused either by the 68ØØØ recognizing an abnormal condition during instruction or execution, or by use of instructions whose normal behavior is trapping (for example, instructions which allow you to trace program progress... kind of like "TRON" and "TROFF" in BASIC).

One class of Trap might be called the **Instruction Trap**. This class includes things like Supervisor Traps 5, 6, and 7 (Zero Divide, CHK Instruction, and TRAPV Instruction respectively). These are fairly self-explanatory; if you try dividing by zero, you'll provoke a trap, and the CHK and TRAPV instructions provoke traps if the user program detects a runtime error like an arithmetic overflow or a subscript out of bounds. These errors can be created by bad software, or by addled hardware.

Another type of trap is that which treats with **Illegal and Unimplemented Errors**. An illegal instruction is one in which the first word bit pattern is not the first word bit pattern of a legal instruction. Supervisor Traps 4, 10, and 11 (Illegal Instruction, Line 1010 Emulator, and Line 1111 Emulator) fall into this category. Traps 10 and 11 (the emulator traps) are very specific; they deal exclusively with word patterns with bits 15 through 12 equaling 1010 or 1111. These words are distinguished as unimplemented instructions, and separate exception vectors are provided for these to increase efficiency. These are typically signatures of freaked memory, although you can write software which will provoke this.

A **Privilege Violation** is caused when an attempt is made to execute one of the so-called "privileged instructions" while in the user state. Essentially, you're trying to do something which you aren't authorized to do, and the 68000 is letting you know in no uncertain terms. This error is rare, and shows up as Supervisor Trap 8.

Supervisor Trap 9 (**Trace**) is a programming aid provided by the 68000. You should almost never see this one... and if you do, I'd suggest a good hard look at the 68000 CPU board.

Supervisor Trap 2 (**Bus Error**) is a commonly seen problem. What is actually happening is that the external logic connected to the 68000 has detected a bus error. It will then request the 68000 to process said bus error by generating an exception (a.k.a. trap). This can be caused by a lot of things, ranging from bad cables between the 68000 CPU and memory boards to something on the Z80 side of the machine grabbing the bus and refusing to let go. The one consistent thread is the fact that it is almost always a hardware problem.

Supervisor Trap 3 (**Address Error**) occurs when the 68000 attempts to access a word or a long word operand or instruction at an odd address (the 68000 runs on even address boundaries). This is usually a sign of a confused 68000 or flaky memory, although other portions of the 68000 circuitry can be suspect as well.

.....

Xenix 3.1.x Error Messages

For Xenix versions 3.1.Ø, 3.1.1, and 3.1.2
(also apply to some extent to Xenix 3.Ø)

.....

Xenix 3.1.x Kernel Error Messages

Here follows a description of the warning and error messages put out by various parts of the kernel, and what they mean. The error messages put out by the 28Ø Control Program are not in this list; they will be listed seperately later in this document. Included with the description of each message and its format are possible causes and their locations.

This information is only totally accurate for kernel versions 3.1(13) and later. For any later versions of Xenix System 3 which may come along the information presented here may be incomplete, but it will generally be correct. Complete error messages for Xenix 3.2 will be listed in a separate section.

Kernal error messages may be categorized as follows:

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 doing something contradictory, eg, write-protecting a hard drive and trying to mount it for write.

System Startup Messages

The following text is diplayed every time the system is booted:

```
Tandy 68ØØØ/XENIX version 3.n(ee)

Microsoft  XENIX  v3.Ø

Copyright Microsoft Corporation, 1984.  All rights reserved.
Licensed to Tandy Corporation.
Portions copyright 1985 Tandy Corporation.  All rights reserved.
Restricted rights:  Use, duplication, and disclosure are subject
to the terms stated in the customer Non-Disclosure Agreement.

System nnnk User nnnk
Root nnnnk Swap nnnnk
```

The first line tells you what version of the kernel you are running. The 3.n(ee) tells you that you are running release n of 3.Ø Xenix. The number in parenthesis is the edit number of the kernel. In this way, you may tell the difference between various versions of the same release (not that you should ever need to, but... neat to know anyway).

After the copyright warning, the system displays how much memory it is using, how much memory is available for users, how big the disk it is booted on is, and how much swap area is available.

Device Driver Messages

Device drivers display error messages in four different formats (two of which are special and related to cartridge disk operation only):

```
device DRIVE n: specific error message
device DRIVE n: type err hhhh on rd/wrt of cyl cyl hd head sect sector
CART DRIVE n: type err hhhh on rd/wrt of blk block
CART DRIVE n: type err hhhh on format
```

device **DRIVE** n: specific error message

This format is used for general errors from the driver which affect all operations. For example, the message "HARD DRIVE Ø: active drive not ready" is displayed when the driver tried some hard disk I/O operation on an already active (open) drive, and the controller reported the drive as busy or not ready.

The following messages are possible from each type of drive:

Further I/O aborted until device closes
Device closed; error cleared
active disk changed
active drive not ready
active drive write protected
bad format
bad sector size
drive not ready
write protected

Further I/O aborted until device closed
Device closed; error cleared
These messages are seen when an unrecoverable error has occurred on a drive 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.

active disk changed
This error occurs when a disk is changed while Xenix was doing I/O on it (i.e., it was mounted). This message is normally seen on floppy drives. It cannot occur on hard drives, and is very difficult to cause on cartridge drives. User Error.

active drive not ready
This error occurs when Xenix tried some I/O operation on an already active (open) drive, and the controller reported the drive busy or not ready. Hardware.

active drive write protected
This error occurs when, after the drive was opened for I/O, the user write-protects the media. This error most often occurs on a hard disk, where the error is ignored until the problem is corrected (i.e., the user removes the write-protection). It is very difficult to create this error on a floppy or cartridge disk as the write-protection devices on these are inaccessable while they are in their drives. User Error.

bad format
This error indicates that when Xenix first looked at a disk to open it for I/O, the media format was incorrect. This usually indicates that the user did not format the media with **diskutil** before attempting to use it. User error.

bad sector size

This is a very unusual error. It indicates that when Xenix looked at a disk to open it for I/O, it had an unusual (i.e., non-standard) sector size. This error is unusual because for raw I/O, Xenix will handle most sector sizes supported by the controller. This error usually indicates something is wrong with the Z8Ø hardware. Hardware.

drive not ready

This error is displayed when Xenix first looks at a drive to do I/O and the controller says the drive is not ready. This is a common error when, for example, the user tries to read a floppy or cartridge drive with no disk in it. This error is more serious on a hard drive, and usually indicates hardware problems. User Error or Hardware.

write protected

This error is displayed when Xenix first looks at a drive to do I/O and the user wants to write to the drive and the media is write-protected. User error.

device **DRIVE** n: type **err** hhhh **on rd/wrt of cyl** cyl **hd** head **sect** sector
CART DRIVE n: type **err** hhhh **on rd/wrt of blk** block
CART DRIVE n: type **err** hhhh **on format**

This format is used for errors which occur during I/O of some sort. You are shown the type of error, some hardware-related status, what type of I/O it occurred on (read, write, or format for disk cartridge), and where it occurred. The type of error can be one of the following:

hard
soft
mailbox
unclassified

A hard error indicates that the driver retried the operation and was unable to complete the requested I/O.

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 mailbox error is a rare error indicating that Xenix handed the Z8Ø control program an invalid request of some sort. This usually indicates a bug in the 68ØØØ software, or, in extreme cases, a 68ØØØ hardware problem.

An unclassified error is an extremely rare error type indicating that the Z8Ø control program returned an undefined error type and Xenix was unable to determine what the problem was. This usually indicates a Z8Ø hardware problem, or, in some cases, a bug in the Z8Ø control program.

The status displayed (hhhh) is data that might be useful from the controller or drive. For the hard drive, the data displayed is the error register and the control/status register from the controller. For example, you might see a **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 is the drive select port and the FDC status port. For the cartridge drive, the data displayed is 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.

Specific or Specialized Device Driver Error Messages

dkustart: GO set on device drive n
panic: GO set at dkustart
These two messages are produced by the disk driver when Xenix asks for some I/O and it discovers that the Z8Ø already claims to be busy doing something which Xenix doesn't think it asked for. This usually indicates a bug in the 68ØØØ software. System Inconsistency, Fatal.

panic: disk major number
This message will only be seen when Xenix is put together improperly, or there is an unusual 68ØØØ hardware problem. It indicates that when Xenix started up, it was unable to find the disk driver listed in the driver tables. System Inconsistency, Fatal.

Miscellaneous Hardware-Related Messages

Supervisor Trap d
Access address=dddddddd **Instr wd=**iiii
&dØ=dddddddd **mmu=**dddddddd **sr=**dddddddd **pc=**dddddddd **usp=**dddddddd
aØ-a7 dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd
dØ-d7 dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd dddddddd
panic: trap in sys

This message indicates that 68ØØØ Trap Number d occurred in supervisor mode (i.e., while the processor was running Xenix code and not user code). For specifics on the trap, consult the first pages of this section.

This error can be caused by a vast menagerie of problems, ranging from bugs in Xenix to 68ØØØ hardware problems. The data presented is a nearly complete dump of the processor's registers and may be used to find where the 68ØØØ was executing when the trap occurred. Hardware or System Inconsistency, Fatal.

panic: memory parity

This message occurs when a 68ØØØ board set equipped with parity chips gives a parity trap. There is nothing that can be done to correct the error, so Xenix ceases processing. Hardware or System Inconsistency, Fatal.

panic: mmusub: chk

This message indicates that, while trying to allocate space for a user program, Xenix discovered that a previously checked and "ok" program has an illegal or impossible size or illegal segments (separate I/D or stack segments). System Inconsistency, Fatal.

interrupt from unknown device, vec=d
panic: unknown interrupt

This message indicates that an unknown, unused interrupt (number d) occurred on the 68ØØØ. This usually indicates a 68ØØØ, or, more rarely, a Z8Ø hardware problem. Hardware, Fatal.

Other System Messages

**** ABNORMAL System Shutdown ****
This message appears when errors occur during 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 this 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.

error on dev name(nn/mm)
This is a way that device driver diagnostic messages can start. In reality, none of Tandy's drivers use this format. This style of message is almost never seen. The message will indicate the specific driver and complaint. The name is a word indentifying the device. (See above section on device drivers).

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 is refused. Although not fatal to the system, this event may damage the operation of various spoolers, daemons, the mailer (if present) and other important utilities. Anomalous results and missing data files are a common result. Abnormal.

no file
There are too many open files, and 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. In order to determine which file system is suffering from the problem, use the "nn/mm" portion of the message to find the device. "nn" is the major node of the affected device, and "mm" is the minor node. For example, a "no space on dev 1/33" could be used with a listing of the /dev directory to allow you to find:

```
brw-r----- 1 sys     1, 33 Sep 23 13:25 /dev/root
```

/dev/root is the root file system, and is located on the primary hard drive.

Although not normally as serious, the warnings discussed for "inode table overflow" apply: often user 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 ****
[Z8Ø Control System Halted] ++

This message appears when the system has been shut down properly. It indicates that the machine may now be rebooted or powered down.

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 the file system is created (using mkfs). 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 file system to a backup device, run mkfs with more inodes specified, and restore the files from backup. Abnormal.

out of text

When shared text programs 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 (i.e., those programs which have the sticky bit set -- for more information, read the help page on chmod) requires a permanent entry in this table; nonsticky pure text programs require an entry only when there is at least one copy being executed. Abnormal.

++ See also the portion of this document addressing /z8Øctl error messages.

panic: blkdev
An internal disk I/O request, already verified as valid, is discovered to be referring to a nonexistent disk. System Inconsistency, Fatal.

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. Hardware, Fatal.

panic: IO err in swap
A hard++ I/O error occurred while reading or writing the swap area. Hardware, Fatal.

panic: no fs
A file system descriptor has disappeared from its table. System Inconsistency, Fatal.

panic: no imt
A mounted file system has disappeared from the mount table. 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 that is scheduled to be swapped. The system refuses to create tasks when it feels there is insufficient disk space, but it is possible to create situations to fool this mechanism. Abnormal, 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. This is usually caused by a bug in a device driver. Abnormal, Fatal.

proc on q
The system attempts to queue a process already on the process ready-to-run queue. System Inconsistency, Fatal.

++ See the above section on device driver messages.

System 3 /z8Øctl Error Messages

The following information is a rather complete overview of the error messages that the Z8Ø control program (Version 3(56)) will produce. They are only guaranteed to be accurate for that version, although most of this will be fairly applicable for other System 3 releases of the 3.Ø and 3.1.x family. Complete Xenix 3.2 errors will be listed later in this document.

Having said that...

.....

Everything You Ever Wanted To Know
About Z8Ø Error Messages
But Were Too Freaked Out To Ask

or

"Shut 'er down, Scotty, she's sucking mud again."

.....

The **z8Øctl** program (found in /z8Øctl) is a piece of code that is loaded into Z8Ø memory when the boot track loads Xenix into the 68ØØØ memory. This code resides in the lower 32K of Z8Ø RAM. In addition, over 1ØK of additional Z8Ø RAM is used for buffers, control blocks, and interrupt vectors. The purpose of **z8Øctl** is to perform I/O related operations for Xenix, since the 68ØØØ (as we use it) is unable to access the hardware itself. Therefore, **ALL** I/O, including disk, terminal, console, and various peripherals is performed by the Z8Ø, acting on a command packet from the 68ØØØ.

Since **z8Øctl** is really a collection of drivers and a scheduler, it only displays error messages of its own when there is no reasonable way to inform the user under Xenix of the problem. Xenix provides a limited number of errors to return to the user (and you just saw them all in the preceeding portion of this appendix), so the nature of an error could be hidden. When **z8Øctl** displays an error message, it always goes to the console, and is preceded by one of the following:

Buginf:	-- Bug Information
Bugchk:	-- Bug Check
Bughlt:	-- Bug Halt

Buginf is used to inform the user of a non-normal situation that arises which the driver will attempt to handle using some pre-coded default. An example of this is booting off of a hard disk that has no bad track map. This will generate a **Buginf** and the boot will try to start the system anyway. In this case, the boot makes the assumption that there are no bad tracks at all on the drive. This assumption may be a bad one, so the message is displayed to warn you that problems may occur. **Buginf** does not occur in Xenix 3.1; this will only show up in earlier versions (as of this writing).

Bugchk indicates that either some synchronization problem arose that should not happen or an unusual hardware error occurred. In the case of a synchronization problem, the request is usually ignored. This may cause any further reference to the device to hang, but this is usually preferable to what **z8Øctl** refused to do. For hardware problems, the request is aborted and an error returned to the 68ØØØ, who usually retries the operation.

Bughlt means that an error was detected that is so serious that a recovery should not even be attempted. For example, when the 68ØØØ makes a disk request, two copies of the request are passed to the Z8Ø. One is inverted. When the Z8Ø detects the request, it compares the copy to the main request. If the two do not agree, then some problem has occurred, the error message is displayed, and the system halted to avoid damage to existing data. When a **Bughlt** occurs, the system goes into a halt state, and halts the 68ØØØ as well. No drive park is performed (if it has been invoked).

If an error message that is displayed on the console does not start with "**Bugxxx**", then it is likely that it came from Xenix. Only a few **z8Øctl** messages in Xenix 3.Ø do not start with one of the bug headers; all of them do in Xenix 3.1.x.

When **z8Øctl** is loaded, it is executed and after initializing its drivers it displays a message like:

[Z8Ø Control System Version 3(56) 28-Jun-85] [Model xxxx]

This is the version and the date code for the version of **z8Øctl** which you are presently running. The [Model xxxx] will display "II" or "6ØØØ/16", based on the results of a test done with one of the onboard CTCs. All Model 12/16B/6ØØØ units will display "6ØØØ/16". On Model II/16[A]'s, the "II" will be displayed if the Z8Ø CPU board revision is Revision A, B, or C. Revision D boards (and later) allowed the 68ØØØ to interrupt the Z8Ø. **z8Øctl** determines if the interrupt is available and if it is it honors it. If the CPU board does not have that circuit, the Z8Ø polls a location in 68ØØØ memory to determine if it is being requested to do something.

After the above message is displayed, the 68ØØØ is reset and then started. If no other messages appear, then the 68ØØØ has not started or is not communicating with the Z8Ø -- a useful thing to know when you are trying to troubleshoot a problem system.

The error messages presented by **z8Øctl** can basically be divided into 3 groups. Group I consists of messages which are directly related to problems with the Z8Ø or I/O devices connected to the Z8Ø. Usually the 68ØØØ does not cause these errors and can be ignored when trying to diagnose the problem.

Group II errors are caused by some handshaking problem between the Z8Ø and the 68ØØØ. I'll amplify on this when we get to that section.

Boot errors are those which occur during the boot sequence. Again, I'll expound further on these when we get there. The boot track errors are pretty much constant between Xenix 3.1 and 3.2, and so they appear in a separate section at the end of this listing.

Group I Errors

These messages are directly related to problems with the Z8Ø or I/O devices connected to the Z8Ø. The 68ØØØ usually does not cause these errors and can usually be ignored when isolating the problem.

```
                          A F  B C  D E  H L  IX   IY   SP
Bughlt: Rst7 Additional:  xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                          xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                          PC   (SP) (Previous Stack contents)
```

or

Bugchk: Rst7 Fetch xxxx

The Z8Ø fetched a ØxFF in an M1 cycle from memory and executed an RST 38H, also known as RST 7. There are no RST 7 instructions in **z8Øctl**, so memory has been corrupted or the program counter has been set to a memory location beyond the bounds of the **z8Øctl** program area. (**z8Øctl** sets all unused memory to ØxFF). It has been noted that sometimes a machine that is having power or other problems would cause an M1 cycle to be botched and the Z8Ø would read a floating buss instead of the intended memory location. On our systems, the Z8Ø buss floats high, producing a ØxFF, which is an RST 38H instruction. When an RST 38H is detected, **z8Øctl** reads the memory location that caused the RST 38H to make sure that the location really contains a ØxFF. If it doesn't, a "Bugchk: Rst7" is generated, and **z8Øctl** resumes execution at the failed location. The memory location that had the bad M1 cycle is displayed. If the memory location contains a ØxFF on the second read, then a "**Bughlt: Rst7**" message is generated. When the Bughlt occurs, all non-alternate registers on the Z8Ø are dumped as well as the last several stack locations. **The first number on the second line is the memory location that caused the error.** This last is a useful piece of information; with a little effort you may spot a bad RAM chip this way.

Action:
- Check all cards for proper seating, particularly the FDC, CPU, disk cartridge and memory cards.
- Make sure all socketed chips are firmly seated.
- Check power supply for correct voltages at the cards.

```
                          A F  B C  D E  H L  IX   IY   SP
Bughlt: Wnd7 Additional: xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                         xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                         PC   (SP) (Previous Stack contents)
```

This message was added starting with Xenix 3.1 to discern between RST 38H instructions being fetched and other Z8Ø problems. At memory location Øx35 is a jump instruction to the Wnd7 halt code. No location below Øx35 is used by **z8Øctl**, and if the program counter ever gets here, it is due to an error. This is routinely seen when an external primary hard disk or cartridge system loses power. Both of these devices tend to garbage the Z8Ø data buss for a moment on power transition, and the Z8Ø can accidentally call, jump, or RST into one of the lower memory locations.

Action:
- o Check all power connections to all devices connected to the system.
- o Check for reversed ground and neutral on any device connected to the system, including terminals. (This usually also causes a lot of physical damage).
- o Make sure all cards are firmly seated.
- o On the Model 12/6ØØØ, make sure the card cage is bolted down and that its card edge has a good connection with the motherboard.
- o Check power levels at the extreme end of the card cage (i.e., on the 68ØØØ boards).

Bugchk: UNKINT - Code: ....

An interrupt was received from one of the CTC's but nothing is connected to that channel and no interrupt should occur.

Action:
- o If the unit has a Multi-terminal interface, make sure all required Technical Bulletins have been installed, then place it in the LOWEST possible card slot in a Model 12/6ØØØ, (right above the hard disk interface), and if the unit is a Model II/16A, put it in front of the video and Z8Ø memory cards.
- o Run all CTC related diagnostics.

Bughlt: Unexpected Floppy Interrupt

The Z8Ø received an interrupt from the FDC at a time when no interrupt should be generated and could not safely recover from it.

Action:
- o Check the Z8Ø interrupt chain, and on ALL units, make sure that there is no empty card slot between any of the Z8Ø cards. (There may be a gap between the Z8Ø and the 68ØØØ cards).
- o Run all CTC diagnostics.
- o Swap disk controller and CTC chips.
- o See "**Bugchk: UNKINT**" too.

Bugchk: SckMud

In **z8Øctl**, there is a loop which calls all of the drivers, including a driver that picks up requests from the 68ØØØ and informs the various I/O drivers that there is work to do. When a driver is finished with what it was doing, it returns to the "idle loop" and gives time to the next driver.

At the bottom of the "idle loop", the system stack pointer is compared against the value it had when this code ran on the previous cycle. If the value is not the same, then a serious Z8Ø problem has occured and the stack pointer has been altered improperly. This condition is also called a stack imbalance. The system should be shut down and rebooted as soon as possible.

An additional message may appear beneath the "Bugchk: SckMud". The message will read something like, "Shut 'er down, Scotty, she's sucking mud again". Fortunately, the conditions which provoke this error are vanishingly rare, so the vast majority of you should not have to explain THAT one to your customers!

Action:
- o Check all Z8Ø buss connections and card edges.
- o Check seating of all socketed parts on the CPU board, particularly the Z8Ø and the DMA.
- o Run Z8Ø memory tests.
- o Check power supply for out-of-tolerance voltages at the Z8Ø and its memory.
- o Check for correct wait state jumpering.
- o Swap Z8Ø CPU and/or support logic if repeatable.

Bugchk: xHDNRDY
The hard disk controller reports that the drive to be used in this command is not ready, and remained not ready after several reset and retry operations were performed. Software cannot tell the difference between a non-existant bubble and one that is not ready. The "x" indicates the drive number.

Action:
- o Make sure the user is referencing a drive they actually have connected to the system.
- o Check hard drive controller cables and interface card seating.
- o Check hard disk drives' fan filters for excessive dirt (overheating).
- o Check controller alignment; run all diagnostics.
- o Check for correct applications of Technical Bulletins.

Bugchk: ØHDCTC
When initializing the drive, a test value was written to the Sector Count register on the hard drive controller board and the value read back does not match.

Action:
- o Check cables and interface card seating.
- o Check alignments; run hard disk diagnostics.
- o Check filters and Technical Bulletins as applicable.

Bughlt: Z8ØHDW - Z8Ø Hardware Fault
If the hard disk drive was ready and mounted and suddenly went non-ready, the **HDETIME** error usually occurs and the drive is reset and re-initialized. If after this fresh start the drive is still acting up, the above message is displayed. This was a common error for early internal hard drive controllers on 16B+/6ØØØHD systems.

Actions: See those listed above.

Bugchk: HDETIME
A command was issued to the hard drive controller and no completion interrupt was ever received. After 1Ø seconds, the command is aborted and the above message is displayed. The 68ØØØ is informed of the failure and usually will retry the operation. In the meantime, the hard disk controller is reset and reinitialized by the Z8Ø.

Actions: See those listed above.

Bughlt: HDSTAT
In versions of **z8Øctl** up to Xenix 3.1 the hard disk controller uses a state machine driven by a table to handle mounting of hard disks. If the state machine reaches an "impossible" state, this error occurs.

Note: A "state machine" is an algorithm which is driven by a set of conditions. In the above case, the illegal state is one which does not fit the set of conditions the state machine was designed to keep track of.

Action:
- o Run all Z8Ø diagnostics, including memory tests.
- o Check hard drive cables.

Bugchk: IOFAKE
The Z8Ø received an interrupt that appeared to come from the cartridge device at a time when no interrupt should be generated.

Action: See "UNKINT".

Bugchk: SCSIFI
The Z8Ø received an interrupt that appears to come from the Iomega device at a time when an interrupt was expected, but a check of the Iomega status port indicates that it did not generate the interrupt.

Action: See "UNKINT".

Bugchk: USCSIE Sense xx xx
The SCSI Buss reported an error, but the error code was not known to the driver. The sense key, class, and code are displayed. The error is treated as a hard error and the command is aborted.

Other System Messages

WARNING: bad configuration, resetting to defaults.
The parameters specified with the cfg command require an unreasonable amount of memory be used for Xenix, and would leave the available user memory area too small to be usuable. The system ignores the table size parameters and uses the defaults. This message can only occur on system boot. This message may be caused by running a system with less memory than it ordinarily has, or with an abnormally small (read: unsupported!) amount of memory (e.g. 256k or 384k). Abnormal.

**** ABNORMAL System Shutdown ****
This message appears when errors occur during 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 this 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.

error on dev name(nn/mm)
This is a way that device driver diagnostic messages can start. In reality, none of Tandy's drivers use this format. This style of message is almost never seen. The message will indicate the specific driver and complaint. The name is a word indentifying the device. (See above section on device drivers).

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 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. Abnormal.

no file

There are too many open files, and 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 user 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 ****
[Z8Ø Control System Halted] ++

This message appears when the system has been shut down properly. It indicates that the machine may now be rebooted or powered down.

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 the file system is created (using mkfs). 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 file system to a backup device, run mkfs with more inodes specified, and restore the files from backup. Abnormal.

++ See also the portion of this document addressing /z8Øctl error messages.

out of text
When shared text programs 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 (i.e., those programs which have the sticky bit set -- for more information, read the help page on chmod) requires a permanent entry in this table; nonsticky pure text programs require an entry only when there is at least one copy being executed. Abnormal.

panic: blkdev
An internal disk I/O request, already verified as valid, is discovered to be referring to a nonexistent disk. System Inconsistency, Fatal.

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. Hardware, Fatal.

panic: IO err in swap
A hard++ I/O error occurred while reading or writing the swap area. Hardware, Fatal.

panic: no fs
A file system descriptor has disappeared from its table. System Inconsistency, Fatal.

panic: no imt
A mounted file system has disappeared from the mount table. 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.

++ See the above section on device driver messages.

panic: Out of swap
There is insufficient space on the swap disk to hold a task that is scheduled to be swapped. The system refuses to create tasks when it feels there is insufficient disk space, but it is possible to create situations to fool this mechanism. Abnormal, 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. This is usually caused by a bug in a device driver. Abnormal, Fatal.

proc on q
The system attempts to queue a process already on the process ready-to-run queue. System Inconsistency, Fatal.

Xenix 3.2 /z8Øctl Error Messages

This information contains a list of the error messages for the **/z8Øctl** which is used with Xenix 3.2 (**/z8Øctl** version 3.2(12Ø) and later). These messages have changed a good bit more than the 68ØØØ error messages, although you Star Trek fans will probably be pleased to know that there is still a "suck mud" error (see Xenix 3.1 **/z8Øctl** messages if you don't understand that one!)

In this version of Xenix, there are only two **/z8Øctl** error prefixes:

Bugchk:	-- Bug Check
Bughlt:	-- Bug Halt

Bugchk messages indicate that either some synchronization problem arose that should not happen or an unusual hardware error occured. In the case of a synchronization problem, the request made by the 68ØØØ is usually ignored. This may cause any future reference to the same driver to hang, but this is usually preferable to what the system refused to do. For hardware operations, the request is aborted and an error is returned to the 68ØØØ, who usually retries the operation. In some cases, the hardware is reset in an attempt to resolve the problem.

Bughlt messages indicate that an error was detected that was so serious that a recovery could not or should not be attempted. For example, if the Z8Ø detected that the program counter had managed to set to an invalid address, an error message is displayed and the system is halted. In this case, there is no way to tell how this happened, and it may indicate a hardware or power problem. When a **Bughlt** occurs, the Z8Ø halts the 68ØØØ and places itself in a semi-halted state. The hard disk drives may not be parked or restored in this case.

If an error message appears on the console which does not start with **Bugchk** or **Bughlt**, then it either came from Xenix or some application program. (Messages displayed at boot and shutdown which are within brackets are generated by /z8Øctl but are not error messages). Z8Ø messages are recorded in **/usr/adm/messages** when possible in Xenix 3.2.

Group I Errors

These messages are directly related to problems with the Z8Ø or I/O devices connected to the Z8Ø. The 68ØØØ usually does not cause these errors and can usually be ignored when isolating the problem.

```
                          A F  B C  D E  H L  IX   IY   SP
Bughlt: Rst7 Additional:  xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                          xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                          PC   (SP) (Previous Stack contents)
```

or

Bugchk: Rst7 Fetch xxxx

The Z8Ø fetched a ØxFF in an M1 cycle from memory and executed an RST 38H, also known as RST 7. There are no RST 7 instructions in **z8Øctl**, so memory has been corrupted or the program counter has been set to a memory location beyond the bounds of the **z8Øctl** program area. (**z8Øctl** sets all unused memory to ØxFF). It has been noted that sometimes a machine that is having power or other problems would cause an M1 cycle to be botched and the Z8Ø would read a floating buss instead of the intended memory location. On our systems, the Z8Ø buss floats high, producing a ØxFF, which is an RST 38H instruction. When an RST 38H is detected, **z8Øctl** reads the memory location that caused the RST 38H to make sure that the location really contains a ØxFF. If it doesn't, a "Bugchk: Rst7" is generated, and **z8Øctl** resumes execution at the failed location. The memory location that had the bad M1 cycle is displayed. If the memory location contains a ØxFF on the second read, then a "**Bughlt: Rst7**" message is generated. When the Bughlt occurs, all non-alternate registers on the Z8Ø are dumped as well as the last several stack locations. **The first number on the second line is the memory location that caused the error.** This last is a useful piece of information; with a little effort you may spot a bad RAM chip this way. **/z8Øctl** attempts to record the crash so that it will be added to **/usr/adm/messages** the next time the system boots, but there is no guarantee that this will be successful.

Action:
- o Check all cards for proper seating, particularly the FDC, CPU, disk cartridge and memory cards.
- o Make sure all socketed chips are firmly seated.
- o Check power supply for correct voltages at the cards.

```
                          A F  B C  D E  H L  IX   IY   SP
Bughlt: Wnd7 Additional: xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                         xxxx xxxx xxxx xxxx xxxx xxxx xxxx
                         PC   (SP) (Previous Stack contents)
```

This message was added starting with Xenix 3.1 to discern between RST 38H instructions being fetched and other Z8Ø problems. At memory location Øx35 is a jump instruction to the Wnd7 halt code. No location below Øx35 is used by **z8Øctl**, and if the program counter ever gets here, it is due to an error. This is routinely seen when an external primary hard disk or cartridge system loses power. Both of these devices tend to garbage the Z8Ø data buss for a moment on power transition, and the Z8Ø can accidentally call, jump, or RST into one of the lower memory locations.

Customers who may experience this message after Xenix has been halted during a normal shutdown can ignore the message.

Action:
- o Check all power connections to all devices connected to the system.
- o Check for reversed ground and neutral on any device connected to the system, including terminals. (This usually also causes a lot of physical damage).
- o Make sure all cards are firmly seated.
- o On the Model 12/6ØØØ, make sure the card cage is bolted down and that its card edge has a good connection with the motherboard.
- o Check power levels at the extreme end of the card cage (i.e., on the 68ØØØ boards).

Bugchk: UNKINT - Code: ....

An interrupt was received from one of the CTC's but nothing is connected to that channel and no interrupt should occur.

Action:
- o If the unit has a Multi-terminal interface, make sure all required Technical Bulletins have been installed, then place it in the LOWEST possible card slot in a Model 12/6ØØØ, (right above the hard disk interface), and if the unit is a Model II/16A, put it in front of the video and Z8Ø memory cards.
- o Run all CTC related diagnostics.

Bughlt: UNEXFIRQ

The Z8Ø received an interrupt from the FDC at a time when no interrupt should be generated and **/z8Øctl** could not safely recover from it.

Action:
- o Check the interrupt chain, and on all units, make certain that there are no gaps between any of the cards on the Z8Ø interrupt chain (such as hard drive interface, multiterminal boards, and disk cartridge interface)
- o Run all CTC related diagnostics
- o Swap the FDC and/or CTC chips
- o Refer to the **UnkInt** message.

Bughlt: SckMud

In **z8Øctl**, there is a loop which calls all of the drivers, including a driver that picks up requests from the 68ØØØ and informs the various I/O drivers that there is work to do. When a driver is finished with what it was doing, it returns to the "idle loop" and gives time to the next driver.

At the bottom of the "idle loop", the system stack pointer is compared against the value it had when this code ran on the previous cycle. If the value is not the same, then a serious Z8Ø problem has occured and the stack pointer has been altered improperly. This condition is also called a stack imbalance. The Z8Ø is halted. (Previous versions attempted to continue running when this occurred).

Action:
- o Check all Z8Ø buss connections and card edges.
- o Check seating of all socketed parts on the CPU board, particularly the Z8Ø and the DMA.
- o Run Z8Ø memory tests.
- o Check power supply for out-of-tolerance voltages at the Z8Ø and its memory.
- o Check for correct wait state jumpering.
- o Swap Z8Ø CPU and/or support logic if repeatable.

Bugchk: HdTimeOut
A command was issued to the hard drive controller and no completion interrupt was ever received. After 1Ø seconds, the command is aborted and the above message is displayed. The 68ØØØ is informed of the failure and usually will retry the operation. Depending on the state of the hardware, a reset may be attempted, which may display additional messages.

Action:
- o Check hard drive controller and interface cabling and interface card seating.
- o Run all hard disk diagnostics.
- o Check cooling fan filters for excessive dirt and clean as necessary.
- o Verify that all modifications have been installed.

Bugchk: HdFail -- Additional aa bb cc dd ee
An attempt to restart the disk controller or a drive after it went not ready has occurred. This message may be proceeded by an **HdTimeOut** error. The additional information is as follows:

aa	Status of disk controller before reset attempt
bb	Status of disk controller after reset attempt
cc	Timeout Control Status
dd	68ØØØ Command Status
ee	Status of disk controller after recalibrate and seek

Action: see **HdTimeOut**.

Bugchk: IoFake
The Z8Ø received an interrupt that appeared to come from the Iomega sub-system at a time when no interrupt should have been generated.

Action: see **UnkInt**.

Bugchk: IoFakeI
The Z8Ø received an interrupt that appeared to come from the Iomega sub-system at a time when an interrupt was expected, but a check of the Iomega status port indicated that it did not generate the interrupt.

Action: see **UnkInt**.

Bugchk: IoUnkEr -- Sense: ss cc
The Iomega/SCSI bus reported an error, but the error code was not known to the driver. The sense key, class, and code are displayed. The error is treated as a hard error and the command is aborted.

Bugchk: DbLock

The Z8Ø has several devices that use the DMA. These include the floppy disk driver, hard disk driver, SCSI driver and some memory-to-memory operations. Because of the fact that only one DMA is present in the machine, a software routine allocates the DMA for drivers that request it and the code forces others to wait until the DMA is available. This error occurs if a driver requests the DMA and already has it. The command is completed successfully, but it indicates that memory is failing or a hardware problem, since a driver managed to exit without releasing the DMA.

Bugchk: IlUlRq

A driver attempted to "give back" the DMA so that other drivers could use it and it was determined that this driver did not currently own the DMA. This internal request is ignored and the system may continue to run normally, but this is an indication of failing Z8Ø memory or hardware since a driver became convinced that it owned the DMA when it had never acquired it.

Group II Errors

These errors are caused by some handshaking problem between the 68ØØØ and the Z8Ø. All requests are validated by the 68ØØØ before the Z8Ø is interrupted. Then the Z8Ø makes sure the request is valid as well. When the Z8Ø reports a problem, it means that:

(1) The data (stored in 68ØØØ RAM) has been damaged since the 68ØØØ checked it or was misread by the Z8Ø.
(2) The validation code on the 68ØØØ has become corrupted.
(3) The validation code on the Z8Ø has become corrupted.

If after rebooting the machine the problem persists, the **/xenix** and **/z8Øctl** files may be suspect. In order to rule out case (2) and (3), reinstall them. If the problem still persists, Case (1) is the probable difficulty, which is a shame -- it's the toughie. It could be one (or more) of the following:

(a) Boards not up to date on modifications, particularly the bad "AS" type parts mods.
(b) Problem in arbitration.
(c) Problem with memory on Z8Ø and/or 68ØØØ.
(d) Problem with Z8Ø DMA.
(e) Excessive noise on 68ØØØ ribbon cables, particularly RAS and CAS.
(f) Incorrect bank selection in Z8Ø memory.
(g) Overheating and/or power problems.
(h) Card cage modifications not correct.

... ad nauseam, ad infinitum.

Since so many things can cause these errors, the above items and any others which come to mind should be used to start any attack on these errors.

Bughlt: BadMaj
The 68ØØØ creates and maintains a queue of requests for the Z8Ø to perform. This queue is kept in 68ØØØ memory and has a pointer indicating where the next item should be added (by the 68ØØØ) and another pointer indicating which item should be handled by the Z8Ø next. This error indicates that a request in this table had an invalid major number. This number can only be in the range Ø to 5; any higher number is considered invalid.

Bugchk: Bd68Rq -- Code xx
This message occurs when a request is fetched from the 68ØØØ queue, and is requesting a legal driver, but the action requested cannot be performed by that driver. For example, setting status on the floppy drive (such as setting a baud rate) is not legal and will generate this message. The request is ignored. This means the 68ØØØ may block all future I/O to that device and the system may have to be rebooted.

Bugchk: FdNoGo
Bugchk: HdNoGo
Bugchk: IoNoGo
Bugchk: SrNoGo
The Z8Ø received a request from the 68ØØØ via the request queue,and when validating the command "mailbox" it found that the mailbox was still marked "unavailable". The command is ignored. If this was a real disk request, further I/O to this device will block and may eventually require the system to be rebooted.

Bughlt: ØFDCMD
A disk request was received and when checked against the inverted check-copy, the two commands did not match. This will also occur if the two copies do match but the request code is not known. To avoid destroying existing data, the system is halted.

Bughlt: SRBADDR
The screen device was passed an address that does not exist in the Z8Ø video RAM. The system is halted.

Bughlt: SRXCSR
The check-copy of the screen device request did not match the master copy. The system is halted.

Bugchk: IoBozo
An I/O request was made to the cartridge driver, but the command bits were invalid. The command is aborted.

Bughlt: 68k crashed
The Z8Ø interrupts the 68ØØØ once every 16.67 msec (2Ø msec in 5Øhz mode). If the 68ØØØ fails to acknowledge any of these interrupts over a 6 second period, the Z8Ø assumes the 68ØØØ has halted or gone into a loop with interrupts disabled. The 68ØØØ must miss at least 36Ø clock interrupts in a row (3ØØ in 5Øhz mode) to generate this error. Once an interrupt is serviced, another 6 seconds of missed interrupts is required for a halt of this type.

[Z8Ø Control System Halted]

The Z8Ø has received a HALT signal from the 68ØØØ. If the Z8Ø has been directed to halt and stay halted, the hard disk drive heads are parked (a message is displayed to confirm this) and the 68ØØØ processor is halted. Then **z8Øctl** recovers any kernel messages that have not been added to **/usr/adm/messages**. Once all this is done, the above message is displayed and the Z8Ø is put into an idle state. It continues to monitor the clock interrupt, so that it can deselect floppy drives and spin-down Iomega drives if they were active when the system was halted.

One note about head parking: Xenix only parks the heads on drives that have been accessed by Xenix since the last boot of Xenix. If a drive has not been accessed since the last boot, it will not be parked. In addition, if a reboot is to be attempted, the drives will not be parked.

If the Z8Ø was directed to attempt a reboot, the 68ØØØ is halted and the hard disk drives are restored to track Ø (to help insure the reboot will work with some older ROMs that do not wait long enough for track Ø indication). Then **z8Øctl** recovers any kernel messages that have not been added to **/usr/adm/messages**. Then the Z8Ø displays the "Halted" message and delays for a few seconds. Then it disables all interrupts, enables the shadow ROM and jumps to address ØxØØØØ. If the auto-boot boot track is installed, the system will be able to restart unattended.

A Quick Look at The Xenix Boot Track
or
What Happens When You Hit "RESET"

The Xenix boot track allows the user to start either a Z8Ø-based utility like **diskutil** or **z8Ødiags**, or boot the Xenix operating system. The process of booting Xenix involves locating the files **/xenix** and **/z8Øctl** on the booting device, loading the 68ØØØ code into 68ØØØ memory from **/xenix**, and loading the Z8Ø code into Z8Ø memory from **/z8Øctl**. The 68ØØØ memory is loaded first. Once the Z8Ø memory is loaded, control is transferred to **/z8Øctl**. When the screen clears, **z8Øctl** is now running.

The boot ROM in the Model II/12/16A/16B/6ØØØ reads the boot track in from floppy drive Ø or hard drive Ø. Once loaded, the Xenix boot track displays something similar to this:

Tandy 68ØØØ/Xenix Boot Version is 3(24) 29-Oct-85 [xØhz]
Copyright 1985 Tandy Corporation. All Rights Reserved.

Xenix Boot>

will be displayed. Any messages displayed before this point have been generated by the boot ROM.

At the prompt you may press:

"?"	-- for brief help
<ENTER>	-- to boot **/xenix** if it exists on the boot device
"diskutil <ENTER>"	-- to run **diskutil**
"z8Ødiags <ENTER>"	-- to run the old Z8Ø diagnostics
"<F2>"	-- to specify an alternate Z8Ø driver
"name <ENTER>"	-- to run a kernal (ie, like /xenix) with an alternate name. This is useful for system protection; if you want to boot off a file named "bozo", it is possible to do so.

The "[xØhz]" indicates the system clock speed that is indicated by the boot ROM. Since the NMI clock is generated by the CRTC, this will indicate the correct system clock interval. In the USA, you should see [6Øhz] at all times. If the 5Øhz ROM is installed, [5Øhz] is displayed. If a reverse video '6' is displayed in the [6Øhz] message, it means that the boot code was unable to determine what speed was being programmed by the ROM and has assumed 6Øhz. The ROM should be replaced if that occurs. The speed information is passed to System 3 Xenix via a fixed memory location so that it will know how many interrupts represent a second.

Boot Track Error Messages

Boot track messages tend to be a bit terse due to the limited amount of code space allowed by the boot ROM. The descriptions below should help to fill out the information given by the error alone.

Program not found

The program name specified at the "Xenix Boot>", "Kernel>", or "Z8Ø Driver>" prompts was not found in the root directory "/" on the boot disk. The files are either missing or the name(s) you typed are not spelled correctly.

Actions:

- If you just pressed <ENTER>, then either **/xenix** or **/z8Øctl** or both have been deleted from the disk. This could have happened during an **fsck** of a dirty system or via a disk problem. Try booting on floppy or another floppy. Then mount the failing disk with the **/etc/mount** command and make sure that the **/xenix** and **/z8Øctl** files are in the root "/" directory. If they are missing, copy the CORRECT versions of these files to the failing disk. Then **umount** and bring the system down and try to boot on the problem disk.
- If you typed one or more characters at the boot prompt, then check the spelling of the file you entered. (The boot ignores all non-printable characters including blank). If the program you specify (like **diskutil**) still cannot be found, try booting Xenix and make sure the file hasn't been deleted.

Badctl

The Z8Ø Control System file **/z8Øctl** was located, but it did not contain the correct code. This may indicate a corrupted system or the customer has copied something over this file.

Action:

- Boot up the system using a floppy (if trying to boot the hard disk) or another floppy (if trying to boot from floppy) and mount the failing disk. Copy the correct version of the file off the installation floppy. Umount the system and halt the system. Then try to reboot.

NoPVH

While booting the system, it was noted that an area that **diskutil** writes to the drive was missing. This is known as the PVH. It contains information about the hard drive. If this is not found, the system no longer knows what size the hard disk is. Older versions of Xenix assumed an 8 meg hard drive in this case. Newer versions don't make this assumption as it only makes things worse if it is not an 8 meg drive.

NoBadT

While booting the system, it was noted that the bad track table is not present on the disk. The boot code will try to load Xenix anyway, and may or may not be successful. If Xenix starts running, it may or may not stay running since tracks that used to contain data are now empty. If an important track, part of the root directory for instance, was missing, then the system would likely crash right away.

In most cases, the best action to take is to boot off floppy and save the data files off the system. You should then reformat, reinstall the system, and reinstall your files. There is no real guarantee that the data will be absolutely sound, so you may want to compare it to an older backup and watch for erratic behavior from the restored system.

NewPal - Hardware Change Required
NewPal

The new PAL that is required for burst mode DMA operations on the 68ØØØ has not been installed on this system.

Iocyc7

The unit is a Model II/16A and has an old video/keyboard card. Rev. A or B boards will answer at ports in the ØxFn and Øx7n range, which conflicts with the multi-terminal boards. **/z8Øctl** is unable to tell the difference between bad mapping and an unprogrammed multi-terminal board. You should perform the relevant Technical Bulletins on the video board to get rid of this error.

BadCyl

In loading **/xenix** or **/diskutil**, a bad directory record was found or calculated.

Reset the machine and try again. If the problem persists, try booting on floppy and running **fsck** on the problem device. If this still does not help, try swapping the Z8Ø memory boards. The memory diagnostics do not check low memory (ie, below Øx4ØØØ) and the boot track resides in memory below this address. **/z8Øctl** uses all memory from ØxØØØØ to Øx7FFF.

As of Xenix 3.1, **BadCyl** can also indicate that the hard disk is formatted for 3.Ø but has the 3.1 boot track present, or is formatted for 3.1 and has the 3.Ø boot track. Boot up on floppy and place the correct boot track on the system.

HdIoEr Info: Cyl=xx HDS=xx Status=xx Error=xx

In loading **/xenix** or **/diskutil**, a hard disk error occurred.

Reset and try again. If you are using a hard disk, power down and check all cable connections. Check for correct terminator placement and correct connections to all secondary drives. If you still can't get the system started, boot off of floppy and try to mount the hard drive. If you can, the problem is in all likelihood located in **/xenix, /z8Øctl** or the boot track. You can try the kernal refresh option on the Tools Disk, or you can just save the data files off the hard disk and reinstall. In either case, you should save the data off and reinstall as soon as possible, since the integrity of the system is somewhat questionable at best even if you can get it running again.

HRDIOE - Can't load Xenix

This message indicates that the boot was unable to load and start Xenix. Other messages may have preceeded this one. This message is a general "catch-all" message which usually indicates some hardware problem.

68kMem - Memory Error

This message will appear if any part of Xenix written to the 68ØØØ memory could not be read back correctly. This can also appear if there is insufficient memory to run Xenix.

Reset and try again. Also check for the following: clean filters, good ventilation, card cage modifications, bad RAM, refresh or decoding circuitry on the 68ØØØ, bad seating on U36 burst mode PAL on 68ØØØ CPU or bad jumper connection to that PAL.

No68k

This error will appear if the Z8Ø is unable to write a test value to 68ØØØ memory and read it back. You will have to RESET to try again. This is an indication of:

- o No 68ØØØ set.
- o Bad 68ØØØ boards or cables.
- o Z8Ø bus decoding problems.

UnkInt

Indicates that some device generated an interrupt when it was not supposed to. Check all interrupt-driven devices and replace the offending part.

****

File Listings

(Taken from Xenix 3.2 System)

****

This is a listing of files from a stock Xenix 3.2 system. I can't guarantee that your system will look exactly like this, nor can I guarantee that this is a 100% complete list, but it should help you decide if your system is relatively close to normal in case of question.

Note well: **this is a listing from a Xenix 3.2 system!** If you are running anything else, take what you see here with a large chunk of salt; it may not be in the least applicable to you.

Files in the /bin directory:

```
-rwx--x--x   4 bin      bin       142234 Apr  8  1987 /bin/view
-rwx--x--x   4 bin      bin       142234 Apr  8  1987 /bin/ex
-rwx--x--x   4 bin      bin       142234 Apr  8  1987 /bin/edit
-rwx--x--x   4 bin      bin       142234 Apr  8  1987 /bin/vi
-rwx--x--x   1 bin      bin        77106 Apr  7  1987 /bin/csh
-rwx--x--x   1 bin      bin        14598 Apr  7  1987 /bin/.lpr
-rws--x--x   1 root     bin        18940 Jan  8  1985 /bin/.mail
-rwx--x--x   1 bin      bin        80160 Apr  7  1987 /bin/awk
-rwx--x--x   2 bin      bin        20594 Apr  8  1987 /bin/backup
-rwx--x--x   1 bin      bin         3432 Apr 30  1985 /bin/basename
-rwx--x--x   1 bin      bin        78186 Apr  8  1987 /bin/csh10
-rwx--x--x   1 bin      bin        11946 Apr  7  1987 /bin/cu
-rwx--x--x   1 bin      bin        19000 Apr  7  1987 /bin/cu.s3
-rwx--x--x   1 bin      bin        10922 Apr  7  1987 /bin/date
-rws--x--x   1 root     bin         7664 Apr  7  1987 /bin/df
-rwx--x--x   1 bin      bin        14288 Apr  7  1987 /bin/diff
-rwx------   2 root     bin         9976 Apr  7  1987 /bin/disable
-rwx--x--x   1 bin      bin         7042 Apr  7  1987 /bin/du
-rwx--x--x   2 bin      bin        20594 Apr  8  1987 /bin/dump
-rwx--x--x   2 bin      bin        20594 Apr  8  1987 /bin/backup
-rwx--x--x   2 bin      bin        32220 Apr  8  1987 /bin/dumpdir
-rwx--x--x   2 bin      bin        31334 Apr  7  1987 /bin/ed
-rwx------   2 root     bin         9976 Apr  7  1987 /bin/enable
-rwx------   2 root     bin         9976 Apr  7  1987 /bin/disable
-rwx--x--x   1 bin      bin        10610 Apr  7  1987 /bin/fgrep
-rwxr-xr-x   1 bin      bin        14416 Apr  8  1987 /bin/format
-rwx--x--x   1 bin      bin        34778 Apr  7  1987 /bin/fsck
-rwx--x--x   1 bin      bin        12708 Apr  7  1987 /bin/grep
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/lc
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/lf
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/lc
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/ll
-rwxr-xr-x   1 bin      bin          134 Apr  1  1987 /bin/login
-rwx--x--x   1 bin      bin        11906 Apr  8  1987 /bin/lpr
-rwx--x--x   1 bin      bin        12640 Apr  8  1987 /bin/lprint
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/lr
-rwx--x--x   5 bin      bin        20542 Apr  7  1987 /bin/lx
```

```
-rws--x--x   1 root     bin        15962 Apr  7  1987 /bin/newgrp
-rwx--x--x   1 bin      bin         7922 Apr  7  1987 /bin/nohup
-rws--x--x   1 root     bin        20120 Apr  7  1987 /bin/passwd
-rwx--x--x   1 bin      bin        17420 Apr  7  1987 /bin/pr
-rws--x--x   1 root     bin        25564 Apr  8  1987 /bin/ps
-rwx--x--x   1 bin      bin         7184 Apr  8  1987 /bin/qps
-rws--x--x   1 root     bin        14308 Apr  7  1987 /bin/quot
-rwx--x--x   2 bin      bin        31334 Apr  7  1987 /bin/red
-rwx--x--x   2 bin      bin        32220 Apr  8  1987 /bin/restore
-rws--x--x   1 root     bin         6402 Apr  7  1987 /bin/rmdir
-rwx--x--x   1 bin      bin        61992 Jan 15  1985 /bin/runcobol
-rwxr-xr-x   1 bin      bin        56235 Jan 15  1985 /bin/runcobol.o
-rwx--x--x   1 bin      bin        22162 Apr  7  1987 /bin/sed
-rwx--x--x   1 bin      bin        14208 Apr  7  1987 /bin/sort
-rws--x--x   1 root     bin        22582 Apr  7  1987 /bin/su
-rw-r--r--   1 bin      bin         3869 Apr 30  1985 /bin/sub.c
-rwx--x--x   1 bin      bin         7094 Apr  7  1987 /bin/tail
-rwx--x--x   1 bin      bin         3640 Apr  7  1987 /bin/tee
-rwx--x--x   1 bin      bin        14952 Apr  7  1987 /bin/tset
-rwx--x--x   1 bin      bin        34278 Apr  8  1987 /bin/tsh
-rwx--x--x   1 bin      bin         5622 Apr  8  1987 /bin/tty
-rwx--x--x   1 bin      bin        28880 Apr 30  1985 /bin/tx
-rwx--x--x   1 bin      bin        12774 Apr  8  1987 /bin/who
-rwx--x--x   1 bin      bin         4912 Apr  8  1987 /bin/yes
-rwx--x--x   2 bin      bin         5556 Apr  7  1987 /bin/[
-rwx--x--x   1 bin      bin         6942 Apr  7  1987 /bin/cat
-rwx--x--x   1 bin      bin         9464 Apr  7  1987 /bin/chgrp
-rwx--x--x   1 bin      bin         6050 Apr  7  1987 /bin/chmod
-rwx--x--x   1 bin      bin         9156 Apr  7  1987 /bin/chown
-rwx--x--x   1 bin      bin        11590 Apr  7  1987 /bin/copy
-rwx--x--x   2 bin      bin        12024 Apr  7  1987 /bin/cp
-rwx--x--x   1 bin      bin        11738 Apr  7  1987 /bin/dd
-rwsr-xr-x   1 root     root        6780 Apr  8  1987 /bin/diskstat
-rwx--x--x   1 bin      bin         3438 Apr  7  1987 /bin/echo
-rwx--x--x   1 bin      bin         9948 Apr  7  1987 /bin/expr
-rwxr-xr-x   1 bin      bin            9 Apr  7  1987 /bin/false
-rwx--x--x   1 bin      bin         5604 Apr  7  1987 /bin/kill
-rwx--x--t   2 bin      bin        16412 Apr  7  1987 /bin/l
-rwx--x--x   2 bin      bin        12024 Apr  7  1987 /bin/ln
-rwx--x--t   2 bin      bin        16412 Apr  7  1987 /bin/ls
-rws--x--x   1 root     bin         6206 Apr  7  1987 /bin/mkdir
-rws--x--x   1 root     bin        12024 Apr  7  1987 /bin/mv
-rwx--x--x   1 bin      bin         6410 Apr  7  1987 /bin/pwd
-rwx--x--x   1 bin      bin         7974 Apr  7  1987 /bin/rm
-rwx--x--t   2 bin      bin        37762 Apr  7  1987 /bin/rsh
-rwx--x--t   2 bin      bin        37762 Apr  7  1987 /bin/sh
-rwx--x--x   1 bin      bin         5664 Apr  7  1987 /bin/sleep
-rwx--x--x   1 bin      bin        20868 Apr  7  1987 /bin/stty
-rwx--x--x   1 bin      bin         2510 Apr  7  1987 /bin/sync
```

```
-rwx--x--x   1 bin      bin         36724 Apr  7  1987 /bin/tar
-rwx--x--x   2 bin      bin          5556 Apr  7  1987 /bin/test
-rwxr-xr-x   1 bin      bin             Ø Apr  7  1987 /bin/true
```

Files in the /etc directory:

```
-rwx------   1 bin      bin        23314 Apr  8  1987 /etc/adj_clock
-rwxr-x---   1 bin      bin          473 Apr  4  1985 /etc/asktime
-rwx------   2 root     root       13422 Apr  8  1987 /etc/autoboot
-rwx------   2 root     root       13422 Apr  8  1987 /etc/park
-rwxr-xr-x   1 bin      bin         5482 Apr  8  1987 /etc/cartstat
-rwx------   1 bin      bin        1548Ø Apr  8  1987 /etc/cfg
-rw-r--r--   1 bin      bin          1Ø Apr 3Ø  1985 /etc/checklist
-rwx------   1 cron     other      14884 Apr  7  1987 /etc/cron
-rw-r--r--   1 bin      bin            Ø Apr  5  1985 /etc/cshrc
-rw-r--r--   2 bin      bin           31 May  1  1985 /etc/default/backup
-rw-r--r--   1 bin      bin           11 Apr 3Ø  1985 /etc/default/cron
-rw-r--r--   1 bin      bin           36 Oct 17  1986 /etc/default/dst
-rw-r--r--   2 bin      bin           31 May  1  1985 /etc/default/dump
-rw-r--r--   1 bin      bin           15 Apr 3Ø  1985 /etc/default/dumpdir
-rw-------   1 cron     other        413 Jul 24  1986 /etc/default/idle
-rw-r--r--   1 bin      bin          114 Apr 3Ø  1985 /etc/default/login
-rw-r--r--   1 cron     other         2Ø Jan  7  1987 /etc/default/lpd
-rw-r--r--   1 bin      bin          1Ø Apr 3Ø  1985 /etc/default/mkuser
-rw-r--r--   1 bin      bin          37 Apr 3Ø  1985 /etc/default/passwd
-rw-r--r--   1 bin      bin          18 Apr 3Ø  1985 /etc/default/restor
-rw-r--r--   1 root     root        114 Sep 26  1986 /etc/default/su
-rw-r--r--   1 bin      bin         182 Oct 17  1986 /etc/default/tsh
-rws--x--x   1 root     root       126ØØ Apr  7  1987 /etc/dmesg
-rwx------   1 root     root       14348 Apr  8  1987 /etc/dstcomp
-rwx--x--x   1 bin      bin        11726 Apr 3Ø  1985 /etc/fixperm
-rwx------   1 bin      bin         8854 Apr  8  1987 /etc/fsck_recover
-rwx------   1 bin      bin        1Ø41Ø Apr  8  1987 /etc/getty
-rw-r--r--   1 cron     other         84 Jul 24  1986 /etc/idle.pgms
-rwx------   1 bin      bin         4252 Apr  7  1987 /etc/inir
-rw-r--r--   1 root     root         668 Nov 17 11:11 /etc/logbook
-rwx------   1 root     bin        21472 Apr  7  1987 /etc/login
-rwx------   1 bin      bin         9833 Apr  8  1987 /etc/makefs
-rwx------   1 root     root        5486 Apr  8  1987 /etc/makemtab
-rwx------   1 root     bin        27666 Apr  7  1987 /etc/mkuser
-rw-r--r--   1 root     root         275 Sep 26  1986 /etc/motd
-rw-r--r--   1 bin      bin         3615 Mar  4  1987 /etc/rc
-rwx------   1 root     bin        14386 Apr  7  1987 /etc/rmuser
-rwx------   1 root     root        9Ø1Ø Apr  7  1987 /etc/setmnt
-rwx------   1 bin      bin        26Ø2Ø Apr  9  1987 /etc/shutdown
-rwx------   1 root     bin        12922 Apr  7  1987 /etc/sulogin
-rwx------   1 sys      sys         88Ø8 Sep 3Ø  1986 /etc/sysadmin
-rwx------   1 bin      bin        1Ø126 Apr  8  1987 /etc/timezone
-rw-r--r--   1 bin      bin           5Ø Dec  9 14:54 /etc/ttys
-rw-r--r--   1 bin      bin           78 Jan  6  1987 /etc/ttytype
-rwx------   1 bin      bin         3Ø55 Nov  6  1986 /etc/tz
-rwx------   1 cron     other       2962 Apr  8  1987 /etc/update
-rw-r--r--   1 bin      bin         23Ø6 Aug 14  1986 /etc/usa.dst
```

```
-rws--x--x   1 root     bin         8814 Apr  8  1987 /etc/verify
-rw-r--r--   1 bin      bin            9 Nov 17 11:1Ø /etc/version
-rw-r--r--   1 bin      bin           45 Sep 17  1986 /etc/versionlog
-rwx------   1 bin      bin          426 Apr  8  1987 /etc/wall
-rw-r--r--   1 bin      bin          265 Apr 3Ø  1985 /etc/default/tar
-rws--x--x   1 root     bin        1ØØ16 Apr  8  1987 /etc/drive
-rwx------   1 bin      bin         498Ø Apr  7  1987 /etc/haltsys
-rw-r--r--   1 bin      bin         3882 Nov 19  1986 /etc/hdinit
-rwx------   1 bin      bin        1Ø342 Apr  7  1987 /etc/init
-rwx--x--x   1 sys      sys        1352Ø Apr  7  1987 /etc/mkfs
-rwx--x--x   1 sys      sys         5588 Apr  7  1987 /etc/mknod
-rwx--x--x   1 bin      bin        15Ø58 Apr  7  1987 /etc/mount
-rw-r--r--   1 bin      bin         7396 Jul 16  1986 /etc/termcap
-rwx--x--x   1 bin      bin        1Ø318 Apr  7  1987 /etc/umount
```

Files in the /dev directory:

```
brw-rw-rw-   1 sys      sys        1, 64 Jun 24 11:Ø9 /dev/cdØ
brw-rw-rw-   1 sys      sys        1, 72 Sep 26  1986 /dev/cd1
brw-rw-rw-   1 root     root       1, 67 Sep 26  1986 /dev/cdbtØ
brw-rw-rw-   1 root     root       1, 75 Sep 26  1986 /dev/cdbt1
crw-rw-rw-   1 root     root       6,192 Sep 26  1986 /dev/clp
crw-rw-rw-   1 root     root       Ø,  Ø Dec 14 16:53 /dev/console
crw-rw-rw-   1 root     root       7,129 Sep 18 12:26 /dev/cuaØ
crw-rw-rw-   1 root     root       7,193 Sep 26  1986 /dev/cuaØ.12ØØ
crw-rw-rw-   1 root     root       7,161 Sep 26  1986 /dev/cuaØ.3ØØ
crw-rw-rw-   1 root     root       7,13Ø Sep 26  1986 /dev/cua1
crw-rw-rw-   1 root     root       7,194 Sep 26  1986 /dev/cua1.12ØØ
crw-rw-rw-   1 root     root       7,162 Sep 26  1986 /dev/cua1.3ØØ
crw-rw-rw-   1 root     root       7,132 Sep 26  1986 /dev/cua4
crw-rw-rw-   1 root     root       7,196 Sep 26  1986 /dev/cua4.12ØØ
crw-rw-rw-   1 root     root       7,164 Sep 26  1986 /dev/cua4.3ØØ
crw-rw-rw-   1 root     root       7,133 Sep 26  1986 /dev/cua5
crw-rw-rw-   1 root     root       7,197 Sep 26  1986 /dev/cua5.12ØØ
crw-rw-rw-   1 root     root       7,165 Sep 26  1986 /dev/cua5.3ØØ
crw-rw-rw-   1 root     root       7,134 Sep 26  1986 /dev/cua6
crw-rw-rw-   1 root     root       7,198 Sep 26  1986 /dev/cua6.12ØØ
crw-rw-rw-   1 root     root       7,166 Sep 26  1986 /dev/cua6.3ØØ
crw-rw-rw-   1 root     root       7,  1 Sep 17 14:Ø6 /dev/culØ
crw-rw-rw-   1 root     root       7,  2 Sep 26  1986 /dev/cul1
crw-rw-rw-   1 root     root       7,  4 Sep 26  1986 /dev/cul4
crw-rw-rw-   1 root     root       7,  5 Sep 26  1986 /dev/cul5
crw-rw-rw-   1 root     root       7,  6 Sep 26  1986 /dev/cul6
brw-rw-rw-   1 sys      sys        1,  Ø Oct 2Ø 14:17 /dev/fdØ
brw-rw-rw-   1 sys      sys        1,  8 Sep 26  1986 /dev/fd1
brw-rw-rw-   1 sys      sys        1, 16 Sep 26  1986 /dev/fd2
brw-rw-rw-   1 sys      sys        1, 24 Sep 26  1986 /dev/fd3
brw-r--r--   1 root     root       1,  3 Sep 26  1986 /dev/fdbtØ
brw-r--r--   1 root     root       1, 11 Sep 26  1986 /dev/fdbt1
brw-r--r--   1 root     root       1, 19 Sep 26  1986 /dev/fdbt2
brw-r--r--   1 root     root       1, 27 Sep 26  1986 /dev/fdbt3
crw-rw-rw-   1 sys      sys        8,  Ø Sep 26  1986 /dev/graphics
brw-r-----   1 sys      sys        1, 32 Sep 26  1986 /dev/hdØ
brw-r-----   1 sys      sys        1, 4Ø Sep 26  1986 /dev/hd1
brw-r-----   1 sys      sys        1, 48 Sep 26  1986 /dev/hd2
brw-r-----   1 sys      sys        1, 56 Sep 26  1986 /dev/hd3
brw-------   1 root     root       1, 35 Dec 1Ø 1Ø:43 /dev/hdbtØ
brw-------   1 root     root       1, 43 Sep 26  1986 /dev/hdbt1
brw-------   1 root     root       1, 51 Sep 26  1986 /dev/hdbt2
brw-------   1 root     root       1, 59 Sep 26  1986 /dev/hdbt3
crw-r-----   1 sys      sys        3,  1 Dec 1Ø 1Ø:41 /dev/kmem
c-w--w--w-   1 root     root       6,128 Nov 2Ø 1Ø:18 /dev/lp
crw-r-----   1 sys      sys        3,  Ø Sep 26  1986 /dev/mem
crw-rw-rw-   1 root     root       3,  2 Dec 14 Ø1:22 /dev/null
```

```
cr--r--r--   1 sys      sys        3,  3 Sep 26  1986 /dev/procs
crw-rw-rw-   1 sys      sys        5, 64 Dec  3 Ø8:53 /dev/rcdØ
crw-rw-rw-   1 sys      sys        5, 72 Oct 2Ø  1986 /dev/rcd1
crw-rw-rw-   1 root     root       5, 67 Sep 26  1986 /dev/rcdbtØ
crw-rw-rw-   1 root     root       5, 75 Sep 26  1986 /dev/rcdbt1
brw-------   1 root     root       1, 36 Nov 24 14:33 /dev/recover
crw-rw-rw-   1 sys      sys        5,  Ø Dec 14 16:27 /dev/rfdØ
crw-r--r--   1 root     root       5,  1 May  6  1987 /dev/rfdØa
crw-r--r--   1 root     root       5,  2 May  6  1987 /dev/rfdØb
crw-rw-rw-   1 sys      sys        5,  8 Sep 26  1986 /dev/rfd1
crw-rw-rw-   1 sys      sys        5, 16 Nov 3Ø 16:Ø2 /dev/rfd2
crw-rw-rw-   1 sys      sys        5, 24 Sep 26  1986 /dev/rfd3
crw-------   1 root     root       5,  3 Sep 26  1986 /dev/rfdbtØ
crw-------   1 root     root       5, 11 Sep 26  1986 /dev/rfdbt1
crw-------   1 root     root       5, 19 Sep 26  1986 /dev/rfdbt2
crw-------   1 root     root       5, 27 Sep 26  1986 /dev/rfdbt3
crw-r-----   1 sys      sys        5, 32 Sep 26  1986 /dev/rhdØ
crw-r-----   1 sys      sys        5, 4Ø Sep 26  1986 /dev/rhd1
crw-r-----   1 sys      sys        5, 48 Sep 26  1986 /dev/rhd2
crw-r-----   1 sys      sys        5, 56 Sep 26  1986 /dev/rhd3
crw-------   1 root     root       5, 35 Sep 26  1986 /dev/rhdbtØ
crw-------   1 root     root       5, 43 Sep 26  1986 /dev/rhdbt1
crw-------   1 root     root       5, 51 Sep 26  1986 /dev/rhdbt2
crw-------   1 root     root       5, 59 Sep 26  1986 /dev/rhdbt3
c-w--w--w-   1 root     root       6,  Ø Sep 26  1986 /dev/rlp
brw-r-----   1 sys      sys        1, 33 Dec 1Ø 1Ø:4Ø /dev/root
crw-r-----   1 sys      sys        5, 33 Sep 26  1986 /dev/rroot
crw-rw-rw-   1 root     root       8,  1 Sep 26  1986 /dev/screen
brw-r-----   1 sys      sys        1, 34 Sep 26  1986 /dev/swap
crw-rw-rw-   1 root     root       2,  Ø Jan  6  1987 /dev/tty
crw-rw-rw-   1 root     root       Ø,  1 Dec  9 14:48 /dev/ttyØ1
crw-rw-rw-   1 root     root       Ø,  2 Sep 26  1986 /dev/ttyØ2
crw-rw-rw-   1 root     root       Ø,  4 Sep 26  1986 /dev/ttyØ4
crw-rw-rw-   1 root     root       Ø,  5 Sep 26  1986 /dev/ttyØ5
crw-rw-rw-   1 root     root       Ø,  6 Sep 26  1986 /dev/ttyØ6
```

Files in the /usr directory:

```
-rwx------   1 sys      sys          1Ø46 Dec 26  1984 /usr/bin/dt1ØØ-install
-rwx------   1 sys      sys          2642 Dec 26  1984 /usr/bin/dt1ØØ-install.
-rws--x--x   1 root     bin         14952 Apr  8  1987 /usr/bin/hello
-rwxr-xr-x   1 sys      sys           933 Apr  1  1987 /usr/bin/install
-rwxr-xr-x   1 bin      bin           16Ø Apr 3Ø  1985 /usr/bin/local
-rwx--x--x   1 bin      bin          8814 Apr  7  1987 /usr/bin/logname
-rwx--x--x   1 bin      bin          8858 Apr  8  1987 /usr/bin/lpq
-rwx--x--x   1 bin      bin          8888 Apr  8  1987 /usr/bin/lprm
-rwxr-xr-x   1 bin      bin         28458 Apr  7  1987 /usr/bin/more
-rwx------   1 bin      bin         113Ø4 Apr  1  1987 /usr/bin/patch
-rwxr-xr-x   1 sys      sys          3199 Apr 3Ø  1985 /usr/bin/save
-rwxr-xr-x   1 bin      bin           16Ø Apr 3Ø  1985 /usr/bin/spool
-rwxr-xr-x   1 bin      bin          9228 Apr  8  1987 /usr/bin/spooler
-rwx--s--x   2 root     sys         167ØØ Apr  8  1987 /usr/bin/uptime
-rwx--s--x   2 root     sys         167ØØ Apr  8  1987 /usr/bin/w
-rwx--x--x   1 bin      bin        1Ø2436 Apr  8  1987 /usr/bin/xmail
-rws--x--x   1 root     bin         12Ø8 Apr  8  1987 /usr/lib/ex3.7preserve
-rws--x--x   1 root     bin         16462 Apr  8  1987 /usr/lib/ex3.7recover
-rw-r--r--   1 bin      bin          63Ø2 Apr  8  1987 /usr/lib/ex3.7strings
-rw-r--r--   1 bin      bin           264 May 13  1987 /usr/lib/crontab
-rwx------   1 cron     other       22224 Apr  8  1987 /usr/lib/idle
-rws--x--x   1 root     bin         21Ø5Ø Apr  7  1987 /usr/lib/lpd
-rw-r--r--   1 sys      sys           332 Oct 16  1986 /usr/lib/mkuser/mkuser.cshrc
-rw-r--r--   1 sys      sys           757 Apr 3Ø  1985 /usr/lib/mkuser/mkuser.help
-rw-r--r--   1 sys      sys           162 Oct 16  1986 /usr/lib/mkuser/mkuser.login
-rw-r--r--   1 sys      sys            19 Apr 3Ø  1985 /usr/lib/mkuser/mkuser.mail
-rw-r--r--   1 sys      sys           288 Apr 3Ø  1985 /usr/lib/mkuser/mkuser.prof
-rw-r--r--   1 bin      bin          1Ø95 Apr  7  1987 /usr/lib/more.help
-rw-r--r--   1 bin      bin           157 Mar  6  1985 /usr/lib/tsh.help/.setenv
-rw-r--r--   1 bin      bin           567 May 22  1985 /usr/lib/tsh.help/again
-rw-r--r--   1 bin      bin          1Ø6Ø May 22  1985 /usr/lib/tsh.help/alias
-rw-r--r--   1 bin      bin          1Ø64 May 22  1985 /usr/lib/tsh.help/auto
-rw-r--r--   1 bin      bin           311 May 22  1985 /usr/lib/tsh.help/backup
-rw-r--r--   2 bin      bin           448 May 22  1985 /usr/lib/tsh.help/cd
-rw-r--r--   2 bin      bin           448 May 22  1985 /usr/lib/tsh.help/chdir
-rw-r--r--   1 bin      bin            98 Apr 25  1985 /usr/lib/tsh.help/cls
-rw-r--r--   1 bin      bin           495 May 22  1985 /usr/lib/tsh.help/copy
-rw-r--r--   1 bin      bin           188 May 21  1985 /usr/lib/tsh.help/device
-rw-r--r--   1 bin      bin          3553 May 22  1985 /usr/lib/tsh.help/dir
-rw-r--r--   1 bin      bin           335 May 22  1985 /usr/lib/tsh.help/dismount
-rw-r--r--   1 bin      bin          4971 May 22  1985 /usr/lib/tsh.help/display
-rw-r--r--   1 bin      bin           283 May 22  1985 /usr/lib/tsh.help/do
-rw-r--r--   1 bin      bin           298 May 22  1985 /usr/lib/tsh.help/exit
-rw-r--r--   1 bin      bin          2Ø59 May 22  1985 /usr/lib/tsh.help/files
-rw-r--r--   1 bin      bin           3Ø7 May 22  1985 /usr/lib/tsh.help/free
-rw-r--r--   1 bin      bin          1454 May 22  1985 /usr/lib/tsh.help/help
-rw-r--r--   1 bin      bin           728 May 22  1985 /usr/lib/tsh.help/history
```

```
-rw-r--r--   1 bin      bin          252 May 22  1985 /usr/lib/tsh.help/i
-rw-r--r--   1 bin      bin          933 May 22  1985 /usr/lib/tsh.help/kill
-rw-r--r--   1 bin      bin          117 Apr 25  1985 /usr/lib/tsh.help/lib
-rw-r--r--   1 bin      bin         1817 Oct 17  1986 /usr/lib/tsh.help/mount
-rw-r--r--   1 bin      bin          493 May 22  1985 /usr/lib/tsh.help/move
-rw-r--r--   1 bin      bin          971 May 22  1985 /usr/lib/tsh.help/print
-rw-r--r--   1 bin      bin          141 May 22  1985 /usr/lib/tsh.help/rename
-rw-r--r--   1 bin      bin          912 May 22  1985 /usr/lib/tsh.help/restore
-rw-r--r--   1 bin      bin          926 May 22  1985 /usr/lib/tsh.help/save
-rw-r--r--   1 bin      bin         134Ø May 22  1985 /usr/lib/tsh.help/set
-rw-r--r--   1 bin      bin          2Ø9 Apr 25  1985 /usr/lib/tsh.help/showdir
-rw-r--r--   1 bin      bin          572 Apr 25  1985 /usr/lib/tsh.help/source
-rw-r--r--   1 bin      bin          113 Apr 25  1985 /usr/lib/tsh.help/time
-rw-r--r--   1 bin      bin          13Ø May 22  1985 /usr/lib/tsh.help/version
-rw-r--r--   1 uucp     uucp         81Ø Apr 3Ø  1985 /usr/lib/uucp/L-devices
-rws--x--x   2 uucp     uucp       17166 Apr  8  1987 /usr/lib/uucp/dial
-rws--x--x   1 uucp     uucp       2Ø568 Apr  8  1987 /usr/lib/uucp/dialHAYES
-rws--x--x   2 uucp     uucp       17166 Apr  8  1987 /usr/lib/uucp/dialTRS
-rws--x--x   1 uucp     uucp       2Ø52Ø Apr  8  1987 /usr/lib/uucp/dialVADIC
-rw-r--r--   1 bin      bin         1177 Jan 19  1987 /usr/lib/xmail.cmd.hlp
-rw-r--r--   1 bin      bin         1174 Apr  8  1987 /usr/lib/xmail.opt.hlp
-rw-r--r--   1 bin      bin          145 Sep 18  1986 /usr/lib/xmail.rc
-rw-r--r--   1 bin      bin         1216 Apr  8  1987 /usr/lib/xmail.set.hlp
-rwx------   1 bin      bin        12164 Apr  8  1987 /usr/lib/xmail_recover
-rws--x--x   1 root     bin         4436 Apr  8  1987 /usr/lib/xmlock
```

Files in the /lib directory

```
-rwx--x--x   1 bin      bin        11Ø4 Apr  7  1987 /lib/cvtdate
```

Files in the / directory

```
-rw-r--r--   1 root     bin         282 Feb  3  1987 /.profile
-rw-r--r--   1 sys      sys        5649 Mar  3  1987 /hdbootr
-rw-r--r--   1 sys      sys        1577 May  3  1985 /z8Ødiags
-rw-r-----   1 sys      sys       16846 Mar  3  1987 /diskutil
-rw-r--r--   1 sys      sys        3277 Mar  3  1987 /fdboot
-rwxr--r--   1 bin      bin        1Ø74 Feb 12  1985 /firsttime
-rw-r--r--   1 sys      sys        5649 Mar  3  1987 /hdboot
-rw-r--r--   1 sys      sys       74935 Dec  9 Ø9:46 /xenix
-rw-r--r--   1 sys      sys       1567Ø Apr  8  1987 /z8Øctl
```

****

A Glossary of Commonly Used Xenix Terms

****

Access Mode
Access Permissions — The protection information for a file. In Xenix, files can be read, written, or executed by the file's owner, members of the owner's group, or others. The access mode details the operations allowed (reading, writing, or executing) by the three classes of users (owner, group, and others). In a shared computer system access modes ensure a degree of privacy and safety for the user's files.

'/appl' — Directory which is a common location for many **application programs**.

Application Program — A computer program for a specialized purpose, such as an accounting program or a word processing program.

Argument — Additional information that is passed to a command. The command name and its arguments are separated from one another by spaces and/or tabs. Sometimes certain arguments called **options** are indicated with the "hyphen" or "minus" symbol (-), (for example, the command "ps -l" where "-l" is an option) but this is a convention which is not well observed. Arguments usually are used to direct the operation of a command.

Background Process — A process that runs unattended in a manner that allows other programs to be initiated and interacted with while the background process is running. Some background processes are started during the Xenix boot procedure in order to perform system management functions while other background processes are started interactively by users to perform personal work. To invoke a background process interactively, one puts an ampersand (&) at the end of the command line.

'/bin' — On Xenix systems, this is the directory that contains the most frequently used commands.

Bourne shell — The shell program which is most frequently invoked by logging in on the Xenix system. Defaults to a dollar sign prompt ($) for users and a pound sign prompt (#) for the root user. Named after its author, S. R. Bourne.

Buffer — In the Xenix sense, this is a place where data are stored temporarily.

Character — The symbols corresponding to the keys on the terminal keyboard including all alphanumerics, punctuation marks, and other special symbols. Characters are stored in a single byte in Xenix.

C Language — A general purpose programming language that is the primary language of the Xenix system.

Cleaning — In Xenix terminology, the process of clearing up inconsistencies in the file system.

Command — An order directing the system to perform some function. Some commands are handled internally by the shell, although most commands result in the execution of a program.

Command File — Similar to a ".BAT" file in MS-DOS, this is an ordinary file which contains shell commands. The term command file is usually used when the file contains just one or a few commands; the term shell program or shell script usually is used when there are a lot of commands or when the shell's facilities for looping and conditional execution are used.

Command Interpreter — A component of an operating system (in Xenix, it is itself a program) which decodes and executes the commands entered by the user. The Xenix system command interpreter is called the shell.

Command Name — The first word of a command. The words following the command name are called the arguments. Sometimes the command name is referred to as the zeroeth argument because in programs the subsequent arguments are numbered consecutively beginning with one.

Compiler — A computer program that translates a text file containing a program written in some high level programming language into a machine language output that can be loaded and executed. On Xenix systems the machine language output from a compiler is called an object file.

Cat
Concatenate — The act of combining several files, one after the other. The operation usually is performed using the **cat** program.

Conditional — A programming language construct that causes a statement (or statements) to be executed only if a certain condition exists. The shell has provisions for conditional statements.

Console — In Tandy Xenix, the host computer.

Context Search — Searching for a body of text in a given file by entering a text pattern you want the system to locate. You can perform a context search within the editor or you can perform context searches using the **grep** command.

Control Character — Control characters are embedded in text in order to control various functions, such as cursor movement or printing functions. A control character is typed at a terminal by depressing the control key (CTL or CTRL) and an alphabetic key simultaneously.

Control-d — See **EOF character**.

Core System — In Xenix, the basic unit of the operating system that you must have to run application programs.

Csh
Cshell — An alternate shell program which comes with the **development system**, or Xenix 3.2. It is so named due to the similiarity it has to the C programming language.

Current Directory — The directory whose files are directly accessible using **relative pathnames**, without having to traverse another directory. At all times in your interactions with the Xenix system there is a current directory; the name of the current directory can be printed using the **pwd** command and you can move to another directory by using the **cd** command.

Current Subtree — The subtree whose root is the **current directory**; that is, the structure composed of the current directory, all of its subdirectories and files, their subdirectories and files, and so on.

'/dev' — The directory where **special files** usually are located.

'/dev/null' — See **Null Device**.

Development System — In Xenix, a group of programs and utilities with which you can make other programs and do typesetting and printing. It is separate from the core system.

Directory — A group of files. Directories are used to organize and structure the file system. Without the organization provided by the Xenix system's hierarchical directory system it would be very difficult to manage the thousands of files that exist in typical Xenix system installations. The **l**, **lc**, and **ls** commands are used to list the files in a directory. When you first log onto the system you are in your **home directory**. You can move to another directory by using the cd command and you can print the name of the current directory by using the **pwd** command.

Disable — Disconnect or disallow electronically or through software.

Disk File — A named collection of information that is stored on a mass storage device. Disk files are said to be nonvolatile because they are retained even when operating power is removed from the mass storage device.

Diskutil — In the 68ØØØ based Xenix versions, the standalone Z8Ø based utility which allows format and copy operations on floppy, hard, and cartridge drives. On hard drives, **diskutil** also records bad track information, size of the drive, and maximum allowed number of bad tracks.

Echoing — The Xenix system's repetition of the user's typed input. The characters that you type are normally sent to the Xenix system and then echoed so that they appear on your terminal. Echoing is occasionally turned off (for instance, when you enter your password).

Edit — To change or alter information. Often used in reference to changing the information in a text file by using an editor.

Electronic Mail — A system which transmits information to other users of the system or to users on other systems.

Enable — Connect or allow electronically or through software.

EOF Character — EOF (meaning End Of File) is usually control-d. This is the Xenix system's end of file character. Since the shell normally stops processing when it encounters the end of file, one way to log off the Xenix system is to strike control-d in response to a shell prompt. The EOF character may be reassigned (with some obvious care taken in choice) to another key with the **stty** command.

Erase Character — The erase character will erase previously typed characters on your current input line one at a time. It is assigned initially to the backspace or control-h key for the most part; it may be reassigned using the **stty** command.

'/etc' — The Xenix system directory which contains miscellaneous files used for system administration.

'/etc/passwd' — The Xenix system file which contains the major login information (password, login name, user ID, group ID, home directory and the name of the user's shell) for each user on the system.

Execute Permission — For ordinary files execute permission is an access mode that allows you to execute the file. For directory files execute permission is an access mode that allows you to search them in the course of resolving a pathname.

Execution Time — The time the computer requires to complete a given command.

File — A named collection of information. Files usually are stored on a mass storage device and Xenix system files are collected into groups called directories (which are also themselves files).

Filename — The name that is used to identify a particular file.

Filename Generation — The procedure used by the shell to expand command line words containing **metacharacters** into the corresponding list of filenames. For example, in a directory containing the files 'x.doc' and 'nm.doc' the shell file name generation process expands the word "*.doc" into the list of files 'nm.doc' and 'x.doc'.

File System The collection of files and file management structures (**inodes**) on a given mass storage device. In Xenix the file system is hierarchical.

Filter A program that reads its information from the **standard input** and writes its results to the **standard output**.

Foreground Process A process that is run interactively. There may be several active background commands, but under normal circumstances there is just one active and perhaps several inactive foreground processes.

Flag See **Option**

Group Several users who are members of the same department, working on the same project, or related in other ways. Each Xenix file is associated with a certain group and members of that group have specified privileges for accessing the file.

Hierarchy Any system of persons or things that can be ranked or ordered one above the other. The Xenix file system is hierarchical.

High Level Language A general term for programming languages which support an abstract view of the process of computing, in contrast to assembly languages which relate directly to a particular computer architecture. C, FORTRAN, BASIC, and Pascal are examples of high level languages.

Home Directory The directory the user is placed into at the conclusion of the login process.

Inode The key internal structure for managing files in the Xenix system. Inodes contain all of the information pertaining to the mode, type, owner, and location of a file. A table of inodes (called the **superblock**) is stored near the beginning of every file system.

Input Redirection The shell's reassignment of the standard input to a file other than the terminal. See **Standard Input**. This may be denoted by the "<" symbol, for example: "mail stewart <text.file".

Interactive System A conversational system that allows for a continuous dialogue between the user and the computer.

I-number — A number specifying a particular inode on a file system.

I/O devices — Peripheral devices to which information can be transmitted from the computer (i.e., printer, terminal, disk, etc.). In the Xenix system I/O devices are accessed via **special files**.

Kernel — The memory resident part of the Xenix system, containing all of the Xenix system functions that are needed immediately and frequently. The kernel (called /xenix) controls I/O transactions (in conjunction with /z8Øctl in 6ØØØ systems), manages and controls the hardware (68ØØØ based hardware in the 6ØØØ) and schedules the user processes for execution. In Xenix, the kernel is endowed with relatively few features (compared to other operating systems) so that these features may be provided more conveniently by individual utility programs.

Kill Character — The kill character allows a user to erase the whole line (although it may still appear on the screen, the shell will ignore all characters typed prior to the kill character). It is initially assigned (normally) to the control-u key; it may be reassigned with the **stty** command.

Link — In Xenix, an inumber-filename pair, such as those seen in directories.

Login — The procedure that provides access to the system for authorized users.

Login Directory — See **Home Directory**.

Login Name — The name that a user uses during the login process.

Logout — The procedure that informs the system that the user will be exiting and placing no further demands on the system.

Mass Storage Device A unit for storing large amounts of information; usually a hard disk, floppy disk, tape, or disk cartridge unit. The information stored on mass storage devices is accessible to the cpu, although the access time is much longer than that for information stored in the main memory. Information is stored magnetically on most mass storage devices.

Metacharacter Keyboard characters which have special meanings in certain situations. For example, the asterisk (*) can be used to match any sequence of characters when you are entering a shell command such as "ls *.c". If you wish to use metacharacters without invoking their special meanings, you must use quotation. Bourne Shell metacharacters are *, ?, and []. Their meanings are:

*	Matches any string, including the null string.
?	Matches any single character
[...]	Matches any one of the characters enclosed in brackets. A pair of characters separated by - matches any character lexically between the pair.

'/etc/motd' The Message Of The Day file, a text file containing timely information or welcome messages placed there by the system administrator. It is displayed when you log in.

Mount In Xenix, to allow access to electronically, in the sense that a secondary hard disk is **mounted** before the information on its filesystem may be accessed.

Multitasking
Multiprogramming The ability to run several programs or routines simultaneously on a single computer.

Multiuser Able to support several users simultaneously.

Null Device The Xenix system null device is called '**/dev/null**'. When you direct output to the null device it is discarded; when you read input from the null device you immediately encounter an end of file. Output occasionally is directed to the null device in order to discard it, and input occasionally is read from the null device in order to read nothing.

Null String — A text string that does not contain any text. The length of a null string is zero.

Object File — A file containing machine language instructions that can be used by a computer. In the Xenix system an object file is the result of a compilation.

Operating System — A program for managing the resources of a computer. Operating systems simplify housekeeping duties such as input/output procedures, process scheduling, and the file system.

Option — An argument that alters the operation of a command. Usually options are single characters preceded by a hyphen. For example, in the shell command "ps -l" the option is the letter "l" which directs the **ps** command to perform a long format list of processes rather than the usual short form list.

Ordinary File — Ordinary files are used for storing data. Ordinary files often contain programs, documents, letter, data bases, and other types of information. In long form directory listings they are denoted by "-".

Output Redirection — The shell's reassignment of the standard output connection to a specific file. See **Standard Output**.

Password — A unique set of letters or digits which a user enters during an identification process.

Pathname — A path through the file system that leads to a file. It is formed by listing directory names separated by "/" in order to define the path. Pathnames may be separated into two basic types: **relative** and **absolute**. **Relative** pathnames begin with the current directory. **Absolute** pathnames begin in the root directory (or /). For example, the absolute pathname /usr/bin/lex specifies a path that starts in the root directory, leads to the /usr directory, then to the /usr/bin directory, then finally to the file called lex in the /usr/bin directory.

PATH shell variable The shell maintains a search string that directs the shell to search in a certain set of directories for each command that is entered. The search list usually includes the current directory (except for root users), the '/bin' directory, and the '/usr/bin' directory. The PATH variable may be altered by assigning a new value to it.

Permissions The access modes associated with a file. See **Access Mode**.

Pipe A connection between the **standard output** of one program and the **standard input** of another program. For example, the shell command to create a pipe between the **ls** command and the **lpr** command is "ls | lpr".

Pipeline A group of commands connected by pipe connections.

Process A program that is being executed. An entry in the system's process table.

Process ID Number A unique process ID number is assigned to each process by the Xenix system kernel. The identification numbers for the current processes are printed by the by the **ps** command. When you run a program in the background the shell prints its PID number.

'.profile' A file of shell commands that may reside in a user's home directory. If there is a '.profile' in a home directory, then the shell will execute the commands in it before executing commands interactively from the terminal each time the user logs on. Commands in '.profile' are often used to initialize shell variables (e.g., the PATH search string), set the terminal handler's modes, and so on.

Prompt A message printed by a program to indicate that the program is ready to accept another command from the user. The prompt for the shell can be changed by assigning a value to the variable PS1. The default prompt for the Xenix system Bourne shell is a '$' for users and a '#' for root.

Queue In Xenix, a list of requests.

Quotation	The process by which metacharacters are prevented from having their special meaning.
Read Permission	Allows a person to execute a program that reads data from a file.
Regular Expression	A regular expression specifies a set of strings of characters.
Shell	A command interpreter and programming language that provides an interface to the Xenix system. As a command interpreter it interactively accepts commands from users and arranges for the requested actions to occur. As a programming language it contains control flow and string valued variables. The program which implements the Bourne shell is called '/bin/sh'.
Shell Program	A program written using the shell programming language. Shell programs can be written and used interactively, although most shell programs are stored in ordinary files.
Source Code	The human readable, text version of a program. A **compiler** transforms **source code** into **object code**.
Single User	Able to support just one user. In Xenix, the single-user mode is usually entered just after the system is booted and it is usually used for file system repair and maintenance and other functions where one person requires exclusive use of the computer. It is the best mode in which to perform backups. Even when it is in single-user mode, Xenix is able to run several processes (**multiprogramming** or **multitasking**).

Special File

Special files are used in Xenix to provide an interface to I/O devices. Each Xenix system contains at least one special file for each I/O device that is connected to the computer. Special files can be accessed using the same techniques that are used to access ordinary files. Special files usually reside in the '**/dev**' directory and there are five types supported in System 3 and System 5. Only two types are supported in Version 7.

Block special (b): based on a specific model, these files provide for dealing with information a block (typically 512 bytes) at a time. Frequently used on disks for filesystems.

Character special (c): these files provide a flexible interface allowing data to be moved back and forth in amounts ranging from a character at a time on up. Terminals are character special files; most disks also have a character file.

FIFO or Named Pipe (p): a permanent pipe which allows two unrelated processes to pass information back and forth without synchronization difficulties. Used for interprocess communication.

Semaphore (s): a device which may be "set" as a flag for another process to interpret. Used for interprocess communication.

Shared memory (m): a section of memory which may be shared among processes. Used for interprocess communication.

Block and **character** special files are supported by all Tandy Xenix versions. Version 7 does not support named pipes, semaphores, or shared memory.

Standard Error

The place where many programs place error messages. In interactive programs, this is typically the terminal.

Standard I/O — Many programs need to read commands and data from the user, write messages to the user, and write error messages. Therefore, the shell prepares three standard I/O connections for each program, the standard input, the standard output, and the standard error. The standard channels usually are connected to the user's terminal although they can be reassigned using redirection.

Standard Input — The sources of input for many programs. See **Input Reassignment**.

Standard Output — The place where many programs place their text output. See **Output Reassignment**.

Subdirectory — A directory below another directory in the file system hierarchy. For example, the directory '/usr/bin' is a subdirectory of the '/usr' directory.

Subtree — A branch of the Xenix file system.

Superuser — A special privilege level that exists in Xenix to allow system managers to perform certain functions that are denied to ordinary users. The superuser is not constrained by the normal file access mode system.

Swapping — Occasionally more processes are executing than can be stored in main memory. When this occurs the excess processes are stored temporarily on a mass storage device, a procedure known as swapping. The act of transferring a process from RAM to swap space is called swapping out; the opposite procedure is called swapping in.

Swap Space — The region on a mass storage device where processes are stored after they are swapped out.

Syntax — The rules that govern the construction of sentences in a language. In computers the term syntax is used to describe the rules for writing legal statements in programming languages or command languages.

System Call — A request by an active process for a service by the Xenix system kernel. The Xenix system contains system calls to perform I/O, to control, coordinate, and create processes, and to read and set various status elements of the system.

TERM variable — The shell variable which informs the command interpreter and many programs what type of terminal the user is on. This may be set from the shell, or in the **.profile** file.

Text Editor — A general purpose program used to prepare text files. A text editor enables a user to enter and correct text. Most editors contain commands to locate specific lines or words in the text and commands to add, delete, change, and print lines in the text. The basic Xenix editor is called **ed**.

Text Formatter — A program that is used to prepare text for final publication or printing. It is also used to prepare the online manual pages. The basic Xenix text formatting commands are nroff, troff, eqn, neqn, and tbl.

Text File — A file comprised solely of ASCII characters. This is not a specific file type, but a colloquialism describing the file contents.

Time Sharing — A technique developed to share a computer's resources among several users, so that each user is given the illusion of having exclusive use of a computer. This is accomplished by switching very rapidly from one task to another so that it appears that all of the activities are occurring simultaneously.

'/tmp' — A directory which is used for temporary work files. Normally, the contents of this directory are removed upon each system re-boot.

'/usr' — A general purpose directory that is the head of a subtree that contains most of the user directories, documentation, and all but the most important programs on the system.

'/usr/bin' — A directory that is used to store the less frequently used utility programs.

Variable — A symbol whose value is allowed to change. In the shell variable names are preceded by a dollar sign except during assignment operations. Some of the standard shell variables are PATH, PS1, and TERM.

Word Processing A generic term that connotes working with text data using a computer to produce documents. The software components of the system usually include a text editing program and a text formatting program.

Write Permission Allows a user's programs to write data to a file.

Zeroeth Argument See **Command Name**.

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.

RADIO SHACK
A Division of Tandy Corporation
Fort Worth, Texas 76102