5

MINIX ON THE APPLE MACINTOSH

In this chapter we will describe how to boot and install MACMINIX on the Apple Macintosh. It is assumed that the reader is already familiar with MACMINIX in general, and has at least some knowledge of UNIX. Readers not at all familiar with UNIX should probably begin by looking at one of the many introductory articles and books about it, as this manual does not contain any tutorial material on UNIX.

If you plan on running multifinder with MACMINIX, be sure to read the multifinder section below.

5.1. MACMINIX HARDWARE REQUIREMENTS

MINIX will run on any Macintosh with at least one megabyte of memory and 128K (or larger) ROMs. MACMINIX has been tested most extensively with system software version 6.0 or later. Earlier versions may present some problems.

5.2. THE MACMINIX DISTRIBUTION

The MACMINIX distribution consists of eight double-sided 800K diskettes. One of them contains the boot application and the root file system, and is used for booting MACMINIX. Since MACMINIX file systems are also ordinary Macintosh operating

system files, all of the diskettes are readable by the normal Macintosh operating system. Here is the list of the diskettes:

center,box; c c c c l l l l. Name Size File system Description _ 00.BOOT 800K MAC OS/MINIX Used for booting MINIX 01.USR1 800K MAC OS/MINIX Commands 02.USR2 800K MAC OS/MINIX Commands 03.ACK 800K MAC OS/MINIX C compiler 04.SRC1 800K MAC OS/MINIX Sources of MINIX 05.SRC2 800K MAC OS/MINIX Sources of commands 06.SRC3 800K MAC OS/MINIX Sources of commands 07.SRC4 800K MAC OS/MINIX Editors

We will refer to these diskettes in the rest of this chapter by their name in the first column of this table, for example, 00.BOOT.

Before you start working with these diskettes we strongly advise you to make copies of them. You can use normal Macintosh procedures to make these copies. If you are not familiar with how to copy a diskette, refer to your Macintosh owner's guide. Keep the original disks write protected under all circumstances. *Make sure that the copies are named identically to the originals*, since the following procedures depend on this. Once you have made the copies, place the originals in a safe place and use the copies. Please note that if you use write protected floppies in the following installation procedures, you will occasionally get write errors on device 2/0. These messages can safely be ignored.

5.3. NATIONAL KEYBOARDS

The Macintosh has different keyboards for different countries. When booting, MACMINIX uses the Toolbox to assist in the creation of the virtual key code to ASCII translation table, so assuming that the international resources have been properly configured for your machine, the table will be correct for your country.

5.4. BOOTING MACMINIX

This section presents a boot procedure for MACMINIX that works on all Macintosh configurations. Following sections describe how to adapt the set of diskettes so that you can use MACMINIX effectively on your particular combination of memory and disk drives. For example, if you have more than 1 megabyte of memory but no hard disk, you may wish to increase the size of the RAM disk to 475K. If you have a hard disk, all of the diskettes can be copied onto one or more of its partitions. Finally, some of the options for booting MACMINIX will be explained. But first, the procedure for booting that works on all configurations.

Throughout the discussion below, lines printed in the Helvetica typeface are either commands you should type on the keyboard or lines that the computer will display for you. In a few of the examples, *italics* characters or words appear in a command. These represent values that you are to fill in.

Booting is a three-stage procedure. First the operating system itself is loaded into memory. The root file system is then copied to a RAM disk allocated in memory. Finally, the script */etc/rc* is executed and you are asked to log on.

To boot MACMINIX, proceed as follows:

1. Follow your normal booting conventions to boot your Macintosh.
2. Place the boot diskette, 00.BOOT in drive 0, and launch the Macboot application by double clicking on it. A window will appear, and the MACMINIX boot application will exhibit a status line as it loads each of the initial software components of MACMINIX: the *kernel*, memory management (*mm*), the file system (*fs*), and process 0 (*init*). When loading is complete, this window will disappear.
3. The main console window will then display (approximately):

   ```
   Booting MACMINIX 1.5.  Copyright 1990 Prentice-Hall, Inc.
   Memory size=775  MINIX=165  RAM disk=190K  Available=475K
   ```

   for a system with 1M of RAM. The memory not accounted for is left for the Macintosh operating system to use. The amount of memory left can be configured by you. See the Configuration section below.
4. A fourth line will be displayed that reads:

   ```
   RAM disk. To load:  190K        Loaded:  0K
   ```

   Again, the number may vary. In rapid succession the number 0 will be increased in steps of 5K, until the whole line is replaced by:

   ```
   RAM disk loaded.
   ```
5. When the RAM disk is loaded, the system initialization file, */etc/rc*, is executed. It ejects the boot diskette (00.BOOT) and asks you to insert the */usr* file system (01.USR1) in a drive and type a RETURN. Do so.
6. After */usr* has been mounted, you will now get the message:

   ```
   login:
   ```

   on the screen. Type:

   ```
   root
   ```

   and wait for the system to ask for your password. When it does, type:

```
Geheim
```

being careful to type the first letter in upper case. Lower and upper case letters are always distinct in MACMINIX. Alternatively, you could have used the name ‘‘ast’’ together with the password ‘‘Wachtwoord’’. This is much preferred when you use the system normally, but for now it is troublesome.

7. If you have successfully logged in, the shell will display a prompt (sharp sign for root, dollar sign otherwise) on the screen. Try typing:

   ```
   ls –l
   ```

   to see what is in the root directory. Note that you need six keystrokes: ‘‘l’’, ‘‘s’’, space, ‘‘–’’, ‘‘l’’, and a RETURN. Then type:

   ```
   ls –l /bin
   ```

   to see what is in the */bin* directory on the root device (RAM disk). After that, try:

   ```
   ls –l /usr/bin
   ```

   to see what is on the drive 0 diskette. To stop the display from scrolling out of view, type CTRL-S; to restart it, type CTRL-Q. (Note that CTRL-S means depress the ‘‘Control’’ key on the keyboard and then hit the *S* key while ‘‘control’’ is still depressed. If your keyboard does not have a control key, you may use the option key instead.)

8. You can now edit files, compile programs, or do many other things. The reference manuals given in chapters 8 and 9 of this manual give a brief description of the programs available. However, before rushing off we advise you to adapt the system to your hardware configuration first, as described in the next section.

9. When you are finished working and want to log out, type CTRL-D.

   ```
   login:
   ```

   will appear, and you or another user can log in again.

10. When you want to leave MACMINIX, make sure all processes have finished, if need be, by killing them with *kill*. Then type: *sync* or just log out. Your can then select the ‘‘Quit’’ menu item from the ‘‘File’’ menu, and this will return you to your familiar Macintosh desktop. Never quit without first running *sync* or logging out (which does an implied *sync*). Failure to obey this rule will generally result in a garbled file system and lost data.

5.5. INCREASING THE SIZE OF YOUR RAM DISK

If you have more then 1M of memory, and are not planning on using a hard disk, we advise you to increase the size of the RAM disk from 190K to 475K. This allows you to use the RAM disk to copy complete or partial file systems from one diskette to another. It also gives you plenty of space to add a few more utilities to the root file system. Finally, it allows you to compile much larger programs without running out of disk space for the intermediate results. However, it leaves you with somewhat less memory to run your MACMINIX applications. Even so, that is more than sufficient to recompile most the sources and perform many other complicated tasks.

To install a 475K RAM disk, you must first make a 475K root file system diskette as described below. When MACMINIX is booted, it looks at the size of the root file system and sets its size accordingly. To do this, proceed as follows.

1. Take an empty, formatted, 800K diskette, name it TEMP, and copy the Macboot application file from your original 00.BOOT onto the new diskette (in the normal Macintosh way).

2. Boot MACMINIX with the original 00.BOOT disk and login as root. Then type:

```
for i in maccreate mkfs mknod chmod; do cp /usr/bin/$i /bin; done
/etc/umount /dev/fd0
/etc/hdclose /dev/fd0
/etc/eject
```

3. Insert the *new* TEMP diskette in drive 0 and type:

```
maccreate 475 TEMP:ROOT
/etc/hdopen TEMP:ROOT /dev/fd0
mkfs /dev/fd0 475
/etc/mount /dev/fd0 /user
cpdir –msv / /user
```

4. Logout by typing CTRL-D.

5. Quit from MACMINIX by selecting ‘‘Quit’’ from the ‘‘File’’ menu. Rename your TEMP diskette 00.BOOT like the original. Restart MACMINIX using the above booting procedure, but use your newly created 00.BOOT diskette in place of the original 00.BOOT diskette.

The program *cpdir* is able to copy the devices in */dev*. *Cpdir* also will tell you that it skipped the directory */user* to avoid recursion.

By changing the argument *475* to *maccreate* and *mkfs* you can adapt the size of the RAM disk. Note that a copy of the programs *maccreate*, *mkfs*, *mknod* and *chmod* will be present in */bin* on the new 00.BOOT.

5.6. ADAPTING PROGRAMS TO USE EXTRA RAM

As distributed, the C compiler is tuned to work on even with the smallest Macintosh memory configuration. This may cause problems if you want to compile large source files. The first part of the C compiler proper, */usr/lib/cem*, as distributed, will compile most source files, but you may need to increase its memory allocation for larger source files.

You are strongly advised to execute the following procedure now if you have more than the minimal 1M of memory.

1. Boot MACMINIX and login as root.
2. Type:

```
cp  /usr/bin/chmem  /bin
/etc/umount  /dev/fd0
/etc/hdclose  /dev/fd0
/etc/eject
```

3. Insert 03.ACK in drive 0 and type:

```
/etc/hdopen  03.ACK:ACK    /dev/fd0
/etc/mount  /dev/fd0  /user
chmem  +50000  /user/usrlib/cem
```

A similar procedure can be executed if you encounter any other program that needs more memory. *chmem* takes a little getting use to, but it is difficult to avoid in a general-purpose multiprogramming system for a machine without a proper memory management unit.

5.7. USING A HARD DISK

The Macintosh version of MINIX is quite different from the other versions in that it is not a *stand-alone* operating system. That is, the IBM and other versions completely take over the hardware once they begin execution, while MACMINIX runs in tandem with the normal Macintosh operating system, even depending on it for

certain services, like accessing the hard disk, drawing and manipulating the menus, and drawing the tty windows. This has some drawbacks, especially with regard to speed, but it has the attraction that you can still have some of the things that Macintosh owner's like about their machine, such as menus and windows. In addition, if you have enough memory, you can run multifinder and simultaneously still use all your other Macintosh software.

However, the Macintosh file system is completely incompatible with the MINIX file system for a number of reasons, and therefore they do not share the same file name space. Instead, MACMINIX uses the Macintosh operating system to request it to set aside some number of (if possible, contiguous) disk blocks into a Macintosh file. The logical blocks of this file are then used as a MACMINIX disk partition.

You can have up to nine of these Macintosh files (as distributed; you can recompile the system to get more), and together they make up a logical MACMINIX disk. A MACMINIX partition can then be mapped onto the file by means of the *hdopen* MACMINIX utility program.

If you so desire, the Macintosh files that make up the disk can also be backed up onto your tape or diskette with the rest of your Macintosh files, using your normal Macintosh backup software. However, if you choose to do your backups in this way, you must remember that the *entire* Macintosh file will be backed up when any new information is written, so be careful where you put where you put things.

Therefore, if you have a hard disk and have some available disk space, you can use it to keep (part of) the distributed diskettes on line. This section describes the steps to set up MACMINIX on such a system.

5.7.1. Step 1: Decide How Much of Your Disk Space to Devote to MINIX

The first decision you must make is how much of your disk you want to give over to MACMINIX. It is really not all that crucial that you be right the first time, since you can reclaim file space for the Macintosh operating system by using the finder to remove one or more of the files that correspond to your MACMINIX ''partitions.'' (Of course, you also lose the information on that ''partition'').

You can also create a new ''partition'' at any time (assuming you have the free disk space), make a MACMINIX file system on it, and then mount it for use by MACMINIX (see the *maccreate*, *hdopen*, *mkfs*, *mount*, and *hdclose* manual pages). Keep in mind, however, that as distributed, MACMINIX allows you to mount a maximum of five such partitions simultaneously.

5.7.2. Step 2: Decide How to Logically Partition Your Disk Space

Once you have decided how much disk space you want to use, you must decide how to split the space into logical disk partitions. This is entirely up to you, but you should probably create at least one small partition to hold the root file system that is copied to the RAM disk at boot time.

Remember that how you logically partition your MACMINIX disk, and what you put on each partition, has potentially great impact on backing up your disk if you plan on doing so with ordinary Macintosh backup software. Also remember that as distributed, MACMINIX will allow a maximum of five of these partitions to be mounted simultaneously.

5.7.3. Step 3: Build a Macintosh File for Each Partition

For each partition that you want, you must create the Macintosh file that will correspond to that partition. If, for example, you want a MACMINIX partition that is 225 blocks and your hard disk is named ‘‘harddisk’’, boot MACMINIX and type:

```
maccreate 225 harddisk:file
```

This will set aside a Macintosh file of 225 blocks (assuming you have 225 free blocks) that can be used a MACMINIX disk partition. When you are running the finder with the normal Macintosh operating system, these 225 blocks will belong to the Macintosh file called *harddisk:file* and will have a MACMINIX file system icon on the desktop.

Follow this procedure for each logical MACMINIX partition you wish to create, changing the second and third parameters to *maccreate* as appropriate. For example, let us assume that you are going to want three partitions: a root partition of 225 blocks that will contain the RAM disk image, a */usr* partition of 10000 blocks that will be mounted on */dev/hd1*, and a */tmp* partition of 1000 blocks that will be mounted on */dev/hd2*. Furthermore, the Macintosh files are to be called ROOT, USR, and TMP respectively and are to go into an existing folder called ‘‘MINIX’’ on your hard disk. To do this, you would type:

```
maccreate 225  harddisk:MINIX:ROOT
maccreate 10000 harddisk:MINIX:USR
maccreate 1000 harddisk:MINIX:TMP
```

5.7.4. Step 4: Make a MINIX File System on Each MINIX Partition

Now that your MACMINIX disk is partitioned, you must put a file system on each.

The first thing to do is to inform MACMINIX that it should set up a correspondence between a given Macintosh file created with *maccreate* and a MACMINIX hard disk device (*/dev/hd?*). This is done with *hdopen*. Then *mkfs* is used to create the file system. So, assuming *maccreate* was used to create a 225 block Macintosh file called "harddisk:MINIX:ROOT" (as described in the previous example), type:

```
hdopen  harddisk:MINIX:ROOT  /dev/hd4
mkfs  /dev/hd4  225
```

Continuing with our example, you would type:

```
hdopen harddisk:MINIX:USR /dev/hd1
mkfs /dev/hd1 10000
hdopen harddisk:MINIX:TMP /dev/hd2
mkfs /dev/hd2 1000
```

You can verify that the file systems have been made by typing:

```
df /dev/hd1
df /dev/hd2
df /dev/hd4
```

which will report on the i-nodes and blocks present on each file system. The total number of blocks should agree with the number you used in the *mkfs* command.

5.7.5. Step 5: Initialize the Root File System

When MINIX boots, it needs a root file system. This file system is the RAM disk (*/dev/ram*) when one is being used, or hard disk partition 0 (*/dev/hd0*) if not. Either way, certain directories and special files must be created on the partition. In the discussion below, we will assume that a RAM disk is being used, and that the new root file system has been created with *maccreate*, and has been associated with */dev/hd4* with the *hdopen* command (just as was done in the last step).

The root file system normally has certain standard directories in it, to be described later. One of these, */dev*, contains all the character and block special files. To create the directories and special files, change to the root directory and use the setup_root command:

```
/etc/setup_root /dev/hd4 ram hd1 hd2 hd3 hd4
```

where *ram* is the size of the RAM disk in blocks (1K), and the next four numbers are the sizes of the four hard disk partitions, also in blocks. You must be logged in as root to run */etc/setup_root*. You must also specify all four partition sizes. If a partition has size zero, use 0.

In our example, the Macintosh file ROOT (225 blocks) is to be the ram disk image, USR (10000 blocks) is to be */dev/hd1*, and TMP (1000 blocks) is to be */dev/hd2*, so we would type:

```
/etc/setup_root /dev/hd4 225 10000 1000 0 0
```

At this point, the new hard disk root will contain the same files as the root file system diskette. To try it out, type *sync*, select ‘‘Quit’’ from the ‘‘File’’ menu, copy the boot application into the MINIX folder on the hard disk, eject the floppy, and launch the boot application from the hard disk. Be certain that the Macintosh file that corresponds to your root file system is named ROOT because this is what the application expects.

5.7.6. Step 6: Initialize */usr*

The next step is creating all the directories. A shell script called */etc/setup_usr* has been provided to do most of the work. It mounts the main hard disk partition and creates a large number of directories. Next, it copies files from the root file system and from diskettes to the */usr* tree on the hard disk. When it is finished, it asks for more diskettes to be inserted so it can copy files from them to the hard disk. Just follow the instructions that appear on the screen until the ''Installation completed'' message appears. To perform the installation be sure you are logged in as *root*.

For instance, with our example, the Macintosh file USR is to be the */usr* partition. To set it up, type:

```
/etc/hdopen harddisk:MINIX:USR /dev/hd1
/etc/setup_usr /dev/hd1
```

and follow the instructions displayed by the *setup_usr* script.

Loading all the diskettes requires 9 megabytes of disk space for the */usr* partition. You can have a smaller partition if you install only the binaries (4MB) onto the hard disk. In order to do so you should change the value of STOP on the third line of the */etc/setup_usr* script to 4, before issuing the commands above.

When this shell script finishes, the entire MINIX file system will be installed on the hard disk. Most of the files on the distribution diskettes are compressed files (with suffix *.Z*) or compressed archives (with suffix *.a.Z*). If, for some reason, installation fails part way through, you may be left with some *.a.Z*, *.a* or *.Z* files on the disk. See the ''UNPACKING THE SOURCES'' section below for information on how to deal with these files.

5.7.7. Step 7: Modifying /etc/rc

Now that the hard disk partitions have been created and initialized, they need to be mounted whenever MACMINIX is booted. This is done by making a small change in */etc/rc* found on the root file system.

First mount the new root file system, which in our example is ''harddisk:MINIX:ROOT,'' by typing:

```
/etc/hdopen harddisk:MINIX:ROOT /dev/hd4
/etc/mount /dev/hd4 /user
```

Use an editor (e.g. *mined*) to change the lines of the file */user/etc/rc* that read:

```
/etc/eject

# Mount the floppy disk
/bin/getlf ″Please insert /usr diskette in drive 0. Then hit RETURN.″
/etc/hdopen 01.USR1:USR1 /dev/fd0
/etc/mount /dev/fd0 /usr
```

with other text. For our example, a suitable replacement text would be the following:

```
/etc/hdopen harddisk:MINIX:USR /dev/hd1
/etc/mount /dev/hd1 /usr
/etc/hdopen harddisk:MINIX:TMP /dev/hd2
/etc/mount /dev/hd2 /tmp
```

Inserting diskette 01.USR1 will no longer be necessary at boot time.

5.7.8. Step 8: Removing the RAM disk

This step is optional. Since you have a hard disk, you may no longer want to have a RAM disk. You can remove the RAM disk by selecting the ''Configuration'' menu item in the ''File'' menu and deselecting the ''Use RAM disk'' check box. If you do this, you must modify */etc/rc* (as described in the last step) and remove the line that reads:

```
/etc/hdclose /dev/fd0
```

The next time you start MACMINIX, the RAM disk will not be used and the root file system will be mounted on */dev/hd0*.

5.8. UNPACKING THE SOURCES

All MACMINIX sources, except the sources for the compiler and the editor, can be found on the SRC disks. These disks are normal MACMINIX file systems, which you can mount using the *hdopen* and *mount* commands: For example, you can mount the 04.SRC1 inserting it in the drive and typing:

```
/etc/hdopen 04.SRC1:SRC1 /dev/fd1
/etc/mount /dev/fd1 /user
```

The files on the distribution diskettes are compressed archives (with suffix *.a.Z*). If you want to extract the sources from a file *file.a.Z* you should first copy this file to either an empty floppy, or to the ram disk, if the latter is large enough. Your copy of *file.a.Z* can be decompressed using:

```
compress –d file.a.Z
```

After decompressing you can remove your copy of *file.a.Z*. Now you can extract the files from the archive with the *ar* command, for example:

```
ar x file.a
```

At this point all files from the archive are extracted, and the file *file.a* can be removed.

5.9. THE MENUS

There are five menus available to MACMINIX user. They are generally self-explanatory, but this section gives a brief description of each.

5.9.1. The Apple Menu

The Apple menu is similar to every other apple menu you have seen. Selecting the first item on the menu will bring up a simple dialog box, describing MACMINIX, while the rest of the items are for your desk accessories.

5.9.2. The Edit Menu

The Edit menu exists for the benefit of those desk accessories that can use it. None of the menu items are used by MACMINIX.

5.9.3. The File Menu

The File menu is used primarily to quit from MACMINIX or to configure various aspects of MACMINIX operation. Remember to sync the disks before quitting. A detailed explanation of your configuration options is given below.

5.9.4. The Windows Menu

The Windows menu is used to manipulate your windows. For example, there are menu items to enable you to rotate the windows, reopen them once you have closed them, or selecting individual windows to bring to the front.

5.9.5. The Debug Menu

Once you have MACMINIX running, some of the internal MINIX tables can be inspected by selecting items from this menu.

5.10. SETTING CONFIGURATION OPTIONS

Selecting the ‘‘Configuration’’ menu item in the "File" menu will bring up a dialog box that allows you to set various operating parameters of MACMINIX. This section describes each of your options.

5.10.1. Heap Space

The dialog item entitled ‘‘Heap Space’’ allows you to specify how much of the application heap should be left by MACMINIX to support normal Macintosh operation, such as desk accessories and dialog boxes. The smaller you make this number, the more memory you will have available for MACMINIX processes. On the other hand, if this number is too small, the Macintosh operating system may run out of heap space, in which case the machine will crash. As distributed, this number is fairly generous so that it has the best chance of working with your configuration. You may want to experiment to see what is best for you. If you plan on using multifinder with MACMINIX, you can get away with making this number somewhat smaller, since desk accessories are not generally loaded into the currently running application’s heap under multifinder.

5.10.2. Keyboard Mappings

There is also a check box labeled ‘‘Use Builtin Keyboard Mappings.’’ As noted earlier, MACMINIX uses the Macintosh ROMs to set up the initial virtual keycode to ASCII mapping. If you prefer, however, you can select this check box and MACMINIX will use the mapping that has been compiled into the kernel. You may want to try this if you experience problems with your keyboard. As distributed, the US keyboard mapping has been compiled into the kernel.

5.10.3. The ROOT Partition

The final option you may set in this dialog box have to do with what Macintosh file is initially mapped to the MACMINIX fd0 disk partition. This is the partition used to initially read the root file system. As distributed, this is set to ‘‘00.BOOT:ROOT’’ meaning that it will attempt to use the Macintosh file called ROOT on the volume named 00.BOOT. Clicking the mouse button on top of the box that displays the name will bring up a standard file dialog box, and you can select a new Macintosh file to use as the root partition.

5.10.4. Effecting The Changes

Once you have made the desired changes to he configuration, the new configuration take effect the next time you boot MACMINIX. The ‘‘Cancel’’ button will cause any changes you made to be ignored.

You may also configure MACMINIX previous to booting by holding down the mouse button as you launch the boot application. If you do this, the configuration

dialog box appears immediately, and you can set the various items as above. In this case, the new configuration is used immediately.

5.11. MACINTOSH SYSTEM CALLS

Supplied with MACMINIX is a partial interface with the Macintosh ROMs. You can find the include files in */usr/include/mac*, and the interface routines in */usr/lib/mac*. The library routines were built automatically from a program that uses the include file prototypes as a guide. A complete interface will be available sometime in the future.

A MACMINIX process may make calls to the ROMs, but please keep in mind that MACMINIX will not preempt a MACMINIX process when it has made a ROM call, since the ROMs are non-reentrant, and preempting may cause major problems.

5.12. RUNNING MACMINIX WITH MULTIFINDER

Before multifinder, there was not much of a distinction between the currently running application and the operating system. Since only one application could run at a time, the Macintosh operating system could be viewed simply as a set of support routines for the application. With the introduction of multifinder, this view changed somewhat, since now a Macintosh could have several applications in memory at any one time. The Macintosh operating system would transparently switch between them, although it could only do so at times when the application agreed to ‘‘give up’’ the processor.

MACMINIX will work with multifinder, giving up the processor at various times so that your other applications may run. In order to give MACMINIX a larger multifinder memory partition, set the memory size of the boot application the same way you do for any other application (see your owner’s guide for a more complete description). There is one thing to remember here however and that is that MACMINIX does not run *at all* while another Macintosh application is running, so you may find that you have inconsistent results when running MACMINIX programs if they are time dependent.

5.13. TROUBLESHOOTING

As a user of MACMINIX you may be confronted with some of the error messages the system can produce. The following sections gives some guidelines on you how to react.

5.13.1. Exhausted Heap Space

If you experience some unexplainable crashes, especially when you do simple things like selecting a menu, the Macintosh operating system may be running out of usable heap space. You can increase this with the ‘‘Configuration’’ menu item, described above. Since you might be having a problem getting running in the first place, you can bring up the configuration dialog box before anything else happens if you hold down the mouse button when you initially launch MACMINIX.

5.13.2. System Software Incompatibilities

If you are running an old version of the Macintosh system software, you may want to bring it up to date in order to minimize incompatibility problems.

5.13.3. Init and Screen Saver Incompatibilities

If you experience unexplained crashes and are using some *inits* and/or *screen-savers* on your system, you may wish to temporarily disable or remove them to see if it solves your problem. One or more of them may be incompatible with MACMINIX.