Chapter 2

Obtaining and Installing Linux

In this chapter, we'll describe how to obtain the Linux software, in the form of one of the various pre-packaged distributions, and how to install the distribution that you choose.

As we have mentioned, there is no single "official" distribution of the Linux software; there are, in fact, many distributions, each of which serves a particular purpose and set of goals. These distributions are available via anonymous FTP from the Internet, on BBS systems worldwide, and via mail on diskette, tape, and CD-ROM.

Here, we present only a general overview of the installation process. Each distribution has its own specific installation instructions, but armed with the concepts presented here you should be able to feel your way through any installation. Appendix A lists sources of information for installation instructions and other help, if you're at a total loss.

2.1 Distributions of Linux

Because Linux is free software, no single organization or entity is responsible for releasing and distributing the software. Therefore, almost anyone is free to put together and distribute the Linux software, as long as the restrictions in the GPL are observed. The upshot of this is that there are many distributions of Linux, available via anonymous FTP or via mail order.

You are now faced with the task of deciding upon a particular distribution of Linux which suits your needs. Not all distributions are alike. Many of them come with just about all of the software you'd need to run a complete system—and then some. Other Linux distributions are "small" distributions intended for users without copious amounts of diskspace. Many distributions contain only the core Linux software, and you are expected to install larger software packages, such as the X Window System, yourself. (In Chapter 4 we'll show you how.)

In Appendix B, a brief list of Linux distributions is given. You should be able to contact the maintainers of each distribution for more information, should you need it. This is only a cursory list of Linux releases; for a more complete list, including information on other services, see the Linux Distribution HOWTO (information on which is in Appendix A).

How can you decide among all of these distributions? If you have access to USENET news, or another computer conferencing system, you might want to ask there for personal opinions from people who have installed Linux. Even better, if you know someone who has installed Linux, ask them for help and advice. There are many factors to consider when choosing a distribution, however, everyone's needs and opinions are different. In actuality, most of the popular Linux distributions contain roughly the same set of software, so the distribution that you select is more or less arbitrary.

2.1.1 Getting Linux from the Internet

If you have access to the Internet, the easiest way to obtain Linux is via anonymous FTP.[1] Appendix C lists a number of FTP archive sites which carry Linux software. One of these is `sunsite.unc.edu`, and the various Linux distributions can be found in the directory

```
/pub/Linux/distributions
```

there.

Many distributions are released via anonymous FTP as a set of disk images. That is, the distribution consists of a set of files, and each file contains the binary image of a floppy. In order to copy the contents of the image file onto the floppy, you can use the `RAWRITE.EXE` program under MS-DOS. This program copies, block-for-block, the contents of a file to a floppy, without regard for disk format.[2]

`RAWRITE.EXE` is available on the various Linux FTP sites, including `sunsite.unc.edu` in the directory

```
/pub/Linux/system/Install/rawwrite
```

Therefore, in many cases, you simply download the set of diskette images, and use `RAWRITE.EXE` with each image in turn to create a set of diskettes. You boot from the so-called "boot diskette" and you're ready to roll. The software is usually installed directly from the floppies, although some distributions allow you to install from an MS-DOS partition on your hard drive. Some distributions allow you to install over a TCP/IP network. The documentation for each distribution should describe these installation methods if they are available.

Other Linux distributions are installed from a set of MS-DOS format floppies. For example, the SLS distribution of Linux requires only one of the image—the "`a1`" diskette—to be written to diskette using `RAWRITE.EXE`. The rest of the files are copied to MS-DOS format diskettes labelled `a2`, `a3`, and so forth, using the MS-DOS `COPY` command. The system installs the software directly from the MS-DOS floppies. This saves you the trouble of having to use `RAWRITE.EXE` for many image files, although it requires you to have access to an MS-DOS system to create the diskettes.

[1] If you do not have direct Internet access, you can obtain Linux via the `ftpmail` service, provided that you have the ability to exchange e-mail with the Internet. See Appendix C for details.

[2] If you have access to a UNIX workstation with a floppy drive, you can also use the `dd` command to copy the file image directly to the floppy. A command such as "`dd of=/dev/rfd0 if=foo bs=16k`" will "raw write" the contents of the file `foo` to the floppy device on a Sun workstation. Consult your local UNIX gurus for more information on your system's floppy devices and the use of `dd`.

Each distribution of Linux available via anonymous FTP should include a `README` file describing how to download and prepare the diskettes for installation. Be sure to read all of the available documentation for the release that you are using—this book is only meant to give you the general idea.

When downloading the Linux software, be sure to use *binary* mode for all file transfers (with most FTP clients, the command "`binary`" enables this mode).

2.1.2 Getting Linux from other online sources

If you have access to another computer network such as CompuServe or Prodigy, there may be a means to download the Linux software from these sources. In addition, many bulletin board (BBS) systems carry Linux software. A list of Linux BBS sites is given in Appendix D. Not all Linux distributions are available from these computer networks, however—many of them, especially the various CD-ROM distributions, are only available via mail order.

2.1.3 Getting Linux via mail order

If you don't have Internet or BBS access, many Linux distributions are available via mail order on diskette, tape, or CD-ROM. Appendix B lists a number of these distributors. Many of them accept credit cards as well as international orders, so if you're not in the United States or Canada you still should be able to obtain Linux in this way.

Linux is free software, although distributors are allowed by the GPL to charge a fee for it. Therefore, ordering Linux via mail order might cost you between US$30 and US$150, depending on the distribution. However, if you know someone who has already purchased or downloaded a release of Linux, you are free to borrow or copy their software for your own use. Linux distributors are not allowed to restrict the license or redistribution of the software in any way. If you are thinking about installing an entire lab of machines with Linux, for example, you only need to purchase a single copy of one of the distributions, which can be used to install all of the machines.

2.2 Preparing to Install Linux

After you have obtained a distribution of Linux, you're ready to prepare your system for installation. This takes a certain degree of planning, especially if you're already running other operating systems. In the following sections we'll describe how to plan for the Linux installation.

2.2.1 Installation overview

While each release of Linux is different, in general the method used to install the software is as follows:

1. **Repartition your hard drive(s).** If you have other operating systems already installed, you will need to *repartition* the drives in order to allocate space for Linux. This is discussed in Section 2.2.4, below.
2. **Boot the Linux installation media.** Each distribution of Linux has some kind of installation media—usually a "boot floppy"—which is used to install the software. Booting this media will either present you with some kind of installation program, which will step you through the Linux installation, or allow you to install the software by hand.
3. **Create Linux partitions.** After repartitioning to allocate space for Linux, you create Linux partitions on that empty space. This is accomplished with the Linux `fdisk` program, covered in Section 2.3.3.
4. **Create filesystems and swap space.** At this point, you will create one or more *filesystems*, used to store files, on the newly-created partitions. In addition, if you plan to use swap space, you will create the swap space on one of your Linux partitions. This is covered in Sections 2.3.4 and 2.3.5.
5. **Install the software on the new filesystems.** Finally, you will install the Linux software on your newly-created filesystems. After this, it's smooth sailing—if all goes well. This is covered in Section 2.3.6. Later, in Section 2.5, we describe what to do if anything goes wrong.

Many distributions of Linux provide an installation program which will step you through the installation process, and automate one or more of the above steps for you. Keep in mind throughout this chapter that any number of the above steps may be automated for you, depending on the distribution.

Important hint: While preparing to install Linux, the best advice that we can give is to *take notes* during the entire procedure. Write down everything that you do, everything that you type, and everything that you see that might be out of the ordinary. The idea here is simple: if (or when!) you run into trouble, you want to be able to retrace your steps and find out what went wrong. Installing Linux isn't difficult, but there are many details to remember. You want to have a record of all of these details so that you can experiment with other methods if something goes wrong. Also, keeping a notebook of your Linux installation experience is useful when you want to ask other people for help, for example, when posting a message to one of the Linux-related USENET groups. Your notebook is also something that you'll want to show to your grandchildren someday.[3]

2.2.2 Repartitioning concepts

In general, hard drives are divided into *partitions*, where a single partition is devoted to a single operating system. For example, on one hard drive, you may have several separate partitions—one devoted to, say, MS-DOS, another to OS/2, and another to Linux.

[3] The author shamefully admits that he kept a notebook of all of his tribulations with Linux for the first few months of working with the system. It is now gathering dust on his bookshelf.

If you already have other software installed on your system, you may need to resize those partitions in order to free up space for Linux. You will then create one or more Linux partitions on the resulting free space for storing the Linux software and swap space. We call this process *repartitioning*.

Many MS-DOS systems utilize a single partition inhabiting the entire drive. To MS-DOS, this partition is known as `C:`. If you have more than one partition, MS-DOS names them `D:`, `E:`, and so on. In a way, each partition acts like a separate hard drive.

On the first sector of the disk is a **master boot record** along with a **partition table**. The boot record (as the name implies) is used to boot the system. The partition table contains information about the locations and sizes of your partitions.

There are three kinds of partitions: **primary**, **extended**, and **logical**. Of these, primary partitions are used most often. However, because of a limit in the size of the partition table, you can only have four primary partitions on any given drive.

The way around this four-partition limit is to use an extended partition. An extended partition doesn't hold any data by itself; instead, it acts as a "container" for logical partitions. Therefore, you could create one extended partition, covering the entire drive, and within it create many logical partitions. However, you may have only one extended partition per drive.

2.2.3 Linux partition requirements

Before we explain how to repartition your drives, you need to have an idea of how much space you will be allocating for Linux. We will be discussing how to create these partitions later, in Section 2.3.3.

On UNIX systems, files are stored on a **filesystem**, which is essentially a section of the hard drive (or other medium, such as CD-ROM or diskette) formatted to hold files. Each filesystem is associated with a specific part of the directory tree; for example, on many systems, there is a filesystem for all of the files in the directory `/usr`, another for `/tmp`, and so on. The **root filesystem** is the primary filesystem, which corresponds to the topmost directory, `/`.

Under Linux, each filesystem lives on a separate partition on the hard drive. For instance, if you have a filesystem for `/` and another for `/usr`, you will need two partitions to hold the two filesystems.

Before you install Linux, you will need to prepare filesystems for storing the Linux software. You must have at least one filesystem (the root filesystem), and therefore one partition, allocated to Linux. Many Linux users opt to store all of their files on the root filesystem, which is in most cases easier to manage than several filesystems and partitions.

However, you may create multiple filesystems for Linux if you wish—for example, you may want to use separate filesystems for `/usr` and `/home`. Those readers with UNIX system administration experience will know how to use multiple filesystems creatively. In Chapter 4 we discuss the use of multiple partitions and filesystems.

Why use more than one filesystem? The most commonly stated reason is safety; if, for some reason, one of your filesystems is damaged, the others will (usually) be unharmed. On the other hand, if you store all of your files on the root filesystem, and for some reason the filesystem is damaged, then you may lose all of your files in one fell swoop. This is, however, rather uncommon;

if you backup the system regularly you should be quite safe.[4]

Another reason to use multiple filesystems is to divvy up storage between multiple hard drives. If you have, say, 40 megabytes free on one hard drive, and 50 megabytes free on another, you might want to create a 40-megabyte root filesystem on the first drive and a 50-megabyte `/usr` filesystem on the other. Currently it is not possible for a single filesystem to span multiple drives; if your free hard drive storage is fragmented between drives you will need to use multiple filesystems to utilize it all.

In summary, Linux requires at least one partition, for the root filesystem. If you wish to create multiple filesystems, you will need a separate partition for each additional filesystem. Some distributions of Linux automatically create partitions and filesystems for you, so you may not need to worry about these issues at all.

Another issue to consider when planning your partitions is swap space. If you wish to use swap space with Linux, you have two options. The first is to use a *swap file* which exists on one of your Linux filesystems. You will create the swap file for use as virtual RAM after you install the software. The second option is to create a *swap partition*, an individual partition to be used only as swap space. Most people use a swap partition instead of a swap file.

A single swap file or partition may be up to 16 megabytes in size. If you wish to use more than 16 megabytes of swap, you can create multiple swap partitions or files—up to eight in all. For example, if you need 32 megabytes of swap, you can create two 16-megabyte swap partitions.

Setting up a swap partition is covered in Section 2.3.4, and setting up a swap file in Chapter 4.

Therefore, in general, you will create at least two partitions for Linux: one for use as the root filesystem, and the other for use as swap space. There are, of course, many variations on the above, but this is the minimal setup. You are not required to use swap space with Linux, but if you have less than 16 megabytes of physical RAM it is strongly suggested that you do.

Of course, you need to be aware of how much *space* these partitions will require. The size of your Linux filesystems (containing the software itself) depends greatly on how much software you're installing and what distribution of Linux you are using. Hopefully, the documentation that came with your distribution will give you an approximation of the space requirements. A small Linux system can use 20 megabytes or less; a larger system anywhere from 80 to 100 megabytes, or more. Keep in mind that in addition to the space required by the software itself, you need to allocate extra space for user directories, room for future expansion, and so forth.

The size of your swap partition (should you elect to use one) depends on how much virtual RAM you require. A rule of thumb is to use a swap partition that is twice the space of your physical RAM; for example, if you have 4 megabytes of physical RAM, an 8-megabyte swap partition should suffice. Of course, this is mere speculation—the actual amount of swap space that you require depends on the software which you will be running. If you have a great deal of physical RAM (say, sixteen megabytes or more), you may not wish to use swap space at all.

Important note: Many SCSI adaptors are incapable of booting software from partitions using cylinders numbered over 1024. Therefore, when setting aside space for Linux, keep in mind that you

[4] The author uses a single 200-megabyte filesystem for all of his Linux files, and hasn't had any problems (so far).

may not want to use a partition in the >1024-cylinder range for your Linux root filesystem. Linux can still *use* partitions with cylinders numbered over 1024, however, you may not be able to *boot* Linux from such a partition. This advice may seem premature, but it is important to know while planning your drive layout.

If you absolutely must use a partition with cylinders numbered over 1024 for your Linux root filesystem, you can always boot Linux from floppy. This is not so bad, actually—it only takes a few seconds longer to boot than from the hard drive. At any rate, it's always an option.

2.2.4 Repartitioning your drives

In this section, we'll describe how to resize your current partitions (if any) to make space for Linux. If you are installing Linux on a "clean" hard drive, you can skip this section and proceed to Section 2.3, below.

The usual way to resize an existing partition is to delete it (thus destroying all of the data on that partition) and recreate it. Before repartitioning your drives, *backup your system.* After resizing the partitions, you can reinstall your original software from the backup. However, there are several programs available for MS-DOS which are able to resize partitions nondestructively. One of these is known as "`FIPS`", and can be found on many Linux FTP sites.

Also, keep in mind that because you'll be shrinking your original partitions, you may not have space to reinstall everything. In this case, you need to delete enough unwanted software to allow the rest to fit on the smaller partitions.

The program used to repartition is known as `fdisk`. Each operating system has its own analogue of this program; for example, under MS-DOS, it is invoked with the `FDISK` command. You should consult your documentation for whatever operating systems you are currently running for information on repartitioning. Here, we'll discuss how to resize partitions for MS-DOS using `FDISK`, but this information should be easily extrapolated to other operating systems.

Please consult the documentation for your current operating systems before repartitioning your drive. This section is meant to be a general overview of the process; there are many subtleties that we do not cover here. You can lose all of the software on your system if you do not repartition the drive correctly.

A warning: Do not modify or create partitions for any other operating systems (including Linux) using `FDISK` under MS-DOS. You should only modify partitions for a particular operating system with the version of `fdisk` included with that operating system; for example, you will create Linux partitions using a version of `fdisk` for Linux. Later, in Section 2.3.3, we describe how to create Linux partitions, but for now we are concerned with resizing your current ones.

Let's say that you have a single hard drive on your system, currently devoted entirely to MS-DOS. Hence, your drive consists of a single MS-DOS partition, commonly known as "`C:`". Because this repartitioning method will destroy the data on that partition, you need to create a bootable MS-DOS "system disk" which contains everything necessary to run `FDISK` and restore the software from backup after the repartitioning is complete.

In many cases, you can use the MS-DOS installation disks for this purpose. However, if you need to create your own system disk, format a floppy with the command

```
FORMAT /s A:
```

Copy onto this floppy all of the necessary MS-DOS utilities (usually most of the software in the directory `\DOS` on your drive), as well as the programs `FORMAT.COM` and `FDISK.EXE`. You should now be able to boot this floppy, and run the command

```
FDISK C:
```

to start up `FDISK`.

Use of `FDISK` should be self-explanatory, but consult the MS-DOS documentation for details. When you start `FDISK`, use the menu option to display the partition table, and *write down* the information displayed there. It is important to keep a record of your original setup in case you want to back out of the Linux installation.

To delete an existing partition, choose the `FDISK` menu option "`Delete an MS-DOS Partition or Logical DOS Drive`". Specify the type of partition that you wish to delete (primary, extended, or logical) and the number of the partition. Verify all of the warnings. Poof!

To create a new (smaller) partition for MS-DOS, just choose the `FDISK` option "`Create an MS-DOS Partition or Logical DOS Drive`". Specify the type of partition (primary, extended, or logical), and the size of the partition to create (specified in megabytes). `FDISK` should create the partition and you're ready to roll.

After you're done using `FDISK`, you should exit the program and reformat any new partitions. For example, if you resized the first DOS partition on your drive (`C:`) you should run the command

```
FORMAT /s C:
```

You may now reinstall your original software from backup.

2.3 Installing the Linux software

After you have resized your existing partitions to make space for Linux, you are ready to install the software. Here is a brief overview of the procedure:

- Boot the Linux installation media;
- Run `fdisk` under Linux to create Linux partitions;
- Run `mke2fs` and `mkswap` to create Linux filesystems and swap space;
- Install the Linux software;
- Finally, either install the `LILO` boot loader on your hard drive, or create a boot floppy in order to boot your new Linux system.

As we have said, one (or more) of these steps may be automated for you by the installation procedure, depending on the distribution of Linux which you are using. Please consult the documentation for your distribution for specific instructions.

2.3.1 Booting Linux

The first step is to boot the Linux installation media. This may be a floppy, tape, or CD-ROM, depending on the distribution.

In most cases, the installation media is a "boot floppy" which contains a small Linux system. Upon booting the floppy, you will be presented with an installation menu of some kind which will lead you through the steps of installing the software. On other distributions, you will be presented with a login prompt when booting this floppy. Here, you usually login as `root` or `install` to begin the installation process.

The documentation which came with your particular distribution will explain what is necessary to boot Linux from the installation media.

2.3.2 Drives and partitions under Linux

Many distributions require you to create Linux partitions by hand using the `fdisk` program. Others may automatically create partitions for you. Either way, you should know the following information about Linux partitions and device names.

Drives and partitions under Linux are given different names than their counterparts under other operating systems. Under MS-DOS, floppy drives are referred to as `A:` and `B:`, while hard drive partitions are named `C:`, `D:`, and so on. Under Linux, the naming convention is quite different.

Device drivers, found in the directory `/dev`, are used to communicate with devices on your system (such as hard drives, mice, and so on). For example, if you have a mouse on your system, you access it through the driver `/dev/mouse`. Floppy drives, hard drives, and individual partitions are all given individual device drivers of their own. Don't worry about the device driver interface for now; it is important only to understand how the various devices are named in order to use them.

Table 2.1 lists the names of these various device drivers.

A few notes about this table. Note that `/dev/fd0` corresponds to the first floppy drive (`A:` under MS-DOS) and `/dev/fd1` corresponds to the second floppy (`B:`).

Also, SCSI hard drives are named differently than other drives. IDE, MFM, and RLL drives are accessed through the devices `/dev/hda`, `/dev/hdb`, and so on. The individual partitions on the drive `/dev/hda` are `/dev/hda1`, `/dev/hda2`, and so on. However, SCSI drives are named `/dev/sda`, `/dev/sdb`, etc., with partition names such as `/dev/sda1` and `/dev/sda2`.

Here's an example. Let's say that you have a single IDE hard drive, with 3 primary partitions. The first two are set aside for MS-DOS, and the third is an extended partition which contains two logical partitions, both for use by Linux. The devices referring to these partitions would be:

Device	Name
First floppy (`A:`)	`/dev/fd0`
Second floppy (`B:`)	`/dev/fd1`
First hard drive (entire drive)	`/dev/hda`
First hard drive, primary partition 1	`/dev/hda1`
First hard drive, primary partition 2	`/dev/hda2`
First hard drive, primary partition 3	`/dev/hda3`
First hard drive, primary partition 4	`/dev/hda4`
First hard drive, logical partition 1	`/dev/hda5`
First hard drive, logical partition 2	`/dev/hda6`
⋮	
Second hard drive (entire drive)	`/dev/hdb`
Second hard drive, primary partition 1	`/dev/hdb1`
⋮	
First SCSI hard drive (entire drive)	`/dev/sda`
First SCSI hard drive, primary partition 1	`/dev/sda1`
⋮	
Second SCSI hard drive (entire drive)	`/dev/sdb`
Second SCSI hard drive, primary partition 1	`/dev/sdb1`
⋮	

Table 2.1: Linux partition names

First MS-DOS partition (`C:`)	`/dev/hda1`
Second MS-DOS partition (`D:`)	`/dev/hda2`
Extended partition	`/dev/hda3`
First Linux logical partition	`/dev/hda5`
Second Linux logical partition	`/dev/hda6`

Note that `/dev/hda4` is skipped; it corresponds to the fourth primary partition, which we don't have in this example. Logical partitions are named consecutively starting with `/dev/hda5`.

2.3.3 Creating Linux partitions

Now you are ready to create Linux partitions with the `fdisk` command. As described in Section 2.2.3, in general you will need to create at least one partition for the Linux software itself, and another partition for swap space.

After booting the installation media, run `fdisk` by typing

`fdisk` ⟨*drive*⟩

where ⟨*drive*⟩ is the Linux device name of the drive you plan to add partitions to (see Table 2.1). For instance, if you want to run `fdisk` on the first SCSI disk in your system, use the command fdisk /dev/sda. /dev/hda (the first IDE drive) is the default if you don't specify one.

If you are creating Linux partitions on more than one drive, run fdisk once for each drive.

```
# fdisk /dev/hda

Command (m for help):
```

Here fdisk is waiting for a command; you can type m to get a list of options.

```
Command (m for help):  m
Command action
a toggle a bootable flag
d delete a partition
l list known partition types
m print this menu
n add a new partition
p print the partition table
q quit without saving changes
t change a partition's system id
u change display/entry units
v verify the partition table
w write table to disk and exit
x extra functionality (experts only)

Command (m for help):
```

The n command is used to create a new partition. Most of the other options you won't need to worry about. To quit fdisk without saving any changes, use the q command. To quit fdisk and write the changes to the partition table to disk, use the w command.

The first thing you should do is display your current partition table and write the information down, for later reference. Use the p command.

```
Command (m for help):  p
Disk /dev/hda:  16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes

   Device Boot  Begin   Start     End  Blocks   Id  System
/dev/hda1   *       1       1     203   61693    6  DOS 16-bit >=32M


Command (m for help):
```

In this example, we have a single MS-DOS partition on /dev/hda1, which is 61693 blocks (about 60 megs). This partition starts at cylinder number 1, and ends on cylinder 203. We have a total of 683 cylinders in this disk; so there are 480 cylinders left to create Linux partitions on.

To create a new partition, use the n command. In this example, we'll create two primary partitions (/dev/hda2 and /dev/hda3) for Linux.

```
Command (m for help): n
Command action
e extended
p primary partition (1-4)
p
```

Here, `fdisk` is asking the type of the partition to create: extended or primary. In our example, we're creating only primary partitions, so we choose `p`.

```
Partition number (1-4):
```

`fdisk` will then ask for the number of the partition to create; since partition 1 is already used, our first Linux partition will be number 2.

```
Partition number (1-4): 2
First cylinder (204-683):
```

Now enter the starting cylinder number of the partition. Since cylinders 204 through 683 are unused, we'll use the first available one (numbered 204). There's no reason to leave empty space between partitions.

```
First cylinder (204-683): 204
Last cylinder or +size or +sizeM or +sizeK (204-683):
```

`fdisk` is asking for the size of the partition to create. We can either specify an ending cylinder number, or a size in bytes, kilobytes, or megabytes. Since we want our partition to be 80 megs in size, we specify `+80M`. When specifying a partition size in this way, `fdisk` will round the actual partition size to the nearest number of cylinders.

```
Last cylinder or +size or +sizeM or +sizeK (204-683): +80M
Warning: Linux cannot currently use 33090 sectors of this partition
```

If you see a warning message such as this, it can be ignored. `fdisk` prints the warning because it's an older program, and dates before the time that Linux partitions were allowed to be larger than 64 megabytes.

Now we're ready to create our second Linux partition. For sake of demonstration, we'll create it with a size of 10 megabytes.

```
Command (m for help): n
Command action
e extended
p primary partition (1-4)
p
Partition number (1-4): 3
First cylinder (474-683): 474
Last cylinder or +size or +sizeM or +sizeK (474-683): +10M
```

At last, we'll display the partition table. Again, write down all of this information—especially the block sizes of your new partitions. You'll need to know the sizes of the partitions when creating filesystems, later. Also, verify that none of your partitions overlap.

```
Command (m for help):  p

Disk /dev/hda:  16 heads, 38 sectors, 683 cylinders
Units = cylinders of 608 * 512 bytes

   Device Boot  Begin   Start     End  Blocks   Id  System
/dev/hda1   *       1       1     203   61693    6  DOS 16-bit >=32M
/dev/hda2         204     204     473   82080   81  Linux/MINIX
/dev/hda3         474     474     507   10336   81  Linux/MINIX
```

As you can see, `/dev/hda2` is now a partition of size 82080 blocks (which corresponds to about 80 megabytes), and `/dev/hda3` is 10336 blocks (about 10 megs).

Note that many distributions (such as Slackware) require you to use the `t` command in `fdisk` to change the type of the swap partition to "Linux swap", which is usually numbered 82. You can use the `L` command to print a list of known partition type codes, and then use `t` to set the type of the swap partition to that which corresponds to "Linux swap".

In this way, the installation software will be able to automatically find your swap partitions based on type. If the installation software doesn't seem to recognize your swap partition, you might want to re-run `fdisk` and use the `t` command on the partition in question.

In the example above, the remaining cylinders on the disk (numbered 508 to 683) are unused. You may wish to leave unused space on the disk, in case you wish to create additional partitions later.

Finally, we use the `w` command to write the changes to disk and exit `fdisk`.

```
Command (m for help):  w

#
```

Keep in mind that none of the changes you make while running `fdisk` will take effect until you give the `w` command, so you can toy with different configurations and save them when you're done. Also, if you want to quit `fdisk` at any time without saving the changes, use the `q` command. Remember that you shouldn't modify partitions for operating systems other than Linux with the Linux `fdisk` program.

Remember that you may not be able to boot Linux from a partition using cylinders numbered over 1024. Therefore, you should try to create your Linux root partition within the sub-1024 cylinder range. Again, if this is impossible, you can simply boot Linux from floppy.

Some Linux distributions require you to reboot the system after running `fdisk`. This is to allow the changes to the partition table to take effect before installing the software. Newer versions of `fdisk` automatically update the partition information in the kernel, so rebooting isn't necessary. To be on the safe side, after running `fdisk` you should reboot the installation media, as before, before proceeding.

2.3.4 Creating the swap space

If you are planning to use a swap partition for virtual RAM, you're ready to prepare it for use.[5] In Chapter 4 we will discuss the preparation of a swap file in case you don't want to use an individual partition.

Many distributions require you to create and activate swap space before installing the software. If you have a small amount of physical RAM, the installation procedure may not be successful unless you have some amount of swap space enabled.

The command used to prepare a swap partition is `mkswap`, and it takes the form

```
mkswap -c ⟨partition⟩ ⟨size⟩
```

where ⟨*partition*⟩ is the name of the swap partition, and ⟨*size*⟩ is the size of the partition, in blocks.[6] For example, if your swap partition is `/dev/hda3` and is 10336 blocks in size, use the command

```
# mkswap -c /dev/hda3 10336
```

The `-c` option tells `mkswap` to check for bad blocks on the partition when creating the swap space.

If you are using multiple swap partitions, you will need to execute the appropriate `mkswap` command for each partition.

After formatting the swap space, you need to enable it for use by the system. Usually, the system automatically enables swap space at boot time. However, because you have not yet installed the Linux software, you need to enable it by hand.

The command to enable swap space is `swapon`, and it takes the form

```
swapon ⟨partition⟩
```

In the example above, to enable the swap space on `/dev/hda3`, we use the command

```
# swapon /dev/hda3
```

2.3.5 Creating the filesystems

Before you can use your Linux partitions to store files, you must create **filesystems** on them. Creating a filesystem is analogous to formatting a partition under MS-DOS or other operating systems. We discussed filesystems briefly in Section 2.2.3.

There are several types of filesystems available for Linux. Each filesystem type has its own format and set of characteristics (such as filename length, maximum file size, and so on). Linux also supports several "third-party" filesystem types such as the MS-DOS filesystem.

[5] Again, some distributions of Linux will prepare the swap space automatically for you, or via an installation menu option.

[6] This is the size as reported by `fdisk`, using the `p` menu option. A block under Linux is 1024 bytes.

The most commonly used filesystem type is the **Second Extended Filesystem**, or *ext2fs*. The *ext2fs* is one of the most efficient and flexible filesystems; it allows filenames up to 256 characters and filesystem sizes of up to 4 terabytes. In Chapter 4, we'll discuss the various filesystem types available for Linux. Initially, however, we suggest that you use the *ext2fs* filesystem.

To create an *ext2fs* filesystem, use the command

`mke2fs -c` ⟨*partition*⟩ ⟨*size*⟩

where ⟨*partition*⟩ is the name of the partition, and ⟨*size*⟩ is the size of the partition in blocks. For example, to create a 82080-block filesystem on `/dev/hda2`, use the command

```
# mke2fs -c /dev/hda2 82080
```

If you're using multiple filesystems for Linux, you'll need to use the appropriate `mke2fs` command for each filesystem.

If you have encountered any problems at this point, see Section 2.5 at the end of this chapter.

2.3.6 Installing the software

Finally, you are ready to install the software on your system. Every distribution has a different mechanism for doing this. Many distributions have a self-contained program which will step you through the installation. On other distributions, you will have to **mount** your filesystems in a certain subdirectory (such as `/mnt`) and copy the software to them by hand. On CD-ROM distributions, you may be given the option to install a portion of the software on your hard drives, and leave most of the software on the CD-ROM.

Some distributions offer several different ways to install the software. For example, you may be able to install the software directly from an MS-DOS partition on your hard drive, instead of from floppies. Or, you may be able to install over a TCP/IP network via FTP or NFS. See your distribution's documentation for details.

As an example, the SLS distribution of Linux uses the `doinstall` command to install the software. It takes the form

`doinstall` ⟨*rootfs*⟩ ⟨*fs1*⟩ ⟨*mount-pt1*⟩ ⟨*fs2*⟩ ⟨*mount-pt2*⟩ ... ⟨*fsN*⟩ ⟨*mount-ptN*⟩

where ⟨*rootfs*⟩ is the name of your root filesystem, ⟨*fs1*⟩ and ⟨*mount-pt1*⟩ are the names of an additional filesystem and the mount point for that filesystem, and so on. For example, if you have a single filesystem for Linux on `/dev/hda2`, you would install the software with the command

```
# doinstall /dev/hda2
```

If you had an additional filesystem on `/dev/hda4` for `/usr`, you would instead use the command

```
# doinstall /dev/hda2 /dev/hda4 /usr
```

Again, the above commands are meant only as examples. The exact method used to install the Linux software differs greatly with each distribution. We're hoping that installing the Linux software should be self-explanatory, as it is with most distributions.

2.3.7 Creating the boot floppy or installing LILO

Every distribution provides some means of booting your new Linux system after you have installed the software. In many cases, the installation procedure will create a "boot floppy" which contains a Linux kernel configured to use your newly-created root filesystem. In order to boot Linux, you would boot from this floppy, and control would be transferred to your hard drive after booting. On other distributions, this "boot floppy" is the installation floppy itself.

Many distributions give you the option of installing **LILO** on your hard drive. LILO is a program that is installed on your drive's master boot record. It is able to boot a number of operating systems, including MS-DOS and Linux, and allows you to select at startup time which to boot.

In order for LILO to be installed successfully, it needs to know a good deal of information about your drive configuration—for example, which partitions contain which operating systems, how to boot each operating system, and so on. Many distributions, when installing LILO, attempt to "guess" at the appropriate parameters for your configuration. Although it's not often, the automated LILO installation provided by some distributions can fail, and leave your master boot record in shambles (although it's very doubtful that any damage to the actual data on your hard drive will take place). In particular, if you use OS/2's Boot Manager, you should *not* install LILO using the automated procedure—there are special instructions for using LILO with the Boot Manager, which will be covered later.

In many cases, it is best to use a boot floppy, until you have a chance to configure LILO yourself, by hand. If you're feeling exceptionally trustworthy, though, you can go ahead with the automated LILO installation if it is provided with your distribution.

In Chapter 4, we'll cover in detail how to configure and install LILO for your particular setup.

If everything goes well, then congratulations! You have just installed Linux on your system. Go have a Diet Coke or something—you deserve it.

In case you did run into any trouble, the next section will describe the most common sticking points for Linux installations, and how to get around them.

2.3.8 Additional installation procedures

Some distributions of Linux provide a number of additional installation procedures, allowing you to configure various software packages such as TCP/IP networking, the X Window System, and so on. If you are provided with these configuration options during installation, you may wish to read ahead in this book for more information on how to configure this software. Otherwise, you should put off these installation procedures until you have a complete understanding of how to configure the software.

It's up to you; if all else fails, just go with the flow and see what happens. It's very doubtful that anything that you do incorrectly now cannot be undone in the future. (Knock on wood.)

2.4 Postinstallation procedures

After you have completed installing the Linux software, there should be very little left to do before you can begin to use the system. In most cases, you should be able to reboot the system, login as `root`, and begin exploring the system. (Each distribution has a different method for doing this—follow the instructions given by the distribution.)

At this point it's a good idea to explain how to reboot and shutdown the system as you're using it. You should never reboot or shutdown your Linux system by pressing the reset switch or with the old "Vulcan Nerve Pinch"—that is, by pressing `ctrl-alt-del` in unison. You shouldn't simply switch off the power, either. As with most UNIX systems, Linux caches disk writes in memory. Therefore, if you suddenly reboot the system without shutting down "cleanly", you can corrupt the data on your drives, causing untold damage.

The easiest way to shut down the system is with the `shutdown` command. As an example, to shutdown and reboot the system immediately, use the following command as `root`:

```
# shutdown -r now
```

This will cleanly reboot your system. The man page for `shutdown` describes the other command-line arguments that are available.

Note, however, that many Linux distributions do not provide the `shutdown` command on the installation media. This means that the first time you reboot your system after installation, you may need to use the `ctrl-alt-del` combination after all. Thereafter, you should always use the `shutdown` command.

After you have a chance to explore and use the system, there are several configuration chores that you should undertake. The first is to create a user account for yourself (and, optionally, for any other users that might have access to the system). Creating user accounts is described in Section 4.4.

If you created more than one filesystem for Linux, or if you're using a swap partition, you may need to edit the file `/etc/fstab` in order for those filesystems to be available automatically after rebooting. (For example, if you're using a separate filesystem for `/usr`, and none of the files that should be in `/usr` appear to be present, you may simply need to mount that filesystem.) Section 4.8 describes this procedure.

2.5 Running Into Trouble

Almost everyone runs into some kind of snag or hangup when attempting to install Linux the first time. Most of the time, the problem is caused by a simple misunderstanding. Sometimes, however, it can be something more serious, such as an oversight by one of the developers, or a bug.

This section will describe some of the most common installation problems, and how to solve them. If your installation appears to be successful, but you received unexpected error messages during the installation, these are described here as well.

2.5.1 Problems with booting the installation media

When attempting to boot the installation media for the first time, you may encounter a number of problems. These are listed below. Note that the following problems are *not* related to booting your newly-installed Linux system. See Section 2.5.4 for information on these kinds of pitfalls.

- **Floppy or media error when attempting to boot.**

  The most popular cause for this kind of problem is a corrupt boot floppy. Either the floppy is physically damaged, in which case you should re-create the disk with a *brand new* floppy, or the data on the floppy is bad, in which case you should verify that you downloaded and transferred the data to the floppy correctly. In many cases, simply re-creating the boot floppy will solve your problems. Retrace your steps and try again.

  If you received your boot floppy from a mail order vendor or some other distributor, instead of downloading and creating it yourself, contact the distributor and ask for a new boot floppy—but only after verifying that this is indeed the problem.

- **System "hangs" during boot or after booting.**

  After the installation media boots, you will see a number of messages from the kernel itself, indicating which devices were detected and configured. After this, you will usually be presented with a login prompt, allowing you to proceed with installation (some distributions instead drop you right into an installation program of some kind). The system may appear to "hang" during several of these steps. During all of these steps, be patient; loading software from floppy is very slow. In many cases, the system has not hung at all, but is merely taking a long time. Verify that there is no drive or system activity for at least several minutes before assuming that the system is hung.

  1. After booting from the `LILO` prompt, the system must load the kernel image from floppy. This may take several seconds; you will know that things are going well if the floppy drive light is still on.
  2. While the kernel boots, SCSI devices must be probed for. If you do not have any SCSI devices installed, the system will "hang" for up to 15 seconds while the SCSI probe continues; this usually occurs after the line

     ```
     lp_init:  lp1 exists (0), using polling driver
     ```

     appears on your screen.
  3. After the kernel is finished booting, control is transferred to the system bootup files on the floppy. Finally, you will be presented with a login prompt, or be dropped into an installation program. If you are presented with a login prompt such as

     ```
     Linux login:
     ```

you should then login (usually as `root` or `install`—this varies with each distribution). After entering the username, the system may pause for 20 seconds or more while the installation program or shell is being loaded from floppy. Again, the floppy drive light should be on. Don't assume that the system is hung.

Any of the above items may be the source of your problem. However, it is possible that the system actually may "hang" while booting, which can be due to several causes. First of all, you may not have enough available RAM to boot the installation media. (See the following item for information on disabling the ramdisk to free up memory.)

The cause of many system hangs is hardware incompatibility. Section 1.8 in the last chapter presented an overview of supported hardware under Linux. Even if your hardware is supported, you may run into problems with incompatible hardware configurations which are causing the system to hang. See Section 2.5.2, below, for a discussion of hardware incompatibilities.

- **System reports out of memory errors while attempting to boot or install the software.**

  This item deals with the amount of available RAM that you have available. On systems with 4 megabytes of RAM or less, you may run into trouble booting the installation media or installing the software itself. This is because many distributions use a "ramdisk", which is a filesystem loaded directly into RAM, for operations while using the installation media. The entire image of the installation boot floppy, for example, may be loaded into a ramdisk, which may require more than a megabyte of RAM.

  The solution to this problem is to disable the ramdisk option when booting the install media. Each release has a different procedure for doing this; on the SLS release, for example, you type "`floppy`" at the LILO prompt when booting the `a1` disk. See your distribution's documentation for details.

  You may not see an "out of memory" error when attempting to boot or install the software; instead, the system may unexpectedly hang, or fail to boot. If your system hangs, and none of the explanations in the previous section seem to be the cause, try disabling the ramdisk.

  Keep in mind that Linux itself requires at least 2 megabytes of RAM to run at all; some distributions of Linux require 4 megabytes or more.

- **The system reports an error such as "`permission denied`" or "`file not found`" while booting.**

  This is an indication that your installation bootup media is corrupt. If you attempt to boot from the installation media (and you're sure that you're doing everything correctly), you should not see any errors such as this. Contact the distributor of your Linux software and find out about the problem, and perhaps obtain another copy of the boot media if necessary. If you downloaded the bootup disk yourself, try re-creating the bootup disk, and see if this solves your problem.

- **The system reports the error "`VFS: Unable to mount root`" when booting.**

  This error message means that the root filesystem (found on the boot media itself), could not be found. This means that either your boot media is corrupt in some way, or that you are not booting the system correctly.

For example, many CD-ROM distributions require that you have the CD-ROM in the drive when booting. Also be sure that the CD-ROM drive is on, and check for any activity. It's also possible that the system is not locating your CD-ROM drive at boot time; see Section 2.5.2 for more information.

If you're sure that you are booting the system correctly, then your bootup media may indeed be corrupt. This is a very uncommon problem, so try other solutions before attempting to use another boot floppy or tape.

2.5.2 Hardware problems

The most common form of problem when attempting to install or use Linux is an incompatibility with hardware. Even if all of your hardware is supported by Linux, a misconfiguration or hardware conflict can sometimes cause strange results—your devices may not be detected at boot time, or the system may hang.

It is important to isolate these hardware problems if you suspect that they may be the source of your trouble. In the following sections we will describe some common hardware problems and how to resolve them.

2.5.2.1 Isolating hardware problems

If you experience a problem that you believe to be hardware-related, the first thing that you should to do is attempt to isolate the problem. This means eliminating all possible variables and (usually) taking the system apart, piece-by-piece, until the offending piece of hardware is isolated.

This is not as frightening as it may sound. Basically, you should remove all nonessential hardware from your system, and then determine which device is actually causing the trouble—possibly by reinserting each device, one at a time. This means that you should remove all hardware other than the floppy and video controllers, and of course the keyboard. Even innocent-looking devices such as mouse controllers can wreak unknown havoc on your peace of mind unless you consider them nonessential.

For example, let's say that the system hangs during the Ethernet board detection sequence at boot time. You might hypothesize that there is a conflict or problem with the Ethernet board in your machine. The quick and easy way to find out is to pull the Ethernet board, and try booting again. If everything goes well, then you know that either (a) the Ethernet board is not supported by Linux (see Section 1.8 for a list of compatible boards), or (b) there is an address or IRQ conflict with the board.

"Address or IRQ conflict?" What on earth does that mean? All devices in your machine use an *IRQ*, or *interrupt request line*, to tell the system that they need something done on their behalf. You can think of the IRQ as a cord that the device tugs when it needs the system to take care of some pending request. If more than one device is tugging on the same cord, the kernel won't be able to detemine which device it needs to service. Instant mayhem.

Therefore, be sure that all of your installed devices are using unique IRQ lines. In general the IRQ for a device can be set by jumpers on the card; see the documentation for the particular device for details. Some devices do not require the use of an IRQ at all, but it is suggested that you configure them to use one if possible (the Seagate ST01 and ST02 SCSI controllers being good examples).

In some cases, the kernel provided on your installation media is configured to use a certain IRQ for certain devices. For example, on some distributions of Linux, the kernel is preconfigured to use IRQ 5 for the TMC-950 SCSI controller, the Mitsumi CD-ROM controller, and the bus mouse driver. If you want to use two or more of these devices, you'll need to first install Linux with only one of these devices enabled, then recompile the kernel in order to change the default IRQ for one of them. (See Chapter 4 for information on recompiling the kernel.)

Another area where hardware conflicts can arise is with DMA (direct memory access) channels, I/O addresses, and shared memory addresses. All of these terms describe mechanisms through which the system interfaces with hardware devices. Some Ethernet boards, for example, use a shared memory address as well as an IRQ to interface with the system. If any of these are in conflict with other devices, then the system may behave unexpectedly. You should be able to change the DMA channel, I/O or shared memory addresses for your various devices with jumper settings. (Unfortunately, some devices don't allow you to change these settings.)

The documentation for your various hardware devices should specify the IRQ, DMA channel, I/O address, or shared memory address that the devices use, and how to configure them. Again, the simple way to get around these problems is just to temporarily disable the conflicting devices until you have time to determine the cause of the problem.

Table 2.2 is a list of IRQ and DMA channels used by various "standard" devices found on most systems. Almost all systems will have these some of devices, so you should avoid setting the IRQ or DMA of other devices in conflict with these values.

Device	I/O address	IRQ	DMA
`ttyS0` (`COM1`)	3f8	4	n/a
`ttyS1` (`COM2`)	2f8	3	n/a
`ttyS2` (`COM3`)	3e8	4	n/a
`ttyS3` (`COM4`)	2e8	3	n/a
`lp0` (`LPT1`)	378 - 37f	7	n/a
`lp1` (`LPT2`)	278 - 27f	5	n/a
`fd0`, `fd1` (floppies 1 and 2)	3f0 - 3f7	6	2
`fd2`, `fd3` (floppies 3 and 4)	370 - 377	10	3

Table 2.2: Common device settings

2.5.2.2 Problems recognizing hard drive or controller

When Linux boots, you should see a series of messages on your screen such as:

```
Console:  colour EGA+ 80x25, 8 virtual consoles
Serial driver version 3.96 with no serial options enabled
tty00 at 0x03f8 (irq = 4) is a 16450
tty03 at 0x02e8 (irq = 3) is a 16550A
lp_init:  lp1 exists (0), using polling driver
...
```

Here, the kernel is detecting the various hardware devices present on your system. At some point, you should see the line

```
Partition check:
```

followed by a list of recognized partitions, for example:

```
Partition check:
hda:  hda1 hda2
hdb:  hdb1 hdb2 hdb3
```

If, for some reason, your drives or partitions are not recognized, then you will not be able to access them in any way.

There are several things that can cause this to happen:

- **Hard drive or controller not supported.** If you are using a hard drive controller (IDE, SCSI, or otherwise) that is not supported by Linux, the kernel will not recognize your partitions at boot time.
- **Drive or controller improperly configured.** Even if your controller is supported by Linux, it may not be configured correctly. (This is particularly a problem for SCSI controllers; most non-SCSI controllers should work fine without any additional configuration).

  Refer to the documentation for your hard drive and/or controller for information on solving these kinds of problems. In particular, many hard drives will need to have a jumper set if they are to be used as a "slave" drive (for example, as the second hard drive). The acid test for this kind of condition is to boot up MS-DOS, or some other operating system, known to work with your drive and controller. If you can access the drive and controller from another operating system, then it is not a problem with your hardware configuration.

  See Section 2.5.2.1, above, for information on resolving possible device conflicts, and Section 2.5.2.3, below, for information on configuring SCSI devices.
- **Controller properly configured, but not detected.** Some BIOS-less SCSI controllers require the user to specify information about the controller at boot time. Section 2.5.2.3, below, describes how to force hardware detection for these controllers.
- **Hard drive geometry not recognized.** Some systems, such as the IBM PS/ValuePoint, do not store hard drive geometry information in the CMOS memory, where Linux expects to find it. Also, certain SCSI controllers need to be told where to find drive geometry in order for Linux to recognize the layout of your drive.

Most distributions provide a bootup option to specify the drive geometry. In general, when booting the installation media, you can specify the drive geometry at the LILO boot prompt with a command such as:

`boot:` *linux hd=⟨cylinders⟩,⟨heads⟩,⟨sectors⟩*

where ⟨*cylinders*⟩, ⟨*heads*⟩, and ⟨*sectors*⟩ correspond to the number of cylinders, heads, and sectors per track for your hard drive.

After installing the Linux software, you will be able to install LILO, allowing you to boot from the hard drive. At that time, you can specify the drive geometry to the LILO installation procedure, making it unnecessary to enter the drive geometry each time you boot. See Chapter 4 for more about LILO.

2.5.2.3 Problems with SCSI controllers and devices

Presented here are some of the most common problems with SCSI controllers and devices such as CD-ROMs, hard drives, and tape drives. If you are having problems getting Linux to recognize your drive or controller, read on.

The Linux SCSI HOWTO (see Appendix A) contains much useful information on SCSI devices in addition to that listed here. SCSI can be particularly tricky to configure at times.

- **A SCSI device is detected at all possible ID's.** This is caused by strapping the device to the same address as the controller. You need to change the jumper settings so that the drive uses a different address from the controller itself.
- **Linux reports sense errors, even if the devices are known to be error-free.** This can be caused by bad cables, or by bad termination. If your SCSI bus is not terminated at both ends, you may have errors accessing SCSI devices. When in doubt, always check your cables.
- **SCSI devices report timeout errors.** This is usually caused by a conflict with IRQ, DMA, or device addresses. Also check that interrupts are enabled correctly on your controller.
- **SCSI controllers using BIOS are not detected.** Detection of controllers using BIOS will fail if the BIOS is disabled, or if your controller's "signature" is not recognized by the kernel. See the Linux SCSI HOWTO for more information about this.
- **Controllers using memory mapped I/O do not work.** This is caused when the memory-mapped I/O ports are incorrectly cached. Either mark the board's address space as uncacheable in the XCMOS settings, or disable cache altogether.
- **When partitioning, you get a warning that "cylinders > 1024", or you are unable to boot from a partition using cylinders numbered above 1023.** BIOS limits the number of cylinders to 1024, and any partition using cylinders numbered above this won't be accessible from the BIOS. As far as Linux is concerned, this affects only booting; once the system has booted you should be able to access the partition. Your options are to either boot Linux from a boot floppy, or boot from a partition using cylinders numbered below 1024. See Section 2.3.7 for information on creating a boot diskette or installing LILO.

- **CD-ROM drive or other removeable media devices are not recognized at boot time.** Try booting with a CD-ROM (or disk) in the drive. This is necessary for some devices.

If your SCSI controller is not recognized, you may need to force hardware detection at boot time. This is particularly important for BIOS-less SCSI controllers. Most distributions allow you to specify the controller IRQ and shared memory address when booting the installation media. For example, if you are using a TMC-8xx controller, you may be able to enter

`boot:` *linux tmx8xx=*⟨*interrupt*⟩*,*⟨*memory-address*⟩

at the LILO boot prompt, where ⟨*interrupt*⟩ is the IRQ of controller, and ⟨*memory-address*⟩ is the shared memory address. Whether or not you will be able to do this depends on the distribution of Linux you are using; consult your documentation for details.

2.5.3 Problems installing the software

Actually installing the Linux software should be quite trouble-free, if you're lucky. The only problems that you might experience would be related to corrupt installation media or lack of space on your Linux filesystems. Here is a list of these common problems.

- **System reports "`Read error`", "`file not found`", or other errors while attempting to install the software.** This is indicative of a problem with your installation media. If you are installing from floppy, keep in mind that floppies are quite succeptible to media errors of this type. Be sure to use brand-new, newly-formatted floppies. If you have an MS-DOS partition on your drive, many Linux distributions allow you to install the software from the hard drive. This may be faster and more reliable than using floppies.

  If you are using a CD-ROM, be sure to check the disc for scratches, dust, or other problems which might cause media errors.

  The cause of the problem may be that the media is in the incorrect format. For example, if using floppies, many Linux distributions require that the floppies be formatted in high-density MS-DOS format. (The boot floppy is the exception; it is not in MS-DOS format in most cases.) If all else fails, either obtain a new set of floppies, or recreate the floppies (using new diskettes) if you downloaded the software yourself.

- **System reports errors such as "`tar: read error`" or "`gzip: not in gzip format`".** This problem is usually caused by corrupt files on the installation media itself. In other words, your floppy may be error-free, but the data on the floppy is in some way corrupted. For example, if you downloaded the Linux software using text mode, rather than binary mode, then your files will be corrupt, and unreadable by the installation software.

- **System reports errors such as "`device full`" while installing.** This is a clear-cut sign that you have run out of space when installing the software. Not all Linux distributions will be able to cleanly pick up the mess; you shouldn't be able to abort the installation and expect the system to work.

The solution is usually to re-create your filesystems (with the `mke2fs` command) which will delete the partially-installed software. You can then attempt to re-install the software, this time selecting a smaller amount of software to install. In other cases, you may need to start completely from scratch, and rethink your partition and filesystem sizes.

- **System reports errors such as "`read_intr: 0x10`" while accessing the hard drive.** This is usually an indication of bad blocks on your drive. However, if you receive these errors while using `mkswap` or `mke2fs`, the system may be having trouble accessing your drive. This can either be a hardware problem (see Section 2.5.2), or it might be a case of poorly specified geometry. If you used the

  `hd=`⟨*cylinders*⟩`,`⟨*heads*⟩`,`⟨*sectors*⟩

  option at boot time to force detection of your drive geometry, and incorrectly specified the geometry, you could be prone to this problem. This can also happen if your drive geometry is incorrectly specified in the system CMOS.

- **System reports errors such as "`file not found`" or "`permission denied`".** This problem can occur if not all of the necessary files are present on the installation media (see the next paragraph) or if there is a permissions problem with the installation software. For example, some distributions of Linux have been known to have bugs in the installation software itself. These are usually fixed very rapidly, and are quite infrequent. If you suspect that the distribution software contains bugs, and you're sure that you have not done anything wrong, contact the maintainer of the distribution to report the bug.

If you have other strange errors when installing Linux (especially if you downloaded the software yourself), be sure that you actually obtained all of the necessary files when downloading. For example, some people use the FTP command

```
mget *.*
```

when downloading the Linux software via FTP. This will download only those files that contain a "`.`" in their filenames; if there are any files without the "`.`", you will miss them. The correct command to use in this case is

```
mget *
```

The best advice is to retrace your steps when something goes wrong. You may think that you have done everything correctly, when in fact you forgot a small but important step somewhere along the way. In many cases, just attempting to re-download or re-install the Linux software can solve the problem. Don't beat your head against the wall any longer than you have to!

Also, if Linux unexpectedly hangs during installation, there may be a hardware problem of some kind. See Section 2.5.2 for hints.

2.5.4 Problems after installing Linux

You've spent an entire afternoon installing Linux. In order to make space for it, you wiped your MS-DOS and OS/2 partitions, and tearfully deleted your copies of SimCity and Wing Commander. You reboot the system, and nothing happens. Or, even worse, *something* happens, but it's not what should happen. What do you do?

In Section 2.5.1, we covered some of the most common problems that can occur when booting the Linux installation media—many of those problems may apply here. In addition, you may be victim to one of the following maladies.

2.5.4.1 Problems booting Linux from floppy

If you are using a floppy to boot Linux, you may need to specify the location of your Linux root partition at boot time. This is especially true if you are using the original installation floppy itself, and not a custom boot floppy created during installation.

While booting the floppy, hold down `shift` or `ctrl`. This should present you with a boot menu; press `tab` to see a list of available options. For example, many distributions allow you to type

`boot:` *linux hd=⟨partition⟩*

at the boot menu, where ⟨*partition*⟩ is the name of the Linux root partition, such as `/dev/hda2`. Consult the documentation for your distribution for details.

2.5.4.2 Problems booting Linux from the hard drive

If you opted to install LILO, instead of creating a boot floppy, then you should be able to boot Linux from the hard drive. However, the automated LILO installation procedure used by many distributions is not always perfect. It may make incorrect assumptions about your partition layout, in which case you will need to re-install LILO to get everything right. Installing LILO is covered in Chapter 4.

- **System reports "`Drive not bootable---Please insert system disk.`"** You will get this error message if the hard drive's master boot record is corrupt in some way. In most cases, it's harmless, and everything else on your drive is still intact. There are several ways around this:

  1. While partitioning your drive using `fdisk`, you may have deleted the partition that was marked as "active". MS-DOS and other operating systems attempt to boot the "active" partition at boot time (Linux pays no attention to whether the partition is "active" or not). You may be able to boot MS-DOS and run `FDISK` to set the active flag on your MS-DOS paritition, and all will be well.

     Another command to try (with MS-DOS 5.0 and higher) is

```
FDISK /MBR
```

This command will attempt to rebuild the hard drive master boot record for booting MS-DOS, overwriting LILO. If you no longer have MS-DOS on your hard drive, you'll need to boot Linux from floppy and attempt to install LILO later.

2. If you created an MS-DOS partition using Linux's version of `fdisk`, or vice versa, you may get this error. You should create MS-DOS partitions only using MS-DOS's version `FDISK`. (The same applies to operating systems other than MS-DOS.) The best solution here is either to start from scratch and repartition the drive correctly, or to merely delete and re-create the offending partitions using the correct version of `fdisk`.

3. The LILO installation procedure may have failed. In this case, you should either boot from your Linux boot floppy (if you have one), or from the original installation media. Either of these should provide options for specifying the Linux root partition to use when booting. Hold down `shift` or `ctrl` at boot time, and press `tab` from the boot menu for a list of options.

- **When booting the system from the hard drive, MS-DOS (or another operating system) starts instead of Linux.** First of all, be sure that you actually installed LILO when installing the Linux software. If not, then the system will still boot MS-DOS (or whatever other operating system you may have) when you attempt to boot from the hard drive. In order to boot Linux from the hard drive, you will need to install LILO (see Chapter 4).

  On the other hand, if you *did* install LILO, and another operating system boots instead of Linux, then you have LILO configured to boot that other operating system by default. While the system is booting, hold down `shift` or `ctrl`, and press `tab` at the boot prompt. This should present you with a list of possible operating systems to boot; select the appropriate option (usually just "`linux`") to boot Linux.

  If you wish to select Linux as the default operating system to boot, you will need to re-install LILO. See Chapter 4.

  It also may be possible that you attempted to install LILO, but the installation procedure failed in some way. See the previous item.

2.5.4.3 Problems logging in

After booting Linux, you should be presented with a login prompt, like so:

```
linux login:
```

At this point, either the distribution's documentation or the system itself will tell you what to do. For many distributions, you simply login as `root`, with no password. Other possible usernames to try are `guest` or `test`.

Most newly-installed Linux systems should not require a password for the initial login. However, if you are asked to enter a password, there may be a problem. First, try using a password equivalent to the username; that is, if you are logging in as `root`, use "`root`" as the password.

If you simply can't login, there may be a problem. First, consult your distribution's documentation; the username and password to use may be buried in there somewhere. The username and password may have been given to you during the installation procedure, or they may be printed on the login banner.

One cause of this may be a problem with installing the Linux login and initialization files. In this is the case, you may need to reinstall (at least parts of) the Linux software, or boot your installation media and attempt to fix the problem by hand—see Chapter 4 for hints.

2.5.4.4 Problems using the system

If login is successful, you should be presented with a shell prompt (such as "`#`" or "`$`") and can happily roam around your system. However, there are some initial problems with using the system that sometimes creep up.

The most common initial configuration problem is incorrect file or directory permissions. This can cause the error message

```
Shell-init: permission denied
```

to be printed after logging in (in fact, any time you see the message "`permission denied`" you can be fairly certain that it is a problem with file permissions).

In many cases, it's a simple matter of using the `chmod` command to fix the permissions of the appropriate files or directories. For example, some distributions of Linux once used the (incorrect) file mode 0644 for the root directory (`/`). The fix was to issue the command

```
# chmod 755 /
```

as `root`. However, in order to issue this command, you needed to boot from the installation media and mount your Linux root filesystem by hand—a hairy task for most newcomers.

As you use the system, you may run into places where file and directory permissions are incorrect, or software does not work as configured. Welcome to the world of Linux! While most distributions are quite trouble-free, very few of them are perfect. We don't want to cover all of those problems here. Instead, throughout the book we help you to solve many of these configuration problems by teaching you how to find them and fix them yourself. In Chapter 1 we discussed this philosophy in some detail. In Chapter 4, we give hints for fixing many of these common configuration problems.