DrivePro®

Version 3.1

Professional Hard Drive Utility

MICROHOUSE®

SOLUTIONS

SYSTEM UTILITIES THAT WORK

DrivePro®

The Micro House Professional Hard Drive Utility

Disclaimer

Trademarks

IBM, IBM PC, PC/XT, PC/AT, PC DOS, PS/2	International Business Machines Corporation
DrivePro, EZ-BIOS, EZ-Copy, EZ-Drive, EZ-S.M.A.R.T., ImageCast	Micro House Solutions, Inc.
MICRO HOUSE®, Micro House Technical Library	Micro House International, Inc.
DriveSpace, Microsoft, MS-DOS, Microsoft Windows, ScanDisk, Windows 95	Microsoft Corporation
Disk Manager	ONTRACK Data International, Inc.
PKZIP	PKWARE, Inc.

Copyright

MICRO HOUSE®

Micro House Solutions, Inc.

P.O. Box 17515

Boulder, CO 80308

Manual Revision 97.04.01, December 1997.

Printed in the United States of America.

Contents

Introduction

About DrivePro

DrivePro is a professional-caliber hard drive utility for 80x86-type personal computers. It contains all of the tools you may need to set up hard drives, as well as diagnose and recover from problems on hard drives and hard drive controllers.

DrivePro is designed to give you the maximum amount of power and flexibility over the setup of your hard drives. Because of this, many of the tools in DrivePro access and manipulate the hard drive and controller electronics directly. This means that *the possibility for loss of data does exist when using DrivePro*; therefore we recommend that all data be backed up from *all* drives in the machine before using the program. Ideally, you should become familiar with DrivePro by using it to set up and work on a test hard drive that does not contain any critical data. As a precaution, DrivePro posts a warning for any operation that may result in a loss of data and will require confirmation to execute the operation.

DrivePro is continually being added to and improved. It is likely that there have been additions or changes since the printing of this manual; therefore it is strongly recommended that you view the contents of the README.TXT file included in the DrivePro directory prior to running the program.

System Requirements

DrivePro can only be run and installed on a system that adheres to the following guidelines:

- The CPU must be Intel 80X86/PC-ISA compatible, 286 or higher. PC/XT and MCA-interface computers are not supported.
- Known software conflicts include operating systems that require their own Master Boot Records (MBRs) and multiple-OS management software. Some anti-virus software can interpret a DrivePro-created MBR as being infected.

The following requirements apply only while actually running the software. Once you are through setting-up a system with DrivePro, it may be run as usual.

- No other software should be resident in memory during registration or use of DrivePro. To avoid conflicts and ensure that there is sufficient memory, you should make a bootable work diskette (**pg. 13**) and boot from it.
- DrivePro has been tested on and is compatible with the following operating systems: MS DOS 3.3 and above, IBM PC DOS 3.3 and above. Other operating systems may not be supported.

- Because DrivePro must regularly access the hardware directly, it will not run under Microsoft Windows, or any other multitasking operating system/environment.
- Any disk-caching hardware or software must be disabled, including SMARTDRV.SYS and caching controllers. Disk-caching interferes with partitioning, low-level, and high-level formatting by intercepting data that should be directly coming from or going to the drive.

About This Manual

This manual contains not only the information necessary to successfully use DrivePro, but also some in-depth information concerning the workings of hard drives. You will find this additional information helpful in diagnosing hard drive-related problems.

Two symbols are used to quickly identify important sections:

This symbol identifies an important point or warning.

This symbol is a prompt to back-up the information on your hard drive in preparation for a procedure that could or will destroy data on the drive.

See the `README.TXT` file on the diskette for the latest update information which may not be contained in this manual.

Note: DrivePro is designed for users with at least a moderate degree of experience working with computer hardware and DOS. This manual is accordingly geared toward the user with a high degree of familiarity with standard computer and DOS terminology. If you have questions about some of the terminology used here, please consult the reference manual that came with the version of DOS you are using.

Quick Start

In a hurry?

BACK UP YOUR DATA!

It is highly recommend that you become thoroughly familiar with DrivePro's features prior to using the software. If you are already familiar with DrivePro and want to get started with a new session right away, this section provides the essentials to get up and running within minutes. Please note that your version of DrivePro may contain new features that could not be included in this manual. A description of these enhancements is contained in the program directory in **README.TXT**.

You can run DrivePro from a previously installed hard drive as well as a floppy disk. There are four steps to quickly get up and running:

1) To run DrivePro from a floppy drive, first create a bootable working floppy diskette. To do this, format a diskette using the command **`FORMAT A: /S`**. If you wish to copy the system files onto a previously formatted diskette, use the command **`SYS A:`**.

2) If you are going to be running from a hard drive, create a **\DRIVEPRO** directory to copy the files into. Copy files from the original **DrivePro Diskette** 1 to the \DRIVEPRO directory of the hard drive or the working floppy diskette. Boot your system to the DOS prompt.

3) Run **DP.EXE** from the \DRIVEPRO directory of the hard drive or the working floppy diskette. You will be prompted for the serial number of your copy of the product the first time you use DrivePro. This number can be found on the back of this manual. If you will be running DrivePro from a diskette, make sure that it is not write-protected; DrivePro needs to write registration information directly to this diskette.

You will additionally need the DOS memory manager HIMEM.SYS loaded to run the installation utility from within DrivePro. For additional instructions, go to **page 5**. Read the rest of the manual as soon as possible to better understand what is actually being done by each function. Appendix C contains extensive troubleshooting information if you find that that you are experiencing problems.

Quick Reference to DrivePro Features

Setting-Up a New IDE Hard Drive

1) Physically install the hard drive(s) according to the manufacturer's instructions. Ensure that jumper settings are correct on the drive(s).

2) Boot the system with the previously installed drive or work diskette. Change the DOS prompt to the appropriate directory where the DrivePro files reside, then type **"`EZDRIVE`"** and follow the on-screen directions.

3) The drive will be installed and bootable in about a minute.

Retrieving Previous Setup Parameters

This procedure is for use on hard drives that were previously set-up and working but have lost their configuration by, for example, moving the drive to a different system, or the CMOS battery going dead.

1) Run DrivePro (**DP<CR>**) from the previously installed drive or working floppy diskette.

2) Choose **Get Lost Drive Parameters** from the **Utilities** menu.

If the drive's partition table is damaged, this feature will not work, since the information is retrieved directly from the partition table.

Removing EZ-BIOS

1) Run DrivePro from the previously installed drive or working floppy diskette.

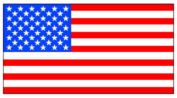
2) Pull Down the **Utilities** menu and select **EZ-BIOS Setup**.
3) Select the drive you want to remove EZ-BIOS from.
4) Set **Controlled by EZ-BIOS** to **DISABLED**. Exit and save changes
5) Repeat the last two steps for all drives that have EZ-BIOS Installed.
6) Select **Uninstall EZ-BIOS**.
7) Follow the prompts to exit the installation utility.

Obtaining Assistance

Please ensure the following before calling Micro House Software Support:

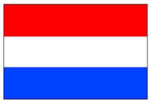
1) Double-check that you have installed the drive correctly before calling. We can only offer support on the DrivePro program, not on the physical installation of hard drives.
2) You must have your DrivePro serial number on hand. We can only give support to valid customers. The serial number can be found on the back of this manual.
3) Send in your registration card today! Be sure to include your serial number, which can be found on the back of this manual. Registration ensures that you get prompt support and lets us keep you informed about the latest product updates. You can also register directly via the World Wide Web: visit **www.solutions.microhouse.com/register**/.

Contacting Micro House



Micro House Solutions (USA)
P.O. Box 17515
Boulder, Colorado 80301

Sales only	(800) 926-8299
Support	(303) 443-3389
Fax	(303) 443-3323
BBS	(303) 443-9957
E-mail	support@microhouse.com
WWW	www.solutions.microhouse.com
FTP	ftp.microhouse.com



Micro House Europe B.V.
Industrieweg 15a
5731 HP Mierlo, The Netherlands

Tel	+31(0)492-430090
Fax	+31(0)492-430026
E-mail	sales@microhouse.nl

For the latest information on DrivePro, as well as our other products, visit our website at www.solutions.microhouse.com.

Chapter One
Getting Started

In order to give you the most power and flexibility, DrivePro accesses the drive hardware directly. This means that *the possibility for loss of data does exist*, so it is strongly recommended that you back up all essential data prior to proceeding.

Please read the README.TXT file if you find your version of DrivePro differs from the information in this text.

This chapter describes how to install DrivePro and navigate around its interface. It is recommended that you read the rest of the manual as soon as possible to better understand each function. If you are having difficulties with any procedures, refer to Appendix C**.** Also see “Obtaining Assistance” on **page 4**.

First Time Use

Some software may interfere with the initial DrivePro registration, therefore you should disable any programs resident in memory or boot from a “clean” floppy before using DrivePro.

You can run DrivePro from a previously installed hard drive or from a working DrivePro floppy disk. It is best if the floppy disk is bootable, since diskette swaps may be required otherwise.

Before using DrivePro, any new drives need to be physically installed into the system. For assistance with drive installation, refer to the hardware manufacturer’s documentation. General installation instructions are also available in Chapter Eight of this manual.

Creating a DrivePro Directory/Bootable Floppy Diskette

1) To run DrivePro from a working floppy diskette, first make a clean DOS-bootable floppy. Do this by formatting a new diskette using the command **`FORMAT A: /S`** or **`SYS A:`** on a previously formatted diskette. Substitute the drive letter with the actual letter of the floppy drive containing the diskette.

If you will be running DrivePro from a hard drive, create a \DRIVEPRO directory on it.

2) Copy all the files from the original DrivePro Diskette 1 onto the working diskette or the \DRIVEPRO directory of the previously installed drive.

For the working floppy, copy HIMEM.SYS and CONFIG.SYS from the version of DOS you will be using. Add a line to the CONFIG.SYS on the working diskette to load HIMEM.SYS. The line should look similar to the following:

```
device=himem.sys
```

3) If you will be using a pointing device, such as a mouse to navigate around the DrivePro interface, then install the DOS driver for the device according to the manufacturer's instructions.

It is generally a good idea to avoid loading unnecessary device drivers or resident programs. This is especially true if you will be running the installation utility (EZ-Drive) from within DrivePro. DrivePro's installation utility requires a large, uninterrupted portion of conventional memory that may conflict with the memory requirements of other programs. To run the installation utility outside of DrivePro, boot from a "clean" floppy disk and run EZDRIVE.EXE from the original DrivePro Diskette 1.

If you will be installing different DOS versions, you can make a bootable DrivePro diskette for each version.

In order to be able to use all features, the floppy diskette must not be write-protected. The reason for this is that log files and registration information must be written to the diskette.

Registering DrivePro

You will only be prompted for this user information once, so be sure to enter it correctly!

When running the software for the first time you will be prompted for your name, company name and the serial number for your copy of DrivePro. Your serial number is found on the back of this manual. If you will be running DrivePro from a diskette, **make sure that it is** *not* write-protected; DrivePro needs to write registration information directly to this diskette. If you encounter an error while attempting to perform this step on your diskette, you should **reboot from a clean floppy diskette** and attempt the procedure again. Booting from a clean floppy prevents the loading of any software that may conflict with DrivePro's user registration procedure.

After registration, you are ready to use the program. Keep your registration number handy in case you have to make a fresh copy or need technical support. Be sure to send in your registration card as soon as possible. Registration ensures that you get prompt support and allows us to inform you about product updates. You can also register directly via the World Wide Web at **www.solutions.microhouse.com/register/**.

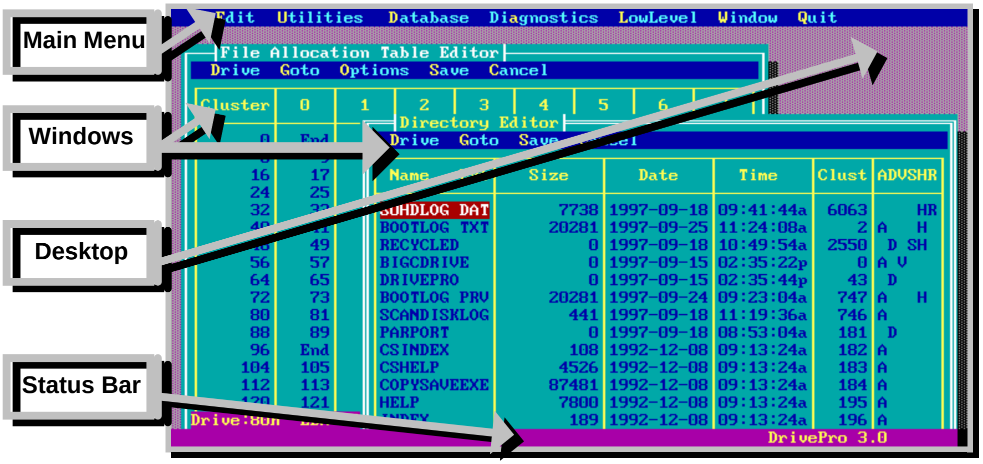
To Run DrivePro

1) First boot from the DrivePro floppy or other clean diskette.

2) Change the prompt to the directory that you will be running DrivePro from. For example **`A:`** from the floppy drive or **`C:\DRIVEPRO`** from a previously installed hard drive.

3) Type **`DP<CR>`** at the prompt.

Getting Around DrivePro



```
 Edit  Utilities  Database  Diagnostics  LowLevel  Window  Quit
 File Allocation Table Editor
 Drive  Goto  Options  Save  Cancel
 Cluster   0    1    2    3    4    5    6
       0  End
                    Directory Editor
                    Drive  Goto  Save
      16   17       Name          Size     Date       Time       Clust ADVSHR
      24   25       SUHDLOG DAT   7738  1997-09-18  09:41:44a   6063     HR
      32            BOOTLOG TXT  20281  1997-09-25  11:24:08a      2 A   H
                    RECYCLED         0  1997-09-18  10:49:54a   2550  D SH
           49       BIGCDRIVE        0  1997-09-15  02:35:22p      0 A V
      56   57       DRIVEPRO         0  1997-09-15  02:35:44p     43  D
      64   65       BOOTLOG PRV  20281  1997-09-24  09:23:04a    747 A   H
      72   73       SCANDISKLOG    441  1997-09-18  11:19:36a    746 A
      80   81       PARPORT          0  1997-09-18  08:53:04a    181  D
      88   89       CSINDEX        108  1992-12-08  09:13:24a    182 A
      96  End       CSHELP        4526  1992-12-08  09:13:24a    183 A
     104  105       COPYSAVEEXE  87481  1992-12-08  09:13:24a    184 A
     112  113       HELP          7800  1992-12-08  09:13:24a    195 A
          121                      189  1992-12-08  09:13:24a    196 A
 Drive:80h
                                                       DrivePro 3.0
```

The DrivePro desktop is very straight-forward and easy to use. Simply pull down menus and select the functions desired. Functions are displayed on the desktop in the form of moveable windows that in turn have their menus. Information relevant to the current operation is displayed in the Status Bar at the bottom of the desktop.

DrivePro Windows

Each time you select an item from the Main Menu, a window will appear. Multiple windows can be open simultaneously, enhancing functionality. Selecting **Window** from the main menu brings up a dialog that allows you to choose from currently open windows. Most windows will remain open until you choose to close them (usually by selecting **Cancel** from the window menu). You can also move windows around the desktop.

General Keyboard Navigation

Use the **left/right arrow** keys to navigate to a menu. Use the **up/down arrow** keys to pull down a menu and highlight menu items.

- To select a menu item, highlight it and press **<CR>**.
- To jump from a function window to the Main Menu, simultaneously press **<ALT>** and the hot key (character indicated by the contrasting color) in the menu desired. In some cases, a local menu function will have the same hot key as a main menu item, in which case the former has precedence.
- You can repeatedly press **<F6>** until a main menu item is highlighted, then pull down the menu with **<CR>** or use the arrow keys.

- To move a window:

  Bring the window you want to move to the front by selecting it from the Windows dialog or using the **<F6>** key.

  1) Press **<F5>**.

  2) Use the arrow keys to move the window around the desktop.

  3) Press **<F5>** again when you are done moving the window.

- If at any point you change your mind about executing a procedure, simply press the **<ESC>** and respond to the prompts to quit or proceed.

Pointing Device Navigation

You can optionally use a cursor to navigate around the DrivePro interface. You must first install the DOS driver for your pointing device (such as a mouse) according to the manufacturer's instructions.

To navigate using a pointing device:

- Place the cursor over a menu and left-click to drop down the menu. Highlight the desired option, then left-click again to select it.
- To move a window, place the cursor on the edge of the window and hold down the left button. Drag the window to the desired location, then release the button.
- To bring a window to the front, left-click on any exposed section of it.

Note: pointing-device navigation is not supported when in the installation utility.

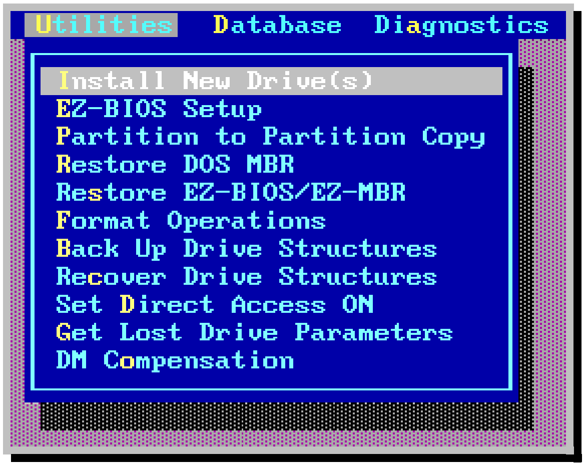
Hot Keys

Hot keys are indicated by their contrasting color in the menu selection. The operation of hot-keys is as follows:

<ALT> + **<hot-key>** to open a drop-down menu. Once a menu is open, do not use **<ALT>** again. Simply press the indicated hot-key alone to bring up the desired window. Pressing **<ALT>** again in combination with a hot-key will pull down another menu or have no effect.

- If a local sub-menu has the same hot-key as a higher level menu, then the sub-menu has precedence. In this unusual case, press **<F6>** to rotate through windows, then press **<ALT>** + **<hot-key>**.
- If a sub-menu does not have same hot-key as a higher-level menu, then DrivePro will look to higher-level menu.

Chapter Two
Drive Setup Functions



This section covers DrivePro's installation and formatting functions. For general information about setting up hard drives, refer to Chapter Eight.

IDE Drive Installation

DrivePro offers easy installation of IDE drives. In most cases, your new drive can be set up and ready to use in under a minute.

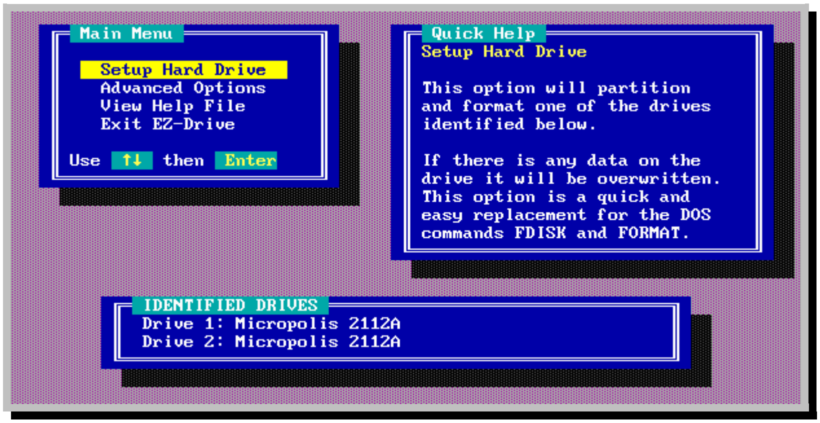
Be sure to back up all data on installed drives before proceeding with any kind of operation that might alter your system configuration.

1) Physically install your new drive(s). Ensure that any jumpers are configured correctly for master/slave designation. If you do not have the manufacturer's documentation on your new hardware, try looking for the proper settings in the installation utility. Pull down the **Utilities** menu and select **Install New Drive(s)**. Go to the **Advanced Options** menu then the **View Drive Jumpers** sub-menu.

2) Go into your computer's CMOS set-up utility and set the proper Drive Type for any new drives.[1] If your CMOS supports automatic detection and setup, use this feature. If you have an older BIOS that does not support the parameters of the drive you are installing or cannot handle large drives (>528MByte capacity), you will need to install EZ-BIOS.[2] DrivePro's set-up utility will automatically recognize this situation and set up the drive as needed.

3) Make sure that the working DrivePro diskette is in a floppy drive.

4) Pull down the **Utilities** menu and select **Install New Drive(s)**.



```
Main Menu
  Setup Hard Drive
  Advanced Options
  View Help File
  Exit EZ-Drive

Use ↑↓ then Enter

Quick Help
Setup Hard Drive

This option will partition
and format one of the drives
identified below.

If there is any data on the
drive it will be overwritten.
This option is a quick and
easy replacement for the DOS
commands FDISK and FORMAT.

IDENTIFIED DRIVES
Drive 1: Micropolis 2112A
Drive 2: Micropolis 2112A
```

5) Follow the prompts and choose from the options given.

The installation utility will analyze the drive and computer, then present you with a recommendation on how to proceed with setup. One of the options you may be given is to install EZ-BIOS. Before proceeding with an installation, you may first want to familiarize yourself with EZ-BIOS and what it does.

What is EZ-BIOS?

Normally, disk access for IDE drives is performed through the system BIOS. In some cases, the system BIOS may not support the full capabilities of a drive. The obvious solution would be to upgrade the system BIOS to a newer version, but in most instances this isn't cost-effective. EZ-BIOS allows you to define parameters for a drive, even if there is not a user-definable type available in the CMOS set-up utility.

[1] The parameters are usually printed on the drive casing. If you do not know the manufacturer's recommended parameters, use the Database Menu to access the Drive Specifications database. Specifications are included for thousands of hard drives dating back to 1984.

[2] Even if you choose to use EZ-BIOS, set the Drive Type to anything except "Not Installed." This forces the BIOS to command the drive to perform a diagnostic routine on itself and reveal any hardware errors that exist.

EZ-BIOS is special code that replaces a conventional Master Boot Record. Once loaded into memory, EZ-BIOS intercepts disk access calls from the BIOS that are intended for the operating system. It then performs the necessary translation to allow the operating system to access the full capacity of the drive. It can also be used to enable special features such as block sector-transfers and Logical Block Addressing (LBA).

EZ-BIOS is most beneficial when:

1) the native BIOS does not support the drive's parameters, or

2) the drive will be moved frequently among different systems.

The most typical use for EZ-BIOS is to compensate for a system BIOS that does not support a drive's setup parameters. Often, especially for IDE drives which do not need to use their physical parameters, a CMOS Drive Type can be found that sacrifices only a small percentage of the disk capacity. If no available type is a satisfactory match, you can usually configure and use a BIOS User-Definable Drive Type. However, many older BIOSes do not have a user-definable type available. These older BIOSes are often also incapable of handling modern hard drives that typically have capacities of 2GB or more.

In such a situation, you must 1) upgrade your BIOS to one that does have a user-definable type, 2) use an EPROM programmer (burner) to write a new drive table to the CMOS, or, 3) least expensively and most easily, install EZ-BIOS.

The second most typical use for EZ-BIOS after compensating for system BIOS limits is to use it for its plug-and-play capabilities. This term refers to the fact that the native BIOS is not used to access a drive set up with EZ-BIOS. The benefits of this independence include:

1) The drive can be moved to any system regardless of the BIOS and can be booted from easily.

2) The setup (CHS) parameters are stored on the drive itself, therefore are not in danger of being lost if the CMOS battery fails.

In either of these cases, the CMOS needs only to be set to anything but 0 or "Not Installed" to get the system to initially look for the drive and run hardware diagnostics. EZ-BIOS will then supply the system with the correct drive parameters.

The use of EZ-BIOS requires special precautions when booting from a floppy. If EZ-BIOS is bypassed and the system BIOS is allowed to control the hard drives, problems may result. For this reason, EZ-BIOS features a delay that allows you to insert a boot floppy after initial system startup. Only insert the boot floppy into the drive after being prompted by EZ-BIOS. See **page 13** for more information on this feature.

Enhancing Drive Performance with EZ-BIOS

See the section above titled “What is **EZ-BIOS?**” for more information on EZ-BIOS. If you have a recently-manufactured computer, it probably natively supports large drives (>528MB). If this is the case it doesn’t need EZ-BIOS on its drives. However, there is no harm in setting up a drive with EZ-BIOS even if it isn’t necessary. In fact, it may be a distinct advantage if you have to move a drive between several systems, since EZ-BIOS is installed on the drive itself. This means that you won’t have to hunt for the correct CMOS parameters every time you move the drive.

- If you did not choose to install EZ-BIOS during the initial drive setup, you can do so at any time by pulling down the **Utilities** menu and selecting **EZ-BIOS Setup**. Select the drive you wish to install EZ-BIOS on and follow the prompts.
- If you already installed EZ-BIOS and wish to remove it, disable it, or change its options, then pull down the **Utilities** menu and select **EZ-BIOS Setup**. Select the drive you wish to change the EZ-BIOS options on.

The following passages detail the EZ-BIOS options. Note that in most cases DrivePro’s installation utility will automatically set up the drive with the appropriate options:

1) **Controlled by EZ-BIOS:** This option designates whether or not EZ-BIOS controls a particular drive. If the drive is not being controlled by EZ-BIOS, then it is being controlled by the system’s native BIOS or it is disabled entirely. If you disable EZ-BIOS and the native BIOS is not set to properly control the drive, you will be given a warning message to that effect. **Disabling EZ-BIOS when your BIOS is not set properly could cause loss of drive access or data corruption**.
2) **Floppy Boot Protection:** This option will allow the disabling of EZ-BIOS’s floppy boot protection. (See the section below for details on properly booting from floppy.) You will need to disable Floppy Boot Protection in order to install Windows NT or OS/2.
3) **Multiple Sector Transfer:** Enables multiple-sectors per interrupt read/write operations for drives that support the feature. Multiple-sector transfers can increase the throughput of a drive significantly.
4) **32-Bit Transfer:** Select this item to enable or disable 32-bit data transfers. This feature requires a local-bus (PCI or VESA) controller, as well as specific hardware support from the drive.

5) **Floppy Boot Delay Time:** This option will increase the amount of delay time for EZ-BIOS boot-up in order to allow the user to boot from a floppy. The message "**EZ-BIOS: Press CTRL key to view status screen or to boot from floppy**" will appear on system start-up, allowing you a specified period of time to press the CTRL key if desired. On faster machines, it may be necessary to increase this number in order to properly boot from a floppy diskette.

6) **Logical Block Addressing:** Logical Block Addressing (LBA) is a method that some drives use to find a particular location of data on a disk. Enabling this option means that EZ-BIOS is communicating with the drive using sequential sectors instead of the physical cylinder, head, and sector to specify a location. LBA requires specific support from the drive in order to be enabled.

   Note that DrivePro's editors can locate a sector in LBA format regardless of which form of addressing the drive actually uses.

7) **Uninstall EZ-BIOS:** Use this option to remove EZ-BIOS from a drive. If the system cannot natively support hard drives, you will be warned of this and asked to confirm EZ-BIOS removal. *If this is the case and you choose to remove EZ-BIOS, be aware that you will lose any information on the drive.*

8) **Save Changes:** You must save any changes to installed features before they can be implemented.

Booting From a Floppy Disk While Using EZ-BIOS

Once installed, EZ-BIOS operation is transparent. You will notice no difference in your system operation–with one exception. A drive set up with EZ-BIOS may use different parameters than the system BIOS will use to access the same drive. Booting straight from a floppy disk will allow the system BIOS to access the hard drives, which will result in data corruption or loss. Therefore, a special procedure is required when you need to boot from a floppy disk.

Whenever you want to boot from a floppy diskette, turn on your PC **without** a floppy in the A: drive (as though you are going to boot from the hard drive). As soon as you see the prompt:

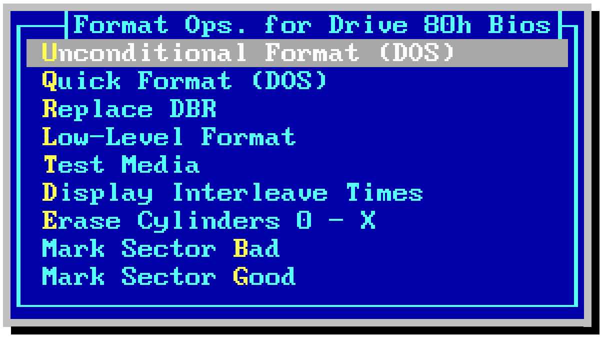
```
EZ-Drive: Hold the Ctrl key for status screen or
              to boot from floppy...,
```

hold down the <CTRL> key[3], and you will be prompted to insert a floppy. Insert a boot floppy and press **<CR>** to boot the computer.

[3] If you get a keyboard error, wait a little longer through the boot sequence before holding the Ctrl key. See **page 12** for information on increasing the boot delay time to facilitate booting from a floppy diskette.

Format Operations

DrivePro offers a variety of high and low-level format operations. These operations are generally destructive to any data already existing on the drive, therefore you should perform a backup prior to proceeding.



Unconditional Format

For a detailed explanation of clusters, refer to page 125 **in** Appendix A**.**

Use this option to perform a full high-level format on a logical volume. **Unconditional Format** allows you to specify the size of clusters, rather than the automatic sizing done by the DOS FORMAT command. In addition, this function will perform a full surface scan and mark off any clusters containing bad sectors in the FAT.

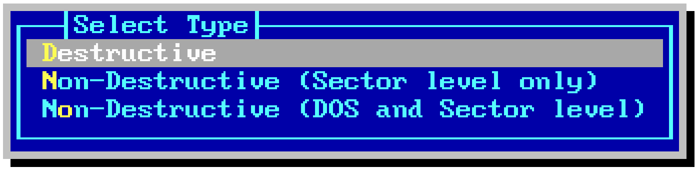
Quick Format

DrivePro allows you to perform a fast high-level DOS format on a logical volume. Pull down the **Utilities** menu and select **Quick Format**. Select the location that you wish to create or overwrite a logical volume. This function will not perform the surface scan that is available through the **Unconditional Format** selection or the DOS FORMAT command. This means that clusters containing bad sectors will not be identified and marked in the FAT.

Low-Level Format

Select this option for low-level format operations. Note that most drives manufactured since the early 90s do not require periodic low-leveling. In fact, most recently manufactured drives use a proprietary low-level that cannot be reproduced outside the factory; using a generic low-level program (such as DrivePro's utility) can destroy such a drive. Consult the drive manufacturer's documentation for more information.

After selecting **Low-Level** Format from the **Format Ops** menu, you will be prompted for the beginning sector, ending sector, and interleave. After entering the appropriate values, you will be prompted to select the manner in which you want the low-level performed:


Select Type
Destructive
Non-Destructive (Sector level only)
Non-Destructive (DOS and Sector level)

Destructive	Reformats sectors without buffering contents to RAM. All information in formatted sectors is destroyed.
Non-Destructive (sector level only)	Buffers tracks to RAM prior to formatting the sectors within, then copies the buffered information back to their original tracks.
Non-Destructive (DOS and sector level)	Same as above, but additionally adjusts the file system if any bad sectors are found in a data area of a logical volume. Clusters containing bad sectors are moved, and the FAT updated to reflect the change. Note: if a bad sector is located within the MBR, DBR, FATs, or root directory, the format will halt and ask you if you want to continue at the level above. If you choose to continue, the rest of the low-level will proceed without regard to any further logical volumes encountered.

After the low-level is finished, a Bad Track Table will appear. Any media failures encountered will be displayed here.

Test Media

Use the Test Media function only if you are sure that the drive can be safely low-leveled. Non-destructive surface scans that are safe for all drives are available in the **Diagnostics** menu.

Test Media verifies the integrity of individual sectors by performing low-level read and write tests in various patterns across the disk surface. The tests require low-level formatting which can be performed destructively or non-destructively as described in the previous section. Test patterns are numbered according to test depth. Higher numbered tests are more thorough, requiring a greater amount of time to perform.

Display Interleave Times

This selection will test the timing of sectors to reveal the interleave the drive was set up with. Unless the drive is an older MFM/ESDI-type, the interleave will almost always be 1:1.

Erase Cylinders 0-x

This selection will zero-out the contents of every sector in the range specified. This feature is most useful for performing a “security erase.” The information in the erased cylinders will be irretrievably lost.

Mark Sector Bad/Good

If specified, makes an entry in the sector header marking the sector as “bad.” Using this function will actually format the entire track that the desired sector resides in, although only the specified sector will be marked bad (if desired). If a sector is marked bad, any data in the corresponding cluster will be moved to one that is free of defects.

Copying Contents from a Previously Set Up Drive

If you are upgrading a drive in your system, you can use DrivePro to ease the transition. With DrivePro, you can set up a new IDE drive and copy the contents of the old to it in one step.

Install the new drive in the system. If necessary, follow the directions outlined in “**IDE Drive Installation**” on page **9**.

Run DrivePro. Pull down the Utilities menu and select **Partition to Partition Copy**. Choose the old drive as the source and the new drive as the target. Follow the prompts until the copying operation is complete.

Chapter Three
Protection and Recovery Features

One of the more frustrating problems commonly encountered in personal computing is being unable to boot or access a particular hard drive. After eliminating the hardware and CMOS setup from the list of possible culprits, you should suspect damage sustained to information in the drive's structural areas. These structures include the MBR, partition tables, DBRs, and FATs.

At first glance, it may seem that the only way to be able to use the drive again is to repartition and/or reformat it. This will destroy any information that it currently holds. Most drive problems, however, can be recovered from without losing the data. With DrivePro, you can store and recover structural information, usually without losing the data on the drive.

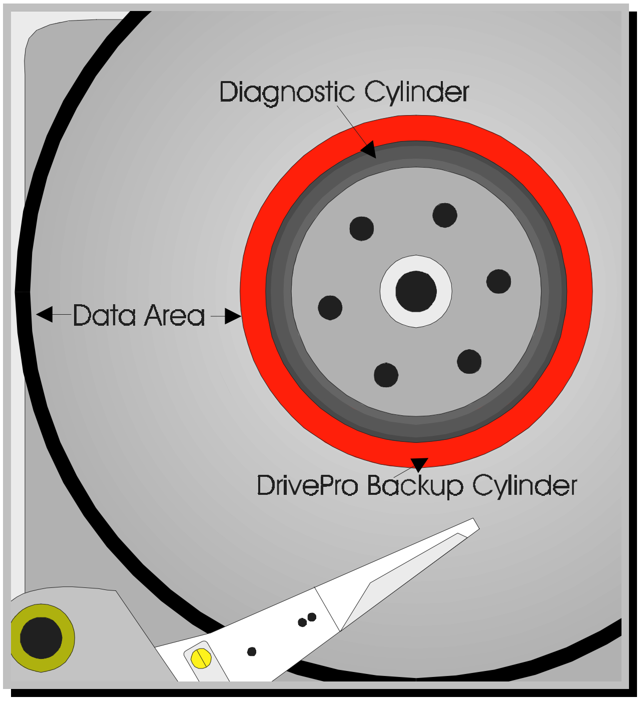
Backing Up Drive Structures

By using the **Back Up Drive Structures** function, you can save vital drive structures to a hidden partition on the drive for later use. These structures include the MBR and partition tables, as well as the DOS Boot Records, FATs, and root directories of any logical volumes on the drive. Using the **Recover Drive Structures** function will restore the state of the system at the time of the last save, as long as the drive can still be written to.

The **Back Up Drive Structures** window can be accessed from the **Utilities** menu. After selecting this option, you will asked to choose the drive(s) to back up information from. When you are done selecting the drives to back up, DrivePro will automatically save all drive structures back-up information.

The backup is saved on the hard drive itself. DrivePro creates a special partition for the backup near the end of the drive. Since the back-up partition is not a DOS partition and is away from the DOS data areas, it will be unaffected by problems that may occur elsewhere on the drive.

The DrivePro backup partition is placed on the second to last physical cylinder of the drive. If this space is already taken up by another partition, DrivePro must crop the partition to free up the needed space. If any data exists there, DrivePro will not overwrite it. You will be prompted to exit DrivePro and move any files taking up the needed space. Defragmenting the drive will typically free up the space. Restart DrivePro to continue the back-up operation.



Automatically Backing Up Drive Structures (BAKER.EXE)

The drive structures backup must be current in order to be effective. In some cases, restoring an old backup can cause permanent loss of data (see below). Because of this, performing a regular drive structures backup routine is crucial. An entry can be placed in the AUTOEXEC.BAT file for a program called **BAKER.EXE**. This program will automatically perform the backup at system startup. To obtain BAKER.EXE, insert DrivePro Disk 2 into the floppy drive. Copy the file UTIL.EXE from the floppy disk to a temporary directory and run it from there. BAKER.EXE will be extracted in the same directory.

To use BAKER.EXE:

1) Run **Back Up Drive Structures** from within DrivePro at least once. This action creates the back-up partition.

2) Copy BAKER.EXE from the root directory of the boot hard drive.

3) Open the AUTOEXEC.BAT file with a text editor and enter:

```
BAKER.EXE {first BIOS drive number} {second BIOS
                  drive number}
```

The numbers after BAKER.EXE designate the drive(s) to back up. You can obtain the BIOS numbers that correspond to all installed drives by going into any DrivePro window with a Drive Select option. Omit the "h" (indicating that the number is hexadecimal) when you enter the number in AUTOEXEC.BAT.

In the example below, BAKER.EXE will back up drive structures on the first, second, and third physical drives installed in the system:

```
BAKER.EXE 80 81 82
```

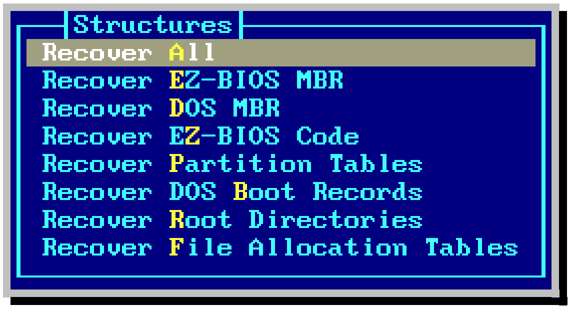
Restoring Drive Structures and Parameters

Important Note: If drive structures become damaged, **do nothing to the drive before using Recover Drive Structures**. This includes attempting to back up data areas. You will risk damaging files beyond repair if you try to access the disk with another program before drive structures are recovered. Especially **do not allow SCANDISK to repair a drive** if you suspect that drive structures have been damaged. Use SCANDISK to check (view) the drive only.

Recover Drive Structures Function

To recover drive structures, first run DrivePro from a floppy or a good hard drive. The **Recover Drive Structures** window can be invoked by calling it up from the **Utilities** menu. You will first be asked to choose the drive to perform the recovery on after selecting this function.

After selecting the drive, you must select the drive structures to restore. Your options will look like this:



Select a structure and press **<CR>** to recover it. **Recover All** will automatically restore all structures. You will be returned to the above menu after each recovery. When you are done, press **<ESC>**.

Be sure that you only restore the information you need. **Be careful NOT to restore drive information that is obsolete. Restoring obsolete data will destroy any changes made to the disk since the last system information backup and can result in data loss.** Typically, the only structures that will be obsolete are the FATs and the root directory. These are updated by DOS each time a file is altered (a file created, deleted or saved). The Root Directory could also be obsolete if it was changed since the last backup.

Only restore old FATs and root directories as a last-ditch measure to make a partial recovery. Files created or changed since the last backup will be lost.

Please note that recovering a drive structure does not guarantee that data files will be recoverable. The contents of individual data files can also be damaged. If you find that a file is corrupt, you must restore it from a previous backup or try to repair it with the **Sector Editor** (**page 25**).

Restoring the Boot Areas of a Drive Without a Backup

DrivePro also features boot recovery functions that don't require a previous backup. If you are primarily interested in just getting a drive to boot, you should at first only attempt to reconstruct the areas of the disk that are required for booting. Begin by resetting the CMOS Drive Type, boot from an alternate source, then back up vital system areas in preparation for recovery.

Important Note: Your first priority in any drive or data recovery operation is to **back up any important data off of the drive as soon as you can access it**. Making the drive bootable is the second priority, once you know the important data is safe!

It may be possible that all you need to recover the drive is to replace the boot code. If drive structures were not previously backed up with BAKER.EXE or the Back Up Drive Structures function, then the following steps will "refresh" the boot code in Track 0.

- The **Restore DOS MBR** function in the **Utilities** menu will replace the MBR with a new, generic DOS MBR. If the main Partition Table has been altered, you will additionally have to repair it to gain access to the data on the drive. See Chapter Five for information on editing the Partition Table.
- If EZ-BIOS has been installed on the drive, select **Restore EZ-BIOS/EZ-MBR** from the **Utilities** menu to replace the EZ-BIOS code. If the main Partition Table has been altered, you will additionally have to repair it to gain access to the data on the drive. See Chapter Five for information on editing the Partition Table.

If the drive was originally set up with EZ-BIOS, a copy of Track 0 (the boot track) was automatically placed in a “hidden” cylinder on the drive for backup purposes. This includes the EZ-BIOS code and the main Partition Table. To restore the backup:

1) Pull down the **Utilities** menu and select **Install New Drive(s)**.

2) Select **Advanced Options** from the **Main Menu**.

3) Select **Backup/Restore Track 0** from the **Advanced Options** menu.

4) Select the most recent backup.

5) When done, exit the installation utility, exit DrivePro, remove the floppy diskette from the drive and reboot the system.

Retrieving Lost Drive Parameters

If your system has lost its CMOS setup and it was not set up with EZ-BIOS, you will need to reset the CHS (Cylinders, Heads, Sectors/Track) parameters for any installed IDE hard drives. The manufacturer’s recommended CHS parameters are usually printed on the drive casing. The setup information is also available in the manufacturer’s documentation (if available) or under DrivePro’s **Database Menu: Drive Specifications**.

DrivePro can also attempt to determine the likely parameters used in setting up the drive by analyzing the Partition Table. To access this function, pull down the **Utilities** menu and select **Get Lost Drive Parameters**. Note that DrivePro must be able to find the Partition Table for this to work.

This page intentionally left blank.

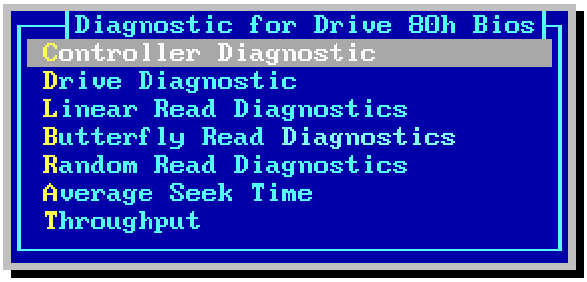
Chapter Four
Diagnostics

If you suspect that there is a hardware problem with your drive, the first step is to back up all information on the drive.

DrivePro has built-in diagnostics capabilities that can quickly assess the state of a drive's health. These tests cover internal drive electronics, surface media analysis, and throughput benchmarks.

Note that modern hard drives are very reliable and that there are other causes of system performance degradation. These include file fragmentation and corruption. It's a good idea to maintain a regular system diagnostic routine that includes utilities such as CHKDSK, ScanDisk, and DEFRAG in addition to DrivePro's diagnostics.

To access DrivePro's diagnostic functions, select **Diagnostics** from the Main Menu. Select the drive you wish to run test on and DrivePro will present you with a list of options.



Controller Internal

This feature executes the controller's internal diagnostics firmware. This function will only work for controllers that support it.

Drive Internal

This feature executes the disk drive's internal diagnostics firmware.

Linear, Butterfly and Random Read Tests

These routines test the drive's ability to read data from the drive. These are all **non-destructive** tests. The reason for the different types of read operations is to subject the drive to a variety of test conditions. Select the type of test desired and DrivePro will present you with a menu of test depths. More thorough tests take longer to finish, but the results are more accurate.

After selecting the test depth, the test will be run. This may take several minutes to an hour or more, depending on the test depth, size of the drive, and overall speed of your system hardware. You will be presented with a report of the results after the test is finished.

Throughput

DrivePro can test the performance of a drive by measuring the average throughput that a drive is capable of sustaining. Throughput is the amount of information the drive is able to transfer to system memory, and vice-versa, measured in bytes per second.

About Measuring Drive Performance

The most widely-published performance rating is **seek** time. Most software utilities will report the **seek** time of the drive. While this can be useful and is one of the most critical aspects of a drive's performance, a more practical measure of drive performance is **throughput**. Seek time is a rating of the time involved when moving the drive heads across the platters. In addition to seek time, however, there are additional factors such as **latency**, which is the time involved in spinning the platter around so that the desired data is below the heads, **head switch time**, which is the time involved in switching to a different head of the head assembly, and controller, BIOS, and system bus to memory overhead.

These factors all affect the speed at which desired information can be retrieved and made available to applications. By performing data transfers from the drive, all of these factors can be measured together as overall system throughput. Dramatic performance (throughput) increases can be realized by such innovations as track buffers, caching, and local bus controllers.

Note: manufacturers of controller cards and drives often publish throughput ratings for their products, but these figures are usually based on ideal conditions. The published figure may therefore differ from the actual overall system throughput achieved and measured by DrivePro. Note also that performance can be affected by the type and manner of hard drive access of a particular application. An application which does more random access of information will perform more poorly than one that confines its accesses mainly to a particular area of the drive.

Chapter Five
Sector-Level Editors

DrivePro has extensive editing capabilities that allow you to directly alter the contents of individual sectors. These editors are most often used in data recovery operations. To access DrivePro's edit functions, pull down the **Edit** menu and select the editor you wish to use.

Warning: these editors are designed for experienced users. If you do not feel that you fit into this category, you may wish to experiment with the editors on a test system that contains no valuable data. In any event, be sure to back up the information on any drives in the system prior to using any of the editors.

For maximum flexibility, you can use the **Sector Editor** to perform all editing chores. The other editors are specifically tailored to individual drive structures to simplify operation.

Note that drive structures are related in a specific manner. Because of this, you should not alter their contents in a random order during a repair/edit procedure. It is recommended that you edit drive structures in this order:

1) Partition Table

2) DOS Boot Record

3) Directory

4) File Allocation Tables

Before you begin, you should be aware of two issues that may affect how you work at the sector level through DrivePro: drive access and Disk Manager© sector offset.

Set Direct Drive Access

Normally, DrivePro will go through the BIOS to access a drive. In some instances, the BIOS may not be able recognize a drive. This occurrence is most commonly caused by ATAPI devices and the use of incorrect drive parameters. If the BIOS cannot access a drive for any reason, then DrivePro must work with the drive directly.

If you are working with an ATAPI device, or having trouble accessing any IDE (ATA) device correctly, pull down the **Utilities** menu and select the **Set Direct Access** option. The default mode for this switch is **OFF**. In most cases, manually toggling the switch is not necessary. DrivePro will set the switch to the appropriate setting automatically.

Note: If the drive parameters are entered incorrectly in CMOS, DrivePro cannot reliably find drive structures, even when Direct Access is ON. If the information displayed in an editor appears incorrect, verify that the drive was booted with the correct parameters.

Disk Manager Compensation

Disk Manager (DM) is a software tool that is similar in function to DrivePro's installation utility. However, it uses a completely different method to achieve its results. DM sets up the drive with a proprietary geometry that offsets the contents of the drive by 63 sectors. When working with sector addresses in a DM drive, you will need to compensate for this offset. The adjusted (DM) address is obtained by subtracting 63 sectors from the physical address.

DrivePro can automatically do this for you. Run DrivePro. Pull down the **Utilities** menu and select **DM Compensation**. The **Select Compensation** window will appear with the following choices:

No Compensation	Cancels any previous DM compensation setting. No compensation will be applied to sector addresses.
Int 13 Compensation Only	Compensates for DM sector offset when drive control is through the Int 13h (BIOS) interface.
Direct Compensation Only	Compensates for DM sector offset when drive control is direct (not through the system BIOS).
Int 13 and Direct Compensation.	Compensates for both DM offset for both Int 13h and direct control of the drive.

Note: DrivePro does not check to see if the drive has been set up with Disk Manager. You will need to know this information prior to working on the drive.

Removing Disk Manager

If you wish to upgrade from the Disk Manager code on a drive to EZ-BIOS, you can do this through DrivePro's installation utility:

1) Pull down the **Utilities** menu and select **Install New Drive(s)**.

2) Install EZ-BIOS according to the instructions contained on page 12. EZ-BIOS *must* be installed before going on to the next step.

3) Select **Advanced Options** from the **Main Menu**, then select **Remove 63 Sector Offset**.

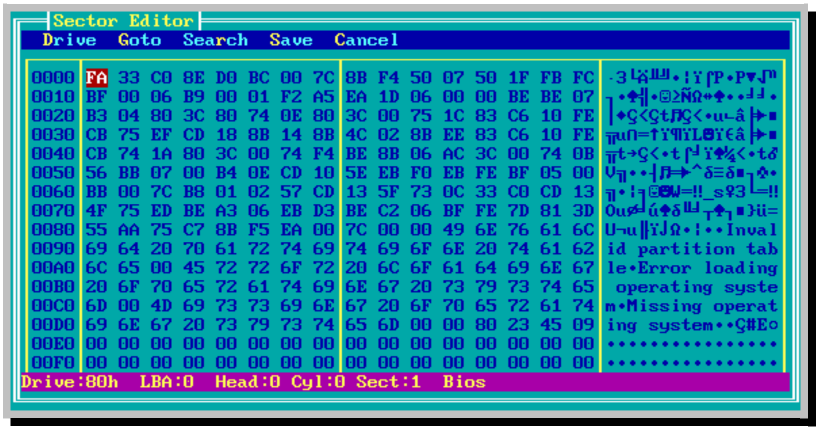
4) Follow the prompts to completion.

By performing the above steps, you will not need to use the DM compensation outlined above.

Note that this procedure will replace the Disk Manager code and return the master boot code, main Partition Table, and DOS Boot Record to a conventional location. The rest of the drive will still have the 63 sector offset. Access to all partitions and data on the drive will be unaffected.

To completely remove the DM code and offset, consult the documentation for the version of Disk Manager contained on the drive.

Sector Editor



```
0000 FA 33 C0 8E D0 BC 00 7C 8B F4 50 07 50 1F FB FC ·3└Ä╨╝•|ï⌠P•P▼√ⁿ
0010 BF 00 06 B9 00 01 F2 A5 EA 1D 06 00 00 BE BE 07 ┐•♠╣•☺≥ÑΩ↔♠••╛╛•
0020 B3 04 80 3C 80 74 0E 80 3C 00 75 1C 83 C6 10 FE │♦Ç<Çt♫Ç<•u∟â╞►■
0030 CB 75 EF CD 18 8B 14 8B 4C 02 8B EE 83 C6 10 FE ╦u∩═↑ï¶ïL☻ïεâ╞►■
0040 CB 74 1A 80 3C 00 74 F4 BE 8B 06 AC 3C 00 74 0B ╦t→Ç<•t⌠╛ï♠¼<•t♂
0050 56 BB 07 00 B4 0E CD 10 5E EB F0 EB FE BF 05 00 V╗••┤♫═►^δ≡δ■┐♣•
0060 BB 00 7C B8 01 02 57 CD 13 5F 73 0C 33 C0 CD 13 ╗•|╕☺☻W═‼_s♀3└═‼
0070 4F 75 ED BE A3 06 EB D3 BE C2 06 BF FE 7D 81 3D Ouφ╛ú♠δ╙╛┬♠┐■}ü=
0080 55 AA 75 C7 8B F5 EA 00 7C 00 00 49 6E 76 61 6C U¬u╟ï⌡Ω•|••Inval
0090 69 64 20 70 61 72 74 69 74 69 6F 6E 20 74 61 62 id partition tab
00A0 6C 65 00 45 72 72 6F 72 20 6C 6F 61 64 69 6E 67 le•Error loading
00B0 20 6F 70 65 72 61 74 69 6E 67 20 73 79 73 74 65  operating syste
00C0 6D 00 4D 69 73 73 69 6E 67 20 6F 70 65 72 61 74 m•Missing operat
00D0 69 6E 67 20 73 79 73 74 65 6D 00 00 80 23 45 09 ing system••Ç#E○
00E0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ••••••••••••••••
00F0 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 ••••••••••••••••
Drive:80h  LBA:0  Head:0 Cyl:0 Sect:1  Bios
```

This is the most powerful and flexible editor option, allowing you to directly alter the contents of any area on a drive. The window is divided into three major sections:

1) The menu at the top of the window allows you to select the drive to edit, jump to a specified sector, save changes, and cancel the edit operation. To access the menu from within the editor, simultaneously press **<ALT>** and the letter key indicated by the contrasting color on the item you wish to select. For example, to select a different drive, press **<ALT>** and the **<D>** keys simultaneously.

2) The status bar at the bottom of the window indicates the drive being edited, and the address of the sector currently being edited in LBA and CHS format.

3) The main editing portion of the window is between the menu and status bar.

- The left-most column indicates the **relative** location of a row of 16 bytes within an individual sector, numbered from 0 to 511 in hexadecimal format (0000h to 01FFh). Note that 512 is the standard number of bytes in a single sector.

- **The two middle columns are the hex edit portion** of the window. These are the contents of individual bytes expressed in hexadecimal format.
- The right-most column represent the contents of individual bytes expressed in extended ASCII format. You can also edit in the ASCII column. **This is also where you will view any changes** you make in the middle (hexadecimal) columns.

Navigating Around the Sector Editor

- Use the **<TAB>** key to advance across a row of bytes. To go back, use **<SHIFT>** + **<TAB>**.
- Use the up/down keys to navigate in columns.
- Due to standard screen size restrictions, only half of the sector can be viewed at once. To scroll up or down, use the **<Page Up>** or **<Page Down>** keys.
- Press **<F4>** to edit in ASCII format. The cursor will appear in the ASCII column.
- To jump to a specific sector, pull down the **Goto** (**<ALT>** + **<G>**) menu and select the sector you want by specifying its LBA or CHS location.
- You can also find data by searching for occurrences of specific character strings or hexadecimal sequences. String searches can be case-sensitive or case-insensitive. To access the search options, pull down the **Search** menu (**<ALT>** + **<R>**) and select the type of search desired.

Saving Changes

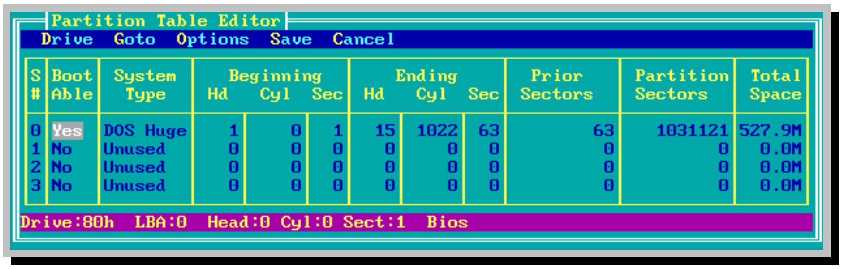
Edits are stored in a buffer until you confirm that you want to write them to disk. To save your changes, select **Save** (**<ALT>** + **<S>**) from the menu. If you do not want to have your changes written to disk, select **Cancel** (**<ALT>** + **<C>**).

If you have made any changes and try to scroll the view or exit the Sector Editor, you will be presented with a dialog that will ask you to confirm changes and give you three options: **Cancel**, **Yes**, and **No**.

- Cancel will not save the changes and return the cursor to the previous location in the Sector Editor.
- Yes will save the changes and complete the operation requested.
- No will not save the changes and complete the operation requested.

Use the **<TAB>** key to rotate through the choices, then press **<CR>**.

Partition Table Editor



S #	Boot Able	System Type	Beginning Hd	Beginning Cyl	Beginning Sec	Ending Hd	Ending Cyl	Ending Sec	Prior Sectors	Partition Sectors	Total Space
0	Yes	DOS Huge	1	0	1	15	1022	63	63	1031121	527.9M
1	No	Unused	0	0	0	0	0	0	0	0	0.0M
2	No	Unused	0	0	0	0	0	0	0	0	0.0M
3	No	Unused	0	0	0	0	0	0	0	0	0.0M

```
Drive:80h  LBA:0  Head:0 Cyl:0 Sect:1  Bios
```

The **Partition Table Editor** is the easiest way to repair or alter parameters for partitions on a drive. Like the Sector Editor, there is a menu and status bar that indicates the current sector location. There are eight major columns in the edit portion of this window. These columns are listed from left to right:

1) The **S#** column indicates the **Slot Number** that the rest of the row relates to. The **Slot Number** is for a partition (for the main partition table) or logical drive (for a secondary partition table).
2) The **Bootable** column designates whether the partition can be used to boot an operating system.
3) The **System Type** column indicates the operating system that is controlling the partition.
4) The **Beginning** and **Ending** columns designate where the partition starts and ends in CHS format.
5) **Prior Sectors** indicates the number of sectors on disk prior to the start of the partition.
6) **Partition Sectors** indicates the total number of sectors in the partition.
7) **Total Space** is the amount of storage capacity in MBytes that the partition contains.

Navigating Around the Partition Table Editor

Use the **<TAB>** and **<SHIFT>** + **<TAB>** keys to navigate across the parameters for a partition. Use the up/down arrow keys to switch between partitions being editing.

To change the drive being edited, pull down the **Drive** menu (**<ALT>** + **<D>**) and select the drive you want to change to.

Some fields in the Partition Table Editor require specific strings to be valid.

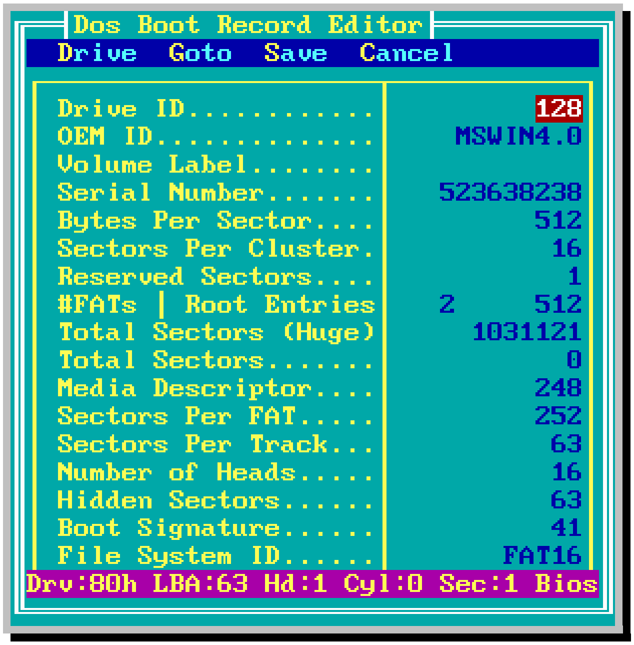
- Entries in the **Bootable** column need to be “Yes” or “No.”

- If you are not sure about the proper entries in the **System Type** column, then try clearing the column and pressing **<TAB>** or the **<DOWN ARROW>**. A pop-up menu will appear listing valid choices for this field. Select the appropriate system type and press **<CR>**.
- Use the Goto menu to quickly jump to the DBR Editor or a secondary partition table. For a DOS-partitioned drive: If Slot 0 is occupied by a logical volume, selecting it from the Goto menu will bring up the DBR Editor for that volume. Selecting Slot 1 will jump to the next secondary partition table in the chain (until reaching the last one). Selecting Back Up One will bring up the previous partition table.

Saving Changes

Edits made to the partition table are not written to disk until you tell DrivePro to save changes by selecting **Save** from the menu (**<ALT>** + **<S>**). Select **Cancel** from the menu (**<ALT>** + **<C>**) to discard any edits made to the partition table and return to the main interface.

DOS Boot Record Editor



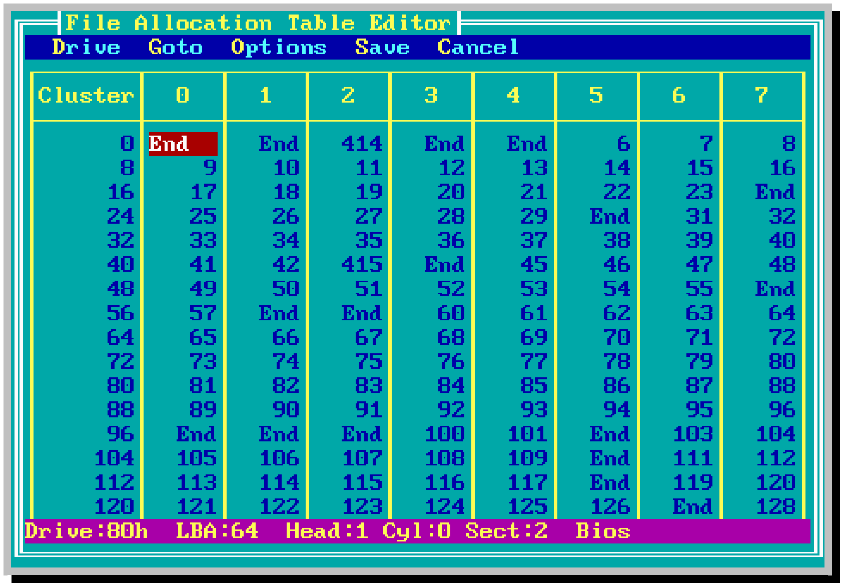
```
Dos Boot Record Editor
 Drive  Goto  Save  Cancel

 Drive ID............            128
 OEM ID..............      MSWIN4.0
 Volume Label........
 Serial Number.......     523638238
 Bytes Per Sector....           512
 Sectors Per Cluster.            16
 Reserved Sectors....             1
 #FATs | Root Entries     2     512
 Total Sectors (Huge)       1031121
 Total Sectors.......             0
 Media Descriptor....           248
 Sectors Per FAT.....           252
 Sectors Per Track...            63
 Number of Heads.....            16
 Hidden Sectors......            63
 Boot Signature......            41
 File System ID......         FAT16
Drv:80h LBA:63 Hd:1 Cyl:0 Sec:1 Bios
```

The DBR Editor is the easiest way to repair or alter parameters for the DBR on a partition.

- Use the **<TAB>** and **<SHIFT>** + **<TAB>** and up/down arrow keys to switch between parameters in the DBR Editor.
- To change the drive being edited, pull down the **Drive** menu (**<ALT>** + **<D>**) and select the drive you want to change to.
- Jump to edit views for either of the FATs or the directory of the currently selected partition by choosing them from the **Goto** menu (<**SHIFT>** + **<G>**).
- Edits made to the DBR are not written to disk until you tell DrivePro to save changes by selecting **Save** from the menu (**<ALT>** + **<S>**). Select **Cancel** from the menu (**<ALT>** + **<C>**) to discard any edits made to the DBR and return to the main interface.

File Allocation Table (FAT) Editor



Cluster	0	1	2	3	4	5	6	7
0	End	End	414	End	End	6	7	8
8	9	10	11	12	13	14	15	16
16	17	18	19	20	21	22	23	End
24	25	26	27	28	29	End	31	32
32	33	34	35	36	37	38	39	40
40	41	42	415	End	45	46	47	48
48	49	50	51	52	53	54	55	End
56	57	End	End	60	61	62	63	64
64	65	66	67	68	69	70	71	72
72	73	74	75	76	77	78	79	80
80	81	82	83	84	85	86	87	88
88	89	90	91	92	93	94	95	96
96	End	End	End	100	101	End	103	104
104	105	106	107	108	109	End	111	112
112	113	114	115	116	117	End	119	120
120	121	122	123	124	125	126	End	128

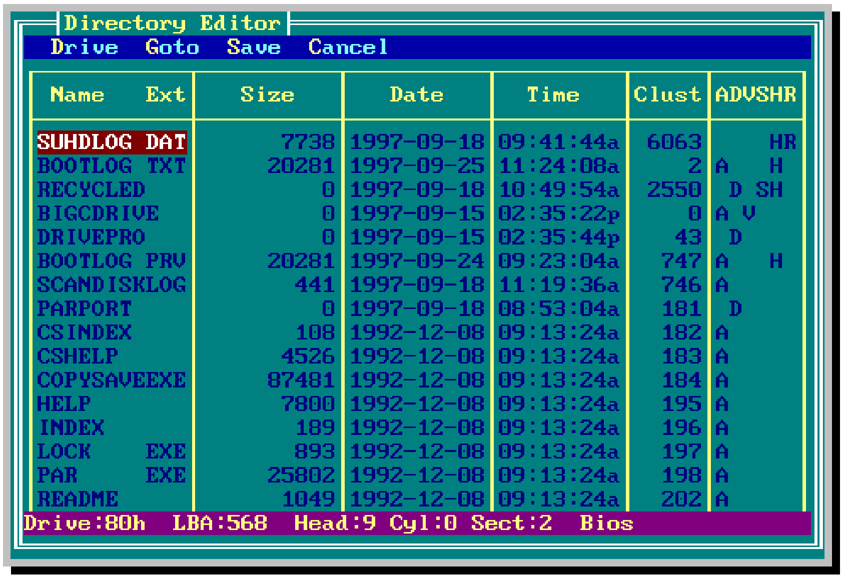
```
Drive:80h  LBA:64  Head:1 Cyl:0 Sect:2  Bios
```

DrivePro will attempt to automatically determine whether the FAT that you are editing is a 12-bit or 16-bit system. In data recovery you should ensure that the file system is correctly identified before making any changes. If you find that the FAT Editor is displaying the wrong FAT type, go into the DBR Editor, bring up the DBR associated with the FAT and correct the “File System ID” entry. Note: the FAT Editor does not currently support viewing 32-bit FATs. You must use the Sector Editor if this is the case.

- Use the **<TAB>** and **<SHIFT>** + **<TAB>** and up/down arrow keys to switch between parameters in the FAT Editor.

- To change the drive being edited, pull down the **Drive** menu (**<ALT>** + **<D>**) and select the drive you want to change to.
- Jump to edit views for either of the FATs, the DBR, or the directory of the currently selected partition by choosing them from the **Goto** menu (<**SHIFT>** + **<G>**).
- Edits made to the FAT are not written to disk until you tell DrivePro to save changes by selecting **Save** from the menu (**<ALT>** + **<S>**). Select **Cancel** from the menu (**<ALT>** + **<C>**) to discard any edits made to the FAT and return to the main interface.

Directory Editor



```
Directory Editor
 Drive   Goto   Save   Cancel

 Name     Ext    Size       Date        Time     Clust  ADVSHR

SVHDLOG  DAT     7738  1997-09-18  09:41:44a   6063      HR
BOOTLOG  TXT    20281  1997-09-25  11:24:08a      2  A    H
RECYCLED            0  1997-09-18  10:49:54a   2550   D  SH
BIGCDRIVE           0  1997-09-15  02:35:22p      0  A V
DRIVEPRO            0  1997-09-15  02:35:44p     43   D
BOOTLOG  PRV    20281  1997-09-24  09:23:04a    747  A    H
SCANDISKLOG       441  1997-09-18  11:19:36a    746  A
PARPORT             0  1997-09-18  08:53:04a    181   D
CSINDEX           108  1992-12-08  09:13:24a    182  A
CSHELP           4526  1992-12-08  09:13:24a    183  A
COPYSAVEEXE     87481  1992-12-08  09:13:24a    184  A
HELP             7800  1992-12-08  09:13:24a    195  A
INDEX             189  1992-12-08  09:13:24a    196  A
LOCK     EXE      893  1992-12-08  09:13:24a    197  A
PAR      EXE    25802  1992-12-08  09:13:24a    198  A
README           1049  1992-12-08  09:13:24a    202  A
Drive:80h  LBA:568  Head:9 Cyl:0 Sect:2  Bios
```

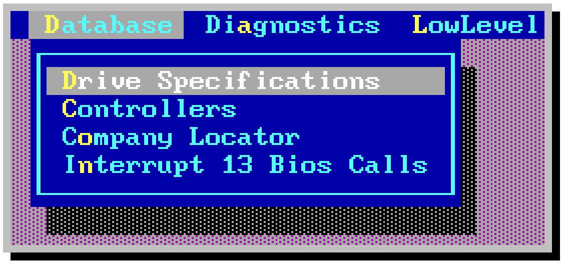
Use this function to edit items in a file directory.

- Use the **<TAB>** and **<SHIFT>** + **<TAB>** and up/down arrow keys to switch between parameters in the Directory Editor.
- To change the drive being edited, pull down the **Drive** menu (**<ALT>** + **<D>**) and select the drive you want to change to.
- To go a subdirectory of the current directory or a directory in another logical volume, use the Goto menu.
- Edits made to the directory are not written to disk until you tell DrivePro to save changes by selecting **Save** from the menu (**<ALT>** + **<S>**). Select **Cancel** from the menu (**<ALT>** + **<C>**) to discard any edits made to the directory.

Chapter Six
Other DrivePro Features

This chapter covers some additional features that you may find useful when working with hard drives.

Databases



This menu leads to reference databases commonly used when working with hard drives and controllers.

Drive Specifications

This section lists the relevant specifications for thousands of hard drives that have been manufactured since 1984. This database gets updated with each new version of DrivePro.

Controllers

This section contains a database of specifications for thousands of common hard drive controller cards.

Company Locator

This function contains relevant organizational contact information.

INT 13h BIOS Calls

The complete list of INT 13h BIOS calls are listed in this section. These are the routines that are contained in the BIOS for drive access and manipulation.

BIOS Drive Table Viewer

This function is accessed by pulling down the **Edit** menu and selecting **BIOS Drive Table Viewer**. This window displays the contents of your system BIOS Drive Table. Use **<Page Up>** and **<Page Down>** to scroll through the table.

Partition to Partition Copy

If you have the need to copy the contents of one partition to another, either within or across different drives, DrivePro can do this for you quickly and easily. To access this function, pull down the **Utilities** menu and select **Partition to Partition Copy**. Be aware that using this function will overwrite any information on the destination partition; back up any information you don't want to lose.

Partition to Partition Copy is intended for drives installed in the same machine. It will not work across a network. If you need this functionality, please visit www.solutions.microhouse.com to find out about ImageCast.

Accessing Low Level Hardware Information and Functions

The Low Level menu contains items that allow access to drive-related hardware electronics through the INT13h interface.

Identify Drive	Returns information about the drive that includes manufacturer, model, CHS and other parameters.
Display Adapter Registers	Returns the current contents of controller registers. You can also use this selection to set and clear register bits.
Unlock Drive Door	Unlocks the door for removable media drives.
Eject Drive Media	Ejects the cartridge for removable media drives.
Seek to Specified Sector	Loads the contents of the specified sector into the hard drive controller's registers.
Park Heads	Sets the drives read/write heads to an area that is away from the data areas of the disk. This function is only required for older drives with stepper motor actuators (such as MFM drives). All modern hard drives automatically park the heads when powered-down.

DrivePro Configuration Options

You can alter the functionality and appearance of DrivePro by changing entries in the configuration file **DPCONFIG.TXT**. Options include cursor behavior, interface colors, and a startup macro.

Configuration File Format

The configuration file consists of groups of commands. Groups of commands are preceded by a header which is indicated by an exclamation point (!).

Configuration File Command Groups

FIELD	Setting the Field command group to "T" allows the cursor to jump to the next field when the end of the field has been reached.
DESK	Sets the ASCII character used for the desktop background.
COLORS	Use this command group to set up the default colors for the various windows and menus of the program.
MACROS	Use Macros to open DrivePro to a specific menu or execute a function on startup. This command group consists of a sequence of keystrokes.

General Syntax

- Removing "!" from a header will cause all the commands within that group to be skipped.
- Each header and command must be on its own separate line of text.
- Comments are preceded by a pound sign (#). Comments on the same line as commands must come after the command.
- Groups are terminated with a "$" symbol.
- Case is not significant.

Color Syntax

command = Foreground/Background

Background colors may only use the first 8 colors below, foreground colors may use the entire list.

Available Colors

Color	Code
Black	BK
Blue	BL
Green	GR
Cyan	CY
Red	RE
Magenta	MA
Brown	BR
Gray	GY
Dark Gray	DG
Light Blue	LB
Light Green	LG
Light Cyan	LC
Light Red	LR
Light Magenta	LM
Yellow	YE
White	WH

Chapter Seven
DrivePro Utilities

This chapter covers the utilities that are part of the DrivePro suite of programs. EZ-Copy and BAKER.EXE are packaged in a compressed file on DrivePro Diskette 2. To unpack them, first copy over the file named UTIL.EXE from DrivePro Diskette 2 to a temporary directory on a hard drive. Running UTIL.EXE from the temporary directory will automatically extract the files.

EZ-S.M.A.R.T.

S.M.A.R.T. stands for Self-Monitoring, Analysis and Reporting Technology. S.M.A.R.T.-capable devices monitor a wealth of information to assess reliability and predict an impending device failure. EZ-S.M.A.R.T. is a Windows-based utility that monitors the S.M.A.R.T. logic in a device and reports any operating degradation or fault conditions that could lead to data loss.

SMART is only an advisory service. It is not an accurate predictor of device reliability!

Here's how EZ-S.M.A.R.T. works: When an operating parameter falls out of tolerance, a S.M.A.R.T. trigger is tripped. EZ-S.M.A.R.T. detects this, analyzes the fault condition and flashes its applet to notify the user. Clicking on the applet will bring up the EZ-S.M.A.R.T. interface, which will display a suggested "Corrective Action" message. You are advised to perform the recommended actions and follow the recommendations by the device's manufacturer.

S.M.A.R.T. cannot detect all impending failures. S.M.A.R.T. should be treated as a feature to assist the computer user in preventing some, but not all, system down-time due to peripheral storage device failure. Data integrity can only be ensured by employing timely back-up procedures. You should not delay back-up or reliability management procedures due to the presence of S.M.A.R.T. on their system.

EZ-S.M.A.R.T. minimum system requirements:

- S.M.A.R.T.-capable drive
- Windows 95/NT or above
- 80486 or above CPU
- 8MB RAM

EZ-S.M.A.R.T. features an easy to use and understand interface and features extensive online help. To install EZ-S.M.A.R.T., boot Windows and run SETUP.EXE on DrivePro Diskette 2. Follow the prompts to completion.

To remove EZ-S.M.A.R.T. from a system, open up the Windows Control Panel and select **Add/Remove Programs**. Select **EZ-S.M.A.R.T. for Windows 95 and NT** and hit **OK**.

EZ-COPY

EZ-Copy is a convenient utility that makes setting up and cloning hard drives a one-step operation. This application has much of the same functionality of the Partition to Partition Copy function of DrivePro's setup utility, but has been arranged specifically to facilitate drive cloning operations.

EZ-Copy was extracted into the temporary directory when you ran UTIL.EXE. To run EZ-Copy, change the DOS prompt to the directory containing EZCOPY.EXE and type **`EZCOPY<CR>`**.

EZ-Copy is intended for drives installed in the same machine. It will not work across a network. If you need this functionality, please visit www.solutions.microhouse.com to find out about ImageCast.

BAKER.EXE

BAKER.EXE is a stand-alone version of the Back Up Drive Structures function found under the Utilities menu. It was extracted into the temporary directory when you ran UTIL.EXE. Use this program to automatically perform drive structure backup each time you boot your computer. For more information on using BAKER.EXE, refer to **page 18**.

Chapter Eight
Drive Installation

Overview

This section covers installation procedures for the two major interface types.

Installation of hard drives can be divided in two major steps: hardware and formatting/partitioning. This section gives step-by-step procedures and tips to make set-up as easy and trouble-free as possible.

Hardware jumper settings will vary depending on manufacturer and model. This means you will need documentation on your main-board, controller card, and hard drive prior to beginning installation procedures.

If this is an upgrade or addition of a secondary drive, before proceeding, make a back up of the entire original disk

Items You Will Need:

✓ Phillips screwdriver

✓ Flat-head screwdriver

✓ Additional/replacement hard drive

✓ Drive cable

✓ Controller card (if upgrading)

✓ Manufacturer's documentation on your:

- hard drive
- controller card
- main-board

✓ DOS Boot-disk with `FDISK.EXE` and `FORMAT.COM` files (DOS installation only)

Items That Will Make Your Life Much Easier:

✓ Micro House Technical Library

✓ Micro House DrivePro or EZ-Drive

Safety Precautions

Take some time to perform some safety precautions prior to moving on to the following steps. These basic precautions could save you from potential financial loss or even personal injury.

- If possible, work over an anti-static mat. In any case, be sure to discharge any static-electricity on your body by touching a grounded object prior to touching any component on your computer. Discharging static into computer components can result in permanent data loss or even damage.
- Turn off the computer and make sure it is not plugged in.
- Use only proper tools and do not attempt to force anything into place.
- Handle the hard drive gently.
- Double check all connections *prior* to plugging in to a power source and turning on.

IDE Drive Installation

If this is an upgrade or addition of a secondary drive, back up the original drive in its entirety.

If you are installing more than one drive, it is recommended that you accomplish all of the steps outlined below sequentially for each drive, rather than installing all of the drives at once. This will make trouble-shooting much easier if you encounter any problems.

IDE Installation Summary

The following outline is a general instruction for installing IDE drives. Please read the entire section for a step-by-step installation guide.

1) Install the hard drive(s), setting the appropriate jumpers and connecting the drive(s) to the controller via the IDE cable.

2) DrivePro and EZ-Drive from Micro House are highly recommended hard drive set-up utilities. The use of these excellent programs could save you some time and/or headaches.

3) If you are proceeding manually, enter the CMOS set-up utility and define the drive-type.

4) Set up the desired partitions, then format the drive using the appropriate FDISK and FORMAT commands.

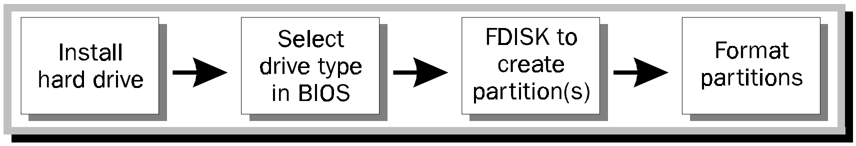


Figure 8-1: IDE Drive Installation

IDE Step-by-Step Installation

Hardware Installation

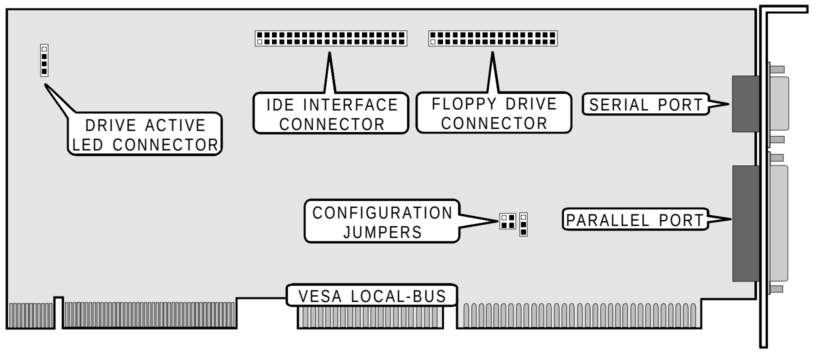
1) Take the general precautions listed in this chapter under the heading "Safety Precautions" prior to proceeding.

2) Remove the outer casing from the computer chassis. The retaining screws are usually located in the back or underneath.

3) Take some time to familiarize yourself with the organization of the chassis and some of the components prior to proceeding. Chassis organization can vary depending on manufacturer, size, and tower or desk-top orientation.

4) If you are upgrading or attaching a new IDE controller, first remove the old one and find an appropriate expansion slot to install the new one. Prior to installation, configure the controller according to the options you would like enabled. Jumper settings can be found in the manufacturer's documentation.

If you cannot find the documentation and do not know the manufacturer of your controller, the Micro House Technical Library contains jumper settings for many of the most popular hard drives, controllers, and main boards manufactured. This handy reference contains the equivalent of thousands of pages of hardware documentation on a single CD-ROM. For more information, visit www.microhouse.com.



This diagram depicts some typical features of after-market IDE controllers, which may have other I/O options. This layout will vary among adapters, but the illustration should help you recognize basic features.

Figure 8-2: IDE Controller Connection Diagram

If you wish to install more than two drives (other than SCSI) on your system, you will need native support such as found in Enhanced IDE systems. The alternative is to enable a secondary controller through third-party software, such as DrivePro's installation utility.

The additional controller card <u>must</u> support secondary port addressing or it cannot be used. To determine if your card does, refer to your documentation to see if there is a jumper for this feature. If you are still unsure, ask the manufacturer. If the card does not support it, you must get one that does. Some points about secondary adapter support:

- One or two drives can be used on the secondary adapter. The drives on the secondary card are not usually set in CMOS.
- IDE drives on the secondary card can be set up by EZ-BIOS while on the secondary port.

Note that any duplicated ports, such as floppy, parallel, or serial will have to be disabled to avoid conflicts. Some floppy controllers have a jumper for disabling, but this is often not reliable. A more reliable method is setting the floppies to the secondary address where the computer will not find them.

In most cases, the secondary card must have its IRQ disabled. An exception is that some multi-tasking operating systems require IRQ15 to be enabled for the secondary adapter.

Some controllers have a jumper to disable IRQ, some set the port address and IRQ with the same jumper. If your card has a jumper that sets IRQ14 or IRQ15, you can often completely disable it by removing the jumper completely. If your secondary card does not have a jumper to disable IRQ14, you can disable it manually by taping over the connection. On a 16-bit controller, the IRQ14 line is found on the back side card edge, on the 16-bit (short) extension. It is the 7th contact from the left or 12th from the right (D7). Placing a piece of simple transparent tape over that contact will disable IRQ14. If you choose to perform this operation, be sure to take the proper safety precautions.

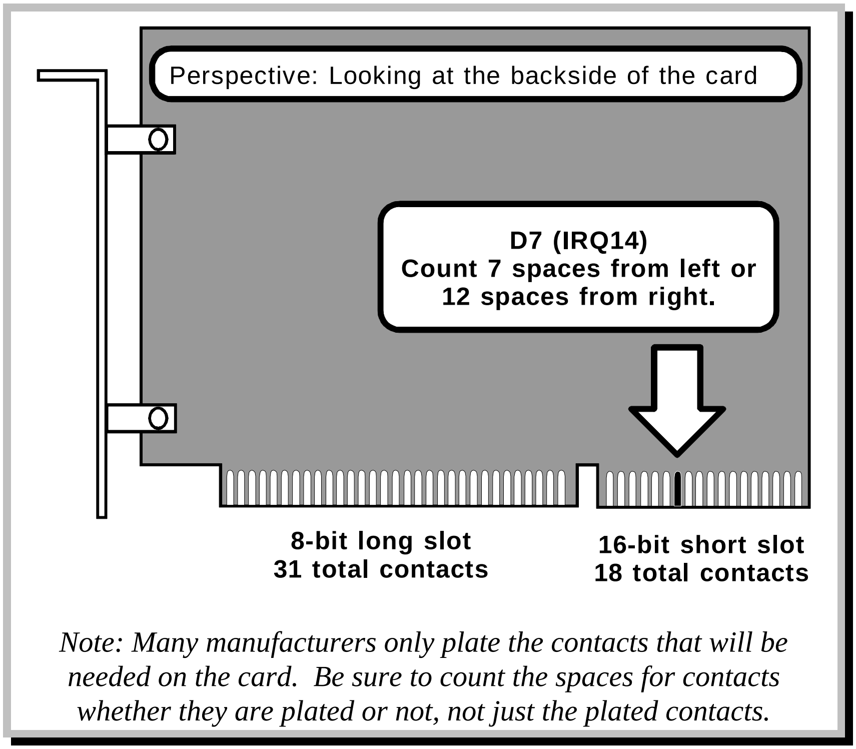

Perspective: Looking at the backside of the card
D7 (IRQ14)
Count 7 spaces from left or
12 spaces from right.
8-bit long slot
31 total contacts
16-bit short slot
18 total contacts
Note: Many manufacturers only plate the contacts that will be needed on the card. Be sure to count the spaces for contacts whether they are plated or not, not just the plated contacts.

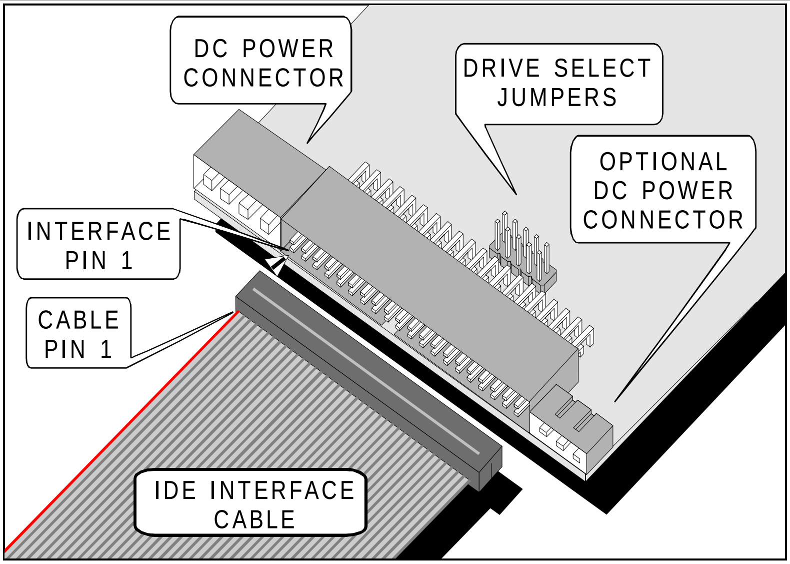
Figure 8-3: Disabling IRQ14

Most current main-boards have an embedded IDE controller. If you are upgrading the controller, the embedded controller must be disabled. Consult the documentation for your mainboard to find the correct jumper settings to disable its embedded controller. If you do not have the manual, contact the manufacturer or reseller.

Some important IDE controller settings of which you should be aware:

- The IDE port enable/disable jumper should be enabled.
- If you intend to use PIO Mode 3 or higher on your IDE drive, the IORDY signal <u>must</u> be enabled. See the section on PIO modes under the heading “IDE Data Transfer” in Appendix B for information on PIO Modes and the IORDY signal.
- If you wish to use one of the DMA data transfer modes, make sure that both the DMARQ and DMACK jumpers are enabled for the DMA channel you will be using.

5) Prior to installation, configure the drive according to the options you would like enabled. Jumper settings can be found in the manufacturer’s documentation.



Although most IDE interface connectors will be oriented this way with respect to the drive, this is not always the case. Be sure to look for a Pin 1 indicator on the interface connector, such as a notch in the connector shroud at that location.

Figure 8-4: IDE Connection Diagram

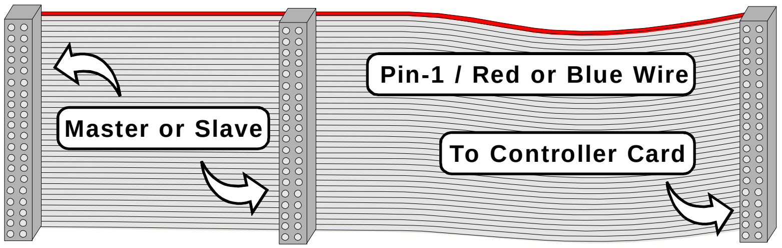
6) Configure the Drive Select Jumpers according to the priority you wish to assign to the drive(s): *single drive only*, *master in a two drive system*, or *slave in a two drive system*. Note that the drive which is designated as the master will be the boot drive. See the sub-section titled “Drive Select Jumpers (Master/Slave),” located under the section “IDE Physical Characteristics” in Appendix B for an explanation of the drive priorities.

7) IDE drives vary widely in options and their corresponding jumpers. Some IDE drive settings of which you should be aware are as follows:

   - If you intend to use PIO Mode 3 or higher on your IDE drive, the IORDY signal must be enabled. See the section on PIO modes under the heading “IDE Data Transfer” in Appendix B for information on PIO Mode 3 and the IORDY signal.
   - If you wish to use one of the DMA data transfer modes, make sure that both the DMARQ and DMACK jumpers are enabled.
   - A few IDE drives offer two choices on activating the spindle motor: activate on system power-up, or activate on command only.

- Some older IDE drives offer spindle synchronization in a two-drive system for special applications. Note that this feature is subject to compatibility problems if the drives are not identical.

8) There are three connectors on the typical IDE drive cable. Attach the appropriate end to the controller connector (the two drive connectors are fairly close together). Determine which connector you will be attaching to the drive(s) according to where the drive(s) will be housed in the computer casing. The cable connector has no bearing on which drive is master or slave in a two drive system.

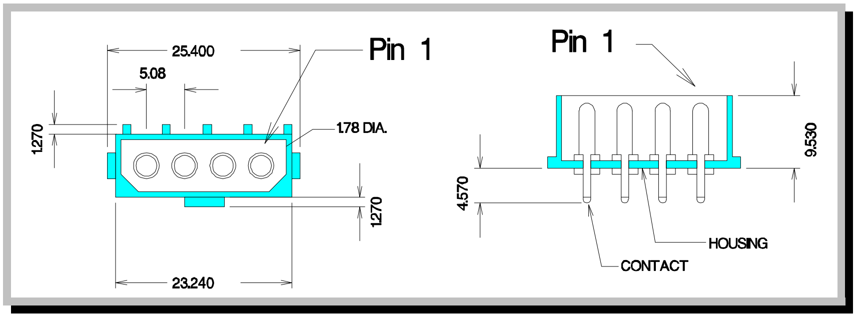


This is the data cable attached to an IDE hard drive. It has 40 pins (Some computers use a 44- or 72-pin connector) and should not exceed 24" in length. The location on the cable is irrelevant to selection of Master or Slave drive.

Figure 8-5: IDE 40-Pin Cable

9) Locate pin 1 on the IDE cable. It can be found by one of three possible markings: a triangle stamped on the connector close to pin 1, a colored stripe on wire #1 (usually red or blue), or a keying tab inside the connector that will not allow the cable to be inserted incorrectly. Orient the cable connector so that it corresponds to pin 1 on the drive interface connector and join the two connectors.

10) Each drive requires its own power cable. The cable is attached to the computer's power supply. The connector is keyed so that it can only be inserted one way.



1	+12 VOLTS
2	+12 VOLTS RETURN
3	+5 VOLTS RETURN
4	+5 VOLTS

Figure 8-6: DC Power Connector

11) Fix the drive(s) into the appropriate drive bays. Most drives can be mounted in any orientation except upside-down, but be sure to check for restrictions if you are not mounting the drive upright and flat (platters parallel to the plane of the floor). Remember that hard drives operate under high tolerances. The selection of an internal location away from hot components and/or with access to good air circulation will help increase the life span of the drive.
12) Perform a quick check to make sure that all connectors are in place. Also make sure all tools and other loose objects are removed from inside the computer chassis.
13) Replace the casing over the computer chassis and complete re-assembly with the appropriate fasteners.
14) Plug the computer back in.

You are now ready to proceed to the BIOS set-up portion of the installation process.

Use DrivePro

At this point, all you have to do is run DrivePro, pull down the **Utilities** menu and select **Install New Drives**. Follow the prompts to completion. That's it! The installation utility will automatically detect your drive's parameters, enters a BIOS drive type, partitions and formats your drive.

If you are going to finish installing the drive manually, please go on to the following steps:

BIOS Set-Up Procedure

The BIOS set-up is also called the CMOS set-up. This procedure enables the BIOS to work with the drive by letting it know the drive's parameters.

1) Double-check all of the connectors prior to plugging in. Power-up the computer and perform the procedure required to enter the CMOS set-up. The CMOS set-up entry procedure can vary depending on the BIOS manufacturer, but is typically accomplished by one of these actions:

    - Pressing and holding down the escape (Esc) key during the Power-On Self-Test (POST).
    - Pressing and holding down the delete (Del) key during the POST.
    - Pressing the (Ctrl), (Alt), and (Esc) keys at the same time while in DOS.
    - Pressing the (Ctrl), (Alt), and (S) keys at the same time while in DOS.
    - Pressing the (Ctrl), (Alt), and (Ent) keys at the same time while in DOS.

    Some BIOS types will automatically go into the CMOS set-up if a system configuration change is detected, which would include hard drive alterations.

2) Go into the hard drive set-up portion of the CMOS set-up and select the Drive Type. This can be done one of three ways: automatic type selection, pre-defined drive type, or user definable drive type.

    Newer BIOS types and IDE drives support automatic detection of the drive parameters. This option is easiest and least open to error. If you are not sure if your drive supports the Identify Drive command, try this option anyway. If automatic selection does not work, you will have to go to one of the other two types selection methods.

    A user-defined drive type should be used if the drive-type table in your BIOS does not include the parameters of your drive. If your BIOS allows manual parameters to be entered, then enter the manufacturer's recommended logical heads, cylinders, and sectors per track for the drive. Be sure to enter the ***logical*** parameters of your drive. These values are usually printed on the drive casing.

Look for the values that match your drive in the BIOS Drive-Type Table. If your BIOS has a defined type that matches your drive's capacity exactly, then use it.

Remember that DOS does not support more than 1,024 cylinders. To access over 1,024 cylinders (in effect > 528MBytes) you will need either hardware or software support such as EZ-BIOS. Please see the section on IDE Technical Information in Appendix B for more information on DOS-imposed capacity limits.

In some instances, you may find that your BIOS does not support automatic type detection or a user-definable drive type, and you cannot find the exact matching type in the table. If this is the case, then try selecting the drive type in the table that most closely matches the parameters of your drive, but has a capacity that is <u>less than or equal</u> to your drive's capacity. This procedure also applies if you don't have the logical conversion information, but know the capacity, because most IDE drives do a translation so that they may be used in systems that do not support the drive type needed.

For example, if the IDE hard drive to be installed is 22 megabytes in size, type 2 would be selected in the BIOS setup for this drive as it is the closest match in megabytes to the IDE drive.

Remember, the physical information is the actual heads, cylinders, and SPT of the hard drive. The logical information is the heads, cylinders, and SPT that the drive and BIOS table can agree on. Although this will result in a working IDE drive, it is not optimal because in some cases you could lose drive space by having to choose a type that is far less than the drive's potential size.

Note: Be safe—don't go over the correct capacity by even one byte!

3) Exit the drive set-up menu, save the settings, and exit the CMOS set-up. The drive(s) are now ready to be partitioned and formatted.

Low-Level Formatting

IDE drives have been factory low-leveled and only need to be FDISKed and DOS formatted. Low-leveling the drive may wipe out the factory-marked defect list and might even render the drive unusable!

Most IDE drives have variable sectors per track and use sector sparing. They may also utilize wedge servos. These servos mark the track and sector boundaries. Wedge servos are written to the platters by a special machine. The drive itself is unable to re-create these. It is impossible to low-level this type of IDE drive because there is no way to rewrite these servos unless the special machine is used. The drive electronics are intelligent enough to refuse a low-level command. On these types of drives a low-level program will simply scrub the data areas clean.

DOS Partitioning

Please see Appendix A for details on partitioning. FDISK is the utility provided with DOS-based OSes to partition a drive. There are different partitioning limitations for each DOS version, the latest supporting partitions of up to 2GB.

You can lose a significant amount of drive capacity by using a large partition. There is usually a portion of the last cluster used for a file which ends up being wasted. Because DOS uses more sectors per cluster for large partitions, there is a proportional increase in the amount of unused sectors in larger clusters. Please refer to Appendix A for a more detailed explanation of clusters and how they affect wasted drive capacity.

If you are installing a master drive only or a master/slave combination, boot from a DOS disk containing the files FDISK.EXE and FORMAT.COM. If you are only installing a slave drive, boot normally from the master drive.

1) Run the FDISK program by typing `FDISK` at the prompt.
2) Use the menu to set the desired partitioning options for your drive(s), depending on the DOS version you are using (see above). Note that the latest version of DOS allows partitions of up to 2GB, which is enough to allow a single (primary) partition on just about every IDE drive manufactured today. Please see Appendix A for an explanation of primary and extended partitions.
3) For the master or sole drive, set the boot partition to “active” if it’s not done automatically.
4) Exit FDISK. The partition(s) are now ready for high-level formatting.

High-Level Formatting

Once the drive is partitioned, each partition must be high-level formatted with the operating system that you will be using. To format under DOS, use the FORMAT command, specifying the partition to be formatted. For example, type `FORMAT C:` to format the partition designated as “C:” drive.

If you want to make this the boot drive, save a step and use the command FORMAT /S to automatically transfer the system files after formatting. If the partition is already formatted, then use the SYS command to transfer the system files to the partition and make it bootable. Simply put:

- If the partition is unformatted, type `FORMAT C: /S` at the prompt.
- If the partition is already formatted type `SYS C:` at the prompt.

Problems?

- Make sure that pin 1 on the interface connector corresponds with pin 1 of the cable connector.
- Double check that the Drive Select jumpers are set properly. It is easy to inadvertently look at the jumper block from the wrong perspective.
- If you manually entered the drive-type in the BIOS table, it is possible that one or more of the cylinders, heads, or SPT values is not supported by the system.

SCSI Drive Installation

If this is an upgrade or addition of a secondary drive, back up the original drive in its entirety.

If you are installing more than one drive, it is recommended that you accomplish all of the steps outlined below sequentially for each drive, rather than installing all of the drives at once. This will make trouble-shooting much easier if you encounter any problems.

SCSI Installation Summary

The following outline is a general instruction on installing SCSI drives in a DOS environment. Please read the entire section for a step-by-step installation guide.

1) Connect the drive to the controller via the cable.
2) If you are proceeding manually, enter the CMOS set-up utility and define the drive-type as “0” or “Not Installed.”
3) The SCSI controller you are using may require a replacement block device driver. Additionally, some SCSI drives may need to be low-level formatted. Refer to the user’s manual, which came with it, to find out how to install it.

4) Set up the desired partitions, then format the drive using the appropriate FDISK and FORMAT commands.

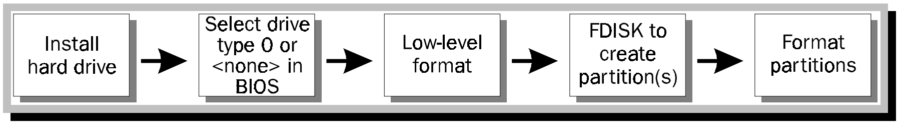

Install hard drive
Select drive type 0 or <none> in BIOS
Low-level format
FDISK to create partition(s)
Format partitions

Figure 8-7: SCSI Drive Installation

SCSI Step-by-Step Installation

Hardware Installation

Take the general precautions listed in this appendix under the heading "Safety Precautions" prior to proceeding.

1) Remove the outer casing from the computer chassis. The retaining screws are usually located in the back or underneath.
2) Take some time to familiarize yourself with the organization of the chassis and some of the components prior to proceeding. Chassis organization can vary depending on manufacturer, size, and tower or desk-top orientation.
3) Install a new SCSI controller if appropriate. If you are upgrading or attaching a new SCSI controller, configure the controller options to reflect your specific environment. Jumper settings can be found in the manufacturer's documentation. Then, remove the old adapter and find an appropriate expansion slot to install the new one.

   If you cannot find the documentation and do not know the manufacturer of your controller, the Micro House Technical Library contains jumper settings for many of the most popular hard drives, controllers, and main boards manufactured. This handy reference contains the equivalent of thousands of pages of hardware documentation on a single CD-ROM. A demonstration of this highly acclaimed tool has been included on the CD-ROM which came with this text.

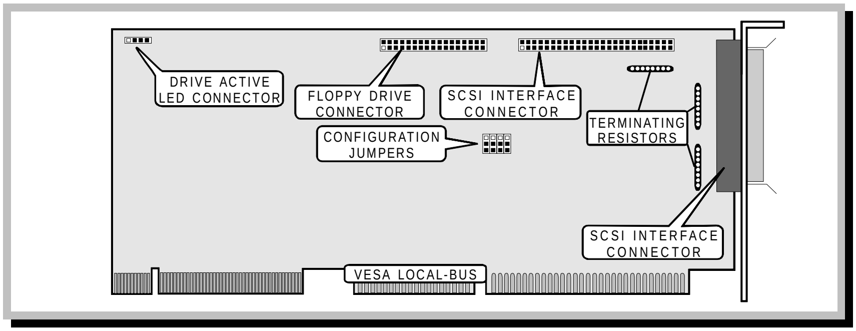


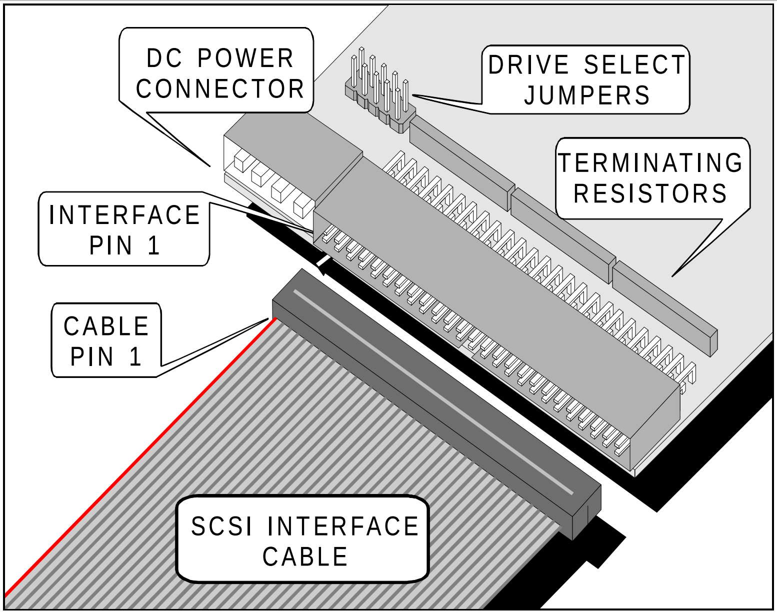
Figure 8-8: SCSI Controller Connection Diagram

4) If your drive is the first or last drive on the device chain and terminating resistors are used to terminate the SCSI bus, make sure these resistors are installed. Drives located at each end of the chain must terminate the SCSI bus so electronic signals at bus ends will be absorbed and not bounce back and interfere with other data.

   Terminating resistors are socketed resistor packs, sometimes called terminators, that are usually yellow and sometimes black or blue. They are typically found on earlier SCSI versions. If your drive uses terminating resistors, there will be several on the underside of the drive near the drive header cable. Leave the resistors installed if your drive is the first or last drive on the device chain. In the case of a single-drive (no other devices) bus, the drive must be terminated. Remove all resistors for drives connected in the middle of the chain. Remember that SCSI controllers are considered one of the eight (or sixteen) devices allowed on the chain. Thus, they also have terminators to terminate the start of the chain.

   On some drives the terminating resistors are not socketed or intended for removal—if this is the case, the resistors can usually be enabled internally by jumper or switch settings.

   Newer SCSI drives do not require terminating resistor packs to terminate drives. Instead, a jumper on these drives enables or disables termination. See next step.



Although most SCSI interface connectors will be oriented this way with respect to the drive, this is not always the case. Be sure to look for a Pin 1 indicator on the interface connector, such as a notch in the connector shroud at that location.

Figure 8-9: SCSI Connection Diagram

5) Configure jumpers on your SCSI drive to define the drive characteristics and functions. Jumper settings can be found in the manufacturer's documentation.

   Common options on Standard/Narrow (8-bit) and Wide (16-bit) SCSI include the following:

   - **SCSI ID** jumpers enable you to assign a unique SCSI ID number for your drive. Three or four 2-pin jumpers are used to set a SCSI ID between 0 and 7 for Standard/Narrow SCSI (0-15 for Wide SCSI), with each jumper representing an individual bit.

Binary Bit Pattern			Decimal Equivalent
0	0	0	0
0	0	X	1
0	X	0	2
0	X	X	3
X	0	0	4
X	0	X	5
X	X	0	6
X	X	X	7

X = Jumper Closed

0 = Jumper Open

LSB (Least Significant Bit)

MSB (Most Significant Bit)

- Depending on the drive, the jumper settings can be reversed. However, the jumper with the smallest assigned value is always called the LSB (Least Significant Bit) and the jumper with the largest value MSB (Most Significant Bit).
- The SCSI ID selected has no relationship to the order in which SCSI devices are attached to the daisy-chain, however it does determined device priority. The SCSI ID number is the sole method of assigning device priority, with higher numbers receiving higher priority. In most cases, the SCSI boot drive should be set to SCSI ID 0 and the SCSI controller to SCSI ID 7, which will give the controller the highest priority on the bus.
- The SCSI ID number corresponds to a physical unit. On systems that support it, drives can also be assigned Logical Unit Numbers **(**LUNs**)**. Each of the seven SCSI ID (physical) addresses on a bus can theoretically have up to eight logical devices attached to it. Not all controllers, drives, or software support logical unit addressing.
- **TERMPWR** jumpers specify whether the drive supplies terminator power *to* the SCSI bus or whether it uses power *from* the bus. If the bus is an A-cable, termination power is supplied to/from pin 26. The pin assignment varies on other cable types.

  Most manufacturers require that at least one device provide terminator power (TERMPWR) to the SCSI bus to prevent accidental grounding or misconnection of terminator power. The device selected for this purpose is usually the adapter. Some manufacturers require all drives provide power, however these devices usually have diodes that protect them from an oversupply of power.

SCSI drives may also draw power from the TERMPWR line to supply on-board resistor circuits or jumpers that control termination. Three commonly seen options allow drive terminator power to be provided by **1)** the drive itself, **2)** by the SCSI bus, or **3)** by both drive and SCSI bus. The options available on your specific drive will be limited by drive circuitry.

- **TERMINATION** jumpers specify whether the on-board termination (usually Active Termination) should be enabled or disabled. Active termination uses voltage regulators rather than terminating resistors to eliminate electrical backflow at bus ends. Active termination is the required termination method for Fast or Wide SCSI. Earlier SCSI versions typically used terminating resistors, a method discussed in step 5. Jumper settings can also be required on earlier drives.

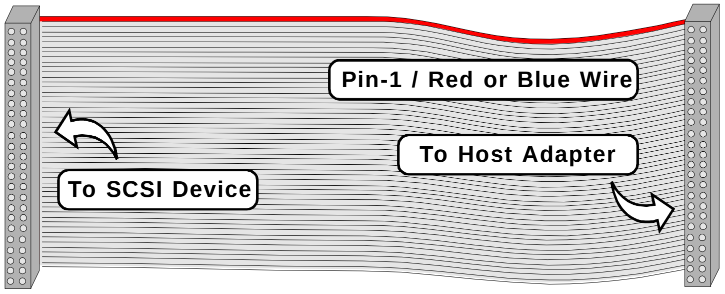
  Many Wide SCSI drives and adapters are allow lower order bytes (bits 0-7) to be terminated separately from high-order bytes (bits 8-15). This is a real advantage because it allows Wide drives to be used on standard 8-bit bus. On adapter cards, separate termination also has advantages because it enables Wide buses to support Narrow device chains. When multiple buses are supported on a single card, it allows the card to support both Wide and/or Narrow device chains.

- Some SCSI drives have **spindle motor power-up option** jumpers. There are two options: spindle motor starts on power-up, or spindle motor starts on command only. Enabling the spindle motor starts on command function allows a compatible controller to stagger the spindle motor power-up. The reason for this is to avoid overloading the power supply by having all the motors start at once.
- On some SCSI drives, a **spindle-motor power-up delay** function is included. This is much the same as the spindle motor starts on command function, except that the drive does not need to be told by the controller when to start the spindle motor, instead it delays power-up by a unit of time multiplied by the SCSI ID number. Lower number drives have shorter delays.
- **Spindle Synchronization** causes all drive heads to rotate to the same read/write location at the same time. This is advantageous in systems such as RAID because it allows a set of drives to function as one. In other systems, an offset is often better because it allows one drive to perform read/write operations while another sets up the next read/write.

- **SCSI parity** should be enabled if your drive *and* controller support this feature. SCSI parity is simply an error checking function which increases the reliability of data transfers.

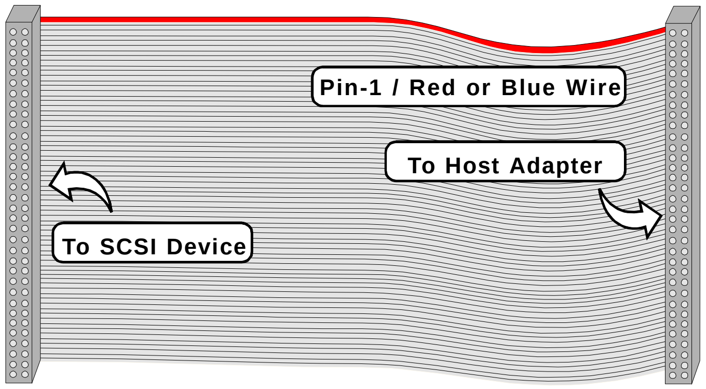
6) There are two or more connectors on the SCSI drive cable. The connectors on the SCSI cable are inter-changeable, meaning either end can be attached to any device. The order in which SCSI devices are attached has no bearing on device priority. This is determined by SCSI ID number.

7) There are two popular SCSI cables in use today, along with their corresponding differential versions. The oldest and most common is the 50-pin A-cable. SCSI-2 specifications accommodate the 68-pin P-cable for Wide SCSI.



This is the 50-pin cable attached to a SCSI-1 or SCSI-2 hard drive or other SCSI device.

Figure 8-10: SCSI Internal A-Cable



This is the 68-pin cable attached to a Wide SCSI-2 hard drive or other SCSI device.

Figure 8-11: SCSI Internal P-Cable

8) Locate pin 1 on the SCSI cable. It can be located by one of three possible markings: a triangle stamped on the connector close to pin 1, a colored stripe on wire #1 (usually red or blue), or a keying tab inside of the connector that will not allow the cable to be inserted incorrectly. Orient the cable connector so that it corresponds to pin 1 on the drive interface connector and join the two connectors. Refer to Figure 8-9 for an illustration.
9) Each drive requires its own power cable. The cable is usually attached to the computer's power supply, but in the case of multiple, external-SCSI device systems, a separate power supply may be necessary. The connector is keyed so that it can only be inserted one way. Electrical assignments and connector specifications may be found in Figure 8-6 on page 46.
10) Fix the drive(s) into the appropriate drive bays. Most drives can be mounted in any orientation except upside-down, but be sure to check for restrictions if you are not mounting the drive upright and flat (platters parallel to the plane of the floor). Remember that hard drives operate under high tolerances. The selection of an internal location away from hot components and/or with access to good air circulation will help increase the life span of the drive.
11) Perform a quick check to make sure that all connectors are correctly configured and secure. SCSI cables can be damaged if connected backwards. Also make sure all tools and other loose objects are removed from inside the computer chassis.
12) Replace the casing over the computer chassis and complete re-assembly with the appropriate fasteners.
13) Plug the computer back in.
14) You are now ready to proceed to the BIOS set-up portion of the installation process.

BIOS Drive Type Selection

The BIOS Setup is also called the CMOS Setup. When installing a SCSI drive into a system, the CMOS drive type should be set to "0" or "NOT INSTALLED" since most SCSI host adapters have a BIOS that supplies the drive type for the drive and handles the read/writes for the drive. If the version of DOS being used is earlier than 5.0, a special software driver will need to be used if more than two SCSI hard drives are installed.

1) Power-up the computer and perform the procedure required to enter the CMOS set-up. The CMOS set-up entry procedure can vary depending on the BIOS manufacturer, but is typically accomplished by one of these actions:
   - Pressing and holding down the escape (Esc) keys during the Power-On Self-Test (POST).

- Pressing and holding down the delete (Del) keys during the POST.
- Pressing the (Ctrl), (Alt), and (Esc) keys at the same time while in DOS.
- Pressing the (Ctrl), (Alt), and (S) keys at the same time while in DOS.
- Pressing the (Ctrl), (Alt), and (Ent) key at the same time while in DOS.

Some BIOS types will automatically go into the CMOS set-up if a system configuration change is detected, which would include hard drive alterations.

2) Go into the hard drive set-up portion of CMOS set-up. and select Drive Type “0” or “Not Installed” from the BIOS Drive Type Table.

Low-Level Formatting

The controllers that do support low-level formatting may usually be accessed through the DOS debug command G=C800:5, or with special firmware that comes with the controller. Refer to the documentation that came with your controller for information on this matter.

SCSI Device Drivers

Some SCSI controllers use a replacement block device driver to set up communication between the operating system and the disk drive. If this is the case, the device driver is specific to the controller model and should come packaged with it. Consult the user’s manual of your particular controller for information on how to install the device driver.

DOS Partitioning

Please see Appendix A for details on partitioning. FDISK is the utility provided with DOS to partition a drive.

If you are installing a new boot drive, boot from a DOS disk containing the files FDISK.EXE and FORMAT.COM. If you are installing additional (non-boot) drive(s), boot normally from the drive that is already installed.

1) Run the FDISK program by typing `FDISK` at the prompt.
2) Use the menu to set the desired partitioning options for your drive(s), depending on the DOS version you are using.
3) For the boot or sole drive, set the boot partition to “active” if it’s not done automatically.
4) Exit FDISK. The partition(s) are now ready for high-level formatting.

High-Level Formatting

Once the drive is partitioned, each partition must be high-level formatted with the operating system that you will be using. To format under DOS, use the FORMAT command, specifying the partition to be formatted. For example, type `FORMAT C:` to format the partition designated as C: drive.

If you want to make this the boot drive, save a step and use the command FORMAT /S to automatically transfer the system files after formatting. If the partition is already formatted, then use the SYS command to transfer the system files to the partition and make it bootable. Simply put:

- If the partition is unformatted, type `FORMAT C: /S` at the prompt.
- If the partition is already formatted type `SYS C:` at the prompt.

Problems?

- *Before applying power*, make sure that pin 1 on the interface connector corresponds with pin 1 of the cable connector.
- Double-check that the SCSI ID jumpers are set properly. It is easy to inadvertently look at the jumper block from the wrong perspective.

This page intentionally left blank.

Chapter Nine

Care and Prevention

Every hard drive will eventually fail. Fortunately, there are steps that can be taken to lessen the likelihood of a spontaneous, catastrophic failure, and to soften the impact when the inevitable occurs. Although hard drive failure is an inescapable eventuality, proper care will allow the drive to fulfill its manufacturer's promise of long life. More importantly, it will allow the drive to give the warnings of normal wear, rather than experience a sudden failure that will leave you stranded with lots of unrecoverable data.

Hard drives are factory-sealed, therefore there is little the end-user can do as far as mechanical maintenance and repair. However, there are steps that can be taken to maximize the life of a hard drive and help prevent a sudden catastrophic failure.

Hard drives operate under very tight tolerances. Exposure to high temperatures could result in a shorter life-span. Try to install the drive away from components that give off large amounts of heat. Also, be sure that there is adequate air circulation inside the computer by checking the cooling fan for dust and obstructions.

Shield the computer from power spikes and surges with a high quality surge suppresser. An erratic power supply could result in corrupt data, and even lead to component damage. Although costly, an uninterruptible power supply would be an even better solution. In addition to shielding the computer from power surges, it allows for an orderly shut down in the case of a general power-outage.

Although modern hard drives are built to withstand minor jolts, it is best to isolate them from shock as much as possible. Because data heads float mere microns from platters spinning at several thousand rpms, an impact sustained during a read/write operation could cause a head "slap," resulting in lost data or even physical damage. Placing the computer on the floor may save desk space, but leaves it vulnerable to the occasional kick or collision with a chair leg.

Warning Signs of Drive Failure

Hard drives sometimes give warning signs when they are about to experience a failure. When these indicators begin to appear, it is best to back up all data on the drive and attempt to determine the cause of any problems through a diagnostic utility.

During the boot sequence, the system BIOS performs a **Power-On Self-Test** (**POST**). If any errors are detected during the POST, it will report the error with a **POST Code** (see **page 69**).

Watch for disk accessing problems, such as files that are corrupt or missing, which seem to appear with greater frequency as time goes by. These are signs of widespread media failure which will eventually become fatal. Listen for any unusual noises that may indicate a faulty actuator or a bearing failure–these will be obvious. Also, run one of the DOS utilities **SCANDISK** or **CHKDSK** often. Do not depend on a drive that shows any sign of unreliability. This does not mean you should throw away drives that exhibit such problems. They can still be used for games and other unimportant files–just remember to keep a current backup.

Some newer drives come equipped with S.M.A.R.T. logic that periodically polls the hardware for signs of degradation. EZ-S.M.A.R.T., included with DrivePro, is a utility that activates S.M.A.R.T. circuits and warns the user of any potential problems.

Not all apparent problems are symptoms of impending failure. Slowing access times are not usually an indicator of impending drive failure–your drive may simply be filling up with so many **fragmented** files that the data heads have to make several jumps to retrieve/write a single file.

On some older drives, periodic low-level formatting is recommended in order to refresh the sector header information. The sector header information is read repeatedly, but never altered during normal read/write operations. On older drives the sector headers tend to degrade over time. Performing a low-level format renews the sector header information, usually fixing apparent media problems.

Most IDE drives are factory low-level formatted with special equipment. These drives cannot be formatted using over-the-counter methods, and attempting to do so may render the drive inoperable. Consult the user’s manual for your drive model prior to attempting a low-level format on it.

Computer Viruses

Some hard drive problems aren’t caused unintentionally, but by high-tech vandals. Computer viruses have in recent years been getting lots of press. The importance of guarding against computer viruses has become greater with the proliferation of effective entry level home computers and increased use of on-line services.

One method of guarding your data is to install a program that automatically detects the presence of computer viruses. For example, a **boot-sector virus** attaches itself to the hidden boot files on your drive, waiting to cause all sorts of damage, such as wrecking the **FAT**, making file access erratic or impossible. Some anti-virus programs are called up during the boot process and can automatically notify the user when a boot-sector virus is detected.

Not all computer viruses hide in the boot sector, however. Consider virus checks as a routine part of your computer maintenance tasks. There are many anti-virus utilities available, and many are shareware. Newer versions of MS-DOS and Windows include easy-to-use anti-virus utilities. Because new viruses are constantly being created, it is important that you are using an anti-virus program with a recently updated database. A good practice is to run a virus check prior to performing a system back up. This incorporates virus checks into your routine, as well as preventing possible infection of crucial back-up data.

Backing Up Data

Modern hard drives are extremely reliable devices which will probably out-last the time it takes for your system to become obsolete. Does this mean that you should have complete confidence in your hardware and that backing-up data on a regular basis is only for the paranoid? The question can only be answered by you. How valuable is your data?

Let's face it: your data is probably much more valuable than the equipment it's stored on. Keeping in mind that all hard drives *will* eventually fail, investing in a data back up system is cheap insurance. If you do not already practice backing up your data routinely, it is strongly recommend that you do so.

Besides eventual hard-drive failure, there are many reasons to be prepared for data loss. Events beyond your control, anything from lightning strikes to accidents and theft could result in losing valuable information. In addition, the inadvertent over-writing of important files is a common occurrence. Having a recent copy of important data will save untold time and effort in reconstructing lost or damaged files.

Back Up Devices

There are many back up methods available today. Again, you will have to balance your needs against what you can afford. It is important to be realistic when choosing a back up method. On the lowest level, you may opt to simply use floppy disks and a data compression program. This will work, but do you really want to be tied to your desk, switching floppies while backing up hundreds of megabytes worth of data? Chances are, this will quickly grow tiresome and seem like it's not worth the effort, which in turn will lead to complacency and an eventual tragic episode.

A back up device should have:

- Removable media to store back up data away from the computer and allow data transfer between computers.
- Adequate storage capacity when compared to the total capacity of the system hard drive.

- A common standard followed by all manufacturers of the device. This allows reliable data transfers among different computers if needed.

There are many choices for backup devices. The one you choose should be based on such considerations as capacity, speed, cost, ease of use, and availability of media.

Tape drives have traditionally been the choice for dedicated backup devices because of their very low cost per megabyte of storage. However, tape drives are comparatively slow and do not allow you to randomly access data. The tape must be wound to the right position before the heads can extract information. Popular tape drive formats include QIC (Quarter Inch Cartridge), DAT (Digital Audio Tape), and 8mm.

Removable media drives are, in theory, a blend of the best features of hard drives and floppy drives – large capacity, high speed, and transferable media. Although not generally used as back up devices, their flexibility and additional features make them an attractive choice over tape drives. The capacity of removable media drives is in theory unlimited–simply add more cartridges for additional capacity. In some cases, the performance of removable media drives is close enough to fixed media drives that they can be considered a viable alternative. Removable media disk drives are currently a growing segment of the computer hardware market. This is mostly because their cost per megabyte has come down considerably in the last couple of years.

Some popular formats for removable media disks include the Syquest products, essentially hard drives with a removable platter, Iomega’s Zip drives, which makes use of high density flexible media, WORM (Write Once, Read Many) and MO (Magnetic Optical) drives.

CD-R (Recordable)/WORM drives were until recently very expensive to purchase and use. CD-R media can be read by conventional CD-ROM drives, but can only be “burned” (written to) once. The main advantage to the CD-R format is that the data is extremely safe once burned into the disk. There is no chance of data corruption due to exposure to magnetic fields. This is because information is stored optically rather than magnetically.

MO drives are outwardly similar to CD-ROM drives, with the important distinction that the cartridges can be written to more than once. Rather than burning dark spots into the disk which prevent light from being reflected back to the optical receiver, the MO disk uses weak magnetic fields to bend the polarized light being reflected from the disk media. The reflective media on MO drives must be excited by heat, applied by a laser beam, before it can receive a magnetic signal. Because a laser can focus the heat on a very tight area, an individual magnetic flux placed on MO media can be much smaller than on plain magnetic media. The result of the ability to tightly localize magnetic fields on the MO disk is that it allows information to be packed more densely than on a straight magnetic disk.

Backup Strategies

As a minimum, you should keep a complete system back up and perform daily back ups of work and changed files. Using the same tape over and over will eventually wear out the media, breaking down the integrity of the stored data. Use multiple tape sets and have a rotation strategy to maximize both economy and reliability. It is also recommended that you keep a copy of the comprehensive back up off the work site. Be sure to update the remote copies on a regular basis. Lastly, check that the backups are reliable by restoring the data every once in a while.

Backing Up Drive Structures

Even a regular data backup routine is only as good as the last save. What about the work that's been done since the last backup? Depending on the nature of the problem, it may be possible to recover some that work if you know enough about repairing structural areas on the drive. An easier solution is to back up those structures every time you boot the drive.

BAKER.EXE is a standalone utility that performs the same function as DrivePro's Backup Drive Structures function. Simply copy BAKER.EXE to a location on the boot drive, then place a line in the AUTOEXEC.BAT to run it during boot. This way, you can protect the structural areas of all installed drives in the system automatically. When a problem occurs, run DrivePro and restore structures as needed. For more information, please see the section in Chapter Three titled "Automatically Backing Up Drive Structures."

Making an Emergency Boot Disk

When your hard drive does experience a failure, you will need to boot the computer by floppy in order to attempt repairs and/or data recovery. An emergency boot disk should contain all the files necessary to boot the computer as well as additional files that contain your system's parameters. To make an emergency boot disk, begin by formatting a clean floppy with the FORMAT /S command. If you are using Windows 3.xx, bring up File Manager, pull down the Disk menu, and choose Make System Disk. If you are using Windows 95, bring up Windows Explorer, right click on the floppy drive and choose Format. Make sure that Copy System Files is checked in. COMMAND.COM and some hidden system files are now transferred to the disk.

Although you can now boot with this disk, you will need additional files for a custom configuration. Copy the CONFIG.SYS and AUTOEXEC.BAT files to the floppy. Using the EDIT command, modify the CONFIG.SYS and AUTOEXEC.BAT files <u>on the floppy disk</u> to streamline the files, eliminating any non-essential instructions. Make sure that any device drivers or other files referred to in CONFIG.SYS and AUTOEXEC.BAT are on the disk and that all paths are changed to work from C: drive to A: drive.

You should also add some files on the disk, which will save some time getting your system back to its pre-failure configuration. Among the files you should copy to the emergency disk are the DOS files FDISK and FORMAT, as well as the diagnostic utilities CHKDSK and SCANDISK. Losing the Master Boot Record (MBR), the partition table, DOS Boot Record (DBR), the File Allocation Table (FAT), or the CMOS drive settings will cause the system to be unable to access the drive correctly.

DrivePro features an Back Up Drive Structures function which allows the user to save the vital disk information to a file for later retrieval in case of disaster. If you are a technician that installs many systems, we suggest that you use this option on every system you set up; that is, save the important system information for every machine to a diskette that will remain with the machine. If any of the mentioned information is lost from the system, as long as the drive can still be written to, running Recover Drive Structures will restore the state of the system at the time of the last system save.

This can save a great deal of time with the newer systems that have many settings, and with IDE drives that must use a user-definable drive type. Specifically, with the IDE translated parameters, if you don't use the original specifications (heads, cylinders and sectors) when resetting this, the IDE drive will not work properly.

Diagnostics

If you are experiencing some problems with your hard drive, the first step is to back up the information on it completely in preparation for a hardware failure. You should then attempt to narrow down the cause of the problem (if it is not already obvious) to determine the reliability of the drive for future use.

DOS contains some basic diagnostic utilities, CHKDSK (DOS only) and SCANDISK (DOS and Windows), which you should run periodically to get an idea of the overall health of your drive(s). SCANDISK can additionally perform a surface scan to spot any media defects. Although SCANDISK and CHKDSK are very useful utilities that should be run as a regular part of disk maintenance, you should not allow either to fix a problem if you suspect that the structural areas of a drive (FATs and directories) have been damaged. Please see **page 83** for more information.

DrivePro contains powerful tools to diagnose and recover from drive problems. The following examples make use of DrivePro's features to illustrate hard drive diagnostics.

A variety of test can help determine if a hardware failure is causing a problem. These tests included controller/hard drive internal diagnostics, and disk surface media examinations.

Controller/Disk Electronics

This option checks the integrity of the controller and disk drive electronics. Since these tests are dependent on the controller and drive, some will not support all commands sent to them. If this is the case with your controller and drive, then messages such as "Can't test controller RAM" (meaning that the controller does not support the controller RAM test) will be returned. This is perfectly normal and does not mean that there is a problem with your controller or drive.

Linear, Butterfly and Random Read Tests

These diagnostics test the drive's ability to locate an area on disk and read the information there. These are all non-destructive tests. Most diagnostic programs also give the user the option of choosing the level of testing thoroughness.

Surface Scan

Surface scan tests give the user various levels of testing to perform on the drive media and report back any defective sectors. Typically, the higher levels will find any bad sectors, but you can run the longer and more thorough tests to be absolutely sure.

Performance

The most comprehensive measure of a drive’s performance is its **data transfer rate**, or **throughput**. Checking seek times and interleave may help diagnose why a drive is not performing up to specifications.

Seek Times

These tests check the drive for average, track-to-track and full-stroke seek times. The seek times are obtained as follows:

Average Seek

Seeks are performed within the highest and lowest cylinders for all possible seek lengths. DrivePro performs the number of cylinders minus 1 seek. This is a more exact measure of average seek time than doing a 1/3-stroke seek as some other programs do.

Track-to-Track Seek

The time it takes the drive to seek between adjacent tracks.

Full-Stroke Seek

The time it takes the drive to seek from the lowest cylinder to the highest cylinder.

These tests are usually accurate but certain variables such as system timing, BIOS type, and transfer rate might cause this figure to deviate slightly. DrivePro's figures will usually not exactly match the manufacturer's due to the fact that their tests usually do not take into account system and controller overhead, and are measured in an idealized way.

Average seek times are most accurate on non-translated drives. Average seek time for SCSI drives may not be a valid test because SCSI drives always perform an addressing translation (see Appendix A). Average seek times on translated ESDI and IDE drives may deviate from manufacturer’s specifications because the seek time between logical tracks may differ from physical tracks, but still can be useful in comparisons.

Throughput

The most widely-published performance rating is seek time. Most software utilities will report the seek time of the drive. While this can be useful and is, in fact, one of the most critical aspects of a drive's performance, a more practical measure of drive performance is throughput. Seek time is a rating of the time involved when moving the drive heads across the platters.

In addition to seek time, however, there are additional factors such as **Latency**, which is the time involved in spinning the platter around so that the desired data is below the heads; **Head Switch Time**, which is the time involved in switching to a different head of the head assembly; and Controller, BIOS, and System Bus-to-Memory overhead. These factors all affect the speed at which desired information can be retrieved and made available to applications. By performing data transfers from the drive, all of these factors can be measured together as overall system throughput.

Dramatic performance (throughput) increases can be realized by such innovations as track buffers, caching, and local bus controllers. Please note: manufacturers of controller cards and drives often publish throughput ratings for their products, but these figures are based on ideal conditions, and for that device only, rather than the whole system, and so may differ from the actual overall system throughput achieved, and measured by DrivePro. Note also that performance can be affected by the type and manner of hard drive access of a particular application. An application that does more random access of information will perform more poorly than one that confines its accesses mainly to a particular area of the drive.

Interleave/Specs

Checks the drive for its current interleave and determines the optimum interleave. The optimum interleave given is only valid for the system in which the drive is currently installed. Interleave testing is usually only needed for ST-506/412 drives.

If you want to permanently adjust the interleave on an ST-506/412 drive, re-do the low-level format. Although this can be done non-destructively, remember to perform a backup first.

Drives manufactured since the late 80s will almost invariably use an interleave of 1:1, as they have a full track buffer. Interleave optimization is not required on these drives.

POST Error Codes

The BIOS performs a Power On Self Test (POST) during the boot process. If an error is detected, the POST will halt the boot process and return an error code corresponding to the problem encountered. No real standard has existed since the original IBM BIOS. Modern error codes are particular to individual BIOS manufacturers.

The table on the following page is only useful for older systems which may use IBM conventions. If your system is fairly current (80486 and above), you should refer to the BIOS manufacturer’s documentation for information. Only the codes pertaining to hard and floppy disks are contained here.

Table 9-1: IBM BIOS Error Codes

01x	Undetermined problem errors	**1703**	Fixed disk drive error
6xx	Floppy drive/adapter errors	**1704**	Fixed disk adapter or drive error
601	Floppy drive/adapter POST failure	**1780**	Fixed disk 0 failure
602	Drive test failure; disk boot record	**1781**	Fixed disk 1 failure
606	Disk change line function failure; drive error	**1782**	Fixed disk controller failure
607	Disk is write protected; drive error	**1790**	Fixed disk 0 error
608	Bad command; drive error	**1791**	Fixed disk 1 error
01x	Undetermined problem errors	**7306**	Disk change line function failure; drive error
6xx	Floppy drive/adapter errors	**7307**	Disk is write protected; drive error
601	Floppy drive/adapter POST failure	**7308**	Bad command; drive error
602	Drive test failure; disk boot record	**7310**	Disk initialization failure; track 0 bad
606	Disk change line function failure; drive error	**7311**	Time-out; drive error
607	Disk is write protected; drive error	**7312**	Bad Controller chi
608	Bad command; drive error	**7313**	Bad Direct Memory Access; drive error
610	Disk initialization failure; track 0 bad	**7314**	Bad Direct Memory Access; boundary overrun
611	Time-out; drive error	**7315**	Bad index timing; drive error
612	Bad Controller chip	**7316**	Drive speed error
613	Bad Direct Memory Access; drive error	**7321**	Bad seek; drive error
614	Bad Direct Memory Access; boundary overrun	**7323**	Record not found; drive error
615	Bad index timing; drive error	**7324**	Bad address mark; drive error
616	Drive speed error	**7325**	Bad Controller chip; seek error
621	Bad seek; drive error	**104xx**	PS/2 ESDI Fixed disk errors
622	Bad Cyclic Redundancy Check; drive error	**10480**	PS/2 ESDI Fixed disk 0 failure
623	Record not found; drive error	**10481**	PS/2 ESDI Fixed disk 1 failure
624	Bad address mark; drive error	**10482**	PS/2 ESDI Fixed disk controller failure
625	Bad Controller chip; seek error	**10483**	PS/2 ESDI Fixed disk controller failure
17xx	Fixed disk errors	**10490**	PS/2 ESDI Fixed disk 0 error
1701	Fixed disk POST error	**10491**	PS/2 ESDI Fixed disk 1 error
1702	Fixed disk adapter error		

Chapter Ten
Data Recovery

If you should experience problems with a drive, hopefully you have taken the appropriate steps necessary to keep down-time at a minimum—the most important being a regular backup routine. Keep an emergency boot disk on hand (see **page 65**).

If the problem is a head malfunction or similarly unrecoverable ailment, getting your system back up to speed means installing a new drive and restoring all necessary files from a back up. If your drive is experiencing difficulties that are somewhere short of catastrophic hardware failure, you may be able to at least recover some of data, if not get the drive fully operational again. The following sections will help you diagnose and recover from common hard drive problems which may arise.

Most data recovery operations are simple and do not require special tools. Some problems, however, require more in-depth recovery methods that involve working directly with the contents of individual sectors. If you wish to attempt this type of data recovery on your own, you should become familiar with drive structures and the manner in which they are related. For more information on this topic, refer to Appendix A.

Warning: recovery operations outlined in this appendix are for drives that were set up conventionally under DOS. If a drive was set up with a proprietary geometry, do not use these procedures. This includes drives set up with Disk Manager. For information on recovering these drives, refer to the documentation that came with the software.

Recovery Tools

As a minimum, you should have the following items to help ensure success during a recovery operation.

- An emergency boot disk (see **page 66**) and DrivePro
- Manufacturer’s hardware documentation
- Alternate working system or drive

Can’t Access a Drive

A common problem is to lose access to a drive. This problem usually occurs after doing something that changed the fundamental setup of a system. Moving the drive to another system, changing the master/slave relationship, or using DOS commands such as FDISK, FORMAT, and SYS all fall under this category.

The first thing to do after discovering that you've lost access to a drive is nothing. Shut down the system as soon as it is safe to do so. Take some time to figure out the cause of the problem. At this point, all you know is that you can't get into the drive. If you rush into the problem without analyzing it, chances are that you will only make the situation worse by destroying data on the drive. There is no magical cure-all for drive access problems. The solution must first be matched to the cause of the problem.

If the cause of the problem isn't immediately apparent, you will have to take a systematic approach to narrow it down. There are four basic problems that can cause drive access problems:

1) hardware failure,
2) wrong BIOS parameters,
3) missing or corrupt drive structures, or
4) missing or corrupt boot files.

If the cause of the problem is not obvious, you should try to eliminate them in the order listed above.

The information in the following sections outlines solutions to simple causes of drive access problems. Often, the problem may be a combination of factors, usually resulting from a misapplication of a recovery procedure. Keep a log of the actions that led up to the problem and the procedures used in the recovery attempt. Be as detailed as possible, writing down all parameters used. Don't do anything that can't be undone, unless you are sure that you will be successful.

First, try to eliminate the hardware as the source of the problem.

Determining Hardware Problems

If you get an error message during boot similar to this one:

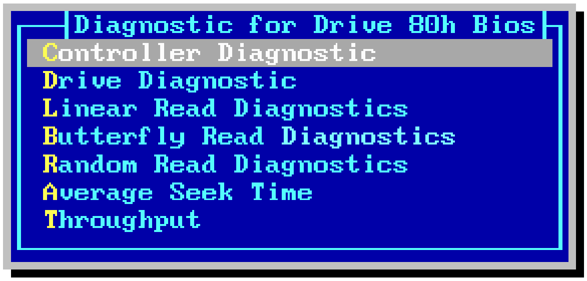
```
HDD controller failure
Press <F1> to RESUME
```

the system is not finding the drive at all, and a hardware problem exists. Note any POST error codes that are being returned at power up. Also check the jumper settings to make sure they are configured properly. A very common mistake is to install a secondary IDE drive without setting the Master and Slave relation properly. Also, check all cable connectors to make sure they are oriented correctly and firmly seated on the interface.

Listen for the noises associated with power-up and initial boot operations. If you hear no noise at all, the interface or power connector may have become unseated. Check the alignment of Pin 1 on the cable and the drive interface. Mechanical problems, such as a head crash, bearing failure, or a defective actuator, tend to be accompanied by distinctive sounds. If this is the case, immediately back up the drive. The drive may still be usable, but should not be trusted with crucial data if you suspect a hardware problem.

If you have verified that the drive(s) are installed correctly, and still cannot identify any other obvious symptoms of hardware failure, the problem may reside in the drive's logic board, the host adapter, or the computer itself. Try switching the drive to a secondary controller (if available), or move it to another system entirely. If the drive is still not working, chances are that the problem is in the drive itself.

DrivePro can diagnose the logic on the drive and host adapter. First, boot with an emergency boot disk (**page 65**) and run DrivePro. Pull down the **Diagnostics** menu and select the drive. Run the **Controller** and **Drive Diagnostic** routines.



A failure in either of these tests indicates a bad logic board, which will require replacement.

Note: depending on the nature of the failure, DrivePro may not be able to see the drive at all. If this is the case, the drive is unrecoverable by software alone.

Lost BIOS Password

Although not technically a drive problem, losing the password set in BIOS setup will effectively lock you out of any drives in the system. If this is the case, you will need to zero-out the CMOS memory to regain access to the system. This action will disable the password protection feature. Most mainboards feature special jumpers that will allow you to do this. If such a jumper block is not available, you will have to disconnect the CMOS battery for a short period of time, then reconnect it. Consult the documentation that came with your mainboard for more information. After zeroing-out the CMOS memory, you will need to set up the BIOS again with the correct parameters.

If you are having problems with resetting the CMOS and can't wait, you can always move the drive(s) to another system. Remember to set up the BIOS in the new system with the exact same parameters.

Wrong BIOS Parameters

After you've inspected the hardware and decided that it is not failing or set up improperly, you need to look at the BIOS setup. A drive often becomes inaccessible due to errors in setting up parameters in the BIOS or loss of power to the CMOS memory. The Drive Type must be re-entered using the parameters that the drive was originally set up with. Using the wrong set of parameters can cause extensive damage to drive structures and data. Hopefully, you have written down the setup information to guard against this possibility. If not, you should try to obtain the parameters using some method that you have confidence in.

If you used the auto-detect feature in the CMOS setup utility to originally set up the drive, doing so again will reset the same parameters. If the drive is installed in an older system with no auto-detect feature, then you must set the Drive Type manually. There are a number of sources that you can use to get the parameters, including the manufacturer's documentation (specifications on usually printed on the drive casing) and drive utilities such as DrivePro.

If you are using a software utility to enable large drive support in an older system (such as EZ-BIOS), it should have a function that will restore the correct Drive Type. *Make sure you restore the correct parameters. Failure to do so will probably result in data loss.*

Figuring out Lost Setup Parameters

To manually figure out lost parameters, you will need to be able to look at the contents of the drive.

1) Start by finding out the manufacturer's recommended set up parameters or running the **Identify Drive** command from DrivePro's **Low Level** menu. Calculate capacity according to this formula:

Capacity in Mbytes = Cylinders x Heads x Sectors per Track x 512bytes per sector

2) Now, find the parameters the drive was set up with one at a time. Open up the Partition Table Editor. You may need to be in Direct mode. Examine the table to see if it makes sense. This will be readily apparent by looking at the numbers in the Total Space column. If everything looks in order, find the number under the Sec portion of the Ending column. This is the number that was used for the Sectors per Track BIOS setup parameter.

3) Open up the DOS Boot Record Editor in Direct mode. More than likely, it will be looking in the wrong spot. Pull down the Goto menu and select LBA Sector. Enter the same numbers as the Sectors per Track you found above. The DOS Boot Record Editor should now be displaying the correct information. The DBR will reveal the correct number of heads and confirm the Sectors per Track that the drive was set up with.

4) You can now fill in the missing parameter, the cylinders, by rearranging the formula outlined above:

Cylinders = Capacity/(Heads x Sectors per Track x 512bytes per sector).

5) Enter the CMOS setup utility and plug in the correct parameters. Reboot the system to and run DrivePro again to confirm the new settings. Look at the drive with DrivePro's editors. If DrivePro is finding the structural areas (in BIOS mode), you know that the correct settings have been applied.

Restoring Drive Structures

Now that you've eliminated the hardware and the BIOS parameters as possible causes to the drive access problem, you can take a closer look at what's on the drive itself.

At this point, you should be able to get the system to recognize that a drive is physically attached to the controller. If this is the case, DrivePro should be able to see and manipulate the contents of the drive.

If you see an error such as this one when attempting to boot:

```
DRIVE NOT READY ERROR
Insert BOOT diskette in A:
Press any key when ready
```

The system is seeing a hard drive where it expects to, but can't boot from it. Another symptom is that the system will simply hang with no error message shortly after power-up. Chances are that damage has been sustained to the boot and other structural areas of the drive or logical volume.

Structural areas include the Master Boot Record (MBR), DOS Boot Records (DBRs), Partition Table(s), and File Allocation Tables (FATs). At first glance, it may seem that the only way to be able to use the drive again is to reformat it, which will destroy all information that it currently holds. If you understand what the structures do and how they are related, you may be able to re-establish access to a drive without destroying the data on it.

Note: *Your first priority is to make the drive accessible so you can back up any data that may be on it.* Making the drive bootable is the second priority, once you know the important data is safe!

At this point, you should try to determine if the drive is still accessible after booting from another source. If you find that it is accessible, do not initiate any actions that may perform a write operation on the drive.

If the drive was set up with non-conventional geometry, such as through EZ-BIOS, then you should be especially careful in attempting to re-establish access. Accessing the drive with the wrong boot parameters can result in permanent destruction of the data on it.

Here's the reason: For systems that do not natively support >528MByte drives, EZ-BIOS sets up the drive with parameters that halve the cylinders and double the heads that the drive reports as the true parameters. If the system is booted without loading the EZ-BIOS code (and therefore the correct parameters for the drive), the system will eventually look for or place data in the wrong part of the disk.

DrivePro allows you to back up and restore the structural areas of a hard drive. If any of the covered information is lost from the system, as long as the drive can still be written to, restoring the structures will return the system to its state at the time of the last save.

Do not restore any obsolete drive structures.

Be sure that you only restore the information you need. *Do not restore any drive structure with old or incorrect information.* Restoring obsolete structures will destroy any changes made to the disk since the last system information backup and can result in data loss.

Typically, the drive structure that causes the most problems is the FAT since it is altered each time a file is created, deleted, or changed. Although there is a secondary FAT that DOS places in the beginning of the volume, it is not very useful for most recovery purposes. This is because the secondary FAT is copied from the primary FAT, meaning that they will probably both contain the same errors.

The directory structure is probably also obsolete because of changes that occurred since the last backup. Any time a file is created, deleted, or saved, the directory is changed.

Because drive structures are related in a specific manner, you should not alter their contents in a random order during a repair/edit procedure. Appendix A contains extensive information about how drive structures are related. Remember to restore or edit drive structures in this order:

1) Partition Table

2) DOS Boot Record

3) FATs

4) Root Directory

Restoring Structural Areas Automatically

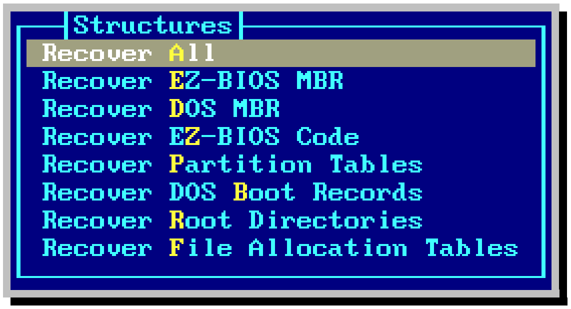
If you previously backed up drive structures:

1) Boot up the system using an emergency diskette (see **page 65**) and run DrivePro.

2) If you have previously backed up the drive structures:

a) Pull down the **Utilities** menu and select **Recover Drive Structures**.

b) Choose the options as desired until you get to the **Structures** dialog.



3) Restore structures only as needed. Use the Recover All selection only if you are confident that all the backup information is valid. If the drive was set up conventionally, choose **Recover DOS MBR**, if was set up using EZ-BIOS, choose **Recover EZ-BIOS/EZ-MBR**.

If you have not backed up drive structures previously, you may be able to restore a generic or EZ-BIOS MBR through DrivePro. It is recommended that you use this feature only after you have exhausted all other resources:

Pull down the **Utilities** menu and select **Recover DOS MBR** (if the drive was set up conventionally) or **Recover EZ-BIOS/EZ-MBR** (if the drive was set up with EZ-BIOS).

Repairing Structures Manually

It is also possible to repair drive structures directly using a sector editor. This operation requires extensive knowledge of drive structures and logical volumes, therefore it should generally be reserved as a last resort, and performed by someone with extensive knowledge of working directly with sectors. If you feel you do not have sufficient experience, try practicing on a spare drive first.

DrivePro contains a sector editor, with dialogs that have been tailored to specific drive structures for ease of use.

Please note that recovering a drive structure successfully does not guarantee that data files will be repaired. The contents of individual clusters in the data area may also be damaged. If you find that a file is corrupt, you must restore it from a previous backup or try to repair it with a sector editor. If the file contains text information, such as a word processing document, you may be able to open it and re-type the corrupted portions. If the file contains binary information, such as a portion of an application, it is probably easier to re-install it using the original setup media. See the section titled “Recovering Files” for more information about data recovery.

Corrupt MBR/Partition Table

Even if the boot code in Track 0 is damaged, you may be able to access the drive after booting from an alternate source. If this is the case, you should back up the contents of the drive before attempting to get the drive bootable again. If the main partition table is also damaged, you will not be able to access the drive even after booting from an alternate source. If this is the case, start Track 0 recovery operations with the main partition table in order to get drive access first. Attempt to fix the boot code only after drive access has been established and the contents backed up.

If you see an error message such as this during boot:

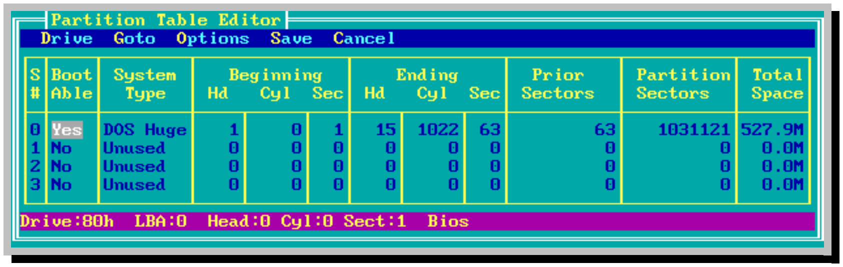
```
NO ROM BASIC
SYSTEM HALTED
```

The drive probably contains an invalid or missing main partition table. This problem is often caused by inappropriate use of the DOS FDISK utility. Provided that the data areas are intact, this problem is usually fairly simple to recover from.

First, make sure that the correct drive parameters are entered in CMOS. Continue by booting from an emergency diskette and run DrivePro.

If you previously backed up the structural areas of the drive, restore the partition tables from the backup as outlined in the section above titled “Restoring Structural Areas Automatically.”

If you did not back up the drive structures previously, you will have to manually edit the partition tables. Begin by pulling down the **Edit** menu and selecting **Partition Table Editor**.



S #	Boot Able	System Type	Beginning Hd	Beginning Cyl	Beginning Sec	Ending Hd	Ending Cyl	Ending Sec	Prior Sectors	Partition Sectors	Total Space
0	Yes	DOS Huge	1	0	1	15	1022	63	63	1031121	527.9M
1	No	Unused	0	0	0	0	0	0	0	0	0.0M
2	No	Unused	0	0	0	0	0	0	0	0	0.0M
3	No	Unused	0	0	0	0	0	0	0	0	0.0M

```
Drive:80h  LBA:0  Head:0 Cyl:0 Sect:1  Bios
```

Enter the partition parameters that the drive was set up with. If you have a simple single-partition setup that occupies the entire capacity of the drive, you only need to edit the contents of Slot 0, all other slots will be unused.

1) Make the Primary Partition bootable by entering “Yes” in the **Bootable** column of Slot 0. All other slots should be marked “No.”

2) Enter the appropriate System Type in all slots. If you’re not sure what to enter, delete what’s there and hit the down arrow. A Primary DOS partition is termed “DOS Huge” by DrivePro. Select the appropriate choice from the pop-up menu.

3) Enter the beginning and ending address (Head, Cylinder, and Sector) for all partitions. The first partition will always begin after Track 0.

If the drive is set up with more than one partition, and you have lost the parameters, you will need to do some investigation to find the correct information. One way to find the beginning of a partition is to look for a pattern of characters that indicate a logical volume boot sector. You can do this quickly by performing a string search from DrivePro’s **Sector Editor**.

You can try to find the end of the first partition by examining the information in the DOS Boot Record (use Direct mode and browse around with Goto until you hit the appropriate starting sector for the DBR). The Total Sectors (Huge) entry will reveal where the partition ends.

4) Enter the Prior Sectors for each partition. This is the total number of sectors on the disk before the partition begins. For example, on a drive set up with 63 sectors per track, the first partition will be preceded by 63 sectors (the number of sectors in the boot track). The next partition will be the sum of Prior Sectors and Partition Sectors in Slot 0, and so on.

5) Enter the total number of sectors in each partition. This number can be calculated by subtracting the beginning address from the end address of each partition. It may be easier to convert the addresses to their LBA values first.

Because inadvertent damage is usually limited to Track 0, repairing a damaged main partition table will probably restore access to all volumes. If the drive was partitioned off into one or more logical drives, you can confirm the integrity of their partition tables by pulling down the Partition Table Editor's Goto menu. Select Slot 1 and inspect the secondary partition table that comes up. Verify the validity of each partition table in the chain this way until you come to the end. See the section in Appendix A titled "Partition Tables" on **page 119** for more information on partition tables.

If you know that the main partition table is valid and the system is still hanging shortly after power-up, the boot code in Track 0 may be corrupt or missing. The boot code can be manually repaired several ways. You can use the command `FDISK /MBR` to refresh the boot code in Track 0.

Another way to repair the boot code is to copy the contents of Track 0 from another drive that was partitioned with the same software. To do this, use DrivePro's Sector Editor to view Track 0 on the working drive. Write down the *exact* hexadecimal code. Now open up DrivePro's Sector Editor to Track 0 on the bad drive and enter the copied code in.

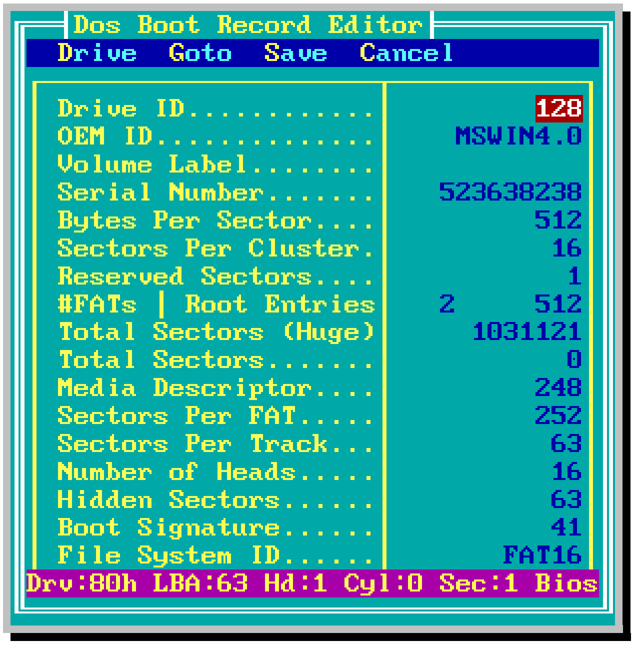
Corrupt Logical Volumes

If you see a message such as this one:

```
            Invalid media type reading drive C
            Abort, Retry, Fail?
```

The drive is getting as far as the first logical volume (C: drive), but cannot continue booting into the operating system once it gets there. Another typical symptom is that the system will hang early in the boot process.

First, determine that the correct BIOS parameters are being used. After that, open the DOS Boot Record Editor and see what it reveals. If the information there looks correct, the problem may be in the directory, FAT, or one of the boot files. If the DBR doesn't make sense, it is either damaged or the BIOS parameters have been set wrong. To eliminate the BIOS parameters as the problem, use the Goto command to browse around sector by sector. You can also look for character patterns of the DBR by using the string search capabilities of the Sector Editor. If you determine that the information in the DBR is corrupt, you can attempt to rebuild it.



```
Dos Boot Record Editor
Drive   Goto   Save   Cancel

Drive ID............       128
OEM ID..............  MSWIN4.0
Volume Label........
Serial Number.......  523638238
Bytes Per Sector....       512
Sectors Per Cluster.        16
Reserved Sectors....         1
#FATs | Root Entries  2    512
Total Sectors (Huge)   1031121
Total Sectors.......         0
Media Descriptor....       248
Sectors Per FAT.....       252
Sectors Per Track...        63
Number of Heads.....        16
Hidden Sectors......        63
Boot Signature......        41
File System ID......     FAT16
Drv:80h LBA:63 Hd:1 Cyl:0 Sec:1 Bios
```

The diagram above illustrates a typical valid DBR. The following passages explain DBR entries in detail:

- Drive ID: this is the identifier of the *physical* drive that the volume exists on. Physical drives are numbered sequentially from 80h (hexadecimal). For instance, the boot drive (C:) will be 80h, the next physical drive will be 81h. DrivePro's DBR Editor represents all figures in decimal notation, therefore you will need to convert into decimal before entering the ID number. For instance, 80h is 128 in decimal.
- OEM ID: This entry identifies the name and version of the operating system the volume belongs to. For instance, a volume formatted under Windows 95 will be identified as MSWIN4.0.
- Volume Label: The name given to the volume when it was high-level formatted. This entry has no affect on drive access, so you can skip it if you like.
- Serial Number: Automatically assigned during the format. This entry should also have no affect on recovery.
- Bytes Per Sector: This value will always be 512.
- Sectors Per Cluster: This value will depend upon the size of the formatted volume. For more information about cluster sizing, see the section in Appendix A titled "Clusters."

- Reserved Sectors: This value is the number of sectors between the DBR and the first FAT in the volume. In other words, one.
- #FATs/Root Entries: These will always be 2 FATs and 512 directory entries.
- Total Sectors (Huge): This value will be the total number of sectors in the volume if the volume is greater than 32MB. If the volume is less than or equal to 32MB, the value is zero.
- Total Sectors: This value will be the total number of sectors in the volume if the volume is less than or equal to 32MB. If the volume is greater than 32MB, then the value is zero. This entry was used on older versions of DOS (pre 4.0).
- Media Descriptor: This value is 248 (decimal) for hard drives. It is the same number that is placed at the beginning of both FAT copies.
- Sectors Per FAT: The number of sectors each FAT in the volume occupies. This number will depend on the size of the volume. One way to figure out this number is by subtracting the LBA address of where the first FAT begins from the LBA address of where the second FAT begins. Remember that the first FAT begins in the sector after the DBR. To find the address of the second FAT, you can try using the Sectors Editor's search function to look for a character sequence matching the beginning of the FAT. For example, a 16-bit FAT will begin with F8 FF FF FF.
- Sectors Per Track: This is the same value as the BIOS SPT parameter.
- Number of Heads: Same as the BIOS Heads parameter.
- Hidden Sectors: The total number of physical sectors on the drive prior to the start of the partition. For the first partition, this equal to the size of Track 0, which is also the BIOS SPT parameter.
- Boot Signature: This value is normally 41 (decimal).
- File System ID: The FAT type the volume was formatted for. Most hard drives are DOS-formatted for a 16-bit FAT (FAT16). Windows 95 can use a 32-bit FAT (FAT-32). Floppy drives use a 12-bit FAT (FAT12).

Note that the DOS Boot Record Editor displays and edits in decimal notation. To use hexadecimal, use the Sector Editor.

If you see this error message:

```
NO SYSTEM
Insert BOOT diskette in A:
Press any key when ready
```

The boot drive is missing system files. First, make absolutely sure that the BIOS parameters, Track 0, and the DBR are correct. If the problem persists, you can try using the DOS SYS command to refresh the system files. Warning: SYSing a volume when a parameters mismatch exists may result in files being permanently overwritten.

Recovering Lost Files

If you are successful in re-establishing access to a problem drive, try to find out the cause of the problem prior to using the drive again. You may only end up repeating the same mistake. Tracking down the source of data corruption is often extremely difficult. If you have been implementing an effective backup strategy, your best bet may be to tear down the drive by partitioning and high-level formatting it again, then restore the data.

What if you've lost access to some files that haven't been backed up yet? If all the file components still physically exist on the drive, you may be able to reconstruct access to them. The success of a file recovery operation will depend on the cause of the problem and whether or not any write actions have occurred in the volume since access was lost.

CHKDSK and SCANDISK

Booting the computer with the emergency disk will allow you to run a diagnostic program, such as CHKDSK and SCANDISK in DOS. These utilities check the directory the directory structure against the FAT. They make sure that all clusters belonging to a file exist and that no FAT entries cross-link file segments. If the FAT and the directory do not agree, the program will automatically attempt to fix the problem, alert the user and ask for direction, or simply report the fact.

These utilities are very useful for locating and cleaning up dead file fragments from applications that terminated early. These file fragments are typically not attached to any directory structure, therefore cannot be deleted by conventional means. If these file fragments do contain lost data, CHKDSK or SCANDISK can recover them by adding them to the directory and giving them file names. The recovered files can then be renamed and opened up by the corresponding application.

Do not allow CHKDSK or SCANDISK to alter the directory if you suspect that structural areas of the drive have sustained damage.

CHKDSK and SCANDISK are useful as diagnostic utilities, but should not be used to repair FATs and directories. *These utilities simply resolve a discrepancy between the FAT and directory by changing the directory to agree with the primary FAT.* If the FAT is corrupt, this will wreck the directory and probably make any further attempt at data recovery impossible. CHKDSK's default mode (no switch specified) is to only check DOS file structures. Running SCANDISK without any parameters allows it to automatically perform repairs; you should instead add the switch /CHECKONLY.

Accidentally Deleted/Overwritten Files

Almost everyone that uses computers on a regular basis has at one time unintentionally deleted a file or overwritten the wrong file. These problems can usually be resolved easily. In order to recover from this type of mistake, the first thing you need to do is refrain from any further operations that will write information to the drive.

Recovering Deleted Files

When a file is "deleted" in DOS (and DOS-based OSes), it is not actually erased. Instead, the file's clusters are designated free in the FAT and the first byte of the file's directory entry is flagged "E5h." E5h, depicted as "σ" in extended ASCII, is used to indicate that the clusters belonging to the directory entry can be overwritten. A file is not actually deleted until the contents of its clusters are replaced.

There are utilities, such as the DOS UNDELETE command, that can restore deleted files, provided that the directory entry still exists and no clusters have been written over. It is important that you do not perform any further save operations because DOS will overwrite the first available directory entry, which may be the one you are trying to recover.

You can also use a sector editor to manually retrieve a deleted file. Start with the directory entry to replace the E5h with the correct first letter of the file. The directory entry will also tell you where the first cluster for the file is located. From there, you can follow the clusters until you come to the end of the file. Replacing the FAT entries to reflect the locations of these clusters will complete the operation.

Recovering Overwritten Files (Saved File With Wrong Name)

When a file is saved, DOS places a new directory entry in the first available slot. If you accidentally save a file with a name previously used by another file in the same directory, the old directory entry may or may not be overwritten. If the old entry *is* the first available slot in the directory, then it will be overwritten. If the old entry is not in that slot, then chances are that it still exists somewhere else in the table with the E5h (σ) flag in the first byte.

Because of the way clusters are allocated by DOS, the new file does not necessarily get written over the clusters belonging to the old file. DOS uses a pointer to indicate where it last wrote information to disk. It will start at the next cluster after the pointer, rather than the first cluster that happens to be marked free in the FAT. Because of this, the chances recovering an accidentally overwritten file are fairly good.

Even if the directory entry no longer exists, it is still possible to recover the file using a sector editor. Each cluster that is marked free in the FAT will need to be examined manually. After all the fragments of the lost file have been located, the FAT can then be altered to reflect their locations. This operation requires extensive knowledge about DOS volumes as well as a high level of attention to detail.

Media Failures/Bad Sectors

Over time, it is possible for manufacturing defects and foreign particles to cause the loss of sectors. When this is the case, you may see a message such as the one below:

```
Cannot read sector
Abort, Retry, Ignore, Fail?
```

If the sectors are part of a program, your best bet is to abort the operation. Run a surface diagnostic utility, such as SCANDISK on the drive and have it mark off any bad sectors in the FAT. If the drive is still healthy overall, re-install the application from the original distribution media.

If the file is some type of document you are working on, and you have no backup, you may still be able to recover most of it. Retry (type “R”) the read operation a number of times. If this doesn’t work, try opening as much of the file you can by ignoring (type “I”) the warning. It is possible that multiple sectors are defective, so try it several times if necessary. Once the file is open, save it somewhere else for later editing. The new file will be corrupted with random characters, but is probably better than starting from scratch. When you are done recovering the file, back up all the files on the drive. Run a surface diagnostic on it and mark off the bad sectors in the FAT.

Early Program Termination

If a program terminates early, such as might happen in a system crash or power outage, chances are that any file(s) it had open were not closed properly. Any work since the last save will probably be lost. However, you can try a few things to recover some of the work.

Some applications create temporary files that the work is done in. When a file is saved, the temporary file is copied over to the permanent name. If a program terminates early, chances are that the temporary file was not closed out properly. This is why \Temp directories tend to accumulate dead files over time. To recover a temporary file, look in the appropriate directory locate recent .tmp (or other extension) files to narrow down the likely candidates. Save these files with the proper extension for the application they correspond to, then open them up. Hopefully, one of these files will contain a significant portion of your lost work.

If a program was terminated in the middle of a save operation, it may still be possible to recover a portion of the file. CHKDSK and SCANDISK can locate file fragments that are not represented in the FAT (see **page 83**). Instead of deleting the "dead" file fragments, you can specify that they be saved out as separate files. Afterwards the recovered file fragments can be renamed with the proper extension, then opened up with the appropriate application to see if the contents match the lost work.

Chapter Eleven
Improving Drive Performance

Simply purchasing a faster drive does not necessarily result in improved performance. The system must mesh together to take advantage of all the features of which the drive is capable. In the case of Enhanced IDE/ATA-2, advanced PIO and DMA modes require additional BIOS support as well as local bus attachment.

Sometimes it is neither necessary nor desirable to invest in new hardware to get improved performance out of your system's hard drive. Some tools are available and waiting for you to take advantage of them at no extra cost.

Caching

Caching is the process where the system loads data from the hard disk to the **RAM** set aside as **cache memory**. As long as the same file is being accessed, the system may refer to the **cache memory** for information instead of going back to the hard disk, thereby increasing the processing speed. Caching does not really increase data throughput, but makes the drive appear faster by eliminating repetitious access.

Read-through caching facilitates read operations by making an educated guess as to which piece of information will be requested next, and holding it in cache memory. When an application wants data from the drive, the cache program intercepts the request and tries to find the data in the cache memory. If the requested data is in the cache memory, a cache **hit** is said to have occurred. If the requested data is not found in the cache memory, a cache **miss** has occurred, and the cache program sends the request to the hard drive. There are different types of caching algorithms that vary in sophistication as to how they anticipate what information will be requested in the near future. System performance can be greatly improved depending on the percentage of cache hits to misses a cache program can produce.

Write-back caching is used to store information on its way to being written to disk. System resources are freed because the cache memory holds the data until the drive is ready to accept it.

Be aware of the possible dangers of using a write-back cache. Depending on the type of cache being used, it could take up to several seconds for the drive to actually perform a write process after being instructed to do so. *If anything inopportune should occur (power outage, system crash) between the time the cache intercepts the data and before the cache can send the data to the drive, the data will be permanently lost.*

Caching is an effective way to improve data transfer performance, but it comes at a cost. The question, once again, is how much performance are you willing to pay for? The data throughput of today's IDE drives is adequate for most, but multi-media applications are making even the casual computer-user look for ways to increase system performance.

There are two types of caches—dedicated **cache memory** and **system memory cache**:

Dedicated cache memory is (usually) the higher performing and subsequently more expensive option. The cache RAM is installed on the controller in this case. Caching controllers are more costly to begin with, and optional RAM additions can drive up the price substantially.

System memory cache serves the same purpose, but does not require additional hardware. A portion of system RAM is set aside as the drive cache. Microsoft calls the caching program it includes with DOS **SmartDrive**. Use **SMARTDRV** in the AUTOEXEC.BAT file to create a cache during the boot sequence. Look in the DOS user's guide or use the command HELP SMARTDRV for more information on the proper syntax.

Data Compression

Data compression is an encoding process where data strings that are repeated often are cut out when written to disk, then replaced before being used by an application. Some programs, such as the popular PKZIP, compress individual files as designated by the user. Others compress a portion of the entire disk drive and require no intervention by the user after the compression program is installed. These programs are the cheapest way to roughly double storage space. DriveSpace comes with latest versions of DOS and Windows.

The cost of increasing capacity in this manner is slower file transfers because of the extra steps involved in compressing and uncompressing data. To maximize drive performance, you may want to store only files you do not use very often in compressed volumes.

Data Transfer Modes

Newer IDE drives support additional data transfer modes that dramatically increase performance. These drives usually require support by the controller and system BIOS in order to take advantage of the new modes. Please see the IDE section in Appendix B for more information on advanced PIO and DMA modes.

Defragmenting Files

If your drive seems to be slowing down as time goes by, you should start by checking to see how "**fragmented**" your drive is. File fragmentation occurs because DOS assigns disk space as it is available. Files do not always fit in the available space. In these cases DOS places pieces of the files to different locations on the disk, which means that the heads must be repositioned more often to retrieve the complete file. As the drive fills up, more files become increasingly fragmented, resulting in a degradation of performance.

The solution is to **defragment** the drive on a regular basis. Newer versions of DOS include a defragmentation program. Simply use the command **DEFRAG** to run the program and choose from among the options presented.

Sleep Modes

One of the current fads in the computer industry is to make hardware more "green," or environmentally friendly, through the addition of energy-saving "**sleep**" modes. Some drives can have this feature enabled through on-board jumpers. There are also utilities available that enable the spindle motor to power down after the program detects a lack of usage, then start it back up on demand. Although commendable in intent, there are some drawbacks to this feature. The drive uses the most power and its components are under the most stress during the power-up sequence because of the torque required to quickly get the disk spinning at operating rpm. The life of the drive may be shortened due to the increase in spin-up sequences. The sleep function has more practical value in secondary drives which see little use, and in laptops to prolong battery life.

Windows 3.1x 32-Bit Access

32-Bit Disk and File Access are Windows 3.1x features that speed up data transfers. **32-Bit Disk Access**, also called **FastDisk**, is a feature which is intended to speed up data transfers. FastDisk is a device driver which effectively replaces the system BIOS for data transfers. By doing so, Windows does not have to switch from protected mode to real mode in order to handle requests by a non-Windows application for disk access. Overall access times appear faster because Windows is not switching back and forth between protected and real modes.

There are potential compatibility problems with using FastDisk. The FastDisk originally bundled with Windows 3.1x will not work with Large IDE drives (see Appendix B). If you encounter this problem, a replacement FastDisk driver that overcomes this compatibility problem is included with DrivePro. Even if you do not wish to use EZ-BIOS, you may still install the Custom FastDisk driver. To do this, enter Windows and run the SETUP program that comes with DrivePro's Disk 1.

32-Bit File Access has more of an effect on overall drive performance because it is intended more for Windows-specific programs. It is actually a caching program that replaces the 16-bit DOS SmartDrive cache and creates a Windows-specific cache. Like 32-Bit Disk Access, it allows file transfers to be accomplished in protected mode. 32-Bit File Access also allows the File Allocation Tables to be altered in protected mode, rather than switching to real mode. Cutting out the extra mode switches speeds up overall system performance.

The default mode for both 32-Bit Disk and File Access is disabled. To enable these functions, go into the Windows Control Panel and double-click on the 386 Enhanced Icon. Choose the Virtual Memory option and subsequently click on the Change button. At the bottom of the box will appear "Use 32-Bit Disk Access" and "Use 32-Bit File Access." Enable both of these and exit the utility.

Some programs conflict with the use of 32-Bit Disk and File Access. If this appears to be a problem, you will have to disable these functions while using the application.

Drive Capacity

This section is limited to a discussion of getting around IDE capacity barriers. There are basically four capacity barriers of current concern:

- 528MByte BIOS/ATA-1 capacity limit
- BIOS cylinder limit
- 2.1GB DOS partition limit
- 8.4GB BIOS capacity limit

Getting Around IDE Capacity Barriers

The 8.4GB barrier will not be discussed here, since there is no solution that doesn't involve redefining the BIOS and/or IDE.

Remember to perform a backup prior to any operation that will change the configuration of the system.

Solutions to the 528MByte limit are well documented, and the barrier no longer exists for systems currently on the market. This problem may still be encountered when attempting to upgrade an older system with a new >528MB drive. If the drive cannot be translated to less than 1,024 cylinders, you must:

- Truncate the drive at 1,024 cylinders in CMOS setup (resulting in the loss of drive capacity).
- Use the >1,024 cylinder support from a third-party software package, such as EZ- Drive or DrivePro.
- Upgrade to a CHS-translating BIOS.

Some of the newer BIOSes can only handle 4,096 cylinders, effectively limiting drive capacity to 2.1GB. As with the 528MByte barrier, there are hardware and software solutions to this problem. If the drive cannot be translated to less than 4,096 cylinders, you will need to:

- Truncate the drive at 4,095 cylinders in CMOS setup (resulting in the loss of drive capacity).
- Use the >4,096 cylinder support from a third-party software package, such as EZ-Drive or DrivePro.
- Upgrade the system BIOS to one that is clearly stated to translate up to 8.4GB BIOS theoretical limit.

See Appendix B for more details on this topic.

2.1GB DOS Partition Limit

Current 16-bit file systems limit the size of the Primary Partition or logical drive to 2.1GB. See Appendix A for details on DOS partition limits. Partition size limits are not so much a problem as they are a hassle. The obvious solution is to back up everything and divide the drive into multiple partitions of less than 2.1GB. This can either be done manually through the DOS FDISK, or more easily through a third-party utility such as DrivePro. Smaller partitions actually have the added benefit of more efficient use of drive space through smaller cluster sizes (see Appendix A). Newer operating systems that use more advanced (32-bit) file systems do not have this limit.

Tips on Maximizing Drive Space

Managing disk capacity effectively will in the long run save you time, money, and effort. A well-organized directory structure makes it easier to find files and selectively perform tasks such as backup, copying, and archiving.

If you find that a drive is running out of space, the first thing you should do is delete any temporary files that may be hanging around. If not cleaned up on a regular basis, the \Windows\Temp directory can accumulate dozens of dead files that waste valuable disk space. Most of these files have been left by applications that terminated early, therefore were not deleted automatically. Creating your own temporary/working directories is a good way of keeping track of draft revisions. Once a document is finished, you can move it over to a permanent directory and safely delete the files in the working directory without fear of losing anything valuable. Otherwise, it's easy to lose track of the draft files. Hunting down these dead files in an effort to free up disk space can be more effort than it's worth.

Remove any applications that you no longer use. If you are using Windows, be sure to use the Uninstall program that came with the applications or remove it from the Control Panel. Simply deleting the directory that the executable resides in will most likely leave dead files and registry entries. Also, archive any files that you do not anticipate using in the near future. You can remove these from the drive altogether onto your backup media for safe storage.

On a lower level, you can get more capacity out of the same drive by creating compressed volumes (see **page 88**). Another method is to divide up the drive into multiple logical drives. Remember that larger logical volumes result in larger clusters and more slack (wasted) space. A good practice when selecting partition/logical drive sizes in FDISK is to choose 1 MB less than the size that will result in the next larger cluster size. See the section titled "Clusters" in Appendix A for more information on this topic.

Appendix A
About Hard Drives

Hard drives are the predominant media used for data storage and retrieval in desktop computers today. A hard drive is a mass-storage device on which data is magnetically written to and read from a rigid platter (disk) spinning at a high rate of speed. The platter is written to/read from by means of a data head that the platter spins underneath. The platter itself is coated with a magnetic medium, much like the film that coats audio tapes.

Over the years, there have been several variations on the disk drive theme. By far the most popular format is the so-called "**Winchester**" drive. The Winchester designation is a commonly accepted industry term that originated from IBM drives that were manufactured in the 1960s. These drives could store 30 **megabytes** of data on removable media and another 30 megabytes on fixed media, leading to the designation "30-30." This also happened to be the caliber of a famous rifle made by the Winchester firearms factory, which came to be used as the popular name of the drive. The term has come to refer to any magnetic data storage device that uses a rigid disk spinning at a high rate of speed. Although the term is beginning to wane in use, any time you hear "Winchester drive," the speaker is referring to a generic hard disk drive.

Hard drives have experienced explosive growth in capability recently due to a combination of factors. Each new jump in **CPU** technology is met with more powerful software, which in turn requires more drive capacity and performance. In spite of this dramatic increase in capacity and performance, modern hard drives work in much the same way as their predecessors did decades ago.

Major Hard Drive Components

Physically, the different types of hard drives available work the same way. They all write to and retrieve data from magnetic media bonded to hard (usually metal) platters that spin at a high rate. The major differences lie in the electronics and instructions that are used to communicate with the host computer, comprehensively known as the **interface**. In fact, many hard drives on the market today are sold in both **SCSI** and **IDE** versions that are mechanically identical.

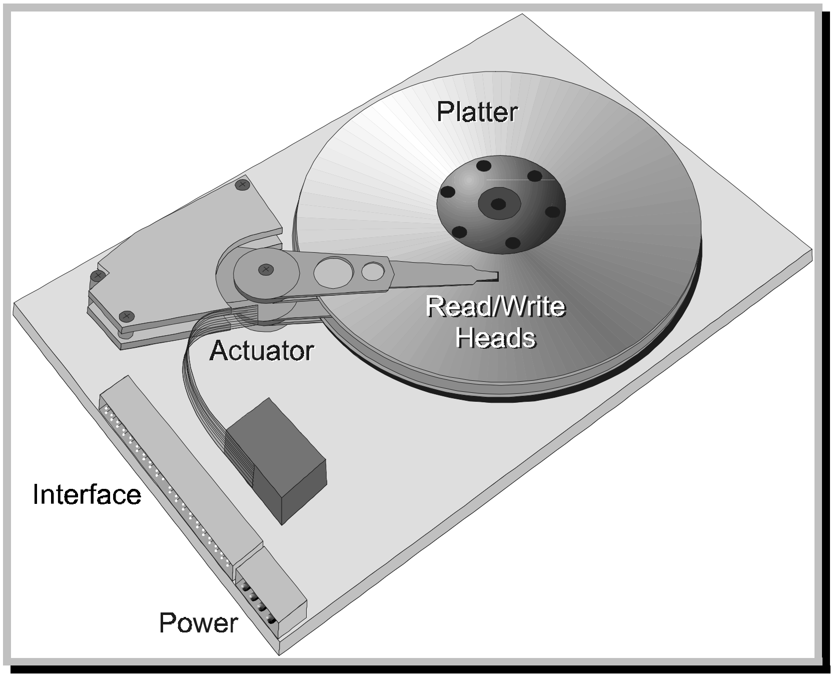


Figure A-1: Disk Drive Basic Physical Layout

Platters

The **platter**, or **disk,** in a hard drive is usually fixed (as opposed to a removable floppy diskette), although several manufacturers offer removable cartridge systems. There are several platters stacked vertically and fixed to the spindle of a high-speed electric motor, called the **spindle motor**. The **substrate** is the material the platters are made from. The substrate must be highly resistant to the stress imposed by high rotation speeds and temperature. It is usually metal, although glass and ceramic have been used.

Because the hard drive’s metallic platter is much more resistant to change than the plastic disk of a floppy drive, it can be subjected to higher stress with more reliable results. This means that the disk on a hard drive can be spun at speeds of 5,000 rpm and greater, while traditional 1.2MB and 1.44MB floppy drives are limited to around 300 rpm. Faster rotation speeds allow faster data **access times** and **throughput**. The access time is the amount of time it takes the drive to locate information (usually measured in thousandths of a second), and throughput is the amount of information the drive is capable of transferring over a period of time (usually measured in millions of bits per second).

The substrate is thinly coated or plated on both sides with a magnetic substance (iron oxide or other thin-film metal media) for non-volatile data storage. Surfaces of disks are usually lubricated to minimize wear during drive start-up or power-down. Plated media are also extremely hard and much less susceptible to damage from head crashes (the head coming into contact with the surface of the platter). A plating will hold more than double the magnetic particles of a coating in the same amount of space, allowing greater storage capacity and throughput. This brings up another important parameter used to describe hard drive capability: **areal density**.

Areal Density

Areal density describes the physical amount of data that can be held on an area of the disk, measured in bits per inch^2. Areal density is obtained by multiplying the **bit density** (bits per inch) by **track density** (tracks per inch), yielding the bits per square inch on the disk surface. See the section titled "Physical Drive Organization" in this for more on tracks and other elements of drive organization. Bit density is measured radially around a track, and track density is measured linearly across the face of the disk (see Figure A-2).

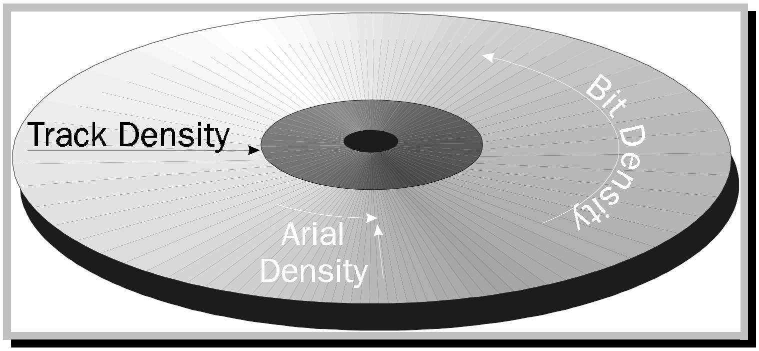


Figure A-2: Bit Density vs. Track Density

Data Heads

Data is written to or read from the platter by means of **data heads** that sense the magnetic fluctuations on the media spinning beneath it. These fluctuations are interpreted as the 1s and 0s of the **binary** data format that a computer understands.

When a platter begins to spin, a cushion of air only a few microns thick forms underneath the head. This is the reason that hard drives are factory sealed and basically have to be replaced in the event of a physical media failure. The tolerances are so tight that even microscopic damage to the disk surface can permanently degrade or destroy the drive. Foreign particles, such as dust, invading the drive assembly can be disastrous.

When looking at the capabilities of different types of data heads, the most important specification is the areal density of the magnetic media that the heads are capable of reading from or writing to. Two types of data heads are in common use today: **inductive**, and **magnetoresistive** (**MR**).

Inductive heads have been the mainstay of the industry since hard drives were invented. A **monolithic** inductive head is basically an electromagnet, that is, a ferrous core wound with current carrying wire. The core is shaped in order to bring the poles together, separated by a small gap (see Figure A-3).

When the head is used for writing data, an electrical current is sent through the wire, producing a magnetic field in the gap. The magnetic flux extends from the gap and is imparted to an area of the recording medium along the track the head is over.

The same head is used to read data by inducing a current in the coils when the gap passes over a change in the magnetic flux over the recording medium. The drive control circuitry controls the switch between read and write operations.

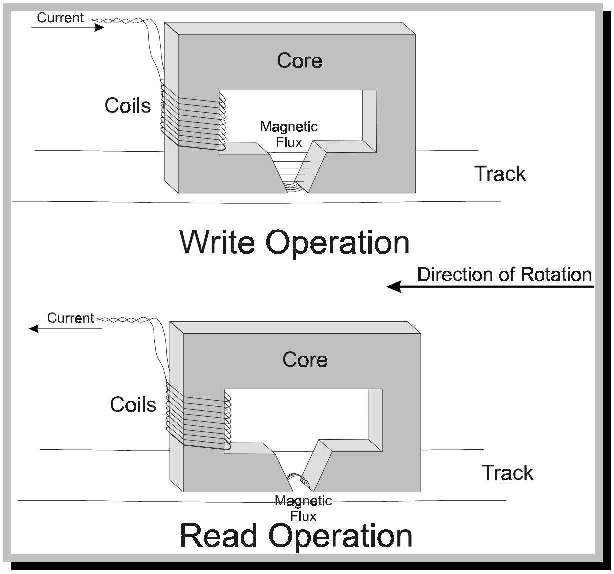


Figure A-3: Monolithic Inductive Head

Over time, the constantly increasing areal density of the recording media used in hard drives has required corresponding improvements in head technology. Higher areal density means that the individual magnetic fluxes are smaller and therefore weaker. This means that read heads must be more sensitive, and write heads must be capable of producing smaller, more accurate fields. Inductive heads have progressed since the days when they were simply U-shaped electromagnets.

Technological improvements have allowed inductive heads to become smaller and more sensitive. **Composite** heads use the heavier ferrous material only where needed, with the bulk of the structure made up of a lighter, non-magnetic substance. **Thin-film** heads are the last in the line that use simple inductive technology. Thin-film heads use the same micro-circuitry technology used to manufacture chips such as CPUs. The advantage is small size and light weight, allowing greater accuracy in read/write operations and head placement.

Simply making inductive heads smaller and lighter is effective only up to a point. Inductive single-head technology puts faster data access and greater capacity at odds because of conflicting physical requirements for read and write operations.

In order to boost the sensitivity of read operations, it is possible to increase the number of turns in the coil, or rotation speed of the disk. Both methods increase the inductance of the current passing through the coil, hindering high-speed write operations. A compromise between coil turns and disk rotation speed is necessary, but limits performance.

Magnetoresistive (**MR**) heads use a material that changes electrical resistance in the presence of a magnetic field. A small direct current is constantly passed through the MR element. When the element passes over a magnetic field, a voltage change occurs in proportion to the change in the MR element’s resistance. The change in resistance depends on the direction of the magnetic field the MR element encounters. The resulting change in voltage is interpreted as a bit 1 or bit 0. This is a simplistic explanation for MR head operation, but illustrates the basic concept.

MR heads are extremely sensitive, but can only be used for read operations. Thin-film inductive heads are still used for write operations. Since read and write operations use separate elements, no compromises have to be made regarding these components.

Table A-1 outlines the increasing performance of the different types of heads:

Table A-1: Typical Read Head Type Capacities

Head Type	Areal Density (BPI x TPI)
Monolithic Inductive	3,600,000bits/in^2
Composite Inductive	12,000,000bits/in^2
Thin-Film Inductive	500,000,000bits/in^2
Magneto-resistive	1,000,000,000,000/in^2 +

There are usually several read/write heads in a hard drive. On larger drives, a single head and platter surface may be reserved for **servo** information. The servo contains information for feedback head positioning systems (see the next section). When looking at manufacturer's drive specifications, the term *heads* usually only applies to actual data heads—those that read or write the data usable by the computer. Such is the case with the drive listings in this guide. There may be eight platter surfaces but only seven data heads. Only one of the several read/write heads is in use at any one time. The controller can only handle data from one head at a time. This in itself does not result in any data transfer delays, as the other elements involved in the transfer of data in the system can only take in so much data at one time.

Actuator

The **actuator** is a device that provides head positioning over the platter surfaces (see Figure A-1). The actuator anchors the **carriage assembly**, which in turn holds the read/write heads. Two kinds of actuators have been commonly used, **stepper motor actuators** and **voice-coil actuators**.

A stepper motor actuator uses a motor that moves the actuator arm in discrete steps. The heads are positioned by rotating the motor a precise number of steps and then converting these steps into linear motion. This means that misalignments can eventually occur due to overheating or mechanical wear. Most older hard drives, especially inexpensive ones, use this kind of actuator. Floppy drives also use this kind of actuator.

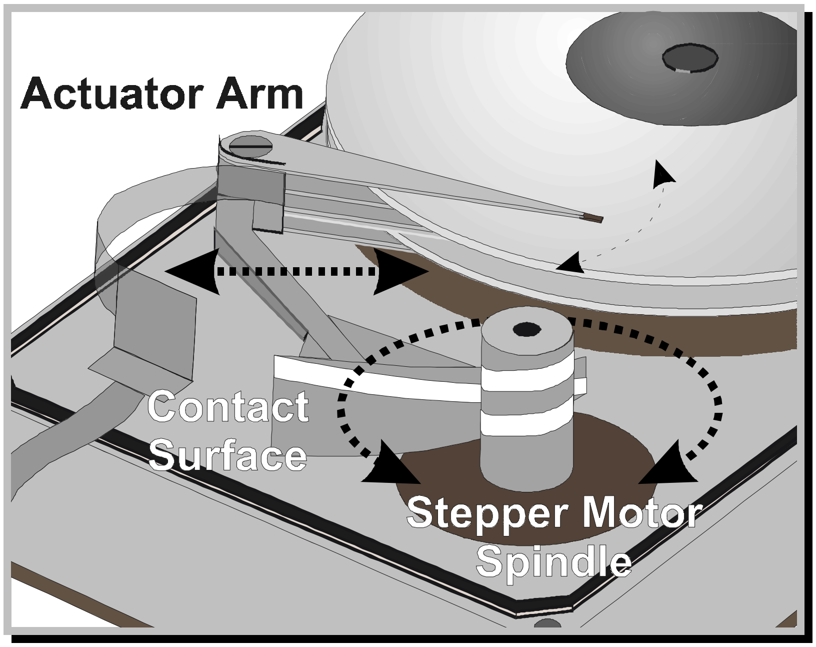


Figure A-4: Stepper Motor Actuator

All modern hard drives use the voice-coil actuatorbecause it is the faster and more reliable of the two types. A voice-coil actuator uses a solenoid (a magnet that pulls on a metal rod) to pull the heads toward the center of the platter. The heads are placed on a hinge mechanism with a spring that pulls in the other direction. The amount that the actuator moves the carriage assembly is regulated by the amount of current sent through the voice coil motor. When the solenoid is released, the heads get pulled back to the outer edge of the platters. Powering off the drive results in the actuator automatically “parking” the data heads. The term “voice-coil” is used because speakers operate in much the same way–using an inductive voice coil and permanent magnet to displace the sound-producing cone. Voice-coil actuators are much faster than stepper actuators because they don’t have to move in incremental steps–they go directly to their destination. They are also more accurate because actuator arm positions are not mechanically preset and therefore less subject to misalignment. Positioning information (servo, see below) is located on the platters themselves and is used for constant realignment of the heads.

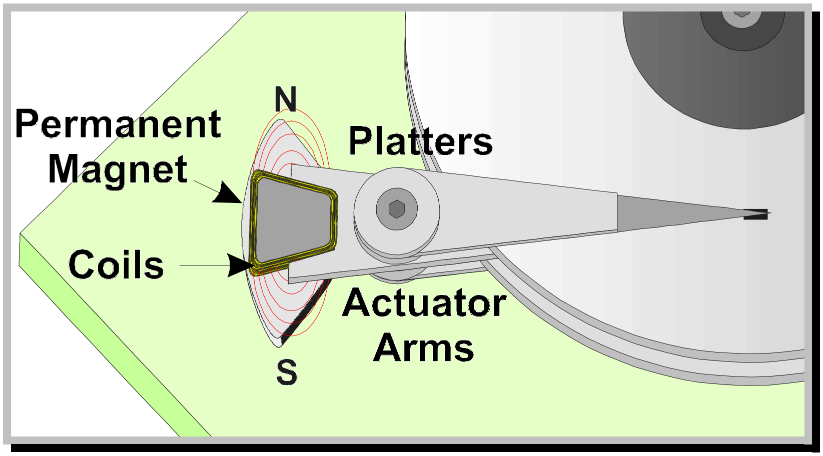


Figure A-5: Voice-Coil Actuator

To give you an idea of the speed difference between a voice-coil and stepper motor, consider that a drive using the stepper motor takes from 65 to 100ms (65 to 100 thousandths of a second) to move from one track to another, as compared to the voice-coil actuator that typically requires 10 to 40ms–more than twice the speed of the stepper motor.

Head positioning in a voice-coil actuator drive is provided by a feedback system known as the **servo**. Servo information is pre-recorded position reference data that is located on a disk. The drive uses this information and compares it against the actual head position, then makes corrections as necessary to place the head at the correct track. There are two general types of servo systems: **dedicated servo head/disk** and **embedded servo data.**

Some of the larger capacity drives use an entire platter surface (dedicated servo) to store this information. This is why you will see a drive that has only seven data heads but four platters (there are usually two heads for every platter). The eighth disk surface is reserved for the servo platter. Hard drives with an odd head count will have a dedicated servo. **Embedded servo** data does not require an additional disk and head, instead using added servo data on every cylinder. The voice-coil actuator knows what track it's on by reading positioning information (servo) that has been permanently placed between the tracks.

Control Circuits

Drive circuitry is lumped into one category for the purpose of simplifying the illustration of the basic hard drive. Most of the electronics exist on the printed circuit board (PCB) on the underside of the drive. Typical circuits that exist on the drive involve coding and decoding the signals being sent to/received by the heads, actuator positioning, and spindle motor control. “Intelligent” drives, such as SCSI and IDE, contain onboard control circuitry, while “dumb” (ST-506/412) drives rely on control circuitry located on an intermediary adapter.

Interface

The term **interface** refers collectively to both the hardware used to make the physical connection between the drive and the computer, and the electronic instructions that manage the transfer of data. The two hard drive interface types most commonly used in the microcomputer environment are IDE and SCSI. ST-506/412 and ESDI are two formerly popular interface standards that are no longer supported.

The ST-506/412 was the first to be developed and was made obsolete by the performance limitations of its design. Even when upgraded with improved RLL recording technology (see page 103), its maximum data transfer rate was only 7.5 **Mb/s** (**Megabits per second**), far short of that now achieved by current IDE and SCSI models. ESDI, a high-performance version of ST-506/412, was much faster, reaching speeds of up to 24 Mb/s. ESDI drives were also considered quite reliable and available with as much as a **gigabyte** of storage capacity. These factors made it the prime choice for a network drive prior to the advanced development of SCSI. Unfortunately, for those with an interest in ESDI, the full benefits of the ESDI interface were never realized and it was eventually surpassed by SCSI in flexibility and performance.

The remaining types, SCSI and IDE, account for the vast majority of drives sold today. In their present forms, SCSI drives are usually seen in high-end applications such as network servers, while IDE drives are the Original Equipment Manufacturer’s (OEM) choice for systems intended for the popular market, such as home computers and work stations. Appendix B provides detailed technical information on these two interfaces.

There are other hard drive interfaces that are not covered in this text. Those interfaces are usually found in minicomputers and mainframe systems. Most are designed for non-PC bus types such as VMEbus, S-Bus, STD Bus, and Multibus. **SMD** (Storage Modular Device) and **IPI** (Intelligent Peripheral Interface) are two examples of common interfaces that are used in the mini and mainframe environments. **PC-Card** drives are small **form-factor** IDE drives that use the **PCMCIA** (**Personal Computer Memory Card International Association**) bus. These are **Plug and Play** (**PnP**) devices that are designed to be used in the laptop computer environment.

Buffer

All modern hard drives have some amount of on-board memory, which is termed the **buffer**. The buffer is a way-station for requested data after it is read from a location on disk. On older drives, a **Track Buffer** was used to allow the drive to operate at an optimum level without having to a wait for a relatively slow (8088/8086 CPU) system to process each read/write sequence to or from memory. Rather than having to wait for the first track to be processed by the host system before moving on to the next, the drive put the contents of the first track into the track buffer and moved on. See the description of **interleaving** on page 105 for more information on how system speed can affect drive performance.

Newer systems are fast enough that even high-performance drives can sometimes seem inadequate. In this case, the advantage of a disk buffer is that it decreases system delays due to the physical limits of the drive speed. Read sequences can be sped up by having the buffer hold information that it anticipates the system will request. There are varying levels of sophistication in the methods buffers use to predict what data will be requested. A **Lookahead Buffer** simply loads requested data and some of the following information located sequentially on disk. A **Segmented Lookahead Buffer** is the same thing, except that it can hold several read sequences simultaneously to avoid repetitious read operations.

Reading/Recording Technology

The technology used to interpret (encode/decode) the flux reversals is just as important as the physical capabilities of the magnetic media and the data heads.

Analog Read Channel

Most drives still use a simple analog peak detection scheme to pick out digital information from the analog signal received from the heads. The peak detection scheme runs into problems when combined with newer media and head technologies that enable high areal densities. In such cases, the individual magnetic fluxes are so close together that some overlap occurs. A less sophisticated read circuit that simply looks for individual analog peaks may experience errors due to this overlap.

A **Partial Response Maximum Likelihood** (**PRML**) circuit can overcome signal overlap and background noise by sampling the signal at several locations along the curve, then projecting the shape to more accurately determine the location of the peaks.

Encoding

Encoding is the method used to convert the **binary** information (0s and 1s) used by the computer into the flux reversals (or transitions) that can be stored and read from magnetic media. As drive technology improves, newer encoding schemes are being employed to allow data to be packed more densely on the disk.

Frequency Modulation (**FM**) encoding is an outdated encoding scheme that is no longer in use. It used up to half of the disk space with timing signals for the encoding process.

FM encoding was replaced by **Modified Frequency Modulation** (**MFM**) encoding. As the name implies, MFM is a refinement of FM. MFM does away with the need for timing signals, allowing it to pack twice as much data as FM encoding on the same drive. MFM does this by using a fixed length encoding scheme. Flux reversals on the disk will always be evenly spaced in time so that the beginning of one bit can be separated from another. This type of scheme also allows even single bit errors to be detected easily and corrected by the controller electronics, where the encoding/decoding process is performed. MFM encoding was replaced by RLL encoding, although MFM is still in use for floppy drives.

FM and MFM use a 1:1 relationship between flux reversals and the bits they represent. This simplifies encoding/decoding logic, but makes the drive more susceptible to the physical limits of the media and electronics. The drive circuitry needs time to react to each flux reversal–the more flux reversals, the slower the performance of the drive. This also means that each flux reversal needs more space to be detected properly. A better way is to maximize the space between flux reversals.

Run Length Limited (**RLL**) is a type of encoding scheme that requires fewer flux reversals for a given amount of data as well as reducing the amount of data-checking information taking up disk space. The logic used is far more complicated than that used in MFM, but allows much more data to be placed on the disk. RLL encoding represents data in bytes instead of individual bits. Bytes are in turn represented by 256 selected 16-bit combinations. These 16-bit combinations are chosen for having the maximum amount of space between 0s and 1s. The *run-length* in RLL refers to the number of consecutive 0s in a 16-bit combination prior to the appearance of a 1. The extra spacing between flux reversals gives the read/write circuitry more time to perform an operation. RLL uses twice as many bits to represent a byte of data as MFM, but the efficient spacing of flux reversals more than makes up for the extra bits by allowing the use of smaller magnetic domains.

There are several types of RLL encoding. In RLL 2,7 the run-length of 0s in each 16-bit code is at least two and limited to a maximum of seven. This allows for a fifty percent increase in disk space over MFM encoding. RLL 3,9 (commonly called Advanced RLL, or ARLL) is also available, which further increases disk space (up to 100% from MFM).

Partial Response Maximum Likelihood (**PRML**) is not really an encoding scheme, but an advanced type of analog read channel used to reliably pick out the weaker signals that result from higher areal densities. The use of a PRML read circuit enables the use of a more efficient RLL encoding scheme that allows consecutive 1s to exist in the bit sequence. For example, PRML/RLL 0,4,4 encoding means that 1 bits can occur consecutively, up to four 0 bits can exist between 1 bits. The last 4 is used to limit the number of 0s between 1s for some specific sequences.

Where the encoding takes place also has an effect on drive performance. On ST-506/412 drives, the encoding/decoding process was performed on the controller card, instead of at the drive itself. ESDI drives reversed this state because of the drawbacks of having a raw signal becoming degraded over the length of the cable from the drive to the controller. All modern hard drives (IDE and SCSI) perform the encoding/decoding process on the drive instead of the controller. In addition to the benefits stated above, host adapters are less expensive and can have more features packed in the space not taken up by drive-controller circuits.

Sector Interleave

In the past, most hard drives could transfer data faster than the computer could process it. **Interleaving** sectors alleviated this problem by forcing the drive to read data a little slower. Since data heads are skipping one or more sectors that rotate underneath, the drive's throughput is effectively lowered.

For example, Figure A-6 shows sectors that are non-interleaved on top, while the diagram on the bottom has sectors interleaved at 1:3. If the interleave is 1:3 then the sectors are renumbered: **1**-7-13-**2**-8-14-**3**-9-15-**4**-10-16-**5**-11-17-**6**-12. This allows the CPU to store the data for sector #1 while #7 and #13 are under the head, and then continue with sector #2 when it's ready. In one revolution, approximately six sectors of 512 bytes will have been read and stored by the CPU. In three revolutions, all of the sectors will have been read and stored. Most 8088/8086-based systems use an interleave of three or four. Most 80286-based systems use an interleave of 1 or 2. The interleave required depends on several factors, including the speed of the interface, CPU, and the controller electronics.

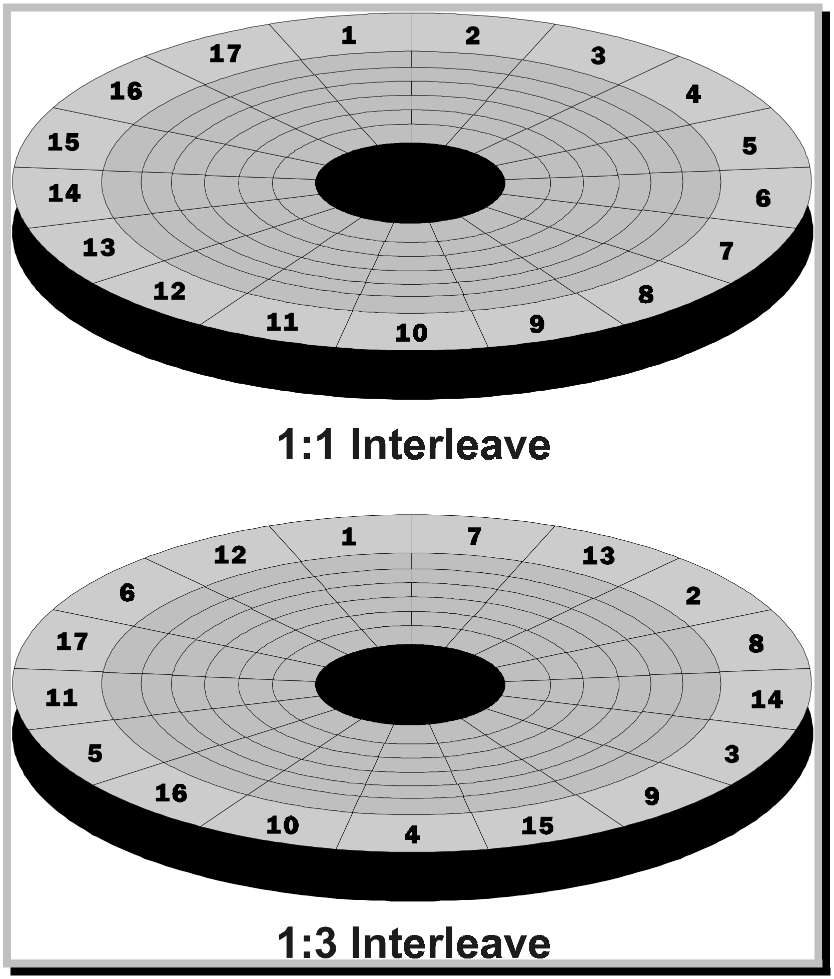
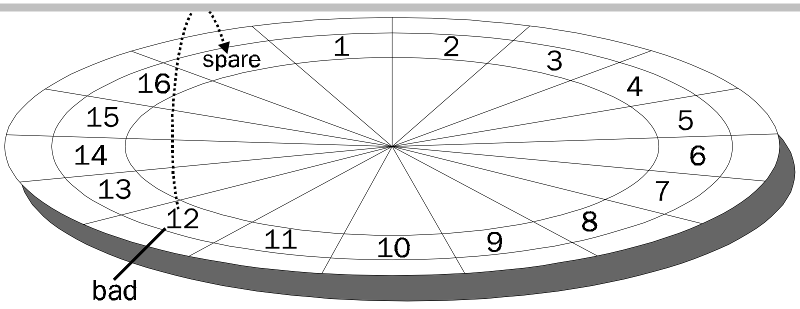


Figure A-6: Interleaving

The computer systems on the market today can easily accommodate the throughput, or data transfer rate, of even the fastest drives. In addition, all drives of recent manufacture use interfaces that can handle data throughput at rates greater than the mechanical part of the drive is capable of. For these reasons, no interleaving is used for current drives. Another way of saying this is that the interleave of modern hard drives is 1:1.

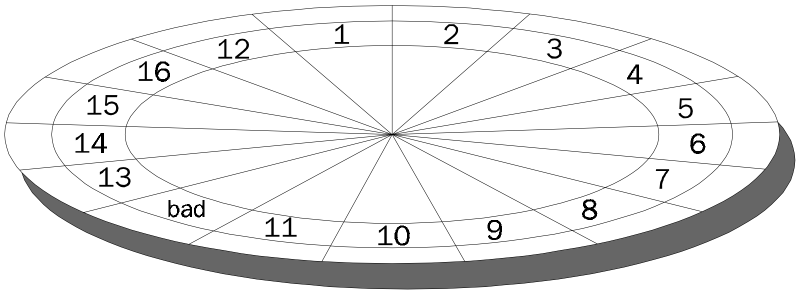
Sector Sparing

Sector sparing is a method used on most newer drives to mask the presence of defective sectors on a disk. Sector sparing effectively reduces the number of usable sectors on each track by one and re-allocates the spare sector for substitution in place of a defective sector. The system will see fewer defects because only the drive is aware of the spare sectors. Sector sparing reduces the total capacity of a drive but is useful if the drive has a relatively large amount of defects and you run applications that require a defect-free drive.



Sector Sparing For A 36-Sector Drive

Operating system sees 35 good sectors and no bad sectors



Same Drive Without Sector Sparing

Operating system sees 35 good sectors and 1 bad sector

Figure A-7: Sector Sparing

Head and Track Sector Skewing

Consider that hard drive disks spin at thousands of rpms. After completing the read/write of a track, the head must reposition itself to start operating on an adjacent track. The head must then make sure it is on the correct track by identifying the first logical sector of that next track. If the first logical sector of every track on the platter is lined up radially along a disk, then the head probably will not be able to re-position itself quickly enough to catch the first sector of the new track after finishing the last sector of the previous track. The head must then allow the disk to spin under it almost a complete revolution before starting with the first sector of the new track. If this is occurring, the drive is not operating optimally.

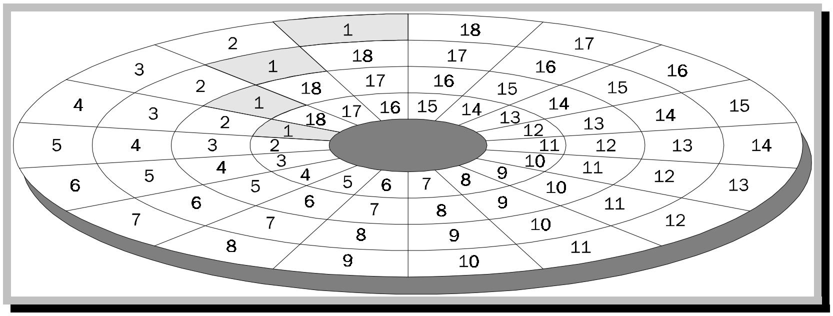


Figure A-8: Head and Sector Skewing

Sector skewing is the repositioning of the first logical sector on each track to allow the data heads time to switch tracks before coming upon the first sector of the new track. The head does not have to wait an entire revolution after jumping to an adjacent track because there is an adequate amount of space between the last sector of the previous track and the first sector of the new track. A skew factor is added when formatting a disk to improve performance by reducing wasted disk revolutions. Because most newer drives are low-leveled at the factory, this is not usually necessary. For those wishing to change the skew factor on older drives, sector skew values must be obtained from the drive manufacturer.

No matter how fast the actuator assembly may be, it takes more time to move the heads physically as opposed to switching operation from one head to another. **Head skew** is the term applied to the organization of tracks among all of the disks in a drive that allows the drive to switch read/write heads in order to minimize actuator movement. For example, a drive locates requested information on the first track of the upper surface of the top disk. After finishing the first track, it switches to the head located on the bottom surface of the same disk and begins reading the next track. The drive goes through all of the heads until it has to switch back to the first head, finally being forced to move the actuator assembly. If the tracks are arranged optimally, the drive will read all of the requested information with no wasted revolutions. If the tracks are not arranged optimally, during one of the head switches, the head will not catch the first sector of the track under it on the first pass and another revolution will be required. The tracks on adjacent surfaces can be offset to ensure that after a switch, the next head assembly catches the first sector of the track it is over. Head skew can be set to 0 if the drive does not support it.

Reduce Write Current

On some older drives, the **Reduced Write Current** (**RWC**) is a signal input that decreases the amplitude of the write current at the actual drive head. Normally, this signal is specified to be used during inner track write operations to lessen the effect of adjacent bit "crowding." When installing a drive in a system, the RWC number requested by the CMOS setup utility is the first track number to begin the area of RWC. That track and all subsequent tracks will be written with RWC. This parameter is ignored for setting up modern drive types.

Write Precompensation

Write precompensation (**WPC**) is a method used on older drives to compensate for the increased density of data on tracks closer to the spindle. Because tracks on the inside of the disk have to hold the same amount of data as the outer tracks, the data must be packed more densely on the inner tracks. Depending on factors like encoding method and areal density of the magnetic media, increasing data density results in some drawbacks.

Packing more signals in less space means that weaker flux reversals must be used, which can result in read/write heads not being able to perform properly. In some cases, WPC means that the inner tracks are written with stronger magnetic fields to provide clear signals.

Bit-shifting is another side-effect of packing magnetic fields more densely on the inner tracks. Magnetic fields that are placed in close proximity can affect each other. Bit-shifting can cause data heads to misinterpret the exact position of individual fluxes.

WPC is also the term applied to varying the timing of the head current from outer tracks to the inner tracks of a disk. This process makes up for the bit-shifting that occurs on the inner cylinders that pack more data into a smaller area.

Although you will sometimes still see a place to enter a WPC value in the CMOS set-up, modern drives do not need it. For this reason, this value should usually be set to 0. WPC is only required on large form-factor drives using oxide magnetic media.

Landing Zone

This is the section of the disk that is designed as the safe zone for head parking after power is removed from the drive, usually the highest cylinder (closest to the spindle) on the drive. Depending on the actuator, the head parking procedure can be an active or passive process.

If the actuator is a stepper motor type, when the power is turned off the drive's heads will remain over the track where the last read/write action was performed. If the drive is jolted, data under the heads can be lost. The stepper motor drive heads must be parked through software that places the head at a safe (for data) landing zone. Some later model stepper actuator drives used the residual energy of the rotating disks to move the heads to the landing zone after power-down.

Voice-coil drives automatically park themselves due to the design of the solenoid-spring mechanism. Head parking software is not needed on these drives and should not be used.

Physical Drive Organization

The physical organization of a hard disk is fairly straight-forward. The basic units of physical organization are **sectors**, **tracks**, **heads**, and **cylinders**.

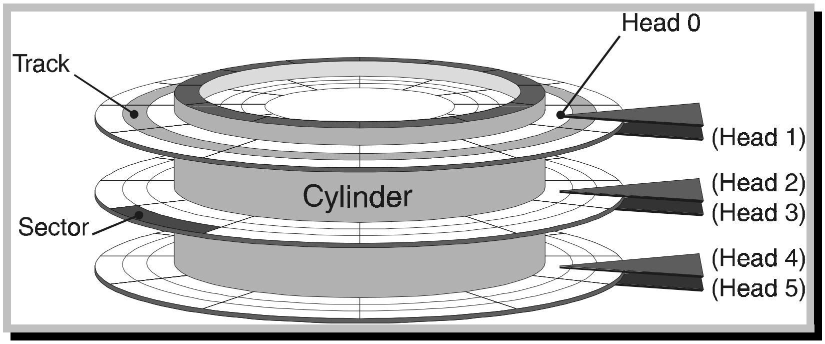


Figure A-9: Disk Physical Organization

Tracks

The concentric circles that hold data on a disk platter are called **tracks**. The number of tracks on a platter (**track density**) depends on the individual hard drive. Tracks in turn are divided into sectors. Modern hard drives maximize capacity by varying the number of sectors per track, increasing the sector count on the outer tracks (see below).

Sectors

A sector is the basic unit of physical storage for hard drives. Sectors are usually 512 bytes in length. Each sector is defined with magnetic markings and an identification number (this identification number is contained in the sector header). All sectors contain a **sector header**, (usually) 512 bytes of data, and an **Error Correction Code** (**ECC**) (See **Figure A-10**).

The sector header is the address portion of a sector. The sector header (ID field) is written during the format operation. It includes the cylinder, head, and sector number of the current sector. When a read or write operation is requested, the disk controller compares this address information with the desired head, cylinder, and sector number before an operation is allowed.

The **Cyclical Redundancy Check** (**CRC**) value is placed at the end of the sector header and is used by the drive to verify sector address integrity. In a typical scheme, 2 CRC bytes are written to each sector. After the drive finds the address it is instructed to look for, it calculates a CRC value that it checks against the CRC value in the sector. The two CRC values should match, meaning the drive is looking in the correct place for the requested data. If the values do not match, an error code is returned, causing drive access to halt.

The **Error Correction Code** (**ECC**) is placed after the data portion of the sector, and is used to verify that the data being read is valid. The ECC hardware on board the controller that serves as an interface between the drive and system can typically correct a single burst error of 11 bits or less. This maximum error burst correction length is a function of the controller. With some controllers (not SCSI or IDE) the user is allowed to the select this length. The most common selection is 11 bits.

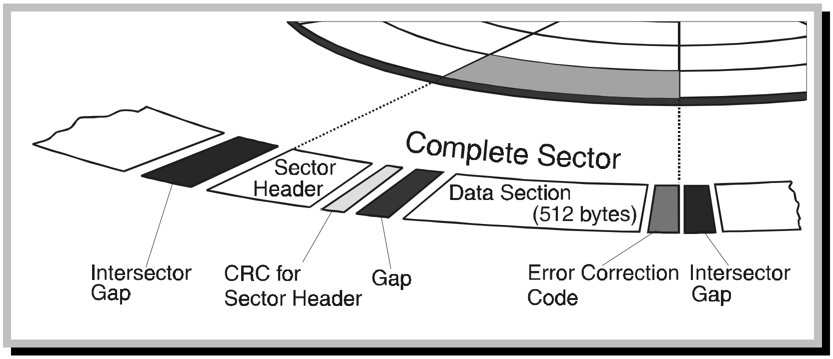


Figure A-10: Sector Detail

On many drives, all the tracks have the same amount of sectors even though the tracks are much larger near the outside of the platter than the inside. This arrangement simplifies the organization of the sectors on the drive, but has the draw-back of giving up potential disk space. More advanced recording methods have been introduced in which tracks on the outside have more sectors per track than those on the inside. Individual manufacturers typically have their own marketing names for this technology, but it is usually referred to as **multiple zone recording**. Note that each sector still contains 512 bytes of data.

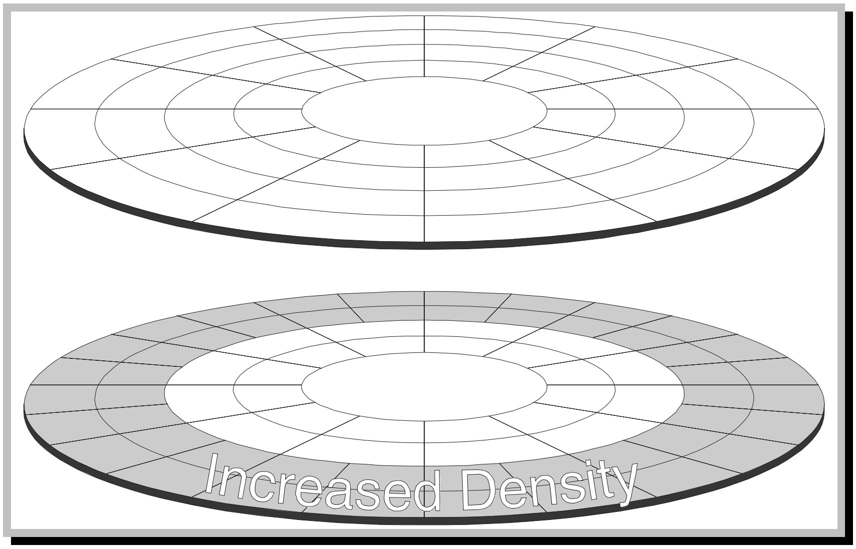


Figure A-11: Multiple Zone Recording

Cylinders

The largest unit of organization is the **cylinder**. A cylinder is a surface formed by the same track number on vertically stacked disks. At any location of the head positioning arm, all the tracks at that position compose that particular cylinder. Operating systems typically write to an entire cylinder before repositioning the actuator over the next one. This minimizes head movement, thus speeding up read/write operations.

Diagnostic Cylinder

The last cylinder (closest to the spindle) is called the diagnostic cylinder. The diagnostic cylinder is normally reserved by the manufacturer for test purposes. DOS is unable to view this cylinder, so it is basically invisible to most applications. Utilities that can directly access the hardware, such as EZ-Drive, make use of this space by storing backup information in this area.

Heads

Data heads can be considered a unit of organization because platters can contain data on both sides. Most manufacturers do not include the number of platters in their specifications, instead listing the number of data heads. Note that some drives dedicate an entire side of a platter to servo (head positioning) information, resulting in an odd number of data heads. See the section in this appendix titled “Actuators” for more information on servos.

Addressing

The organizational scheme used to locate information on a physical area of the drive is known as addressing.

Buildings are usually numbered according to some system in order to help people find them. Sectors on a hard drive must also use a system so information can be found. The organizational scheme used to locate information on a physical area of a disk is known as **addressing**. There are two major methods of data addressing: **CHS** (**Cylinders, Heads, Sectors**) and **LBA** (**Logical Block Addressing**).

CHS is used on most older IDE, ST-506/412, and ESDI drives. Information is accessed according to the three-dimensional (physical) location (cylinder, head, and sector) it falls under. The BIOS (see page 115) and lower levels of operating systems use this format to communicate with hard drives.

LBA is used by SCSI and newer IDE drives (see Appendix B). This addressing scheme simply assigns each sector in the drive a logical address or sequential one-dimensional number, starting with the number 0. LBA is a desirable addressing scheme because it is the format used by software applications and higher levels of the operating system; it is simpler to perform calculations with LBA than with the three-dimensional CHS format. LBA addressing requires a host system BIOS which supports it or a controller that bypasses the host system BIOS with a replacement block device driver or its own on-board BIOS ROM.

See the Technical Information sections for IDE and SCSI in Appendix B for examples which may aid you in understanding CHS and LBA.

The Data Transfer Chain

There are several steps involved in extracting data from its physical location on the disk to placing it in the host memory (and vice-versa). All of the data handling elements in this chain represent potential bottlenecks to overall performance. Simply installing a bigger, faster drive may not give you the results you expected. This section presents a simplification of the process for those wishing to become more familiar with the way a hard drive works.

There are five major elements in the data transfer chain:

- Mass storage device (disk drive)
- Device controller
- Basic Input/Output System (BIOS)
- Operating System (OS)
- Host memory (RAM).

Each of these elements is detailed in the sections below.

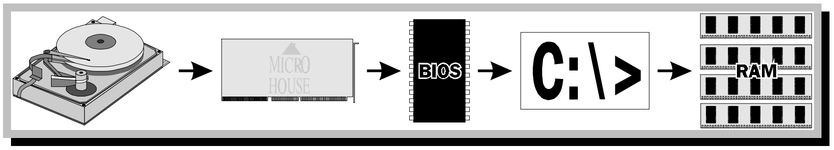


Figure A-12: Data Transfer Chain

Controller/Host Adapter

The controller, or host adapter, is the interpreter between the host computer and the hard drive. The level of actual control the adapter has over a drive depends on the interface. SCSI and IDE host adapters do not perform drive control and encoding/decoding functions, as these are accomplished by circuits on the drive itself. These types of adapters are sometimes termed "dumb" controllers. ST-506/412 controllers are "intelligent" in that they must tell the drive where to find a read/write location, as well as perform the encoding and decoding of raw signals into data.

Controllers can exist as expansion cards that plug into one of the bus slots on the computer's main board, or can be integrated directly into the main board itself.

BIOS

The Basic Input/Output System **(**BIOS**) is the interpreter that sets up a specific computer to allow the operating system (such as DOS) to communicate with peripherals.**

The **Basic Input/Output System** (**BIOS**) is the interpreter that sets up a specific computer to allow low-level software (like the operating system) to communicate with peripherals. The BIOS can be thought of as the "glue" that holds together system hardware that may be made by a variety of manufacturers. The BIOS is firmware, that is, the instructions are permanently written on a Read Only Memory (ROM) chip. Newer BIOSes often use a Flash ROM, a non-volatile memory chip that can be rewritten-in order to incorporate upgrades and bug-fixes.

Because system parameters are changed often, a method of updating parameter information used by the BIOS is needed. The **CMOS** (**Complementary Metal Oxide Semiconductor**) RAM stores this information, which can be changed through a BIOS utility often called the CMOS Setup. A small battery is used to power the CMOS when the computer is turned off. During the boot sequence, the BIOS refers to the information contained in CMOS to initialize (boot) the computer and perform a **Power On Self Test** (**POST**).

Most 80286 and 80386-type computers divide the BIOS into two chips. Although the CPU on these computers can handle a 16-bit data path, the BIOS design is limited to an 8-bit data path. If the entire BIOS were to be placed on one chip, access to it would be through a slow 8-bit path. By dividing the BIOS between two 8-bit chips, data can be transferred from each chip simultaneously, effectively creating a 16-bit data path. The access order is staggered for each byte of data. One chip contains all the odd numbered bytes and the other contains all the even numbered bytes.

If you were to look at the BIOS area with the DOS DEBUG command (df000:0), you would probably see a copyright message such as:

```
CCOOPPYYRRIIGGHHTT
```

This is because each of the two chips has a copyright message in it and when meshed together form this composite message.

More current system designs (80486 and above) use only one BIOS chip. They get by the 8-bit BIOS data-path problem by utilizing BIOS Shadow RAM. Shadow RAM is fast RAM memory that is set aside for the placement of a copy of the BIOS contents. The Shadow RAM can then be accessed through 16, 32, or even 64-bit data paths. In these types of systems, the shadow feature must be turned on.

Most systems that use two chips use 27256-type chips. The 256 part represents is the number of kilobits that the chip can hold. If you divide this number by 8 you can get the number of bytes that the chip holds, which is 32KB. So two of them together equal 64KB, the normal size for current BIOSes. Some older systems use two 27128-type chips, which is a total of 32KB (16KB x 2), and a few newer systems use two 27512 chips which is a total of 128KB (64KB x 2). The systems that use a single BIOS chip usually have one 27512 (64KB).

BIOS Imposed Limits

Please read this section for information on BIOS/DOS-imposed drive limits.

The original IBM (80x86 CPU) BIOS originally allocated just enough bits for 1,024 cylinders, 255 heads, and 63 sectors/track in its drive parameter look-up table. This imposed a 1,024 cylinder limit that was soon exceeded. Drives accessed through this system BIOS got around the limit by performing a translation (see the IDE section of Appendix B).

Other older BIOSes (regardless of the operating system) also limited the acceptable number of cylinders in a drive to 1,024. Some BIOSes provided for 4,096 cylinders, but this feature was useful only with newer operating systems other than DOS. This is because of the interface originally provided between the BIOS and the operating system, called INT 13h.

INT 13h

The method that is provided for applications (such as an operating system) to access hard drives through the BIOS is **Interrupt 13h** (**INT 13h**). The "h" is used to indicate that the number is hexadecimal. Issuing an INT 13h call allows an application to let the BIOS know when it needs to use the hard drive. Note that this is not the same as a hardware interrupt request (IRQ). Due to the number of bits allocated for drive parameters, INT 13h also only provides for 1,024 cylinders. Because DOS uses INT 13h for disk access, it and related operating systems are subject to the 1,024 cylinder limit.

Recently manufactured BIOSes (Enhanced BIOS) use a translation scheme at the INT 13h interface to enable DOS-based operating systems to work with large capacity drives (see the IDE section of Appendix B). These translation schemes allow the use of new large-capacity drives while maintaining INT 13h compatibility for older DOS-based software.

Another way of getting around BIOS-imposed limits is to bypass the system BIOS altogether. For example, SCSI controllers with an onboard BIOS do not use the system BIOS for drive access.

Operating System

The Operating System **(**OS**) consists of a suite of programs that allow interaction between the user, software, and hardware.**

The **Operating System** (**OS**) consists of a suite of programs that allow interaction between the user, software, and hardware. Some of the more popular operating systems include **DOS** (**Disk Operating System**), Windows, OS/2, and UNIX. There are many excellent resources available on the various operating systems, therefore this text only deals with operating system matters that specifically relate to hard drives.

The operating system provides a standardized way for programs to access a computer's resources (usually through the BIOS). This is important because software and peripherals are produced by a multitude of manufacturers. The advantage is that consumers have a reasonable assurance of buying hardware/software products that are compatible with their systems.

Many hard drive conventions were established by DOS and carry over into more advanced operating systems. DOS is still in widespread use because the flexibility it offers for working directly with hardware. Most of the drive terminology used in this text relates to DOS.

Preparing a Drive for an Operating System

One analogy used to understand how a hard drive functions is to compare it to a warehouse building. A warehouse requires various structures of organization in order to allow efficient storage and shipment of goods. Hard drives also require organizational structure. These structures include **partitions**, **MBR** (**Master Boot Record**), **partition tables**, **DOS Boot Record** (**DBR**), **FATs** (**File Allocation Tables**), and **directories**. Because of its prevalence, most of the information in the following sections apply specifically to DOS.

Low-Level Formatting

Low-level formatting **physically divides the hard drive's disks into tracks and sectors in preparation for the reception of data and records the sector header data that organizes the tracks into sequential sectors on the disk surfaces.**

Prior to receiving goods, the space in a warehouse needs to be divided into individually addressed shelf or floor spaces. This allows the shipping workers to place items in known locations to facilitate retrieval. On a hard drive, this process would equate to the **low-level format**.

Low-level formatting is the first step in preparing a drive to store information. The process sets up the "handshake" that allows communication between the drive and the controller. Low-level formatting physically divides the hard drive's disks into tracks and sectors in preparation for the reception of data. It also records a sector header that organizes the tracks into sequential sectors on the disk surfaces. Sector header information identifies the sector number and contains the head and cylinder address. This information is never altered during normal read/write operations.

For older drives, such as ST-506/412 and ESDI, the low-level format was not only an initial setup requirement, it also needed to be performed periodically as a maintenance task for the life of the drive. This was because the sector boundaries tended to degrade over time and need to be "refreshed." In the 8088/8086 (XT) system, the low-level format program was usually accessed using the DOS DEBUG command. In an 80x86-type system, the low-level utility is typically built into the BIOS. Third-party software is available for performing the low-level format on both 8088/8086 and 80x86-type machines.

Most IDE and SCSI drives are low-level formatted at the factory using special equipment, therefore you should not attempt to low-level these drives.

Partitioning

Partitioning **is a mandatory operation which organizes the physical drive into logical volumes.**

Sometimes a warehouse isn't used exclusively by one company, or a company finds it convenient to divide the warehouse physically between several product lines. In this case, a wall or some other type of clear delineation is provided. Hard drives are **partitioned** for similar reasons.

A partition is a division of the available space on the hard drive into separate logical volumes. Each partition on a drive is logically viewed as a separate drive by the host computer. Drive partitions were instituted due to conditions that existed in the early days of hard drive development. The current level of standardization didn't exist in the computer industry then, so it was more common to see several different operating systems being used to run the various applications available. Because of the expense of mass data storage in the past, a single hard drive often had to be shared by several operating systems. Each operating system requires its own partition.

Because of the conditions outlined above, partitions became a standardized requirement for setting up hard drives in a PC-compatible operating system. Partitioning is a mandatory operation in preparing a hard drive for data storage, even if only one partition occupies the entire capacity of the drive. Partitioning involves writing the Master Boot Record (MBR) and Partition Table to the first track (Track 0) on the drive. **FDISK** is the DOS program that performs this operation.

MBR (Master Boot Record)

When a computer is powered on, the ROM BIOS boot program looks to the first physical sector on the primary (boot) drive for the MBR, which is then loaded and executed. The MBR reads the Partition Table to see which partition is "active," meaning that the partition contains the program needed to boot the operating system. The MBR then hands off control of the computer to the boot program in the active partition.

Partition Tables

The main Partition Table resides in the first sector of Track 0, and is used by the MBR to determine the locations and parameters of existing partitions, as well as identify the active (OS boot) partition. In the case of DOS, there can be several additional partition tables on a drive, depending on the number of logical volumes the drive holds. This topic will be covered in detail in the following sections.

PCs have provisions for up to four partitions in their main Partition Table. Each partition is unique to a specific operating system as identified in the **System Type** entry of the table (see Table A-2).

DOS can occupy one or two partitions on a single drive. The other two partitions can be unused or occupied by other operating systems. In either case, the other two partitions are essentially non-existent to DOS. The first DOS partition is called the **Primary Partition**. After being high-level formatted (see **page 122**), the Primary Partition will contain one DOS volume, and will be designated the C: (boot) drive. The Primary Partition cannot be divided into **logical drives** (see below). In a strictly DOS system, the Primary Partition will be marked active (bootable) in the main Partition Table. On the latest versions of DOS, a Primary Partition can occupy up to 2.1 GB of drive space. A Primary Partition that is larger than 32 MB is termed a **DOS Huge** partition because early versions of DOS were limited to a single partition of 32 MB or less. See **page 126** for more information on partition limits. DOS/Windows systems today are typically set up by vendors with one large (DOS Huge) partition, unless the total drive capacity exceeds 2.1 GB.

The other type of DOS partition is called the **Extended Partition**. The concept of the Extended Partition was instituted to overcome capacity limitations of earlier versions of DOS. The Extended Partition is further divided into sub-partitions that are called **logical drives**. There is no limit to the number of logical drives dividing up the Extended Partition (D:, E:, F:, …), but there must be at least one. To DOS and the end-user, logical drives function exactly like physical drives, except that they cannot be designated as DOS bootable. Each one must be high-level formatted to be used as a DOS volume.

Although the primary (boot) drive must have a Primary Partition, any secondary (non-boot) physical drives can be occupied by one large Extended Partition. No Primary Partition is required on a secondary drive. The secondary drive will still have an MBR and main Partition Table, but the MBR in the secondary drive is unused. The main Partition Table in the secondary drive is read by the MBR in the primary drive.

Table A-2: Typical Main Partition Table

Slot	Active	System Type	Beginning (Hd/Cyl/Sec)	Ending (Hd/Cyl/Sec)	Prior Sectors	Partition Sectors	Total Space
1	Yes	DOS 12	0/0/1	6/83/35	35	20545	11M
2	No	DOS Extended	0/84/1	6/843/35	20580	186200	95M
3	Unused	0	0	0	0	0	0
4	Unused	0	0	0	0	0	0

The table above depicts the main Partition Table on a 106 MB hard drive. It contains one Primary Partition (DOS 12-bit File Allocation Table) and one Extended Partition.

Slot Number	Indicates the partition (for the main table) or logical drive (for a secondary table, see below) that the rest of the row relates to.
Active	Designates whether the partition will be used to boot the system to an OS.
System Type	Indicates the specific operating system that is controlling the partition. See "File System" on **page 123** for more information.
Beginning	Designates where the partition starts in CHS format.
Ending	Designates where the partition ends in CHS format.
Prior Sectors	Indicates the number of sectors on disk prior to the start of the partition.
Partition Sectors	**Partition Sectors** indicates the total number of sectors in the partition.
Total Space	**Total Space** is the amount of storage capacity in MBytes that the partition contains.

The Extended Partition and each logical drive contains its own partition table. These tables look the same as the main Partition Table, containing four slots and the same types of entries. In fact, it is possible to have a non-DOS System Type defined in a secondary partition table.

Only the first two slots of these secondary partition tables can be occupied by information. The other two slots are filled with zeros. The first slot will contain information about the parameters of the Extended Partition or logical drive. The second slot will point to the location of the partition table belonging to the next logical drive. As you can see, this means that the tables are joined together in a kind of daisy-chain. The last table in the chain "terminates" the chain by filling the second slot with a zero (since there is no other partition table to point to).

The table below depicts the secondary partition table belonging to the Extended Partition of Table A-2.

Table A-3: Typical Secondary Partition Table

Slot	Active	System Type	Beginning (Hd/Cyl/Sec)	Ending (Hd/Cyl/Sec)	Prior Sectors	Partition Sectors	Total Space
1	Yes	DOS Huge	1/84/1	6/501/35	35	102375	52M
2	No	DOS Extended	0/502/1	6/843/35	102410	83790	43M
3	Unused	0	0	0	0	0	0
4	Unused	0	0	0	0	0	0

Note that *52 Mbytes + 43 Mbytes = 95 Mbytes*, which is the total capacity of the Extended Partition in Table A-2.

The following is a summary of DOS partitioning terms:

Primary Partition	The partition that holds the boot sector for DOS. When the partition is marked active (bootable) it is normally designated by DOS as the C:\ drive.
Extended Partition	The secondary DOS partition.
Huge Partition	Same as a primary partition, but can occupy as much as 2.1GB of disk space. Early versions of DOS limited the Primary Partition to 32 MB (see Table A-8).
Logical Drive	A further division of the Extended Partition represented by drive letters (E:, F:, G:, etc.).
Non-DOS Partition	Any partition that is not the DOS Primary or Extended Partition. These partitions would occupy the third and fourth slots in the Partition Table of an initially DOS-partitioned (FDISKed) drive.

See the next section for information on DOS partition limits.

High-Level Formatting

High-level formatting **organizes the drive partitions for a filing system by constructing a root directory from which sub-directories can be created.**

Once the warehouse has been set up with shelf space and divided to separate product lines, a cataloguing system must be instituted to keep track of inventory. This system allows the shipping department to know where to find items as well as where and how much space is available. To a hard drive, **high-level formatting** is the process of setting-up file structures that are similar in nature to a warehouse inventory system.

High-level formatting, prepares a partition or logical drive for use with a specific operating system. High-level formatting is sometimes called “DOS Formatting.” It does so by writing several structures to the beginning of the partition. These structures include the **boot sector** (sometimes called the **DOS Boot Record** or **DBR**), the **file system** (for DOS, this is the **File Allocation Table** or **FAT**), and the **Directory Structure**. When a partition or logical drive is high-level formatted for DOS, it is said to contain a DOS volume, meaning it is usable by DOS.

Boot Sector

High-level formatting writes a program and a data table to the first sector of the partition that is used to boot the operating system. The generic name for this area, when not referring to any specific operating system, is the **boot sector**. After the MBR determines the active partition, it goes to the beginning of that partition to find the boot sector. The MBR then hands off control of the computer to the boot program that resides there, which then boots the operating system.

The DOS Boot Record (DBR) is a data table that the boot program reads to find the parameters of the logical volume. Information from the DBR is copied to an area of memory called the BIOS Parameters Block (BPB), which is then used by DOS to access the drive. **Table A-4** depicts the information that might appear in the DBR of the Primary Partition of a boot drive.

All DOS Primary Partitions and logical drives will contain a boot sector at the beginning. However, only the boot program from the Primary Partition on the boot drive can be used to load the operating system, and it reads the data tables for all other DOS volumes in the system. The boot programs in any other Primary Partitions (on secondary physical drives) or logical drives are unused.

Table A-4: Typical DOS Boot Record

Drive ID	128
OEM ID	MSWIN4.0
Volume Label	SANDYSC
Serial Number	1012077765
Bytes Per Sector	512
Sectors Per Cluster	8
Reserved Sectors	1
Number of FATs	2
Root Entries	512
Total Sectors	415948
Media Descriptor	248
Sectors Per FAT	203
Sectors Per Track	52
Number of Heads	8
Hidden Sectors	52
Boot Signature	41
File System ID	FAT16

File System

Because files are constantly created, updated, and deleted, it is impossible to maintain them as an uninterrupted string of physical sectors on the drive. More than likely, files will exist in fragments that are placed in whatever space is available. A file system is a kind of table of contents for disk space. It keeps track of the status of individual clusters (see below) in a partition–whether they are free, occupied, or defective.

Incompatible file systems are the primary reason that different operating systems require their own partitions. DOS calls the file system the **File Allocation Table** (**FAT**).

DOS FAT

DOS actually writes two FATs to a logical volume. The second FAT is a backup just in case the primary one becomes damaged. The FAT is a simple table of numbers in which each entry indicates the status of a cluster (see the next section) or the location of the next cluster belonging to a file. If you wished to manually trace a file's clusters, you would start with the directory (see **page 128**) to find the location of the first cluster. From there, you would go to the indicated entry in the FAT. The entry will contain the location of the next cluster of the file. Follow FAT entries in this manner until you reach the END OF FILE designator. A FAT entry of 0 indicates that the cluster is free to be written to. BAD tells DOS not to use a cluster due to bad sectors contained within.

FAT entries are coded by their numerical values. Table A-5 shows the hex numbers that correspond to 16-bit FAT entries.

Table A-5: 16-Bit FAT Entries

Entry	Meaning
0	Free
2-65519 (2-FFEF hex)	Location of next cluster in the file
(FFF0-FFF6 hex)	Reserved/not used
BAD (FFF7 hex)	Bad sectors in cluster
END (FFF8-FFFF hex)	End of file

If you were to look at the FAT with a sector editor, you would see numbers such as those on the left side of the table. Most sector-editing utilities, such as contained in DrivePro, automatically translate hexadecimal into the decimal values and coded meanings.

There are two types of DOS FATs: 12-bit and 16-bit. Each cluster (see below) that the FAT keeps track of is allocated 12 or 16 bits for its number. DOS volumes with less than 32 MB get a 12-bit FAT, while those with 32 MB or greater get a 16-bit FAT. A 12-bit FAT can keep track of up to 4096 (2^{12}) clusters, while a 16-bit FAT can have up to 65536 (2^{16}) entries. See **page 126** for more information on limits imposed on DOS volumes by the FAT.

If you were to view a 12-bit FAT using a sector editor, it would look very different from a 16-bit FAT. This is because the 16-bit FAT meshes perfectly with hexadecimal notation–one cluster entry equals two bytes. In contrast the 12-bit FAT is one and a half bytes. Therefore, you would actually be seeing two FAT entries for every Three hexadecimal numbers (bytes).

Table A-6: 12-Bit FAT Entries

Entry	Meaning
0	Free
2-4079 (2-FEF hex)	Location of next cluster in the file
(FF0-FF6 hex)	Reserved/not used
BAD (FF7 hex)	Bad sectors in cluster
END (FF8-FFF hex)	End of file

Note that both 12-bit and 16-bit FATs start with the third cluster entry (number two). The first entry (number zero) is used for the **Media Descriptor Byte**, which is used by DOS and other applications to tell the type of storage device the FAT is installed on. For hard drives, this number is F8h. The second entry (number one) is padding, containing the END OF FILE value.

Other drive structures, such as boot sectors and partition tables, are fairly easy to recover because their contents are fairly static. FATs, however are difficult to repair because file space management is a dynamic process–they are updated every time a file is saved or deleted. Attempting to restore an old FAT to a drive can result in serious data corruption. Only files that were not altered since the last FAT backup can be reliably recovered.

Clusters

A cluster **is the smallest unit of drive space that an operating system can work with.**

The FAT does not see individual sectors. Instead, it sees the partition as a mass of **clusters,** which are also known as **allocation units**. A single cluster, composed of several sectors, is the smallest unit of drive space that an operating system can work with. A cluster can exist in one of three states: used, unused, or defective (contains bad sectors). At any time, the FAT keeps track of the status of each cluster in the partition according to these terms.

The reason that the operating system does not work with individual sectors is that it would significantly degrade system performance. Keeping track of the state of every single sector on the drive would require an enormous table, with proportionally longer maintenance overhead. Organizing groups of physical sectors into clusters streamlines **FAT** processing chores and keeps the size of the FAT down to a more manageable level. The net effect is to allow DOS to run faster. The drawback to using clusters is that more physical disk space is wasted. Since a file *must* take up at least one cluster and usually several, there will almost always be some unused sectors in the last cluster allocated for that file, called **slack space**. The result is that much more physical space will remain unused than if the drive could be allocated by the sector.

The number of sectors composing a cluster depends on the DOS version and size of the partition (see the next section). Note that a significant disadvantage of using larger partitions is that DOS assigns more sectors per cluster as the partition increases in size. It does so in order to keep the FAT size from degrading system performance. This means that as partition size goes up, slack space increases proportionally.

For example, one partition uses eight sectors per cluster and wastes an average of half the last cluster in slack space. Another partition uses 64 sectors per cluster and has the same portion of the last cluster wasted. The smaller partition would waste 4 sectors, or 2,048 bytes per file, while the larger partition would waste 32 sectors, or 16,384 bytes per file. Multiply this times thousands of files and you can see that a significant amount of storage space can be lost in this manner.

The following table illustrates the progressive increase in cluster size relative to partition size for DOS:

Table A-7: DOS Cluster Sizes

Partition Size	Cluster Size (512 bytes/sector)
16 - 127 megabytes	4 sectors/2KB
128 - 255 megabytes	8 sectors/4KB
256 - 511 megabytes	16 sectors/8KB
512 - 1,023 megabytes	32 sectors/16KB
1,024 - 2,048 megabytes	64 sectors/32KB

File Systems and Partition Limits

FAT-16 means that sixteen bits are allocated for the address of each cluster. This means that FAT-16 can keep track of a total of 2^{16}, or 65,536 clusters in the DOS volume. The net effect is that the ultimate capacity limit for a single DOS (versions 4.0 and later) volume is 2.1GB. Since FDISK creates partitions for a specific file system (see **page 119**), it limits partition sizes accordingly. For those interested, here is the math:

(bits allocated)(sectors/cluster)(512 bytes/sector) = maximum size

or

$$2^{16} \times 64 \times 512 = 2{,}147{,}483{,}648 \text{ bytes (2.1GB)}$$

Windows for Workgroups (WFW) uses an optional system called 32-bit disk access, and Windows 95 uses the VFAT (Virtual File Allocation Table). These file systems are also bound by the 2.1GB limit. If a drive is under the 2.1GB limit, then multiple partitions are not necessary (although the partitioning process is still required).

Some operating systems do not have this limitation because they use more advanced file access systems for keeping track of file locations. Windows 95 uses FAT-32, Windows NT uses a system called NTFS (NT File System), and OS/2 uses the HPFS (High Performance File System). These are 32-bit file systems that are not limited to 2.1GB volumes, but are also not compatible with the DOS 16-bit FAT.

The following information details the partition limits of the various DOS versions:

- DOS versions 2.0 to before 3.00 support a single partition of up to 16MB.
- DOS 3.00 to before 3.30 support a single partition of up to 32MB.
- DOS versions from 3.30 to before 4.00 may have two partitions (Primary and Extended). The Primary Partition can be no greater than 32MB, while the Extended Partition has no limit. However, the Extended Partition must contain at least one logical drive (DOS volume), and the logical drives are all limited to 32 MB each.
- DOS versions from 4.00 to before 5.00 may have two partitions (Primary and Extended) up to 2.1GB each. All logical drives can be up to 2.1GB.
- Versions 5.00 and later have the same restrictions as 4.xx, except they allow support for up to eight physical drives.

Table A-8: DOS Version Partition Limits

Version	Number of DOS Partitions (Primary and Extended)	Max Partition Size
2.x	1	16MB
3.00-3.29	1	32MB
3.30-3.99	2	32MB (Primary)/32MB (Logical Drives)
4.x	2	2.1GB
5x-6.2	2	2.1GB

Note that while later versions of DOS theoretically allow partitions of up to 2.1GB, *none of the DOS versions can handle a hard drive with over 1,024 cylinders unaided* (see **page 116**). This results in the capacity of an individual IDE drive being limited by combined DOS/BIOS/IDE constraints to 528MByte. See the IDE section of Appendix B for more information on this topic.

Directory Structure

The **Root Directory** of the DOS logical volume is physically located after the second FAT. A directory is a table filled with records that contain information about all files stored in the volume. Each record is 32 bytes long and contains (in order) the file's name, extension, attributes, time and date last modified, starting cluster location, and size.

The root directory only contains information about files located at the highest level of the directory structure. All sub-directories are stored as files elsewhere in the volume. Sub-directories are identified by the D attribute (see below). Each sub-directory contains the same type of information as the Root Directory.

File names are eight bytes long (thus the eight letter file name restriction for DOS). The byte values correspond to characters in the extended ASCII table. Some characters in the first byte are reserved because they are used to indicate a property of the directory entry. A "0" means that the entry has never been used. "σ" (Greek letter sigma) means that the file has been deleted or overwritten. Period (.) and double period (..) in the this location in a directory or sub-directory is used to tell DOS the rest of the line contains information about the directory structure.

DOS additionally provides for a three byte (letter) long extension after the file name. The extension is generally used to identify the purpose of the file or the application that uses it.

Attributes tell DOS how to handle a file. A file can be given several attributes simultaneously. These are (in order): Archive, Directory, Volume, System, Hidden, and Read-Only. Attributes are encoded by the value of the bit in the corresponding location in the directory entry. A bit value of “1” indicates that the file is designated with the corresponding attribute.

Time (two bytes) is stored in hours, minutes, and seconds, while date (two bytes) includes the year, month, and day.

The last two entries in a directory entry are the starting cluster location (two bytes) and file size (four bytes) in total bytes. The directory only keeps track where the file starts. The FAT holds the information concerning the locations of all file fragments.

Table A-9: Directory Entries

Byte	Meaning
1-8	File name
12 (last 6 bits)	Extension
13-22	Reserved/not used
23-26	Time and date
27-28	Starting cluster number
29-32	File size

Putting It All Together

It is sometimes easier to understand the organization of a drive by looking at what happens after a computer is powered up. The example below makes use of a typical DOS boot operation.

When a computer is first powered on, it will load the boot program from the BIOS ROM. This program first performs a Power On Self Test (POST), performing a cursory diagnostic of the hardware in the system. When the POST is finished, the BIOS goes to the first physical sector on the primary (boot) drive for the Master Boot Record (MBR), which is then loaded and executed. The MBR reads the Partition Table to identify existing partitions and to see which partition is active. If a DOS Extended Partition exists, it goes to the secondary partition table at the beginning of the Extended Partition and follows the pointers to the partition tables of all designated logical drives. The MBR then hands over control to the boot program for the active partition. The boot program reads the DBR, reads the FAT and the Directory, then finishes loading the operating system.

This page intentionally left blank.

Appendix B
Hard Drive Interfaces

The purpose of this section is to provide technical details and historical information on the four major interface types.

The IDE Interface

IDE General Description

IDE stands for "**Integrated Drive Electronics**." The term simply means that all the electronics required to control data transfer exist on the drive's circuit board. The alternative would be to have the necessary hardware reside on the controller card. IDE is not the actual interface name, but the term has become commonly accepted in the industry to encompass the various facets of the format. The interface itself is actually called the **ATA**, or **AT Attachment** (as in IBM AT), and is defined by a set of standards laid down by boards of industry engineers currently called the X3T13 committee and the ATA Working Group.

Because the drive electronics are self-contained, IDE drives (like SCSI drives) are said to be intelligent. This is a distinct advantage to the manufacturer, because they don't have to sacrifice performance to ensure wide compatibility. It is also a benefit to the end-user, where, in contrast to ST-506/412 and ESDI drives, controller compatibility is not an issue of concern.

The electronic instructions of IDE drives are installed on the drive. Because of this, the card (controller) that plugs into the I/O expansion bus is just a "dumb" adapter. This adapter card serves little purpose other than connecting the drive to the computer. Incidentally, the absence of drive electronics on the adapter leaves room for other applications. This is how manufacturers are able to put a floppy drive, two serial ports, a game port, and parallel port on a single half-length IDE card. Most new motherboards even have an embedded IDE drive connector, eliminating the need for an adapter and freeing up an additional bus slot.

IDE drives perform a translation, converting the logical information which the drive and the BIOS tables can agree on into the actual information concerning where information is physically stored on a disk. This translation became necessary because most IDE drives started using 36 or more sectors per track, which at the time was not supported in BIOS drive-type tables. Other physical parameters have also exceeded the BIOS/DOS-defined limits due to advances in drive technology. See the section below concerning IDE drive capacity for details on physical and logical specifications.

If you have recently been shopping for an IDE drive, you may have experienced some confusion regarding the terms **Enhanced IDE** (**EIDE**), **ATA-2**, **ATA-3**, and **Fast ATA**. These are all terms that relate to drives incorporating updates of the ATA interface. As explained previously, the term ATA refers to the actual interface, while IDE refers to the storage format that implements the ATA interface. Correspondingly, the updated interface, which is standardized and approved by an ANSI-sponsored sub-committee, is known as ATA-2. ATA-2 is sometimes being marketed as Fast ATA. Enhanced IDE was originally a marketing term for Western Digital Corporation's version of ATA-2. It is now often used synonymously with the latest versions of the ATA interface.

The primary reason for improvements on the ATA interface is to provide performance upgrades at minimal cost, both to the manufacturer and end-user. ATA-2 ensures backward-compatibility with IDE (AT) products through the use of the standard 40-pin connector and the same signal definitions. Some manufacturers add more comprehensive enhancements that may require BIOS and controller upgrades to take full advantage of the new drive's capabilities.

Until recently, most IDE controllers could only handle two attached devices, even though the interface has always provided for a secondary address. The bottleneck in this case (and others) was the PC BIOS, which would only supports two IDE drives off of the primary address. Recently manufactured BIOSes now provide support for the alternate address, allowing the controller to attach up to 4 devices.

In the recent past, the only devices that used the ATA interface were IDE hard drives. The ATA-2 standard has addressed this shortcoming. The reason for wanting a greater variety of IDE devices is to eliminate the need for a separate expansion board for some commonly attached peripherals. In addition to simplifying device installation, this has an additional advantage of freeing up more expansion slots. The ATA-2 convention already supports the attachment of CD-ROM and tape drives to standard IDE connectors via the **ATA Packet Interface** (**ATAPI**).

The net effect of these improvements has been to maintain IDE as the dominant interface for the desktop computer market.

History of the IDE Interface

This interface originated in 1988 when a number of peripheral suppliers formed the **Common Access Method** (**CAM**) **Committee** to push an industry-wide effort of adopting a standard software interface for SCSI peripherals. Part of their goal was to specify what is now known as the ATA (AT Attachment interface), which would allow an interface to be designed into the new low cost AT-compatible motherboards. The ATA interface is usually not mentioned because it is encompassed in the term IDE. ATA refers to the interface itself and IDE to the hard drive. A standard was established and approved by the **X3T9** committee and sent to **ANSI** for approval.

The original IDE convention was designed to meet the mass-storage needs of the mid-1980s. As the convenience and low cost of IDE started to be overcome by its limitations, end-users desiring more speed and capacity from their hard drives have increasingly turned to the SCSI interface. In a move to protect their market share, several major manufacturer's of IDE drives have, in recent years, coordinated to set a new IDE standard under the banner of the **Small Form Factor Committee** (**SFF**). The revised IDE interface, properly known as ATA-2 was approved by ANSI under the **X3T10** specification in 1994. The ATA standards have since moved to its own X3 subgroup **X3T13**. ATA-2 is the latest approved standard, but **ATA-3** and **ATA-4** are currently in the works. In fact, most drive manufacturers already ship products with the ATA-3 designation. The goal of participating **SFF** members was to address the advantages that have made SCSI increasingly the interface of choice, while keeping IDE's strong points intact – low cost and ease of use.

The complete ATA standards documents can be obtained from **Global Engineering Documents**.

IDE Physical Characteristics

See the section in Appendix A for information regarding the basic workings of a hard drive. IDE drives vary widely in physical dimensions, but are commonly seen in 3.5 in., third or half-height formats in order to fit in the drive bays of most PC cases manufactured today.

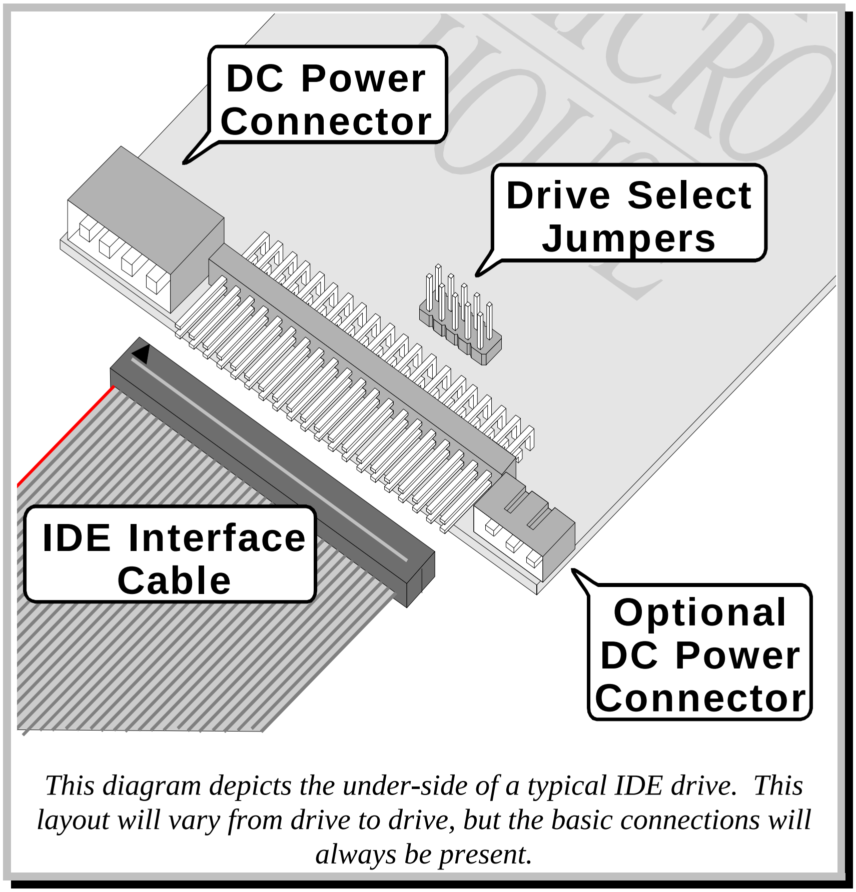

DC Power Connector
Drive Select Jumpers
IDE Interface Cable
Optional DC Power Connector
This diagram depicts the under-side of a typical IDE drive. This layout will vary from drive to drive, but the basic connections will always be present.

Figure B-0-1: IDE Connection Diagram

Cabling

IDE drives normally attach to the computer via a 40-pin connector (44-pin and 72-pin connectors have also been used) and a controller card. The controller is often embedded in the motherboard, meaning the drive cable connects directly to the motherboard.

Drive Select Jumpers (Master/Slave)

The drives connected to the IDE interface are called **Master** and **Slave**. The relationship between the two drives (which is master and which is slave) is usually determined by jumper settings on the drives. The physical order of the drives on the cable connector is irrelevant. ATA and ATA-2/Enhanced IDE hard drives may be paired on a single cable. With the old IDE (XT) interface, up to two drives may be installed in a system, but the controller must have a separate connector and port address for each drive.

Some drives offer a **Cable Select** option, which overrides the setting of the Master/Slave jumpers and designates Master/Slave based on the position of the drives relative to the cable.

The jumpers for master and slave selection are located on the underside of the drive, close to the data cable. These jumpers are almost always set to "Single Drive Only" at the factory, and usually need to be altered according to the Master/Slave designation for each drive installed on the cable.

IDE Technical Information

How to Calculate Disk Capacity on IDE Drives

Cylinders, Heads, and Sectors Per Track (**CHS**) are the parameters used to determine the capacity of a hard drive. See the section titled "Basic Drive Concepts" in Appendix A for definitions on cylinders, heads, and sectors per track.

The following formula can be used if your BIOS does not supply the capacity for the drive types, or if you have two of the drive's parameters and need the third.

Capacity in megabytes = (Cylinders x Heads x Sectors per Track x Sector Size)/(1,048,576 bytes per megabyte)

Table B-1: IDE Drive Parameters

Cylinders	Total usable drive cylinders
Heads	Drive's data heads
Sectors/Track	Usually variable in recently manufactured drives. Maximum value is 63 sectors/track
Sector Size	Usually 512 bytes. This figure is supplied by DOS, rather than the BIOS

Physical vs. Logical Specifications

Calculating capacity is perhaps the most confusing and least understood aspect of IDE drives. The reason for this confusion is that two contradictory sets of parameters appear to exist that are used to calculate drive capacity. This dual set of parameters became necessary when IDE drives began to use more than 1,024 cylinders, which was the old BIOS/DOS-imposed maximum. A translation scheme therefore became necessary to get the BIOS to think that the drive's parameters were within its limits.

The physical parameters are the actual values. The logical (translated) specifications are the heads, cylinders, and sectors that both the BIOS and DOS can work with.

The **physical parameters** (or specifications) are the actual values. If you opened up the hard drive and looked inside and saw 4 heads, that would be the physical (actual) number of heads. The logical (translated) specifications are the heads, cylinders, and sectors that both the BIOS and the IDE interface can work with. Ideally, both the physical and logical specifications can be plugged into the capacity formula (see below) and reach the same figure.

When setting up an IDE drive, the BIOS must be informed of the drive's specifications. The logical specifications are entered into the BIOS as the Drive Type. The hard drive then performs all translations from logical to physical values. The manufacturer will usually have a recommended logical translation for its drives that can be provided by its technical support department. Note that newer BIOSes have features that detect and set the Drive Type parameters automatically, so this information is usually only necessary for systems with older BIOSes.

In the example below, the physical cylinders exceed 1,024 cylinders, so the drive performs a translation, reducing the cylinders and increasing the heads and SPT count in order to arrive at the same capacity.

Table B-1: Physical vs. Logical Parameters Example

Spec. Type	Heads	Cylinders	Sectors/Track	Sector Size	Capacity
Physical	4	1,985	62	512	252.04MB
Logical	10	895	55	512	252.03MB

Notice that .01MB of capacity has been lost due to the translation. Because specifications must be whole numbers, logical translations rarely match the potential physical capacity, and some of that capacity must be sacrificed. Manufacturers' recommended logical translations are designed to minimize lost capacity.

It is not necessary to use the manufacturer's specifications for setup—any translation within the acceptable parameters will work.

IDE Capacity Barriers

The capacity of IDE drives is limited by the highest common values of the parameters acceptable to the host system BIOS, OS and the IDE drive itself.

Drives that meet the ATA-1 specifications are limited to a capacity of 528MBytes. The 528MBytes barrier exists because of the number of bits allocated for specifying the cylinder, head, and sector address information at both the BIOS and the IDE interface levels. At the BIOS level, 10 bits are allocated for the cylinder number, 8 bits are allocated for the head number, and 6 bits are allocated for the sector number. At the ATA-1 interface level, 16 bits are allocated for the cylinder number, 4 bits are allocated for the head number, and 8 bits are allocated for the sector number. The IDE and BIOS limits cannot be taken individually, however. The lesser of the IDE/BIOS bit values allocated to a parameter is what determines the actual capacity limit of the ATA-1 interface. **Figure B-2** may help you understand this process.

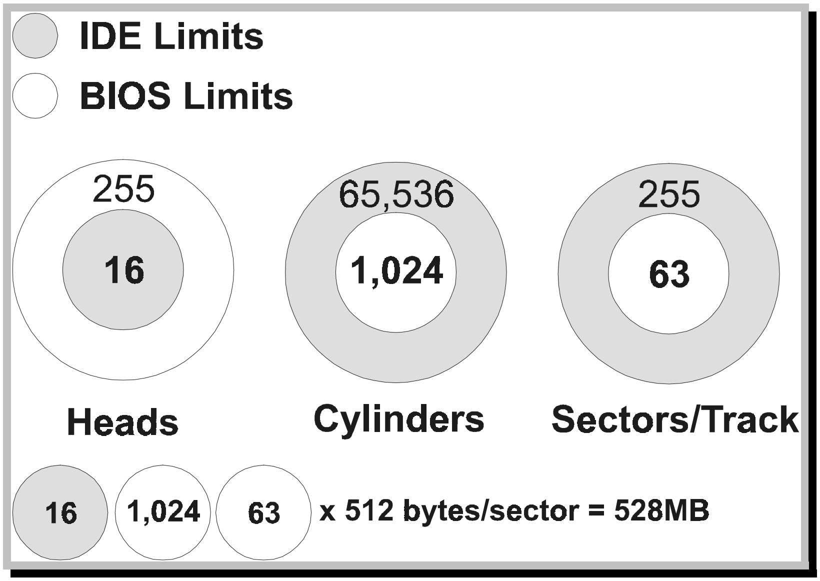


Figure B-2: ATA-1/BIOS Parameter Limits

The actual ATA-1 capacity limit is calculated by taking the lesser of the parameter limits between the BIOS and the IDE interface. The capacity of drives in the IDE format is determined by the highest common values of the parameters acceptable to both the host system BIOS and the IDE drive itself.

Table B-2: ATA-1/BIOS Imposed Limits

	Heads	**Cylinders**	**Sectors Per Track**	**Sector Size (DOS)**	**Capacity**
IDE	16	65,536	255	512	136.9GB
BIOS	255	1,024	63	512	8.4GB
ATA-1 (Lesser Value)	16	1,024	63	512	528MBytes

IDE drives can be instructed to use parameter translations, these utilize the entire disk with no loss of disk space if they do not exceed 528MBytes. The IDE 528MByte limit is not necessarily written in stone.

Almost all IDE drives manufactured today have parameters that are greater than 16 heads, 1,024 cylinders, and 63 sectors/track. This means the capacity of these drives exceeds the ATA-1 528MByte limit. Enhanced IDE hardware offers support for large drives, but requires native support for more than 1,024 cylinders by the BIOS (Enhanced BIOS). Most older BIOSes do not allow a user to enter parameters larger than those above, therefore the BIOS must be upgraded or third-party software support must be added.

DrivePro's installation utility can install EZ-BIOS, a software solution that will allow large IDE hard drives (>528MBytes) to be compatible with systems that do not natively offer large drive support. EZ-BIOS features support for large drives with capacities between up to the ultimate IDE limit of 8.4GB. Note that drives still need to be divided among volumes that do not exceed 2.1GB. See Appendix A for more information on partition limits.

If you have a BIOS that has large drive support, you will not need to use EZ-BIOS to access the full capacity of your hard drive. If this is the case, you can still use the installation utility for hassle-free installation. Most drives are partitioned and high-level formatted in under a minute.

WARNING: If an IDE drive is told to use parameters that exceed the DOS restrictions (number of cylinders greater in number than 1,024), DOS (and DOS-based OSes) will accept and *use these parameters* with no error message. Some BIOS tables, in fact, even have entries that are invalid.

For example, some BIOS tables have 15 x 1,224 x 17 as parameters for type 46. These parameters may even be the physical parameters for some drives, but if you select these parameters for a drive and the drive is told to write to a logical sector beyond the 1,024 cylinder limit, DOS will write to the drive's cylinder 0 in a "wrap-around" effect *without giving any error to the user*. This is due to a little known anomaly in DOS. Writing to this location can be disastrous, since it will write over, in approximate order, the Master Boot code and Partition Table, DOS Boot Record, FATs, Root Directory, and ultimately your data. Note that the drive will perform correctly until the disk is filled to the point that it begins to wrap around. So, even though the drive boots and can be written-to and read-from, the DOS restrictions still apply. The results could become disastrous in time. Note also that this is a DOS software limitation and does not necessarily apply to newer systems that specifically support more than 1,024 cylinders, and with operating systems other than DOS. Using EZ-BIOS will completely overcome this problem.

The most obvious solution to the DOS 2.1GB partition limit (see "High-Level Formatting" in Appendix A) is to divide the drive into a Primary Partition and logical drives smaller than 2.1GB. DrivePro's installation utility will automatically set your drive up with the necessary partitioning scheme to maximize capacity. In addition to overcoming the partition size limit, this solution has the advantage of using more efficient cluster sizes. Operating systems that use a more advanced file system (such as OS/2 Warp or Windows NT) do not have this limit.

One final note about IDE capacity limits: at the rapid rate that IDE drives are increasing in capacity, the 8.4GB limit (see Table B-2) imposed by the BIOS is the next challenge facing the disk drive industry. Overcoming this barrier will require a comprehensive change in the IDE/BIOS interface.

IDE Addressing Schemes

The IDE interface has traditionally used CHS addressing (see Appendix A). CHS originally used the physical specifications to perform disk addressing, but physical limits prompted the use of logical translation schemes. Some BIOSes simply translate CHS to under 1,024 cylinders to get around the INT 13h limit, but the natural progression is to simply translate CHS addresses into linear block numbers that are easier to work with.

IDE Logical Block Addressing (LBA) is also a translation scheme, but only uses one-dimensional sequential sector addresses rather than the three-dimensional CHS system. LBA is the disk access method used by Enhanced IDE for large-capacity (>528MBytes) disk drives. LBA is a comprehensive change over the old IDE format, and requires specific support from the host-system's BIOS (Enhanced BIOS or EBIOS) and the drive's firmware.

IDE LBA works like this: On power-up, the host system's BIOS performs a Power-On Self Test (POST) during which it interrogates the drive with the Identify Drive command. The BIOS checks the drive response to see if it supports the LBA mode. IDE LBA consists of a 28-bit address for up to 268,435,456 sectors of 512 bytes, making the interface theoretically capable of capacities of up to 137,438,953,472 bytes or 128 gigabytes. Please note that DOS-based file systems still limit drive volumes to 2.1GB.

After confirming that the drive supports LBA addressing, the BIOS constructs an **Enhanced Drive Parameter Table** (**EDPT**). The EDPT is actually two tables—one that accepts CHS information from the drive's response to the Identify Drive command, and another that is filled with translated (LBA) information for the operating system. From this point, the BIOS and the drive communicate using the LBA format.

IDE Data Transfer

There are two methods whereby data transfer is accomplished in IDE drives. The first is **Processor Input/Output** (**PIO**)**.** There are several PIO modes defined that allow increased transfer rates, the most recent being **PIO Mode 4**. The other method is called **Direct Memory Access** (**DMA**)**.** There are two variations on DMA: **Single-word DMA** and **Multi-word DMA**. These two DMA categories are further divided into modes that allow for increased performance. Further details on PIO and DMA are provided in the following sections.

Another way to speed up data transfer is through **caching**. **Caching** is the process where the system loads data from the hard disk to the **RAM** set aside as **cache memory**. The system may then refer to **cache memory** for information instead of going back to the hard disk, thereby increasing the processing speed. Please read on for more information on caching.

PIO Modes

PIO is the transfer method used on most IDE drives. PIO relies on the CPU to handle data transfer tasks. **PIO Modes 0, 1, and 2** are considered "blind" transfer modes, because the amount of data sent to the CPU is not modulated. In the case of PIO Modes 0-2, the CPU and drive do not have a means to monitor the availability of data transfer resources, and accordingly lack a means to "throttle" the amount of data transferred. Because of this, the amount of data sent by the drive in response to the CPU during a given transfer cycle must be fixed. In addition, the amount of the transferred data must be limited to the worst case of the amount of the drive's memory buffer which is guaranteed to be available for host transferring duties. PIO Modes 0, 1, and 2 are defined to provide increased performance levels within the limits of the blind transfer format.

Block PIO (**BPIO**), sometimes called **multiple-sector data transfer**, is supported by local-bus controllers (and needs BIOS support as well) and counts the 512-byte transfer unit of normal PIO as a portion of a block that consists of n times 512-bytes. This feature greatly reduces interrupt overhead because more than one sector is processed per interrupt call. This has the effect of improving system performance. Multiple-sector data transfers require a compatible drive and controller that support the **Read Multiple** and **Write Multiple** commands, as well as specific support from the BIOS. Software packages that redirect BIOS calls (such as EZ-BIOS) can also support multi-sector data transfer.

PIO Mode 3 is defined in the ATA-2 update and is available on newer IDE drives. There are presently **PIO Mode 4** drives on the shelves, and **PIO Mode 5** is right around the corner. PIO Modes 3 and higher are also called **Programmed I/O**, **Throttled PIO**, or **Flow Control**, because they allow modulation of the amount of data being transferred in a cycle. This is accomplished through the use of a signal issued by the drive known as the **I/O Channel Ready** (**IORDY**), which is used to control data throttling.

PIO Modes 3 and greater require support by the drive, local-bus controller, and BIOS. If any one of these elements is not PIO Mode 3 capable, then only a blind transfer (PIO Modes 0, 1, or 2) can be enabled. This means that simply plugging in an Enhanced IDE drive into an older system will not result in higher drive performance.

Table B-3: PIO Modes

PIO Mode	Average Transfer Rate	Cycle Time	Flow Controlled?
0	3.3MB/s	600ns	No
1	5.2MB/s	383ns	No
2	8.3MB/s	240ns	No
3	11.1MB/s	180ns	Yes
4	16.6MB/s	120ns	Yes

DMA Modes

DMA is usually offered as an option on IDE drives that can be enabled by jumpers. DMA allows the drive to bypass the CPU and control data transfer with the host memory directly. There are two types of DMA: **Single Word DMA** and **Multi-Word DMA**. DMA uses two handshaking signals, **DMA Request** (**DMARQ** or **DRQ**) and **DMA Acknowledge** (**DMACK** or **DACK**) to control the flow of data between the drive and host memory. DMARQ is asserted by the drive when it is ready to transfer data to or from the host. DMACK is returned by the host when the data transfer cycle is completed. The drive must wait until the host asserts the DMACK signal before negating the first DMARQ signal and asserting another DMARQ, if there is more data to transfer.

Single-word DMA means that a data string of specified length is transferred per DMARQ/DMACK handshake cycle. **DMA Modes 0**, **1**, and **2** define progressive performance increases.

Table B-4: Single-Word DMA Modes

DMA MODE	AVERAGE TRANSFER RATE	CYCLE TIME	REQUIREMENTS
0	2.08MB/s	960ns	DMA support
1	4.16MB/s	480ns	DMA support
2	8.33MB/s	240ns	DMA support

Multi-word DMA is more sophisticated. Multi-word DMA means that the host withholds the DMACK signal while there is data remaining to be transferred between the host and drive. Multi-Word DMA requires a local bus controller that supports the defined **Multi-Word DMA Modes 0** and **1**.

Table B-5: Multi-Word DMA Modes

MULTI-WORD DMA MODE	AVERAGE TRANSFER RATE	CYCLE TIME	REQUIREMENTS
0	4.16MB/s	480ns	DMA support/Local-bus controller
1	13.33MB/s	150ns	DMA support/Local-bus controller

Why are there two data transfer methods supported on modern IDE drives? The main reason is to maintain backwards-compatibility for the vastly diverse IDE hardware on the market today. In the past, advances in processor and data-bus technologies have alternately made PIO or DMA the most practical way to increase data transfer speeds.

Which mode should you choose? You should enable the highest performing mode that your system supports. If your equipment is not of very recent manufacture, it may not support PIO Mode 3 or greater, or either of the Multi-Word DMA modes. Remember, the best data transfer rate your system is capable of depends on the highest PIO or DMA mode that is supported by the drive, controller, and BIOS.

Ultra DMA/Ultra ATA

Ultra DMA is the term used to describe the portion of the upcoming ATA-4 specification dealing with data transfers up to 33MB/s. The performance gains are achieved with physical improvements in the ATA interface. This means that even though an Ultra DMA drive will be backwards compatible with the existing ATA specification, specific hardware support in the BIOS and controller is required in order support the fastest transfer modes.

IDE Encoding

IDE drives use RLL encoding. See Appendix A for more information on encoding methods.

IDE Identify Drive Command

The **Identify Drive** command is supported by almost all newer IDE drives. It allows the host system to interrogate the IDE drive concerning its parameters and available data transfer modes. Automatic detection of drive type requires that the drive support this command.

Low-Level Formatting IDE Drives

All IDE drives are low-level formatted by the factory. *Reformatting these drives is not recommended as it may erase the factory defect map and may even destroy the drive.*

You may come across programs that claim to low-level IDE drives. These programs simply erase the areas of the disk that contain data. They do not re-initialize the sector boundaries. These programs can be quite helpful when nothing else will get the drive running.

IDE Drive Interleave

Modern IDE drives do not require interleaving, or, in a technical sense, IDE drives use an interleave of 1:1. Please see Appendix A for an explanation of sector interleaving.

ATAPI

ATA-2 addresses a major shortcoming of the original ATA specification in that it allows for the attachment of peripherals other than hard drives. The ATA Packet Interface (ATAPI), is incorporated into the ATA-2 specification for this purpose. ATAPI, which was derived from parts of the SCSI-2 command set, offers a more standardized way of connecting some of the most popular peripherals—making the use of proprietary adapter cards unnecessary. No other system hardware support is required—simply attach the ATAPI device to an available IDE channel and use the appropriate driver for the OS. IDE tape drives and CD players are now commonly available.

The SCSI Interface

SCSI General Description

SCSI stands for "**Small Computer Systems Interface**." The SCSI interface can more accurately be described as a bus rather than a device interface. Up to seven (fifteen on recent variations) SCSI devices, such as hard drives, tape drive units, CD-ROM drives, and even printers, can be daisy-chained together on a SCSI bus. Most device interface functions reside on the devices themselves rather than the host adapter, with the SCSI host adapter performing only limited functions as it passes communications between the bus and the host system.

All devices communicate on the SCSI bus. SCSI devices can send and receive commands and data, with one device (usually the host adapter) acting as the **initiator** and the other as the **target**. Specific functions are assigned to each role. Because the bus is shared by all devices, the initiator must “listen” to see whether the bus is in use before beginning transmission (signal lines on the cable indicate the presence or absence of activity). When ready, the initiator selects a target along the bus and sends commands to begin an operation. .

Each device on the bus is assigned a unique SCSI ID so commands can be delivered to the intended target. The intended target processes the I/O operation and, if it is a read, transfers requested data over the SCSI bus byte-by-byte. Target and initiator continue handshaking throughout the transfer to verify that it is complete and error-free. The assignment of initiators and targets is done during the initial installation. Some SCSI systems support only one initiator, while others support multiple initiators.

SCSI drives are usually found in high-end systems that require more performance than the average workstation. SCSI drives are generally more expensive than their IDE counterparts, despite the fact that both drive types are mechanically identical. The major difference in performance and cost lies in SCSI drive logic.

All SCSI devices, like IDE, are considered intelligent, meaning the logic used to control drive functions is completely contained on the devices rather than the host adapter. Unlike IDE, however, on-board SCSI devices can multitask, processing multiple I/O operations concurrently. SCSI adapters can pass multiple requests to each SCSI device on the bus, with each device managing its own read and write functions, buffering data, and interspersing transmissions with others over the bus. In contrast, IDE devices are limited to completing one operation before starting the next.

SCSI devices do not depend on the system BIOS to process instructions needed to communicate with the host system. Instead, the logic for these tasks is contained on the drive itself. INT 13h BIOS calls are redirected to the SCSI device by a software device driver or a BIOS that resides on the SCSI controller. This is one of the reasons that SCSI devices are difficult to set up. IDE devices must work with the system BIOS in order to operate, therefore installation procedures are very standardized and compatibility with other IDE devices is generally assured. In contrast, SCSI drives from one manufacturer are often not compatible with drives or controllers from another manufacturer.

The SCSI communication system gives the interface tremendous versatility. Initiators are able to direct targets to perform a wide range of diagnostic operations, such as on-board defect and error management functions, then report the results of these diagnostic operations to the host system. Some SCSI systems are also capable of configuring themselves, assigning SCSI ID numbers to each device on the bus automatically.

Overall, SCSI is the device of choice for large computer environments that utilize network servers or massive storage banks. In such environments, SCSI speed, variable device support and communications system are real assets.

SCSI Variations

The number of SCSI versions in the marketplace today can be overwhelming at first glance. However, the many versions can be easily organized by SCSI standard and physical interface characteristics, which include cable type, data path width and bus transfer rate.

The three SCSI standards in existence today are products of the American National Standards Institute (ANSI) and its subcommittees. In general, the purpose of these standards is to specify the physical interface and how information is to be transferred across that interface. The physical interface of a SCSI drive unit includes such items as cables (electrical wiring), connectors and power levels. Specifications on how information is to be transferred include command sets, messages and data transfer rules.

SCSI-1, SCSI-2, and SCSI-3

The original industry standard, SCSI-1, was published by ANSI in 1986 and is now mostly obsolete. Products developed under this standard varied considerably due to vendor's liberal interpretation, producing compatibility problems and difficulty in set-up operations. Problems were further fueled by the lack of driver specifications which resulted in too many device and host-specific driver programs.

The SCSI-2 standard addressed these compatibility issues by incorporating a **Common Command Set** and some much needed technical improvements. Hardware improvements resulted in increased transfer rates from 5 to 10 MB/sec, making SCSI-2 the standard of choice in today's marketplace.

SCSI-3, still under development, includes several projects. Only one of these, commonly known as Ultra SCSI, will be backwards compatible with SCSI-1 and SCSI-2. The other projects move from the traditional parallel bus to a new serial interface that affords much faster transfer rates, as well as longer cable lengths and more devices per bus. Serial SCSI is not yet available on the market, although some manufacturers have already incorporated Ultra SCSI into their drive products.

The term SCSI encompasses many different physical interfaces. All represent combinations of combining two different electronic cables (differential vs. single-ended) with different data path widths (8-bit or 16-bit) and clock speeds. Clock speeds control the rate at which data is transferred across the cable. The initial SCSI version, upon which the others were based, was the **Standard** or **Narrow SCSI-1.** This physical interface used an A-cable (50-lines with 8-bit data path) with a data transfer rate of 5 MB/s. Its SCSI-2 equivalent increased data transfer rate to 10 MB/s using the same cable. Some of the more prominent SCSI variations are listed in Table B-6.

Table B-6: SCSI Variations

SCSI Type	SCSI Spec.	Connector	Max. Devices	Data Path (bits)	Max. Throughput
Standard	SCSI-1 /SCSI-2	A-cable	8	8	5 MB/s
SCSI Fast	SCSI-2	A-cable	8	8	10 MB/s
SCSI Wide	SCSI-2	P-cable	16	16	10 MB/s
SCSI Fast-Wide	SCSI-2	P-cable	16	16	10-20 MB/s
SCSI Fast-20 (Ultra SCSI)	SCSI-3	A-cable	8	8	20 MB/s
SCSI Fast-40 (Ultra SCSI)	SCSI-3	P-cable	16	16	40 MB/s
SSA	SCSI-3	SSA	126	serial	80 MB/s
FC-AL	SCSI-3	Fiber Channel	126	serial	200 MB/s

Not all SCSI variations listed above are compatible. Although parallel SCSI standards tend to be backward compatible, meaning that a newer standard will encompass previous ones, several hardware variations are incompatible. Combining incompatible components can damage the system in which they are installed. Hardware variations and related restrictions are described in detail below.

Single-ended vs. Differential SCSI

SCSI standards are implemented on two electrically incompatible cables types: single-ended or differential. Drives that operate on one cable type cannot operate on the other. Single-ended cables are designed to be used for applications that require no more than 6m for the total bus length (cable). If more than 6m is required, then a differential cable (and compatible devices) must be used. Differential cables can be up to 25m in length. See **page 159** for a more detailed description of single and differential cabling.

Electronic signaling makes differential and single-ended SCSI devices incompatible, each requiring different host adapters. Attempting to connect a differential drive to a single-ended host adapter (or vice-versa) will result in damage to the system.

Standard vs. Fast SCSI

Fast SCSI is a feature of SCSI-2 that effectively doubles the average throughput over the same cable by using synchronous (vs. asynchronous) data transfer. Fast SCSI allows transfers up to 10MB/s.

Standard vs. Wide SCSI

Wide SCSI increases data transfer rates by doubling the data path of standard SCSI. Wide SCSI uses a 68-pin **P-cable** to carry a 16-bit data path (vs. 8-bits on the Standard **A-cable**). This extended width allows Wide SCSI to transfer twice as much data in the same amount of time, producing average data transfer rates of up to 20MB/s. Wide SCSI also supports 15 devices rather than the standard eight. Combining Wide data paths with Fast transfer techniques results in Fast Wide SCSI, enabling possibilities in the 80MB/s transfer range.

Ultra SCSI and Wide Ultra SCSI

Ultra SCSI is the popular term for the portion of the SCSI-3 standard dealing with advanced parallel data transfer techniques. Ultra SCSI doubles the Fast SCSI data transfer rate. This increase is primarily due to advances in the speed of SCSI chipsets which enable faster cycle times for data transfer and SCSI command processing. Ultra SCSI doubles the data transfer rates for Fast SCSI-2 (A-cable) from 10 MB/s (Fast-20) to 20 MB/s (Fast-40). Wide (P-cable) variations essentially double the throughput yet again (40 MB/s for Wide Fast-20 and 80 MB/s for Wide Fast-40). Ultra SCSI supports both single-ended and differential cable types, and is backwards compatible with SCSI-2 and SCSI-1.

Serial SCSI

The SCSI variations in the preceding sections transport signals along parallel lines. Drive manufacturers have been eyeing high-bandwidth serial techniques as the next step in advancing drive throughput because they offer faster data transfer rates, longer cables, and support more devices per cable. All the features required by today's massive networking systems.

Serial data transfer involves transporting data in packets, similar to techniques used in local area networks (LANs). Such transport allows the SCSI interface to interact with a variety of protocols, giving it the future capability of communicating with systems of servers, workstations and on-line devices not available to parallel SCSI. Unfortunately, serial SCSI will not be compatible with parallel SCSI versions.

Two SCSI-3 serial standards currently under development are **Fiber Channel** and a subset of Fiber Channel, **FC-AL (Fiber Channel Arbitrated Loop)**. The FC-AL interface is being designed to interact with networks and WANs. Transfer techniques currently in development will communicate with both storage devices and TCP/IP (the language of the Internet). Data transfer rates of Fiber Channel are expected to be 100 MB/s, while those of FC-AL are expected to range from 200 MB/s (400 MB/s in the near future). Number of devices supported is 126, with cable-lengths of 30m. using copper (10km using fiber optics).

SSA (**Serial Storage Architecture**), another SCSI-3 serial project, also offers superior performance over traditional SCSI. Transfer rate increases are primarily due to a unique interface that features two full-duplex ports. Each port is capable of carrying two 20 MB/s transmissions at once, one incoming and one outgoing. The combined channels yield a total bandwidth of 80 MB/s. In contrast, parallel SCSI and FC-AL can only transmit one direction at a time.

Like Fiber Channel, SSA also supports longer cables and more devices. Each device along the bus can be separated by as much as 25m. (The entire cable length is 6m for single-ended parallel SCSI and 25m for differential).

RAID

RAID is a general scheme to increase I/O performance by making a modular array of hard drives appear as a single storage unit. The term **RAID** stands for "**Redundant Array of Inexpensive** (or **Independent**) **Disks**." RAID is most useful in high-end environments such as those that use network file servers, where numerous I/O requests compete for drive access, or in graphics intensive environments, where sequential file retrievals monopolize I/O resources.

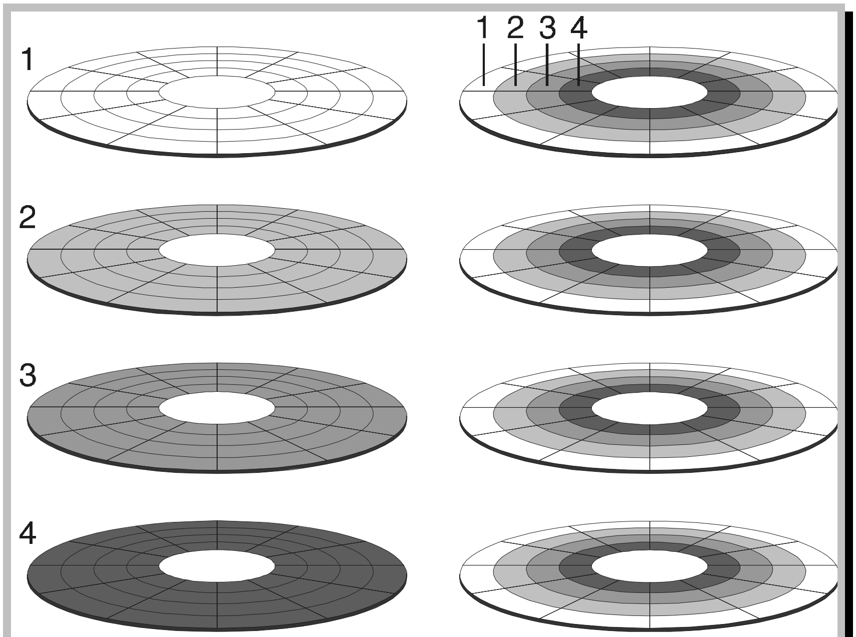
Although RAID is not intended to be interface-specific, SCSI drives are generally used by default due their availability and high performance. Typical RAID controllers support multiple SCSI interfaces that include both 8-bit and 16-bit data paths (Standard and Wide SCSI). On most external RAID cabinets, the power and interface connection between the host computer and cabinet is accomplished via a single 80-pin **Single Connector Attachment** (**SCA**) connector. Some manufacturers use proprietary connectors.

The primary advantage to RAID is that it improves I/O performance by enabling faster data transfer. The scheme consists of an array of standard hard drives installed to create one large logical drive. Data files are broken into segments and distributed throughout the array so drive heads can access these segments in parallel (simultaneously).

Another advantage to RAID is that it provides options to guard data against media failure by simultaneously duplicating complete data sets or by using parity schemes from which data may be reconstructed. Newer RAID versions allow “hot-swapping”–the ability to pull out a bad drive and install a new one while the array is still in use.

Coordinating multiple hard disks to operate as one logical unit is typically achieved through a RAID controller. These controllers typically house processors that manage data access and distribution functions, as well as parity checking functions used in data recovery. Many have their own CPU and RAID BIOS, allowing drive arrays to be initialized and managed independently of the host computer. On the lower-end, RAID controllers reside on a bus within the host computer, with attached hard drives comprising the RAID array. On high-end RAID systems, both the controller and hard drives are housed in an external cabinet. In both systems, the RAID unit operates with all internal functions transparent to the host computer.

RAID accomplishes the formation of a single logical drive from multiple units through a process called “**striping**.” Striping involves dividing individual files into segments that are then interleaved (distributed) among drives in the array. An individual file, for example, may have one segment on Drive 1, a second segment on Drive 2, and so on. The portion of the file that exists on a single drive is a stripe segment. A stripe segment can consist of a single sector or extend over multiple sectors.



The platters on the left represent non-striped disks. The platters on the right are striped. Both would contain the same information, but the striped disks have the data arrayed across all five platters, rather than sequentially arranged (Disk 1, then Disk 2, etc.).

Figure B-3: RAID Striping

The size of stripe segments has considerable effect on the performance of a RAID system. In I/O environments that process large files, such as graphics, it is desirable to select smaller stripe segments (such as a sector) so that data from a single file is distributed across all disks. All drive heads can then operate simultaneously to retrieve or write segments of the same file, increasing throughput substantially. In network environments, where there are many I/O requests for smaller files, a larger stripe segment is preferable. Smaller stripe segments would tie up all drive heads on a single operation. Larger stripe segments allow single small files to fit on individual drives, so multiple files may be accessed simultaneously across the array.

Data recovery in RAID systems offers real advantages over that in a single drive unit. When a drive within an array fails, one of two options exist depending on the RAID level:

If data has been duplicated on pairs, RAID switches data access to the duplicate drive.

If parity information has been saved, the lost data is rebuilt on a spare drive that is brought into the array. Some systems require manual intervention to add this drive and initiate the rebuild process. Newer high-end versions support back-up drives that automatically come online whenever a drive fails.

There are six levels of RAID offered, each with unique functions:

- **RAID 0** distributes (stripes) data across the array without any duplication and without parity. This RAID version is the simplest and has the fastest data transfer rates because stripe segments are transferred in parallel between array elements and the controller. RAID 0 is in effect a misnomer because it does not offer the protection of redundancy against mechanical failure. If one drive module goes down, the entire array is down.
- **RAID 1**, in contrast to RAID 0, consists of a pair of drives to which duplicate information is written simultaneously, a process known as **data mirroring**. The smallest unit of RAID 1 is two drives, called a "mirrored pair." The simplest RAID 1 scheme consists of a single mirrored pair. When more than one mirrored pair exists in the array, data is striped across drives on one half of the array, with the stripes being mirrored on the other half. If one half of a mirrored pair experiences a fault condition, the other is there for error recovery. Read operations occur in parallel in RAID 1, significantly increasing throughput. The advantages of RAID 1 are several. First, it allows parallel read operations, significantly increasing throughput. Secondly, it offers automatic, real-time backup of data. The disadvantage is that it doubles the number of drives required in order to operate.
- **RAID 2** has been superseded by RAID 3, over which it offers no advantages.
- **RAID 3** combines data striping with a dedicated parity drive for error recovery. The simplest scheme requires at least three drives: two for striped data and one for parity information. The data is striped in smaller segments and distributed across all drives so segments can be accessed in parallel. To do this, drive spindles must be synchronized. If a drive experiences a fault condition, data is recovered by performing Boolean calculations (exclusive OR) on sector parity information and information remaining on other drives. One advantage to RAID 3 is that long records can be accessed in less time because data segments are retrieved in parallel. Another is that it provides some protection against media failure. The disadvantage is that the format is a poor choice for networks, as each I/O operation occupies the entire array.

- **RAID 4** is similar to RAID 3, except that the striping process uses longer segments. This provides no advantage because each operation must access the parity drive, a sequential process which offsets any gain in parallel throughput.
- **RAID 5** does not require a separate parity drive because data and parity information are striped together across the array. This means that drives do not have to wait to store parity information (as they did in RAID 3 and RAID 4) so I/O operations can be overlapped. For this reason, larger stripe segments that confine individual files to one or two drives is preferable, permitting multiple files to be retrieved simultaneously. Stripe segment size is optional at this level, however.

History of the SCSI Interface

The SCSI interface originated as SASI (Shugart Associates System Interface) in 1979. It was one of several disk interfaces that worked at a logical level instead of the widely accepted physical level. Working at a logical level provided for a stable interface allowing physical drive access to change rapidly.

The SASI interface received only limited acceptance by the computer industry despite an attempt by Shugart Associates in 1980 to replace the IPI (Intelligent Peripheral Interface) through the X3T9 Standards Committee. X3T9 is an ANSI (American National Standards Institute) group responsible for the development of I/O interface standards. When NCR added features to Shugart's original interface in 1982, the X3T9 committee decided to start a SCSI project based on SASI.

The SCSI interface's appeal was enhanced by several significant developments over the following years. The first was the addition of optical WORM (Write Once, Read Many) commands, which meant the SCSI interface was no longer limited to hard drives. The second was the development of the NCR 5380 chip in 1984. This chip included single-ended drivers/receivers that allowed the industry to produce inexpensive SCSI interfaces. Apple Computer implemented this chip in their original Macintosh computers, which helped boost widespread acceptance for the SCSI interface.

ANSI finally approved SCSI-1 as a standard (ANSI X3.131-1986) in 1986, however it was too loosely defined. To address its limitations, a SCSI-2 project was immediately initiated to incorporate technical improvements and a Common Command Set.

After many years of delay, the SCSI-2 standard (ANSI X3.131-1994) was approved in 1994. In addition to addressing the need for compatibility among manufacturers' products, this standard contained significant technical improvements over SCSI-1. Among these were optional wider data paths, faster transfer speeds, command queuing and an expanded ability to connect to a wider variety of devices.

Several new SCSI projects are currently under development, comprehensively covered by the SCSI-3 set of standards. Ultra-SCSI, already on the shelves, doubles the transfer rates over SCSI-2 by decreasing time required to process SCSI commands and transfer data across the SCSI bus. It is based on the traditional parallel bus and is backward compatible with earlier SCSI versions.

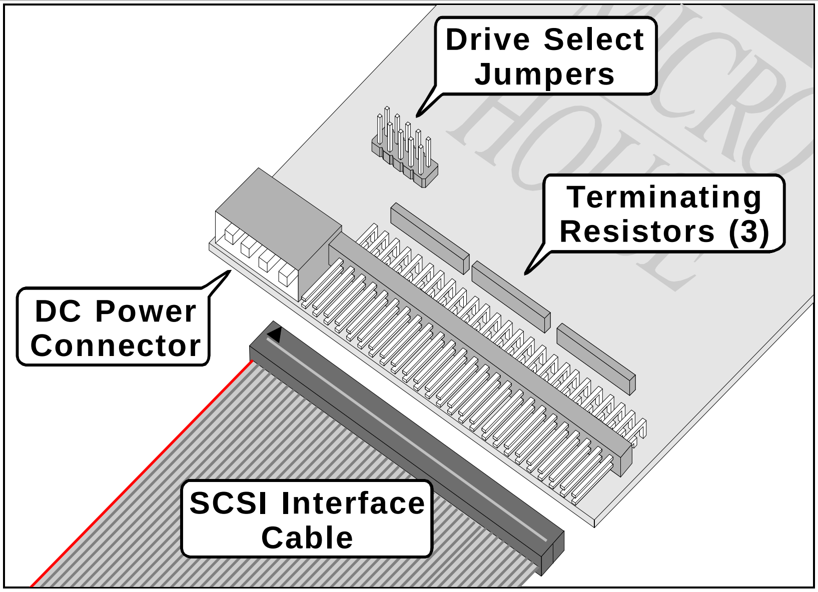
Other SCSI-3 projects address serial interfaces: Serial Storage Architecture (SSA) and Fiber Channel. Serial technologies offer faster transfer, however they will not be compatible with the parallel interfaces of SCSI-1 and SCSI-2. Complete SCSI standards documents can be obtained from ANSI.

SCSI Physical Characteristics

SCSI drive units vary in physical dimensions, but are commonly seen in 3.5 in., third or half-height formats in order to fit in the drive bays of most PC casings being manufactured today. Mechanically, the drive units are identical to their IDE counterparts. Refer to Appendix A for a description of these components. This section focuses on the external components of the SCSI drive, those found on the underside of a typical casing.

There are four basic components found on most SCSI drives. These include a cable interface, power connector, jumper block and optional terminating resistors. A common placement of these components is shown in Figure B-4. Component placement on recent hard drives, such as SCSI-2 Fast/Wide, differ slightly with the jumper block placed between the interface cable and power connector, or on the side opposite the SCSI connector.

Several differences between SCSI and IDE components should be noted. The first is cable width. SCSI includes a wider signal band, which varies to accommodate either an 8-bit or 16-bit data path width. The second includes the jumper block and on-board terminators. Both provide functions unique to drives that "share" a common bus (cable).



This diagram depicts the under-side of a typical SCSI drive. This layout will vary from drive to drive, but the basic connections will always be present.

Figure B-4: SCSI Connection Diagram

SCSI Bus Configuration, Termination and Terminating Resistors

Because the SCSI bus is shared by the drives on the device chain, all drives must define themselves with respect to others and perform certain functions depending upon their physical position on the chain. All drives must adhere to the same rules in initializing jumper settings and on-board terminators. Three rules are presented here: 1) each SCSI drive must define a unique SCSI ID number for itself so its communication can be delivered correctly, 2) beginning and ending drives on the bus must terminate the end of the bus so signals do not bounce back and interfere with incoming data, and 3) at least one SCSI drive must supply termination power (TERMPWR) to the SCSI cable. These rules affect how both terminators and jumpers are initialized.

The SCSI bus can support a maximum of eight devices (Narrow SCSI) or sixteen devices (Wide SCSI), with the bus running continuously from one device to the next. Jumpers on the underside of each drive allow the device to select a unique SCSI ID number for itself so communications can be delivered. The assigned SCSI ID is independent of the drive's position on the chain, but does determine drive priority.

If the drive is the first or last device on the chain of SCSI devices, the drive must terminate the bus. This can be done in several ways, and varies from manufacturer to manufacturer, and drive to drive. On many drives, a network of terminating resistors controls signal fluctuations at bus ends. This type of termination is called **passive termination**. If such terminators are present on your drive, they will be on the underside of the drive near the drive header cable. There will be several and they are usually colored yellow and sometimes black or blue. These terminating resistors must be physically installed on your drive if it is either the first or last drive on the SCSI bus. They will absorb electrical signals at the bus end and prevent loss of data. The terminating resistors must be removed if the drive is in the middle of the chain to permit signals to pass. Terminating resistors are found on SCSI-1 and some SCSI-2 (narrow bus) drives.

On some drives, the terminating resistor packs are not socketed or intended for removal. If this is the case on your drive, the resistors can usually be enabled internally by jumper or switch settings.

Newer SCSI drives use **active termination**, a method that eliminates electrical back-flow through the use of voltage regulators. This type of termination is effective over longer distances and is required for Fast or Wide SCSI. Active termination is controlled by a switch or jumper mounted on the underside of the drive, usually part of the jumper block. If your drive is installed at either end of the SCSI bus, termination must be enabled. Otherwise, termination must be disabled. Termination settings are described in detail in the Chapter Eight.

Drives that use passive or active termination may draw terminator power (TERMPWR) from a line on the SCSI bus to supply on-board resistor circuits or set jumpers that control termination. At least one device on the SCSI bus is required to supply power to this line–usually it is the adapter.

In a third type of termination, Forced Perfect Termination (FPT), manufactured terminators are inserted into SCSI connectors to end the SCSI bus.

Termination of a simple bus configuration, one that consists of an adapter card and one drive, is illustrated in Figure B-5. Each device lies at the end of the SCSI bus and therefore must terminate the bus. (Remember that the SCSI adapter card is also a device.) For purposes of illustration, terminating resistors were used.

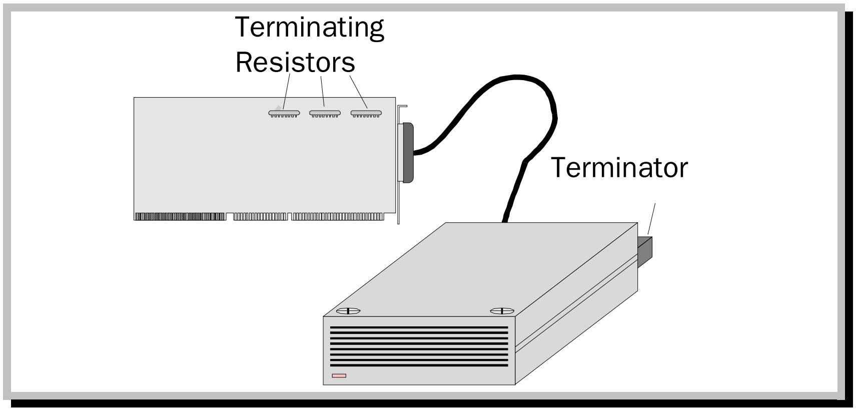


Figure B-5: Terminator configuration (Single Internal or External Drives)

Figure B-6 illustrates the bus configuration when a second device is added between the adapter and original drive. Terminators on the middle drive must be removed (or disabled) to allow electrical signals to pass along the bus.

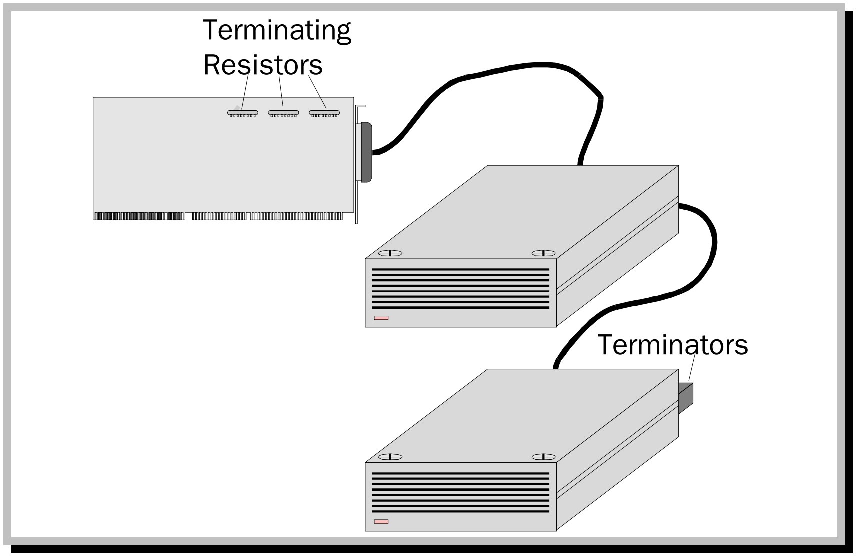


Figure B-6: Terminator configuration (Multiple Internal or External Drives)

Some adapter cards have both an external and internal connector, as shown in Figure B-7. In many cases, these connectors will share the same bus (channel). Thus, when both external and internal drives are attached to this type of adapter card, the card lies in the middle of the chain and therefore must not terminate the bus. On-board terminators should be removed or disabled, as shown. If only external devices (or internal devices) are attached, then the adapter lies at one end of the bus so its on-board termination must be installed or enabled.

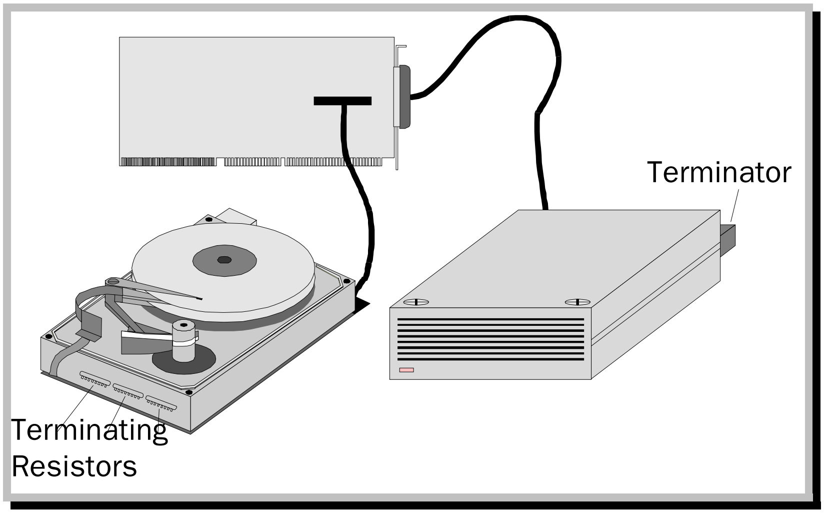


Figure B-7: Terminator configuration (Internal and External Drives)

Some external and internal connectors will reside on separate buses (channels). This means that the card supports two distinct device chains so that the card always resides at the end of each chain. In this case, termination is permanently set on the card.

Many recent adapter cards support multiple buses. This is especially common on RAID adapter cards, where the external connector usually shares a bus with one internal connector, and the other internal connectors have their own bus line. In this case, each individual bus must be separately terminated.

Jumper Block

Common functions that can be selected via the jumper block include the following:

- SCSI ID assignment
- Termination
- Provision of termination power to SCSI bus
- Staggering of drive power-up along SCSI bus
- Synchronization of drive spindles
- Write Protection

Several of these functions address how a drive interacts with others on the bus. Some functions permit the drive to define unique characteristics for itself. These and the other functions are discussed in detail in the Chapter Eight.

SCSI Cabling

The most common SCSI cable types in use today are A-cable and P-cable. These cables are further divided into two electrically incompatible versions: single-ended and differential.

Single-ended cables account for the vast majority of SCSI bus configurations. They are the standard cables that generally allow total bus lengths of up to 6m. The main reason for the length restriction is to ensure that all devices are receiving clean signals. Electrical signals can be degraded by radio interference that may exist in the environment. The longer the cable, the more interference the signals are exposed to. Keeping the total length of the bus under 6m is generally enough to keep outside interference from becoming a problem.

Some applications may require the total length of the bus to be more than 6m. In this case, some type of protection or tolerance must be provided in the cabling. Differential cables are used for this purpose, allowing the total length of the bus to be up to 25m. Differential cables use twice the signals of the single-ended versions. The second set of signals are inverted from their counterparts in the first set of wires. The SCSI device at the receiving end takes the difference (differential) between the two signals to determine the intended input. Any noise introduced into the line between the sending and receiving devices affects both lines equally and is therefore canceled out by the differential process.

Single-ended and differential cables and devices are electrically incompatible. They cannot be interchanged, *combining differential and single-ended components can result in damage to the system.* Single-ended and differential cables look similar because half of the wires on the single-ended version are unused.

The single-ended version of the **A-cable** has 50 electrical lines that include an 8-bit data path and optional parity bit. The remaining lines are dedicated to control signals, termination power and ground. Drives utilizing the single-ended A-cable are referred to as Narrow or Standard SCSI and those that utilize differential A-cable are referred to as Differential SCSI.

The P-cable is used for SCSI devices that use a 16-bit data path. Its use is defined in both the SCSI-2 and Ultra SCSI (SCSI-3) specifications. Drives utilizing the single-ended P-cable are called Wide SCSI and those that utilizing differential P-cable are Differential Wide SCSI. Both versions are electronically incompatible and combining components can result in damage.

The SCA cable is an 80-pin cable capable of carrying several different data paths. It is used almost exclusively in RAID systems (see **page 149**), where a variety of drives are connected in massive storage systems.

SCSI Technical Information

SCSI Drive Addressing

SCSI drives access data through **Logical Block Addressing** (**LBA**). This means that the LBAs, or sectors, are ordered sequentially, instead of the three-dimensional **CHS** addressing format used in drives using other interfaces. See Appendix A for more information on CHS and LBA addressing.

To find the capacity of SCSI drives, simply multiply the number of sectors by the sector size (usually 512 bytes). For example:

1,032,192sectors x 512bytes per sector = 528,482,304bytes

Although SCSI drives "think" in terms of LBA, the host system BIOS does not usually support this type of addressing. Generally, SCSI adapters bypass the host system BIOS either through a BIOS-ROM installed on the adapter itself, or through a replacement block device driver that is specific to the adapter model.

SCSI controllers with a replacement BIOS ROM perform a translation from CHS into LBA in order to remain compatible with standard BIOS INT 13h software interrupts. This means that in this case the drive is still subject to the limits imposed on CHS parameters by the bits allocated to INT 13h. Therefore SCSI drives using this type of controller are limited to approximately 8GB, the same as Enhanced IDE drives. See page 137 for information on BIOS-imposed CHS limits.

BIOS Drive Type Selection

Almost all SCSI controllers do not use the host system BIOS, therefore the drive type selection in the CMOS setup should be set to “0” or “Not Installed.”

Low-Level Formatting

All newer SCSI drives are low-level formatted by the factory. Reformatting these drives is not usually necessary.

Low-level software for SCSI drives is specific to the host adapter model because of wide variations at the register level. The low-level program is usually resident in the adapter’s firmware. Consult the user’s manual that came with your SCSI adapter for information on accessing the low-level program.

SCSI INT 13h Handling

When implemented in a DOS-based system, SCSI drives access the host system through INT 13h to remain compatible with the manner in which hard drives were intended to be interfaced through the AT BIOS. Usually, the programs necessary to access the hard drive reside in the computer’s BIOS and are invoked through this software interrupt. In the case of SCSI drives, the host BIOS is bypassed by the firmware on the SCSI controller or the loaded SCSI device driver which redirect INT 13h calls to themselves. This is the reason that the “Drive Type 0” or “Not Installed” is selected for most SCSI drives in the CMOS setup.

Caching

SCSI drive intelligence usually includes caching algorithms that speed processing by utilizing look-ahead buffering. This allows the drive to not only read the portion of data requested by a specific command, but read remaining portions as well and store these on-board, in expectation of forthcoming requests for the data. The cache is discarded if no such requests are received.

SCSI controllers with on-board cache are common because they are used in high-performance environments where cost is generally less of a consideration than it is for IDE systems.

Interleave

Modern SCSI drives do not require interleaving, or, in a technical sense, SCSI drives use an interleave of 1:1. Please see Appendix A for an explanation of sector interleaving.

SCSI Adapters and Drivers

SCSI adapters provide the interface between the SCSI bus and the host system. These adapters, like IDE, perform minimal drive controlling functions because the drives are intelligent and perform many of these functions themselves. SCSI adapters, unlike IDE, do not depend on the interfacing programs resident in the computer's BIOS. Instead, they have their own method of communicating with the host system.

A lack of standardization in the past gave early manufacturers of SCSI adapters more freedom in choosing how their products accomplished the interface. As a result, the software used to set up SCSI adapters, such as device drivers, was often device-specific. Driver support for hard drives was (and still is) typically incorporated into the SCSI adapter BIOS. However, driver support for many peripherals types was unique to the device and externally installed. Unfortunately, adapters did not necessarily support all manufactured devices.

Several device driver standards were developed to address this problem. One popular standard is **Advanced SCSI Programming Interface** (**ASPI**), initially developed by Adaptec. The intent of ASPI is to standardize SCSI device communications so that a device driver for any ASPI-compatible peripheral will work with any ASPI-compatible adapter, regardless of the manufacturer. Another standard with similar intent is the **Common Access Method** (**CAM**), developed by ANSI. CAM allows devices to interact with Unix as well as other common operating systems.

Appendix C

Frequently Asked Questions

Following are some of the problems and solution suggestions for problems the user might experience when using DrivePro. There is also a section explaining some of the common error messages and their solutions.

For the most current information available, please visit www.solution.microhouse.com/support/.

FAQ INDEX

Frequently Asked Questions

There is no matching type in BIOS for my IDE drive.

If the BIOS has a user-definable drive type and the total cylinders of the drive are under 1024 (and sectors 63 or less and heads 16 or less), use the physical parameters. It is preferable to use the physical parameters because they minutely enhance drive access speed since a translation is not required, and are probably the first parameters that someone trying to recover the drive will guess to be parameters used to install the drive.

If the BIOS does not have a user-definable drive type or the cylinders of the drive are over 1024 (or other DOS limitations are exceeded), you may select the closest fitting table entry that does not use the entire disk, or install EZ-BIOS. Note that you must not set the number of cylinders greater than 1024 with any version of DOS, unless you use EZ-BIOS or a replacement block device driver for DOS that supports over 1024 cylinders. Exceeding DOS limitations without one of the above installed may result in the loss of all data on the drive! In addition, older system BIOSes cannot handle more than 63 sectors or 16 heads, unless your controller has an on-board BIOS that allows greater values. This is because of older 80x86-standard BIOS limitations.

There are basically two software methods used to exceed the standard 80x86 BIOS cylinder, head, and sector limitations.

1) You can use a translated set of parameters that equal the same capacity without exceeding these limitations, or

2) use the best matching BIOS entry, choose the entry that is equal to or less than, but never more than, the drive's capacity in megabytes.

Other options are hardware ones, including upgrading your system BIOS, a controller with the appropriate BIOS, and purchasing an EPROM programmer (burner) to physically alter the BIOS.

IDE drive was moved to another system or computer lost its CMOS setup. Now the IDE drive won't work properly.

The parameters for the IDE drive must be set identically as when the computer was initially set up. Note that different parameters can yield the same capacity, so using just any parameters that yield the correct capacity does not necessarily mean you are using the correct parameters. To determine from the drive's partition table the original parameters used to setup the drive, use DrivePro's **Get Lost Drive Parameters** feature (under the **Utilities** menu). Upon determining the correct parameters, either:

1) match them to a drive type in the CMOS, or configure the CMOS user-definable or

2) install EZ-BIOS with the correct parameters without erasing the partition table.

The drive may then be used in any machine as long as the CMOS is set to a value besides 0 or not installed.

CMOS shows drives capacity as less than it should be.

CMOS and FDISK calculate drive capacity by the formula HEADS x CYLINDERS x SECTORS / 2048. DIR, CHKDSK, and DrivePro use the formula HEADS x CYLINDERS x SECTORS x 512. CxHxSx512 yields decimal number of bytes, which when divided by 1,000,000 gives you decimal Mbytes (million bytes). (CxHxS)/2048 yields binary megabytes (2^{20} bytes). There are 1,000,000 bytes in each decimal Mbytes. There are 1,048,576 bytes in a binary megabyte. Therefore, if the CMOS's value for the capacity of the drive is less by ~5% than some other indicator, the difference is due to different calculation methods.

How do I remove EZ-BIOS and just get back to a standard setup?

Go to the **Utilities** menu and select **EZ-BIOS Setup**. Select the drive you want to remove EZ-BIOS from and choose **Uninstall EZ-BIOS** from the menu. Note that if you are using EZ-BIOS to enable the use of a >528Mbytes drive (native BIOS doesn't support it), then you will lose access to the data on that drive once EZ-BIOS is removed.

My BIOS has no provisions for displaying the drive type table during boot-up.

Pull down the **Edit** menu and select **BIOS Drive Table Viewer** to display the contents of your system BIOS Drive Table.

What BIOS Drive Type is used for SCSI drives?

In most installations set the type to “0” or “none,” and the SCSI controller and drive will take it from there. Also, its a good idea not to low-level the drive unless the controller came with its own firmware or software to do so.

Drive won't boot after FDISK and DOS format.

This is a problem that arises with some IDE drives. DOS appears to have some problems marking the first partition of a drive “active.” The first partition must be marked active for it to boot. FDISK is supposed to do this for you on the first partition but it sometimes fails to do so. The most problematic DOS version is 3.3. You can fix this problem by running FDISK and setting the first partition to active, or by using DrivePro's **Partition Table Editor** and marking the first partition Bootable: "YES."

Unable to delete a Novell or other non-DOS partition or unable to DOS format such a partition.

This is particularly troublesome with IDE and SCSI drives due to the fact that they usually cannot be low-level formatted. Use DrivePro's **Partition Table Editor** or **Sector Editor** to edit or delete a non-DOS partition table entry.

I get a message "HARD DRIVE CONTROLLER FAILURE" after installing a second IDE drive.

You should carefully check and reconfirm the jumper settings and check or switch out the cabling, and try changing such jumpers as I/O Channel Ready and the Drive Slave Present jumpers (consult the drive manufacturer's Technical Support for additional help with this). You can also try switching the two hard drives around, setting the jumpers on the Master drive to be the Slave, and on the Slave drive to be the Master. Also, due to the proprietary beginnings of the IDE interface, older IDE drives are very frequently incompatible with IDE drives from other manufacturers. This is especially true of some older Conner and Seagate drives. The solution to this problem when adding a second IDE drive is to use drives of the same make, or alternatively, using the included utility ADDADRV to allow the second drive to be used on a separate controller card.

Ran FDISK on an IDE drive several times, but it won't save the partition tables.

Translation values into the CMOS setup that are greater than the capacity of the IDE drive or parameters that the IDE drive does not support can cause this problem. Try another set of heads, cylinders and sectors. Be sure that you don't go over the drive's total capacity in megabytes. You should also recheck jumper settings and check or switch out the cabling.

Two IDE drives will not work together even though I have the Master/Slave settings set correctly.

Check or switch out the cabling. Recheck the I/O Channel Ready (IORDY) and Drive Slave Present jumpers (contact the drive or controller manufacturer's Technical Support for more information). In some cases both IDE drives must be of the same make in order to work together. If the two IDE drives work independently of each other, try switching the drives in terms of Master and Slave.

Installed an IDE hard drive, now the floppy drives don't work.

If you have installed an IDE drive and now the floppy drives don't work or even light up during boot, then you have installed the IDE cables backwards.

Many IDE drives do not show the location of pin-1. If you can't locate pin-1, then a good rule-of-thumb is to assume pin-1 is the closest pin to the DC power connector.

Drive doesn't spin.

1) Check that drive has power supplied to it.

2) Some of the older hard drives have a problem with the media becoming sticky after years of use. The heads get stuck to the platter and prevent it from spinning. To get them spinning again, make sure you touch a good ground (like bare metal on the case of your computer if it is plugged into a good 3-prong outlet), remove the drive from the case, turn the drive over and slightly turn the spindle (be careful!) to free it. It should then work long enough to make a backup. Get your data off quickly and replace the drive!

Getting intermittent operation errors or read/write errors at random.

Ensure that the termination resistors on the hard drive(s) are properly placed. Also ensure that the power supply can support the added hard drive and voltages from the power supply are within range.

This page intentionally left blank.

Appendix D
Glossary of Terms

A

ACTUATOR

The device that moves the read/write heads across the platter surfaces. There are two kinds of actuators that are commonly used, stepper motor actuators and voice-coil actuators. Today all new drives use the voice-coil actuator because it is the faster and sturdier of the two. To give you an idea of the speed difference, a typical stepper motor drive takes from 65 to 100 milliseconds (65 to 100 thousandths of a second) to move from one track to another, as compared to the voice-coil actuator that usually only requires 10 to 20 milliseconds.

ALLOCATION UNIT

See CLUSTER.

AREAL DENSITY

Areal density tells how densely packed data is on a disk surface. Areal density is the product of bit density (bits per inch, or BPI) multiplied by track density (tracks per inch, or TPI), or bits per square inch of the disk surface. Bit density is measured around the outside (circumference) of a track, and track density is measured from the center to the outside (radius) of a track.

ATA

AT Attachment interface

See IDE.

ATA-2

Updated AT Attachment interface.

See ENHANCED IDE.

AT INTERFACE

See IDE.

ATAPI

ATA Packet Interface

ATAPI defines a set of commands supported through the ATA-2 interface that are used for peripherals other than hard drives, such as CD-ROM drives and tape drives.

AVERAGE LATENCY

A measurement of how long a drive must wait before a specified bit of data rotates under the heads. An average figure is one half a platter rotation.

B

BCAI

Byte Count After Index

Used in defect mapping to indicate the position of defects with relation to index.

BFI

Bytes From Index

Used in defect mapping to indicate the position of defects with relation to index.

BIOS

Basic Input/Output System

The **BIOS** is the interpreter that sets up a specific computer to allow the operating system (such as DOS) to communicate with peripherals. The BIOS can be thought of as the "glue" that holds together hardware made by a variety of manufacturers. The BIOS is firmware, that is, the instructions are permanently written on a Read Only Memory (ROM) chip. The BIOS contains the parameters required to initialize (boot) the computer, and performs a Power On Self Test (POST) during the boot sequence.

BIT DENSITY

or Bits Per Inch

Expressed as "BPI" (bits per inch), bit density defines how many bits can be written onto one inch of a track on a disk surface. Bit density is measured around the outside (circumference) of a track. It is usually specified for "worst case," which is the inner track. Data is the densest in the inner tracks where track circumferences are the smallest. See Track Density, Areal Density.

BLOCK

A group of bytes handled, stored, and accessed as a logical data unit, such as an individual file record. Typically, one block of data is stored as one physical sector of data on a disk drive.

BPB

BIOS Parameters Block

An area of memory reserved by DOS for drive parameters. Most of the numbers in this table are obtained from the DBR.

Also see DBR.

BPI

Bits Per Inch

See Bit Density

BUFFER

All modern hard drives have some amount of on-board memory, which is termed the buffer. The buffer is a way-station for requested data after it is read from a location on disk. The advantage to a disk buffer is that it decreases system delays due to the physical limits of the drive speed. Read sequences can be sped up by having the buffer hold information that it anticipates the system will request.

C

CACHE MEMORY/CACHING

Caching is the process where the system loads data from the hard disk to the **RAM** set aside as **cache memory**. As long as the same file is being accessed, the system may refer to the **cache memory** for information instead of going back to the hard disk, thereby increasing the processing speed. Caching does not really increase seek times, but makes the drive appear faster by eliminating repetitious access. Cache memory can either be dedicated or set aside from host RAM. Modern SCSI and IDE hard disk controllers are equipped with their own cache memory.

CAPACITY

The total capacity of a hard drive in bytes is found by using the formula (Cylinders x Heads x Sectors per Track x 512). Divide this number by 1,048,576 to obtain the capacity in binary megabytes. Note that the formatted (usable) capacity is even less due to space taken up in defining sector boundaries. For example, the Maxtor XT-4380E has 384 megabytes of unformatted space but only 319 megabytes of formatted space available to the user. This is a 65 megabyte difference.

CHS

Cylinders, Heads, Sectors per track

The parameters used to determine the capacity of a hard drive. Physical CHS parameters are the actual number of cylinders, heads, and sectors per track under the casing. Logical CHS parameters are translations used to bypass parameter limits imposed by the operating system.

CLUSTER

An operating system term describing the number of sectors that the operating system allocates each time disk space is needed. For example, if the cluster size is 16 (Thirty-two, 512-byte sectors per cluster) then every file will use 16K, even though the actual file size may be less. Clusters are sometimes called “allocation units.”

CMOS

Ceramic Metal Oxide Semiconductor

A type of RAM chip that is often used to hold BIOS setup information.

Also see BIOS.

CONTROLLER

The printed circuit board required to interpret data access commands from host computer (via the bus), and send track seeking, read/write, and other control signals to and from a hard drive.

CPU

Central Processing Unit

CRC

Cyclic-Redundancy-Check

Used to verify data block integrity. In a typical scheme, 2 CRC bytes are added to each user data block. The 2 bytes are computed from the user data by digital logic chips. The mathematical model is polynomials with binary coefficients. When reading back data, the CRC bytes are read and compared to new CRC bytes computed from the read back block to detect a read error. The read back error check process is mathematically equivalent to dividing the read block, including its CRC, by a binomial polynomial. If the division remainder is zero, the data is error free.

CYLINDER

The cylindrical surface formed by identical track numbers on vertically stacked disks. At any location of the head positioning arm, all tracks under all heads are the cylinder. Cylinder number is one of the three address components required to find a specific address, the other two being head number and sector number.

D

DATA COMPRESSION

Data compression is an encoding process where data strings that are repeated often are cut out when written to disk, then replaced before being used by an application.

DBR

DOS Boot Record

A table that describes a logical volume. The DBR is located in the first sector of the volume, and contains information that includes the DOS version it was formatted under, type of file system, and the volume label.

DMA

Direct Memory Access

A method used to speed up data transfers between peripherals (such as hard drives) and RAM. As the name denotes, DMA avoids the delays imposed by the CPU's data-flow regulation, accessing RAM directly.

DTR

Data Transfer Rate

Speed at which bits are sent: In a disk storage system, the communication is between CPU and controller, plus controller and the disk drive. Typical units are bits per second (BPS) or bytes per second.

DRIVE SELECT JUMPERS

These set the control channel for the hard drive so that the controller knows which drive it's controlling. See the section on hard drive installation and your hard drive manual for the proper setting of these jumpers.

DRIVE TYPE

A number representing a standard configuration of physical parameters: cylinders, heads, and sectors per track of a particular type of disk drive. Each AT system BIOS contains a list of drive types that the system considers "standard types." These types are not necessarily the same from one BIOS to the next.

DROP-IN/DROP-OUT

Types of disk media defects are usually caused by a pin size hole in the disk coating. If the coating is interrupted, the magnetic flux between medium and head is zero. A large interruption will induce two extraneous pulses, one at the beginning and one at the end of the pin-hole (2 DROP-INs). A small coating interruption will result in no playback from a recorded bit (a DROP-OUT).

E

ECC

Error Correction Code

The ECC hardware in the controller used to interface the drive to the system can typically correct a single burst error of 11 bits or less. This maximum error burst correction length is a function of the controller. With some controllers the user is allowed to select this length.

EIDE

Enhanced Integrated Drive Electronics

See IDE

ESDI

Enhanced Small Device Interface

The ESDI standard was approved by the ISO (International Organization for Standardization) as ISO 10222:198x, and X3T9 standard X3.170. This standard has been abandoned by hard drive manufacturers in favor of the SCSI and ATA (IDE) interfaces.

ESDI can transfer data up to 24 megabits per second, although most transfer at 10 megabits. It is an “intelligent” controller that can potentially handle hard drives, floppy drives, tape backup units, CD-ROM and direct file transfer between these devices. This interface also performs much better error checking than the ST506 standard. Used on most of the large capacity hard drives.

Usually the drive type is set to 1 (one) and the on-board BIOS on the ESDI interface handles any compatibility problems. When implemented in a DOS system, access to the host system is done through software interrupt 13h (INT13h).

EXTENDED PARTITION

The secondary DOS partition on a drive. The Extended Partition cannot be high-level (DOS) formatted into a logical volume—it must be further divided into at least one logical drive.

Also see Logical Drive, Logical Volume, Partition.

F

FAT

File Allocation Table

What the operating system uses to keep track of which clusters are allocated to which files, and which are available for use.

FCI

Flux Changes Per Inch

Synonymous with FRPI (flux reversals per inch). In MFM recording 1 FCI equals 1 BPI (bit per inch). In RLL encoding schemes, 1 FCI generally equals 1.5 BPI.

FLUX CHANGE

Location on the data track, where the direction of magnetization reverses in order to define a 1 or 0 bit. See FCI for more information.

FM ENCODING

Frequency Modulation Encoding

This is an outdated encoding scheme that is no longer in use. It used up to half of the disk space with timing signals for the encoding process. The technology was refined and replaced with a new standard called MFM encoding. See MFM, RLL Encoding.

FORMAT.COM

The DOS program that is used to high-level format a hard drive. This program will low and high-level format a floppy diskette.

FORM FACTOR

This is the hard drive’s physical external size and the mounting space it will take up. Usually 3 1/2 inch or 5 1/4 inch for desk-top computers, and 2.5 inch for portables and laptops.

G

G

A “G” is a unit of force applied to a mass at rest equal to the force exerted on it by gravity. Hard disk drive shock specifications are usually called out in Gs. A shock specification of 40 Gs non-operating means that a drive will not suffer any permanent damage if subjected to a 40 G shock. This is roughly equivalent to a drop of the drive to a hard surface from a distance of 1 inch.

GIGABYTE

A unit of storage equal to exactly 1,073,741,824 bytes (roughly 1,000 megabytes). Some of the newer hard drives and optical drives are in the gigabyte range.

H

HEAD

There are usually several read/write heads in a hard drive. On larger drives a single head and platter surface is reserved for the servo. This keeps track of where the rest of the heads are positioned (See Voice-Coil Actuators). So the term “heads” usually only pertains to actual data heads, those that read/write. Such is the case with the drive listings in this guide. There may be eight platter surfaces but only seven data heads. Only one of the several read/write heads is in use at any one time. The controller can only handle data from one head at a time, which does not result in any loss of speed as the computer can only take in so much data at one time.

HEAD CRASH

As a normal operation, head landing occurs when the disk drive is turned off. When the heads land, a thin film of lubricant on the disk surface protects the disk from damage. A head crash occurs when the head and disk damage each other during landing because of rough handling, or because a small particle gets between them. Head crash is a catastrophic failure and causes permanent damage and loss of data.

HEAD LANDING AND TAKEOFF

In Winchester drives, the head is in contact with the platter when the drive is not powered. During the power up cycle, the disk begins rotation and an “air bearing” is established as the disk spins up to full speed. This air bearing prevents any mechanical contact between head and disk.

HEAD PARKING

Parking the heads places them in a safe zone away from data on the platter, usually the highest cylinder on the drive. When a stepper motor drive is turned off, its heads land on the platter wherever it was last doing a read/write. If the drive is jolted, data under the heads can be lost. The stepper motor drive must be parked through software that places the head at a safe zone. Voice-coil drives automatically park themselves due to the design of the solenoid-spring mechanism (see Voice-Coil Actuators). Head parking software is not needed on these drives and should not be used.

HEIGHT

The drive's height. See Form Factor.

I

IDE

Integrated Drive Electronics

Also known as the ATA interface, IDE is the most popular hard drive interface on the market today. The hard drives used with this type of interface are “intelligent” devices that have most of the controller functions built into the drive circuitry. These drives link up to the computer via a 40-pin connector (IBM uses a 44-pin or 72-pin connector) and a small paddle card, or directly to the motherboard on some newer computers.

The latest version of IDE is sometimes called Enhanced IDE (EIDE) or ATA-2. Improvements include capacities of up to 8.4GB, increased data transfer rates, more interfaces, and a greater variety of attachable devices.

ID FIELD

See Sector Header.

INDEX PULSE

The Index Pulse is the starting point for each disk track. The index pulse provides initial synchronization for sector addressing on each individual track.

INDEX TIME

The time interval between similar edges of the index pulse, which measures the time for the disk to take one revolution. This information is used by a disk drive to verify correct rotational speed of the media.

INTERLEAVE

Most hard drives can transfer data faster then the CPU or computer can accept it. Interleaving alleviates this problem by forcing the drive to read the data a little slower. This is accomplished by renumbering the sectors. Instead of going from 1 to 17 sequentially, they are staggered.

If the interleave is 3, then the sectors are renumbered: **1**-7-13-**2**-8-14-**3**-9-15-**4**-10-16-**5**-11-17-**6**-12. This allows the CPU to store the data for sector #1, while #7 and #13 are under the head, and then continue with sector #2 when it's ready. In one revolution, approximately six sectors of 512 bytes will have been read and stored by the CPU. In three revolutions, all of the sectors will have been read and stored. Most PC/XTs use an interleave of three or four. Most PC/ATs use an interleave of 1 or 2. It really depends on the speed of the CPU and the controller electronics. An interleave of three should be fine for most XT installations.

IOCHRDY

Input/Output Channel Ready

A signal sent by the CPU to a peripheral to let the peripheral know that more of the CPU's resources are available for data transfer.

Also see PIO.

IPI

Intelligent Peripheral Interface

A hard drive interface used with the larger 8" and 14" mainframe and minicomputers.

L

LANDING ZONE

This is the section of the disk that is designed as the safe zone for head parking (see Head Parking).

LATENCY (ROTATIONAL)

See Average Latency.

LBA

Logical Block Addressing

LBA is an alternative disk-accessing method consisting of a logical address number, rather than CHS information. LBA requires a BIOS and drive that support this access method. The BIOS translates CHS parameters from the drive during initialization. All communication between the operating system and drive from that point is accomplished with LBA.

LOGICAL DRIVE

A further division of an Extended Partition.

LOGICAL VOLUME

What the operating system perceives as a drive (C:, D:, E:, etc.) is a logical volume. A high-level (DOS) format creates a logical volume within the boundaries of a partition or logical drive.

Also see Logical Drive, Partition.

LOW-LEVEL FORMAT

The first step in preparing a drive to store information after physical installation is complete. The process sets up the “handshake” between the drive and the controller. In an XT system, the low-level format is usually done using DOS's debug utility. In an 80x86 system, 80x86 advanced diagnostics is typically used (most of the newer “clone” 80x86 BIOSes have a built-in low-level program). Other third party software may also be used to do low-level format on both XTs and 80x86 systems.

Do not low-level IDE drives unless specifically told to do so by the manufacturer! These drives have been factory low-leveled and damage may occur if the low-level is re-done.

LUN

Logical Unit Number

Also called the SCSI ID number. The units device number on a Daisy Chain. Each device on the chain must have its own unique LUN.

Also see SCSI.

M

MBR

Master Boot Record

A program that exists on the first physical sector on a drive that is used to boot the system. The main Partition Table is considered part of the MBR.

Also see Partition Table.

MBYTE

As defined in this text, one million (decimal) bytes. Due to a lack of industry standardization, some organizations may use a different definition. Also see Capacity, Gigabyte, Megabyte.

MEDIA DESCRIPTOR BYTE

An entry in the FAT and DBR used by DOS and other applications to tell the type of storage device the FAT is installed on. For hard drives, this number is F8h.

MEGABYTE

As defined in this text, exactly 2^{20} (binary) bytes. Also see Capacity, Gigabyte, Mbyte.

MFM ENCODING

Modified Frequency Modulation Encoding

A type of encoding scheme that converts the digital bits from the computer into a pattern of magnetic changes or “flux reversals” that are then stored on the hard drive. MFM uses a fixed length encoding scheme. Flux reversals on the disk always will be evenly spaced in time so that the beginning of one bi can be separated from another. This type of scheme allows even single bit errors to be detected easily and corrected by the controller electronics. The encoding/decoding process is done on the controller card not at the hard drive. MFM is modified from FM, doing away with the need for timing signals. This allows twice as much data on the same drive. MFM has in turn been replaced by RLL for encoding data on hard drives. Also see FM Encoding, RLL Encoding.

MCA

Micro Channel Architecture

The type of BUS used in the IBM PS/2 line of computers. Each MCA hard disk controller board can support two hard disks. If two hard disk boards are installed in a PS/2 system, only one drive may be attached to each controller (you can never have more then a maximum of two hard drives in a PS/2 system, no matter how many controller cards you have). If you have one ESDI and one ST506/412 drive, the MCA architecture always selects the ESDI drive as drive C:.

MTBF

Mean Time Between Failures

This figure is supposed to indicate how long a drive is expected to last between needing repairs. The figure is normally rounded to the nearest thousand or sometimes even five thousand. To arrive at these figures, the manufacturers perform a battery of stress tests that are analyzed and worked into real-time working-condition hours. MTBF values are measured in power-on hours (conditions with the drive on).

MTTR

Mean Time To Repair

The average time to repair a failed drive. Being that replacement is usually recommended for most failed IDE drives, most manufacturers will list this spec as 30 minutes or less, which is actually *Mean Time To Replace.*

MEDIA

The magnetic layers of a disk or tape.

MEDIA DEFECT

A media defect can cause a considerable reduction of the read signal (missing pulse or Drop-Out) or create an extra pulse (Drop-In). The factories that manufacture the hard drives have equipment that can detect media defects which the standard controller might not. These areas will potentially give the user problems. Even if your own extensive diagnostics find less defects than the list supplied with the drive, enter in all defects when low level formatting to ensure error-free operation. Also see Drop-In/Drop-Out.

N

NRZ

Non-Return To Zero

A method of magnetic recording of digital data in which a flux reversal denotes a one bit and no flux reversal, a zero bit. NRZ recording requires an accompanying synchronization clock to define each cell time (unlike MFM or RLL recording). ESDI drives usually use this type of encoding.

O

OEM

Original Equipment Manufacturer

A manufacturer of equipment that is installed in the machine by the factory/vendor.

P

PARITY

A computer data-checking method using an extra bit in which the total number of binary 1s (or 0s) in a byte is always odd or always even; thus, in a parity scheme, every byte has eight bits of data and one parity bit. If using odd parity and the number of 1 bits comprising the byte of data is not odd, the 9th or parity bit is set to 1 to create the odd parity. In this way, a byte of data can be checked for accurate transmission by simply counting the bits for an odd parity indication. If the count is ever even, an error is indicated.

PARTITION

Partitioning divides a physical hard drive into logical volumes. Partitioning was originally intended as a way to divide precious storage space among the different operating systems that a computer had to use. Partitioning a hard drive is a mandatory operation, even if only one large partition is created. There are different partitioning limitations for each DOS version.

PARTITION TABLE

A drive structure that defines the boundaries of individual partitions and logical drives. The main Partition Table resides in the boot track (Track 0) and defines the four main partitions. The Extended Partition and each logical drive also has its own partition table.

Also see Partition, Extended Partition, Logical Drive, MBR.

PIO

Processor Input/Output

A means of data transfer that requires the use of the host processor, or CPU. PIO is very limited in data transfer rates because the CPU does not communicate with the peripheral (hard drive), meaning the peripheral can only send a minimum amount of data through the CPU during a cycle in order to guarantee transfer. The potential transfer capacity is wasted because the peripheral does not know how much of the CPU’s resources are free for data transfer.

Block PIO (BPIO) uses the advances made possible by local-bus technology to increase data transfer rates by simply multiplying the amount of information sent during a transfer cycle. BPIO counts the 512-byte transfer unit of normal PIO as a portion of a “block” which consists of “n” times 512-bytes.

Programmed PIO, sometimes called Throttled PIO or Flow Control, is an improved data transfer method in which the CPU sends an I/O Channel Ready (IOCHRDY) to the peripheral to let it know when more of its resources are available for data transfer.

Also see DMA.

PLATED THIN FILM DISKS

Magnetic disk memory media having its surface plated with a thin coating of a metallic alloy instead of being coated with oxide. Plated disks hold far more data than coated media. Also see Platter.

PLATTER

The circular aluminum or other surface where the actual data is stored. They are usually either coated with an oxide substance (like a floppy disk) or plated. Coating will hold far less magnetic particles than plating. Coated media can hold up to 20,000 magnetic domains in an inch of track while plated media can hold up to 50,000. Plated media are also extremely hard and much less susceptible to head crashes. Because of the precision and cost needed in making plated media, it is much more expensive and is generally used only in higher-capacity drives.

POST

Power-On Self-Test

A series of system tests and initializations performed by the BIOS on power-up. The POST is stored in the BIOS ROM.

R

RECALIBRATE

Return to Track Zero. A common disk drive function in which the heads are returned to track 0 (outermost track) to ensure good positioning by having a stable starting point.

RAID

Redundant Array of Inexpensive Disks

A relatively inexpensive way to increase capacity and reliability in mass storage. Basically consists of standard hard drives installed as modules in a central cabinet. In addition to multiplying the storage capacity of drive units, one of the most important features is the ability to guard data against media failure by simultaneously duplicating all information.

RWC

Reduced Write Current

A signal input (to some older drives) that decreases the amplitude of the write current at the actual drive head. Normally this signal is specified to be used during inner track write operations to lessen the effect of adjacent bit “crowding.” When installing a drive in a system, the number requested is the first track number to begin the area of reduced write current. That track and all subsequent tracks will be written with reduced write current.

RLL

Run Length Limited

A type of encoding scheme that reduces the amount of data-checking information that is stored, and requires less flux reversals (changes in the media) for a given amount of data (see MFM). The logic used is much more complicated than MFM, but this allows much more data to be placed on the disk. In RLL 2,7 the “run length” of zeros is limited to 7. The codes are chosen so that sequences of zeros in the codes always range from 2 to 7. This allows for a fifty percent increase in disk space over MFM encoding. RLL 3,9 (commonly called ARLL) is also available which further increases disk space (up to 100% from MFM), but 2,7 is more commonly used. With the exception of IDE and SCSI, the encoding/decoding process is done on the controller card, not on the hard drive.

A warning about RLL: Do not use MFM drives on an RLL controller! It can be done (and often is), but the life of the hard drive will be greatly reduced and may result in the loss of data.

ROTATIONAL SPEED

The speed at which the media spins. On a 5-1/4 or 3-1/2” Winchester-type drive it is usually 3,600 RPM and above.

ROOT DIRECTORY

The root directory is the master (highest level) directory of the drive from which all the sub-directories will branch out.

S

SA-400 Interface

Shugart Associates designed the SA-400 floppy disk drive in 1978, it was the first floppy drive to gain wide acceptance. This drive utilized a 34-pin cable that is still used in floppy drives today. The interface was later modified for hard drives and this modified version became the ST-506 interface. The SA-400 pin-outs have been slightly modified over the years, but the industry standard floppy interface is still referred to as the SA-400.

SCA

Single Connector Attachment

The method of attachment used in SCSI RAID storage systems to connect drive modules to the cabinet. Also see RAID.

SCSI

Small Computer Systems Interface

This interface originated as the SASI (Shugart Associates System Interface) in 1979. It was one of several disk interfaces that worked at a logical level instead of the widely accepted device level. Working at a logical level allowed for a stable interface while the disk devices could change rapidly.

In 1986 ANSI approved SCSI as ANSI X3.131-1986. Since then, the SCSI standard has been in a state of constant evolution, with SCSI-2 and SCSI-3 products currently being offered by manufacturers. The complete SCSI standards document can be obtained from ANSI.

The SCSI interface is not only designed for hard drives. It is a device interface that leaves most of the interfacing to the devices attached to it. A large number of devices, such as hard drives, tape drive units, floppy drives, and even printers, can be chained together off one interface. Each device must have its own Logical Unit Number (LUN) so that the interface can identify it. These are usually set by jumpers on the devices.

For hard drive installation, the BIOS Drive Type is usually set to “0” or “Not Installed” and the on-board BIOS on the SCSI interface handles any compatibility problems. When implemented in a DOS system, access to the host system is done through software interrupt 13h (INT 13h).

SCSI ID

See LUN.

SECTOR

A section of one track is called a sector. Each sector is defined with magnetic markings and an identification number from 0 to 65,535 (this identification number is contained in the sector header). All versions of DOS contain 512 bytes of data per sector (in the data section). All tracks have the same amount of sectors, even though the tracks are much larger near the outside of the platter than the inside. This is only done by DOS to avoid extra complications and as a result, gives up much valuable disk space. More advanced recording methods have been introduced such as ZBR (Zone Bit Recording) in which tracks on the outside cylinders have more sectors per track than the inside cylinders, but each sector still contains 512 bytes of data.

Also see Sector Header.

SECTOR HEADER

The address portion of a sector. The sector header (ID field) is written during the format operation. It includes the cylinder, head, and sector number of the current sector. This address information is compared by the disk controller with the desired head, cylinder, and sector number before a read or write operation is allowed.

SECTOR SPARING

This reduces the number of sectors on each track by one and places defect information on it. The application will see less defects, as only the drive is aware of the spare sectors. This reduces the total capacity of your drive, but is useful if the drive has a large amount of defects and your application requires a defect free drive.

SEEK

The radial movement of the heads to a specified track address.

SEEK TIME

Seek time usually refers to the average time it takes the heads to move between one track to another, on average. This is the seek time referred to in the hard drive listings in this book. There is also the track-to-track seek time which reflects how long it takes the heads to move between adjacent tracks.

SERVO TRACK

A pre-recorded reference track on the dedicated servo surface of certain disk drives. All data positions are compared to their corresponding servo track to determine “off-track/on-track” position, and correct cylinder positioning.

SKEWING

Skewing is the repositioning of the first logical sector on each track to mask the time required to switch heads or cylinders (tracks). A skew factor is added, when formatting a disk, to improve performance by reducing wasted disk revolutions.

Head skew reflects the time it takes for a drive to switch between head groups. It will be set to 0 if the drive does not support head skewing, as is typically the case.

SPINDLE

The rotating hub structure to which the disks are attached.

SPINDLE MOTOR

This is the motor that spins the platters. These motors are direct connect, they never use belts or gears. Most spindle motors spin at 3,600 rpms and faster, and are mounted outside of the enclosed platter area. Maxtor uses a unique design in some of their drives, placing it in between the actual platters thus allowing for more platters in a 5¼" casing.

SPINDLE MOTOR GROUND STRAP

Most older hard drives had this strap attached to the circuit board pressing against the spindle motor. This dissipated the static generated by operation and grounds the spindle motor casing. This strap sometimes caused high pitched or scraping noises. Placing a drop of oil between it and the spindle or adjusting slightly would usually stop the noise.

ST-506/412 INTERFACE

A very popular format in the 1980s, this interface was developed by Seagate Technologies (originally Shugart) in 1980, solely for use with their ST-506 hard drive (a 5 megabyte formatted drive). It was later revised in 1981 with a feature called "buffered seek" for their ST-412 (a 10 megabyte formatted drive). ST-506/412 drives are the direct ancestors to modern SCSI and IDE drives.

STEP

An increment or decrement of the head positioning arm to move the heads in or out respectively, one track from their current position. In buffered mode, the head motion is postponed until the last of a string of step pulses has been received.

STEPPER MOTOR ACTUATOR

The head actuator is responsible for moving the heads back and forth over the platters. A stepper motor actuator uses a motor that moves the heads by rotating the motor a "step" at a time. By rotating the motor a precise number of steps and then converting these steps into linear motion, the heads are moved. Alignment is provided by a metal band that can cause the drive to be put out of alignment if overheated or worn. Hard drives no longer use this type of actuator, although floppy drives still do. Also see Actuator, Voice-Coil Actuator.

STEP PULSE

The pulse sent from the controller to the stepper motor on the step interface signal line to initiate a step operation.

T

THIN FILM HEADS

A read/write head whose read/write element is deposited using integrated circuit techniques rather than being manually fabricated by grinding ferrite and hand winding coils. Also see Head.

TERMINATING RESISTORS

Hard drives are always shipped with the terminating resistor in place. This is a small resistor pack usually located near the bottom or near the drive select jumpers. They are usually yellow, and sometimes black or blue. They provide electrical signal termination so that the control signals do not echo along the drive cables. They also provide the proper electrical load for the controller and not using them properly could damage the controller.

TPI

Tracks Per Inch

A measurement of track density.

TRACK

The concentric circles that hold data on a disk platter. A track is composed of a circle of magnetic flux changes. Each track is divided into sectors, which are normally 512 bytes in length.

TRACK DENSITY

Track density is measured in Tracks per Inch and defines how many tracks can be written onto a disk surface. Track density is measured from the center to the outside (radius) of a track.

V

VIRTUAL MEMORY

Virtual memory is a scheme used by multi-tasking operating systems, such as Windows, to supplement limited system RAM with hard drive space. This is accomplished with “swap files,” which active applications write-to and read-from instead of system RAM. The disadvantage to virtual memory is that it can slow down system performance considerably, depending on the speed of the drive.

VIRTUAL SPLIT

A virtual split is a logical (not actual) split of the disk drive. Some controller cards support this option when low level formatting to make one physical drive appear as two drives to the operating system. This helps to overcome the partitioning limitations of MS/PC-DOS.

VOICE-COIL ACTUATOR

The head actuator is responsible for moving the heads back and forth over the platters. Voice-coil actuators use a solenoid (a magnet that pulls on a metal rod) to pull the heads toward the center of the platter. The heads are placed on a hinge mechanism with a spring that pulls in the other direction. When the solenoid is let go (magnetism released), the heads get pulled back to the outer edge of the platters. The term “voice-coil” is used due to the likeness of this system to speakers; speakers pull and push a cone.

Voice-coil actuators are much faster than stepper actuators because they don't have to “step” a little at a time, they just fly to their destination. The voice-coil knows what track it's on by reading information, called the “servo” data, that has been permanently placed between the tracks. Some of the larger capacity drives use one whole platter surface to store this information. That is why you will see a drive that has only seven data heads, but four platters (there are usually two heads for every platter). The eighth is reserved for the servo platter.

See Actuator, Stepper Motor Actuator.

W

WINCHESTER DRIVE

A term that originated from the old IBM drives in the 1960s that had 30 megabytes of removable media and 30 megabytes of fixed media. Thus the name 30-30, which is the caliber of the rifles made by the Winchester gun factory. The name now refers to non-removable media hard drives found in almost all PCs.

WRITE CURRENT

See Reduced Write Current.

WORM DRIVE

Write Once Read Many

WORM drives contain removable cartridges that can be written to once and read from an indefinite number of times, similar to a phonograph record.

WP

Write Precompensation

WP is the variance of the timing of the head current, from the outer tracks to the inner tracks of a disk, to compensate for the bit-shifting that occurs on the inner cylinders, which pack more data into a smaller area (see Sectors). Sometimes this number must be entered at the time of low-level formatting, and is only required on some oxide media drives.

X

XT INTERFACE

See IDE, ATA.

Z

ZBR

Zone Bit Recording

Trademark of Seagate Technology. A media optimization technique where the number of sectors per track is dependent upon the cylinder circumference; e.g., tracks on the outside cylinders have more sectors per track than the inside cylinders. The ZBR format is only done at the factory.

These drives have been factory low-level formatted and should not be low-level formatted by the end-user!

Index