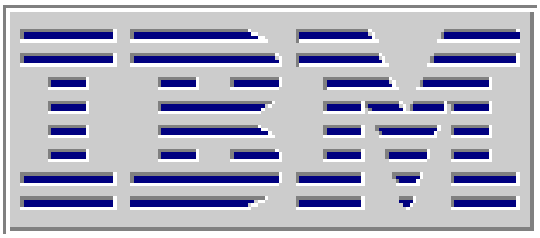
“El Torito”

Bootable CD-ROM Format Specification

Version 1.0

January 25, 1995



Curtis E. Stevens
Phoenix Technologies
2575 M^{c}Cabe Way
Irvine, Ca. 92714
Phone: (714) 440-8330
Fax: (714) 440-8300
curtis_stevens@bannet.ptltd.com

Stan Merkin (Formerly of IBM, Currently with DELL)
IBM
1000 NW 51st
BocaRaton, Fl. 33431
Phone: (407) 443-3264
Fax: (407) 982-8823

Contents

1.0 Overview

This specification defines how makers of CD-ROMs can package several "images" of floppy and hard disks on a single CD with the ability to catalog these images and to selectively boot from any single image.

The possibility of booting a PC from a CD ROM has raised several possibilities, including:

1. Self-configuring CD-ROMs that manage their own resources, including operating systems and drivers
2. The embedding of multiple operating systems and drivers on the same CD-ROM for a variety of applications, e.g., multi-language
3. The ability of the end user to select which operating system to boot
4. Copy protection for the CD ROM software and data

To accomplish this facility, there are currently two available technologies:

1. DOS-based drivers (e.g., SCSI or ATAPI)
2. The system BIOS.

Attempting to use DOS-based drivers (e.g., **SCSI** or **ATAPI)** to boot from a CD ROM creates a number of problems such as resource conflicts and inordinate use of memory.

The BIOS, however, avoids these problems and offers other advantages, including:

1. Can boot from a variety of operating systems by accessing a Boot Catalog on the CD-ROM.
2. Offers the choice of configuring the CD ROM as a hard disk (C: or D:) or floppy (A:).
3. Renames existing drives when necessary.
4. Uses existing BIOS technology (Logical Block Access) to access code and data.
5. Compatible with all DOS and Windows applications using INT 13 functions.

Using the BIOS to boot from the CD ROM requires using the available system header on the CD ROM

1.1 Scope

This document describes in detail how to format a CD-ROM from which you can boot a suitably-equipped computer system. It assumes you are familiar with standard BIOS INT 13 functions, ISO document number 9660, IBM/Microsoft INT 13 Extensions Dated 9/92, Version 1.0 of the Enhanced Disk Drive Specification (authored by Phoenix), and ATAPI..

1.2 Notation and Conventions

All numbers in this document are hexadecimal unless otherwise specified. All reserved or unused bytes are assumed to be zero. All character strings are padded to their full length with hex zeros. All word, double word and quad word values listed in this spec are in “Little Indian” (Intel) format.

1.3 Introduction

ATAPI-compliant IDE as well as SCSI CD-ROM drives can optionally provide new boot capabilities for personal computer manufacturers and users. In the past IBM and compatible personal computers would first attempt to boot from the floppy drive and if no floppy was present, from the hard disk.

This specification explains how the BIOS boot procedure can be enhanced to support a new medium, the CD-ROM. The CD-ROM is a removable media type, much like a floppy, but with the capacity of a hard disk.

Maintaining compatibility with current software requires using INT 13 calling conventions but providing an entirely new device interface. INT 13 places some restrictions on the information which a CD-ROM contains and also provides several new options for booting a system. The following is a list of new system capabilities and constraints:

1. The CD-ROM may boot as the A drive or the C drive.
2. If the CD-ROM boots as the A, drive it will contain a 1.2M, 1.4M or 2.88M floppy image.
3. If the CD-ROM boots as the A drive the systems normal A drive will become the B drive. If the system has a B drive it will become inaccessible.
4. If the CD-ROM boots as the C drive, it replaces the C drive.
5. No device drivers are necessary for applications which use the INT 13 interface. This means that normal MS-DOS compatible software can access the CD-ROM without a device driver. Some OS software, such as Windows, can be configured to use INT 13 giving the user immediate benefits from his CD.
6. Formatting a portion of the CD ROM as a "write protected" hard disk is an easy method of copy-protecting off-the-shelf software as well as allowing MS-DOS to function normally.
7. You can also copy-protect the non-disk-formatted portions of the CD ROM by embedding special CD-ROM drivers on the CD.

1.4 Implementation Options

Adding CD-ROM booting to a system will also require changes to the "SETUP" user interface. Many systems allow the following boot options.

```
A: then C:
C: then A:
C: only
```

CD-ROM booting now adds several new possibilities. The table below is a complete list:

```
A: then CD ROM then C:
CD ROM then A: then C:
CD ROM then C: then A:
C: then A: then CD ROM
CD ROM only
```

Many systems will not require all of these options, but it is necessary to give the user a CD-ROM boot disable capability. The ability to disable CD-ROM booting is required because there has been no standard for using the first few sectors of the CD. In the event that random data causes the system to find a valid "Validation Entry" on a non-compliant CD the user must have the recourse of disabling the boot capability.

If the end-user selects one of the Boot CD-ROM options on the Setup Menu, and if during POST the BIOS detects the presence of a CD-ROM drive, INT 19 at the end of POST will then attempt to load the operating system using the boot sequence specified in Setup.

To accomplish this, the installable Boot CD ROM feature makes available one of two INT 19 functions:

1. **Single-Image INT 19**. This feature, if instructed to boot from the CD ROM, does the following:
    a. Accesses the **Booting Catalog** in the CD-ROM header (See below).
    b. Verifies the existence of a boot image on the CD ROM.
    c. Reads the **Initial/Default Entry** (See below) and boots from the disk image specified in this entry.

    Single-Image INT 19 knows nothing about multiple-images, nor does it know about their possible entries listed in the Booting Catalog.
2. **Multiple-Image INT 19**. This feature, if instructed to boot from the CD ROM, does the following:
    a. Accesses the Booting Catalog in the CD-ROM header.
    b. Verifies the existence of a boot image on the CD ROM

c. Boots from either the image specified in the Initial/Default Entry or from one of the other images listed in **Section Headers** and **Section Entries** (See below) that follow the Initial/Default entry. The selection of which image to boot from depends on **Selection Criteria** determined by the OEM (See below).

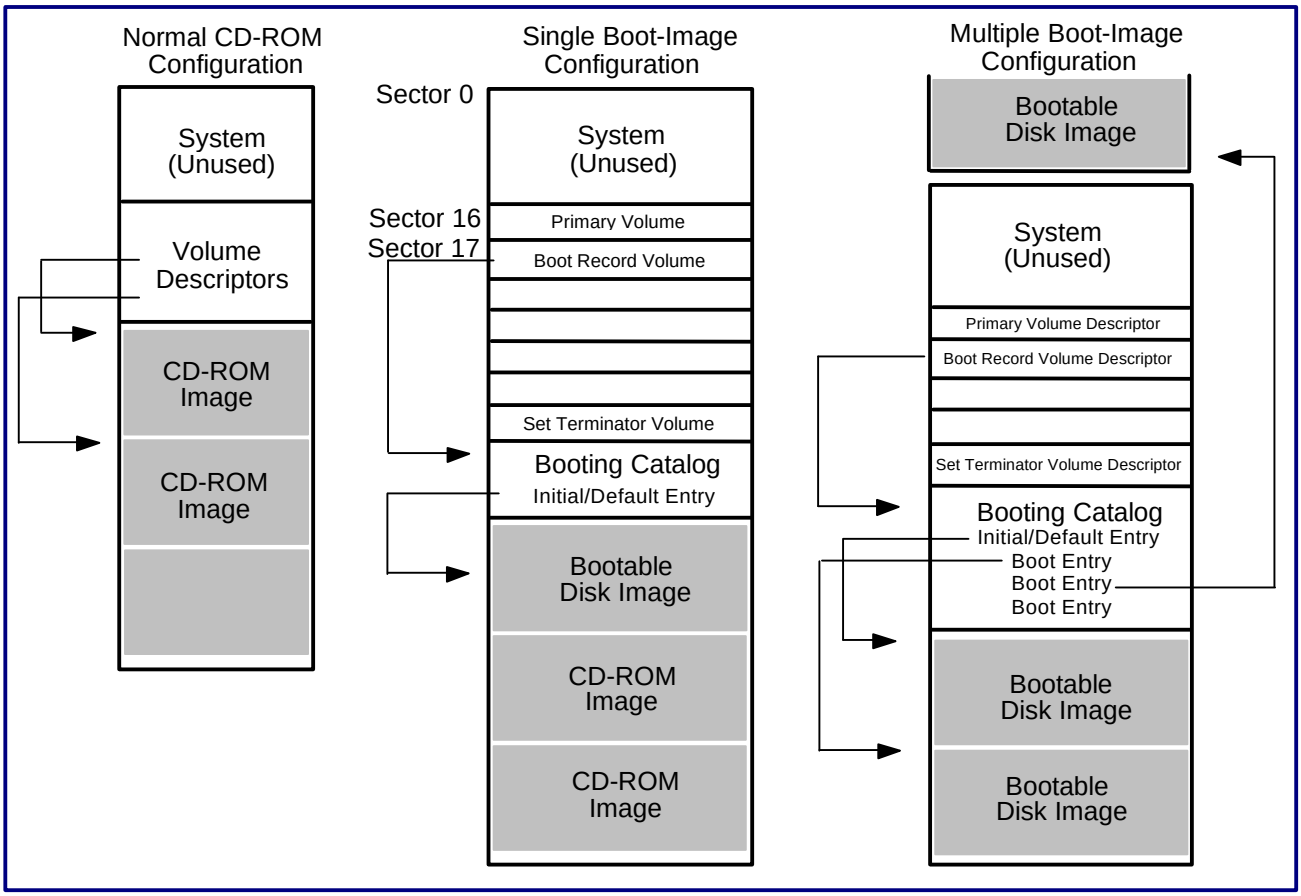


Figure 1. Three types of CD-ROM configuration:

1. The Normal CD-ROM configuration is not bootable, uses Root Directory and CD-ROM drivers to access CD-ROM images.
2. A BIOS with Single Boot-Image capability accesses the Initial/Default Entry to access a single bootable disk image. After loading the operating system, the system can revert to standard CD-ROM drivers and the Root Directory to access CD-ROM images.
3. A BIOS with Multiple Boot-Image capability can access any one of a number of Bootable Disk Images listed in the Booting Catalog. After loading the operating system, the system can access other items in the disk image with standard INT 13 calls or return to normal access of CD-ROM images using CD-ROM drivers and the Root Directory.

1.5 Definition of Terms

For the remainder of this document the following definitions apply:

Sector - Is a sector of data on a CD. This is in all cases 800 bytes.

Virtual Sector - Is a sector of data on an emulated device. This document only addresses 200 byte device (for emulation)

Virtual Disk - A series of sectors on the CD which INT 13 presents to the system as a drive with 200 byte **virtual sectors**. There are 4 virtual sectors found in each “sector” on a CD.

2.0 ISO-9660 and the Booting Catalog

One goal of this specification is to maintain ISO-9660 compatibility while providing system BIOSs with a simple way of getting to the location on the CD that contains the material to be booted. ISO-9660 defines that a "Primary Volume Descriptor" must reside at sector 10 (16 decimal), relative to the start of the session, followed by any number of other Volume Descriptors, followed by a "Volume Descriptor Set Terminator." The El Torito Bootable CD Specification builds on this format by requiring a "Boot Record" Volume Descriptor as defined in section 8.2 of ISO-9660. *See figure 7 for a description of the Boot Record.* This "Boot Record" must reside at sector 11 (17 decimal) in the last session on the CD. The Boot Record contains an absolute pointer to the Boot Catalog. The Boot Catalog is a collection of 20 byte entries (as described below), packed 40 entries to the sector. There is no limit to the number of sectors the Boot Catalog uses. This catalog allows the system to pick a proper boot image and then to boot from the selected image. The image may be virtualized into INT 13 drive number 00 or 80 for bootable disk emulation, or n+1 for a "non-bootable" emulation, where n is the number of the last hard drive initialized by the BIOS. The image may also simply be some code which will be loaded at boot time (no emulation). The BIOS will choose a drive number between 81 and FF when "no emulation" is specified. There are 5 types of entries diagrammed in figures 2-6. These entries define a validation procedure for the bootable CD, an Initial/Default entry, a section header, a section entry, and a Section Entry Extension.

2.1 Validation Entry

This must be the first entry in the boot catalog. The Validation entry validates that a booting catalog is present on the disk and identifies the manufacturer of the CD. If this entry is valid, it is assumed that the rest of the entries are valid. *See figure 2 for a description of the Validation Entry.*

2.2 Initial/Default Entry

The initial entry must contain a boot image which consists of generic programs/drivers that use only the BIOS provided INT 13 interface. The BIOS INT 13 interface consists of functions 0-19 and may optionally include functions 40-48. This entry will always be used by a BIOS that does not use any of the provided section headers. *See figure 3 for a description of the Initial/Default Entry.*

2.3 Section Header

If the CD was created for a machine with a BIOS incorporating this specification, the Section Header precedes a group of entries from which the BIOS may boot the computer. The section header has an identification string. If the BIOS understands the ID, string it may choose to boot the system using one of these entries in place of the INITIAL/DEFAULT entry. A good example would be a BIOS that supports multi-language capability. The BIOS manufacturer defines a header string and selection criteria. When CD's have a section that incorporates the header string and section entries, the BIOS automatically boots software in the native language of the BIOS. *See Figure 4 for a description of the Section Header.*

2.4 Section Entry

Section Entries must follow a Section Header. The Section Entry looks a lot like the Initial/Default Entry except the unused bytes contain selection criteria. The format of the selection criteria is a function of the BIOS vendor, the other fields are standardized for compatibility reasons. In the case of a foreign language BIOS three bytes would be used to identify the language. If the BIOS does not support languages, the Default entry determines the appropriate language and reinitiates the boot procedure using the proper boot image. If the 13 bytes provided for selection criteria are insufficient, a Section Entry Extension can be added by setting bit 5 of byte 1. See "Section Entry Extension" for details. *See Figure 5 for a description of the Section Entry.*

2.5 Section Entry Extension

Section Entry Extension must follow a Section Entry. This Extension defines additional selection criteria. When the 13 bytes provided by the Section entry are insufficient for representing the selection criteria a Section Entry Extension should immediately follow the Section Entry. Several Section Entry Extensions may be chained together by setting bit 5 of byte 1 for as many Extensions as necessary. The final extension should have bit 5 clear (set to 0) to indicate that no more Extensions follow. *See Figure 6 for a description of the Section Entry Extension*

Offset	Type	Description
0	Byte	Header ID, must be 01
1	Byte	Platform ID 0 = 80x86 1=Power PC 2=Mac
2-3	Word	Reserved, must be 0
4-1B	Character	ID string. This is intended to identify the manufacturer/developer of the CD-ROM.
1C-1D	Integer	Checksum Word. This sum of all the words in this record should be 0.
1E	Byte	Key byte, must be 55. This value is included in the checksum.
1F	Byte	Key byte, must be AA. This value is included in the checksum.

Figure 2 - Validation Entry

Offset	Type	Description
0	Byte	Boot Indicator. 88 = Bootable, 00 = Not Bootable
1	Byte	Boot media type. This specifies what media the boot image is intended to emulate in bits 0-3 as follows, bits 4-7 are reserved and must be 0. Bits 0-3 count as follows: 0 No Emulation 1 1.2 meg diskette 2 1.44 meg diskette 3 2.88 meg diskette 4 Hard Disk (drive 80) 5-F Reserved, invalid at this time
2-3	Word	Load Segment. This is the load segment for the initial boot image. If this value is 0 the system will use the traditional segment of 7C0. If this value is non-zero the system will use the specified segment. This applies to x86 architectures only. For "flat" model architectures (such as Motorola) this is the address divided by 10.
4	Byte	System Type. This must be a copy of byte 5 (System Type) from the Partition Table found in the boot image.
5	Byte	Unused, must be 0
6-7	Word	Sector Count. This is the number of virtual/emulated sectors the system will store at Load Segment during the initial boot procedure.
8-0B	D Word	Load RBA. This is the start address of the virtual disk. CD's use Relative/Logical block addressing.
0C-1F	Byte	Unused, must be 0.

Figure 3 - Initial/Default Entry

Offset	Type	Description
0	Byte	Header Indicator as follows: 90 -Header, more headers follow 91 - Final Header
1	Byte	Platform ID: 0 = 80x86 1=Power PC 2=Mac
2-3	Word	Number of section entries following this header
4-1F	Character	ID string. This identifies a section. This string will be checked by BIOS and BOOT software. If the string matches, the section should be scanned for boot images.

Figure 4 - Section Header Entry

Offset	Type	Description
0	Byte	Boot Indicator. 88 = Bootable, 00 = Not Bootable
1	Byte	Boot media type. This specifies what media the boot image emulates in bits 0-32. Bits 6 and 7 are specific to the type of system. Bits 0-3 count as follows 0 No Emulation 1 1.2 meg diskette 2 1.44 meg diskette 3 2.88 meg diskette 4 Hard Disk (drive 80) 5-F Reserved, invalid at this time bit 4 - Reserved, must be 0 bit 5 - Continuation Entry Follows bit 6 - Image contains an ATAPI driver bit 7 - Image contains SCSI drivers
2-3	Word	Load Segment. This is the load segment for the initial boot image. If this value is 0 the system will use the traditional segment of 7C0. If this value is non-zero the system will use the specified segment. This applies to x86 architectures only. For "flat" model architectures (such as Motorola) this is the address divided by 10.
4	Byte	System Type. This must be a copy of byte 5 (System Type) from the Partition Table found in the boot image.
5	Byte	Unused, must be 0
6-7	Word	Sector Count. This is the number of virtual/emulated sectors the system will store at Load Segment during the initial boot procedure.
8-0B	D Word	Load RBA. This is the start address of the virtual disk. CD's use Relative/Logical block addressing.
0C	Byte	Selection criteria type. This defines a vendor unique format for bytes 0D-1F. The following formats have currently been assigned: 0 - No selection criteria 1- Language and Version Information (IBM) 2-FF - Reserved
0D-1F	Byte	Vendor unique selection criteria.

Figure 5 - Section Entry

Offset	Type	Description
0	Byte	Extension Indicator. Must be 44
1	Byte	Bits 1-4 - Unused 5 - 1 = Extension Record follows, 0 = This is final Extension 6-7 - Unused

2-1F	Byte	Vendor unique selection criteria

Figure 6 - Section Entry Extension

Offset	Type	Description
0	Byte	Boot Record Indicator, must be 0
1-5	Byte	ISO-9660 Identifier, must be “CD001”
6	Byte	Version of this descriptor, must be 1
7-26	Byte	Boot System Identifier, must be “EL TORITO SPECIFICATION” padded with 0’s.
27-46	Byte	Unused, must be 0.
47-4A	Dword	Absolute pointer to first sector of Boot Catalog.
4A-7FF	Byte	Unused, must be 0.

Figure 7 - Boot Record Volume Descriptor

3.0 The INT 13 Accessible Image

The boot image is an exact replica of the floppy or hard disk the system is intended to emulate. You can create this image by reading from the source media as if it were a Logical-Block Addressable (LBA) device and incrementing the logical-block value until all sectors have been read. Concatenate all blocks in a single file. X3T9.2 (ATA 4.0) defines the LBA translation as follows:

LBA 0 = Cylinder 0, Head 0, Sector 1

LBA X = ((Cylinder * Maximum Heads + Head) * Sectors per Track) + Sector -1

Place the resulting file on the CD in ISO-9660 format, and make a catalog entry in sector 0 of the CD-ROM. It is the responsibility of the INT 13 interface to recreate the disk geometry and to properly access the information.

4.0 INT 13 and CD-ROMs

Once the system has selected the proper boot image, the INT 13 interface must recreate the Floppy/Hard disk. It is the responsibility of the BIOS to create a valid geometry for the virtual disk and then present this geometry in function 08. The INT 41 pointer is not valid at this time which means that software that uses INT 41 is not valid on a CD-ROM. When the boot code is loaded, INT 13 must reverse the CHS addresses it receives creating a valid LBA address which is then offset by the Load LBA. This procedure can emulate all standard Floppy/Hard disk transactions. The remainder of this section assumes that the system is using the Initial/Default Entry, or has chosen a Section Entry.

4.1 INT 13 Function 08

This function normally gets the geometry information directly from the selected catalog entry. In the case of a floppy simulation the geometry is well known and defined as follows:

Size	Tracks x Heads x Sectors
1.44 Meg	50 x 2 x 12
2.88 Meg	50 x 2 x 24
1.2 Meg	50 x 2 x 0F

When the simulated device is a Hard Disk, the BIOS should use a geometry which fits the partition table located in the Load LBA sector. Hard Disk images may only have 1 partition in the partition table and it must be the first entry.

4.2 INT 13 Function 48

The IBM/Microsoft INT 13 extensions use function 48 for returning drive capability information. The Phoenix Enhanced Disk Drive Specification adds to this information by providing a pointer to a table which contains extended drive information. Byte 10, bit 6 of this table is set to 1 if the attached device is a CD-ROM. This function gives software residing on a CD the ability to determine that it is running from a CD-ROM.

4.3 INT 13 and Booting

The selected image is only a boot image if the Boot Indicator in byte 0 of the section entry is an 88. Any other value flags the image as not bootable, requiring the system to check the next boot device, in most cases the floppy drive. When the boot image is in a floppy format, the CD becomes drive 00, the former drive 00 moves to 01 and all other drive identifiers remain intact. This means that the system always has a usable floppy drives as well as maintaining all hard disk drive letters so that a CD can boot and then install software on the user's hard disk. The floppy remains accessible as drive 01 to allow the software vendor to update the CD with a supplemental floppy if necessary.

When the boot image is a hard disk, all drives numbered 80 and above are incremented by 1, becoming 81 and up, and, the CD will become drive 80. This allows software vendors to create stand-alone CD's that normally run under MS-DOS and use the standard INT 13 interface to place their software on the CD without regard to the media. The system's hard drive remains accessible because the stand alone CD may need some temporary disk storage.

When the boot image is simply a "loader" or stand alone program, and NO EMULATION is desired, the drive numbers will be unaffected after the boot image is loaded. The specified number of sectors are loaded at the specified segment (usually 7C0), and then the BIOS jumps to the load address. The software can retrieve a pointer to its boot information by issuing INT 13, Function 4B, AL=1.

If the Boot Indicator in Byte 0 is not an 88, the system accesses the next boot device, and the CD will now become the last drive in the user's drive sequence regardless of the image type. This allows software vendors to distribute data CD's which do not require device drivers for MS-DOS style accessing.

4.4 Boot Entry Selection

If the CD has several boot entries, a default entry which boots a selection program may be provided as the Default/Initial catalog entry. This image will usually be a floppy image which loads a program that selects the proper boot image by examining the system configuration or questioning the user.

5.0 CD Boot Procedures

This purpose of this section is to describe the intended sequence of events for booting from the various CD formats described above. These procedures should guarantee that software can boot and operate as the software vendor intends. The BIOS provides three INT 13 functions for assigning and deassigning the CD. These functions are documented in the next section and referenced in the paragraphs below.

5.1 Floppy Booting

The total capacity of a CD is 600+ meg (decimal), give or take a little compression. The purpose of using a floppy emulation is to get the system booted under a specific operating system, load a device driver which understands the true format of the CD, usually ISO-9660, and then put the system back in its normal operating configuration. This sequence of events gives the software on the CD the "normal" configuration of the system and makes the CD available as the same drive letter the user would normally have assigned. Only a small portion of the CD is wasted on the boot image. Another advantage of the floppy image is that you can place several of them on a single CD, each with a customized system configuration. Several entries can be made in the catalog, each with different booting criteria.

5.2 Hard Disk Booting

The purpose of booting as a hard disk is to provide a large MS-DOS compatible CD-ROM storage space on which programs may reside and/or access data. This lends itself well to stand alone/protected applications which do not use full system resources. Booting the CD as a hard disk normally requires that all applications on the CD are INT 13/MS-DOS based. The system is not returned to its "normal" operating configuration and software is not intended for installation. This is for applications that run directly from the CD. This does not restrict an application from installing a device driver and using it. In fact, this is encouraged for those applications which require encryption or copy protection. What must be kept in mind is that the CD is a "slow" device, the INT 13 interface can access the data at about the same speed as any driver in this environment.

5.3 No Emulation Booting

When the Media Type is set to zero the BIOS does not use the CD to emulate a disk. The boot operation loads the requested number of sectors directly to the specified segment. When loading is complete the BIOS will jump to segment:0. The associated piece of software can be a "loader" (which provides its own CD interface), or it can be a stand alone program. The El Torito specification allows for the loading of FFFF sectors (This would allow the BIOS to fill the entire low 640k memory area with data). Once the system jumps to segment:0, the program can retrieve its boot information by issuing INT 13, Function 4B, AL=01. After the boot process has been initiated the INT 13 Extensions (functions 41-48) will access the CD using 800 byte sectors and the LBA address provided to INT 13 is an absolute sector number. This gives any program running in no emulation mode the ability to locate the boot catalog, and any other information on the CD, without providing a device driver.

5.4 System Optimization

The Default/Initial Catalog Entry may contain a floppy image which can examine the system and/or query the user about his system configuration, select a proper Section Entry, and then reboot the system using this Section Entry. This program on its first invocation may write configuration information to the users Hard Disk and then not question the user on subsequent booting operations. This is particularly useful when a CD has multiple uses or the user purchases several CD's from the same software vendor. The selection program will already have configuration information available.

6.0 New INT 13 Functions

Transparent system operation is not always desirable because software installation usually requires the system to be in its “normal” configuration. For this reason 3 new INT 13 functions are provided. These functions allow the system to return to its normal state or to enter the “boot” state a second time after the boot process has already been initiated.

6.1 INT 13 Function 4A - Initiate Disk Emulation

This function takes a pointer to the Specification Packet and uses it to make the specified drive letter available, using the Image LBA “pointer” in the supplied packet.

Registers at call:

AH = 4A
AL = 00
DS:SI - Specification Packet as defined in figure 9

Registers on return

AX - Return Codes
CF - Clear if selected drive is emulating
CF - Set if system not in emulation mode

6.2 INT 13 Function 4B - Terminate Disk Emulation

This function returns the system to a configuration that matches a normal floppy or hard disk boot. If the CD booted as a floppy, the “A” drive returns and the CD becomes driver addressed. If the CD booted as a hard disk, the drive numbers are all decremented by 1, this returns all hard disks to their standard/normal device numbers.

Registers at call:

AH = 4B
AL = 00, Return Status and Terminate Emulation
AL = 01, Return Status only, Do Not Terminate Emulation
DL = Drive number to terminate, 7F = Terminate all
DS:SI - Empty Specification Packet, *See figure 9*

Registers on return:

DS:SI - Completed Specification Packet, *See Figure 9*
AX - Return Codes
CF - Clear if system released
CF - Set if system not in emulation mode

6.3 INT 13 Function 4C - Initiate Disk Emulation & Boot

This function takes a pointer to the Specification Packet and uses it to reboot the system from scratch. This function only returns if the supplied packet does not contain a pointer to a bootable disk image.

Registers at call:

AH = 4C
AL = 00
DS:SI - Specification Packet as defined in figure 9

Registers on return

AX - Return Codes
CF - Set if system failed to boot
CF Clear will not occur because when this function succeeds no return is generated.

6.4 INT 13 Function 4D - Return Boot Catalog

This function takes a drive number in DL, a command packet in DS:SI, and returns the requested number of sectors from the boot catalog.

Registers at call:

AH = 4D
AL = 00
DS:SI - Pointer to command packet as defined in figure 8

Registers on return:

AX - Return Codes
CF - Clear if successful
CF - Set if device is not an emulated CD, or command packet is out of bounds

Offset	Type	Description
0	Byte	Packet Size in bytes, currently 8
1	Byte	Sector count. Number of sectors in the boot catalog to transfer
2-5	Dword	Pointer to the buffer where the boot catalog will be moved
6-7	Word	Beginning sector to transfer. This number is relative to the start of the boot catalog. The initial call to INT 13 Function 4D should always set this value to 0.

Figure 8 - Command Packet

Offset	Type	Description
0	Byte	Packet Size, currently 13
1	Byte	Boot media type. This specifies what media the boot image is intended to emulate in bits 0-3. Bits 6 and 7 are specific to the type of system. Bits 0-3 count as follows: 0 - No Emulation 1 - 1.2 meg diskette 2 -1.44 meg diskette 3 -2.88 meg diskette 4 -Hard Disk (drive C) 5-F-Reserved, invalid at this time bits 4-5 - Reserved, must be 0 bit 6 - Image contains an ATAPI driver, bytes 8 & 9 refer to IDE interface bit 7 - Image contains SCSI drivers, bytes 8 & 9 refer to SCSI interface
2	Byte	Drive Number. This is the drive number on which emulation is being initiated or terminated. This must be 0 for a floppy image, 80 for a bootable hard disk, and 81-FF for a “non-bootable” or “no emulation” drive.
3	Byte	Controller Index. This specifies the controller number of the specified CD drive.
4-7	Dword	Logical Block Address for the disk image to be emulated.
8-9	Word	Device Specification. For SCSI controllers byte 8 is the LUN and PUN of the CD Drive, byte 9 is the Bus number. For IDE controllers the low order bit of byte 8 specifies master/slave device, 0 = master.
A-B	Word	User Buffer Segment. If this field is non-zero it specifies the segment of a user supplied 3k buffer for caching CD reads. This buffer will begin at Segment:0.
C-D	Word	Load Segment. This is the load address for the initial boot image. If this value is 0 the system will use the traditional segment of 7C0. If this value is non-zero the system will use the specified address. *This field is only valid for function 4C*
E-F	Word	Sector Count. This is the number of virtual sectors the system will store at Load Segment during the initial boot procedure. *This field is only valid for function 4C*
10	Byte	Bits 0-7 of the cylinder count. This should match the value returned in CH when INT 13 function 08 is invoked.
11	Byte	This is the value returned in the CL register when INT 13 function 08 is invoked. Bits 0-5 are the sector count. Bits 6 and 7 are the high order 2 bits of the cylinder count.
12	Byte	This is the head count, it should match the value in DH when INT 13 Function 08 is invoked.

Figure 9. - Specification Packet

Last Page

(This page intentionally left blank)

