Disc Manufacturing, Inc.
A QUIXOTE COMPANY

Introduction to ISO 9660,

what it is, how it is implemented, and how it has been extended.

Clayton Summers

WHO IS DMI?

Disc Manufacturing, Inc. (DMI) manufactures all compact disc formats (i.e., CD-Audio, CD-ROM, CD-ROM XA, CDI, PHOTO CD, 3DO, KARAOKE, etc.) at two plant sites in the U.S.; Huntsville, AL, and Anaheim, CA. To help you, DMI has one of the largest Product Engineering/Technical Support staff and sales force dedicated solely to CD-ROM in the industry.

The company has had a long term commitment to optical disc technology and has performed developmental work and manufactured (laser) optical discs of various types since 1981. In 1983, DMI manufactured the first compact disc in the United States. DMI has developed extensive mastering expertise during this time and is frequently called upon by other companies to provide special mastering services for products in development.

In August 1991, DMI purchased the U.S. CD-ROM business from the Philips and Du Pont Optical Company (PDO). PDO employees in sales, marketing and technical services were retained.

DMI is a wholly-owned subsidiary of Quixote Corporation, a publicly owned corporation whose stock is traded on the NASDAQ exchange as QUIX. Quixote is a diversified technology company composed of Energy Absorption Systems, Inc. (manufactures highway crash cushions), Stenograph Corporation (manufactures shorthand machines and computer systems for court reporting) and Disc Manufacturing, Inc.

We would be pleased to help you with your CD project or answer any questions you may have. Please give us a call at 1-800-433-DISC for pricing or further information.

We have four additional technical papers available entitled

Integrating Mixed-Mode CD-ROM

An Overview to MultiMedia CD-ROM Production

Compact Disc Terminology - 2nd Edition

A Glossary of CD and CD-ROM Terms

These are available upon request
800-433-DISC
302-479-2500
Fax: 302-479-2527

This paper was written in response to the many questions we, as a CD-ROM manufacturer, have received concerning ISO 9660. Our intent was to provide clarity and simplification to a very technical subject. We hope you find it helpful.

For reprinting privileges, call or write to:
Nancy Klocko
Disc Manufacturing, Inc.
1409 Foulk Rd., Suite 102
Wilimington, DE 19803
800-433-DISC, 302-479-2527

Acknowledgments

Without the help of many people, this paper would not have been possible.

Special Thanks go to:

Wendy Upham and Breck Rowell for working on how MS-DOS and the Macintosh implement ISO 9660 and what some of the quirks are

John Sands at Young Minds

Apple Computer

Doug Carson and Associates

and Nancy Klocko and Pam Sansbury for invaluable help editing and revising and Lori Magno for formatting.

Table of Contents

Tables

Figures

Introduction to ISO 9660

The Digital Audio Compact Disc has been called the most successful consumer product ever launched. Since it's introduction in June of 1980, the CD has come to dominate the music industry and become the format of choice for millions of music listeners due to the ultra high fidelity afforded by the digital recording technique and the near indestructibility afforded by the optical design. These same features make the CD very attractive as a carrier of other types of digital information. Another feature that makes the Compact Disc attractive as a medium for digital information is that CDs can be manufactured in large quantities quickly and inexpensively. Also, due to the industry standards defined by the Red Book, Yellow Book, and ISO 9660, any CD can be used on almost any hardware/software platform. It, therefore, comes as no surprise that this 15 grams of poly-carbonate and aluminum which contains billions of bits of data would be embraced by the computer industry to store and distribute huge amounts of data. However, creating a disc that works on multiple platforms is not as simple as copying files to a floppy.

Background

Before ISO 9660, all CD-ROM discs could be read by all CD-ROM drives; however, CD-ROM drives were not supported by any readily available computer operating system. Application developers were required to have software device drivers for each computer and CD-ROM drive combination that they wanted to support. In addition, most applications require a file structure and this had to be provided by each developer as well.

Typically, application developers used a systems house to provide device drivers and file system software as well as build and retrieval engines. The result was that application developers requested both enhancements to the build and retrieval engines and support for additional drive types. The systems houses had to spend critical resources to develop these drivers when they would have preferred to spend those resources in other areas. A committee called High Sierra was formed to develop an industry standard to address the file system software.

The High Sierra proposal was designed to enable data interchange between computers using standardized software. When a computer is equipped for either High Sierra or ISO 9660, data on any properly encoded CD-ROM may be read using standard operating system instructions such as list directory, open, read and close. This reduces the amount of effort required to bring an application to market. In addition, discs may be read by any drive that has standard driver software.

High Sierra was defined and submitted to the **I**nternational **S**tandards **O**rganization in May of 1986. During the time it was being debated and approved, the companies involved in creating the High Sierra proposal went ahead and implemented High Sierra. ISO 9660 was published in April of 1988. ISO made several minor changes during the process that made the ISO 9660 standard incompatible with the High Sierra proposal. The changes involved rearranging the order of the information in the directory record and changing the code that identifies the disc as a High Sierra or ISO 9660 disc, among other, more esoteric, items.

To accurately and repeatedly create ISO-9660 discs requires an understanding of what ISO-9660 is and how it is implemented by different platforms. First a little background information will be presented that helps put the concept of a common format for data interchange into perspective. This will be accomplished by discussing file systems and how they relate to the computer's operating system. Then ISO-9660 will be covered in general terms and some of the more commonly used features and data structures will be covered in some detail. A description, at a conceptual level, of how some operating systems implement support for ISO-9660 and notes regarding some of the peculiarities this causes will then be presented. Then some of the ways ISO-9660 has been extended to provide better support for two particular operating systems, the Macintosh and UNIX environments, will be introduced. Finally, the Frankfurt proposal, an extension to ISO 9660 that allows updating information on a recordable CD, will be discussed.

File Systems

Most operating systems store information in both fast, short term memory usually referred to as Random Access Memory or RAM, as well as in relatively slow, long term memory. Typically, the slow, long term memory takes the form of a floppy disk, or hard disk and is stored as files. If we compare this to someone's office, the fast, short term memory can be compared to the desktop, where things are actually being worked on. The slow, long term disk can be compared to the file cabinet, where unused items are put until needed. The way the operating system keeps track of where files are located is called the file system. Examples of file systems are MS-DOS's FAT (File Allocation Table), the Macintosh HFS (Hierarchical File System), OS/2's HPFS (High Performance File System), and the UNIX File System. All of these file systems are specific to, and optimized for, a particular operating system. ISO-9660 is also a file system. However, it was designed to be independent of any operating system, and because it was designed for CD-ROM, is also "read only". This means that unlike the other file systems mentioned, it does not provide any way to add to or change the information in it. Since ISO 9660 was intended to be used on a diverse group of operating systems, it includes only the minimum information that can be utilized by the widest variety of systems.

Overview of ISO-9660 structure.

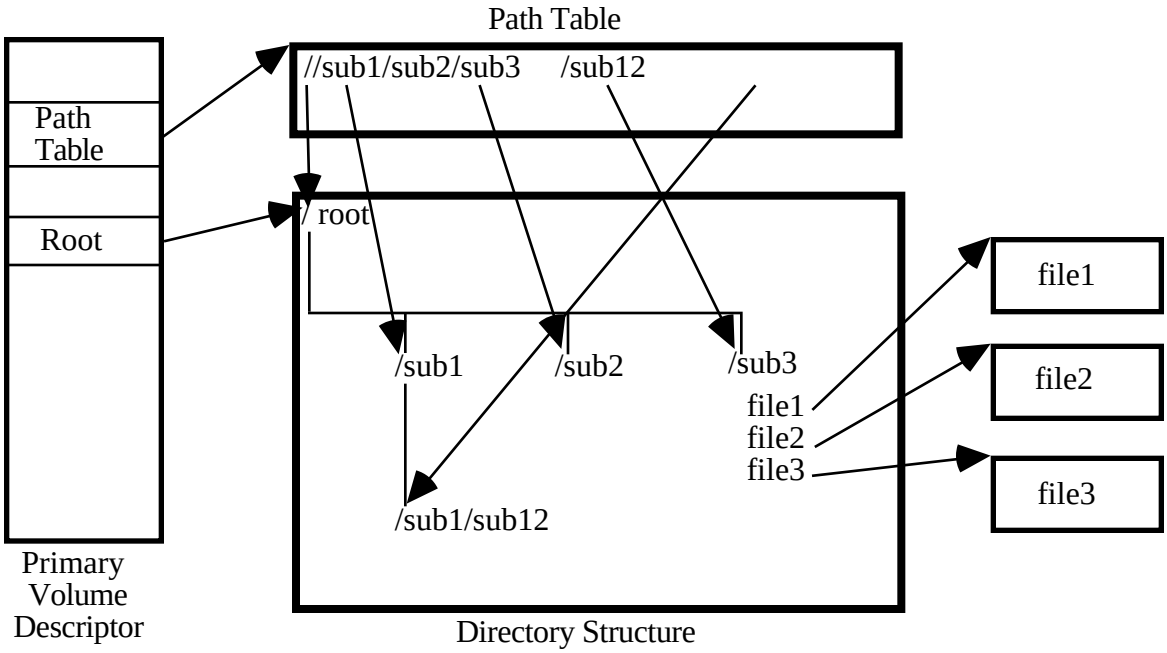

Figure 1. ISO 9660 structures

ISO 9660 data structures fall into three main categories: the Volume Descriptors, the Directory Structures, and the Path Tables. These structures are interrelated as shown in figure 1. The Volume Descriptor tells where the directory structure and the Path Table are located, the directories tell us where the actual files are located, and the Path table gives us short cuts to each directory .

The Volume Descriptors

There are currently four types of Volume Descriptors defined in ISO 9660. Only one of these, the Primary Volume Descriptor, is commonly used. The other types are the Boot Record, the Supplementary Volume Descriptor, and the Volume Partition Descriptor. The Boot Record can be used for systems that must perform some type of initialization before the user can access the volume, although ISO 9660 does not specify what information must be in the Boot Record or how it is to be used. The Supplementary Volume Descriptor can be used to identify an alternate character set for use by systems that do not support the ISO 646 character set. The Volume Partition Descriptor can be used to logically divide the volume into smaller volume partitions, although ISO 9660 does not

specify how to do this, only that it can be done.[1] The Volume Descriptors are recorded starting at Logical Sector 16 (which corresponds to two seconds and sixteen sectors into the CD, or in CD "Atime", 00:02:16).

The Primary Volume Descriptor

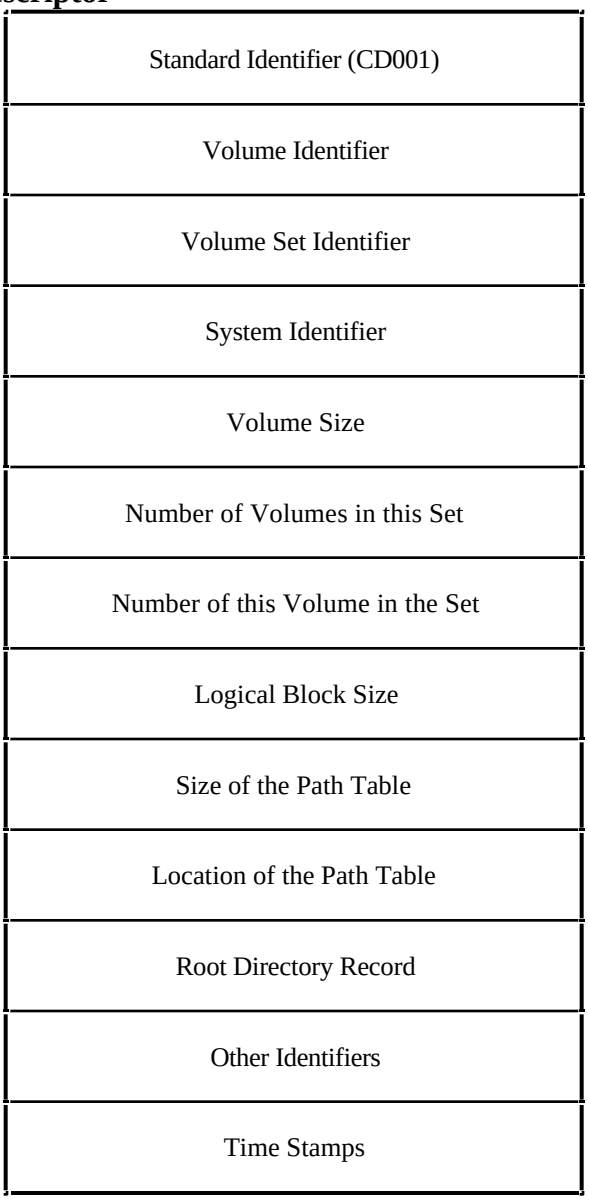

Figure 2. The Primary Volume Descriptor

The Primary Volume Descriptor as seen in figure 2 is the starting point in identifying a CD-ROM. It contains the Standard Identifier, the Volume Identifier, the Volume Set Identifier, the System

[1] ISO 9660-1988, pp. 11, section 8.1.1

Identifier ,the size of the Volume, the number of Volumes in the Volume Set it belongs to, the sequence within the Volume Set that this Volume belongs, the Logical Block size of the blocks in this volume, the size of the Path Table, the location of the Path Table, the Directory record for the Root Directory, other identifiers and important times relating to the Volume.[2]

The Standard Identifier is a set of characters, defined by ISO 9660 to be `CD001`, that tells the Operating System that this is an ISO 9660 disc. This is to distinguish the volume from other file systems that use a similar layout, such as High Sierra, whose Standard Identifier is CDROM, and Compact Disc Interactive, whose Standard Identifier is CD-I.

The Volume Identifier is simply the name that is given to the ISO 9660 volume.
The characters that can be used in the Volume Identifier are restricted to what ISO 9660 calls d-characters and the length is restricted to 31 characters. The d-characters are shown in figure 3.

```
A B C D E F G H I J K L M N O P Q R S
T U V W X Y Z 0 1 2 3 4 5 6 7 8 9 _
```

Figure 3. The d-characters

Some systems, such as the Macintosh, use the Volume Identifier extensively. Others, such as MS-DOS, use it somewhat, and some, such as UNIX, barely use it at all.

The Volume Set Identifier is the name given to the Multiple Volume Set that this Volume belongs to. Like the Volume Identifier, it is restricted to the d-characters and cannot be more than 31 characters long. For example, if this Volume where named DICTIONARY_E_H, it might have a Volume Set Identifier of DICTIONARY, meaning that this Volume contains the words starting with the letter E through the letter H, and the Volume Set is the set of discs for the entire alphabet.

---

[2] see Appendix A: Table 14 and ISO 9660-1988, pp. 12-16, section 8.4

The System Identifier identifies a system that can recognize and act on logical sectors 0 through 15.[3] While ISO 9660 specifies that this is what the System Identifier is used for, it does not specify what is in sectors 0 through 15, nor does it specify how the data is used. The characters that can be used in the System ID are what ISO 9660 calls a-characters and the length is restricted to 31 characters. The a-characters are shown in figure 4.

```
A B C D E F G H I J K L M N O P Q R S
T U V W X Y Z 0 1 2 3 4 5 6 7 8 9 _ sp
! " % & ' ( ) * + , - . / : ; < = > ?
```

Figure 4. The a-characters

The Volume Size is a number that tells the operating system how many Logical Blocks are in this Volume. A Logical Block is the basic way of locating things in the Volume. All locations are given as Logical Block Numbers. If the Volume is pictured as an Interstate highway, then the Logical Block Numbers are the mile markers.

Volume Set Size is a number that tells the operating system how many volumes are in the Volume Set to which this Volume belongs. The Volume Sequence Number is the place within a multiple volume set that this volume belongs. For example on a disc with Volume Set Size of 5 and Volume Sequence Number of 3, this disc is the third disc of a five disc set.

The Logical Block Size is the number of bytes that make up the smallest amount of space that is allocated in this volume. This number can be 512, 1024, or 2048 bytes. Most ISO 9660 discs use a Logical Block Size of 2048, the same as the Sector Size.

The Path Table Size tells the operating system how many bytes are in the Path Table. Most operating systems that use the Path Table keep it in fast, local memory (RAM), and this number is a quick way

[3]ISO 9660-1988, pp.13, section 8.4.5

for the operating system to know how much memory it needs to allocate before it reads the Path Table. This way the Operating system only reads the Path Table once, saving time. The location of the Path Table must be in the Primary Volume Descriptor since the Path Table itself may be anywhere in the Volume.

The Root Directory record contains the information the operating system needs to locate and read the top level directory. It is formatted exactly the same as any other directory record.[4]

Other identifiers in the Primary Volume Descriptor contain information about who published this Volume, who prepared the data, what the application is, and what the names of the files are that contain the copyright notice, the abstract, and the bibliography.

The time stamps are fields in the Primary Volume Descriptor that contain information about when the Volume was created, when it may have been modified, when the data becomes effective, and when the data becomes obsolete.

[4]See Appendix A, Table 15, page IV

The Directory Structure

The ISO 9660 directory structure is organized in a hierarchical manner similar to most modern file systems.[5] At the top of the hierarchy is the Root Directory, the location of which is identified in the Primary Volume Descriptor. When drawn hierarchically, the directory structure resemble the roots of a tree, with the Root directory at the top of the structure, as shown in figure 5.

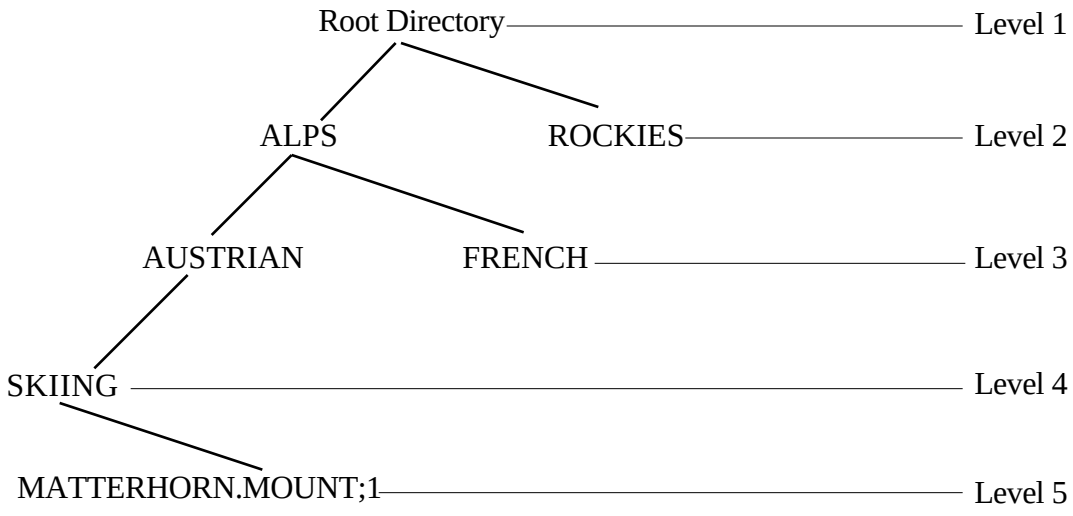

Figure 5. The Directory Hierarchy

[5]ISO 9660:1988, pp.7, section 6.8.2

As shown in figure 5, there are distinct levels in this hierarchy. The Root Directory is the only directory at level 1. In the example illustrated by figure 5, Subdirectories ALPS and ROCKIES are at level 2, Subdirectories AUSTRIAN and FRENCH are at level 3, Subdirectory SKIING is at level 4, and the file MATTERHORN.MOUNT;1 is at level 5. To insure compatibility, ISO 9660 imposes a limit of eight levels to the depth of the directory structure.[6] It also imposes a limit on the length of the path to each file. The length of the path is the sum of the lengths of all relevant directories ,the length of the File Identifier, and the number of relevant directories. The length of the path cannot exceed 255. Again, in figure 5, the sum of the length of the File Identifier ,the lengths of the relevant directories , and the number of relevant directories is 39 as shown in table 1.

Table 1. Length of the Path

Identifier	Length
ALPS	4
AUSTRIAN	8
SKIING	6
MATTERHORN.MOUNT;1	18
number of directories	3
sum of lengths and number of directories	39

A directory in an ISO-9660 volume is recorded as a file containing a set of directory records. Each directory record describes a file or another directory. Every directory has a parent directory. The parent directory contains the directory record that identifies that directory, as shown in figure 6.

[6]ISO 9660:1988, pp.7, section 6.8.2.1

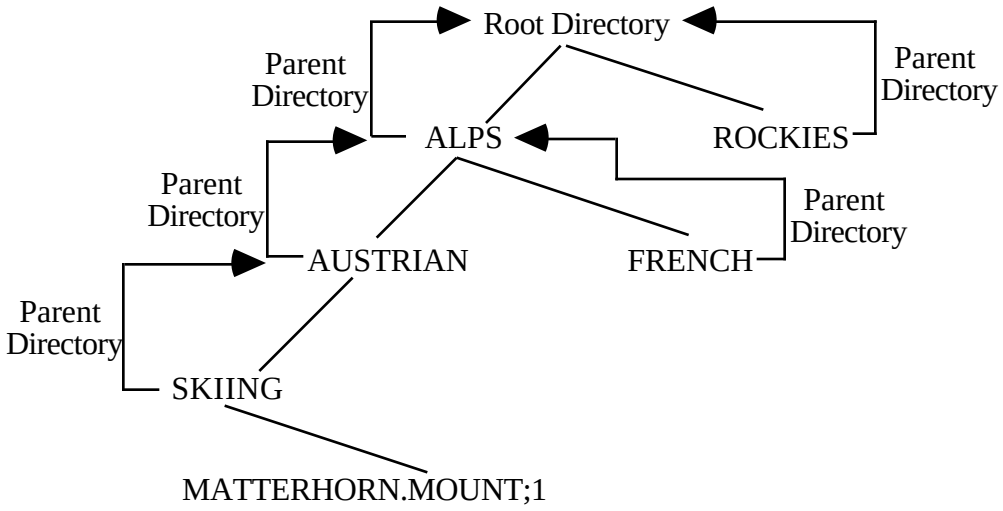

Figure 6. Parent Directories

The Root directory's parent is the Root directory itself.

Each directory also contains a record for its parent directory. A given directory may contain entries for several files as well as for several directories, all of which have the same parent.

File Names

Every file and directory in an ISO 9660 Volume has a name an identifying name associated with it. This name is called the File Identifier. An ISO 9660 File Identifier actually consists of five parts as shown in table 2.[7]

Table 2. The File Identifier

	1) File Name	2) SEPARATOR 1	3) File Name Extension	4) SEPARATOR 2	5) File Version Number
contents	d-characters (see figure 3)	.	d-characters (see figure 3)	;	a number from 1 to 32767
file 1	MATTERHORN	.	MOUNT	;	1
file 2	PIKES_PEAK	.		;	1
file 3		.	HILLS	;	1
directory	SKIING				

The File Identifier must also meet the following conditions:[8]

- If the File Name has no characters, then the File Name Extension must have at least one character, as shown in table 2, file 3.

- If the File Name Extension has no characters, then the File Name must have at least one character, as shown in table 2, file 2.

- The sum of the lengths of the File Name and the File Name Extension cannot exceed 30.

[7]ISO 9660:1988, pp. 10, section 7.5.1
[8]ISO 9660:1988, pp. 10, section 7.5.1

An ISO 9660 directory name is limited to only a Name; it cannot have a SEPARATOR 1 (.), an Extension, a SEPARATOR 2 (;) or a version number, as shown in table 2, directory.[9]

Order of Directory Records

ISO 9660 also specifies the order of the records in a directory.[10] They must be sorted by the relative value of the File Identifier field. Table 3 shows a set of unsorted File Identifiers.

Table 3. File Identifiers

1) File Name	2) SEPARATOR 1	3) File Name Extension	4) SEPARATOR 2	5) File Version Number
MATTERHORN	.	MOUNT	;	1
PIKES_PEAK	.		;	1
	.	HILLS	;	1
SKIING				

The relative value of two File Identifiers is determined in the following way:

- If two File Names have the same content in all byte positions, then these two file names are said to be equal in value.

- If two File Names do not contain the same number of byte positions, the shorter File Name shall be treated as if it were padded on the right with all padding bytes set to (20) (the SPACE character) and as if both File Names contained the same number of byte positions.

[9] ISO 9660:1988, pp. 11, section 7.6.1
[10] ISO 9660:1988, pp.21-22, section 9.3

- After any necessary padding to make the two File Names the same length, the corresponding characters are compared, starting with the first byte position, until a byte position is found that does not have the same character in both File Names. The greater File Name is the one that contains the character whose ASCII value is greater. Table 4 shows the File Names from Table 3 and their relative values.

Table 4. Relative Value of File Names

1) File Name	ASCII Code of first character	Relative value
PIKES_PEAK	80	3
MATTERHORN	77	2
	32	1
SKIING	83	4

The File Name Extensions and Directory Names are also sorted in this manner. The File Version Number is sorted by padding the values with (30) (ASCII "0") on the left to make the lengths equal, then comparing them as above. Table 5 shows the Extensions and Version Numbers from table 3 and their relative values.

Table 5. Relative Value of Extensions and Version Numbers

3) File Name Extension	ASCII Code of first character	Relative Value	5) File Version Number	ASCII Code of first character	Relative Value
MOUNT	77	3	1	31	2
	32	1	1	31	2
HILLS	72	2	1	31	2
	32	1		30	1

The Directory records are then sorted as follows:

- First in ascending order relative to the File Name (or Directory Name)
- Second in ascending order relative to the File Name Extension.
- Third, in descending order relative to the File Version Number.
- Forth, in descending order relative to the value of the Associated File Flag in the File Flags Field. (The associated file comes first, then the file it is associated with).
- Last, in the order of the File Sections of the file. (Only valid if the file is recorded in interleaved mode).

following the above rules, the example started in table 3 is sorted as shown in table 6.

Table 6. Sorted File Identifiers

1) File Name	2) SEPERATOR 1	3) File Name Extension	4) SEPERATOR 2	5) File Version Number
	.	HILLS	;	1
MATTERHORN	.	MOUNT	;	1
PIKES_PEAK	.		;	1
SKIING				

The Path Table

The Path Table indicates to the operating system a short cut to each directory on the disc rather than making the operating system read through each directory to get to the file it needs. This is done primarily to enhance performance. For each directory other than the Root directory, the path table contains a record that identifies the directory, its parent directory, and its location[11].

[11] See Appendix A, Table 16, page VII

Most operating systems read the Path Table once and keep it in memory, rather than reading it over and over again. In the example shown in figure 5, page 9 (the Directory Hierarchy), a system that does not make use of the path table would have to read the root directory to find the location of the ALPS directory, then read the ALPS directory to find the location of the AUSTRIAN directory, then read the SKIING directory to find the location of the file MATTERHORN.MOUNT;1. By making use of the Path Table, the operating system can look up the location of the SKIING directory in the Path Table, read the SKIING directory and find the location of the file. This requires only one seek on the CD-ROM, rather than four. The time difference, for a typical drive with a seek time of 250 msec, is 3/4 of a second. When accessing many files, this difference can significantly affect performance.

Levels of Interchange

ISO 9660 defines three nested levels of interchange which affect the length of the File Identifiers and whether the files must be contiguous or not. Level 1 imposes the most restrictions above and beyond what is specified in ISO 9660. Level 2 impose fewer restrictions, and Level 3 imposes none beyond what is specified in ISO 9660.

Level 1:

- each file shall consist of only one File Section. This means that the files must be contiguous.
- a File Name cannot contain more than eight d-characters or d1-characters.
- a File Name Extension cannot contain more than three d-characters or d1-characters.
- a Directory Identifier cannot contain more than eight d-characters or d1-characters.

An example of Level 1 would be an ISO 9660 disc for the MS-DOS environment, restricted to Level 1 interchange to accommodate MS-DOS's file naming limitations. An important point to note here is that the Level 1 restrictions are more restrictive than the MS-DOS naming conventions. File

Identifiers are still limited to the d-characters as shown in table 1, and Directory Identifiers are limited to d-characters and cannot have an extension.

Level 2:

- each file shall consist of only one File Section. This means that the files must be contiguous.

An example of Level 2 would be an ISO 9660 disc for the Macintosh and UNIX environments, restricted to Level 2 interchange, to allow longer file names.

Level 3:

no restrictions apply beyond what is specified by ISO 9660.

An example of Level 3 would be a CD-ROM-XA disc, which has interleaved data and audio files.

ISO 9660 Implementation Requirements

ISO 9660 also defines requirements for systems that originate or create ISO 9660 volumes and for systems that receive or read ISO 9660 volumes. The requirements for the originating system are primarily of interest to people writing pre-mastering software and will be only briefly mentioned. Pre-mastering is the actual process of creating an ISO 9660 volume. In general, an originating, or pre-mastering system must be able to record a set of files and all of the descriptors described by ISO 9660. The originating system may, however, be restricted to one of the Levels of Interchange. Anyone interested in this level of understanding needs to read the actual ISO 9660:1988 specification.

The receiving system is the system that reads an ISO 9660 disc and makes the data accessible to the user. The receiving system is expected to be able to read the files and descriptors on a volume that

meets at least one of the above interchange levels. It is also expected to make the information in these files available to the user. All of them except for the Associated Files, which we will discuss in more detail later. The receiving system is also expected to make available to the user much of the information in the Volume Descriptors and Path Table. All of the following information should be visible to the user:

From the Primary Volume Descriptor:

- Volume Identifier
- Volume Set Identifier
- Copyright File Identifier
- Abstract File Identifier
- Bibliographic File Identifier

From each Supplementary Volume Descriptor:

- Volume Identifier
- Bit 0 of the Volume Flags field, which indicates if the escape sequences used are registered
- Escape Sequences, which define the d1-character set
- Volume Set Identifier
- Copyright File Identifier
- Abstract File Identifier
- Bibliographic File Identifier

From each Path Table Record:

- Parent Directory Number
- Directory Identifier

From each Directory Record:

- File Name of a File Identifier
- File Name Extension of a File Identifier
- Directory Bit of the File Flags field

Similar to the Levels of Interchange, which apply to the originating system, ISO 9660 also defines Levels of Implementation, which apply to the receiving system:

Level 1 implementation:

- At level 1, the receiving system does not have to make any of the data contained in and pointed to by the Supplementary Volume Descriptors available to the user. Most implementations are Level 1.

Level 2 implementation has no such restrictions.

Implementations of ISO 9660

As figure 7. shows, each operating system has its own specific file system. Independent of the operating systems is ISO 9660.

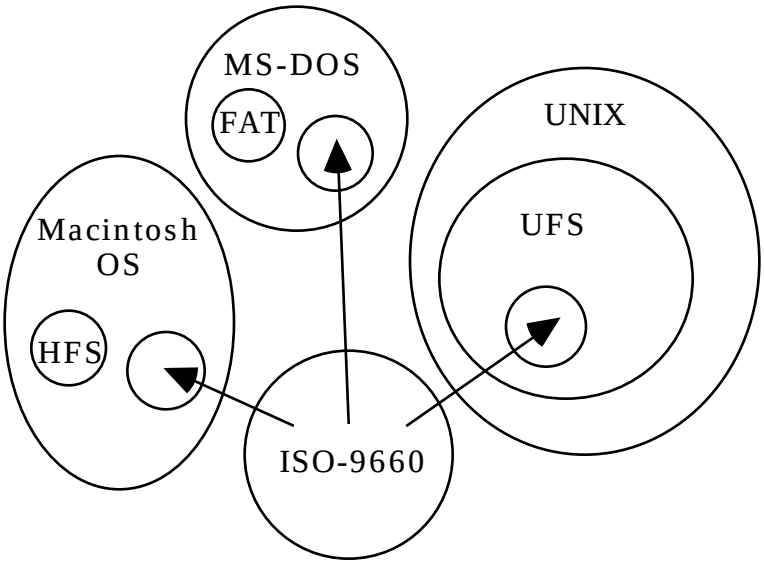

Figure 7. ISO 9660 World View

In order for a given platform to transparently implement ISO-9660, the operating system must convert the ISO-9660 file system into something that looks like its normal file system. This way, the application programs can use the ISO-9660 disc as if were a native read only file system.

DOS

Under MS-DOS, a program called Microsoft CD-ROM Extensions (MSCDEX.EXE) intercepts operating system calls to the CD-ROM and makes the ISO-9660 volume look like a normal, read only, hard disk. However, MS-DOS does not support File Version Numbers, which are part of an ISO 9660 File Identifier. To properly handle the File Version number, when MS-DOS requests a File Identifier from the ISO-9660 volume, Extensions modifies the File Identifier it returns to the operating system by stripping off the File Version Number and only returning the file with the highest version number. To the operating system and the user, the ISO-9660 disc is accessed using a drive letter, just like a write protected hard disk or floppy disk, or network drive.

When creating discs for the DOS environment, the biggest pitfall that developers encounter is that the ISO 9660 file naming conventions. ISO-9660 is much more restrictive than DOS as to the characters that may be used, but much less so as to the length of the names.

MS-DOS file names are limited to eight characters in the file name, and 3 characters in the file name extension. ISO-9660 File Identifiers can be up to 30 characters, and does not specifically limit the length of the Name or the Extension. If the pre-mastering system is DOS based, the length is not a problem, but if the disc is being pre-mastered on a Macintosh or UNIX machine, for example, the file names may be longer than DOS allows. When this disc is read on a DOS machine, the names appear to be truncated, but DOS cannot access the files, as shown in table 7.

Table 7. Long ISO File Identifiers under MS-DOS

Original ISO-9660 File Identifier	File Name as seen by MS-DOS	Response when accessed
MATTERHORN.MOUNT;1	MATTERHO.RNM	File Not Found
EVEREST.MNT;1	EVEREST.MNT	OK

MS-DOS allows a much larger group of characters to be included in file names than does ISO -9660. If an ISO-9660 Volume is made using characters that are allowed under MS-DOS, but do not conform to the d-character set, as shown in figure 2, there is a good chance that some files will not be readable. The files appear normally in the directory, but DOS may or may not be able to open all of the files. This problem is particularly perplexing since the file with the illegal character can be read, but the next file after it may not be. This problem occurs if you have a set of files, which have the same names up to some point, after which one file has an illegal character, and one file a SEPARATOR 1 (period, "."), as demonstrated in table 8.

Table 8. Illegal d-characters and Microsoft extensions

INSTALL-1.TXT;1	Reads OK
INSTALL.BAT;1	Reports "File not found"
INSTALL.EXE;1	Reports "File not found"
INSTALL.TXT;1	Reports "File not found"

In the example shown in table 10, the first name has an illegal ISO-9660 character, the dash (-). This file is readable. However, when attempting to read any of the other three files, MS-DOS reports "File not found". This appears to be occurring because Microsoft Extensions assumes that the directory records are sorted properly and when it sees a name that, if sorted properly, would come after the name it is searching for, it stops searching and proclaims that it cannot find the file. In the example shown in table 9, if MS-DOS is searching for INSTALL.BAT, it reads the first File Name and compares it to INSTALL.BAT. If properly sorted according to the rules specified in ISO 9660, INSTALL.BAT would appear before INSTALL-1.TXT, as shown in table 11. However, the actual order of the records has INSTALL-1.TXT first, before any of other INSTALL files. Therefore, when MS-DOS searches for INSTALL.BAT, it sees INSTALL-1.TXT, which should come after INSTALL.BAT, and comes to the conclusion that INSTALL.BAT is not there.

Table 9. Sorting illegal ISO-9660 File Identifiers

Order of directory records in ISO-9660 Volume	Correctly sorted directory records
INSTALL-1.TXT;1	INSTALL.BAT;1
INSTALL.BAT;1	INSTALL.EXE;1
INSTALL.EXE;1	INSTALL.TXT;1
INSTALL.TXT;1	INSTALL-1.TXT;1

What appears to be causing this is that most pre-mastering packages do not sort the names properly **IF** they are told to put illegal d-characters in the ISO 9660 volume. In particular, they appear to no longer sort the File Name and the File Name Extension separately, but treat the entire File Identifier as the File Name. The best way to avoid this is to not have any illegal d-characters in the File Names, Directory Names, or File Name Extensions.

Macintosh

The Macintosh supports ISO 9660 by adding an extension onto the operating system, called the foreign file access extension. This operating system extension, along with code that tells it how to convert ISO-9660, makes an ISO-9660 disc appear on the desktop just like any other write-protected HFS Volume, with its own icon on the Macintosh desktop.

Even though the ISO 9660 disc looks like a normal volume, not all of the data that the Macintosh needs to display the volume on the desktop is available in the ISO 9660 data. The additional information gets created by the ISO-9660 access software when an ISO directory is opened. The result of this is that the user has no control over the placement of folders and files, and an ISO 9660 disc loses some of the look and feel so important to the Macintosh.

Another problem that occurs with ISO 9660 on the Macintosh has to do with how the Macintosh file system implements executable files, or applications as they are known to Mac users. All of the files on a standard ISO 9660 disc will appear on the desktop as generic documents. To correct this deficiency, Apple uses the reserved system use field in the directory record and associated files in ISO 9660 to add extra information that allows applications to run from an ISO 9660 disc. See the discussion on extensions to ISO 9660 for more details, page 29.

For some developers, another source of difficulties is the File Version Number. The File Version Number is added to each file when it is pre-mastered to make the File Identifiers comply with the ISO specification. Applications that worked from the hard disk may not work from the CD-ROM for the simple reason that they are trying to find a file whose name now has a ';1' on the end of it.

UNIX

UNIX systems typically support ISO 9660 by including the programs into the operating system, rather than using external conversion programs or extension drivers. This is normally done by recompiling the operating system along with the new code to support CD-ROM and ISO 9660. The CD-ROM can then be "mounted" on an existing directory in the UNIX directory structure. On UNIX style systems, instead of having different drive letters or volumes, different drives, or devices, are "mounted" on a directory. This means that each UNIX system only has one directory structure, and all devices are part of that structure as shown in figure 8.

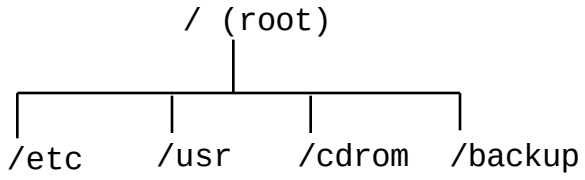

Figure 8. UNIX directory

In this example, the `etc` directory may be on the hard disk with the operating system, the `usr` directory may be a second hard disk used to store user's files, the `cdrom` directory may be an ISO-9660 disc, and the `backup` directory may be a network drive on an entirely different system.

Unfortunately, because of the way UNIX has historically been implemented, there is quite a bit of variation among UNIX systems as to how they support CD-ROM and how the files on the CD-ROM

will appear to the user. As there are well over 100 UNIX compatible Operating Systems currently available, no attempt will be made to address each one separately. The best way to find out how a system will react is to read the manual pages for the mount command and for cdrom and the CD-ROM file system. Some of the most common idiosyncrasies will be mentioned, however.

Some UNIX systems do not modify the file names at all and the user will see upper case files names, with the version number appended. This seems reasonable, since it shows the complete File Identifier, but on many systems the semi-colon character has special meaning and causes access difficulties. Other systems will strip off the version number in much the same way that MSDOS does, and have the option to either convert the characters to lower case or to leave them uppercase. Of the systems that have this option, some default to converting to lower case, while others default to leaving everything upper case. See table 10 for examples of what these different options look like.

Table 10. UNIX File name conversions

ISO 9660 File Identifier	Conversion being applied	File Name as it appears to user
MANPAGES.1;1	none	MANPAGES.1;1
MANPAGES.1;1	no version number	MANPAGES.1
MANPAGES.1;1	lower case, no version number	manpages.1

This provides the system administrator with a lot of flexibility, but makes it difficult for applications that have file names "hard coded" into the programs. These discs have to include clear instructions to the user to have the system mount the disc with filenames in the proper case. Anyone who has been involved in porting applications to different UNIX systems will find this situation not at all unusual.

Extensions to ISO 9660

ISO 9660 works well for a majority of operating systems. In some cases, however, it has proven to be difficult or impossible to use. To make it more usable in their respective environments, Apple and the UNIX community established extensions to ISO 9660. The Apple Extensions were created by Apple and are generally known as Apple ISO 9660. The UNIX community assembled a group of people created what is known as the Rock Ridge Proposal.

Recordable CD, or CD-R, opens up the possibility of updating information already on a CD-ROM. ISO-9660 was never intended to support this function. To meet this new requirement, a group met at Frankfurt, Germany and developed an update to ISO 9660. This update is called the Frankfurt Group Proposal - Volume and File Structure for Read-Only and Write-Once Compact Disc. At this point in time, the Frankfurt Proposal has been approved by the European Computer Manufacturers Association as standard ECMA 168. The Frankfurt Proposal is very complex and difficult to implement. To simply provide a way to update information on an ISO 9660 volume, a much simpler method has been implemented by several companies. This method is commonly known as Updatable ISO 9660 or Multi-Session ISO 9660.

Apple ISO 9660[12]

The Macintosh operating system requires a lot of specialized data to support its graphical user interface. The information needed to implement these features is stored in two places by the HFS file system. Some information is stored in special fields in the HFS directory record. Most of the data is stored in the file itself, in a specially formatted area called the resource fork. The data in the resource

[12] For further information, see "CD-ROM & the Macintosh Computer, A Gentle, Technical Introduction to Creating CD-ROMS for the Apple Macintosh computer family" and "The Apple CD-ROM Handbook"

fork is normally only accessible through system calls to a part of the Macintosh Operating system known as the Resource Manager. The resource fork contains things like the Menu layout, the Window definitions, user preferences, and text messages that the application displays, and the actual executable code in the case of an application. The remaining data in a file is called the data fork. The data fork is the same as a DOS or UNIX file.

Most Macintosh files have both forks, a data fork and a resource fork. ISO 9660 accommodates the resource fork very well through the use of the associated file. When a Macintosh file is recorded in an ISO volume, the resource fork is recorded as an associated file, and the data fork is recorded as a normal file.

Unlike the resource fork, there is some information vital to the Macintosh environment that can not be stored on a standard ISO 9660 volume. The native Macintosh file system, HFS, stores information regarding the file icon's position on the screen, what the icon looks like, what type of file it is, what application created it, and file attributes. The file attributes contain information such as if the file is visible, if the file is locked, and if it is an alias. To capture this information on an ISO 9660 volume, Apple created an extension to ISO 9660, making use of the System Use Field in the Directory Record. This field is allocated in ISO 9660, but how it is used is left open.

Apple was not, however, able to make an ISO disc look exactly like an HFS disk. Currently, there is no way to record the positions of icons on the desktop in an ISO 9660 volume, so these are created when a directory is opened, with no way to control where they are located. Also, because of the way the Finder works, files and folders on an ISO disc only appear with generic icons, as seen in figure 9.

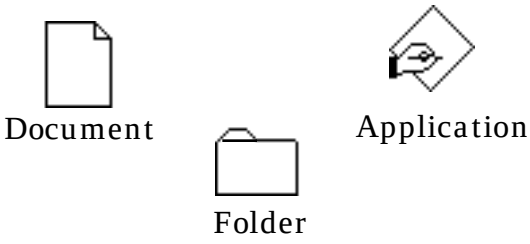

Figure 9. Apple Macintosh generic Icons

These generic icons are displayed even if the correct icons are in the Apple extensions area because the Finder assumes there is a desktop database from which to retrieve these icons and uses a special call to retrieve them. This part of the Finder was written to be very HFS specific and does not work with ISO 9660 volumes even if there is a file called desktop in the volume. If these files are copied to an HFS volume, however, the correct icon will be shown. If the Apple extensions are not present, all files will show as generic documents.

The Protocol Identifier

To identify a volume as having the Apple extensions and inform the operating system what type of file name translation to perform, the System Identifier field in the Primary Volume Descriptor is defined to be the following[13]:

"APPLE COMPUTER, INC., TYPE: " followed by a four byte identifier.

The four byte identifier tells the system what version of Apple Extensions this is and whether or not to perform automatic file name translation for ProDOS (used on the Apple //GS). A typical identifier, that tells the system not to perform ProDOS translation and that the version number is 2 is "0002".

[13] "CD-ROM & the Macintosh Computer...", pp. 23

The Directory Record System Use Field

The System Use Field[14] of an ISO 9660 Directory record is used to store the additional information needed by the Macintosh Operating system. The additional fields are defined as shown in table 11[15].

Table 11. Apple ISO 9660 Directory Record System Use Field

BP	Field Name	Content
1 to 2	Signature ID	(41)(41) "AA" Apple signature
3	SystenUse Extension Length	(0E) bytes
4	System Use ID	(02) for HFS
5 to 8	HFS fileType	(MSB-LSB)
9 to 12	HFS fileCreator	(MSB-LSB)
13 to 16	HFS finder flags	(MSB-LSB)

Apple also allows more than one System Use Extension field in a single directory entry, limited only by the size of the directory entry (it cannot be larger than a logical block). Apple intended for the Signature ID to be used to identify extensions for different systems. Each system could then ignore the fields whose Signature ID it did not recognize. This method of sharing the System Use field was later adopted by both the Rock Ridge Group and the Frankfurt Group for extending the ISO 9660 Directory record.

[14] See ISO 9660:1988, pp. 21, section 9.1.13
[15] "CD-ROM & the Macintosh Computer...", pp. 25

The Rock Ridge Proposals

Rock Ridge is a group of companies that began meeting in July 1990 to resolve issues with ISO 9660 that make it difficult to use as a distribution medium for some operating systems (UNIX based systems being their primary concern). Some of the issues addressed include long filenames with lower case characters, directory structures much deeper than the eight levels allowed by ISO 9660, different file types and access privileges. The Rock Ridge proposals offer industry standard solutions for the distribution of data and software on CD-ROM media by extending the ISO 9660:1988 specification while remaining completely compliant with it.

These proposals deal with two main areas. First, it establishes a standardized way for multiple file system extensions to coexist in one ISO 9660 Directory record. I then defines a way to record POSIX[16] files and directories in an ISO volume without modifying their original directory information. POSIX is a standard for a Portable Operating System Interface much of which is based on how the UNIX operating system works. This allows standard UNIX style file names and directories to be recorded in an ISO 9660 volume without any modification.

Rock Ridge System Use Sharing Protocol (SUSP)

The System Use Sharing Protocol provides a standard way for multiple systems to record system specific extensions in the System Use field by defining a generic field format for System Use Fields, and a set of generic System Use Fields for that can be used to:

- continue the System Use Fields in an area outside of the directory record
- do additional padding
- identify that SUSP is being used

[16] Institute of Electrical and Electronic Engineers Portable Operating System Interface IEEE Std. 1003.1-1990

- terminate the SUSP area
- identify which system specific extension is being used.

The System Use Field Format is as shown in table 12.

Table 12. SUSP System Use Field

BP	Field Name	Content
1 to 2	Signature Word	Identifies what type of System Use Field this is
3	Length of System Use Field (LEN_SUF)	bytes
4	System Use Field Version	Version Number of System Use Field
5 to LEN_SUF	Data	Content of System Use Field

The Signature Word identifies what the following system use field will be used for. The Signature Words that are defined by SUSP are:

"CE" Continuation Area. This field points to the Logical Block Number where the System Use Area is continued. This allows the System Use data to extend beyond a single Logical Block.

"PD" Padding Field. More than one of this field may appear in any given System Use Area. This is used to fill up empty areas such as might occur at the end of a Logical Block before going on to the Continuation Area.

"SP" System Use Sharing Protocol Indicator. Only one of this field can be in a Volume. It must be in the System Use Area of the first directory record of Root directory. This identifies the Volume as adhering to the System Use Sharing Protocol.

"ST" System Use Sharing Protocol Terminator. This field is an optional field to mark the end of SUSP's use of the System Use area.

"ER" Extensions reference. This field may be mandatory or not, depending on the specification which it describes. This is determined by specifications that make use of the System Use Sharing Protocol.

For more information regarding these System Use Fields, see the System Use Sharing Protocol proposed specification, available on the DMI "demo" disc. The System Use Sharing Protocol does not provide new features by itself, but provides a common base on which to establish new ISO 9660 extensions, such as the Rock Ridge Interchange Protocol.

Rock Ridge Interchange Protocol (RRIP)

Rock Ridge Interchange Protocol was designed to allow users of POSIX and other UNIX like systems to retain much of the directory information that is in the native file system. These systems use directory entries for much more than just pointing to files. Directory entries can point to other entries (symbolic links or aliases) or to device drivers that are linked to peripheral devices such as hard disks, tape drives and CD-ROM drives (device files). The directory entry includes information that lets the system know the file type. Is it a regular file, a directory, a symbolic link, or a device file? The Directory entry also has information regarding who has permission to read, write and execute each file. Most of these systems are multi-user systems, and must be able to limit who can

write to the device file that contains the operating system, or it could accidentally (or purposely?) be erased.

File Names

The Rock Ridge Interchange Protocol is intended to be portable across a variety of POSIX compliant systems, so it makes suggestions that increase portability. However, unlike ISO 9660, it does not set hard and fast limits on the character set that can be used in file names and directory names. Systems that support RRIP must, however, treat file names that are the same except for the case of the letters as being different files (a file named `alpine` is not the same as a file named `Alpine`). RRIP suggests that the only the characters in table 13 be used for maximum portability.

Table 13. Suggested Characters for RRIP File Identifiers

Suggested Characters	ASCII Hexadecimal Code
'A' through 'Z"	(41) - (5A)
'a' through 'z'	(61) - (7A)
period .	(2E)
underscore _	(5F)
hyphen -	(2D)

Deep directories

On most POSIX type systems, not only do the file names tend to be long, but the directories tend to get much deeper than the 8 levels allowed by ISO 9660. For this reason, RRIP defines a way to remap deep directories that allows it to be ISO compliant, while at the same time retaining the deep directory structure on systems supporting RRIP.

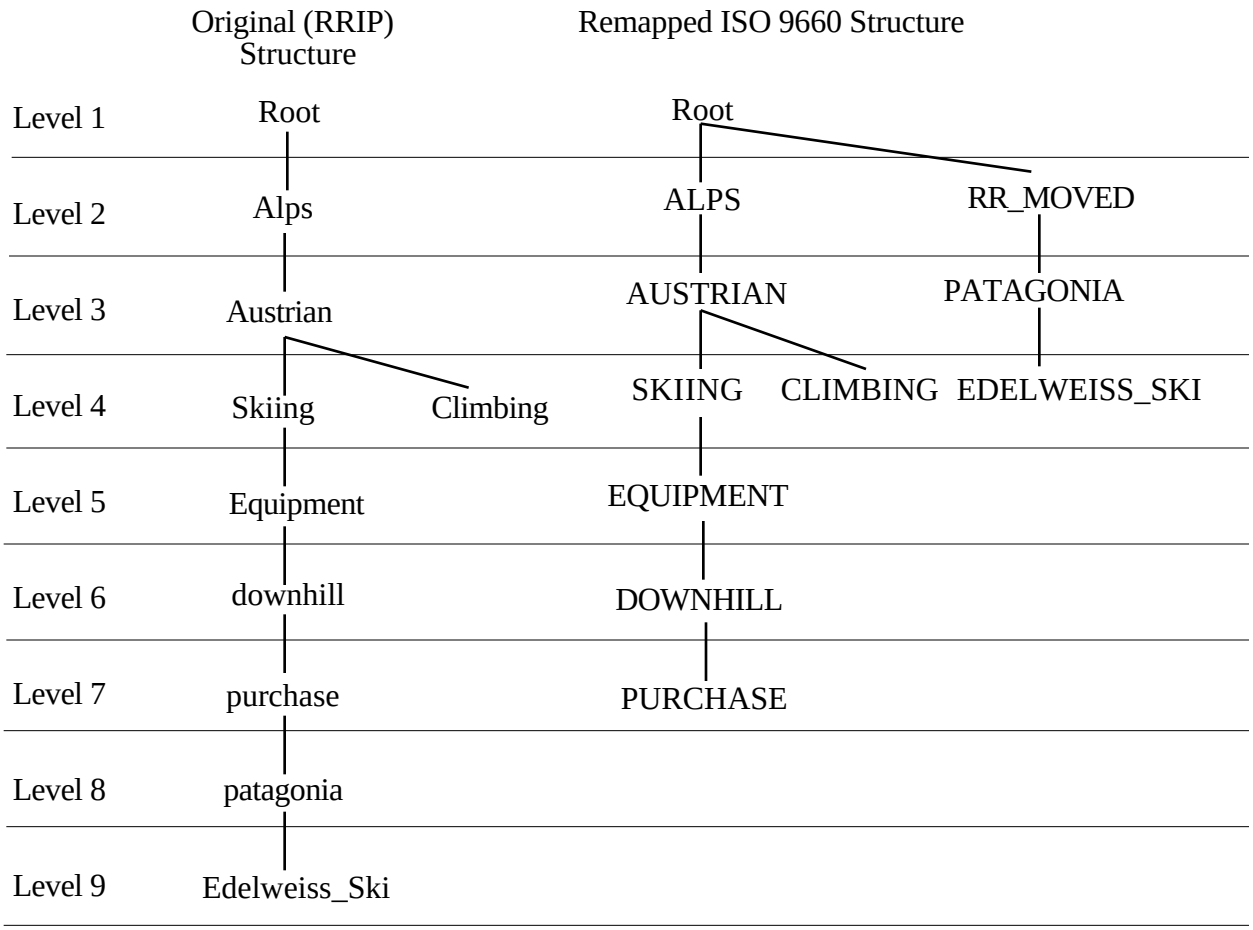

Figure 10. Remapped Directory structure

A directory that was originally at the eighth level is relocated higher up in the directory structure as shown in figure 10. Generally, the pre-mastering software will handle relocating directories automatically. Probably the most common pre-mastering package that supports RRIP, Makedisc from Young Minds, Inc., creates a new directory at the root level, called RR_MOVED, and places all relocated directories here. This directory is only visible on systems that do not support RRIP. On RRIP systems, you see the original, deep, directory structure.

For more information on the exact layout of the RRIP System Use Fields, see the Rock Ridge Interchange Protocol proposed specification, available on the DMI "demo" disc.

Updatable ISO 9660

A simple method for allowing CD-WO discs to be updated has been implemented by several companies. This technique involves writing a new, complete directory structure each time the disc is updated. The new structure is recorded as an ISO 9660 structure, starting at 00:02:16 into the latest recording session, another session being recorded each time the disc is updated. For an explanation of recording sessions and CD-WO in general, see the Compact Disc Terminology paper, available from DMI, and the Philips and Sony Orange book, *Recordable Compact Disc Systems.* A simple volume that has not been updated, and then the same disc after updating is shown in figure 11. The ISO 9660 directory structure in Session 1 only reflects files 1 and 2. The directory structure recorded in session 2, however, reflects not only files 3 and 4, recorded during session 2, but also files 1 and 2, recorded in the session 1.

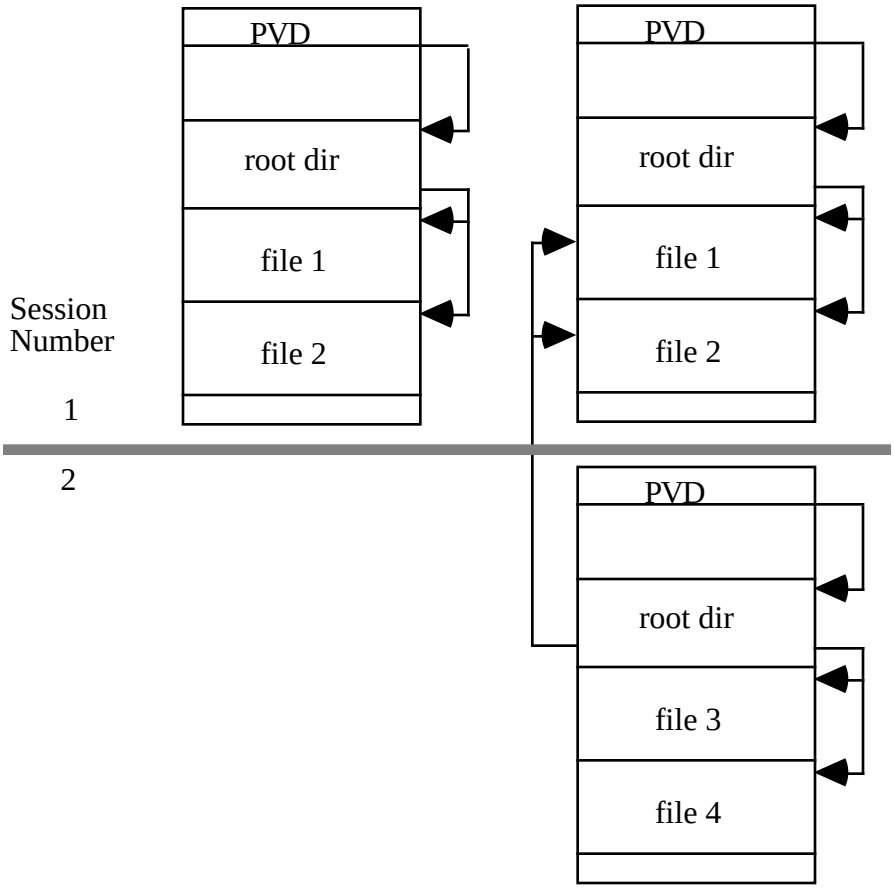

Figure 11. Updatable ISO 9660

In order for a CD-ROM drive to recognize the updated areas, the drive must be able to recognize that there are multiple sessions on the disc, and the ISO 9660 implementation must be able to use the Primary Volume Descriptor from the last session on the disc. This is commonly known as "Multi-Session" compatibility.

The Frankfurt Group Proposal, ECMA 168

The Frankfurt Group is an ad hoc group of companies which share common interests and goals concerning Read Only and Write Once Compact Disc Technology. The Write Once Technology is governed by the Philips and Sony Orange Book, *Recordable Compact Disc Systems*. The Frankfurt Group Proposal covers two types of Volume and File Structures. Type 1 is the original ISO 9660 specification, and is used only for Read-Only discs. Type 2, the Frankfurt Proposal, is an extension to Type 1 that allows incremental recording of a Compact Disc Write Once (CD-WO or CD-R) and updates to be recorded to an existing CD-WO. This proposal is much more complex and flexible than the Updatable ISO 9660. In addition to providing a way to incrementally record and update discs, Type 2 also defines a framework that furnishes the same type of functionality that the Rock Ridge Proposals provide. The specification, however, is not the same as the Rock Ridge Proposals. For more information regarding the Frankfurt Proposal, see ECMA 168 - Volume and File Structure for Read-Only and Write-Once Compact Disc.

Summary of ISO 9660

- An ISO 9660 volume consists of the following data structures:
  - Volume descriptors (what is this volume, and where are the other data structures)
  - The Directory Structure (where are and what are the names of the files and directories)
  - The Path Table (what are the locations and parent directories of each directory)
- ISO 9660 defines three levels of interchange for the Originating System:
  - Level 1 (File names limited to 8.3, directory names limited to 8, and all files must be contiguous) To insure compatibility across the most platforms, restrict the ISO volume to Level 1 Interchange.
  - Level 2 (All files must be contiguous)
  - Level 3 (no limitations beyond the ISO specification itself)
- ISO 9660 defines two levels of Implementation for the Receiving System:
  - Level 1 (Receiving System can ignore any Supplementary Volume Descriptors)
  - Level 2 (Receiving System must utilize all data available)
- Example of ISO 9660 Implementations:
  - MS-DOS:
    - MS-DOS supports ONLY Level 1 Interchange.
    - Access to ISO 9660 volumes is performed by Microsoft Extensions, MSCDEX.EXE.
    - Extensions strips off the version number and only recognizes the highest version.
    - Extensions can not find files and directories with File Identifiers longer than 8.3.
    - Extensions does not handle illegal ISO characters (non d-characters) very well.
  - Apple Macintosh:
    - Apple Macintosh supports Level 2 Interchange. Volumes that must be usable under MS-DOS, as well as Macintosh, must be restricted to Level 1 Interchange.

- Access to ISO 9660 volumes performed by system extensions.
- Layout of folders and files on desktop is created when an ISO 9660 directory is opened.
- ISO 9660 disc must include Apple extensions to run Macintosh applications from the ISO volume.
- ISO 9660 file identifiers include the version number.

- UNIX
  - Most UNIX type systems support Level 2 Interchange. Volumes that must be usable under MS-DOS, as well as UNIX, must be restricted to Level 1 Interchange.
  - Access to ISO 9660 volumes is usually incorporated into the operating system.
  - There is considerable variation between UNIX type implementations as to how the File Identifiers appear to the user.
  - Some systems have options to convert File Identifiers to lower case, and remove the File Version Number, so the same volume can appear different, even on the same machine.

• Extensions to ISO 9660
  - Apple ISO 9660 provides the Macintosh system with additional data needed to launch applications from an ISO 9660 volume.
  - The Rock Ridge Proposals provide a more UNIX like environment for distributing data to a variety of UNIX like platforms.
  - Updatable ISO 9660 provides a simple way to add more data to a previously recorded CD-WO.
  - The Frankfurt Group Proposal, ECMA 168, provides a way to append data to a CD-WO, and provide a more UNIX like environment for distributing data to a variety of UNIX like platforms.

Appendix A: ISO 9660 Structures

Table 14. Primary Volume Descriptor

BytePosition	Field Name	Content
1	Volume Descriptor Type	1
2 to 6	Standard Identifier	CD001
7	Volume Descriptor Version	1
8	Unused Field	(00)[17] byte
9 to 40	System Identifier	a-characters allowed[18]
41 to 72	Volume Identifier	d-characters allowed[19]
73 to 80	Unused Field	(00) bytes
81 to 88	Volume Space Size	Number of logical blocks in the Volume
89 to 120	Unused Field	(00) bytes
121 to 124	Volume Set Size	The assigned Volume Set size of the Volume
125 to 128	Volume Sequence Number	The ordinal number of the volume in the Volume Set
129 to 132	Logical Block Size	The size in bytes of a Logical Block
133 to 140	Path Table Size	Length in bytes of the path table
141 to 144	Location of Type L Path Table	Logical Block Number of first Block allocated to the Type L Path Table, Type L meaning multiple byte numerical values are recorded with least significant byte first. This value is also recorded with least significant byte first.
145 to 148	Location of Optional Type L Path Table	0 if Optional Path Table was not recorded, otherwise, Logical Block Number of first Block allocated to the Optional Type L Path Table.
149 to 152	Location of Type M Path Table	Logical Block Number of first Block allocated to the Type M Path Table, Type M meaning multiple byte numerical values are recorded with most significant byte first. This value is also recorded with most significant byte first.
153 to 156	Location of Optional Type M Path Table	0 if Optional Path Table was not recorded, otherwise, Logical Block Number of first Block allocated to the Type M Path Table.

[17] Numbers surrounded by parentheses () are hexadecimal numbers.

[18] a-characters are A-Z, 0-9, _, space, !, ", %, &, ', (,), *, +, ,, -, ., /, :, ;, <, =, >, ?
see ISO-9660:1988, Annex A, Table 15

[19] d-characters are A-Z, 0-9, _
see ISO-9660:1988, Annex A, Table 14

157 to 190	Directory record for Root Directory	This is the actual directory record for the top of the directory structure. See the section on directory records for the format of this data.
191 to 318	Volume Set Identifier	Name of the multiple volume set of which this volume is a member. d-characters allowed.
319 to 446	Publisher Identifier	Identifies who provided the actual data contained in the files. a-characters allowed.
447 to 574	Data Preparer Identifier	Identifies who performed the actual creation of the current volume. a-characters allowed.
575 to 702	Application Identifier	Identifies the specification of how the data in the files are recorded. For example, this field might contain SGML if the files were recorded according to the Standard Generalized Markup Language
703 to 739	Copyright File Identifier	Identifies the file in the root directory that contains the copyright notice for this volume. If there is no copyright file, this field should contain all spaces (20) Level 1 interchange restrictions apply.[20]
740 to 776	Abstract File Identifier	Identifies the file in the root directory that contains the abstract statement for this volume. If there is no copyright file, this field should contain all spaces (20) Level 1 interchange restrictions apply.
777 to 813	Bibliographic File Identifier	Identifies the file in the root directory that contains bibliographic records. ISO-9660 does not specify the format of these records. If there is no copyright file, this field should contain all spaces (20) Level 1 interchange restrictions apply.
814 to 830	Volume Creation Date and Time	Date and time at which the volume was created. Represented by seven bytes: 1: Number of years since 1900 2: Month of the year from 1 to 12 3: Day of the Month from 1 to 31 4: Hour of the day from 0 to 23 5: Minute of the hour from 0 to 59 6: second of the minute from 0 to 59 7: Offset from Greenwich Mean Time in number of 15 minute intervals from -48(West) to +52(East)
831 to 847	Volume Modification Date and Time	Date and time at which the volume was last modified. Represented the same as the Volume Creation Date and Time
848 to 864	Volume Expiration Date and Time	Date and Time at which the information in the volume may be considered obsolete. Represented the same as the Volume Creation Date and Time
865 to 881	Volume Effective Date and Time	Date and Time at which the information in the volume may be used. Represented the same as the Volume Creation Date and Time

[20] For a description of the level 1 interchange restrictions, see page <?>

882	File Structure Version	1
883	Reserved for future standardization	(00)
884 to 1395	Application Use	This field is reserved for application use. Its content is not specified by ISO-9660.
1396 to 2048	Reserved for future standardization	All bytes must be set to (00).

Table 15. Directory Record

BP	Field Name	Content
1	Length of directory Record (LEN_DR)	Bytes
2	Extended Attribute Record Length	Bytes - this field refers to the Extended Attribute Record, which provides additional information about a file to systems that know how to use it. Since few systems use it, we will not discuss it here. Refer to ISO 9660:1988 for more information.
3 to 10	Location of Extent	This is the Logical Block Number of the first Logical Block allocated to the file.
11 to 18	Data Length	Length of the file section in bytes
19 to 25	Recording Date and Time	This is recorded in the same format as the Volume Creation Date and Time
26	File Flags	One Byte, each bit of which is a Flag: Bit 0 File is Hidden if this bit is 1 1 Entry is a Directory if this bit is 1 2 Entry is an Associated file is this bit is 1 3 Information is structured according to the extended attribute record if this bit is 1 4 Owner, group and permissions are specified in the extended attribute record if this bit is 1 5 Reserved (0) 6 Reserved (0) 7 File has more than one directory record if this bit is 1
27	File Unit Size	This field is only valid if the file is recorded in interleave mode. Otherwise this field is (00)
28	Interleave Gap Size	This field is only valid if the file is recorded in interleave mode. Otherwise this field is (00)
29 to 32	Volume Sequence Number	The ordinal number of the volume in the Volume Set on which the file described by the directory record is recorded.
33	Length of File Identifier (LEN_FI)	Byte

34 to (33 + LEN_FI)	File Identifier	Interpretation depends on the setting of the directory bit in the File Flags
		
		If set to ZERO, then
		
		The field refers to a File Identifier, as described below
		
		If set to ONE, then
		
		The field refers to a Directory Identifier, as described below.

34 + LEN_FI	Padding Field	Present only if the length of the File Identifier is an even number. If present, value is (00)
LEN_DR - LEN_SU + 1	System Use (LEN_SU)	Reserved for system use. If necessary, so that the length of the directory record is an even number of bytes, a (00) byte may be added to terminate this field.

The Path Table Record contains the following fields:[21]

Table 16. Path Table Record

BP	Field Name	Content
1	Length of Directory Identifier (LEN_DI)	Length in Bytes
2	Extended Attribute Record Length	If an Extended Attribute Record is recorded, this is the length in Bytes. Otherwise, this is (00)
3 to 6	Location of Extent	Logical Block Number of the first Logical Block allocated to the Directory
7 to 8	Parent Directory Number	The record number in the Path Table for the parent directory of this directory
9 to (8 + LEN_DI)	Directory Identifier	This field is the same as in the Directory Record
(9 + LEN_DI)	Padding Field	Present only if LEN_DI is an odd number. (00)

[21] ISO 9660:1988, pp. 22, section 9.4

Appendix B: Common Q&A

1. What do I need to do to make my MS-DOS data ready for ISO-9660?

There are three things that need to be checked to insure that your disc will painlessly translate to ISO-9660:

1- The characters used in the File Names must only be A-Z, 0-9, and _. See figure 3, on page 6.

2- The depth of the directory structure can not exceed 8 levels. See figure 5, on page 9.

3- The length of the path to any file can not exceed 255 characters. See Table 1, on page 10.

To improve performance, you may also want to minimize the number of files in each directory. If there are over 50 files in a directory, you may notice some slowing while reading files in this directory, If there are over 250 files, you are very likely to notice that it takes much longer to read some files in this directory.

2. What do I need to do to make my Macintosh data ready for ISO-9660?

See question 1. In order for a disc to have Macintosh applications on it, the ISO-9660 data must include the Apple extensions, otherwise, every file on the disc will appear to be a text file to the Macintosh. Also, ISO-9660 discs will have the version number appended to each file name. See Macintosh implementation of ISO-9660, page 23, and Apple Extensions to ISO-9660, page 26.

3. Why use ISO 9660 on the Mac and what are the affects?

The primary reason for using ISO-9660 on a disc that will be running on the Macintosh is that the same disc will need to run on other platforms, such as MS-DOS. The primary disadvantage to using ISO-9660 on the Macintosh is the loss of some of the "look and feel" of the Macintosh user interface. In particular, you must live with the restrictions on the file names (see File Names, page 12), and you can not control where the folders open up to

and how they are viewed (see Macintosh implementation of ISO-9660, page 23, and Apple Extensions to ISO-9660, page 26).

4. What is a hybrid disc; what are the issues to consider?

In an effort to alleviate the disadvantages to using ISO-9660 on the Macintosh (see question 3), a scheme called the hybrid disc has been developed. A hybrid disc is basically a disc with two partitions on it. It has both an ISO-9660 partition and an HFS partition. When this disc is mounted on the Macintosh, the Macintosh only sees the HFS partition. On other platforms, only the ISO-9660 partition is visible.

Until recently, the primary disadvantage to a hybrid disc was that any data that was common to the different platforms had to be duplicated in both partitions. This could significantly limit the amount of data you could place on a hybrid disc. There is now becoming available a type of hybrid disc that blends the two partitions. This allows us to only put data on the disc once, and have both ISO-9660 and HFS directory structures point to the same data.

5. How do I make one disc that works on both my PC and Macintosh?

see questions 3 and 4. Often, a disc such as this is created entirely on the Macintosh. An important point to note here is that if the ISO-9660 partition is created on the Macintosh, any files that must be readable by MS-DOS must meet ISO-9660 level 1 Interchange. That is, the file names can not be longer than 8.3 (see Levels of Interchange, page 16 and Implementations of ISO-9660, DOS, page 20).

6 .What do I need to do to make my UNIX data ready for ISO-9660?

See question 1. The primary differences between ISO-9660 for UNIX and MS-DOS is that UNIX generally can support Interchange level 2 (file names longer than 8.3), and UNIX typically can handle larger subdirectories with less performance degradation. An important point to remember when dealing with ISO-9660 and UNIX is that the file names can appear

different on different systems. Even on the same system, different mount options can affect how, or if, the system translates the ISO-9660 file names. (See Implementations of ISO-9660, UNIX, page 24).

7. Will ISO discs work on UNIX, MAC and DOS?

Yes, but... Only files that meet the proper level of Interchange will be readable on any given platform, i.e., UNIX and Mac files that are recorded longer than 8.3 will not be readable by DOS (see Implementations of ISO-9660, DOS, page 19).

8. What is an Image file?

In the jargon of the industry, an Image file is a single data file that contains all of the data that will be recorded to a CD. Typically, this file will contain a complete ISO-9660 volume, or an HFS volume, or a hybrid image with both ISO and HFS volumes. If you can generate an image file, your data is *image ready.*

9. How do I create an Image file?

There are software packages available that will create an ISO-9660 volume and record it to an Image file. These packages are normally call pre-mastering packages and are available from companies such as Dataware Technologies, Meridian Data, Optical Media International, and Trace. There are too many packages to try and list them all here. The best way to choose one is to read reviews and talk to people who are using the package you are considering purchasing.

10. How do I send the Image file to the Mastering Facility?

Most pre-mastering packages contain options to write an ISO-9660 volume to a SCSI tape drive as well as to an Image file. Some packages also support writing to a CD Recordable device. Most mastering houses will accept 8mm Exabyte tapes, 4mm DDS tapes, and CD-R discs as standard input. If none of these options are available, but you do have an Image file, the Image file may be backed up with a standard backup program such as tar on UNIX systems, Retrospect on the Macintosh, and Sytos+ or Novaback on DOS. If you will be sending your Image file this way, be sure and call the mastering house to make sure they can restore the particular backup format you are using.