

ISO9660 Simplified for DOS/Windows

by Philip J. Erdelsky

1. Introduction

We weren't sure about it a few years ago, but by now it should be clear to everyone that CD-ROM's are here to stay. Most PC's are equipped with CD-ROM readers, and most major PC software packages are being distributed on CD-ROM's.

Under DOS (and Windows, which uses the DOS file system) files are written to both hard and floppy disks with a so-called FAT (File Allocation Table) file system.

Files on a CD-ROM, however, are written to a different standard, called ISO9660. ISO9660 is rather complex and poorly written, and obviously contains a number of diplomatic compromises among advocates of DOS, UNIX, MVS and perhaps other operating systems.

The simplified version presented here includes only features that would normally be found on a CD-ROM to be used in a DOS system and which are supported by the Microsoft MS-DOS CD-ROM Extensions (MSCDEX). It is based on ISO9660, on certain documents regarding MSCDEX (version 2.10), and on the contents of some actual CD-ROM's.

Where a field has a specific value on a CD-ROM to be used with DOS, that value is given in this document. However, in some cases a brief description of values for use with other operating systems is given in square brackets.

ISO9660 makes provisions for sets of CD-ROM's, and apparently even permits a file system to span more than one CD-ROM. However, this feature is not supported by MSCDEX.

2. Files

The directory structure on a CD-ROM is almost exactly like that on a DOS floppy or hard disk. (It is presumed that the reader of this document is reasonably familiar with the DOS file system.) For this reason, DOS and Windows applications can read files from a CD-ROM just as they would from a floppy or hard disk.

There are only a few differences, which do not affect most applications:

1. The root directory contains the notorious "." and ".." entries, just like any other directory.
2. There is no limit, other than disk capacity, to the size of the root directory.
3. The depth of directory nesting is limited to eight levels, including the root. For example, if drive

E: contains a CD-ROM, a file such as E:\D2\D3\D4\D5\D6\D7\D8\FOO.TXT is permitted but E:\D2\D3\D4\D5\D6\D7\D8\D9\FOO.TXT is not.

4. If a CD-ROM is to be used by a DOS system, file names and extensions must be limited to eight and three characters, respectively, even though ISO9660 permits longer names and extensions.
5. ISO9660 permits only capital letters, digits and underscores in a file or directory name or extension, but DOS also permits a number of other punctuation marks.
6. ISO9660 permits a file to have an extension but not a name, but DOS does not.
7. DOS permits a directory to have an extension, but ISO9660 does not.
8. Directories on a CD-ROM are always sorted, as described below.

Of course, neither DOS, nor UNIX, nor any other operating system can WRITE files to a CD-ROM as it would to a floppy or hard disk, because a CD-ROM is not rewritable. Files must be written to the CD-ROM by a special program with special hardware.

3. Sectors

The information on a CD-ROM is divided into sectors, which are numbered consecutively, starting with zero. There are no gaps in the numbering.

Each sector contains 2048 8-bit bytes. (ISO9660 apparently permits other sector sizes, but the 2048-byte size seems to be universal.)

When a number of sectors are to be read from the CD-ROM, they should be read in order of increasing sector number, if possible, since that is the order in which they pass under the read head as the CD-ROM rotates. Most implementations arrange the information so sectors will be read in this order for typical file operations, although ISO9660 does not require this in all cases.

The order of bytes within a sector is considered to be the order in which they appear when read into memory; i.e., the "first" bytes are read into the lowest memory addresses. This is also the order used in this document; i.e., the "first" bytes in any list appear at the top of the list.

4. Character Sets

Names and extensions of files and directories, the volume name, and some other names are expressed in standard ASCII character codes (although ISO9660 does not use the name ASCII). According to ISO9660, only capital letters, digits, and underscores are permitted. However, DOS permits some other punctuation marks, which are sometimes found on CD-ROM's, in apparent defiance of ISO9660.

MSCDEX does offer support for the kanji (Japanese) character set. However, this document does not cover kanji.

5. Sorting Names or Extensions

Where ISO9660 requires file or directory names or extensions to be sorted, the usual ASCII collating sequence is used. That is, two different names or extensions are compared as follows:

1. ASCII blanks (32) are added to the right end of the shorter name or extension, if necessary, to make it as long as the longer name or extension.
2. The first (leftmost) position in which the names or extensions are not identical determines the order. The name or extension with the lower ASCII code in that position appears first in the sorted order.

6. Multiple-Byte Values

A 16-bit numeric value (usually called a word) may be represented on a CD-ROM in any of three ways:

Little Endian Word:

The value occupies two consecutive bytes, with the less significant byte first.

Big Endian Word:

The value occupies two consecutive bytes, with the more significant byte first.

Both Endian Word:

The value occupies FOUR consecutive bytes; the first and second bytes contain the value expressed as a little endian word, and the third and fourth bytes contain the same value expressed as a big endian word.

A 32-bit numeric value (usually called a double word) may be represented on a CD-ROM in any of three ways:

Little Endian Double Word:

The value occupies four consecutive bytes, with the least significant byte first and the other bytes in order of increasing significance.

Big Endian Double Word:

The value occupies four consecutive bytes, with the most significant first and the other bytes in order of decreasing significance.

Both Endian Double Word:

The value occupies EIGHT consecutive bytes; the first four bytes contain the value expressed as a little endian double word, and the last four bytes contain the same value expressed as a big endian double word.

7. The First Sixteen Sectors are Empty

The first sixteen sectors (sector numbers 0 to 15, inclusive) contain nothing but zeros. ISO9660 does not define the contents of these sectors, but for DOS they are apparently always written as zeros. They are apparently reserved for use by systems that can be booted from a CD-ROM.

8. The Volume Descriptors

Sector 16 and a few of the following sectors contain a series of volume descriptors. There are several kinds of volume descriptor, but only two are normally used with DOS. Each volume descriptor occupies exactly one sector.

The last volume descriptors in the series are one or more Volume Descriptor Set Terminators. The first seven bytes of a Volume Descriptor Set Terminator are 255, 67, 68, 48, 48, 49 and 1, respectively. The other 2041 bytes are zeros. (The middle bytes are the ASCII codes for the characters CD001.)

The only volume descriptor of real interest under DOS is the Primary Volume Descriptor. There must be at least one, and there is usually only one. However, some CD-ROM's have two or more identical Primary Volume Descriptors. The contents of a Primary Volume Descriptor are as follows:

length in bytes	contents
-----	-----
1	1
6	67, 68, 48, 48, 49 and 1, respectively (same as Volume Descriptor Set Terminator)
1	0
32	system identifier
32	volume identifier
8	zeros
8	total number of sectors, as a both endian double word
32	zeros
4	1, as a both endian word [volume set size]
4	1, as a both endian word [volume sequence number]
4	2048 (the sector size), as a both endian word
8	path table length in bytes, as a both endian double word
4	number of first sector in first little endian path table, as a little endian double word
4	number of first sector in second little endian path table, as a little endian double word, or zero if there is no second little endian path table
4	number of first sector in first big endian path table, as a big endian double word

4	number of first sector in second big endian path table, as a big endian double word, or zero if there is no second big endian path table
34	root directory record, as described below
128	volume set identifier
128	publisher identifier
128	data preparer identifier
128	application identifier
37	copyright file identifier
37	abstract file identifier
37	bibliographical file identifier
17	date and time of volume creation
17	date and time of most recent modification
17	date and time when volume expires
17	date and time when volume is effective
1	1
1	0
512	reserved for application use (usually zeros)
653	zeros

The first 11 characters of the volume identifier are returned as the volume identifier by standard DOS system calls and utilities.

Other identifiers are not used by DOS, and may be filled with ASCII blanks (32).

Each date and time field is of the following form:

length	
in bytes	contents

4	year, as four ASCII digits
2	month, as two ASCII digits, where 01=January, 02=February, etc.
2	day of month, as two ASCII digits, in the range from 01 to 31
2	hour, as two ASCII digits, in the range from 00 to 23
2	minute, as two ASCII digits, in the range from 00 to 59
2	second, as two ASCII digits, in the range from 00 to 59
2	hundredths of a second, as two ASCII digits, in the range from 00 to 99
1	offset from Greenwich Mean Time, in 15-minute

intervals,

as a twos complement signed number, positive for time zones east of Greenwich, and negative for time zones west of Greenwich

If the date and time are not specified, the first 16 bytes are all ASCII zeros (48), and the last byte is zero.

Other kinds of Volume Descriptors (which are normally ignored by DOS) have the following format:

length

in bytes contents

1 neither 1 nor 255

6 67, 68, 48, 48, 49 and 1, respectively (same as Volume Descriptor Set Terminator)

2041 other things

9. Path Tables

The path tables normally come right after the volume descriptors. However, ISO9660 merely requires that each path table begin in the sector specified by the Primary Volume Descriptor.

The path tables are actually redundant, since all of the information contained in them is also stored elsewhere on the CD-ROM. However, their use can make directory searches much faster.

There are two kinds of path table -- a little endian path table, in which multiple-byte values are stored in little endian order, and a big endian path table, in which multiple-byte values are stored in big endian order. The two kinds of path tables are identical in every other way.

A path table contains one record for each directory on the CD-ROM (including the root directory). The format of a record is as follows:

length

in bytes contents

1 N, the name length (or 1 for the root directory)

1 0 [number of sectors in extended attribute record]

4 number of the first sector in the directory, as a double word

2 number of record for parent directory (or 1 for the

root

directory), as a word; the first record is number 1, the second record is number 2, etc.

N name (or 0 for the root directory)

0 or 1 padding byte: if N is odd, this field contains a zero;

if

N is even, this field is omitted

According to ISO9660, a directory name consists of at least one and not more than 31 capital letters, digits and underscores. For DOS the upper limit is eight characters.

A path table occupies as many consecutive sectors as may be required to hold all its records. The first record always begins in the first byte of the first sector. Except for the single byte described above, no padding is used between records; hence the last record in a sector is usually continued in the next following sector. The unused part of the last sector is filled with zeros.

The records in a path table are arranged in a precisely specified order. For this purpose, each directory has an associated number called its level. The level of the root directory is 1. The level of each other directory is one greater than the level of its parent. As noted above, ISO9660 does not permit levels greater than 8.

The relative positions of any two records are determined as follows:

1. If the levels are different, the directory with the lower level appears first. In particular, this implies that the root directory is always represented by the first record in the table, because it is the only directory with level 1.
2. If the levels are identical, but the directories have different parents, then the directories are in the same relative positions as their parents.
3. Directories with the same level and the same parent are arranged in the order obtained by sorting on their names, as described in Section 5.

10. Directories

A directory consists of a series of directory records in one or more consecutive sectors. However, unlike path records, directory records may not straddle sector boundaries. There may be unused space at the end of each sector, which is filled with zeros.

Each directory record represents a file or directory. Its format is as follows:

```
length
in bytes  contents
-----
```

```

-----
1      R, the number of bytes in the record (which must be
even)
1      0 [number of sectors in extended attribute record]
8      number of the first sector of file data or directory
      (zero for an empty file), as a both endian double
word
8      number of bytes of file data or length of directory,
      excluding the extended attribute record,
      as a both endian double word
1      number of years since 1900
1      month, where 1=January, 2=February, etc.
1      day of month, in the range from 1 to 31
1      hour, in the range from 0 to 23
1      minute, in the range from 0 to 59
1      second, in the range from 0 to 59
      (for DOS this is always an even number)
1      offset from Greenwich Mean Time, in 15-minute
intervals,
      as a twos complement signed number, positive for time
      zones east of Greenwich, and negative for time zones
      west of Greenwich (DOS ignores this field)
1      flags, with bits as follows:
      bit      value
      -----
0 (LS) 0 for a normal file, 1 for a hidden file
1      0 for a file, 1 for a directory
2      0 [1 for an associated file]
3      0 [1 for record format specified]
4      0 [1 for permissions specified]
5      0
6      0
7 (MS) 0 [1 if not the final record for the file]
1      0 [file unit size for an interleaved file]
1      0 [interleave gap size for an interleaved file]
4      1, as a both endian word [volume sequence number]
1      N, the identifier length
N      identifier
P      padding byte: if N is even, P = 1 and this field
contains
      a zero; if N is odd, P = 0 and this field is omitted
R-33-N-P unspecified field for system use; must contain an even
      number of bytes

```

The length of a directory includes the unused space, if any, at the ends of sectors. Hence it is always an exact multiple of 2048 (the sector size). Since every directory, even a nominally empty one, contains at least two records, the length of a directory is never zero.

All fields in the first record (sometimes called the "." record) refer to the directory itself, except that the identifier length is 1, and the identifier is zero. The root directory record in the Primary Volume Descriptor also has this format.

All fields in the second record (sometimes called the ".." record) refer to the parent directory, except that the identifier length is 1, and the identifier is 1. The second record in the root directory refers to the root directory.

The identifier for a subdirectory is its name. The identifier for a file consists of the following fields, in the order given:

1. The name, consisting of the ASCII codes for at least one and not more than eight capital letters, digits and underscores.
2. If there is an extension, the ASCII code for a period (46). If there is no extension, this field is omitted.
3. The extension, consisting of the ASCII codes for not more than three capital letters, digits and underscores. If there is no extension, this field is omitted.
4. The ASCII code for a semicolon (59).
5. The ASCII code for 1 (49). [On other systems, this is the version number, consisting of the ASCII codes for a sequence of digits representing a number between 1 and 32767, inclusive.]

Some implementations for DOS omit (4) and (5), and some use punctuation marks other than underscores in file names and extensions.

Directory records other than the first two are sorted as follows:

1. Records are sorted by name, as described above.
2. Every series of records with the same name is sorted by extension, as described above. For this purpose, a record without an extension is sorted as though its extension consisted of ASCII blanks (32).
3. [On other systems, every series of records with the same name and extension is sorted in order of decreasing version number.]
4. [On other systems, two records with the same name, extension and version number are permitted, if the first record is an associated file.]

[ISO9660 permits names containing more than eight characters and extensions containing more than three characters, as long as both of them together contain no more than 30 characters.]

It is apparently permissible under ISO9660 to use two or more consecutive records to represent consecutive pieces of the same file. Bit 7 of the flags byte is set in every record except the last one. However, this technique seems pointless and is apparently not used. It is not supported by MSCDEX.

Interleaving is another technique that is apparently seldom used. It is not supported by MSCDEX (version 2.10).

11. Arrangement of Directory and Data Sectors

ISO9660 does not specify the order of directory or file sectors. It merely requires that the first sector of each directory or file be in the location specified by its directory record, and that the sectors for directories and non-interleaved files be consecutive.

However, most implementations arrange the directories so each directory follows its parent, and the data sectors for the files in each directory lie immediately after the directory and immediately before the next following directory. This appears to be an efficient arrangement for most applications.

Some implementations go one step further and order the directories in the same manner as the corresponding path table records.

12. Extended Attribute Records

Extended attribute records contain file and directory information used by operating systems other than DOS, such as permissions and logical record lengths.

A CD-ROM written for DOS normally does not contain any extended attribute records.

When reading a CD-ROM containing extended attribute records, early versions of MSCDEX simply returned incorrect results. Later versions learned to skip over extended attribute records.

Philip J. Erdelsky

San Diego, California USA

pje@acm.org

<http://www.alumni.caltech.edu/~pje/>