DEVELOPMENT STANDARD

QIC-40-MC
Revision M
02 Sep 92

FLEXIBLE-DISK-CONTROLLER-COMPATIBLE RECORDING FORMAT FOR INFORMATION INTERCHANGE

0.250 in. (6.35 mm) Tape
20 Tracks
Data Density: 10,000 BPI (394 BPMM)
MFM Encoded

Formatted Capacity: 40 Mbytes
(With DC2000 or Equivalent Minicartridge)

**Quarter-Inch
Cartridge
Drive Standards, Inc.**

311 East Carrillo Street
Santa Barbara, California 93101
Telephone (805) 963-3853
Fax (805) 962-1541
www.qic.org

(See important notices on the following page)

Important Notices

TABLE OF CONTENTS

TABLE OF CONTENTS
(continued)

TABLE OF CONTENTS
(continued)

LIST OF FIGURES

1.0 SCOPE AND INTRODUCTION

1.1 SCOPE

This standard describes a 40, 60, or 217 Megabyte, 20 Track, 10,000 bpi (394 bpmm) MFM Encoded, Flexible Disk Controller Compatible Recording Format using 1/4" (6.35 mm) Mini Data Cartridge Tape. The standard uses a Reed Solomon error correction code to achieve a corrected error rate of

$$10^{-14}$$

or less given a raw event error rate of 10^{-8} as provided by requirements in section 6.1. This format and recording standard for the 0.250 in (6.35 mm) wide magnetic tape cartridge is to be used for information interchange among information processing systems, communications systems and associated equipment. Compliance with the standard for unrecorded magnetic tape cartridge (ref. ANSI X3B5/85-135) is a requirement for information interchange.

1.2 INTRODUCTION

This standard defines the requirements and supporting test methods necessary to ensure interchange at acceptable performance levels. It is distinct from a specification in that it delineates a minimum of restrictions consistent with compatibility in interchange transactions.

The performance levels contained in this standard represent the minimum acceptable levels of performance for interchange purposes. They therefore represent the performance levels which the interchanged items should meet or surpass during their useful life and thus define end-of-life criteria for interchange purposes. The performance levels in this standard are not intended to be employed as substitutes for purchase specifications.

Wherever feasible, quantitative performance levels which must be met or exceeded in order to comply with this standard are given. In all cases, including those in which quantitative limits for requirements falling within the scope of this standard are not stated but are left to agreement between interchange parties, standard test methods and measurement procedures shall be used to determine such quantities.
U. S. Engineering units are the original dimensions in this standard. Conversions of tolerance dimensions from customary U.S. engineering units (similar to British Imperial Units) to SI units have been done in this standard according to ANSI Z210.1-1976 and ISO 370 Method A, except as noted. Method A should be used for economy unless a requirement for absolute assurance of a fit justifies use of Method B. In the national standards of ISO member nations, additional rounding may be done to produce "preferred" values. These values should lie within or close to the original tolerance ranges.

Except as indicated in the second preceding paragraph, interchange parties complying with the applicable standards should be able to achieve compatibility without need for additional exchange of technical information.

2.0 DEFINITIONS

azimuth - the angular deviation, in minutes of arc, of the mean flux transition line from the line normal to the cartridge reference plane.

BOT - beginning of tape marker indicating beginning of tape.

bit - a single digit in the binary number system.

bit cell - a length of magnetic recording tape within which a flux transition written at the center signifies a "one" bit and the absence signifies a "zero" bit.

block - a group of 1024 consecutive bytes transferred as a unit.

byte - a group of 8 binary bits operated on as a unit.

cartridge - a 2.406 x 3.188 in (61.11 x 80.98 mm) enclosure containing 0.250 in (6.35 mm) wide magnetic tape wound on two coplanar hubs and driven by an internal belt which is coupled by an internal belt capstan to the external drive (ref. ANSI X3B5/85- 135).

cyclical redundancy check - a two byte code derived from information contained in a data block or ID block used for read check.

cyclical redundancy check failure - a data error in a sector not detected by the cyclical redundancy check.

data segment - a segment containing directory and/or file information

density - the maximum allowable flux transitions per unit length for a specific recording standard.

early warning - marker on tape indicating the end of the permissible recording area for even numbered tracks and indicating the start of the permissible recording area for odd numbered tracks.

ECC - error correction code

EOT - end of tape marker indicating the end of tape.

erase - to remove all magnetically recorded information from the tape.

file - a logical unit of information

file set - a group of files and their directories

2.0 DEFINITIONS (continued)

flux transition - a point on the magnetic tape which exhibits maximum free space flux density normal to the tape surface.

flux transition spacing - the distance on the magnetic tape between flux reversals.

format - the operation of preparing a cartridge for use by formatting the tape

load point - marker on tape indicating the start of the permissible recording area for even numbered tracks and indicating the end of the permissible recording area for odd numbered tracks.

logical format - a directory and file structure suitable for information storage and retrieval

MFM encoding - a method of encoding data for magnetic recording in which a "one" is represented by a flux transition in the bit cell center and a "zero" by no flux transition. A clock flux transition is written at the end of a bit cell containing a "zero" if the "zero" was preceded by a "zero".

magnetic tape - an oxide coated mylar base tape capable of accepting and retaining magnetically recorded information.

postamble - guard information recorded after the data block.

preamble - synchronization information recorded before the data block or ID block.

RS - refers to Reed Solomon, a method of error correction

recorded block - a group of consecutive bits comprising preamble, data address mark, data block, CRC and postamble.

re-direction - restoration of a file or sub-directory to a directory other than from where it originated

reference tape cartridge - a magnetic tape cartridge selected for a specific property to be used as a reference.

sector - same as block

segment - a group of 32 blocks operated on as a unit

2.0 DEFINITIONS (continued)

segment number - an integer from 0 to the total number of formatted segments minus one, indicating a specific physical track and segment position

sync - same as preamble

streaming - a method of recording on magnetic tape where the tape is continuously moving and data blocks are continuously recorded.

track - a recording strip parallel to the edge of the magnetic tape containing recorded information.

underrun - a condition developed when host transmits or receives data at a rate less than that required by the device for streaming operation.

volume table segment - a segment containing the names and storage positions of all file sets placed upon the cartridge.

3.0 RECORDING

3.1 METHOD

The method of recording shall be Modified Frequency Modulation (MFM) where a "one" (1) is represented by a flux transition in the middle of the bit cell and a "zero" (0) is represented by the absence of flux transition. A clock flux transition is written at the end of a bit cell containing a "zero" followed by a "zero".

3.2 CODE

The code for recording shall consist of an eight (8) bit byte recorded in a bit serial manner, where each bit cell corresponds to a binary digit. The most significant bit of the byte shall be recorded first.

3.3 RECORDING MEDIA

Recorded data for interchange shall be recorded on a 1/4 inch (6.35 mm) Mini Data Cartridge, ANSI X3B5/85-135 (205 foot length), or the related ANSI specifications for the same tape cartridge with 307.5 or 1,100 feet of recording area).

3.4 TAPE SPEED

The linear speed of the media across the magnetic recording head in the intended application is 25.0 inches (635 mm) per second at the 250 Kb/s transfer rate or 50.0 inches (1270 mm) per second at the 500 Kb/s transfer rate. Other speeds and compatible transfer rates are possible. Long term speed variation shall be a maximum of

cartridge	± 1.3%
drive	± 1.7%
total product	± 3.0%

Short term speed variation, variations in speed from the long term speed for frequencies of 0 to 1000 Hz, shall be a maximum of

cartridge	± 4.0%
drive	± 2.0%
total product	± 6.1%

3.5 NOMINAL DENSITY

The maximum nominal recording density shall be 10,000 bits per inch (bpi) or 394 bits per millimeter (bpmm).

3.6 NOMINAL BIT-CELL LENGTH

The nominal bit-cell length shall be 100 microinches (2.54 micrometers).

3.7 AVERAGE BIT-CELL LENGTH

The average bit-cell length is the sum of the distances over N bit-cells divided by N.

Any continuously recorded MFM pattern may be used to measure the average bit-cell length.

3.8 LONG TERM AVERAGE BIT-CELL LENGTH

The long term average bit-cell length is the average bit-cell length taken over a minimum of 1,000,000 bit-cells.

The long term average bit-cell length shall be within ± 3.0 % of the nominal bit-cell length of 100 microinches (2.54 micrometers).

3.9 MEDIUM TERM AVERAGE BIT-CELL LENGTH

The medium term average bit-cell length is the average bit-cell length taken over a minimum of 8,000 and a maximum of 8,800 bits.

The medium term average bit-cell length shall be within ± 6.1% of the long term average bit-cell length.

3.10 SHORT TERM AVERAGE BIT-CELL LENGTH

The short term average bit-cell length is the average bit-cell length taken over 16 bit-cells.

The short term average bit-cell length shall be within ± 2.0% of the medium term average bit-cell length.

3.11 INSTANTANEOUS FLUX TRANSITION SPACING

The instantaneous spacing between flux transitions is influenced by the reading and writing processes, the pattern recorded (pulse-crowding effect) and other factors.

Instantaneous spacings between flux transitions shall satisfy the following conditions.

In a sequence of flux transitions defined by the repetitive flux transition pattern 11101110, the center flux transition of each group of "ones" is called a reference flux transition. The maximum displacement of flux transitions on either side of the reference flux transitions shall not exceed ± 12.5% of the bit cell length averaged over the four bit cells between the reference flux transitions indicated in figure 1 below.

```
           <----- 4 bit cells ----->
    _____x      ___________      x_____             ___
___|     |_____|           |_____|     |___________|
      1     1     1     0     1     1     1  0    1
     <->         <->         <->         <->     <->
  ± 12.5%     ± 12.5%     ± 12.5%     ± 12.5%  ± 12.5%
```

note - x indicates reference flux transition

Figure 1. Flux Transition Spacing

3.12 SIGNAL AMPLITUDE REFERENCE TAPE CARTRIDGE

A signal amplitude reference tape cartridge is a magnetic tape cartridge selected as a standard for signal amplitude when recorded at 10,000 ftpi (394 ftpmm).

3.13 MEASUREMENT OF SIGNAL AMPLITUDE

The signal amplitude shall be measured at a point in the read channel where the signal is proportional to the rate of change of voltage from the read head.

3.14 AVERAGE STANDARD REFERENCE AMPLITUDE

The average standard reference amplitude is the peak-to- peak output signal read from the Signal Amplitude Reference Tape Cartridge averaged over a minimum of 10,000 bit-cells.

3.15 AVERAGE SIGNAL AMPLITUDE

The average peak-to-peak signal amplitude of a tape cartridge recorded at 10,000 bpi shall deviate no more than ± 25% from the average standard reference amplitude. Averaging shall be done over a minimum of 10,000 bit-cells.

3.16 SIGNAL DECAY

Signal decay is a measurement of loss in signal amplitude of a recorded tape due to cycling a tape in contact. The tape under test is cycled from BOT to EOT to BOT 55 times. The loss in amplitude from the 5th pass to the 55th pass shall not exceed 15%.

3.17 OVERWRITE AND ERASURE

The overwritten or erased area shall not contain any component of previously recorded information whose amplitude exceeds -30 dB relative to the amplitude of the newly written data (reference section 3.15).

3.18 AZIMUTH

The angular deviation of the mean bit-cell transition line from a line normal to the magnetic tape cartridge reference base shall be less than 10 minutes of arc.

4.0 TRACKS

There shall be 20 parallel tracks. Even tracks shall be recorded in the forward tape direction. Odd tracks shall be recorded in the reverse tape direction.

4.1 TRACK LOCATIONS

4.1.1Track Center Lines

The track center line locations are shown in figure 2 below. Even tracks are relative to the center line of a forward reference burst. Odd tracks are relative to the center line of a reverse reference burst offset from the forward reference burst. Track distances above their reference burst are considered positive. Track distances below their reference burst are considered negative.

	track #	CL from CL of Reference	
		inches	millimeters
	18	.108 ± .0014	2.7432 ± .0356
	16	.096 ± .0014	2.4384 ± .0356
	14	.084 ± .0014	2.1336 ± .0356
	12	.072 ± .0014	1.8288 ± .0356
	10	.060 ± .0014	1.5240 ± .0356
	08	.048 ± .0014	1.2192 ± .0356
	06	.036 ± .0014	0.9144 ± .0356
	04	.024 ± .0014	0.6096 ± .0356
	02	.012 ± .0014	0.3048 ± .0356
Fwd Ref Burst ---->	00	.000 ± .0014	0.0000 ± .0356
	01	.012 ± .0014	-0.3048 ± .0356
Rev Ref Burst <----	03	-.000 ± .0014	-0.0000 ± .0356
	05	-.012 ± .0014	-0.3048 + .0356
	07	-.024 ± .0014	-0.6096 ± .0356
	09	-.036 ± .0014	-0.9144 ± .0356
	11	-.048 ± .0014	-1.2192 ± .0356
	13	-.060 ± .0014	-1.5240 ± .0356
	15	-.072 ± .0014	-1.8288 ± .0356
	17	-.084 ± .0014	-2.1336 ± .0356
	19	-.096 ± .0014	-2.4384 ± .0356

Figure 2. Track Locations

4.1.2Forward Reference Burst Location

The forward reference burst center line shall be located 0.0070 ± .0020 inch (0.1778 ± .0508 mm) above the tape center line and written at the BOT end of tape in the forward direction.

4.1.3Reverse Reference Burst Location

The reverse reference burst center line shall be located 0.0250 + .0010 inch (0.6350 ±.0254 mm) below the forward reference burst center line and written at the BOT end of tape in the reverse direction.

4.2 TRACK DIMENSIONS

4.2.1Nominal Track Spacing

The nominal track spacing shall be 0.012 inch (.3048 mm) between even tracks and between odd tracks. The nominal track spacing between track 00 and track 01 shall be 0.013 inch (.3302 mm).

4.2.2Recording Head

The recording head shall be in accordance with the QIC-106 DEVELOPMENT STANDARD FOR A SINGLE CHANNEL MAGNETIC HEAD.

5.0 TRACK FORMAT

The tape format consists of three (3) areas: reference burst, beginning gap and recording area.

5.1 REFERENCE BURST

Reference bursts shall be recorded at 5K frpi between the innermost BOT hole and not more than one inch (25.4mm) prior to the LP hole. The remaining area between the BOT hole and the LP hole shall be completely erased, for all other tracks including between tracks, at format time.

The following suggested safe detect ranges are to be interpreted as being measured physically on the tape. This can be extrapolated by a tape drive manufacturer to compensate for the distance between a tape drive's hole detect sensor and that tape drive's head.

The safe detect range for the track zero reference burst as read in the forward direction should begin 2.20 inches inside or after the inside hole of the inside BOT pair as measured physically on the tape.

The safe detect range for the track zero reference burst as read in the forward direction should end 2.00 inches outside or before the load point hole as measured physically on the tape.

The safe detect range for the track three reference burst as read in the reverse direction should begin 2.00 inches outside or after the load point hole as measured physically on the tape.

The safe detect range for the track three reference burst as read in the reverse direction should end 2.00 inches inside or before the inside hole of the inside BOT pair as measured physically on the tape. The track 00 reference burst may start at any point outside or before the BOT hole pair, but must start before the 2.20 inch safe reference burst detect range.

5.2 BEGINNING GAP

The beginning gap shall be a minimum of 0.5 inch (12.7 mm) and a maximum of 2 inches (50.8 mm) of erased tape between LP and the recording area for even tape tracks. The beginning gap shall be a minimum of 1.5 inches (38.1 mm) and a maximum of 2 inches (50.8 mm) between EW and the recording area for odd tape tracks. The area between EOT and EW on odd track locations shall also be erased.

5.3 RECORDING AREA

The recording area of a tape track consists of 68 segments separated from each other by 1.0 inches (25.4 mm) $\pm$ 9.3% of erased tape. Each segment contains 29 data sectors and 3 ECC sectors for a total of 32 sectors. Each tape track contains 2,176 sectors (205 feet), 3,264 sectors (307.5 feet), or 11,680 sectors (1,100 feet).

5.3.1Identification of Sectors

Each sector on the tape shall be uniquely identified by 3 sector identification components: flexible disk side (FSD), flexible disk track (FTK), and flexible disk sector (FSC).

The first sector on tape track 00 shall have the physical identification vector (FSD,FTK,FSC) = (0,0,1). The physical identification vector shall be related to the logical sector number (LSN) by the equations:

LSN = 21,760 x FSD + 128 x FTK + (FSC - 1)
(205 feet)

0 <= LSN <= 43,519
0 <= FSD <= 1
0 <= FTK <= 169
1 <= FSC <= 128

or

LSN = 32,640 x FSD + 128 x FTK (FSC-1)
(307.5 feet)

0 <= LSN <= 65,279
0 <= FSD <= 1
0 <= FTK <= 254
1 <= FSC <= 128

or

LSN = 32,512 x FSD + 128 x FTK (FSC-1)
(1,100 feet)

0 <= LSN <= 233,600
0 <= FSD <= 7
0 <= FTK <= 253
1 <= FSC <= 128

The physical identification vector shall be related to the logical segment number (SEG) by the equation

```
SEG = 680 x FSD + 4 x FTK + int ((FSC-1)/32)
where   0 <= SEG <= 1,359 (205 feet)

--OR--

SEG = 1,020 x FSD + 4 x FTK + int ((FSC-1)/32)
where   0 <= SEG <= 2,039 (307.5 feet)

--OR--

SEG = 1,016 x FSD + 4 x FTK + int ((FSC-1)/32)
where   0 <= SEG <= 7,300 (1,100 feet)
```

5.3.2Segment Formatting

Each segment is formatted as shown below.

# bytes	hex value	description
SEGMENT HEADER		
80	4E	gap 4A
12	00	sync
3	C2 (note 1)	index addr mark
1	FC (note 2)	index addr mark
50	4E	gap 1
146		total for index
repeat below 32 times		
TRACK SECTOR ID		
12	00	sync
3	A1 (note 3)	sector id addr mark
1	FE (note 4)	sector id addr mark
1	FTK	floppy track number
1	FSD	floppy side number
1	FSC	floppy sector number
1	03	1024 bytes/sector
2	crc(note 5)	CRC for trk/sector id
22	4E	gap 2
DATA BLOCK		
12	00	sync
3	A1 (note 3)	data address mark
1	FB (note 6)	data address mark
1024	data	data block
2	crc (note 7)	CRC for data block
SPEED TOLERANCE ± 9.3%		
218	4E	gap 3

DROP OUT GUARD .012 in (.305 mm)

15	4E	gap 3
1319 end of repeat		total for sector

TIMER TOLERANCE ± 0.3%

274	4E	gap 4B (until index)
42,628		nominal total for floppy track segment

note 1 - each hex "C2" in the index address mark shall have a missing clock between bits 3 and 4

```
hex "C2"       1   1   0   0   0   0   1   0

MFM            |   |     |   |   |     |

MFM with       |   |     |       |     |
missing clock
```

note 2 - hex "FC" completes the index address mark

note 3 - each hex "A1" in the sector address mark, data address mark or deleted data address mark shall have a missing clock between bits 4 and 5

```
hex "A1"       1   0   1   0   0   0   0   1

MFM            |       |     |   |   |     |

MFM with       |       |     |       |     |
missing clock
```

note 4 - hex "FE" completes the sector address mark

note 5 - CRC for the preceding eight (8) bytes generated by the polynominal

$$X^{16} + X^{12} + X^{5} + 1$$

with the accumulating register set to all ones prior to transfer

note 6 - hex "FB" completes the data address mark or a hex "F8" completes the deleted data address mark

note 7 - CRC for the preceding 1028 bytes generated by the polynominal and accumulating register condition defined in note 5

5.4 FORMAT CONTROL

The track format defines recording areas for both even and odd tracks. A precision timer (xtal oscillator based) is required to define recording areas other than those defined by tape holes. Distances shown are converted to time by dividing by 25 or 50 ips, depending on the speed of the drive.

The erased gap to EW hole distance for even tracks or the erased gap to LP hole distance for odd tracks provides 72 inches (205 feet), 108 inches (307.5 feet), or 385 inches (1,100 feet) to tolerance the long term speed error of 3.0%.

```
 2,388" x (1.03) =  2,460" min.tape length (205 feet)
 3,583" x (1.03) =  3,690" min.tape length (307.5 feet)
12,816" x (1.03) = 13,200" min.tape length (1,100 feet)
```

5.4.1Even Track Format

(all dimensions nominal unless otherwise stated)

	in	mm
beginning tape to BOT pair	12	304.8 minimum
BOT pair to BOT pair	12	304.8
BOT pair to BOT pair	12	304.8
BOT hole pair to LP hole (note 1)	30	762.0(205 foot tape)
	36	914.4(307.5 foot tape)
	54	1,371.6(1,100 foot tape)
beginning gap	0.5 to 2	12.7 to 50.8
floppy track segment	34.1	866.1

repeat below 67 times (205 feet), 101 times (307.5 feet), or 364 times (1,100 feet).

erased gap	1.0	25.4
floppy track segment	34.1	866.1

end of repeat

erased gap to EW hole	72	1,829
EW hole to EOT hole	30	762.0 (205 foot tape)
	36	914.4 (307.5 foot tape)
	54	1,371.6 (1,100 foot tape)
EOT hole to EOT hole	12	304.8
EOT hole to EOT hole	12	304.8
EOT hole to end of tape	12	304.8 minimum

note 1 - erased tape except for reference burst on track 00

5.4.2Odd Track Format

(all dimensions nominal unless otherwise stated)

	in	mm
end of tape to EOT hole	12	304.8 minimum
EOT hole to EOT hole	12	304.8
EOT hole to EOT hole	12	304.8
EOT hole to EW hole	30	762.0 (205 foot tape)
(note 1)	36	914.4 (307.5 foot tape)
	54	1,371.6 (1,100 foot tape)
beginning gap	1.5 to 2	38.1 to 50.8
floppy track segment	34.1	866.1

repeat below 67 times (205 feet), 101 times (307.5 feet), or 364 times (1,100 feet).

erased gap	1.0	25.4
floppy track segment	34.1	866.1

end of repeat

erased gap to LP hole	72	1,829
LP hole to BOT pair	30	762.0 (205 foot tape)
	36	914.4 (307.5 foot tape)
	54	1,371.6 (1,100 foot tape)
BOT pair to BOT pair	12	304.8
BOT pair to BOT pair	12	304.8
BOT to beginning tape	12	304.8 minimum

note 1 erased tape except for reference burst on track 00

6.0 ERROR CONTROL

Hard errors are caused by media defects in the recording area. Two strategies shall be used to defeat hard errors:
(1) elimination of sectors with defects before use and
(2) correction of sectors with hard errors after use.

6.1 ELIMINATION OF SECTORS WITH DEFECTS

Sectors containing media defects shall be excluded from use via the bad sector map (section 7.0).

6.1.1Format Read Check

The cartridge format operation shall include a read check which shall identify all sectors containing media defects of .003 inches (.0762 mm) or greater. Sectors so identified shall be excluded from use via the bad sector map. Based on the recording head of section 4.2.2, a media defect of .003 inches (.0762 mm) implies a 50% threshold referenced to the signal amplitude reference cartridge of section 3.12.

6.1.2Data Read Check

The cartridge data recording operation shall in clude a read check which shall identify all sectors containing media defects of .006 inches (.1524 mm) or greater. Sectors so identified shall be excluded from use via the bad sector map. Based on the recording head of section 4.2.2, a media defect of .006 inches (.1524 mm) implies a zero (0) threshold referenced to the signal amplitude reference cartridge of section 3.12.

6.2 ERROR CORRECTION

The cartridge data recording operation shall include three (3) sectors of ECC information in each segment which shall be used during the data reading operation to reconstruct sectors in error. The error correction algorithm shall correct errors in each data segment as follows:

(1) correct 3 or less sectors with CRC errors
(2) correct 1 sector with CRC error and 1 sector with CRC failure
(3) correct 1 sector with CRC failure

The error correction algorithm can also detect but not necessarily correct errors in each data segment as follows:

(1) detect 2 sectors with CRC failures
(2) detect 2 sectors with CRC errors and 1 sector with CRC failure

6.2.1ECC Format

The bytes in a segment are considered to be arranged in a 32 x 1024 matrix, and the parity bytes shall be chosen so that each column of the matrix is an independent Reed-Solomon codeword of redundancy three, with 8-bit characters, as shown in figure 3. Data shall be written on the tape row by row, starting with row 0, and within each row (sector) the bytes shall be written starting with column 0.

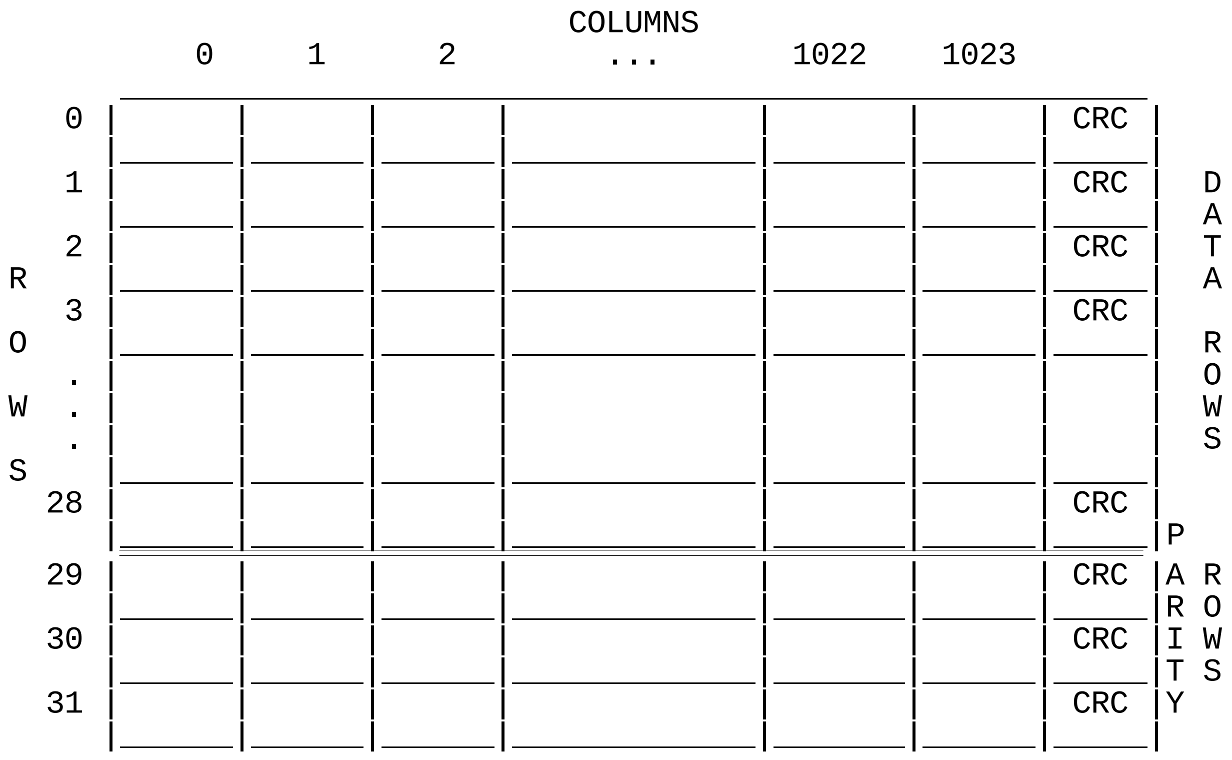


Figure 3. ECC Format

See section 6.2.5 for treatment of sectors excluded from use via the bad sector map.

6.2.2 Field Presentation

GF(256) is a field consisting of 256 elements. Each field element a has the form:

$$a = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

where each a_i is either 0 or 1. A field element a shall be represented by a byte as shown in figure 4.

MSB							LSB
a7	a6	a5	a4	a3	a2	a1	a0

Figure 4. Bit Numbering Convention

Field math operations (addition, multiplication, division) are defined to be polynomial math modulo an irreducible binary polynomial of degree eight, f(x), where binary addition is the logical exclusive or operation and binary multiplication is the AND operation. The irreducible polynomial used to generate the field GF(256) shall be:

$$f(x) = x^8 + x^7 + x^2 + x + 1$$

6.2.3Code Generator Polynominal

The generator polynomial for the Reed-Solomon code shall be:

$$g(x) = x^3 + r^{105} x^2 + r^{105} x + 1$$

with r a root of f(x). The hex representation of r^{105} is C0 (decimal 192).

Each column of data contains data bytes, as in figure 3, from d_0 to d_{28}. The column's parity bytes d_{29}, d_{30}, and d_{31} shall be chosen such that the polynomial

$$d(x) = \sum_{i=0}^{N} d_i x^i,$$

is divisible by g(x), using polynomial division over GF(256), where N = 31 less number of "bad blocks" in the segment.

6.2.4Test Codewords

The polynomials defined in the preceding sections shall produce the following test codewords.

	Row #							
	0 :	00	00	00	00	00	00	01
	1 :	00	00	00	00	00	00	02
	2 :	00	00	00	00	00	01	03
	3 :	00	00	00	00	00	C0	04
	4 :	00	00	00	00	00	C0	05
	5 :	00	00	00	00	00	01	06
	6 :	00	00	00	00	00	01	07
	7 :	00	00	00	00	00	00	08
Data	8 :	00	00	00	00	00	67	09
	9 :	00	00	00	00	00	A6	0A
	10 :	00	00	00	00	00	C0	0B
	11 :	00	00	00	00	00	01	0C
	12 :	00	00	00	00	00	00	0D
	13 :	00	00	00	00	00	00	0E
	14 :	00	00	00	00	00	FF	0F
	15 :	00	00	00	00	00	99	10
	16 :	00	00	00	00	00	67	11
	17 :	00	00	00	00	00	01	12
	18 :	00	00	00	00	00	00	13
	19 :	00	00	00	00	00	00	14
	20 :	00	00	00	00	00	00	15
	21 :	00	00	00	00	00	A3	16
	22 :	00	00	00	00	00	5D	17
	23 :	00	00	00	00	00	FF	18
	24 :	00	00	00	00	01	01	19
	25 :	00	00	00	01	00	00	1A
	26 :	00	00	01	00	00	00	1B
	27 :	00	01	00	00	00	00	1C
	28 :	01	00	00	00	00	00	1D
Parity	29 :	C0	67	FF	A3	AD	AD	5D
Parity	30 :	C0	A6	99	5D	0F	0F	FF
Parity	31 :	01	CO	67	FF	A3	A3	A3

Figure 5. Test Codewords

6.2.5Excluded Sectors

Sectors containing media defects are excluded from use via the bad sector map. The last three nonexcluded sectors of a segment shall contain the parity bytes as shown in figure 6 below. The preceding nonexcluded sectors of the segment shall contain data. Excluded sectors in the data portion of the segment shall be skipped over as shown in figure 6 below.

```
                              ROW #

                    1 1 1 1 1 1 1 1 1 1 2 2 2 2 2 2 2 2 2 2 3 3
0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
-----------------------------------------------------------------
D D D D D D D D D D B D B D D D D D D D D D D D D D P P P P P P *
D D D D D D D D D D D D D D D D D D D D D D D D D D P P P P    **

D D D D D D D D D D D D D D D D D D D D D D D D D D P B P B P B *
D D D D D D D D D D D D D D D D D D D D D D D D D D P P P      **

D B D D D D D D D D D D D D D D B D D D D P B B B P B B B P B B *
D D D D D D D D D D D D D D D D D D D P P P                    **
```

D = good sector, filled with data
B = bad sector, skipped over
P = good sector, filled with parity
* = segment as it appears on tape
** = segment as it appears in memory

Figure 6. Excluded Sectors

7.0 HEADER SEGMENT

The header segment shall be created in the format operation following the read check. The first defect free segment shall be the header segment. The second defect free segment shall be a duplicate of the header segment. Sectors in segments preceding the first header segment and between the 1st and 2nd header segments shall be recorded with deleted data marks (see note 6 of section 5.3.2). Sectors of the header segment shall be used as follows:

SECTOR(S)	USAGE
0 - 1	Format parameter record
2 - 28	Bad sector map
29 - 31	ECC

7.1 FORMAT PARAMETER RECORD

A cartridge history shall be maintained in the format parameter record. Bytes of word and double word fields shall be recorded low to high order.

SECTOR 0 OFFSET(S)	USAGE
0 - 3	Header segment signature string. Used to verify that the header segment has been correctly found. Must be set to hex '55 AA 55 AA' by format.
4	Format code. Identifies logical structure of tape. Hex '02' designates tapes of 205 or 307.5 feet, and hex '03' designates tapes of 1,100 feet.
5	Unused, set to zero.
6 - 7	Word: segment number of header segment.
8 - 9	Word: segment number of duplicate header segment.
10 - 11	Word: first logical area data segment.
12 - 13	Word: last logical area data segment.
14 - 17	Double word: date and time of most recent format. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970-2097) 24- 0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))

18 - 21	Double word: date and time of most recent write or format to cartridge. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970-2097) 24- 0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))
22 - 23	Unused, set to zero.
24 - 25	Word: Tape segments per tape track. Defined Values: 0 - QIC-40 Rev E and earlier. Interpret as 68. 68 - QIC-40 Rev F and later, 205 foot tape. 102 - QIC-40 Rev F and later, 307.5 foot tape. 365 - QIC-40 Rev F and later, 1,100 foot tape
26	Byte: Tape tracks per cartridge. Defined Values: 0 - QIC-40 Rev E and earlier. Interpreted as 20. 20 - QIC-40 Rev F and later.
27	Byte: Maximum floppy side. Defined values: 0 - QIC-40 Rev E and earlier, 205 and 307.5 foot tapes Interpret as 1. 1 - QIC-40 Rev F and later, 205 and 307.5 foot tapes. 7 - QIC-40 Rev F and later, 1,100 foot tape.
28	Byte: Maximum floppy track. Defined values: 0 - QIC-40 Rev E and earlier. Interpret as 169. 169 - QIC-40 Rev F and later, 205 foot tape. 254 - QIC-40 Rev F and later, 307.5 foot tape. 253 - QIC-40 Rev F and later, 1,100 foot tape.
29	Byte: Maximum floppy sector. Defined values: 0 - QIC-40 Rev E and earlier. Interpret as 128. 128 - QIC-40 Rev F and later.

30 - 73	Tape name. Any ASCII string, left justified and space filled.
74 - 77	Double word: date and time that the tape name was written. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970=2097) 24-0: Month, day, hour, minute and second. Let MO=month (0-11). DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Store here is : SC+60* (HR+24* (DY=31*MO)))
78 - 79	Word: beginning segment number of COMPRESSION MAP SEGMENTS which contains uncompressed byte offsets within a volume on a segment by segment basis.
80 - 127	Unused, set to zero.
128	Reformat error flag. = FF if any of the remaining fields were lost due to tape error during re-format, else = 00.
129	Unused, set to zero.
130 - 133	Double word: Number of segments written, formatted or verified throughout life of tape.
134 - 137	Unused, set to zero.
138 - 141	Double word: date and time of initial tape format. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970-2097) 24- 0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))
142 - 143	Word: format count. Number of times tape has been formatted.
144 - 145	Word: failed sector count. Number of entries in failed sector log.
146 - 189	Original manufacturer name/code. For pre-formatted tapes, this field holds a manufacturer identification string. All zero if tape not pre-formatted.
190 - 233	Original manufacturer lot code. For pre-formatted tapes, this field holds tape lot numbering information. All zero if tape not pre-formatted.
234 - 255	Unused, set to zero.

256 - 1023	Failed sector log. A two word entry is appended here whenever a sector fails to verify during normal use. The first word holds the segment number of the failed sector, the second holds the date of failure. The date is encoded as follows: Bits 15-9: Year-1970 (0-127 = 1970-2097) 8-5: Month (1-12) 4-0: Day (1-31)
0 - 1023	Continuation of failed sector log.

7.2 BAD SECTOR MAP

Tapes with format code 2 (205 and 307.5 foot tapes).

The bad sector map shall be a table of 4-byte entries corresponding to every physical segment on the tape, covering the bad sector map and any deleted segments in front of it. Each 4 byte entry is viewed as an integer, stored low to high order. Sector 0 of a segment is mapped by bit zero (the low order bit) of this integer, sector 1 by bit one (the next to low order bit), and so forth. A "1" bit indicates a bad sector. Bad sectors shall not be written to or read from, or included in ECC calculations.

Pictorially, the bits of a bad sector entry can be represented as follows:

```
               Bit/Sector correspondence
               -------------------------
              Bit   7  6  5  4  3  2  1  0

         Byte zero  07-06-05-04-03-02-01-00
         Byte one   15-14-13-12-11-10-09-08
         Byte two   23-22-21-20-19-18-17-16
         Byte three 31-30-29-28-27-26-25-24
```

Tapes with format code 3 (1,100 foot tapes).

When reading the bad sector map for tapes with a format code of 3 the following bad sector map shall be used. The bad sector map shall be a list of 3-byte structures that defines each bad sector on the tape. This list is organized in ascending order so that as each bad sector is encountered during the format verify process, it is added to the next available structure location. This format allows a total of 9,216 sectors to be marked bad and can address up to 16,777,215 sectors.

Each 3-byte structure is viewed as an array of bytes from least significant byte to most significant byte, which can be converted to a double word representation of the logical sector number. Sector numbering starts with one (1) so that a zero entry can be interpreted as the end of the list. For example, the following byte string would indicate 4 bad sectors with logical sector numbers of 0, 45, 999, and 4321:

```
0x01,0x00,0x00,0x2e,0x00,0x00,0xe8,0x03,0x00,0xe2,0x10,0x00,
0x00,0x00,0x00,0x00
```

8.0 VOLUME TABLE SEGMENT

The volume table segment shall be created in the format operation following creation of the header segment as an all "zeroes" segment. The volume table shall be stored in the first segment of the logical area.

The volume table will contain an entry for each file set stored on the cartridge. The entry capacity of the table is determined by the amount of usable sectors within the segment it is stored upon. Each entry within the table shall be 128 bytes long.

The first 4 bytes of a used entry shall hold the uppercase ASCII string "VTBL" as a signature. Following all used entries, the end of the volume table shall be indicated by an entry with no such signature. If the first table slot contains no signature, the cartridge is empty. If the last slot contains a signature, the cartridge is full.

It shall be required that each successive volume table entry allocate a segment range beyond any currently in use. Thus the ending segment number of the final table entry acts as a starting point for further allocation.

The volume table entry for supplementary cartridges in a multi-cartridge file set shall be required to be identical to that of the initial cartridge, except for the following fields:

Offset(s)		
	4-7:	Starting, ending segment numbers.
	56:	Multiple cartridge bit (bit 1).
	57:	Multi-cartridge sequence number.
	58-83:	Vendor extension data.

All character strings shall be left-justified and blank filled. Blank - Hex 20. All other character codes are permitted and system specific.

Each volume table entry shall be formatted as follows (Word and Double word bytes are stored low to high order).

OFFSET(S)	USAGE
0 - 3	Volume entry signature. Must equal the ASCII uppercase string "VTBL" for all used entries.
4 - 5	Starting segment number. Word: Number of first segment within the range allocated to this volume entry (for this cartridge).
6 - 7	Ending segment number. Word: Number of last segment within the range allocated to this volume entry (for this cartridge).

OFFSET(S)	USAGE
8 - 51	Volume entry description string. May be any string of ASCII text. "First byte zero if none specified".
52 - 55	Volume entry storage date and time. Double word, the bits are encoded as follows: <u>Bits:</u> 31-25: Year-1970 (0-127 = 1970-2097) 24-0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), R=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))
56	Volume flags. Low order bit is zero: <u>Bit:</u> 0: Vendor specific volume bit. Set if remainder of volume entry is vendor specific. Only this bit and the previous bytes (0-55) are defined. 1: Multiple cartridge bit. Set if file set spans to another cartridge. 2: Non-verification bit. Set if file set was written without verification. 3: Re-direction inhibit. Set if file set re-direction is disallowed. 4: Segment-spanning bit QIC-40 through Rev. L: use bit 4 = 0 QIC-40 Rev. M. and higher: use bit 4 = 0 if segment spanning was not used in recording the file set; use bit 4 = 1 if segment spanning was used in recording the file set 5-7: Reserved, set to zero.
57	Multi-cartridge sequence number. Verifies a correct multiple cartridge loading sequence. Initial cartridge= 1, next cartridge= 2, etc. Can be greater than 1 only in the first volume table entry written.
58 - 83	Vendor extension data. Reserved for unique vendor extensions to the volume entry. If used, should be combined with a signature to insure recognition of valid data only.
84 - 91	File set password. First byte zero if none specified. Otherwise, may be any ASCII string. Controls file set read access only.
92 - 95	Directory section size. Double word: Area at beginning of file set image reserved for the directory table. The table shall not be required to fill this area completely. The

XOSI Directory Extension, if present, will be included within this area, following the final entry in the directory table.

OFFSET(S)	USAGE
96 - 99	Data section size. Double word: Total size of the file data section following the directory table. Includes all cartridges if a multi-cartridge sequence. This total size consists of all File Header data and File data (if present), including all XOSI Files containing file-specific extended Operating System information.
100-101	OS version number. First byte is operating system major version number, second is minor. If undefined, these bytes shall be zero. This version is associated with Operating System Type field (Offset 121). For example, if the OS Type field is 3 (OS/2), and the version is 1.21, the first byte is 1 and second is 21 (or 15h).
102-117	Source drive volume label. First byte zero if none specified. Otherwise, may be any ASCII string.
118	Logical device file set originated from. If undefined, this byte shall be zero.
119	Physical device file set originated from. If undefined, this byte shall be zero.
120	Compression method used. See QIC-123 for definition. <u>Bits:Usage:</u> 0-5: QIC compression code identifier field (See QIC 123), hex 3F is reserved to indicate compression algorithm is not registered with QIC. 6: Always 0. used. 7: Set to one if compression is used.
121 - 122	Operating System type bit mask 00h = Unknown 01h = DOS 02h = UNIX 04h = OS/2 08h = Macintosh 10h = Novell NetWare 20h = Lan Manager/Server 40h-80h = reserved for future OS support

(note: these bits are mutually exclusive)

OFFSET(S)	USAGE
123	Byte: ISO compression method used

Bits:	Usage:
0 - 7:	ISO compression algorithm ID field (hex values 00, 01 or FF indicates the algorithm is not ISO registered)

OFFSET(S)	USAGE
124 - 127	Reserved, set to zero.

8.1 COMPRESSION MAP SEGMENTS (OPTIONAL)

This new segment type is optional and provides a framework for a map of the tape which duplicates the FILE SET BYTE OFFSET information at a known location (one segment, except for 1,100 foot tapes only, which use 3 contiguous segments). This map allows the drive to rapidly find the exact starting segment number of a particular file in a compressed volume.

OFFSET(S)	USAGE
Byte Offset:	
0 - 3	COMPRESSION MAP SEGMENT signature string. Used to verify that the COMPRESSION MAP SEGMENTS have been correctly found. Must be set to ascii "DCMS" (Data Compression Map Segment) when map is created.
4 - 5	Word: total length of COMPRESSION MAP in bytes. Will be 5,440 for QIC-40 205 foot tape, 8,160 for 307.5 foot tape, and 29,200 for 1,100 foot tape.
6 - 11	Unused, set to zero.
12 - nn	Double word: FILE SET BYTE OFFSET value for segment 3 to last segment number on standard length tape.

Each 4 byte double word in the COMPRESSION MAP indicates the FILE SET BYTE OFFSET for that segment. Unused entries must be set to zero. The offset into the compression map for any segment can be calculated as:

Offset = Segment Number*4

Each tape may have only one set of COMPRESSION MAP SEGMENTS, and its existence and location can be determined by examination of the HEADER SEGMENT. Note that the COMPRESSION MAP SEGMENTS, if present, may exist only within the LOGICAL AREA.

The COMPRESSION MAP SEGMENT (or beginning segment of the set of segments for 1,100 foot tapes) should optimally be placed immediately after the VOLUME TABLE SEGMENT, segment size permitting. This will not always be possible as when appending compressed volumes onto a tape previously written only by older non-compression drive software.

To determine the location of a particular file in a particular volume, the VOLUME TABLE SEGMENT is first accessed to determine whether that volume contains compressed data. If not, the location of the file is determined from the FILE SET DIRECTORY and the BAD SECTOR MAP.

If the volume does contain compressed data, the starting segment of the file is determined from examination of the FILE SET DIRECTORY and the COMPRESSION MAP; the FILE SET DIRECTORY determines the byte offset into the volume, and the COMPRESSION MAP determines the segment which contains the first byte of the file.

9.0 FILE SET

A file set shall consist of a directory section and a data section concatenated together to form a complete file set image. File set images (joined with a leading volume table) combine to fill the logical data area of the cartridge. In front of this data area are the header segments (see appendix D for an example of this arrangement).

The directory section of the file set shall precede the data section, and shall be contained completely on the first cartridge used. A data header shall precede each item within the data section to provide redundant directory information.

9.1 FILE SET DIRECTORY

The file set directory shall be a table of individual file and sub-directory entries, each entry of variable size. The table shall maintain the overall tree structure that existed on the source drive, by requiring its entries to follow a preorder traversal of the source tree (see appendix C).

The root sub-directory shall be placed first in the directory table. Its last entry (as will the last entry of all sub-directories) shall be indicated by the setting of an indicator bit. Following will be the first root sub-directory, and below it all of its sub-directories. The end of the entire directory table is indicated by the final entry having a special "end-of-table" indicator bit set.

The case of an empty root directory yields a totally empty file set. This shall be recorded by storing a zero total data section size within the file set volume table entry.

9.1.1 File Set Directory Entry

An entry shall be made in the file set directory table for each file and sub-directory. These entries consist of three parts: a fixed portion, an optional system specific portion, and a name portion.

Each individual directory entry is variable in length, and shall be formatted as follows (Word and Double word bytes are stored low to high order):

```
OFFSET(S)      USAGE
---------      ---------------------------------------
    0     Fixed/system size.  Size of fixed portion + system
               specific portion of entry.  Value does NOT
               include this byte.

*****  Start of fixed portion of directory entry  *****

    1         Attribute byte  (low order bit is zero):
               Bit:
                0: Set if owner read access allowed
                1: Set if owner write access allowed
                2: Set if owner execute access allowed
                3: Set if object hidden
                4: Set if object system
                5: Set if object a sub-directory
                6: Set if last entry in the current
                   sub-directory
                7: Set if last entry in entire
                   directory table

               In those cases where a particular attribute
               bit's meaning is undefined by the source
               environment, that attribute bit shall be set
               to a default value.  The default values for
               the bits shall be as follows:
               Bit:
                0: (Read Access):    SET
                1: (Write Access):   SET
                2: (Execute Access): SET
                3: (Object Hidden):  NOT SET
                4: (Object System):  NOT SET

  2 - 5   Last modification date and time double word.  The
               bits are encoded as follows:
               bits:
               31-25: Year-1970 (0-127 = 1970-2097)
               24-0: Month, day, hour, minute and second. Let
               MO=month (0-11), DY=day (0-30), HR=hour (0-
               23), MN=minute (0-59) and SC=second (0-59).
               Stored here is:
               SC+60*(MN+60*(HR+24*(DY+31*MO)))

  6 - 9   Entry data section size.  Double word:
               If a file:
                    Size of the data header plus the count of
                    the file data bytes.
               If a file that has been "linked" or
               If a device or
               If an empty sub-directory:
                    Size of the data header only.
               If a non-empty sub-directory:
                    Zero.

               See section 9.2 for a description of the
               "data header" component.
```

***** End of fixed portion of directory entry *****

System-specific data may exist at this point. Such data exists when the size at offset 0 of the entry exceeds 9.

OFFSET(S)	USAGE
10	System specific data. 0=Vendor specific data follows. 1= UNIX extension area follows. 2= File unreadable at backup time. The file could not be read at the time of the backup. This can be used if the person doing the backup did not have access rights to this file for some reason (e.g. file open at time of backup, or, file access changed since start of backup). The number of data bytes specified in the volume directory are written to tape, not the number of bytes available at the time of file access.

The UNIX extension area is described below. These fields will exist when the value at offset zero exceeds 9, and the value at offset 10 equals 1.

OFFSET(S)	USAGE
11	UNIX attribute byte 1: Bit 0: Set if group read access allowed 1: Set if group write access allowed 2: Set if group execute access allowed 3: Set if other read access allowed 4: Set if other write access allowed 5: Set if other execute access allowed 6-7: Reserved, set to zero
12	UNIX attribute byte 2: Bit 0: Set user ID on execution 1: Set group ID on execution 2: Save text image after execution 3: Set if item has been "linked" 4: Set if object is a character device 5: Set if object is a block device 6: Set if object is a regular device 7: Set if object is a FIFO device
13 - 16	UNIX creation date and time double word. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970-2097) 24-0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))

OFFSET(S)	USAGE
17 - 20	UNIX last access date and time double word. The bits are encoded as follows: 31-25: Year-1970 (0-127 = 1970-2097) 24- 0: Month, day, hour, minute and second. Let MO=month (0-11), DY=day (0-30), HR=hour (0-23), MN=minute (0-59) and SC=second (0-59). Stored here is: SC+60*(MN+60*(HR+24*(DY+31*MO)))
21 - 24	UNIX Inode number. If the link bit is set, this field will have the same number as the directory entry that this entry is linked to. The data for this entry will be stored in the directory entry with this Inode number that does not have the link bit set.
25 - 28	UNIX user ID.
29 - 32	UNIX group ID.
33	Unix Major Device Number. This field is undefined if object is not a device (offset 12 bits 4-7).
34	UNIX Minor Device Number, This field is undefined if object is not a device (offset 12 bits 4-7).

The remaining offsets assume no system specific data exists within the directory entry.

***** Start of name portion of directory entry *****

10	Entry name size. Value does NOT include this byte.
11 - nn	Entry name. ASCII file or sub-directory name string.

***** End of name portion of directory entry *****

9.2 XOSI Directory Extension Area

The file set directory section contains extended directory information in the area after the final directory entry. This section is referred to as the XOSI Directory Extension, and is invisible to old software, which merely skips to the end of the directory section after encountering the final directory entry. The directory section size field (offsets 92 - 95) in the volume table includes the XOSI Directory Extension.

The XOSI Directory Extension Area is used to store all OS-specific information pertaining to the entire volume, and all such information pertaining to directories within the volume. In addition, this section MAY be used to store file-specific information which vendors consider important to have available during volume preprocessing; all such file-specific information stored in this section MUST also appear in the relevant redundant file in the File Set data section, however.

The XOSI Directory Extension is composed of an Operating System Extended Volume Information record, followed by XOSI Directory record(s), which is a recapitulation of those entities from the primary directory entry for which extended OS information exists.

Extended Volume Information Record

Byte Offset:	Usage
0 - 3	"XOSI" Signature
4 - x	List of Global Volume XOSI records

The XOSI Directory Extension contains XOSI records pertaining to the entire volume. For example, under NetWare 386 v3.0, this section will contain the Bindery information records for the entire volume and the Volume Restrictions for all objects.

* Repeat for each directory (or file) entry with extended information: *

XOSI Records

Byte Offset:	Usage
0 - x	The original directory entry for the following XOSI list
x+1 - x+y	XOSI record list, terminated by a null XOSI record.

The XOSI directory extension structure is stored in standard QIC left to right traversal with entries present only for those directory or file entries that have XOSI records. Each entry duplicates the original entry in the standard directory section (with the possible exception of the "last entry" bits), and is immediately followed by a list of XOSI records pertaining to that entry. Every XOSI list is terminated by the null XOSI record. It is permissible, though not required, to include as "place-holder" entries which contain only the null XOSI entry in their XOSI list. The end of the XOSI Directory Section is signified by the presence of an entry with the "last entry in entire directory" bit set; the next null XOSI record encountered terminates the XOSI Directory Section.

The XOSI directory extension area will contain only a subset of extended Operating System records. In any environment, if the Operating System supports multiple file names for a single entry, these names (or aliases) are to present in the directory area. For NetWare, this subset includes the NetWare Bindery records for a volume, Name Space information and security information. Under OS/2, this subset includes the Extended Attributes for sub-directories. Under Lan Manager/Server, this subset includes the Resource list, User Account Subsystem data base, and Server Control Information.

The "total" directory section size shall be enlarged during the committing of this section to ensure that records may be added during the re-writing phase, after the backup operation has completed, if required. If the number of records causes this section to overflow, the Overflow Exception Record will be placed indicating that this information is not complete.

9.2.1 Extended Operating System Information Records

Extended Operating System Information records shall be defined for a wide variety of OS-specific attributes which may apply to the entire Volume, to specific directories, or to specific files. The goal is to allow for duplication of all the information needed to allow restoration of all directories or files to a state in the target environment as close as possible to that item's original state in its source environment.

Several operating systems maintain creation dates, last accessed dates, user access rights, icon information, Extended Attributes, long names, and multiple names. This information is stored on tape as a series of records (data structures) that are specific to the source Operating System. These records always contain an initial OS ID, an XOSI record ID, and a length field.

This document describes the particular structures used for storing DOS, Novell NetWare 286 v2.1 and v2.15, Novell NetWare 386 v3.0 and v3.1, and OS/2 v1.2 Extended Attributes and HPFS.

Standard Structure of Extended OS Information Records

**** Start of <u>Fixed Length Fields</u> ****

<u>Byte Offset</u>

0	Entry type	
	0	= Terminal ("null") XOSI Record
	1	= Vendor specific
	2	= DOS
	3	= UNIX
	4	= OS/2
	5	= Macintosh
	6	= Novell NetWare
	7	= LAN Manager
	8 - 254	= reserved for future OS support
	255	= Special Control records
1	XOSI ID	
2 - 5	Length of record (not including Entry type, XOSI ID, or this field)	

**** End of <u>Fixed Length Fields</u> ****

**** Start of <u>Variable Length Fields</u> ****

6 - x Extended Operating System Record (of type XOSI ID)

**** End of <u>Variable Length Fields</u>

9.2.1.1 Special Control Records Entry Type = 255

9.2.1.1.1 Overflow Exception Record XOSI ID = 1

The Overflow Exception Record: indicates that the directory area was rewritten, but the total size needed to commit all the XOSI directory section records would overflow the "total" directory section size. Therefore this information is incomplete.

<u>Variable Length Fields:</u>

none

9.2.1.1.2 Span Tape Offset Record XOSI ID = 2

The Span Tape Offset Record: shall be placed in the XOSI directory section to indicate to a spanned tape the total bytes placed on all tapes in the volume set. This facilitates the subsequent read operations allowing new software to indicate directly on what tape a file may exist. Also, if this record does not exist on a spanned tape, it can be inferred that the tape directory section was not re-written and the tape directory may be "out-of-synch" with the actual file set data committed during the backup process.

Variable Length Fields:

Byte Offset

6	Total Tape Volume Count
7 - x	Repeating fixed length structure of Tape Total Bytes

**** Repeat for the number of total tape volume count.****

y - (y + 3)

**** End of Repeating fixed length structure ****

9.2.1.2 Extended DOS Information Record(s) Entry Type 2

9.2.1.2.1 DOS Long Path Translation Record XOSI ID = 1

This record shall appear in the XOSI directory section in order to specify the true pathname (greater than 64 characters in length) of the associated alias directory.

Variable Length Fields:

Byte Offset:

6 - 7	Path length
8 - x	Complete pathname

9.2.1.3 Extended UNIX Information Record Entry Type 3

9.2.1.3.1 UNIX Extended Information Record XOSI ID = 1

Variable Length Fields:

Byte Offset:

6 Attribute (BYTE 1)

bit:
0 = Group Read Access allowed
1 = Group Write Access allowed
2 = Group Execute Access allowed
3 = Other Read Access allowed
4 = Other Write Access allowed
5 = Other Execute Access allowed
6 - 7 = reserved (set to zero)

```
     7    Attribute (BYTE 2)

     bit:
     0 =  User ID on Execution
     1 =  Group ID on Execution
     2 =  Save text image after Execution
     3 =  Item has been "linked"
     4 =  Object is a Character Device
     5 =  Object is a Block Device
     6 =  Object is a Regular Device
     7 =  Object is a FIFO Device

8 - 11    Creation Date and Time

     bit
     0 - 24    = Month, Day, Hour, Minute, and Second
                 Let MO=Month (0-11), DY=Day (0-30), HR=Hour
                 (0-23),   MN=Minute (0-59), and SC=Second (0-
              59).
                 Formula is:
                   SC+(60*(MN+(60*(HR+(24*(DY+(31*MO)))))))
     31-35= Year - 1970 (0-127 = 1970-2097)



12 - 15   Last Accessed Date and Time (stored same as above)

16 - 19   Inode Number: If the "linked" bit is set, this
          field will have the same number as the directory
          entry that this entry is linked to.  The data for
          this entry will be stored int the directory entry
          with this Inode number that does not have the link
          bit set.

20 - 23   User ID

24 - 27   Group ID

28 - 29   Major Device Number: This field is undefined if
          object is not a device (offset1 bits 4-7)

30 - 31   Minor Device Number: This field is undefined if
          object is not a device (offset1 bits 4-7)
```

9.2.1.4 Extended OS/2 Information Records Entry Type 4

OS/2 originally was shipped using the FAT filing system, therefore, the current base QIC entry structure satisfies the requirements.

With the release of OS/2 v1.2, Extended Attributes could be saved for files and directories stored in a FAT volume. Also, OS/2 v1.2 introduced the High Performance File System which added extended file and directory information maintained within the file system and ability to handle entry names as large as 255 characters.

The OS/2 extended information will be maintained in separate records to maintain optimum space utilization. For example, files or directories which do not have any Extended Attributes will not have an Extended Attribute information record.

9.2.1.4.1 OS/2 EA Record XOSI ID = 1

Variable Length Fields:

Byte Offset:

6	EA Flags*
7	Length of EA Name
8 - 11	Length of EA Value
12 - y	EA Name

9.2.1.4.2 HPFS Information Record XOSI ID = 2

Variable Length Fields:

Byte Offset:

6 - 7	Creation Date
8 - 9	Creation Time
10 - 11	Last Accessed Date
12 - 13	Last Accessed Time
14 - 15	Last Modified Date
16 - 17	Last Modified Time
18 - 21	Entry Size
22	Attributes

9.2.1.4.3 HPFS Long Name Record XOSI ID = 3

Variable Length Fields:

Byte Offset:

6 - 7	Long Name Length
8 - y	Long Name

9.2.1.5 Extended Macintosh Information Records Entry Type 5

Currently, the QIC file formats have not been extended to inherently support the Macintosh filing system. Although, both Novell NetWare and 3Com's 3+Open and 3+Share support Macintosh files, the extended generic information structure is limited to supporting only the Macintosh Long Name and the file Resource Fork. By creating a Macintosh class, any further true QIC support can be made by adding more sub-classes.

Apple Computer's Apple and Macintosh computer is inherently based on the Motorola 68000 series of microprocessor. The mechanism used by the 68000 series stores the high (most significant) byte of a word before the low (least significant) byte. This differs from Intel's low-high format. Any storage of this information shall be placed on tape in the Intel format, unless otherwise stated.

9.2.1.5.1 Macintosh Long Name Record XOSI ID = 1

Variable Length Fields:

Byte Offset:

6	Long Name Length
7 - y	Long Name

9.2.1.5.2 Macintosh Resource Fork Record XOSI ID = 2

Variable Length Fields:

Byte Offset:

6 - 9	Macintosh Resource Fork Length
10 - y	Resource Fork (low-hi)

9.2.1.5.3 Macintosh AFP Volume Information Record XOSI ID =3

Variable Length Fields:

Byte Offset:

6 - 32	Volume Name
33 - 34	Volume Signature
35 - 38	Creation Date and Time
39 - 42	Last Modification Date and Time
43 - 46	Last Archived Date and Time
47 - 54	Volume Password

9.2.1.5.4 Macintosh AFP Information Record XOSI ID = 4

Variable Length Fields:

Byte Offset:

6 - 7	Attributes
8 - 39	Finder Information
40 - 43	Creation Date and Time
44 - 47	Last Modification Date and Time
48 - 51	Last Archived Date and Time
52 - 55	Owner ID

56 - 59	Group ID
60	Owner Access Rights
61	Group Access Rights
62	World Access Rights
63 - 68	Pro DOS Info

9.2.1.6 Extended Novell NetWare Information Records

Entry Type 6

Up until Novell released NetWare 286 v2.15, it supported connectivity inherently only for DOS based clients. NetWare 286 v2.15 released with built in support for Macintosh files utilizing the AppleTalk Filing Protocol (AFP) standard on any NetWare managed volume. The original Application Programming Interface (API) of retrieving and setting entry information was changed to support an extended data structure embodying both standard and Macintosh filing information. This information is maintained in the NetWare Name Space record for Macintosh.

Novell released NetWare 386 v3.0 and v3.1 supporting only a standard set of entry information with the ability to install Macintosh Name Space. Two separate APIs are utilized to retrieve and set this information. NetWare v3.11 (release in April, 1991) supports the ability to also install UNIX's NFS, FTAM, and HPFS Name Space. As NetWare evolves further, more operating systems will be installable and new API's and data structures will emerge.

Novell NetWare 286 maintains security information at the directory level. Security is designed to manage objects. A user is an object and has a trustee ID. The security data base under NetWare is the Bindery. All trustees have certain rights on any NetWare managed volume. NetWare 386 supports security information to the file level.

Novell also incorporated size restrictions for directory and objects in NetWare 386 v3.0, known as Directory and Volume Restrictions, respectively. For example, a user may not be able to put more than 5 megabytes of files on a particular NetWare volume.

The NetWare extended information will be maintained in separate records to maintain optimum space utilization.

9.2.1.6.1 NetWare Server Information Record

XOSI ID = 1

Variable Length Fields:

Byte Offset:

6 - 53	Server Name
54	NetWare Version
55	NetWare Sub-Version

9.2.1.6.2 NetWare Volume Name Record XOSI ID = 2

Variable Length Fields:

Byte Offset:

6 - 20	Volume Name

9.2.1.6.3 NetWare 286 v2.1 File Information Record XOSI ID = 3

Variable Length Fields:

Byte Offset:

6	Attributes
7	NetWare Attributes
8 - 11	Entry Size (hi,low)*
12 - 13	Creation Date (hi,low)
14 - 15	Last Accessed Date (hi.low)
16 - 17	Last Modified Date (hi,low)
18 - 19	Last Modified Time (hi,low)
20 - 23	NetWare Owner Trustee ID
24 - 25	Last Archived Date (hi,low)
26 - 27	Last Archived Time (hi,low)

note:
* Notation designating that this fields byte order is maintained in Motorola high byte precedes the low byte format (due to Novell's earlier 68000 based hardware).

9.2.1.6.4 NetWare 386 v3.0 Information Record XOSI ID = 4

Variable Length Fields:

Byte Offset:

6	Attributes
7	NetWare Attributes
8 - 9	NetWare 386 Attributes
10	Name Space Type (0=DOS, 1=Macintosh, 2-255=Undefined)
11 - 12	Creation Time
13 - 14	Creation Date
15 - 18	NetWare Owner Trustee ID (hi,low)
19 - 20	Last Archived Time
21 - 22	Last Archived Date
23 - 26	Last Archiver NetWare Trustee ID

27 - 30	Last Modifier Trustee ID (hi,low)
31 - 34	Data Fork Size
35 - 36	NetWare Inherited Rights Mask
37 - 38	Last Accessed Date

9.2.1.6.5 NetWare Name Space Record XOSI ID = 5

Variable Length Fields:

Byte Offset:

6 - yNetWare Name Space record

Byte Offset:

6	Type = 1 (Macintosh Name Space record)
7	reserved
8 - 39	Macintosh Finder Information
40	Macintosh Name Length
41 - 72	Macintosh Name
73 - 76	Macintosh Resource Fork Length

Byte Offset:

6	Type = 2 (HPFS Name Space record)
7	reserved
8 - y	Currently not available.

Byte Offset:

6	Type = 3 (NFS Name Space record)
7	reserved
8 - y	Currently not available.

Byte Offset:

6	Type = 4 (FTAM Name Space record)
7	reserved
8 - y	Currently not available.

9.2.1.6.6 NetWare Security Record XOSI ID = 6

Variable Length Fields:

Byte Offset:

6 - y	NetWare Bindery or Restriction record

Byte Offset:

6	Type = 1 (NetWare Bindery record)
7	Reserved
8 - 11	Object (or Trustee) ID (hi,low)
12 - 13	Object Type (hi,low)

```
          1  = User
          2  = Group
          3  = PrintServer

14 - 61   Object Name
62        Object Flag (0=Static, 1=Dynamic)
63        Object Security Mask
          Low nibble= Read security
          High nibble    = Write security

          bit:
          0  = ANYONE
          1  = LOGGED
          2  = OBJECT
          3  = SUPERVISOR

64        Object  Properties Flag
          0 = No Properties

65 - y    Property  List  is  a  list  of  the  following
          random size structures:

Byte Offset:

65 - 80   Property Name ASCIIZ
81        Property Flag Mask

          bit:

          0 = 0   STATIC
          0 = 1   DYNAMIC
          7 = 0   ITEM
          7 = 1   SET

82   Property Security Mask
          Low nibble = Read security
          High nibble = Write security

          bit:
          0  = ANYONE
          1  = LOGGED
          2  = OBJECT
          3  = SUPERVISOR

83        Number of 128 byte Property Records

84 - y    Property Records

Byte Offset:

6         Type  =  2  (NetWare  286  Directory  Restriction
          record)

7         Reserved

8 - y  Trustee and Rights list
```

*** Repeating Structure ***

Byte Offset:

0 - 3	Trustee ID
4	Rights Mask

*** End of Repeating Structure ***

Byte Offset:

6	Type = 3 (NetWare 386 Entry Restriction record)
7	Reserved
8 - y	Trustee and Rights list

*** Repeating Structure ***

Byte Offset:

0 - 3	Trustee ID
4 - 5	Rights Mask

*** End of Repeating Structure ***

Byte Offset:

6	Type = 4 (NetWare 386 Directory Restriction record)
7	Reserved
8 - 11	NetWare Directory Maximum Blocks

Byte Offset:

6	Type = 5 (NetWare 386 Volume Restriction record)
7	Reserved
8 - y	Volume Restrictions List

*** Repeating Structure ***

Byte Offset:

0 - 3	Trustee ID
4 - 7	Restriction

*** End of Repeating Structure ***

9.1.2.6.7 NetWare 386 v3.0 Sparse File Record XOSI ID = 7

Variable Length Fields:

Byte Offset:

6 - ySparse File Data records is a list of the following random length structures

6 - 9	Fragment Offset
10 - 13	Fragment Size
14 - y	Fragment Data

9.1.2.6.8 NetWare 286 v2.1 Directory Information Record XOSI ID = 8

Variable Length Fields:

Byte Offset:

6 - 7	Creation Time (hi-low)
8 - 9	Creation Date (hi-low)
10 - 13	Netware Owner Trustee ID
14	Netware Inherited Rights Mask

9.2.1.7 Extended Lan Manager Information Records Entry Type 7

Lan Manager emerged from Microsoft's PC-Net roots. It was originally released as an OEM product that was shipped as IBM's Lan Server and 3Com's 3+Open. Microsoft released Lan Manager v2.0 directly. As this product evolves, more QIC vendors will support it directly. Future releases of Microsoft's Lan Manager will include support for Macintosh and other file systems, similar to Novell NetWare 386.

Lan Manager is based on OS/2 and therefore supports Extended Attributes, HPFS extended entry information, and HPFS Long Names. Lan Manager v2.0 also supports HPFS-386, a local security system for non-dedicated file servers.

The following data structures are incomplete and final data structures are forthcoming.

9.2.1.7.1 Lan Manager Server Control Record XOSI ID = 1

Variable Length Fields:

Byte Offset:

6 Type = 1

9.2.1.7.2 Lan Manager Resource List Record XOSI ID = 2

9.2.1.7.3 Lan Manager User Account Sub-System Record XOSI ID = 3

9.3 FILE SET DATA

The data section of a file set consists of data bytes for all of the items listed in the directory table (except for non-empty sub-directories), concatenated together. Data items occur in the data section in the same order as do their corresponding entries within the directory table.

Each block of data shall be preceded by a data header formatted as shown below. To achieve maximum density, no sector or segment boundaries shall be observed when this area is created.

Directory entries are included into this area only when they are for an empty sub-directory. This inclusion is to allow as much of the tree structure as possible to be rebuilt if the directory table (or cartridge containing it) is lost. Such entries consist of a data header only, with no additional data following.

OFFSET(S)	USAGE
0 - 3	Data header signature. Hexadecimal string 'CC33CC33'.
4 - n	Data header directory entry. Copy of entry from the directory table for this item.
n+1	Data header path size. Does NOT include this byte.
n+2 - n+?	Data header path. Full ASCII path to the sub-directory this item resides in. The file or sub-directory name itself is NOT included. Each sub-directory name in the path is separated by a null (HEX 00) character. No leading or trailing null is stored.

Each file located in the data section may be preceded by an XOSI file containing extended OS information for that file. Both this XOSI file and the actual file shall appear separately in the standard QIC directory, and some or all of the XOSI records within the XOSI file may optionally appear in the XOSI Directory Extension area as an aid to read access preprocessing.

The total size of the extended Operating System information is the size of the redundant file. Since there is no way of guaranteeing that a file's information will remain constant (in a multi-user environment) from the time the directory is built until the time the actual file or directory is copied to tape, the extended Operating System information size is only known when the file is to be committed to tape. This size is also used in the file set directory.

Historically, the QIC 40/80 standards emerged to support the DOS FAT based filing system with all entries implied to be in the 8.3 format. Although UNIX extensions supporting longer path and file names were added to early versions of QIC-40/80, these extension were not part of any comprehensive strategy for extensibility. Therefore, all further enhancements to the QIC40/80 standard, including this one, and all other new naming methods will be encapsulated as XOSI records. This new version of QIC 40/80 supporting XOSI records does not invalidate UNIX software and tapes with long path and file names, that conform to these early versions of QIC 40/80.

For consistency with this standard in file system environments that do not conform to this naming convention, a placeholder file name shall be generated for the standard QIC directory entry. The corresponding XOSI directory entry will contain a name space record containing the original file name. The method generating the placeholder name shall create a name corresponding as closely as possible to the original name in a truncated form.

Each XOSI file will have a consistent naming convention allowing for easy exclusion for old software. Only files are allowed to have associated XOSI files. These files will be named "!DELETE!.ME!" indicating to old software users that this file should be excluded from any normal operations.

To preserve file access capability using software implemented for DOS only, path lengths in the primary directory section shall be limited to 64 characters. Whenever a path longer than this is encountered, an alias directory shall be created under "\XOSIDIR\" in the primary directory section. These alias directories shall be named sequentially, beginning with "00000001". Every unique native path which exceeds 64 characters in length shall be assigned to one unique alias directory; no hierarchical directory structure shall exist beneath each alias directory.

The true long pathname will be contained in the XOSI directory section as a Long Path Translation Record associated with each alias directory.

9.3.1 Compression of File Set Data

Case 1: No segment spanning used (bit 4 of byte 56 of the volume table entry = 0; refer to section 8.0)

Data compression of file set directory, data and file headers is performed on a segment by segment basis. The first 6 bytes of each segment are not compressed and are called the COMPRESSION SEGMENT STRUCTURE. It contains the following information:

COMPRESSION SEGMENT STRUCTURE

FILE SET BYTE OFFSET	- 4 bytes
BYTE COUNT	- 2 bytes

(INTEL lo-hi byte format is used)

The legal range of values for the BYTE COUNT field is from 1 to 29K-6. The values of 0 and the range 29K-5 to 29K-1 are illegal.

The FILE SET BYTE OFFSET found in either the COMPRESSION MAP SEGMENT or in the COMPRESSION SEGMENT STRUCTURE at the beginning of each compressed segment, provides a means to seek out to individual files without decompressing preceding segments. The FILE SET BYTE OFFSET can be described as the sum of the number of uncompressed data bytes in all preceding segments of the current volume. For example, in the first segment of a volume, this offset is zero. In the second segment, it is the total number of uncompressed data bytes contained in the first segment. In the third segment, it is the sum of the uncompressed data bytes in the first and second segments.

The BYTE COUNT defines how many bytes of data follow in this segment. The actual data may be in compressed or uncompressed form. The high order bit is set if the data is uncompressed, and clear if the data is compressed. The remaining 15 bits indicate the number of data bytes that follow. If the number of bytes available in the current segment, less 18, is greater than the BYTE COUNT, then another BYTE COUNT must follow. This feature allows compressed and uncompressed data to reside in the same segment. The last 12 bytes of a segment need not be used.

Case 2: Segment spanning used (bit 4 of byte 56 of the volume table entry = 1; refer to section 8.0) The following section describes differences from case 1.

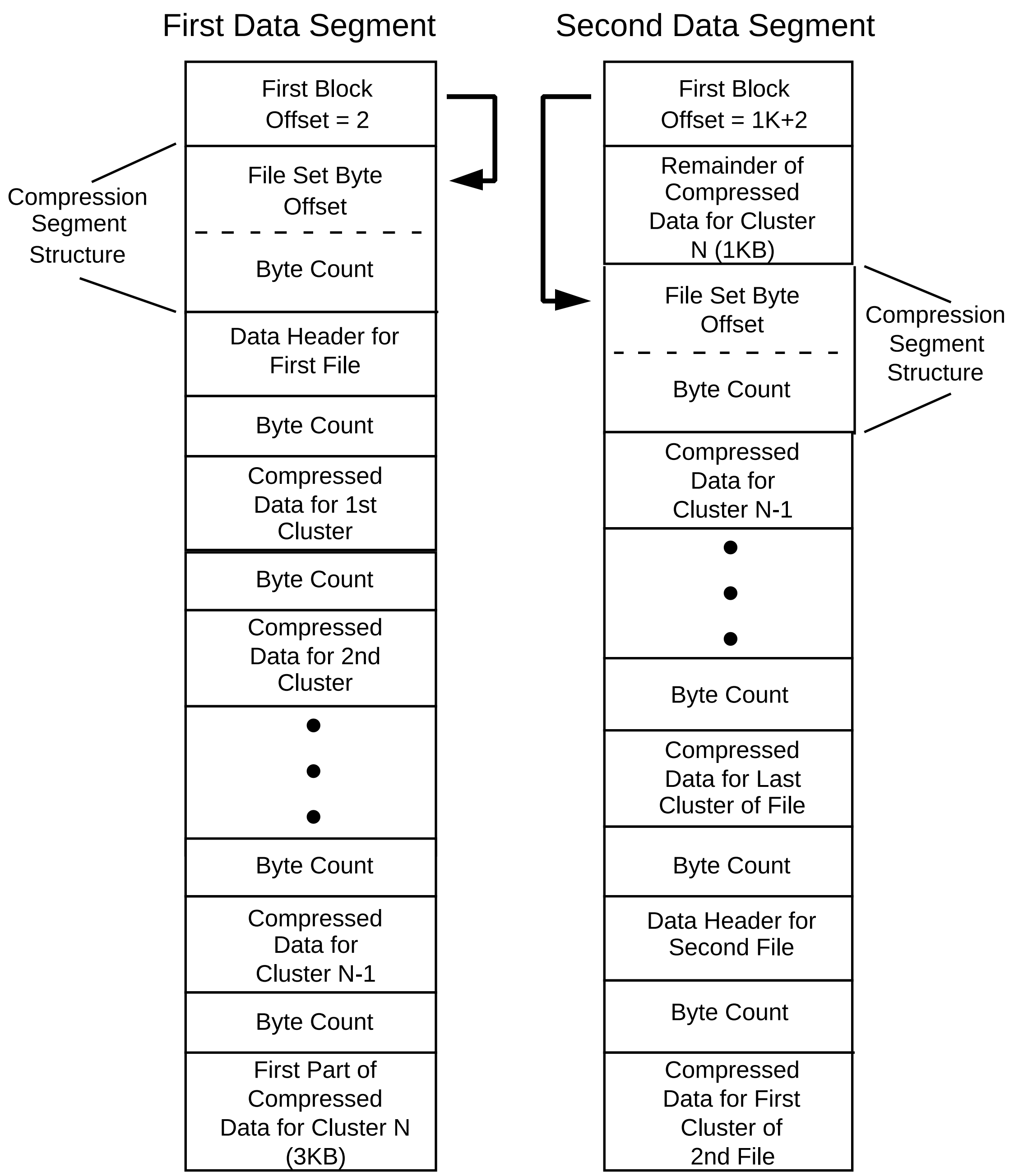
If the segment spanning bit for the file set is one then it is possible that the beginning of a given segment contains the end of a compressed data block which began in the previous segment. To allow a drive to find the beginning of the first block of compressed data which starts within the segment, an uncompressed two-byte FIRST BLOCK OFFSET occurs at the beginning of each segment of data. The FIRST BLOCK OFFSET contains the offset in bytes from the start of the segment to a COMPRESSION SEGMENT STRUCTURE which immediately precedes the first block of compressed data which starts within the segment. The value of the FIRST BLOCK OFFSET includes the two bytes of the FIRST BLOCK OFFSET itself.

A FIRST BLOCK OFFFSET in the range from 29K-1 to 29K-18 indicates that the last data block of the previous segment ends within the last eighteen bytes of the current segment. In this case the remaining bytes at the end of the segment are not used and the next block of compressed data begins in the next segment.

A FIRST BLOCK OFFSET of zero indicates that the last data block of the previous segment fills the current segment and continues on into the next segment.

If the segment spanning bit is set to one then a COMPRESSION SEGMENT STRUCTURE occurs at the beginning of the first compressed data block which starts within each segment. The definition of this structure is the same as in QIC-40 Rev. L and earlier except that the location is not necessarily at the beginning of the segment.

The FILE SET BYTE OFFSET for each segment that is stored in the COMPRESSION MAP SEGMENT records the value of the FILE SET BYTE OFFSET of the first compressed data block which begins within that segment. For all segments other than the first segment in the file set a FILE SET BYTE OFFSET value of zero indicates that no compressed data block begins within the segment.



First Data Segment
Second Data Segment
First Block Offset = 2
File Set Byte Offset
Byte Count
Compression Segment Structure
Data Header for First File
Byte Count
Compressed Data for 1st Cluster
Byte Count
Compressed Data for 2nd Cluster
Byte Count
Compressed Data for Cluster N-1
Byte Count
First Part of Compressed Data for Cluster N (3KB)
First Block Offset = 1K+2
Remainder of Compressed Data for Cluster N (1KB)
File Set Byte Offset
Byte Count
Compression Segment Structure
Compressed Data for Cluster N-1
Byte Count
Compressed Data for Last Cluster of File
Byte Count
Data Header for Second File
Byte Count
Compressed Data for First Cluster of 2nd File

Example of Segment Spanning by a Compressed Data Block

Figure 7

APPENDIX A

RECORDING SPECIFICATION SUMMARY

CAPACITY (Bytes before ECC)	205 FOOT TAPE
unformatted/cart (nom)	61,500,000
formatted/cart	44,564,480
formatted/tape track	2,228,224
formatted/segment	32,768
formatted/sector	1,024

CAPACITY (Bytes after ECC)	
unformatted/cart (nom)	61,500,000
formatted/cart	40,386,560
formatted/tape track	2,019,328
formatted/segment	29,696
formatted/sector	1,024

FORMAT	
tape trks/cart	20
segments/tape track	68
data sectors/segment	29
ECC sectors/segment	3
bytes/sector	1,024
bits/inch	10,000
tape tracks/inch	83.3
encode method	MFM

SECTOR ADDRESSING	
sectors/cartridge	43,520
sectors/side	21,760
sectors/tape track	2,176
sectors/floppy track	128

SIDE (0:1)   TRACK (0:169), SECTOR (1:128)

FORMAT PARAMETER RECORD IDENTIFICATION

Sector 0,offset 24,25	tape segments/track	0 or 68
Sector 0,offset 26	tape trks/cartridge	0 or 20
Sector 0,offset 27	max floppy side	0 or 1
Sector 0,offset 28	max floppy track	0 or 169
Sector 0,offset 29	max floppy sector	0 or 128

RECORDING SPECIFICATION SUMMARY

CAPACITY (Bytes before ECC)	307.5 FOOT TAPE
unformatted/cart (nom)	92,250,000
formatted/cart	66,846,720
formatted/tape track	3,342,336
formatted/segment	32,768
formatted/sector	1,024

CAPACITY (Bytes after ECC)

unformatted/cart (nom)	92,250,000
formatted/cart	60,579,840
formatted/tape track	3,028,992
formatted/segment	29,696
formatted/sector	1,024

FORMAT

tape trks/cart	20
segments/tape track	102
data sectors/segment	29
ECC sectors/segment	3
bytes/sector	1,024
bits/inch	10,000
tape tracks/inch	83.3
encode method	MFM

SECTOR ADDRESSING

sectors/cartridge	65,280
sectors/side	32,640
sectors/tape track	3,264
sectors/floppy track	128

SIDE (0:1)   TRACK (0:254), SECTOR (1:128)

FORMAT PARAMETER RECORD IDENTIFICATION

Sector 0, offset 24,25	tape segments/track	102
Sector 0, offset 26	tape tracks/cartridge	20
Sector 0, offset 27	maximum floppy side	1
Sector 0, offset 28	maximum floppy track	254
Sector 0, offset 29	maximum floppy sector	128

RECORDING SPECIFICATION SUMMARY

CAPACITY (Bytes before ECC)	1,100 FOOT TAPE
unformatted/cart (nom)	330,110,294
formatted/cart	239,206,400
formatted/tape track	11,960,320
formatted/segment	32,768
formatted/sector	1,024

CAPACITY (Bytes after ECC)

unformatted/cart (nom)	330,110,294
formatted/cart	216,780,800
formatted/tape track	10,839,040
formatted/segment	29,696
formatted/sector	1,024

FORMAT

tape trks/cart	20
segments/tape track	365
data sectors/segment	29
ECC sectors/segment	3
bytes/sector	1,024
bits/inch	10,000
tape tracks/inch	83.3
encode method	MFM

SECTOR ADDRESSING

sectors/cartridge	233,600
sectors/side	32,512
sectors/tape track	11,680
sectors/floppy track	128

SIDE (0:7)   TRACK (0:253), SECTOR (1:128)

FORMAT PARAMETER RECORD IDENTIFICATION

Sector 0, offset 24,25	tape segments/track	365
Sector 0, offset 26	tape tracks/cartridge	20
Sector 0, offset 27	maximum floppy side	7
Sector 0, offset 28	maximum floppy track	254
Sector 0, offset 29	maximum floppy sector	128

APPENDIX B REED-SOLOMON ECC

1.0 ECC FORMAT

Tape data shall be formatted in segments of 32K bytes, in 32 blocks of 1024 bytes. Each 1K block shall be written as a sector on the tape by the FDC, with appropriate header information, and the data in the sector shall be protected by a 16-bit CRC. Of the thirty-two sectors, twenty-nine contain data, while the last three contain parity information. The ECC overhead is thus 3/32 or 9.4%. The bytes are considered to be arranged in a 32 x 1024 matrix, and the parity bytes shall be chosen so that each column of the matrix is an independent Reed-Solomon codeword of redundancy three, with 8-bit characters, as shown in figure 1. In the ensuing discussion, rows and columns will be referred to as numbered in the figure. Data shall be written on the tape row by row, starting with row 0, and within each row (sector) the bytes shall be written starting with column 0.

```
                            COLUMNS
           0     1      2          ...       1022    1023
       _______________________________________________________
   0 |      |      |      |            |       |      | CRC |
     |______|______|______|____________|_______|______|_____|
   1 |      |      |      |            |       |      | CRC |  D
     |______|______|______|____________|_______|______|_____|  A
   2 |      |      |      |            |       |      | CRC |  T
R    |______|______|______|____________|_______|______|_____|  A
   3 |      |      |      |            |       |      | CRC |
O    |______|______|______|____________|_______|______|_____|  R
   . |      |      |      |            |       |      |     |  O
W  . |      |      |      |            |       |      |     |  W
   . |      |      |      |            |       |      |     |  S
S    |______|______|______|____________|_______|______|_____|
  28 |      |      |      |            |       |      | CRC |
     |______|______|______|____________|_______|______|_____|P
     ========================================================
  29 |      |      |      |            |       |      | CRC |A R
     |______|______|______|____________|_______|______|_____|R O
  30 |      |      |      |            |       |      | CRC |I W
     |______|______|______|____________|_______|______|_____|T S
  31 |      |      |      |            |       |      | CRC |Y
     |______|______|______|____________|_______|______|_____|
```

Figure 8. Data format

If either of the last three sectors of a segment is marked as "bad" during the tape pre-formatting operation, the sector assignment shall be changed so that the last three "good" sectors of a segment are filled with parity. Any "bad" sectors preceding the parity sectors shall be skipped (see examples in section 5).

2.0 DEFINITION OF ECC

Each column of data shall be a codeword of a (32,29) Reed-Solomon code of redundancy three over GF(256). Such a code can be used to correct up to three rows (data or parity) detected to be in error by the CRC in each row. Also, single bytes in error in any column which are not detected by the sector CRC can be corrected using the Reed-Solomon code. If no errors occur, the parity bytes also serve as a vertical CRC to provide additional assurance that each column is correct.

Two levels of specification are necessary in order to implement such a code. First, it is necessary to specify the mapping from bytes to field elements in order to perform finite-field mathematical operations on the bytes of the codewords. Second, given the field representation, it is necessary to choose a generator polynomial for the code, which defines how codewords will be encoded (i.e., generation of parity bytes) and decoded (i.e., correction of errors).

3.0 FIELD REPRESENTATION

GF(256) is a field consisting of 256 elements. There are many ways of representing the field elements, but, in the canonical form, each element of the field is represented as a unique polynomial of degree seven or less with binary coefficients. In other words, each field element a has the form:

$$a = a_7x^7 + a_6x^6 + a_5x^5 + a_4x^4 + a_3x^3 + a_2x^2 + a_1x + a_0$$

Field math is defined to be polynomial math modulo an irreducible binary polynomial of degree eight, f(x), where binary addition is the exclusive-or (XOR) operation and binary multiplication is the AND operation. Each such field element a shall be represented by a byte as shown in figure 2.

MSB LSB

a7	a6	a5	a4	a3	a2	a1	a0

Figure 9. Bit Numbering Convention

The irreducible polynomial used to generate the field GF(256) shall be:

$$f(x) = x^8 + x^7 + x^2 + x + 1$$

All field operations on pairs of bytes (addition, multiplication, division) shall be performed modulo this polynomial. Observe that addition is identical to subtraction, since 1 + 1 = 0. References are given at the end of this section for further information on field definitions and arithmetic.

4.0 CODE GENERATOR POLYNOMINAL

The generator polynomial for the Reed-Solomon code is of the form

$$g(x) = x^3 + ax^2 + bx + c \ ,$$

where a, b, and c are field elements. However, a, b, and c are not arbitrary. They must be chosen so that the roots of g(x) over the field are consecutive powers of a primitive element of the field (e.g., a root of f(x)). There are eight roots of f(x) in GF(256), so let r be one of them. Then g(x) will have the form

$$g(x) = (x - q)\ (x - rq)\ (x - r^2q) \ ,$$

where q is an arbitrary nonzero field element. It is simple to choose q such that g(x) becomes a "reversible" generator polynomial (IEEE Transactions on Information Theory, vol. 28, pp. 869-874, 1982). In other words, c=1, so that only a single multiplication is required for performing the feedback operation. Observe that this multiplication can be accomplished in software using a single 256 byte lookup table. Using this method, the reversible generator polynomial shall be:

$$g(x) = x^3 + r^{105} x^2 + r^{105} x + 1$$

with r a root of f(x). The hex representation of r^{105} is CO (decimal 192).

To encode a column of data, label the data bytes in a given column by their row number, as in figure 1, from d_0 to d_{28}. The column's parity bytes d_{29}, d_{30}, and d_{31} shall be chosen such that

$$d(x) = \sum_{i=0}^{N} d_i x^i,$$

is divisible by g(x), using polynomial division over the field, where N = 31 less number of "bad blocks" in the segment. In practice, encoding is done by taking the remainder using a linear feedback shift register. References are given at the end of this section for examples of implementations of such encoders, as well as the algorithms underlying the encoding/decoding processes.

A CRC failure is defined as a row of a segment in which errors occurred which were not detected by the CRC. Although such an occurrence is unlikely, the ECC shall be able to detect and correct a single such instance; otherwise, the output error rate would be adversely affected. One consequence of this requirement is that the ECC decoding routine must be called every time a segment is read from the tape, even if no CRC errors have occurred.

The ECC code as defined can correct any of the following error patterns:

a) 3 or fewer CRC errors, no CRC failures
b) 1 CRC error, 1 CRC failure
c) 1 CRC failure.

In addition, the ECC software shall be capable of detecting almost all other error patterns, which can be accomplished using the distance properties of Reed-Solomon codes.

5.0 EXAMPLES

Using the polynomials defined in the previous sections, several example codewords have been generated. The bytes are displayed below in figure 5 using hexadecimal format.

	Row #							
	0 :	00	00	00	00	00	00	01
	1 :	00	00	00	00	00	00	02
	2 :	00	00	00	00	00	01	03
	3 :	00	00	00	00	00	C0	04
	4 :	00	00	00	00	00	C0	05
	5 :	00	00	00	00	00	01	06
	6 :	00	00	00	00	00	01	07
	7 :	00	00	00	00	00	00	08
Data	8 :	00	00	00	00	00	67	09
	9 :	00	00	00	00	00	A6	0A
	10 :	00	00	00	00	00	C0	0B
	11 :	00	00	00	00	00	01	0C
	12 :	00	00	00	00	00	00	0D
	13 :	00	00	00	00	00	00	0E
	14 :	00	00	00	00	00	FF	0F
	15 :	00	00	00	00	00	99	10
	16 :	00	00	00	00	00	67	11
	17 :	00	00	00	00	00	01	12
	18 :	00	00	00	00	00	00	13
	19 :	00	00	00	00	00	00	14
	20 :	00	00	00	00	00	00	15
	21 :	00	00	00	00	00	A3	16
	22 :	00	00	00	00	00	5D	17
	23 :	00	00	00	00	00	FF	18
	24 :	00	00	00	00	01	01	19
	25 :	00	00	00	01	00	00	1A
	26 :	00	00	01	00	00	00	1B
	27 :	00	01	00	00	00	00	1C
	28 :	01	00	00	00	00	00	1D
Parity	29 :	C0	67	FF	A3	AD	AD	5D
Parity	30 :	C0	A6	99	5D	0F	0F	FF
Parity	31 :	01	CO	67	FF	A3	A3	A3

Figure 10. Example Codewords

6.0 PERFORMANCE ANALYSIS

In this section, the improvement in bit error rate due to the ECC will be informally examined. Let r be the input event error rate coming off the tape. If L is the sector size in bytes (L = 1024), then the probability that a given sector is in error (i.e., the input sector error rate) is given by:

$$P = 1 - (1 - r)^{8L} \sim- \ 8Lr.$$

The probability that exactly k sector errors occur in 32 sectors is

$$p(k) = C(32,k)\ (1 - P)^{32-k}\ P^k \sim- \ C(32,k)\ P^k,$$

where C(n,k) = n!/(k!(n - k)!) is the combinatorial "choose" function.

Since the Reed-Solomon ECC can correct up to 3 CRC errors, the probability that a given segment has an unrecoverable error due to CRC errors is simply the probability that four or more sector CRC errors occur. To first order, this probability is simply the probability that exactly four such errors occur. That is, the probability of an uncorrectable error pattern (segment error) due to CRC error is given by

$$Q1 \sim- \ C(32,4)\ P^4.$$

However, the probability of sector CRC failure (using a 16-bit CRC) is P/65536, and a segment error can also occur with two CRC failures or two CRC errors and one failure. The probability of these two error mechanisms are, respectively,

$Q2 \sim- \ C(32,2)(P/65536)^2$ and $Q3 \sim- \ 3C(32,3)(P^3/65536)$.

The overall error rate is (again to first order)

$$Q \sim- \ Q1 + Q2 + Q3.$$

As an example, suppose $p = 10^{-8}$, then $P \sim- \ 8 \times 10^{-5}$. and

$Q1 \sim- \ 1.5 \times 10^{-12}$, $Q2 \sim- \ 7 \times 10^{-16}$, $Q3 \sim- \ 10^{-13}$,

so the dominating term is Q1, as we would expect. However, notice that if the ECC did not handle the case of a single CRC failure, the segment error rate would be given by

$$C(32,1)\ P/65536 \ \sim- \ 4 \times 10^{-8}.$$

In other words, it is imperative that the CRC failure mode not be ignored.

Given the above, Q ~- Q1 ~- 1.5 x 10^{-12}, and the output bit error rate is given by

$$p_0 \text{ ~- } Q/(29 \times 8 \times L) \text{ ~- } 6.4 \times 10^{-18}.$$

From a user's standpoint, a much more important figure is the tape error rate. Since a tape with 40 MB of data has exactly 1360 segments, the tape error rate is

$$C(1360,1) \times Q \text{ ~- } 2 \times 10^{-9}.$$

In other words, only in one of 500,000,000 full read operations would an uncorrectable error occur. The corresponding number without ECC is roughly = 3.3, so that each full read of a tape would fail 3 times without ECC.

7.0 REFERENCES

Blahut, Richard, Theory and Practice of Error Control Codes, Addison-Wesley, Reading, Massachusetts, 1983.

Peterson, W. and E. Weldon, Error-Correcting Codes, MIT Press, Cambridge, Massachussetts, 1972.

MacWilliams, J. and N. Sloane, The Theory of Error-Correcting Codes, North Holland Publishing Co, Amsterdam, 1977.

APPENDIX C

FILE SET DIRECTORY TABLE STRUCTURE

The file set directory table is actually a collection of sub-directories from the original drive "tree." The tree structure of this collection is maintained by strictly specifying the order of appearance of the sub-directories within the directory table.

Each sub-directory is stored corresponding to its position in a preorder traversal of the original drive tree. Preorder traversal is defined as visiting first the root, then traversing left to right all of its children (here, left to right means the leftmost child occurs first in the parent, the next leftmost occurs second, and so forth).

There are two distinct advantages to this method. First, the path alterations required to perform the entire traversal are minimized. Second and more importantly, the data area of any complete sub-tree will be contiguous upon the tape, allowing rapid recovery of a sub-directory and its children.

As an example, consider the following sub-directory tree:

```
            ROOT
           /    \
      COMEXE    TEXT
     /      \
 STUFF      LANGUAGE
           /   |    \
  APL   C    BASIC
```

The sub-directories of this tree would be stored in the following order (indentation is for clarification):

```
ROOT
  COMEXE
    STUFF
    LANGUAGE
      APL
      C
      BASIC
  TEXT
```

While complete, the structure can be difficult to use in its native form. To simplify operations, it is useful to add forward and/or backward sub-directory links to the table when it is in memory.

As an added example, consider the following directory structure:

```
                (ROOT)
              /    |    \
       SUBA    FILE1   SUBB
       /  \            /   \
  FILE2   SUBC      SUBE   FILE 4
          /   \
      SUBD   FILE3
```

The file set directory entries would be stored in the following order:

	Attribute Byte		
	Subdir	End of Subdir	End of Table
SUBA	X		
FILE1			
SUBB	X	X	
FILE2			
SUBC	X	X	
SUBD	X		
FILE3	X		
SUBE	X		
FILE		X	X

APPENDIX D

EXAMPLE OF LOGICAL CARTRIDGE LAYOUT

In the example below, each box represents a tape segment. The corresponding segment numbers appear above within angle brackets. Segments enclosed by equal signs (=) are generated during the format operation.

```
<0>                      <1>
+========================+========================+
+    damaged segment     +    header segment      +
+   (deleted, unused)    +    (initial copy)      +
+========================+========================+
                                       |
                                       |
          <2>                     <3>  |
    +========================+========================+
    +    header segment      +     Volume table      +
    +   (secondary copy)     +       segment         +
    +========================+========================+
                 ________________________|   |
                 |                           |
<4>              |             <100>         |
+-----------------------+       +---------------------+
+ 1st file set starting + ...   + 1st file set ending +
+     data segment      +  ...  +    data segment     +
+-----------------------+       +---------------------+
```

The only impact the segments (<0> to <2>) have on the logical area is that they define the range of segment numbers it will occupy. In this example, the starting segment of this area is <3>. This initial segment shall always hold the volume table. The ending logical area segment is not shown. In a fully formatted tape it will be 20x68-1, or 1359 (tracks x seg/trk -1).

A single file set is pictured, with its starting <4> and ending <100> segment numbers pointed to by the volume table. This simplified volume table has only a single entry; however, it is likely to find many such entries within the table. Each entry would designate its own (higher numbered) range of segments.