FAT 32 Filing system routines

A reference guide and worked example

Compiled by Ed Brown - 28/05/01

1) Read Sector 0 of the drive.

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   33 C0 8E D0 BC 00 7C FB  50 07 50 1F FC BE 1B 7C BF 1B 06 50 57 B9 E5 01 F3 A4 CB BE BE 07 B1 04   3À□Ð¼.|ûP.P.ü¾.|¿..PW¹å.ó¤Ë¾¾.±.
000000020   38 2C 7C 09 75 15 83 C6  10 E2 F5 CD 18 8B 14 8B EE 83 C6 10 49 74 16 38 2C 74 F6 BE 10 07 4E AC   8,|.u.ƒÆ.âõÍ.‹.‹îƒÆ.It.8,tö¾..N¬
000000040   3C 00 74 FA BB 07 00 B4  0E CD 10 EB F2 89 46 25 96 8A 46 04 B4 06 3C 0E 74 11 B4 0B 3C 0C 74 05   <.tú».. ´.Í.ëò‰F%–ŠF.´.<.t.´.<.t.
000000060   3A C4 75 2B 40 C6 46 25  06 75 24 BB AA 55 50 B4 41 CD 13 58 72 16 81 FB 55 AA 75 10 F6 C1 01 74   :Äu+@ÆF%.u$»ªUP´AÍ.Xr.□ûUªu.öÁ.t
000000080   0B 8A E0 88 56 24 C7 06  A1 06 EB 1E 88 66 04 BF 0A 00 B8 01 02 8B DC 33 C9 83 FF 05 7F 03 8B 4E   .Šà^V$Ç.¡.ë.^f.¿..¸..‹Ü3Éƒÿ.□.‹N
0000000A0   25 03 4E 02 CD 13 72 29  BE 46 07 81 3E FE 7D 55 AA 74 5A 83 EF 05 7F DA 85 F6 75 83 BE 27 07 EB   %.N.Í.r)¾F.□>þ}UªtZƒï.□Ú…öuƒ¾'.ë
0000000C0   8A 98 91 52 99 03 46 08  13 56 0A E8 12 00 5A EB D5 4F 74 E4 33 C0 CD 13 EB B8 00 00 81 32 50 23   Š˜'R™.F..V.è..ZëÕOtä3ÀÍ.ë¸..□2P#
0000000E0   56 33 F6 56 56 52 50 06  53 51 BE 10 00 56 8B F4 50 52 B8 00 42 8A 56 24 CD 13 5A 58 8D 64 10 72   V3öVVRP.SQ¾..V‹ôPR¸.BŠV$Í.ZX□d.r
000000100   0A 40 75 01 42 80 C7 02  E2 F7 F8 5E C3 EB 74 49 6E 76 61 6C 69 64 20 70 61 72 74 69 74 69 6F 6E   .@u.B□Ç.â÷ø^ÃëtInvalid partition
000000120   20 74 61 62 6C 65 00 45  72 72 6F 72 20 6C 6F 61 64 69 6E 67 20 6F 70 65 72 61 74 69 6E 67 20 73    table.Error loading operating s
000000140   79 73 74 65 6D 00 4D 69  73 73 69 6E 67 20 6F 70 65 72 61 74 69 6E 67 20 73 79 73 74 65 6D 00 00   ystem.Missing operating system..
000000160   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000180   00 00 00 8B FC 1E 57 8B  F5 CB 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ...‹ü.W‹õË......................
0000001A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 94 5E BF 9A 00 00 81 01   ........................”^¿š..□.
0000001C0   01 00 0C FE 7F BA 3F 00  00 00 BC 9F 62 02 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ...þ□°?...¼Ÿb..................
0000001E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 55 AA   ...............................Uª
```

Check 0x1c2, 0x1d2, 0x1e2, 0x1f2 for 0x0c or 0x0b (partition header)

2) PartitionFirstSector = 0x1n6..9 (4 bytes) (n=c,d,e,f) lsb first 3F 00 00 00
PartitionFirstSector = 0x0000003F (x512 = byte 7E00)

3) PartitionSize = 0x1nA (4 bytes) lsb first BC 9F 62 02
PartitionSize = 0x02629FBC

4) lba=PartitionFirstSector (0x3F = .63)
Read this sector - it contains the BPB information.

```
           00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   EB 58 90 4D 53 57 49 4E  34 2E 31 00 02 20 20 00 02 00 00 00 00 F8 00 00 3F 00 FF 00 3F 00 00 00   ëX□MSWIN4.1..  ......ø..?.ÿ.?...
000000020   BC 9F 62 02 28 26 00 00  00 00 00 00 02 00 00 00 01 00 06 00 00 00 00 00 00 00 00 00 00 00 00 00   ¼Ÿb.(&.........................
000000040   80 00 29 F1 12 21 22 4D  50 33 20 56 4F 4C 55 4D 45 20 46 41 54 33 32 20 20 20 FA 33 C9 8E D1 BC   □.)ñ.!"MP3 VOLUME FAT32   ú3É□Ñ¼
000000060   F8 7B 8E C1 BD 78 00 C5  76 00 1E 56 16 55 BF 22 05 89 7E 00 89 4E 02 B1 0B FC F3 A4 8E D9 BD 00   ø{□Á½x.Åv..V.U¿".‰~.‰N.±.üó¤□Ù½.
000000080   7C C6 45 FE 0F 8B 46 18  88 45 F9 38 4E 40 7D 25 8B C1 99 BB 00 07 E8 97 00 72 1A 83 EB 3A 66 A1   |ÆEþ.‹F.ˆEù8N@}%‹Á™».è—.r.ƒë:f¡
0000000A0   1C 7C 66 3B 07 8A 57 FC  75 06 80 CA 02 88 56 02 80 C3 10 73 ED BF 02 00 83 7E 16 00 75 45 8B 46   .|f;.ŠWüu.□Ê.ˆV.□Ã.sí¿..ƒ~..uE‹F
0000000C0   1C 8B 56 1E B9 03 00 49  40 75 01 42 BB 00 7E E8 5F 00 73 26 B0 F8 4F 74 1D 8B 46 32 33 D2 B9 03   .‹V.¹..I@u.B».~è_.s&°øOt.‹F23Ò¹.
0000000E0   00 3B C8 77 1E 8B 76 0E  3B CE 73 17 2B F1 03 46 1C 13 56 1E EB D1 73 0B EB 27 83 7E 2A 00 77 03   .;Èw.‹v.;Îs.+ñ.F..V.ëÑs.ë'ƒ~*.w.
000000100   E9 FD 02 BE 7E 7D AC 98  03 F0 AC 84 C0 74 17 3C FF 74 09 B4 0E BB 07 00 CD 10 EB EE BE 81 7D EB   éý.¾~}¬˜.ð¬„Àt.<ÿt.´.»..Í.ëî¾□}ë
000000120   E5 BE 7F 7D EB E0 98 CD  16 5E 1F 66 8F 04 CD 19 41 56 66 6A 00 52 50 06 53 6A 01 6A 10 8B F4 60   å¾□}ëà˜Í.^.f□.Í.AVfj.RP.Sj.j.‹ô`
000000140   80 7E 02 0E 75 04 B4 42  EB 1D 91 92 33 D2 F7 76 18 91 F7 76 18 42 87 CA F7 76 1A 8A F2 8A E8 C0   □~..u.´Bë.'’3Ò÷v.‘÷v.B‡Ê÷v.ŠòŠèÀ
000000160   CC 02 0A CC B8 01 02 8A  56 40 CD 13 61 8D 64 10 5E 72 0A 40 75 01 42 03 5E 0B 49 75 B4 C3 03 18   Ì..Ì¸..ŠV@Í.a□d.^r.@u.B.^.Iu´Ã..
000000180   01 27 0D 0A 49 6E 76 61  6C 69 64 20 73 79 73 74 65 6D 20 64 69 73 6B FF 0D 0A 44 69 73 6B 20 49   .'..Invalid system diskÿ..Disk I
0000001A0   2F 4F 20 65 72 72 6F 72  FF 0D 0A 52 65 70 6C 61 63 65 20 74 68 65 20 64 69 73 6B 2C 20 61 6E 64   /O errorÿ..Replace the disk, and
0000001C0   20 74 68 65 6E 20 70 72  65 73 73 20 61 6E 79 20 6B 65 79 0D 0A 00 00 00 49 4F 20 20 20 20 20 20    then press any key.....IO
0000001E0   53 59 53 4D 53 44 4F 53  20 20 20 53 59 53 7E 01 00 57 49 4E 42 4F 4F 54 20 53 59 53 00 00 55 AA   SYSMSDOS   SYS~..WINBOOT SYS..Uª
```

5) Check bytes .510 and .511 = 0x55 and 0xAA

6) BPB_BytsPerSec = bytes 0x0B and 0x0C (.11 and .12) lsb first 00 02
BPB_BytsPerSec = 0x0200 (.512)

7) BPB_NumFats = byte 0x10 (.16) 02
BPB_NumFats = 2

8) BPB_FatSz32=0x3f..42 x2 (.36-39) lsb first (=bytes 7E24..27) 28 26 00 00
BPB_FatSz32= 0x2628 x 2 = 0x4C50

9) BPB_RsvdSecCnt=bytes 0x0E and 0x0F (.14& .15) lsb first 20 00
BPB_RsvdSecCnt = 0x20

10) Calculate the first data sector.

FirstDataSector
= BPB_FatSz32 + BPB_RsvdSecCnt + PartitionFirstSector
= 0x4C50 + 0x20 + 0x3F
= 0x4CAF (.19631)

11) BPB_SecPerCluster=byte 0x0D (.13) 20
BPB_SecPerCluster = 0x20 = .64 sectors = 32K cluster size (!)

12) Root_dir_cluster=bytes 0x2c..f lsb first 02 00 00 00
Root_dir_cluster=0x02

13) FirstDataSector (lba) = 0x4CAF (.19631)

This is the first sector of the root directory.

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   4D 50 33 53 20 20 20 20  20 20 20 08 00 00 00 00 00 00 00 00 00 00 A8 AC B5 2A 00 00 00 00 00 00   MP3S       ..........¨¬µ*......
000000020   52 45 43 59 43 4C 45 44  20 20 20 16 00 21 DA 6B 2F 29 BA 2A 01 00 DB 6B 2F 29 72 91 00 00 00 00   RECYCLED   ..!Úk/)°*..Ûk/)r'....
000000040   41 4C 00 69 00 6C 00 6C  00 69 00 0F 00 0C 61 00 6E 00 20 00 41 00 78 00 65 00 00 00 00 00 FF FF   AL.i.l.l.i....a.n. .A.x.e.....ÿÿ
000000060   4C 49 4C 4C 49 41 7E 31  20 20 20 10 00 46 08 A5 2F 29 BA 2A 01 00 09 A5 2F 29 73 91 00 00 00 00   LILLIA~1   ..F.¥/)°*...¥/)s'....
000000080   41 47 00 61 00 62 00 72  00 69 00 0F 00 E6 65 00 6C 00 6C 00 65 00 00 00 FF FF 00 00 FF FF FF FF   AG.a.b.r.i...æe.l.l.e...ÿÿ..ÿÿÿÿ
0000000A0   47 41 42 52 49 45 7E 31  20 20 20 10 00 19 6E A5 2F 29 BA 2A 01 00 6F A5 2F 29 74 91 00 00 00 00   GABRIE~1   ...n¥/)°*..o¥/)t'....
0000000C0   41 49 00 72 00 6F 00 6E  00 20 00 0F 00 E8 6D 00 61 00 69 00 64 00 65 00 6E 00 00 00 00 00 FF FF   AI.r.o.n. ...èm.a.i.d.e.n.....ÿÿ
0000000E0   49 52 4F 4E 4D 41 7E 31  20 20 20 10 00 4E B2 A5 2F 29 BA 2A 01 00 B3 A5 2F 29 75 91 00 00 00 00   IRONMA~1   ..N²¥/)°*..³¥/)u'....
000000100   41 42 00 69 00 6C 00 6C  00 79 00 0F 00 05 20 00 4A 00 6F 00 65 00 6C 00 00 00 00 00 FF FF FF FF   AB.i.l.l.y.... .J.o.e.l.....ÿÿÿÿ
000000120   42 49 4C 4C 59 4A 7E 31  20 20 20 10 00 AB DD A6 2F 29 BA 2A 01 00 E0 A6 2F 29 76 91 00 00 00 00   BILLYJ~1   ..«Ý¦/)°*..à¦/)v'....
000000140   41 4D 00 61 00 72 00 69  00 6C 00 0F 00 2D 6C 00 69 00 6F 00 6E 00 00 00 FF FF 00 00 FF FF FF FF   AM.a.r.i.l...-l.i.o.n...ÿÿ..ÿÿÿÿ
000000160   4D 41 52 49 4C 4C 7E 31  20 20 20 10 00 2C 28 A7 2F 29 BA 2A 01 00 29 A7 2F 29 77 91 00 00 00 00   MARILL~1   ..,(§/)°*..)§/)w'....
000000180   41 53 00 61 00 76 00 61  00 67 00 0F 00 D9 65 00 20 00 47 00 61 00 72 00 64 00 00 00 65 00 6E 00   AS.a.v.a.g...Ùe. .G.a.r.d...e.n.
0000001A0   53 41 56 41 47 45 7E 32  20 20 20 10 00 AD 47 AF 3C 2A BA 2A 04 00 48 AF 3C 2A DC 08 00 00 00 00   SAVAGE~2   ..-G¯<*°*..H¯<*Ü.....
0000001C0   46 46 41 53 54 55 4E 20  46 46 4C 22 18 6D 6F 7F BB 2A BC 2A 10 00 0A B7 BC 2A 04 00 00 60 00 00   FFASTUN FFL".mo□»*¼*...·¼*...`..
0000001E0   41 4F 00 7A 00 7A 00 79  00 00 00 0F 00 3D FF FF FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF   AO.z.z.y.....=ÿÿÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
```

14) Locate a directory or file using a search/select procedure.

First character of file name:

Any valid ASCII character dictates a valid allocated entry.
0xE5 This entry has been deleted. It is not allocated and is freely available.
0x00 As above and there are no further entries.
0x05 Allocated - first char is 0xE5

Short entry bit allocations are listed below.

Name	Offset	Size	Description
Name	0x00	8	DOS file name * See above for Char[0] implications
Extension	0x08	3	DOS File extension
Attributes	0x0B	1	0x01 ReadOnly 0x02 Hidden 0x04 System 0x08 Volume_ID 0x10 Directory 0x20 Archive
NT flags	0x0C	2	Ignore these
Time	0x0E	2	Time of when the file was created
Date	0x10	2	Date of when the file was created
Last accessed	0x12	2	Date of when the file was last accessed
Cluster high word	0x14	2	High word of the start cluster number
Last write time	0x16	2	Time when the file was last written
Last write date	0x18	2	Date when the file was last written
Cluster low word	0x1A	2	Low word of the start cluster number
File size in bytes	0x1C	2	Size of the file in bytes

This is a sector taken from within the root directory list.

```
          00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   46 46 41 53 54 55 4E 20  46 46 41 22 18 92 70 7F BB 2A BC 2A 10 00 E7 A6 BC 2A 03 00 1F 10 00 00   FFASTUN FFA".'p□»*¼*..ç¦¼*......
000000020   54 45 44 53 54 45 53 54  4C 53 54 20 00 C6 F8 AE BC 2A BC 2A 10 00 B2 B1 B7 2A 0A 00 90 01 01 00   TEDSTESTLST .Æø®¼*¼*..²±·*..□...
000000040   42 61 00 63 00 6B 00 73  00 00 00 0F 00 C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF   Ba.c.k.s.....Àÿÿÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
000000060   01 4D 00 6F 00 76 00 69  00 65 00 0F 00 C0 20 00 73 00 6F 00 75 00 6E 00 64 00 00 00 74 00 72 00   .M.o.v.i.e...À .s.o.u.n.d...t.r.
000000080   4D 4F 56 49 45 53 7E 31  20 20 20 10 00 0C 88 53 64 2A BA 2A 05 00 89 53 64 2A C3 00 00 00 00 00   MOVIES~1   ...ˆSd*º*..‰Sd*Ã.....
0000000A0   42 6D 00 70 00 33 00 00  00 FF FF 0F 00 8A FF FF FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF   Bm.p.3...ÿÿ..Šÿÿÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
0000000C0   01 45 00 50 00 54 00 69  00 6D 00 0F 00 8A 69 00 6E 00 67 00 52 00 69 00 66 00 00 00 66 00 2E 00   .E.P.T.i.m...Ši.n.g.R.i.f...f...
0000000E0   45 50 54 49 4D 49 7E 31  4D 50 33 00 00 6E E8 AC B5 2A BA 2A 06 00 C7 61 B3 2A 1F 00 11 59 10 00   EPTIMI~1MP3..nè¬µ*º*..Ça³*...Y..
000000100   43 6E 00 64 00 20 00 49  00 6E 00 0F 00 FE 73 00 65 00 63 00 74 00 2E 00 6D 00 00 00 70 00 33 00   Cn.d. .I.n...þs.e.c.t...m...p.3.
000000120   02 74 00 77 00 65 00 65  00 6E 00 0F 00 FE 20 00 41 00 6E 00 67 00 65 00 6C 00 00 00 20 00 61 00   .t.w.e.e.n...þ .A.n.g.e.l... .a.
000000140   01 50 00 61 00 50 00 61  00 20 00 0F 00 FE 52 00 6F 00 41 00 63 00 48 00 20 00 00 00 42 00 65 00   .P.a.P.a. ...þR.o.A.c.H. ...B.e.
000000160   50 41 50 41 52 4F 7E 31  4D 50 33 00 00 56 96 02 A5 2A BA 2A 08 00 22 08 A5 2A 02 00 31 3F 39 00   PAPARO~1MP3..V–.¥*º*.."¥*..1?9.
000000180   42 2E 00 6D 00 70 00 33  00 00 00 0F 00 A5 FF FF FF FF FF FF FF FF FF FF FF FF 00 00 FF FF FF FF   B..m.p.3.....¥ÿÿÿÿÿÿÿÿÿÿ..ÿÿÿÿ
0000001A0   01 4B 00 69 00 74 00 74  00 79 00 0F 00 A5 20 00 2D 00 20 00 52 00 61 00 76 00 00 00 65 00 6E 00   .K.i.t.t.y...¥ .-. .R.a.v...e.n.
0000001C0   4B 49 54 54 59 2D 7E 31  4D 50 33 20 00 83 3D 09 A5 2A B5 2A 08 00 16 0A A5 2A A3 00 48 FE 13 00   KITTY-~1MP3 .ƒ=.¥*µ*....¥*£.Hþ..
0000001E0   43 20 00 46 00 75 00 63  00 6B 00 0F 00 16 20 00 4F 00 66 00 66 00 2E 00 6D 00 00 00 70 00 33 00   C .F.u.c.k.... .O.f.f...m...p.3.
```

eg For file TEDSTEST.LST

Name	0x0B	0x15	0x14	0x1B	0x1A	0x1F	0x1E	0x1D	0x1C
	Attr	MSB.....LSB				MSB.....LSB			
TESTESTLST	0x20	0x00	0x10	0x00	0x0A	0x00	0x01	0x01	0x90

Attributes = Archive

Cluster -> lba

*Lba =((CLUSTER-Root_dir_cluster) * BPB_SecPerCluster) + FirstDataSector*
*= (0x10000A – 0x02) * 0x20 + 0x4CAF*
*= 0x100008 * 0x20 + 0x4CAF*
= 0x2004DAF (.33574319)

FileSize = 0x010190 = .65936 (correctly shown as 65Kb)

15) lba=0x2004DAF (.33574319)
This is where the data starts. Read BPB_SecPerCluster sectors - this is one cluster (32K in this worked example)
The first sector is shown below (this is an MPLAB generated assembler list file.)

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   4D 50 41 53 4D 20 30 32  2E 37 30 20 52 65 6C 65 61 73 65 64 20 20 20 20 20 20 20 20 54 45 44   MPASM 02.70 Released         TED
000000020   53 54 45 53 54 2E 41 53  4D 20 20 20 35 2D 32 33 2D 32 30 30 31 20 20 32 32 3A 31 33 3A 33 35 20   STEST.ASM   5-23-2001  22:13:35
000000040   20 20 20 20 20 20 20 20  50 41 47 45 20 20 31 0D 0A 0D 0A 0D 0A 4C 4F 43 20 20 4F 42 4A 45 43 54           PAGE  1......LOC  OBJECT
000000060   20 43 4F 44 45 20 20 20  20 20 4C 49 4E 45 20 53 4F 55 52 43 45 20 54 45 58 54 0D 0A 20 20 56 41    CODE     LINE SOURCE TEXT..  VA
000000080   4C 55 45 0D 0A 0D 0A 20  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30 30   LUE....                      000
0000000A0   30 31 20 3B 0D 0A 20 20  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30 30 30   01 ;..                      0000
0000000C0   32 20 20 20 20 20 20 20  20 20 54 49 54 4C 45 20 22 45 64 73 54 65 73 74 20 3C 52 4F 4D 7A 61 70   2         TITLE "EdsTest <ROMzap
0000000E0   20 54 65 6D 70 6C 61 74  65 20 56 32 37 30 34 30 31 3E 22 0D 0A 20 20 20 20 20 20 20 20 20 20 20    Template V270401>"..
000000100   20 20 20 20 20 20 20 20  20 20 20 30 30 30 30 33 20 3B 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20            00003 ;..
000000120   20 20 20 20 20 20 20 20  20 20 30 30 30 30 34 20 20 20 20 20 20 20 20 20 4C 49 53 54 20 50 20 3D           00004         LIST P =
000000140   20 31 36 46 38 37 37 0D  0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30    16F877..                     0
000000160   30 30 30 35 20 3B 0D 0A  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30   0005 ;..                     00
000000180   30 30 36 20 20 20 20 20  20 20 20 20 49 4E 43 4C 55 44 45 20 22 50 31 36 66 38 37 37 2E 69 6E 63   006         INCLUDE "P16f877.inc
0000001A0   22 0D 0A 20 20 20 20 20  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30 30 30 31 20   ".                        00001
0000001C0   20 20 20 20 20 20 20 4C  49 53 54 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20          LIST..
0000001E0   20 20 20 30 30 30 30 32  20 3B 20 50 31 36 46 38 37 37 2E 49 4E 43 20 20 53 74 61 6E 64 61 72 64     00002 ; P16F877.INC  Standard
```

The last sector of this cluster is at 0x2004DAF + BPB_SecPerCluster -1 = 0x2004DCE (.33574350) and is shown below...

```
000000000   46 58 0D 0A 30 30 35 33  20 20 20 30 30 30 38 20 20 20 20 20 20 20 20 20 20 20 30 30 32 35 39 20   FX..0053   0008           00259
000000020   20 20 20 20 20 20 20 20  52 45 54 55 52 4E 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20           RETURN..
000000040   20 20 20 20 20 20 30 30  32 36 30 20 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20         00260 ..
000000060   20 20 20 20 30 30 32 36  31 20 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20       00261 ..
000000080   20 20 30 30 32 36 32 20  3B 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D     00262 ;-----------------------
0000000A0   2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D   -------------------------------
0000000C0   2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 0D 0A 20 20 20 20 20 20 20 20 20   ----------------------..
0000000E0   20 20 20 20 20 20 20 20  20 20 20 20 20 30 30 32 36 33 20 3B 20 57 72 69 74 65 20 64 61 74 61 20               00263 ; Write data
000000100   57 4F 52 44 20 74 6F 20  44 72 69 76 65 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20   WORD to Drive..
000000120   20 20 20 20 20 30 30 32  36 34 20 3B 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D       00264 ;-----------------------
000000140   2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D   -------------------------------
000000160   2D 2D 2D 2D 2D 2D 2D 2D  2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 2D 0D 0A 30 30 35 34 20 20   ------------------------..0054
000000180   20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20 30 30 32 36 35 20 57 52 49 54 45 57 4F 52 44 3A                 00265 WRITEWORD:
0000001A0   0D 0A 20 20 20 20 20 20  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30 32 36 36 20 3B 55   ..                      00266 ;U
0000001C0   73 65 73 20 63 73 2C 61  64 72 2C 64 61 74 0D 0A 30 30 35 34 20 20 20 32 30 31 46 20 20 20 20 20   ses cs,adr,dat..0054   201F
0000001E0   20 20 20 20 20 20 30 30  32 36 37 20 20 20 20 20 20 20 20 20 43 41 4C 4C 20 20 20 53 45 54 41         00267         CALL    SETA
```

We can use this to check that the next cluster calculated points to the correct data.

16) Determine where in the FAT table we need to jump to in order to get the address of the next cluster...

FAT table entry point derivation

```
FATOffset=CLUSTER*4
FatSecNum = BPB_RsvdSecCnt + (FATOffset / BPB_BytesPerSec)
FatEntryOffset = remainder of (FatOffset / BPB_BytsPerSec)
Note: .128 = 0x80

FatSecNum       = 0x20 + (CLUSTER * 4 / .512)
                = 0x20 + (CLUSTER / .128)
                = 0x20 + (0x10000A / .128)
                = 0x20 + 0x2000 (.8192)
                = 0x2020 (.8224)
FatEntryOffset
                = rem(CLUSTER * 4 / .512)
                = rem(CLUSTER / .128)
                = rem(0x10000A >> 7)
                = rem(0x2000)
                0x2000 * 0x80 (.128)
                = 0x2000 << 7
                = 0x100000
                0x10000A - 0x100000
                = 0x0A
```

Notes: The FatSecNum is relative to the sector 0 of the FAT volume, so add 0x3F (PartitionFirstSector)
Also note that the offset is the offset in entries, not bytes. 0x0A is the 0x0Ath 32bit entry.

Sector 0x2020+0x3F = 0x205F (.8287) data shows the following...

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   00 00 00 00 00 00 00 00  FF FF FF 0F FF FF FF 0F 00 00 00 00 00 00 00 00 00 00 00 00 27 00 10 00   ........ÿÿÿ.ÿÿÿ.............'...
000000020   FF FF FF 0F FF FF FF 0F  0B 00 10 00 0C 00 10 00 0D 00 10 00 0E 00 10 00 FF FF FF 0F 00 00 00 00   ÿÿÿ.ÿÿÿ.................ÿÿÿ.....
000000040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000080   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF 0F   ..............................ÿÿÿ.
0000000A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000000C0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000000E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000100   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000120   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000140   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000160   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000180   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001C0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
```

The 0x0A (.10th) entry is at byte offset 0x0A * 4 = 0x28 (.40)
From the table above...

0x28 0x29 0x2A 0x2B
0x0B 0x00 0x10 0x00 = cluster number 0x0010000B.

Cluster -> lba

*Lba = ((CLUSTER-2) * BPB_SecPerCluster) + FirstDataSector*
*= (0x10000B-2) * 0x20) + 0x4CAF*
*= (0x100009 * 0x20) + 0x4CAF*
= 0x100029 + 0x4CAF
= 0x2004DCF (.33574351)

This is the start sector of the second cluster of information pertaining to this file.

```
000000000   44 44 52 45 53 53 0D 0A  20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 30 30   DDRESS..                      00
000000020   32 36 38 20 20 20 20 20  20 20 20 20 3B 4D 53 42 2F 4C 53 42 20 69 73 20 73 77 61 70 70 65 64 20   268         ;MSB/LSB is swapped
000000040   28 73 65 65 20 77 72 69  74 65 53 65 63 74 6F 72 20 66 75 6E 63 74 69 6F 6E 29 0D 0A 30 30 35 35   (see writeSector function)..0055
000000060   20 20 20 31 36 38 33 20  31 33 30 33 20 20 20 20 20 20 30 30 32 36 39 20 20 20 20 20 20 20 20 20      1683 1303      00269
000000080   42 41 4E 4B 53 45 4C 20  54 52 49 53 41 0D 0A 4D 65 73 73 61 67 65 5B 33 30 32 5D 3A 20 52 65 67   BANKSEL TRISA..Message[302]: Reg
0000000A0   69 73 74 65 72 20 69 6E  20 6F 70 65 72 61 6E 64 20 6E 6F 74 20 69 6E 20 62 61 6E 6B 20 30 2E 20   ister in operand not in bank 0.
0000000C0   20 45 6E 73 75 72 65 20  74 68 61 74 20 62 61 6E 6B 20 62 69 74 73 20 61 72 65 20 63 6F 72 72 65    Ensure that bank bits are corre
0000000E0   63 74 2E 0D 0A 30 30 35  37 20 20 20 30 31 38 36 20 20 20 20 20 20 20 20 20 20 20 20 30 30 32 37 30   ct...0057   0186           00270
000000100   20 20 20 20 20 20 20 20  20 43 4C 52 46 20 20 20 20 54 52 49 53 42 0D 0A 4D 65 73 73 61 67 65 5B            CLRF    TRISB..Message[
000000120   33 30 32 5D 3A 20 52 65  67 69 73 74 65 72 20 69 6E 20 6F 70 65 72 61 6E 64 20 6E 6F 74 20 69 6E   302]: Register in operand not in
000000140   20 62 61 6E 6B 20 30 2E  20 20 45 6E 73 75 72 65 20 74 68 61 74 20 62 61 6E 6B 20 62 69 74 73 20    bank 0.  Ensure that bank bits
000000160   61 72 65 20 63 6F 72 72  65 63 74 2E 0D 0A 30 30 35 38 20 20 20 30 31 38 38 20 20 20 20 20 20 20   are correct...0058   0188
000000180   20 20 20 20 30 30 32 37  31 20 20 20 20 20 20 20 20 20 43 4C 52 46 20 20 20 20 54 52 49 53 44 0D       00271         CLRF    TRISD.
0000001A0   0A 30 30 35 39 20 20 20  31 32 38 33 20 31 33 30 33 20 20 20 20 20 20 20 30 30 32 37 32 20 20 20 20   .0059   1283 1303      00272
0000001C0   20 20 20 20 20 42 41 4E  4B 53 45 4C 20 50 4F 52 54 41 0D 0A 20 20 20 20 20 20 20 20 20 20 20 20        BANKSEL PORTA..
0000001E0   20 20 20 20 20 20 20 20  20 20 30 30 32 37 33 20 20 20 20 20 20 20 20 20 0D 0A 30 30 35 42 20 20            00273         ..005B
```

We can see that this is the correct cluster as the data in it follows on from the data shown above (last sector of first data cluster).

17) End of file marker.
If we follow the cluster chain to the end, the following cluster chain is shown.

.1048586 0x10000A
.1048587 0x10000B

.1048588 0x10000C
.1048589 0x10000D
.1048590 0x10000E

These are consecutive and show that the drive and the file are both defragmented.
When traversing the FAT table for the final entry, we get the following data.

FAT table entry point derivation

*FATOffset=CLUSTER*4*
FatSecNum = BPB_RsvdSecCnt + (FATOffset / BPB_BytesPerSec)
FatEntryOffset = remainder of (FatOffset / BPB_BytsPerSec)
Note: .128 = 0x80

FatSecNum
*= 0x20 + (CLUSTER * 4 / .512)*
= 0x20 + (CLUSTER / .128)
= 0x20 + (0x10000E / .128)
= 0x20 + 0x2000 (.8192)
= 0x2020 (.8224)

*FatEntryOffset = CLUSTER - (int(CLUSTER/128) * .128)*
FatEntryOffset
*= remainder of (CLUSTER * 4 / .512)*
temp = CLUSTER / .128
temp = 0x10000E >> 7
temp = 0x2000

*0x2000 * 0x80 (.128)*
= 0x2000 << 7
= 0x100000
0x10000E - 0x100000
= 0x0E

Notes: The FatSecNum is relative to the sector 0 of the FAT volume, so add 0x3F (PartitionFirstSector)
Also note that the offset is the offset in entries, not bytes. 0x0A is the 0x0Ath 32bit entry.

Sector 0x2020+0x3F = 0x205F (.8287) data shows the following...

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   00 00 00 00 00 00 00 00  FF FF FF 0F FF FF FF 0F 00 00 00 00 00 00 00 00 00 00 00 00 27 00 10 00   ........ÿÿÿ.ÿÿÿ.............'...
000000020   FF FF FF 0F FF FF FF 0F  0B 00 10 00 0C 00 10 00 0D 00 10 00 0E 00 10 00 FF FF FF 0F 00 00 00 00   ÿÿÿ.ÿÿÿ.................ÿÿÿ.....
000000040   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000060   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000080   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF 0F   ..............................ÿÿÿ.
0000000A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000000C0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000000E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000100   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000120   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000140   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000160   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
000000180   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001A0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001C0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
0000001E0   00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................................
```

The 0x0E (.14th) entry is at byte offset 0x0E * 4 = 0x38 (.56)
From the table above...

```
0x38    0x39    0x3A    0x3B
0xFF    0xFF    0xFF    0x0F = cluster number 0x0FFFFFFF.
```

0x0FFFFFFF = End of file marker
0x0FFFFFF7 = Bad sector - do not use

Note: The upper 4 bits of the end marker should be ignored when reading, and preserved when writing.

18) Long File Names

Dump of sector 0x4CAF

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000000   02 73 00 68 00 2D 00 4E  00 61 00 0F 00 16 7A 00 69 00 20 00 50 00 75 00 6E 00 00 00 6B 00 73 00   .s.h.-.N.a....z.i. .P.u.n...k.s.
000000020   01 31 00 32 00 2D 00 50  00 6C 00 0F 00 16 61 00 6E 00 65 00 74 00 20 00 54 00 00 00 72 00 61 00   .1.2.-.P.l....a.n.e.t. .T...r.a.
000000040   31 32 2D 50 4C 41 7E 31  4D 50 33 20 00 BC 1C B0 A8 2A BC 2A 0A 00 A6 B0 A8 2A 02 00 C6 EA 0E 00   12-PLA~1MP3 .¼.°¨*¼*..¦°¨*..Æê..
000000060   44 20 00 66 00 69 00 6C  00 65 00 0F 00 A6 20 00 6E 00 61 00 6D 00 65 00 00 00 00 00 FF FF FF FF   D .f.i.l.e...¦ .n.a.m.e.....ÿÿÿÿ
000000080   03 63 00 75 00 6C 00 6F  00 75 00 0F 00 A6 73 00 6C 00 79 00 20 00 6C 00 6F 00 00 00 6E 00 67 00   .c.u.l.o.u...¦s.l.y. .l.o...n.g.
0000000A0   02 65 00 20 00 77 00 69  00 74 00 0F 00 A6 68 00 20 00 61 00 20 00 72 00 69 00 00 00 64 00 69 00   .e. .w.i.t...¦h. .a. .r.i...d.i.
0000000C0   01 54 00 68 00 69 00 73  00 20 00 0F 00 A6 69 00 73 00 20 00 61 00 20 00 66 00 00 00 69 00 6C 00   .T.h.i.s. ...¦i.s. .a. .f...i.l.
0000000E0   54 48 49 53 49 53 7E 31  20 20 20 20 00 C6 F8 AE BC 2A BC 2A 10 00 B2 B1 B7 2A 0A 00 90 01 01 00   THISIS~1    .Æø®¼*¼*..²±·*..□...
000000100   E5 46 41 53 54 55 4E 54  46 46 4F 22 00 A2 E6 A6 BC 2A BC 2A 10 00 E7 A6 BC 2A 09 00 00 40 00 00   åFASTUNTFFO".¢æ¦¼*¼*..ç¦¼*...@..
000000120   E5 51 4D 41 50 20 20 20  44 41 54 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 60 F0 D8 01 00 00   åQMAP   DAT .....□*.. .’!`ðØ...
000000140   E5 51 4D 41 50 31 36 20  45 58 45 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 61 F0 15 F5 07 00   åQMAP16 EXE .....□*.. .’!að.õ..
000000160   E5 51 4D 41 50 33 32 20  45 58 45 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 81 F0 B0 FC 06 00   åQMAP32 EXE .....□*.. .’!□ð°ü..
000000180   E5 45 41 44 4D 45 20 20  54 58 54 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 9D F0 A8 1C 00 00   åEADME  TXT .....□*.. .’!□ð¨...
0000001A0   E5 55 4E 44 4F 53 20 20  45 58 45 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 9E F0 F4 C5 00 00   åUNDOS  EXE .....□*.. .’!□ðôÅ..
0000001C0   E5 4E 55 54 49 4C 20 20  45 58 45 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 A2 F0 80 20 00 00   åNUTIL  EXE .....□*.. .’!¢ð□ ..
0000001E0   E5 59 53 20 20 20 20 20  50 49 46 20 00 00 00 00 00 00 00 00 90 2A 05 00 A0 16 92 21 A3 F0 C7 03 00 00   åYS     PIF .....□*.. .’!£ðÇ...
```

The file of interest here is THISIS~1. The long filename of this file is 'This is a file with a ridiculously long file name.'

The bit settings for the standard directory entries is repeated below:

First character of file name:
0xE5 This entry is not allocated and is freely available.
0x00 As above and there are no further entries.
0x05 Allocated - first char is 0xE5

Name	Offset	Size	Description
Name	0x00	8	DOS file name * See above for Char[0] implications
Extension	0x08	3	DOS File extension
Attributes	0x0B	1	0x01 ReadOnly 0x02 Hidden 0x04 System 0x08 Volume_ID 0x10 Directory 0x20 Archive
NT flags	0x0C	2	Ignore these
Time	0x0E	2	Time of when the file was created
Date	0x10	2	Date of when the file was created
Last accessed	0x12	2	Date of when the file was last accessed
Cluster high word	0x14	2	High word of the start cluster number
Last write time	0x16	2	Time when the file was last written
Last write date	0x18	2	Date when the file was last written
Cluster low word	0x1A	2	Low word of the start cluster number
File size in bytes	0x1C	2	Size of the file in bytes

For long filenames, a new directory entry is introduced (DELFN - Directory Entry for Long File Name)

Name	Offset	Size	Description
SeqNumber	0x00	1	This field describes the number of this entry is the sequence of entries which creates the total LFN. If bit 7 is set then it it the last entry in the chain.
Name 1	0x01	10	Chars 1-5 of the LFN sub-entry. Even bytes (2,4,6,8 amd 10) are set to 0x00.
Attributes	0x0B	1	Always set to the impossible value of 0x0F (ReadOnly + System + Hidden + Volume_ID) - this shows that it is a long file name entry - not a file, or directory.
Reserved	0x0C	1	Always set to 0x00
Checksum	0x0D	1	Short DOS filename cecksum - see below
Name 2	0x0E	12	Chars 6-11 of the LFN sub-entry - Odd bytes are set to 0x00
Name 3	0x1C	4	Chars 12-13 of then LFN sub-entry - Odd bytes are set to 0x00

```
           00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F

000000060  44 20 00 66 00 69 00 6C  00 65 00 0F 00 A6 20 00 6E 00 61 00 6D 00 65 00 00 00 00 00 FF FF FF FF   D .f.i.l.e...¦ .n.a.m.e.....ÿÿÿÿ
000000080  03 63 00 75 00 6C 00 6F  00 75 00 0F 00 A6 73 00 6C 00 79 00 20 00 6C 00 6F 00 00 00 6E 00 67 00   .c.u.l.o.u...¦s.l.y. .l.o...n.g.
0000000A0  02 65 00 20 00 77 00 69  00 74 00 0F 00 A6 68 00 20 00 61 00 20 00 72 00 69 00 00 00 64 00 69 00   .e. .w.i.t...¦h. .a. .r.i...d.i.
0000000C0  01 54 00 68 00 69 00 73  00 20 00 0F 00 A6 69 00 73 00 20 00 61 00 20 00 66 00 00 00 69 00 6C 00   .T.h.i.s. ...¦i.s. .a. .f...i.l.
0000000E0  54 48 49 53 49 53 7E 31  20 20 20 20 00 C6 F8 AE BC 2A BC 2A 10 00 B2 B1 B7 2A 0A 00 90 01 01 00   THISIS~1    .Æø®¼*¼*..²±·*..□...
```

The data can be split down as shown below.

Entry	Value	Description
0x00	0x44	The 4th and last DELFN entry for the following DOS filename directory entry.
		Char value 'n' = 0 + (13 * (lower nibble of entry 0x00-1)) = char 0 + (13 * (4-1)) = 39
0x01	' '	Char n+0 (n=39)
0x03	'f'	Char n+1
0x05	'i'	Char n+2
0x07	'l'	Char n+3
0x09	'e'	Char n+4
0x0B	0x0F	This is a DELFN entry.
0x0D	0xA6	Checksum
0x0E	' '	Char n+5
0x10	'n'	Char n+6
0x12	'a'	Char n+7
0x14	'm'	Char n+8
0x16	'e'	Char n+9
0x18	0x00	Char n+10 - null = end of filename
0x1C	0xFF	Char n+11
0x1E	0xFF	Char n+12

Entry	Value	Description
0x00	0x03	The 3rd DELFN entry for the following DOS filename directory entry.
		Char value 'n' = 0 + (13 * (lower nibble of entry 0x00-1)) = char 0 + (13 * (3-1)) = 26
0x01	'c'	Char n+0 (n=26)
0x03	'u'	Char n+1
0x05	'l'	Char n+2
0x07	'o'	Char n+3
0x09	'u'	Char n+4
0x0B	0x0F	This is a DELFN entry.
0x0D	0xA6	Checksum
0x0E	's'	Char n+5
0x10	'l'	Char n+6
0x12	'y'	Char n+7
0x14	' '	Char n+8
0x16	'l'	Char n+9
0x18	'o'	Char n+10 - null = end of filename
0x1C	'n'	Char n+11
0x1E	'g'	Char n+12

Entry	Value	Description
	0x00	0x02 The 2nd DELFN entry for the following DOS filename directory entry.
		Char value 'n' = 0 + (13 * (lower nibble of entry 0x00-1)) = char 0 + (13 * (2-1)) = 13
	0x01	'e' Char n+0 (n=13)
	0x03	' ' Char n+1
	0x05	'w' Char n+2
	0x07	'i' Char n+3
	0x09	't' Char n+4
	0x0B	0x0F This is a DELFN entry.
	0x0D	0xA6 Checksum
	0x0E	'h' Char n+5
	0x10	' ' Char n+6
	0x12	'a' Char n+7
	0x14	' ' Char n+8
	0x16	'r' Char n+9
	0x18	'i' Char n+10 - null = end of filename
	0x1C	'd' Char n+11
	0x1E	'i' Char n+12

Entry	Value	Description
	0x00	0x01 The 1st DELFN entry for the following DOS filename directory entry.
		Char value 'n' = 0 + (13 * (lower nibble of entry 0x00-1)) = char 0 + (13 * (1-1)) = 0
	0x01	'T' Char n+0 (n=0)
	0x03	'h' Char n+1
	0x05	'i' Char n+2
	0x07	's' Char n+3
	0x09	' ' Char n+4
	0x0B	0x0F This is a DELFN entry.
	0x0D	0xA6 Checksum
	0x0E	'i' Char n+5
	0x10	's' Char n+6
	0x12	' ' Char n+7
	0x14	'a' Char n+8
	0x16	' ' Char n+9
	0x18	'f' Char n+10 - null = end of filename
	0x1C	'i' Char n+11
	0x1E	'l' Char n+12

The attribute tells us that this is not a DEFLN entry so we will treat it differently...

Entry	Value	Description
0x00	'T'	Char 0
0x01	'H'	Char 1
0x02	'I'	Char 2
0x03	'S'	Char 3
0x04	'I'	Char 4
0x05	'S'	Char 5
0x06	'~'	Char 6
0x07	'1'	Char 7
0x08	' '	Extension Char 0
0x09	' '	Extension Char 1
0x0A	' '	Extension Char 2
0x0B	0x20	Attributes - Archive bit is set... this is not a DELFN entry
0x14	0x10	Cluster[2]
0x15	0x00	Cluster[3] (msb)
0x1A	0x0A	Cluster[0] (lsb)
0x1B	0x00	Cluster[1]
0x1C	0x90	FileSize[0] (lsb)
0x1D	0x01	FileSize[1]
0x1E	0x01	FileSize[2]
0x1F	0x00	FileSize[3] (msb)

```
            00 01 02 03 04 05 06 07  08 09 0A 0B 0C 0D 0E 0F 10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F
0000000E0   54 48 49 53 49 53 7E 31  20 20 20 20 00 C6 F8 AE BC 2A BC 2A 10 00 B2 B1 B7 2A 0A 00 90 01 01 00   THISIS~1    .Æø®¼*¼*..²±·*..□...
```

Generic algorithms.
---

1) Determining the checksum of the DOS filename.

```
sum=0;
for i=1 to 11
        if (sum and 1) then
                sum = ( sum + 0x80 + (sum / 2) + Filename (i) ) and 0xFF
        else
                sum = ( sum + (sum / 2) + Filename (i) ) and 0xFF
        end if
next
```

For example - Filename = "THISIS~1" gives 0xA6

2) Using the attribute byte to provide entry type determination.

0x01 ReadOnly
0x02 Hidden
0x04 System
0x08 Volume_ID
0x10 Directory
0x20 Archive

```
if (((ATTRIBUTES & 0x0F) = 0x0F) && (Filename[0]!=0xE5) then Found_DELFN_Directory_Entry
if (((ATTRIBUTES & 0x0F)!= 0x0F) && (Filename[0]!=0xE5) then
        if (ATTRIBUTES & 0x18) = 0x00 then Found_File
        if (ATTRIBUTES & 0x18) = 0x10 then Found_Directory
        if (ATTRIBUTES & 0x18) = 0x08 then Found_Volume_ID
end if
```