FAT File System Forensics
pd_image1.dd

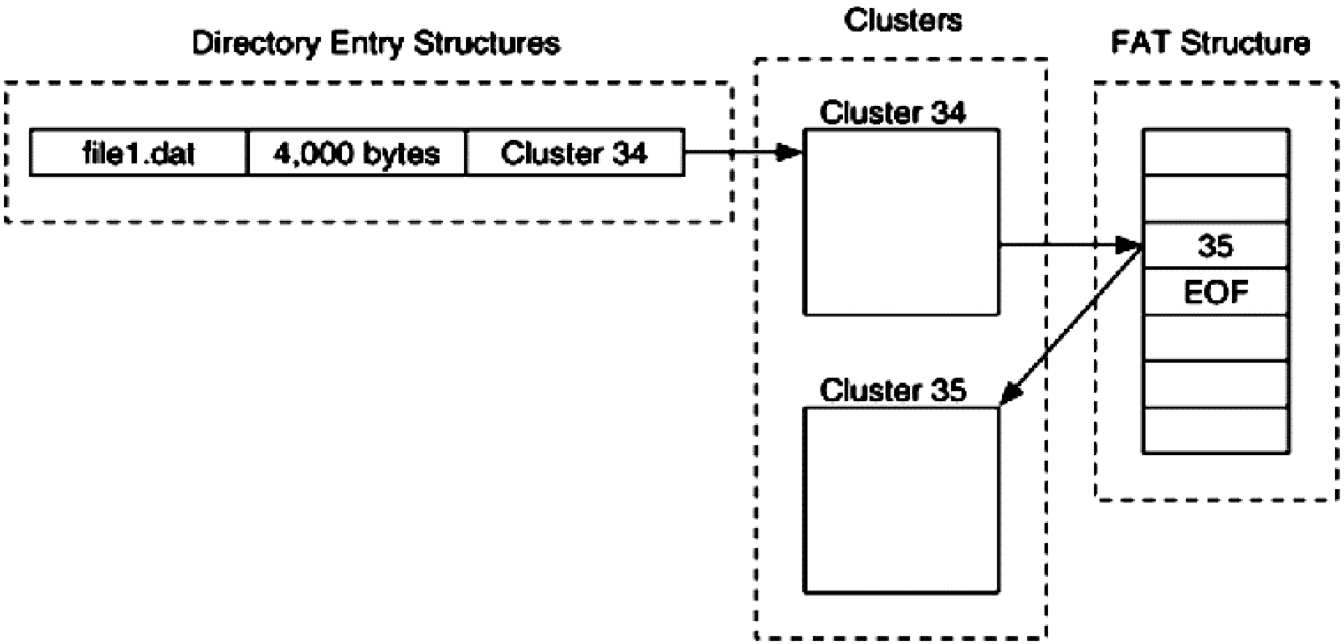
File Allocation Table (FAT)

- The File Allocation Table (FAT) file system is one of the most simple file systems found in common operating systems.

- FAT is the primary file system of the Microsoft DOS and Windows 9x operating systems, but the NT, 2000, and XP line has defaulted to the New Technologies File System (NTFS).

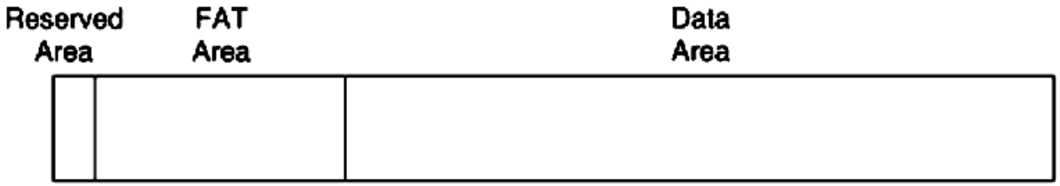
- FAT is supported by all Windows and most Unix operating systems.

- FAT is frequently found in compact flash cards for digital cameras and USB "thumb drives."



- The basic concept of a FAT file system is that each file and directory is allocated a data structure, called **a directory entry**, that contains the file name, size, starting address of the file content, and other metadata.
- File and directory content is stored in data units called **clusters**.
- If a file or directory has allocated more than one cluster, the other clusters are found by using **a structure** that is called **the FAT**.
- The FAT structure is used to identify the next cluster in a file, and it is also used to identify the allocation status of clusters. Therefore it is used in both the content and metadata categories.
- There are three different versions of FAT: FAT12, FAT16, and FAT32. The major difference among them is the size of the entries in the FAT structure.

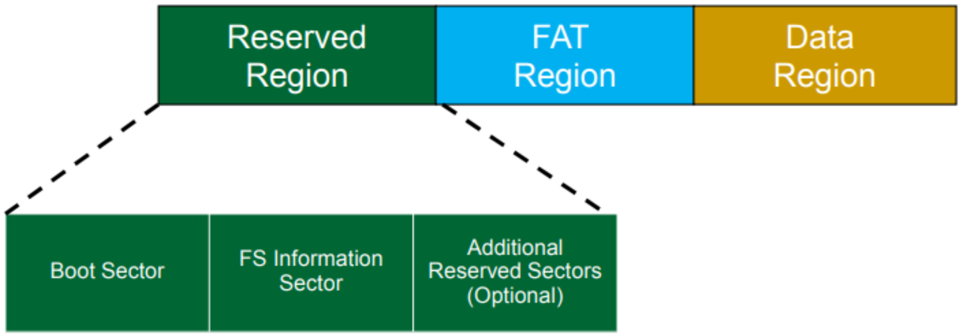
The typical layout of the FAT file system has three physical sections to it



Physical layout of a FAT file system.

The first section is the **reserved area.** In the reserved area are a backup boot sector and a FSINFO data structure. In FAT12 and FAT16 this area is typically only 1 sector in size, but the size is defined in the boot sector. The reserved area starts in sector 0 of the file system, and its size is given in the boot sector.

Reserved Region – Includes the boot sector, the extended boot sector, the file system information sector, and a few other reserved sectors

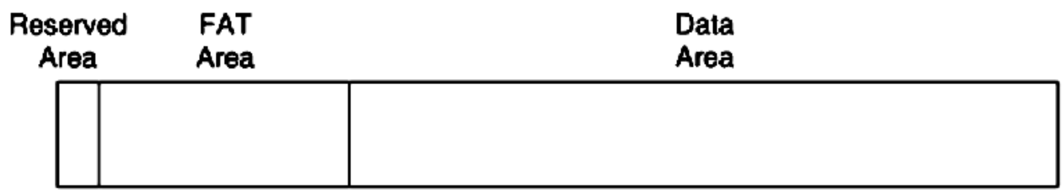


FSInfo sector

FAT32 stores extra information in the FSInfo sector, usually sector 1.

```
Bytes   Content
0-3     0x41615252 - the FSInfo signature
4-483   Reserved
484-487 0x61417272 - a second FSInfo signature
488-491 Free cluster count or 0xffffffff (may be incorrect)
492-495 Next free cluster or 0xffffffff (hint only)
496-507 Reserved
508-511 0xaa550000 - sector signature
```

The typical layout of the FAT file system has three physical sections to it

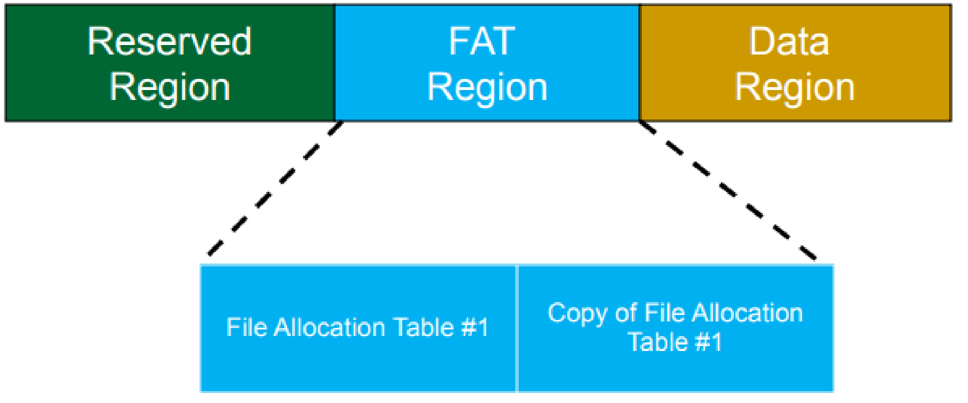


Physical layout of a FAT file system.

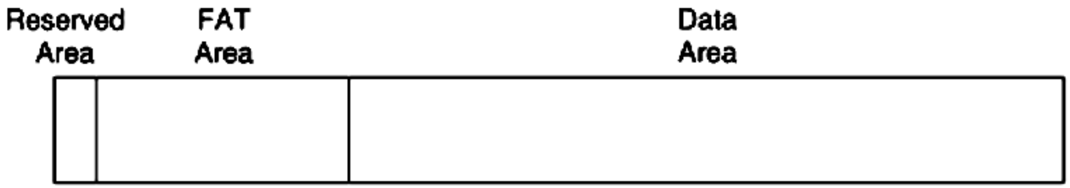
The second section is the **FAT area**, and it contains the primary and backup FAT structures. It starts in the sector following the reserved area.

Its size is calculated by multiplying the number of FAT structures by the size of each FAT; both of these values are given in the boot sector.

The allocation status of each cluster can be determined by looking at the cluster's entry in the FAT. **Entries with a zero value are unallocated and non-zero entries are allocated**. If we wanted to extract the contents of all unallocated clusters, we would read the FAT and extract each cluster with a zero in the table.



The typical layout of the FAT file system has three physical sections to it:

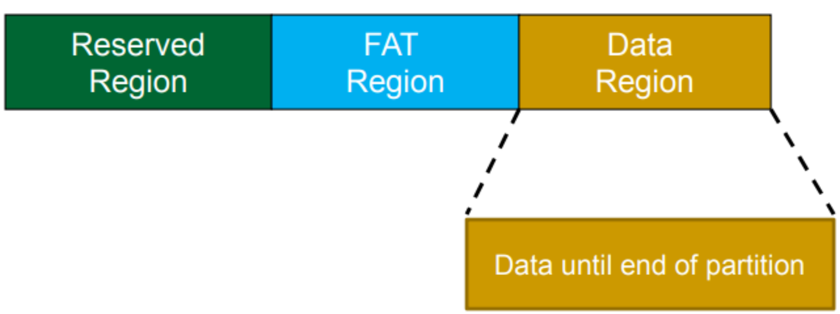


Physical layout of a FAT file system.

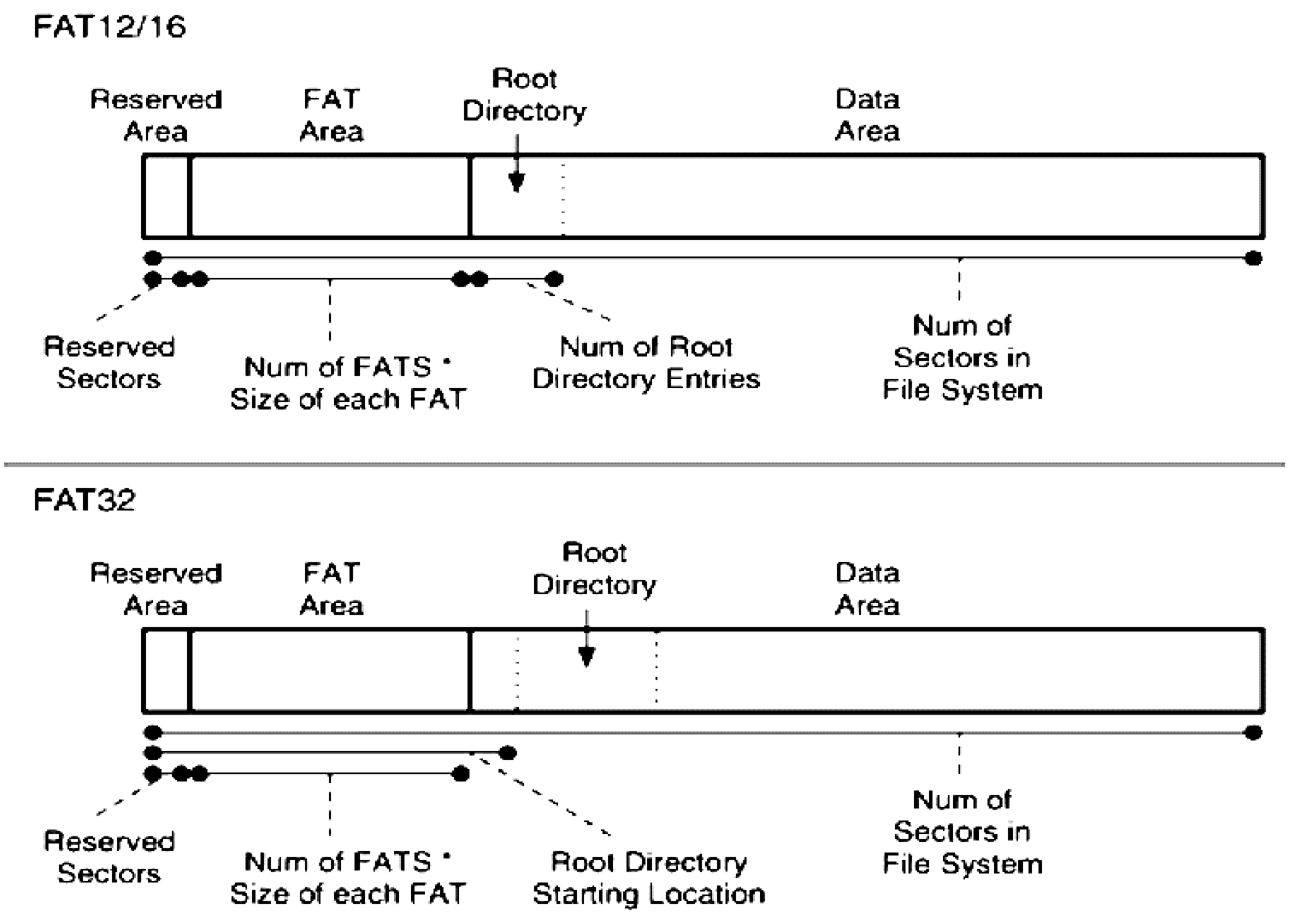
The third section is the **data area**. It contains the clusters that will be allocated to store file and directory content. It begins in the sector after the FAT area.

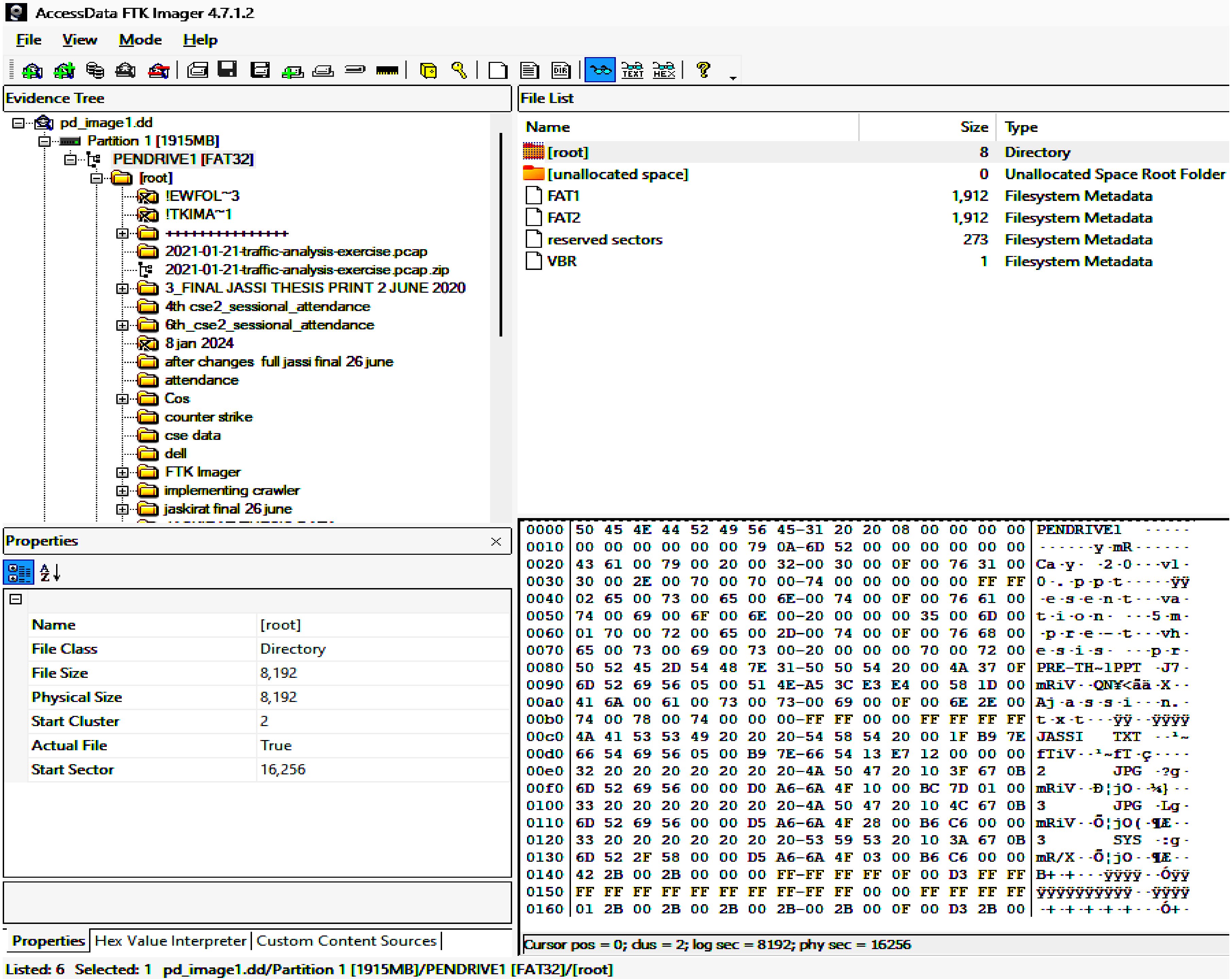
Its size is calculated by subtracting the starting sector address of the data area from the total number of sectors in the file system, which is specified in the boot sector.

Data Region – Using the addresses from the FAT region, contains actual file/directory data



- The layout of the data area is slightly different in FAT12/16 and FAT32.
- In FAT12/16 the beginning of the data area is reserved for the root directory(fixed size in FAT12/16) , but in FAT32 the root directory (dynamic size) can be anywhere in the data area
- The dynamic size and location of the root directory allows FAT32 to adapt to bad sectors in the beginning of the data area and allows the directory to grow as large as it needs to.




AccessData FTK Imager 4.7.1.2
File
View
Mode
Help
Evidence Tree
pd_image1.dd
Partition 1 [1915MB]
PENDRIVE1 [FAT32]
[root]
!EWFOL~3
!TKIMA~1
+++++++++++++++
2021-01-21-traffic-analysis-exercise.pcap
2021-01-21-traffic-analysis-exercise.pcap.zip
3_FINAL JASSI THESIS PRINT 2 JUNE 2020
4th cse2_sessional_attendance
6th_cse2_sessional_attendance
8 jan 2024
after changes full jassi final 26 june
attendance
Cos
counter strike
cse data
dell
FTK Imager
implementing crawler
jaskirat final 26 june
File List
Name
Size
Type
[root]
8
Directory
[unallocated space]
0
Unallocated Space Root Folder
FAT1
1,912
Filesystem Metadata
FAT2
1,912
Filesystem Metadata
reserved sectors
273
Filesystem Metadata
VBR
1
Filesystem Metadata
Properties
Name
[root]
File Class
Directory
File Size
8,192
Physical Size
8,192
Start Cluster
2
Actual File
True
Start Sector
16,256
Properties
Hex Value Interpreter
Custom Content Sources
0000 50 45 4E 44 52 49 56 45-31 20 20 08 00 00 00 00 PENDRIVE1
0010 00 00 00 00 00 00 79 0A-6D 52 00 00 00 00 00 00
0020 43 61 00 79 00 20 00 32-00 30 00 0F 00 76 31 00
0030 30 00 2E 00 70 00 70 00-74 00 00 00 00 00 FF FF
0040 02 65 00 73 00 65 00 6E-00 74 00 0F 00 76 61 00
0050 74 00 69 00 6F 00 6E 00-20 00 00 00 35 00 6D 00
0060 01 70 00 72 00 65 00 2D-00 74 00 0F 00 76 68 00
0070 65 00 73 00 69 00 73 00-20 00 00 00 70 00 72 00
0080 50 52 45 2D 54 48 7E 31-50 50 54 20 00 4A 37 0F PRE-TH~1PPT
0090 6D 52 69 56 05 00 51 4E-A5 3C E3 E4 00 58 1D 00
00a0 41 6A 00 61 00 73 00 73-00 69 00 0F 00 6E 2E 00
00b0 74 00 78 00 74 00 00 00-FF FF 00 00 FF FF FF FF
00c0 4A 41 53 53 49 20 20 20-54 58 54 20 00 1F B9 7E JASSI TXT
00d0 66 54 69 56 05 00 B9 7E-66 54 13 E7 12 00 00 00
00e0 32 20 20 20 20 20 20 20-4A 50 47 20 10 3F 67 0B 2 JPG
00f0 6D 52 69 56 00 00 D0 A6-6A 4F 10 00 BC 7D 01 00
0100 33 20 20 20 20 20 20 20-4A 50 47 20 10 4C 67 0B 3 JPG
0110 6D 52 69 56 00 00 D5 A6-6A 4F 28 00 B6 C6 00 00
0120 33 20 20 20 20 20 20 20-53 59 53 20 10 3A 67 0B 3 SYS
0130 6D 52 2F 58 00 00 D5 A6-6A 4F 03 00 B6 C6 00 00
0140 42 2B 00 2B 00 00 00 FF-FF FF FF 0F 00 D3 FF FF
0150 FF FF FF FF FF FF FF FF-FF FF 00 00 FF FF FF FF
0160 01 2B 00 2B 00 2B 00 2B-00 2B 00 0F 00 D3 2B 00
Cursor pos = 0; clus = 2; log sec = 8192; phy sec = 16256
Listed: 6 Selected: 1 pd_image1.dd/Partition 1 [1915MB]/PENDRIVE1 [FAT32]/[root]

```
┌──(kali㉿kali)-[~/pdbackup]
└─$ fsstat -o 8064 pd_image1.dd
FILE SYSTEM INFORMATION
--------------------------------------------
File System Type: FAT32

OEM Name: MSDOS5.0
Volume ID: 0xe6dad666
Volume Label (Boot Sector): NO NAME
Volume Label (Root Directory): PENDRIVE1
File System Type Label: FAT32
Next Free Sector (FS Info): 32960
Free Sector Count (FS Info): 241376

Sectors before file system: 8064

File System Layout (in sectors)
Total Range: 0 - 3922047
* Reserved: 0 - 545
** Boot Sector: 0
** FS Info Sector: 1
** Backup Boot Sector: 6
* FAT 0: 546 - 4368
* FAT 1: 4369 - 8191
* Data Area: 8192 - 3922047
** Cluster Area: 8192 - 3922047
*** Root Directory: 8192 - 3153927

METADATA INFORMATION
--------------------------------------------
Range: 2 - 62621702
Root Directory: 2

CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
```

```
CONTENT INFORMATION
--------------------------------------------
Sector Size: 512
Cluster Size: 4096
Total Cluster Range: 2 - 489233

FAT CONTENTS (in sectors)
--------------------------------------------
8192-8199 (8) -> 3153920
8200-8303 (104) -> EOF
8304-8495 (192) -> EOF
8496-8599 (104) -> EOF
8600-8607 (8) -> EOF
8608-13775 (5168) -> EOF
13776-19039 (5264) -> EOF
19040-19055 (16) -> EOF
19056-19119 (64) -> EOF
19120-19471 (352) -> EOF
19472-21767 (2296) -> EOF
21768-22039 (272) -> EOF
22040-22111 (72) -> EOF
```

- We can see that there are 545 reserved sectors until the first FAT. In the reserved area are a backup boot sector and a FSINFO data structure.
- There are two FAT structures, and they span from sectors 546 to 4368 and 4369 to 8191.
- The data area starts in sector 8192, and it has clusters that are 4096 bytes in size.

To determine the configuration of a FAT file system, we need to process **the first sector of the disk--->**

1F80 * 200= 3F0000

pd_image1.dd

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
00000000	FA	BE	00	7C	BF	00	7A	B9	00	01	FC	0E	1F	0E	07	F3	ú¾.|¿.z¹..ü....ó
00000010	A5	EA	16	7A	00	00	BB	BE	7B	33	C9	80	3F	80	75	06	¥ê.z..»¾{3É€?€u.
00000020	FE	C5	8B	F3	EB	07	80	3F	00	75	02	FE	C1	83	C3	10	þÅ‹óë.€?.u.þÁƒÃ.
00000030	81	FB	FE	7B	72	E5	83	F9	04	74	0B	81	F9	03	01	74	.ûþ{råƒù.t..ù..t
00000040	0A	BB	A5	7A	EB	2C	BB	87	7A	EB	27	8B	4C	02	8B	14	.»¥zë,»‡zë'‹L.‹.
00000050	B8	01	02	BB	00	7C	CD	13	73	05	BB	BC	7A	EB	13	2E	¸..».|Í.s.»¼zë..
00000060	A1	FE	7D	3D	55	AA	74	05	BB	BC	7A	EB	05	EA	00	7C	¡þ}=Uªt.»¼zë.ê.|
00000070	00	00	2E	8A	07	3C	00	74	0C	53	BB	07	00	B4	0E	CD	...Š.<.t.S»..´.Í
00000080	10	5B	43	EB	ED	EB	FE	4E	6F	20	62	6F	6F	74	61	62	.[CëíëþNo bootab
00000090	6C	65	20	70	61	72	74	69	74	6F	6E	20	69	6E	20	74	le partiton in t
000000A0	61	62	6C	65	00	49	6E	76	61	6C	69	64	20	50	61	72	able.Invalid Par
000000B0	74	69	74	6F	6E	20	74	61	62	6C	65	00	49	6E	76	61	titon table.Inva
000000C0	6C	69	64	20	6F	72	20	64	61	6D	61	67	65	64	20	42	lid or damaged B
000000D0	6F	6F	74	61	62	6C	65	20	70	61	72	74	69	74	69	6F	ootable partitio
000000E0	6E	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	n...............
000000F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000100	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000110	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000120	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000130	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000140	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000150	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000160	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000170	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001A0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001B0	00	00	00	00	00	00	00	00	EF	01	8F	C4	00	00	00	0C	........ï..Ä....
000001C0	00	0F	0B	0F	60	FB	80	1F	00	00	80	D8	3B	00	00	00	....`û€...€Ø;...
000001D0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001E0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
000001F0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	55	AA	..............Uª
00000200	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
00000210	55	53	42	20	54	65	73	74	65	72	20	32	30	30	38	2D	USB Tester 2008-
00000220	30	33	2D	31	37	20	31	2E	31	36	00	00	00	00	00	00	03-17 1.16......
00000230	32	30	30	38	2F	30	33	2F	33	30	00	00	00	00	00	00	2008/03/30......

pd_image1.dd

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
003F0000	EB	58	90	4D	53	44	4F	53	35	2E	30	00	02	08	22	02	ëX.MSDOS5.0...".
003F0010	02	00	00	00	00	F8	00	00	3F	00	FF	00	80	1F	00	00	.....ø..?.ÿ.€...
003F0020	80	D8	3B	00	EF	0E	00	00	00	00	00	00	02	00	00	00	€Ø;.ï...........
003F0030	01	00	06	00	00	00	00	00	00	00	00	00	00	00	00	00	....
003F0040	80	00	29	66	D6	DA	E6	4E	4F	20	4E	41	4D	45	20	20	€.)fÖÚæNO NAME
003F0050	20	20	46	41	54	33	32	20	20	20	33	C9	8E	D1	BC	F4	FAT32 3ÉŽÑ¼ô
003F0060	7B	8E	C1	8E	D9	BD	00	7C	88	4E	02	8A	56	40	B4	41	{ŽÁŽÙ½.|ˆN.ŠV@´A
003F0070	BB	AA	55	CD	13	72	10	81	FB	55	AA	75	0A	F6	C1	01	»ªUÍ.r..ûUªu.öÁ.
003F0080	74	05	FE	46	02	EB	2D	8A	56	40	B4	08	CD	13	73	05	t.þF.ë-ŠV@´.Í.s.
003F0090	B9	FF	FF	8A	F1	66	0F	B6	C6	40	66	0F	B6	D1	80	E2	¹ÿÿŠñf.¶Æ@f.¶Ñ€â
003F00A0	3F	F7	E2	86	CD	C0	ED	06	41	66	0F	B7	C9	66	F7	E1	?÷â†ÍÀí.Af.·Éf÷á
003F00B0	66	89	46	F8	83	7E	16	00	75	38	83	7E	2A	00	77	32	f‰Føƒ~..u8ƒ~*.w2
003F00C0	66	8B	46	1C	66	83	C0	0C	BB	00	80	B9	01	00	E8	2B	f‹F.fƒÀ.».€¹..è+
003F00D0	00	E9	2C	03	A0	FA	7D	B4	7D	8B	F0	AC	84	C0	74	17	.é,. ú}´}‹ð¬„Àt.
003F00E0	3C	FF	74	09	B4	0E	BB	07	00	CD	10	EB	EE	A0	FB	7D	<ÿt.´.»..Í.ëî û}
003F00F0	EB	E5	A0	F9	7D	EB	E0	98	CD	16	CD	19	66	60	80	7E	ëå ù}ëà˜Í.Í.f`€~
003F0100	02	00	0F	84	20	00	66	6A	00	66	50	06	53	66	68	10	...„ .fj.fP.Sfh.
003F0110	00	01	00	B4	42	8A	56	40	8B	F4	CD	13	66	58	66	58	...´BŠV@‹ôÍ.fXfX
003F0120	66	58	66	58	EB	33	66	3B	46	F8	72	03	F9	EB	2A	66	fXfXë3f;Før.ùë*f
003F0130	33	D2	66	0F	B7	4E	18	66	F7	F1	FE	C2	8A	CA	66	8B	3Òf.·N.f÷ñþÂŠÊf‹
003F0140	D0	66	C1	EA	10	F7	76	1A	86	D6	8A	56	40	8A	E8	C0	ÐfÁê.÷v.†ÖŠV@ŠèÀ
003F0150	E4	06	0A	CC	B8	01	02	CD	13	66	61	0F	82	75	FF	81	ä..Ì¸..Í.fa.‚uÿ.
003F0160	C3	00	02	66	40	49	75	94	C3	42	4F	4F	54	4D	47	52	Ã..f@Iu”ÃBOOTMGR
003F0170	20	20	20	20	00	00	00	00	00	00	00	00	00	00	00	00	....
003F0180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
003F0190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
003F01A0	00	00	00	00	00	00	00	00	00	00	00	00	0D	0A	52	65	..............Re
003F01B0	6D	6F	76	65	20	64	69	73	6B	73	20	6F	72	20	6F	74	move disks or ot
003F01C0	68	65	72	20	6D	65	64	69	61	2E	FF	0D	0A	44	69	73	her media.ÿ..Dis
003F01D0	6B	20	65	72	72	6F	72	FF	0D	0A	50	72	65	73	73	20	k errorÿ..Press
003F01E0	61	6E	79	20	6B	65	79	20	74	6F	20	72	65	73	74	61	any key to resta
003F01F0	72	74	0D	0A	00	00	00	00	00	AC	CB	D8	00	00	55	AA	rt......¬ËØ..Uª
003F0200	52	52	61	41	00	00	00	00	00	00	00	00	00	00	00	00	RRaA............

0x00: 3 bytes, representing the jump instruction;

0x03: 8 bytes, representing the vendor logo and OS version number;

0x0B: 53 bytes, representing BPB;

0x40: 26 bytes, representing extended BPB;

0x5A: 420 bytes, representing bootstrap code;

0x1FE: 2 bytes, representing the valid end flag;

Bios Parameter Block

Data structure for the first 36 bytes of the FAT boot sector.

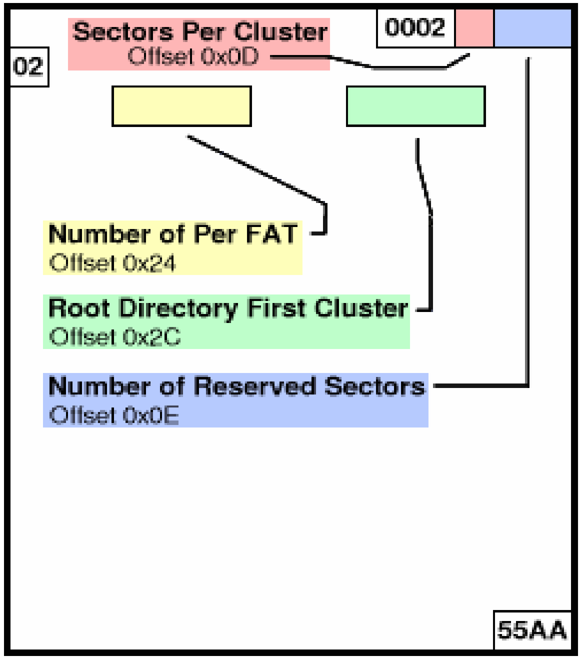
Byte Range	Description
0–2	Assembly instruction to jump to boot code.
3–10	OEM Name in ASCII.
11–12	Bytes per sector. Allowed values include 512, 1024, 2048, and 4096.
13–13	Sectors per cluster (data unit). Allowed values are powers of 2, but the cluster size must be 32KB or smaller.
14–15	Size in sectors of the reserved area.
16–16	Number of FATs. Typically two for redundancy, but according to Microsoft it can be one for some small storage devices.
17–18	Maximum number of files in the root directory for FAT12 and FAT16. This is 0 for FAT32 and typically 512 for FAT16.
19–20	16-bit value of number of sectors in file system. If the number of sectors is larger than can be represented in this 2-byte value, a 4-byte value exists later in the data structure and this should be 0.
21–21	Media type. According to the Microsoft documentation, 0xf8 should be used for fixed disks and 0xf0 for removable.
22–23	16-bit size in sectors of each FAT for FAT12 and FAT16. For FAT32, this field is 0.
24–25	Sectors per track of storage device.
26–27	Number of heads in storage device.
28–31	Number of sectors before the start of partition.[1]
32–35	32-bit value of number of sectors in file system. Either this value or the 16-bit value above must be 0.

Extended Bios Parameter Block used by FAT 36-89 bytes

Data structure for the remainder of the FAT32 boot sector.

Byte Range	Description
36–39	32-bit size in sectors of one FAT.
40–41	Defines how multiple FAT structures are written to. If bit 7 is 1, only one of the FAT structures is active and its index is described in bits 0–3. Otherwise, all FAT structures are mirrors of each other.
42–43	The major and minor version number.
44–47	Cluster where root directory can be found.
48–49	Sector where FSINFO structure can be found.
50–51	Sector where backup copy of boot sector is located (default is 6).
52–63	Reserved.
64–64	BIOS INT13h drive number.
65–65	Not used.
66–66	Extended boot signature to identify if the next three values are valid. The signature is 0x29.
67–70	Volume serial number, which some versions of Windows will calculate based on the creation date and time.
71–81	Volume label in ASCII. The user chooses this value when creating the file system.
82–89	File system type label in ASCII. Standard values include "FAT32," but nothing is required.
90–509	Not used.
510–511	Signature value (0xAA55).

Some Critical fields



FAT32 Volume ID, critical fields

Field	Microsoft's Name	Offset	Size	Value
Bytes Per Sector	BPB_BytsPerSec	0x0B	16 Bits	Always 512 Bytes
Sectors Per Cluster	BPB_SecPerClus	0x0D	8 Bits	1,2,4,8,16,32,64,128
Number of Reserved Sectors	BPB_RsvdSecCnt	0x0E	16 Bits	Usually 0x20
Number of FATs	BPB_NumFATs	0x10	8 Bits	Always 2
Sectors Per FAT	BPB_FATSz32	0x24	32 Bits	Depends on disk size
Root Directory First Cluster	BPB_RootClus	0x2C	32 Bits	Usually 0x00000002
Signature	(none)	0x1FE	16 Bits	Always 0xAA55

pd_image1.dd

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
003F0000	EB	58	90	4D	53	44	4F	53	35	2E	30	00	02	08	22	02
003F0010	02	00	00	00	00	F8	00	00	3F	00	FF	00	80	1F	00	00
003F0020	80	D8	3B	00	EF	0E	00	00	00	00	00	00	02	00	00	00
003F0030	01	00	06	00	00	00	00	00	00	00	00	00	00	00	00	00
003F0040	80	00	29	66	D6	DA	E6	4E	4F	20	4E	41	4D	45	20	20
003F0050	20	20	46	41	54	33	32	20	20	20	33	C9	8E	D1	BC	F4
003F0060	7B	8E	C1	8E	D9	BD	00	7C	88	4E	02	8A	56	40	B4	41
003F0070	BB	AA	55	CD	13	72	10	81	FB	55	AA	75	0A	F6	C1	01
003F0080	74	05	FE	46	02	EB	2D	8A	56	40	B4	08	CD	13	73	05
003F0090	B9	FF	FF	8A	F1	66	0F	B6	C6	40	66	0F	B6	D1	80	E2
003F00A0	3F	F7	E2	86	CD	C0	ED	06	41	66	0F	B7	C9	66	F7	E1
003F00B0	66	89	46	F8	83	7E	16	00	75	38	83	7E	2A	00	77	32
003F00C0	66	8B	46	1C	66	83	C0	0C	BB	00	80	B9	01	00	E8	2B
003F00D0	00	E9	2C	03	A0	FA	7D	B4	7D	8B	F0	AC	84	C0	74	17
003F00E0	3C	FF	74	09	B4	0E	BB	07	00	CD	10	EB	EE	A0	FB	7D
003F00F0	EB	E5	A0	F9	7D	EB	E0	98	CD	16	CD	19	66	60	80	7E
003F0100	02	00	0F	84	20	00	66	6A	00	66	50	06	53	66	68	10
003F0110	00	01	00	B4	42	8A	56	40	8B	F4	CD	13	66	58	66	58
003F0120	66	58	66	58	EB	33	66	3B	46	F8	72	03	F9	EB	2A	66
003F0130	33	D2	66	0F	B7	4E	18	66	F7	F1	FE	C2	8A	CA	66	8B
003F0140	D0	66	C1	EA	10	F7	76	1A	86	D6	8A	56	40	8A	E8	C0
003F0150	E4	06	0A	CC	B8	01	02	CD	13	66	61	0F	82	75	FF	81
003F0160	C3	00	02	66	40	49	75	94	C3	42	4F	4F	54	4D	47	52
003F0170	20	20	20	20	00	00	00	00	00	00	00	00	00	00	00	00
003F0180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003F0190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003F01A0	00	00	00	00	00	00	00	00	00	00	00	00	0D	0A	52	65
003F01B0	6D	6F	76	65	20	64	69	73	6B	73	20	6F	72	20	6F	74
003F01C0	68	65	72	20	6D	65	64	69	61	2E	FF	0D	0A	44	69	73
003F01D0	6B	20	65	72	72	6F	72	FF	0D	0A	50	72	65	73	73	20
003F01E0	61	6E	79	20	6B	65	79	20	74	6F	20	72	65	73	74	61
003F01F0	72	74	0D	0A	00	00	00	00	00	AC	CB	D8	00	00	55	AA
003F0200	52	52	61	41	00	00	00	00	00	00	00	00	00	00	00	00

Offset **0x0B** contains 2 bytes which specify the number of **bytes per sector.**

offset **0x0D** contains 1 byte which specify the **number of sectors per each cluster**.

offset **0x0E** contains 2 bytes which specify the **number of reserved sectors** in this FAT32 partition. This gives us **0222** which is in little-endian. i.e. **546 sectors** in reserved

offset **0x10** contains 1 byte which specify the **number of FAT tables** we have in this partition i..e **2**

offset **0x11** contains 2 bytes which specify the **maximum number of file entries available in the root directory. This** applies only to FAT12 and FAT16 versions of FAT.

pd_image1.dd

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
003F0000	EB	58	90	4D	53	44	4F	53	35	2E	30	00	02	08	22	02
003F0010	02	00	00	00	00	F8	00	00	3F	00	FF	00	80	1F	00	00
003F0020	80	D8	3B	00	EF	0E	00	00	00	00	00	00	02	00	00	00
003F0030	01	00	06	00	00	00	00	00	00	00	00	00	00	00	00	00
003F0040	80	00	29	66	D6	DA	E6	4E	4F	20	4E	41	4D	45	20	20
003F0050	20	20	46	41	54	33	32	20	20	20	33	C9	8E	D1	BC	F4
003F0060	7B	8E	C1	8E	D9	BD	00	7C	88	4E	02	8A	56	40	B4	41
003F0070	BB	AA	55	CD	13	72	10	81	FB	55	AA	75	0A	F6	C1	01
003F0080	74	05	FE	46	02	EB	2D	8A	56	40	B4	08	CD	13	73	05
003F0090	B9	FF	FF	8A	F1	66	0F	B6	C6	40	66	0F	B6	D1	80	E2
003F00A0	3F	F7	E2	86	CD	C0	ED	06	41	66	0F	B7	C9	66	F7	E1
003F00B0	66	89	46	F8	83	7E	16	00	75	38	83	7E	2A	00	77	32
003F00C0	66	8B	46	1C	66	83	C0	0C	BB	00	80	B9	01	00	E8	2B
003F00D0	00	E9	2C	03	A0	FA	7D	B4	7D	8B	F0	AC	84	C0	74	17
003F00E0	3C	FF	74	09	B4	0E	BB	07	00	CD	10	EB	EE	A0	FB	7D
003F00F0	EB	E5	A0	F9	7D	EB	E0	98	CD	16	CD	19	66	60	80	7E
003F0100	02	00	0F	84	20	00	66	6A	00	66	50	06	53	66	68	10
003F0110	00	01	00	B4	42	8A	56	40	8B	F4	CD	13	66	58	66	58
003F0120	66	58	66	58	EB	33	66	3B	46	F8	72	03	F9	EB	2A	66
003F0130	33	D2	66	0F	B7	4E	18	66	F7	F1	FE	C2	8A	CA	66	8B
003F0140	D0	66	C1	EA	10	F7	76	1A	86	D6	8A	56	40	8A	E8	C0
003F0150	E4	06	0A	CC	B8	01	02	CD	13	66	61	0F	82	75	FF	81
003F0160	C3	00	02	66	40	49	75	94	C3	42	4F	4F	54	4D	47	52
003F0170	20	20	20	20	00	00	00	00	00	00	00	00	00	00	00	00
003F0180	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003F0190	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00
003F01A0	00	00	00	00	00	00	00	00	00	00	00	00	0D	0A	52	65
003F01B0	6D	6F	76	65	20	64	69	73	6B	73	20	6F	72	20	6F	74
003F01C0	68	65	72	20	6D	65	64	69	61	2E	FF	0D	0A	44	69	73
003F01D0	6B	20	65	72	72	6F	72	FF	0D	0A	50	72	65	73	73	20
003F01E0	61	6E	79	20	6B	65	79	20	74	6F	20	72	65	73	74	61
003F01F0	72	74	0D	0A	00	00	00	00	00	AC	CB	D8	00	00	55	AA
003F0200	52	52	61	41	00	00	00	00	00	00	00	00	00	00	00	00

offset **0x16** in the boot sector has 2 bytes which **specify the number of sectors in each FAT table**.

if the location contains all zeros in those two bytes, that means, the space is not enough to specify the information. In that case, we have to go to the offset **0x24** and interpret 4 bytes there. i.e 0000 0E EF = **3823 Sectors in each FAT.**

offset **0x2C** contains 1 byte which specify the **first cluster of the root directory**. i.e 02 , therefore cluster #2 is allocated to root directory and there are two clusters in this disk image before the root directory, namely cluster #0 and cluster #1.

The root directory is located right after the two FAT tables.

That means, we just have to walk through the reserved area from the beginning of the partition, then through the FAT1 and FAT2 tables and **there we find the root directory.**

Offset to root directory =

= (number of sectors in reserved area) + (number of sectors in a FAT table) x 2

= 546 + 3823 x 2

= 8192 (SECTORS)

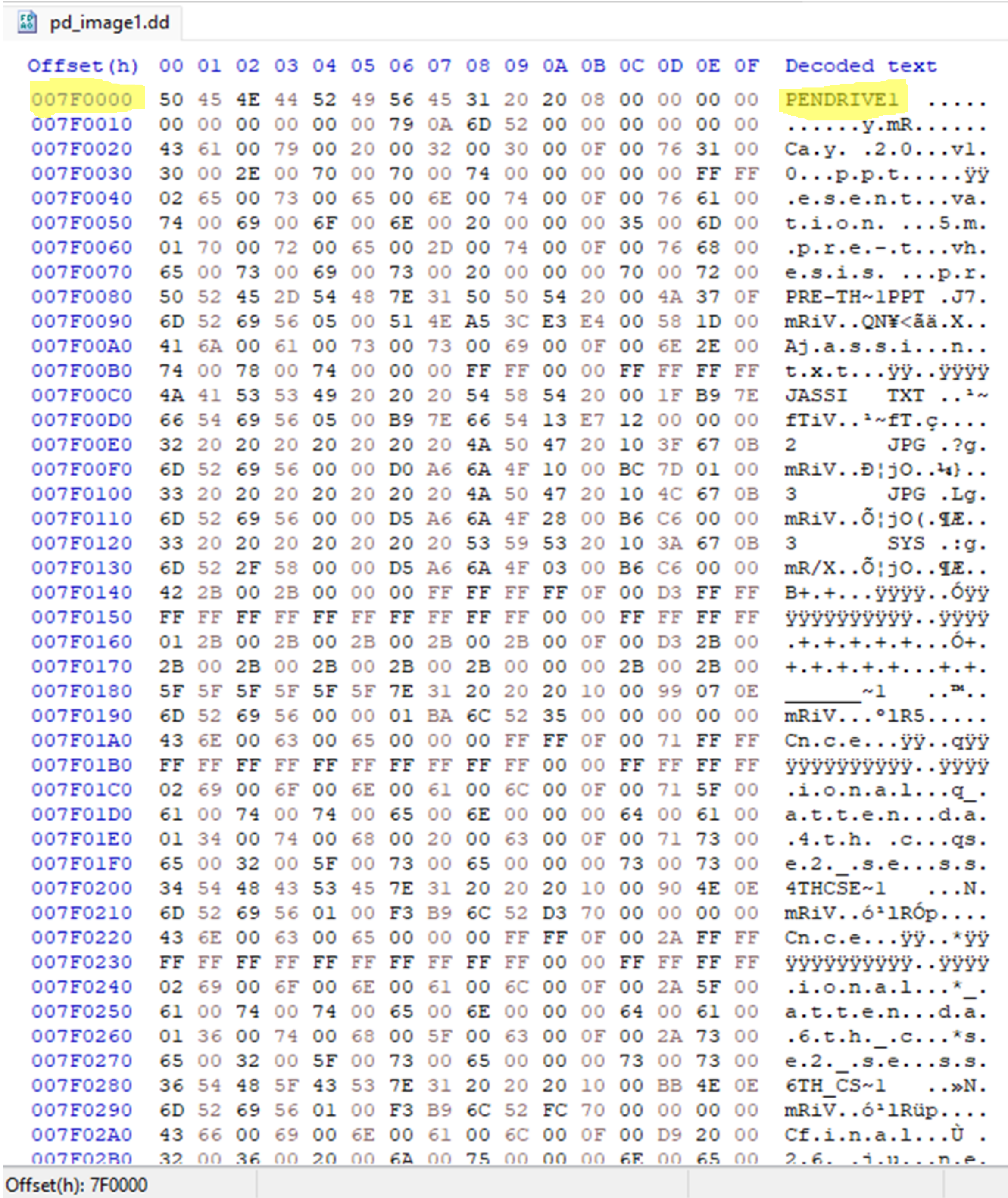
=8192 x 512 (bytes)

= 41,94,304 (bytes)

= 40 0000 (hex)

= 40 0000 + 3F0000 (offset specifies the location from the beginning of the partition)

= 7F 0000 (hex)



The Root directory contains entries which are 32 bytes long.

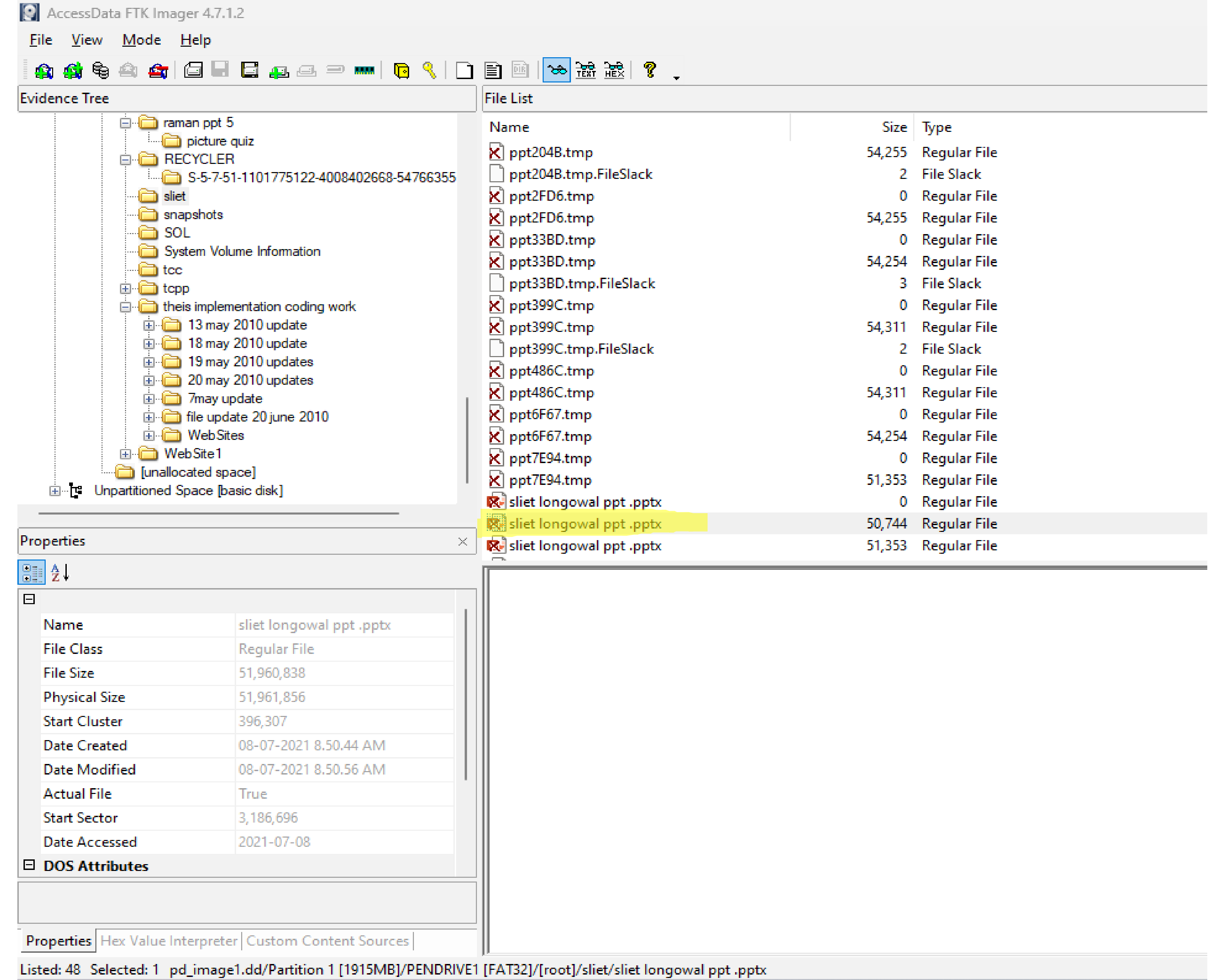
The **first byte of a root directory entry** is important.

If a file is deleted, the first byte of a root directory entry is simply set to **0xE5**.

For ex: in this pendrive image
A file named
Sliet longowal.pptx
has been deleted.

pd_image1.dd

Offset(h)	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	Decoded text
613EDAD0	E8	52	E8	52	00	00	70	54	E8	52	CC	78	33	BB	4F	03	èRèR..pTèRÌx3»O.
613EDAE0	E5	6C	00	20	00	70	00	70	00	74	00	0F	00	C7	20	00	ål. .p.p.t...Ç .
613EDAF0	2E	00	70	00	70	00	74	00	78	00	00	00	00	00	FF	FF	..p.p.t.x.....ÿÿ
613EDB00	E5	73	00	6C	00	69	00	65	00	74	00	0F	00	C7	20	00	ås.l.i.e.t...Ç .
613EDB10	6C	00	6F	00	6E	00	67	00	6F	00	00	00	77	00	61	00	l.o.n.g.o...w.a.
613EDB20	E5	4C	49	45	54	4C	7E	31	50	50	54	20	00	40	56	46	åLIETL~1PPT .@VF
613EDB30	E8	52	E8	52	06	00	B6	54	E8	52	F3	AA	82	B6	4F	03	èRèR..¶TèRóª‚¶O.
613EDB40	E5	70	00	70	00	74	00	36	00	46	00	0F	00	0A	36	00	åp.p.t.6.F....6.
613EDB50	37	00	2E	00	74	00	6D	00	70	00	00	00	00	00	FF	FF	7...t.m.p.....ÿÿ
613EDB60	E5	50	54	36	46	36	37	20	54	4D	50	20	00	0F	B7	54	åPT6F67 TMP ..·T
613EDB70	E8	52	E8	52	00	00	B8	54	E8	52	00	00	00	00	00	00	èRèR..¸TèR......
613EDB80	E5	70	00	70	00	74	00	36	00	46	00	0F	00	0A	36	00	åp.p.t.6.F....6.
613EDB90	37	00	2E	00	74	00	6D	00	70	00	00	00	00	00	FF	FF	7...t.m.p.....ÿÿ
613EDBA0	E5	50	54	36	46	36	37	20	54	4D	50	20	00	40	56	46	åPT6F67 TMP .@VF
613EDBB0	E8	52	E8	52	06	00	C0	54	E8	52	CC	78	69	B6	4F	03	èRèR..ÀTèRÌxi¶O.
613EDBC0	E5	36	30	39	44	43	30	37	54	4D	50	20	10	40	56	46	å609DC07TMP .@VF
613EDBD0	E8	52	E8	52	00	00	B6	54	E8	52	F3	AA	82	B6	4F	03	èRèR..¶TèRóª‚¶O.
613EDBE0	42	6C	00	20	00	70	00	70	00	74	00	0F	00	C7	20	00	Bl. .p.p.t...Ç .
613EDBF0	2E	00	70	00	70	00	74	00	78	00	00	00	00	00	FF	FF	..p.p.t.x.....ÿÿ
613EDC00	01	73	00	6C	00	69	00	65	00	74	00	0F	00	C7	20	00	.s.l.i.e.t...Ç .
613EDC10	6C	00	6F	00	6E	00	67	00	6F	00	00	00	77	00	61	00	l.o.n.g.o...w.a.
613EDC20	53	4C	49	45	54	4C	7E	31	50	50	54	20	00	40	56	46	SLIETL~1PPT .@VF
613EDC30	E8	52	69	56	06	00	C0	54	E8	52	CC	78	69	B6	4F	03	èRiV..ÀTèRÌxi¶O.
613EDC40	E5	77	00	61	00	6C	00	20	00	70	00	0F	00	F9	70	00	åw.a.l. .p...ùp.
613EDC50	74	00	20	00	2E	00	70	00	70	00	00	00	74	00	78	00	t. ...p.p...t.x.
613EDC60	E5	7E	00	24	00	73	00	6C	00	69	00	0F	00	F9	65	00	å~.$.s.l.i...ùe.
613EDC70	74	00	20	00	6C	00	6F	00	6E	00	00	00	67	00	6F	00	t. .l.o.n...g.o.
613EDC80	E5	24	53	4C	49	45	7E	31	50	50	54	22	00	0E	CF	55	å$SLIE~1PPT"..ÏU
613EDC90	E8	52	E8	52	00	00	4D	59	E8	52	00	0C	A5	00	00	00	èRèR..MYèR..¥...
613EDCA0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
613EDCB0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....
613EDCC0	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	00	....


AccessData FTK Imager 4.7.1.2
File View Mode Help
Evidence Tree
raman ppt 5
picture quiz
RECYCLER
S-5-7-51-1101775122-4008402668-54766355
sliet
snapshots
SOL
System Volume Information
tcc
tcpp
theis implementation coding work
13 may 2010 update
18 may 2010 update
19 may 2010 updates
20 may 2010 updates
7may update
file update 20 june 2010
WebSites
WebSite1
[unallocated space]
Unpartitioned Space [basic disk]
File List
Name Size Type
ppt204B.tmp 54,255 Regular File
ppt204B.tmp.FileSlack 2 File Slack
ppt2FD6.tmp 0 Regular File
ppt2FD6.tmp 54,255 Regular File
ppt33BD.tmp 0 Regular File
ppt33BD.tmp 54,254 Regular File
ppt33BD.tmp.FileSlack 3 File Slack
ppt399C.tmp 0 Regular File
ppt399C.tmp 54,311 Regular File
ppt399C.tmp.FileSlack 2 File Slack
ppt486C.tmp 0 Regular File
ppt486C.tmp 54,311 Regular File
ppt6F67.tmp 0 Regular File
ppt6F67.tmp 54,254 Regular File
ppt7E94.tmp 0 Regular File
ppt7E94.tmp 51,353 Regular File
sliet longowal ppt .pptx 0 Regular File
sliet longowal ppt .pptx 50,744 Regular File
sliet longowal ppt .pptx 51,353 Regular File
Properties
Name sliet longowal ppt .pptx
File Class Regular File
File Size 51,960,838
Physical Size 51,961,856
Start Cluster 396,307
Date Created 08-07-2021 8.50.44 AM
Date Modified 08-07-2021 8.50.56 AM
Actual File True
Start Sector 3,186,696
Date Accessed 2021-07-08
DOS Attributes
Properties Hex Value Interpreter Custom Content Sources
Listed: 48 Selected: 1 pd_image1.dd/Partition 1 [1915MB]/PENDRIVE1 [FAT32]/[root]/sliet/sliet longowal ppt .pptx

Root Directory Entry Format (SFN)

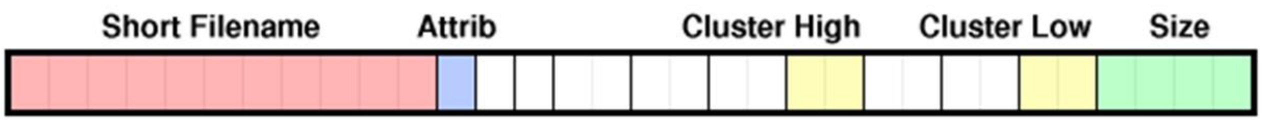
Root Directory SFN Entry Data Structure	
Bytes	**Purpose**
0	First character of file name (ASCII) *or* allocation status (0x00=unallocated, 0xe5=deleted)
1-10	Characters 2-11 of the file name (ASCII); the "." is implied between bytes 7 and 8
11	File attributes (see File Attributes table)
12	Reserved
13	File creation time (in tenths of seconds)*
14-15	Creation time (hours, minutes, seconds)*
16-17	Creation date*
18-19	Access date*
20-21	High-order 2 bytes of address of first cluster (0 for FAT12/16)*
22-23	Modified time (hours, minutes, seconds)
24-25	Modified date
26-27	Low-order 2 bytes of address of first cluster
28-31	File size (0 for directories)

File Attributes	
Flag Value	**Description**
0000 0001 (0x01)	Read-only
0000 0010 (0x02)	Hidden file
0000 0100 (0x04)	System file
0000 1000 (0x08)	Volume label
0000 1111 (0x0f)	Long file name
0001 0000 (0x10)	Directory
0010 0000 (0x20)	Archive

** Bytes 13-22 are unused by DOS*

Directory data is organized in 32 byte records.

At the end of the directory is a record that begins with zero.
All other records will be non-zero in their first byte, so this is an easy way to determine when you have reached the end of the directory.



32 Byte Directory Structure, Short Filename Format

Field	Microsoft's Name	Offset	Size
Short Filename	DIR_Name	0x00	11 Bytes
Attrib Byte	DIR_Attr	0x0B	8 Bits
First Cluster High	DIR_FstClusHI	0x14	16 Bits
First Cluster Low	DIR_FstClusLO	0x1A	16 Bits
File Size	DIR_FileSize	0x1C	32 Bits

The first 11 bytes are the short filename. The extension is always the last three bytes.

If the file's name is shorter than 8 bytes, the unused bytes are filled with spaces (0x20).

The starting cluster number is found as two 16 bit sections, and the file size (in bytes) is found in the last four bytes of the record.

The first cluster number tells you where the file's data begins on the drive, and the size field tells how long the file is.

There are four types of 32-byte directory records.

Normal record with short filename : Attrib is normal
Long filename text : Attrib has all four type bits set
Unused - First byte is 0xE5
End of directory - First byte is zero
The Attrib byte has six bits defined, as shown in the table below.

Attrib Bit	Function	LFN	Comment
0 (LSB)	Read Only	1	Should not allow writing
1	Hidden	1	Should not show in dir listing
2	System	1	File is operating system
3	Volume ID	1	Filename is Volume ID
4	Directory	x	Is a subdirectory (32-byte records)
5	Archive	x	Has been changed since last backup
6	Ununsed	0	Should be zero
7 (MSB)	Ununsed	0	Should be zero

Most simple firmware will check the Attrib byte to determine if the 32 bytes are a normal record or long filename data, and to determine if it is a normal file or a subdirectory.

Long filename records have all four of the least significant bits set.

Normal files rarely have any of these four bits set.