File Allocation Table

File Allocation Table (**FAT**) is a computer file system architecture and a family of industry-standard file systems utilizing it. The FAT file system is a continuing standard which borrows source code from the original, legacy file system and proves to be simple and robust.[3] It offers useful performance even in lightweight implementations, but cannot deliver the same performance, reliability and scalability as some modern file systems. It is, however, supported for compatibility reasons by nearly all currently developed operating systems for personal computers and many mobile devices and embedded systems, and thus is a well-suited format for data exchange between computers and devices of almost any type and age from 1981 up to the present.

Originally designed in 1977 for use on floppy disks, FAT was soon adapted and used almost universally on hard disks throughout the DOS and Windows 9x eras for two decades.[4] As disk drives evolved, the capabilities of the file system have been extended accordingly, resulting in three major file system variants: FAT12, FAT16 and FAT32. The FAT standard has also been expanded in other ways while generally preserving backward compatibility with existing software.

With the introduction of more powerful computers and operating systems, as well as the development of more complex file systems for them, FAT is no longer the default file system for usage on Microsoft Windows computers.[5]

FAT file systems are still commonly found on floppy disks, flash and other solid-state memory cards and modules (including USB flash drives), as well as many portable and embedded devices. FAT is the standard file system for digital cameras per the DCF specification.

FAT	
Developer(s)	Microsoft, NCR, SCP, IBM, Compaq, Digital Research, Novell, Caldera
Full name	File Allocation Table
Variants	8-bit FAT, FAT12, FAT16, FAT16B, FAT32, ExFAT, FATX, FAT+
Introduced	1977 with Standalone Disk BASIC-80
Partition identifier	MBR/EBR: FAT12: 0x01 e.a. (Extended Attribute) FAT16: 0x04 0x06 0x0E e.a. FAT32: 0x0B 0x0C e.a. BDP: EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
Structures	
Directory contents	Table
File allocation	Linked list
Bad blocks	Cluster tagging
Limits	
Max. volume size	FAT12: 32 MiB (256 MiB for 64 KiB clusters) FAT16: 2 GiB (4 GiB for 64 KiB clusters) FAT32: 2 TiB (16 TiB for 4 KiB sectors)
Max. file size	4,294,967,295 bytes

Contents

	(4 GiB – 1) with FAT16B and FAT32[1]
Max. number of files	FAT12: 4,068 for 8 KiB clusters FAT16: 65,460 for 32 KiB clusters FAT32: 268,173,300 for 32 KiB clusters
Max. filename length	8.3 filename, or 255 UCS-2 characters when using LFN[nb 1]
Features	
Dates recorded	Modified date/time, creation date/time (DOS 7.0 and higher only), access date (only available with ACCDATE enabled),[2] deletion date/time (only with DELWATCH 2)
Date range	1980-01-01 to 2099-12-31 (2107-12-31)
Date resolution	2 seconds for last modified time, 10 ms for creation time, 1 day for access date, 2 seconds for deletion time
Forks	Not natively
Attributes	Read-only, Hidden, System, Volume, Directory, Archive
File system permissions	FAT12/FAT16: File, directory and volume access rights for Read, Write, Execute, Delete only with DR-DOS, PalmDOS, Novell DOS, OpenDOS, FlexOS, 4680 OS,

Overview

Concepts

The name of the file system originates from the file system's prominent usage of an index table, the *File Allocation Table* (*FAT*), statically allocated at the time of formatting. The table contains entries for each *cluster*, a contiguous area of disk storage. Each entry contains either the number of the next cluster in the file, or else a marker indicating end of file, unused disk space, or special reserved areas of the disk. The *root directory* of the disk contains the number of the first cluster of each file in that directory; the operating system can then traverse the FAT, looking up the cluster number of each successive part of the disk file as a *cluster chain* until the end of the file is reached. In much the same way, *sub-directories* are implemented as special files containing the *directory entries* of their respective files.

Originally designed as an 8-bit file system, the maximum number of clusters has been significantly increased as disk drives have evolved, and so the number of bits used to identify each cluster has grown. The successive major variants of the FAT format are named after the number of table element bits: 12 (FAT12), 16 (FAT16), and 32 (FAT32). Except for the original 8-bit FAT precursor, each of these variants is still in use. The FAT standard has also been expanded in other ways while generally preserving backward compatibility with existing software.

Uses

	4690 OS, Concurrent DOS, Multiuser DOS, System Manager, REAL/32 (Execute right only with FlexOS, 4680 OS, 4690 OS; individual file / directory passwords not with FlexOS, 4680 OS, 4690 OS; World/Group/Owner permission classes only with multiuser security loaded) FAT32: Partial, only with DR-DOS, REAL/32 and 4690 OS
Transparent compression	FAT12/FAT16: Per-volume, SuperStor, Stacker, DoubleSpace, DriveSpace FAT32: No
Transparent encryption	FAT12/FAT16: Per-volume only with DR-DOS FAT32: No

The FAT file system has a long history (over three decades) of usage on desktops and portable computers, and it is frequently used in embedded solutions.

FAT offers reasonably good performance and robustness, even in very light-weight implementations.[3] It is therefore widely adopted and supported by virtually all existing operating systems for personal computers as well as some home computers and a multitude of embedded systems. This also makes it a useful format for solid-state memory cards and a convenient way to share data between operating systems.

FAT file systems are the default file system for removable media (with the exception of CDs and DVDs) and as such are commonly found on floppy disks, super-floppies, memory and flash memory cards or USB flash drives and are supported by most portable devices such as PDAs, digital cameras, camcorders, media players, or mobile phones. While FAT12 is omnipresent on floppy disks, FAT16 and FAT32 are typically found on the larger media.

FAT was also commonly used on hard disks throughout the DOS and Windows 9x eras, but its use on hard drives has declined since the introduction of Windows XP, which primarily uses the newer NTFS. FAT is still used in hard drives expected to be used by multiple operating systems, such as in shared Windows, GNU/Linux and DOS environments.

Due to the widespread use of FAT-formatted media, many operating systems provide support for FAT through official or third-party file system handlers. For example, OS/2, GNU/Linux, FreeBSD and BeOS provide built-in support for FAT, even though they also support more sophisticated file systems such as ext4 or btrfs. Mac OS 9 and macOS support FAT file systems on volumes other than the boot disk. AmigaOS supports FAT through the CrossDOS package.

For many purposes, the NTFS file system is superior to FAT in terms of features and reliability; its main drawbacks are its complexity and the size overhead for small volumes as well as the very limited support by anything other than the NT-based versions of Windows, since the exact specification is a trade secret of Microsoft. The availability of NTFS-3G since mid-2006 has led to much improved NTFS support in Unix-like operating systems, considerably alleviating this concern. It is still not possible to use NTFS in DOS-like operating systems without third-party drivers, which in turn makes it difficult to use a DOS floppy for recovery purposes. Microsoft provided a Recovery Console to work around this issue, but for security reasons it severely limited what could be done through the Recovery Console by default. The movement of recovery utilities to boot CDs based on BartPE, Linux (with NTFS-3G), or WinPE is eroding this drawback, but the complexity of NTFS prevents its implementation in light-weight operating systems or most embedded systems.

The DCF file system adopted by almost all digital cameras since 1998 defines a logical file system with 8.3 filenames and makes the use of either FAT12, FAT16, FAT32 or exFAT mandatory for its physical layer in order to maximize platform interoperability.[6]

FAT is also used internally for the EFI system partition (partition type `0xEF`) in the boot stage of EFI-compliant computers.[7]

Hidden FAT filesystems are also used in the UEFI boot partition on modern PC's where the UEFI specification requires compliant firmware to be capable of reading FAT12, FAT16 and FAT32 compliant partitions.

For floppy disks, FAT has been standardized as ECMA-107[8] and ISO/IEC 9293:1994[9] (superseding ISO 9293:1987[10]). These standards cover FAT12 and FAT16 with only short 8.3 filename support; long filenames with VFAT are partially patented.[11]

Nomenclature

Technically, the term "FAT file system" refers to all three major variants of the file system, FAT12, FAT16 and FAT32, and most parties clearly distinguish between them where necessary. In contrast to this, Microsoft typically no longer distinguishes between all three of them since the introduction of FAT32, and refers to both FAT12 and FAT16 as "FAT", whereas "FAT32" gets treated specially in dialog boxes and documentation. This can sometimes lead to confusion if the actual type of the file system used is not mentioned or cannot be explicitly specified (e.g., "Do you want to format as FAT or FAT32?" instead of "Do you want to format as FAT12, FAT16 or FAT32?").

Another common cause of confusion exists within the group of FAT16 file systems, since the term "FAT16" refers to both, either the whole group of FAT file systems with 16-bit wide cluster entries, or specifically only the original implementation of it with 16-bit sector entries, when it becomes necessary to differentiate between the original and the later implementation. While technically the newer variant with 32-bit sector entries is called "FAT16B", it is commonly referred to under the name "FAT16" as well, in particular since the original variant is rarely seen today and typically only used on small media when backward compatibility with DOS before 3.31 is required.

Further, the term "VFAT" has led to various misconceptions as well,[nb 2] as it is sometimes erroneously used as if it would describe another variant of FAT file system to be distinguished from the FAT12, FAT16 and FAT32 file systems, while in reality it does not specify another file system, but an optional extension, which can work on top of any FAT file system, FAT12, FAT16 or FAT32. Volumes utilizing VFAT long-filenames can be read also by operating systems not supporting the VFAT extension, as long any operating systems that support the underlying file system (FAT12, FAT16, or FAT32).

Yet another cause for misconceptions stems from some apparent redundancy and possible ambiguity in the definition of FAT volumes. The general type of file system (FAT12, FAT16 or FAT32) is determined by the width of the cluster entries in the FAT. Specific threshold values for the amount of clusters[7] (as stored in the BPB) have been defined to determine which FAT type is used. Even though other properties such as the size of the volume, the count of sectors, the BPB format, the file system name in an EBPB, or -in case of partitioned media- the used partition ID may often seem to be well-suited distinguishing criteria as well, they cannot reliably be used to derive the file system type from in all scenarios.[7] Whilst uncommon, it is technically possible to define a FAT12 or FAT16 volume using a "FAT32 EBPB" (which is sort of a misnomer for the EBPB variant introduced with DOS 7.1), which is normally used for FAT32 volumes, only.[nb 3] Also, while partition IDs sometimes indicate special properties such as hidden, secure, CHS or LBA access to an operating system, and as such are often used in conjunction with particular file system variants only, they are typically not used to specify a type of file system by themselves, but rather to keep (older or foreign) operating systems not aware of a partition ID from accessing partitions they cannot handle or should not work with.[12] It is therefore necessary to distinguish generic *FAT file system types* such as FAT12, FAT16 or FAT32 from *FAT partition types* such as FAT12, FAT16, FAT16B, FAT16X, FAT32, FAT32X etc.

Types

Original 8-bit FAT

The original FAT file system (or *FAT structure*, as it was called initially) was designed and coded by Marc McDonald,[15] based on a series of discussions between McDonald and Bill Gates.[15] It was introduced with 8-bit table elements[13][14][15] (and valid data cluster numbers up to `0xBF`[13][14]) in a precursor to Microsoft's *Standalone Disk BASIC-80* for an 8080-based successor[nb 4] of the NCR 7200 model VI data-entry terminal, equipped with 8-inch (200 mm) floppy disks, in 1977[16] or 1978.[nb 4] In 1978, *Standalone Disk BASIC-80* was ported to the 8086 using an emulator on a DEC PDP-10,[17] since no real 8086 systems were available at this time. The FAT file system was also utilized in Microsoft's MDOS/MIDAS,[15] an operating system for 8080/Z80 platforms written by McDonald since 1979. The *Standalone Disk BASIC* version supported three FATs,[13][14][18] whereas this was a parameter for MIDAS. Reportedly, MIDAS was also prepared to support 10-bit, 12-bit and 16-bit FAT variants. While the size of directory entries was 16 bytes in *Standalone Disk BASIC*,[13][14] MIDAS instead occupied 32 bytes per entry.

Tim Paterson of Seattle Computer Products (SCP) was first introduced to Microsoft's FAT structure when he helped Bob O'Rear adapting the *Standalone Disk BASIC-86* emulator port onto SCP's S-100 bus 8086 CPU board prototype during a guest week at Microsoft in May 1979.[17] The final product was shown at Lifeboat Associates' booth stand at the National Computer Conference in New York[17] on June 4–7, 1979, where Paterson learned about the more sophisticated FAT implementation in MDOS/MIDAS[15] and McDonald talked to him about the design of the file system.[16]

FAT12

Between April and August 1980, while borrowing the FAT concept for SCP's own 8086 operating system QDOS 0.10,[17] Tim Paterson extended the table elements to **12 bits**,[19] reduced the number of FATs to two, redefined the semantics of some of the reserved cluster values, and modified the disk layout, so that the root directory was now located between the FAT and the data area for his implementation of **FAT12**. Paterson also increased the nine-character (6.3) filename[13][14] length limit to eleven characters in order to support CP/M-style 8.3 filenames and File Control Blocks. The format used in Microsoft *Standalone Disk BASIC's* 8-bit file system precursor was not supported by QDOS. By August 1980, QDOS had been already renamed 86-DOS.[20] Starting with 86-DOS 0.42, the size and layout of directory entries was changed from 16 bytes to 32 bytes[21] in order to add a file date stamp[21] and increase the theoretical file size limit beyond the previous limit of 16 MB.[21] 86-DOS 1.00 became available in early 1981. Later in 1981, 86-DOS evolved into Microsoft's MS-DOS and IBM PC DOS.[15][19][22] The capability to read previously formatted volumes with 16-byte directory entries[21] was dropped with MS-DOS 1.20.

8-bit FAT

Developer(s)	Microsoft, NCR, SCP
Full name	8-bit File Allocation Table
Introduced	1977/1978: NCR Basic +6 for NCR 1978: Standalone Disk BASIC-80 (16-byte directory entries)[13][14] (1978: Standalone Disk BASIC-86 internal only) 1979-06-04: Standalone Disk BASIC-86 for SCP (16-byte directory entries) 1979: MIDAS (32-byte directory entries)
Limits	
Max. file size	8 MB
File size granularity	record-granularity (128 bytes)[13][14]
Max. filename length	6.3 filename (binary files), 9 characters (ASCII files)[13][14]
Max. directory depth	no sub-directories
Allowed characters in filenames	ASCII (`0x00` and `0xFF` not allowed in first character)[13][14]
Features	
Dates recorded	No
Attributes	Write protected, EBCDIC conversion, Read after write, Binary (random

Originally designed as a file system for floppy disks, FAT12 used 12-bit entries for the cluster addresses in the FAT, which not only limited the maximum generally possible count of data clusters to 4078 (for data clusters `0x002` to `0xFEF`)[23][24] or in some controlled scenarios even up to 4084 (for data clusters `0x002` to `0xFF5`),[7][8][25] but made FAT manipulation tricky with the PC's 8-bit and 16-bit registers. (While MS-DOS and PC DOS support up to 4084 data clusters on FAT12 volumes in general, cluster value `0xFF0`[nb 5] is treated as additional end-of-chain marker on any FAT12 volume[12] since MS-DOS/PC DOS 3.3, which also introduced the `0xF0` media descriptor value, therefore restricting the maximum practical number of data clusters to 4078 for compatibility purposes with these operating systems.)

The disk's size was stored and calculated as a 16-bit count of sectors, which limited the size to 32 MiB for a logical sector size of 512 bytes. FAT12 was used by several manufacturers with different physical formats, but a typical floppy disk at the time was 5.25-inch (130 mm), single-sided, 40 tracks, with 8 sectors per track, resulting in a capacity of 160 KiB for both the system areas and files. The FAT12 limitations exceeded this capacity by a factor of ten or more. (NB. The 32 MiB limit was later circumvented using logical sectored FATs with logical sector sizes larger than 512 bytes in some OEM versions of MS-DOS 3.x, but this fell into disuse when FAT16B became available with DOS 3.31, which supported 32-bit sector numbers and thereby further lifted the limits.)

By convention, all the control structures were organized to fit inside the first track, thus avoiding head movement during read and write operations, although this varied depending on the manufacturer and physical format of the disk. A limitation which was not addressed until much later (with FAT32) was that any bad sector in the control structures area, track 0, could prevent the disk from being usable. The DOS formatting tool rejected such disks completely. Bad sectors were allowed only in the file data area and (since DOS 2.0) were marked with the reserved value `0xFF7` in the FAT. They made the entire containing cluster unusable.

While 86-DOS supported three disk formats (250.25 KiB, 616 KiB and 1232 KiB with FAT IDs `0xFF` and `0xFE`) on 8-inch (200 mm) floppy drives, IBM PC DOS 1.0, released with the original IBM Personal Computer in 1981, supported only an 8-sector floppy format with a formatted capacity of 160 KiB (FAT ID `0xFE`) for single-sided 5.25-inch floppy drives, and PC DOS 1.1 added support for a double-sided format with 320 KiB (FAT ID `0xFF`). PC DOS 2.0 introduced support for 9-sector floppy formats with 180 KiB (FAT ID `0xFC`) and 360 KiB (FAT ID `0xFD`).

86-DOS 1.00 and PC DOS 1.0 directory entries included only one date, the last modified date. PC DOS 1.1 added the last modified time. PC DOS 1.x file attributes included a hidden bit and system bit, with the remaining six bits undefined. At this time, DOS did not support a hierarchical file system, which was still acceptable, given that the number of files on a disk was typically not more than a few dozen.

	rather than sequential file)[13][14]

FAT12

Developer(s)	SCP, Microsoft, IBM, Digital Research, Novell
Full name	12-bit File Allocation Table
Introduced	1980-07 (QDOS 0.10, 16-byte directory entries) 1981-02-25 (86-DOS 0.42, 32-byte directory entries, several reserved sectors) ca. 1981-08/10 (PC DOS 1.0, 32-byte directory entries, 1 reserved sector) 1982-03-03 (MS-DOS 1.25, 32-byte directory entries, 1 reserved sector)
Partition identifier	MBR/EBR: FAT12: 0x01 e.a. BDP: `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`
Limits	
Max. volume size	16 MiB (with 4 KiB clusters) 32 MiB (with 8 KiB clusters)
Max. file size	limited by volume size
File size granularity	1 byte
Max. number of files	4,068 for 8 KiB clusters
Max. filename	8.3 filename with OEM characters,

The PC XT was the first PC with a hard drive from IBM, and PC DOS 2.0 supported that hard drive with FAT12 (FAT ID `0xF8`). The fixed assumption of 8 sectors per clusters on hard disks practically limited the maximum partition size to 16 MiB for 512 byte sectors and 4 KiB clusters.

The *BIOS Parameter Block* (*BPB*) was introduced with PC DOS 2.0 as well, and this version also added read-only, archive, volume label, and directory attribute bits for hierarchical sub-directories.[26]

MS-DOS 3.0 introduced support for high-density 1.2 MiB 5.25-inch diskettes (media descriptor `0xF9`), which notably had 15 sectors per track, hence more space for the FATs.

FAT12 remains in use on all common floppy disks, including 1.44 MiB and later 2.88 MiB disks (media descriptor byte `0xF0`).

Initial FAT16

On August 14, 1984, IBM released the PC AT, which featured a 20 MiB hard disk and PC DOS 3.0.[27][28] Microsoft introduced MS-DOS 3.0 in parallel. Cluster addresses were increased to 16-bit, allowing for up to 65,524 clusters per volume, and consequently much greater file system sizes, at least in theory. However, the maximum possible number of sectors and the maximum (partition, rather than disk) size of 32 MiB did not change. Therefore, although cluster addresses were 16 bits, this format was not what today is commonly understood as **FAT16**. A partition type `0x04` indicates this form of FAT16 with less than 65536 sectors (less than 32 MiB for sector size 512).

With the initial implementation of FAT16 not actually providing for larger partition sizes than FAT12, the early benefit of FAT16 was to enable the use of smaller clusters, making disk usage more efficient, particularly for large numbers of files only a few hundred bytes in size.

As MS-DOS 3.0 and higher formatted all 16 MiB-32 MiB partitions in the FAT16 format, a 20 MiB hard disk formatted under MS-DOS 3.0 was not accessible by the older MS-DOS 2.0 although MS-DOS 2.0 was able to access 16 MiB-32 MiB FAT12 partitions[29]. MS-DOS 3.0 to MS-DOS 3.30 could still access FAT12 partitions under 15 MiB but required all 16 MiB-32 MiB partitions to be FAT16, and so failed to access MS-DOS 2.0 partitions in this size range. MS-DOS 3.31 and higher could access 16 MiB-32 MiB FAT12 partitions again.

Logical sectored FAT

When hard disks grew larger and the FAT12 and FAT16 file system implementation in MS-DOS / PC DOS did not provide means to take advantage of the extra storage, several manufacturers developed their own FAT variants to address the problem in their MS-DOS OEM issues.[30]

length	255 UCS-2 characters[nb 1] when using LFN
Max. directory depth	32 levels or 66 characters (with CDS), 60 levels or more (without CDS)
Features	
Dates recorded	Modified date (not with 86-DOS before 0.42), modified time (not with PC DOS 1.0 and 86-DOS), creation date/time (DOS 7.0 and higher only), access date (only available with ACCDATE enabled),[2] deletion date/time (only with DELWATCH 2)
Date range	1980-01-01 to 2099-12-31 (2107-12-31)
Date resolution	2 seconds for last modified time, 10 ms for creation time, 1 day for access date, 2 seconds for deletion time
Attributes	Read-only (since DOS 2.0), Hidden, System, Volume (since MS-DOS 1.28 and PC DOS 2.0), Directory (since MS-DOS 1.40 and PC DOS 2.0), Archive (since DOS 2.0)
File system permissions	File, directory and volume access rights for Read,

Some vendors (AST and NEC[30]) supported eight, instead of the standard four, primary partition entries in their custom extended *Master Boot Record* (*MBR*), and they adapted MS-DOS to use more than a single primary partition.

Other vendors worked around the volume size limits imposed by the 16-bit sector entries and arithmetics by increasing the *size* of the sectors the file system dealt with, thereby blowing up dimensions.

These so-called *logical sectors* were larger (up to 8192 bytes) than the *physical sector* size (still typically 512 bytes) as expected by the ROM-BIOS INT 13H or the disk drive hardware. The DOS-BIOS or System BIOS would then combine multiple physical sectors into logical sectors for the file system to work with. These changes were transparent to the file system implementation in the DOS kernel, since on the file system's abstraction level volumes are seen as a linear series of logically addressable sectors, also known as *absolute sectors* (addressed by their *Logical Sector Number* (*LSN*), starting with LSN 0) independent of the physical location of the volume on the physical medium and its geometry. The underlying DOS-BIOS translated these logical sectors into physical sectors according to partitioning information and the drive's physical geometry.

The drawback of this approach was a less memory-efficient sector buffering and deblocking in the DOS-BIOS, thereby causing an increased memory footprint for the DOS data structures. Since older DOS versions were not flexible enough to work with these logical geometries, the OEMs had to introduce new partition IDs for their FAT variants in order to hide them from off-the-shelf issues of MS-DOS and PC DOS. Known partition IDs for logical sectored FATs include: `0x08` (Commodore MS-DOS 3.x), `0x11` (Leading Edge MS-DOS 3.x), `0x14` (AST MS-DOS 3.x), `0x24` (NEC MS-DOS 3.30[30]), `0x56` (AT&T MS-DOS 3.x), `0xE5` (Tandy MS-DOS), `0xF2` (Sperry IT MS-DOS 3.x, Unisys MS-DOS 3.3 – also used by Digital Research DOS Plus 2.1).[31] OEM versions like Toshiba MS-DOS, Wyse MS-DOS 3.2 and 3.3,[32] as well as Zenith MS-DOS are also known to have utilized logical sectoring.[33]

While non-standard and sub-optimal, these FAT variants are perfectly valid according to the specifications of the file system itself. Therefore, even if default issues of MS-DOS and PC DOS were not able to cope with them, most of these vendor-specific FAT12 and FAT16 variants can be mounted by more flexible file system implementations in operating systems such as DR-DOS, simply by changing the partition ID to one of the recognized types.[nb 6] Also, if they no longer need to be recognized by their original operating systems, existing partitions can be "converted" into FAT12 and FAT16 volumes more compliant with versions of MS-DOS/PC DOS 4.0–6.3, which do not support sector sizes different from 512 bytes,[34] by switching to a BPB with 32-bit entry for the number of sectors, as introduced since DOS 3.31 (see FAT16B below), keeping the cluster size and reducing the logical sector size in the BPB down to

	Write, Execute, Delete only with DR-DOS, PalmDOS, Novell DOS, OpenDOS, FlexOS, 4680 OS, 4690 OS, Concurrent DOS, Multiuser DOS, System Manager, REAL/32 (Execute right only with FlexOS, 4680 OS, 4690 OS; individual file / directory passwords not with FlexOS, 4680 OS, 4690 OS; World/Group/Owner permission classes only with multiuser security loaded)
Transparent compression	Per-volume, SuperStor, Stacker, DoubleSpace, DriveSpace
Transparent encryption	Per-volume only with DR-DOS

FAT16

Developer(s)	Microsoft, IBM, Digital Research, Novell
Full name	16-bit File Allocation Table (with 16-bit sector entries)
Introduced	1984-08-14 (PC DOS 3.0) 1984-08 (MS-DOS 3.0)
Partition identifier	MBR/EBR: FAT16: `0x04` e.a. BDP: `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`

512 bytes, while at the same time increasing the counts of logical sectors per cluster, reserved logical sectors, total logical sectors, and logical sectors per FAT by the same factor.

A parallel development in MS-DOS / PC DOS which allowed an increase in the maximum possible FAT size was the introduction of multiple FAT partitions on a hard disk. To allow the use of more FAT partitions in a compatible way, a new partition type was introduced in PC DOS 3.2 (1986), the *extended partition* (EBR),[15] which is a container for an additional partition called *logical drive*. Since PC DOS 3.3 (April 1987), there is another, optional extended partition containing the next *logical drive*, and so on. The MBR of a hard disk can either define up to four primary partitions, or an extended partition in addition to up to three primary partitions.

Final FAT16

In November 1987, Compaq Personal Computer DOS 3.31 (a modified OEM version of MS-DOS 3.3 released by Compaq with their machines) introduced what today is simply known as *the FAT16* format, with the expansion of the 16-bit disk sector count to 32 bits in the BPB. Although the on-disk changes were minor, the entire DOS disk driver had to be converted to use 32-bit sector numbers, a task complicated by the fact that it was written in 16-bit assembly language. The result was initially called the *DOS 3.31 Large File System*. Microsoft's DSKPROBE tool refers to type `0x06` as *BigFAT*,[36] whereas some older versions of FDISK described it as *BIGDOS*. Technically, it is known as **FAT16B**.

Since older versions of DOS were not designed to cope with more than 65535 sectors, it was necessary to introduce a new partition type for this format in order to hide it from pre-3.31 issues of DOS. The original form of FAT16 (with less than 65536 sectors) had a partition type `0x04`. To deal with disks larger than this, type `0x06` was introduced to indicate 65536 or more sectors. In addition to this, the disk driver was expanded to cope with more than 65535 sectors as well. The only other difference between the original FAT16 and the newer FAT16B format is the usage of a newer BPB format with 32-bit sector entry. Therefore, newer operating systems supporting the FAT16B format can cope also with the original FAT16 format without any necessary changes.

If partitions to be used by pre-DOS 3.31 issues of DOS need to be created by modern tools, the only criteria theoretically necessary to meet are a sector count of less than 65536, and the usage of the old partition ID (`0x04`). In practice however, type `0x01` and `0x04` primary partitions should not be physically located outside the first 32 MiB of the disk, due to other restrictions in MS-DOS 2.x, which could not cope with them otherwise.

Limits	
Max. file size	limited by volume size
File size granularity	1 byte
Max. number of files	65,536 for 32 KiB clusters
Max. filename length	8.3 filename with OEM characters, 255 UCS-2 characters[nb 1] when using LFN
Max. directory depth	32 levels or 66 characters (with CDS), 60 levels or more (without CDS)
Features	
Dates recorded	Modified date/time, creation date/time (DOS 7.0 and higher only), access date (only available with ACCDATE enabled),[2] deletion date/time (only with DELWATCH 2)
Date range	1980-01-01 to 2099-12-31 (2107-12-31)
Date resolution	2 seconds for last modified time, 10 ms for creation time, 1 day for access date, 2 seconds for deletion time
Attributes	Read-only, Hidden, System, Volume, Directory, Archive
File system permissions	File, directory and volume access rights for Read,

In 1988, the FAT16B improvement became more generally available through DR DOS 3.31, PC DOS 4.0, OS/2 1.1, and MS-DOS 4.0. The limit on partition size was dictated by the 8-bit signed count of sectors per cluster, which originally had a maximum power-of-two value of 64. With the standard hard disk sector size of 512 bytes, this gives a maximum of 32 KiB cluster size, thereby fixing the "definitive" limit for the FAT16 partition size at 2 GiB for sector size 512. On magneto-optical media, which can have 1 or 2 KiB sectors instead of 0.5 KiB, this size limit is proportionally larger.

Much later, Windows NT increased the maximum cluster size to 64 KiB, by considering the sectors-per-cluster count as unsigned. However, the resulting format was not compatible with any other FAT implementation of the time, and it generated greater internal fragmentation. Windows 98, SE and ME also supported reading and writing this variant, but its disk utilities did not work with it and some FCB services are not available for such volumes. This contributes to a confusing compatibility situation.

Prior to 1995, versions of DOS accessed the disk via CHS addressing only. When MS-DOS 7.0 / Windows 95 introduced LBA disk access, partitions could start being physically located outside the first ca. 8 GiB of this disk and thereby out of the reach of the traditional CHS addressing scheme. Partitions partially or fully located beyond the CHS barrier therefore had to be hidden from non-LBA-enabled operating systems by using the new partition type `0x0E` in the partition table instead. FAT16 partitions using this partition type are also named **FAT16X**.[37] The only difference, compared to previous FAT16 partitions, is the fact that some CHS-related geometry entries in the BPB record, namely the number of sectors per track and the number of heads, may contain no or misleading values and should not be used.

The number of root directory entries available for FAT12 and FAT16 is determined when the volume is formatted, and is stored in a 16-bit field. For a given number `RDE` and sector size `SS`, the number `RDS` of root directory sectors is `RDS=ceil((RDE×32)/SS)`, and `RDE` is normally chosen to fill these sectors, i.e., `RDE*32=RDS*SS`. FAT12 and FAT16 media typically use 512 root directory entries on non-floppy media. Some third-party tools, like mkdosfs, allow the user to set this parameter.[38]

FAT32

In order to overcome the volume size limit of FAT16, while at the same time allowing DOS real-mode code to handle the format, Microsoft designed a new version of the file system, **FAT32**, which supported an increased number of possible clusters, but could reuse most of the existing code, so that the conventional memory footprint was increased by less than 5 KiB under DOS.[39] Cluster values are represented by 32-bit numbers, of which 28 bits are used to hold the cluster number. The boot sector uses a 32-bit field for the sector count, limiting the FAT32 volume size to 2 TiB for a sector size of

	Write, Execute, Delete only with DR-DOS, PalmDOS, Novell DOS, OpenDOS, FlexOS, 4680 OS, 4690 OS, Concurrent DOS, Multiuser DOS, System Manager, REAL/32 (Execute right only with FlexOS, 4680 OS, 4690 OS; individual file / directory passwords not with FlexOS, 4680 OS, 4690 OS; World/Group/Owner permission classes only with multiuser security loaded)
Transparent compression	Per-volume, SuperStor, Stacker, DoubleSpace, DriveSpace
Transparent encryption	Per-volume only with DR-DOS

FAT16B

Developer(s)	Compaq, Digital Research, IBM, Microsoft, Novell
Full name	16-bit File Allocation Table (with 32-bit sector entries)
Introduced	1987-11 (Compaq MS-DOS 3.31) 1988-06-28 (DR DOS 3.31) 1988 (IBM DOS 4.0) 1988 (OS/2 1.1) 1988 (MS-DOS 4.0)
Partition identifier	MBR/EBR: FAT16B: `0x06` `0x0E`

512 bytes and 16 TiB for a sector size of 4,096 bytes.[40][41] FAT32 was introduced with MS-DOS 7.1 / Windows 95 OSR2 in 1996, although reformatting was needed to use it, and DriveSpace 3 (the version that came with Windows 95 OSR2 and Windows 98) never supported it. Windows 98 introduced a utility to convert existing hard disks from FAT16 to FAT32 without loss of data. In the Windows NT line, native support for FAT32 arrived in Windows 2000. A free FAT32 driver for Windows NT 4.0 was available from Winternals, a company later acquired by Microsoft. The acquisition of the driver from official sources is no longer possible. Since 1998, Caldera's dynamically loadable DRFAT32 driver could be used to enable FAT32 support in DR-DOS.[42][43] The first version of DR-DOS to natively support FAT32 and LBA access was OEM DR-DOS 7.04 in 1999. That same year IMS introduced native FAT32 support with REAL/32 7.90, and IBM 4690 OS added FAT32 support with version 2.[44] Ahead Software provided another dynamically loadable FAT32.EXE driver for DR-DOS 7.03 with Nero Burning ROM in 2004. IBM PC DOS introduced native FAT32 support with OEM PC DOS 7.10 in 2003.

The maximum possible size for a file on a FAT32 volume is 4 GiB minus 1 byte or 4,294,967,295 ($2^{32} - 1$) bytes. This limit is a consequence of the file length entry in the directory table and would also affect huge FAT16 partitions with a sufficient sector size.[1] Large video files, DVD images and databases often exceed this limit.

As with previous file systems, the design of the FAT32 file system does not include direct built-in support for long filenames, but FAT32 volumes can optionally hold VFAT long filenames in addition to short filenames in exactly the same way as VFAT long filenames have been optionally implemented for FAT12 and FAT16 volumes.

Two partition types have been reserved for FAT32 partitions, `0x0B` and `0x0C`. The latter type is also named **FAT32X** in order to indicate usage of LBA disk access instead of CHS.[42][45][46][47][48] On such partitions, CHS-related geometry entries, namely the CHS sector addresses in the MBR as well as the number of sectors per track and the number of heads in the EBPB record, may contain no or misleading values and should not be used.[49][47][48]

	(LBA), e.a. BDP: `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`
Limits	
Min. volume size	8 MiB (with 128 byte sectors) 32 MiB (with 512 byte sectors) 256 MiB (with 4 KiB sectors)
Max. volume size	2 GiB (with 32 KiB clusters) 4 GiB (with 64 KiB clusters) (NT 4, PTS-DOS, EDR-DOS) 8 GiB (with 128 KiB clusters and 1 or 2 KiB sectors) (NT 4 and EDR-DOS only) 8 GiB (with 128 KiB clusters and 512 byte sectors) (EDR-DOS only) 16 GiB (with 256 KiB clusters and 2 or 4 KiB sectors) (NT 4 only)
Max. file size	2,147,483,647 bytes (2 GiB – 1) (without LFS) 4,294,967,295 bytes (4 GiB – 1) (with LFS) limited by volume size only (with FAT16+[35])
File size granularity	1 byte
Max. number of files	65,460 for 32 KiB clusters
Max.	8.3 filename with

Extensions

Extended Attributes

OS/2 heavily depends on extended attributes (EAs) and stores them in a hidden file called "`EA␣DATA.␣SF`" in the root directory of the FAT12 or FAT16 volume. This file is indexed by two previously reserved bytes in the file's (or directory's) directory entry at offset `0x14`.[50] In the FAT32 format, these bytes hold the upper 16 bits of the starting cluster number of the file or directory, hence making it impossible to store OS/2 EAs on FAT32 using this method.

However, the third-party FAT32 installable file system (IFS) driver FAT32.IFS version 0.70 and higher by Henk Kelder & Netlabs for OS/2 and eComStation stores extended attributes in extra files with filenames having the string "␣EA.␣SF" appended to the regular filename of the file to which they belong. The driver also utilizes the byte at offset 0x0C in directory entries to store a special mark byte indicating the presence of extended attributes to help speed up things.[51][52] (This extension is critically incompatible with the FAT32+ method to store files larger than 4 GiB minus 1 on FAT32 volumes.)[35]

Extended attributes are accessible via the Workplace Shell desktop, through REXX scripts, and many system GUI and command-line utilities (such as 4OS2).[53]

To accommodate its OS/2 subsystem, Windows NT supports the handling of extended attributes in HPFS, NTFS, FAT12 and FAT16. It stores EAs on FAT12, FAT16 and HPFS using exactly the same scheme as OS/2, but does not support any other kind of ADS as held on NTFS volumes. Trying to copy a file with any ADS other than EAs from an NTFS volume to a FAT or HPFS volume gives a warning message with the names of the ADSs that will be lost. It does not support the FAT32.IFS method to store EAs on FAT32 volumes.

Windows 2000 onward acts exactly as Windows NT, except that it ignores EAs when copying to FAT32 without any warning (but shows the warning for other ADSs, like "Macintosh Finder Info" and "Macintosh Resource Fork").

Cygwin uses "EA␣DATA.␣SF" files as well.

Long file names

One of the user experience goals for the designers of Windows 95 was the ability to use long filenames (LFNs—up to 255 UTF-16 code units long)[nb 1], in addition to classic 8.3 filenames (SFNs). For backward and forward compatibility LFN were implemented as an optional extension on top of the existing FAT file system structures using a workaround in the way directory entries are laid out.

This transparent method to store long file names in the existing FAT file systems without altering their data structures is usually known as **VFAT** (for "Virtual FAT") after the Windows 95 virtual device driver.[nb 2]

Non VFAT-enabled operating systems can still access the files under their short file name alias without restrictions, however, the associated long file names may get lost, when files with long file names are copied under non VFAT-aware operating systems.

In Windows NT, support for VFAT long filenames started from version 3.5.

filename length	OEM characters, 255 UCS-2 characters[nb 1] when using LFN
Max. directory depth	32 levels or 66 characters (with CDS), 60 levels or more (without CDS)
Features	
Dates recorded	Modified date/time, creation date/time (DOS 7.0 and higher only), access date (only available with ACCDATE enabled),[2] deletion date/time (only with DELWATCH 2)
Date range	1980-01-01 to 2099-12-31 (2107-12-31)
Date resolution	2 seconds for last modified time, 10 ms for creation time, 1 day for access date, 2 seconds for deletion time
Attributes	Read-only, Hidden, System, Volume, Directory, Archive
File system permissions	File, directory and volume access rights for Read, Write, Execute, Delete only with DR-DOS, PalmDOS, Novell DOS, OpenDOS, FlexOS, 4680 OS, 4690 OS, Concurrent DOS, Multiuser DOS, System Manager, REAL/32 (Execute

GNU/Linux provides a VFAT filesystem driver to work with FAT volumes with VFAT long filenames. For some while, a UVFAT driver was available to provide combined support for UMSDOS-style permissions with VFAT long filenames.

OS/2 added long filename support to FAT using extended attributes (EA) before the introduction of VFAT; thus, VFAT long filenames are invisible to OS/2, and EA long filenames are invisible to Windows, therefore experienced users of both operating systems would have to manually rename the files.

Human68K supported up to 18.3 filenames and (Shift JIS) Kanji characters in a proprietary FAT file system variant.

In order to support Java applications, the FlexOS-based IBM 4690 OS version 2 introduced its own virtual file system (VFS) architecture to store long filenames in the FAT file system in a backwards compatible fashion. If enabled, the virtual filenames (VFN) are available under separate logical drive letters, whereas the real filenames (RFN) remain available under the original drive letters.[54]

Forks and Alternate Data Streams

The FAT file system itself is not designed for supporting Alternate Data Streams (ADS), but some operating systems that heavily depend on them have devised various methods for handling them on FAT volumes. Such methods either store the additional information in extra files and directories (classic Mac OS and macOS), or give new semantics to previously unused fields of the FAT on-disk data structures (OS/2 and Windows NT).

Mac OS using PC Exchange stores its various dates, file attributes and long filenames in a hidden file called "`FINDER.DAT`", and resource forks (a common Mac OS ADS) in a subdirectory called "`RESOURCE.FRK`", in every directory where they are used. From PC Exchange 2.1 onwards, they store the Mac OS long filenames as standard FAT long filenames and convert FAT filenames longer than 31 characters to unique 31-character filenames, which can then be made visible to Macintosh applications.

macOS stores resource forks and metadata (file attributes, other ADS) using AppleDouble format in a hidden file with a name constructed from the owner filename prefixed with "`._`", and Finder stores some folder and file metadata in a hidden file called "`.DS_Store`" (but note that Finder uses `.DS_Store` even on macOS' native filesystem, HFS+).

UMSDOS permissions and filenames

Early GNU/Linux distributions also supported a format known as UMSDOS, a FAT variant with Unix file attributes (such as long file name and access permissions) stored in a separate file called "`--linux-.---`". UMSDOS fell into disuse after VFAT was released and it is not enabled by default in Linux

	right only with FlexOS, 4680 OS, 4690 OS; individual file / directory passwords not with FlexOS, 4680 OS, 4690 OS; World/Group/Owner permission classes only with multiuser security loaded)
Transparent compression	Per-volume, SuperStor, Stacker, DoubleSpace, DriveSpace
Transparent encryption	Per-volume only with DR-DOS

FAT32

Developer(s)	Microsoft, Caldera
Full name	32-bit File Allocation Table (with 28-bit cluster entries)
Introduced	August 1996 (Windows 95 OSR2)
Partition identifier	MBR/EBR: FAT32: `0x0B` `0x0C` (LBA), e.a. BDP: `EBD0A0A2-B9E5-4433-87C0-68B6B72699C7`
Limits	
Min. volume size	32 MiB-4.5 KiB (with 65525 clusters and 512 byte sectors) 256 MiB-36 KiB (with 65525 clusters and 4 KiB sectors)
Max. volume size	2 TiB (with 512 byte sectors) 8 TiB (with 2 KiB sectors and 32 KiB

from version 2.5.7 onwards.[55] For some time, Linux also provided combined support for UMSDOS-style permissions and VFAT long filenames through UVFAT.

Derivatives

Turbo FAT

In its NetWare File System (NWFS) Novell implemented a heavily modified variant of a FAT file system for the NetWare operating system. For larger files it utilized a performance feature named Turbo FAT.

FATX

FATX is a family of file systems designed for Microsoft's Xbox video game console hard disk drives and memory cards,[56][57] introduced in 2001.

While resembling the same basic design ideas as FAT16 and FAT32, the **FATX16** and **FATX32** on-disk structures are simplified but fundamentally incompatible with normal FAT16 and FAT32 file systems, making it impossible for normal FAT file system drivers to mount such volumes.

The non-bootable superblock sector is 4 KiB in size and holds an 18 byte large BPB-like structure completely different from normal BPBs. Clusters are typically 16 KiB in size and there is only one copy of the FAT on the Xbox. Directory entries are 64 bytes in size instead of the normal 32 bytes. Files can have filenames up to 42 characters long using the OEM character set and be up to 4 GiB minus 1 byte in size. The on-disk timestamps hold creation, modification and access dates and times but differ from FAT: in FAT, the epoch is 1980; in FATX, the epoch is 2000. On the Xbox 360, the epoch is 1980.[58]

exFAT

exFAT is a file system introduced with Windows Embedded CE 6.0 in November 2006 and brought to the Windows NT family with Vista Service Pack 1 and Windows XP Service Pack 3 (and/or separate installation of Windows XP Update KB955704). It is loosely based on the File Allocation Table architecture, but incompatible, proprietary and protected by patents.[59]

exFAT is intended for use on flash drives (such as SDXC and Memory Stick XC), where FAT32 is otherwise used. Microsoft's GUI and command-line format utilities offer it as an alternative to NTFS (and, for smaller partitions, to FAT16B and FAT32). The MBR partition type is `0x07` (the same as used for IFS, HPFS, and NTFS). Logical geometry information located in the VBR is stored in a format not resembling any kind of BPB.

	clusters) 16 TiB (with 4 KiB sectors and 64 KiB clusters)
Max. file size	2,147,483,647 bytes (2 GiB – 1) (without LFS) 4,294,967,295 bytes (4 GiB – 1)[1] (with LFS) 274,877,906,943 bytes (256 GiB – 1) (only with FAT32+[35])
File size granularity	1 byte
Max. number of files	268,173,300 for 32 KiB clusters
Max. filename length	8.3 filename with OEM characters, 255 UCS-2 characters[nb 1] when using LFN
Max. directory depth	32 levels or 66 characters (with CDS), 60 levels or more (without CDS)
Features	
Dates recorded	Modified date/time, creation date/time (DOS 7.0 and higher only), access date (only available with ACCDATE enabled),[2] deletion date/time (only with DELWATCH 2)
Date range	1980-01-01 to 2099-12-31 (2107-12-31)
Date resolution	2 seconds for last modified time, 10 ms for creation time,

FAT+

In 2007 the open **FAT+** draft proposed how to store larger files up to 256 GiB minus 1 byte or 274,877,906,943 (2^{38} – 1) bytes on slightly modified and otherwise backward-compatible FAT32 volumes,[35] but imposes a risk that disk tools or FAT32 implementations not aware of this extension may truncate or delete files exceeding the normal FAT32 file size limit. Support for **FAT32+** and **FAT16+** is limited to some versions of DR-DOS and not available in mainstream operating systems.[60] (This extension is critically incompatible with the `/EAS` option of the FAT32.IFS method to store OS/2 extended attributes on FAT32 volumes.)

	1 day for access date, 2 seconds for deletion time
Attributes	Read-only, Hidden, System, Volume, Directory, Archive
File system permissions	Partial, only with DR-DOS, REAL/32 and 4690 OS
Transparent compression	No
Transparent encryption	No

Patents

Microsoft applied for, and was granted, a series of patents for key parts of the FAT file system in the mid-1990s. All four pertain to long-filename extensions to FAT first seen in Windows 95: U.S. patent 5,579,517,[61] U.S. patent 5,745,902,[62] U.S. patent 5,758,352,[63] U.S. patent 6,286,013.[64]

On December 3, 2003 Microsoft announced[65] that it would be offering licenses for use of its FAT specification and "associated intellectual property", at the cost of a US$0.25 royalty per unit sold, with a $250,000 maximum royalty per license agreement.[66] To this end, Microsoft cited four patents on the FAT file system as the basis of its intellectual property claims.

In the EFI FAT32 specification[7] Microsoft specifically grants a number of rights, which many readers have interpreted as permitting operating system vendors to implement FAT.[67]

Non-Microsoft patents affecting FAT include: U.S. patent 5,367,671, specific to the OS/2 extended object attributes (expired in 2011).[68]

Challenges and lawsuits

The Public Patent Foundation (PUBPAT) submitted evidence to the US Patent and Trademark Office (USPTO) in 2004 disputing the validity of U.S. patent 5579517,[61] including prior art references from Xerox and IBM.[69] The USPTO opened an investigation and concluded by rejecting all claims in the patent.[70] The next year, the USPTO further announced that following the re-examination process, it affirmed the rejection of '517 and additionally found U.S. patent 5,758,352[63] invalid on the grounds that the patent had incorrect assignees.

However, in 2006 the USPTO ruled that features of Microsoft's implementation of the FAT system were "novel and non-obvious", reversing both earlier decisions and leaving the patents valid.[71]

In February 2009, Microsoft filed a patent infringement lawsuit against TomTom alleging that the device maker's products infringe on patents related to VFAT long filenames. As some TomTom products are based on GNU/Linux, this marked the first time that Microsoft tried to enforce its patents against the GNU/Linux platform.[72] The lawsuit was settled out of court the following month with an agreement that Microsoft be given access to four of TomTom's patents, that TomTom will drop support for the VFAT long filenames from its products, and that in return Microsoft not seek legal action against TomTom for the five-year duration of the settlement agreement.[73]

In October 2010, Microsoft filed a patent infringement lawsuit against Motorola alleging several patents (including two of the VFAT patents) were not licensed for use in the Android operating system.[74] They also submitted a complaint to the ITC.[75] Developers of open source software have designed methods intended to circumvent Microsoft's patents.[76][77]

In 2013, patent EP0618540 "common name space for long and short filenames" was invalidated in Germany.[78]

See also

- Comparison of file systems
- Design of the FAT file system
- Drive letter assignment
- List of file systems
- Transaction-Safe FAT File System

Notes

1. Since Windows 2000, Microsoft Windows uses UTF-16 instead of UCS-2 for the internal "Unicode". In UTF-16, a "character" (code point) may take up two code units.
2. A driver named VFAT appeared before Windows 95, in Windows for Workgroups 3.11, but this older version was only used for implementing 32-bit file access and did not support long file names.
3. Windows XP has been observed to create similar hybrid disks when reformatting FAT16B formatted ZIP-100 disks to FAT32 format. The resulting volumes were FAT32 by format, but still used the FAT16B EBPB. (It is unclear how Windows determines the location of the root directory on FAT32 volumes, if only a FAT16 EBPB was used.)
4. Sources differ in regard to the first NCR data entry terminal integrating support for the FAT file system. According to Stephen Manes and Paul Andrews, "Gates", development was for a NCR 8200 in late 1977, incorrectly classified as a floppy-based upgrade to the NCR 7200, which had been released in 1975-11 (model I and IV) and was built around an Intel 8080 8-bit processor, but was cassette-based only. However, the NCR Century 8200 was a 16-bit minicomputer, onto which several data entry terminals could be hooked up. Marc McDonald even remembered a NCR 8500, a mainframe of the Criterion series, which can be ruled out as well. Announced 1977-10 for shipment in 1978-02, NCR also introduced the NCR I-8100 series including the 8080-based NCR I-8130 and NCR I-8150 models of small business systems featuring dual floppy disks. Other sources indicate that either the NCR 7200 series itself or the successor series were the actual target platform. NCR Basic Plus 6 (based on Microsoft Extended BASIC-80) became available for the cassette-based NCR 7200 model VI in Q1/1977. The NCR 7500 series was released in 1978, based on a similar 8080 hardware, but now including NCR 7520 and 7530 models featuring 8-inch diskettes. NCR Basic +6, a precursor or adaptation of Standalone Disk BASIC-80 was available for them at least since 1979. One source claims that a special NCR 7200 model variant with two 8-inch diskettes and Microsoft BASIC existed and was imported by NCR Sydney into Australia the least.
5. See FAT end-of-chain marker for special precautions in regard to occurrences of a cluster value of `0xFF0` on FAT12 volumes under MS-DOS/PC DOS 3.3 and higher.
6. DR-DOS is able to boot off FAT12/FAT16 logical sectored media with logical sector sizes up to 1024 bytes.

References

1. "File Systems" (https://technet.microsoft.com/en-us/library/cc938937.aspx). Microsoft TechNet. 2001. Retrieved 2011-07-31.
2. Microsoft (2006-11-15). Windows 95 CD-ROM CONFIG.TXT File (http://c-bit.org/kb/135481/EN-US/) Article 135481, Revision: 1.1, retrieved 2011-12-22: "For each hard disk, specifies whether to record the date that files are last accessed. Last access dates are turned off for all drives when your computer is started in safe mode, and are not maintained for floppy disks by default. Syntax: `ACCDATE=drive1+|- [drive2+|-]...`"

3. "FAT File System (Windows Embedded CE 6.0)" (http://msdn.microsoft.com/en-us/library/ee489982%28v=winembedded.60%29.aspx). Microsoft. January 6, 2010. Retrieved 2013-07-07.
4. "A brief introduction to FAT (File Allocation Table) formats | Wizcode's articles | HowTos, Guides, Hints and Tips, Articles" (https://web.archive.org/web/20150925082826/http://www.wizcode.com/articles/comments/a-brief-introduction-to-fat-file-allocation-table/). *www.wizcode.com*. Archived from the original (http://www.wizcode.com/articles/comments/a-brief-introduction-to-fat-file-allocation-table/) on September 25, 2015. Retrieved September 24, 2015.
5. "Comparing NTFS and FAT file systems" (http://windows.microsoft.com/en-us/windows-vista/comparing-ntfs-and-fat-file-systems). Microsoft. Retrieved 2014-01-27.
6. JEIDA/JEITA/CIPA (2010). "Standard of the Camera & Imaging Products Association, CIPA DC-009-Translation-2010, Design rule for Camera File system: DCF Version 2.0 (Edition 2010)" (https://web.archive.org/web/20130930190707/http://www.cipa.jp/english/hyoujunka/kikaku/pdf/DC-009-2010_E.pdf) (PDF). Archived from the original (http://www.cipa.jp/english/hyoujunka/kikaku/pdf/DC-009-2010_E.pdf) (PDF) on September 30, 2013. Retrieved 2011-04-13.
7. "Microsoft Extensible Firmware Initiative FAT32 File System Specification, FAT: General Overview of On-Disk Format" (http://download.microsoft.com/download/1/6/1/161ba512-40e2-4cc9-843a-923143f3456c/fatgen103.doc). Microsoft. March 30, 2011. Retrieved 2018-12-21.
8. "Volume and File Structure of Disk Cartridges for Information Interchange" (http://www.ecma-international.org/publications/standards/Ecma-107.htm). *Standard ECMA-107 (2nd ed., June 1995)*. ECMA. 1995. Retrieved 2011-07-30.
9. "Information technology – Volume and file structure of disk cartridges for information interchange" (http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=21273). *ISO/IEC 9293:1994*. ISO catalogue. 1994. Retrieved 2012-01-06.
10. "Information processing – Volume and file structure of flexible disk cartridges for information interchange" (http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=16948). *ISO 9293:1987*. ISO catalogue. 1987. Retrieved 2012-01-06.
11. Reynolds, Aaron R.; Adler, Dennis R.; Lipe, Ralph A.; Pedrizetti, Ray D.; Parsons, Jeffrey T.; Arun, Rasipuram V. (May 26, 1998). "Common name space for long and short filenames" (http://www.google.com/patents?id=bUohAAAAEBAJ). *US Patent 5758352*. Retrieved 2012-01-19.
12. Chappell, Geoff; (1994); *DOS Internals*, Addison Wesley, ISBN 0-201-60835-9, ISBN 978-0-201-60835-9
13. *Xerox BASIC-80 – basic-80 reference manual* (http://bitsavers.trailing-edge.com/pdf/xerox/820-II/BASIC-80_5.0.pdf) (PDF). 5.0. Microsoft, Xerox. 1979. 610P70641. Retrieved 2014-06-02. (NB. For Microsoft (Standalone Disk / Disk / Extended / 8K) BASIC-80, (Standalone Disk / Extended) BASIC-86, BASIC Compiler, release 5.0)
14. *MICROSOFT BASIC-80 version 5.0 reference manual / BASIC-80 Interpreter and Compiler Addendum Release 5.1* (http://bitsavers.trailing-edge.com/pdf/xerox/820-II/BASIC-80_5.0.pdf) (PDF). 5.1. Microsoft. 1979. Retrieved 2014-06-02. (NB. For Microsoft (Standalone Disk / Disk / Extended / 8K) BASIC-80, (Standalone Disk / Extended) BASIC-86, BASIC Compiler, release 5.1)
15. Zbikowski, Mark; Allen, Paul; Ballmer, Steve; Borman, Reuben; Borman, Rob; Butler, John; Carroll, Chuck; Chamberlain, Mark; Chell, David; Colee, Mike; Courtney, Mike; Dryfoos, Mike; Duncan, Rachel; Eckhardt, Kurt; Evans, Eric; Farmer, Rick; Gates, Bill; Geary, Michael; Griffin, Bob; Hogarth, Doug; Johnson, James W.; Kermaani, Kaamel; King, Adrian; Koch, Reed; Landowski, James; Larson, Chris; Lennon, Thomas; Lipkie, Dan; McDonald, Marc; McKinney, Bruce; Martin, Pascal; Mathers, Estelle; Matthews, Bob; Melin, David; Mergentime, Charles; Nevin, Randy; Newell, Dan; Newell, Tani; Norris, David; O'Leary, Mike; O'Rear, Bob; Olsson, Mike; Osterman, Larry; Ostling, Ridge; Pai, Sunil; Paterson, Tim; Perez, Gary; Peters, Chris; Petzold, Charles; Pollock, John; Reynolds, Aaron; Rubin, Darryl; Ryan, Ralph; Schulmeisters, Karl; Shah, Rajen; Shaw, Barry; Short, Anthony; Slivka, Ben; Smirl, Jon; Stillmaker, Betty; Stoddard, John; Tillman, Dennis; Whitten, Greg; Yount, Natalie; Zeck, Steve (1988). "Technical advisors". *The MS-DOS Encyclopedia: versions 1.0 through 3.2*. By Duncan, Ray; Bostwick, Steve; Burgoyne, Keith; Byers, Robert A.; Hogan, Thom; Kyle, Jim; Letwin, Gordon; Petzold, Charles; Rabinowitz, Chip; Tomlin, Jim; Wilton, Richard; Wolverton, Van; Wong, William; Woodcock, JoAnne (Completely reworked ed.). Redmond, Washington, USA: Microsoft Press. ISBN 1-55615-049-0. LCCN 87-21452 (https://lccn.loc.gov/87-21452). OCLC 16581341 (https://www.worldcat.org/oclc/16581341). (xix+1570 pages; 26 cm) (NB. This edition was published in 1988 after extensive rework of the withdrawn 1986 first edition by a different team of authors. [1] (https://www.pcjs.org/pubs/pc/reference/microsoft/mspl13/msdos/encyclopedia/))

16. Manes, Stephen; Andrews, Paul (1993). *Gates: How Microsoft's Mogul Reinvented an Industry—and Made Himself the Richest Man in America*. Doubleday. ISBN 0-385-42075-7.
17. Hunter, David (1983). "Tim Paterson – The roots of DOS" (http://www.patersontech.com/dos/softalk.aspx). *Softalk for the IBM Personal Computer* (March 1983). Retrieved 2014-06-02.
18. Schulman, Andrew; Brown, Ralf; Maxey, David; Michels, Raymond J.; Kyle, Jim (1994). *Undocumented DOS – A programmer's guide to reserved MS-DOS functions and data structures – expanded to include MS-DOS 6, Novell DOS and Windows 3.1* (2 ed.). Addison Wesley. p. 11. ISBN 0-201-63287-X. ISBN 978-0-201-63287-3.
19. Paterson, Tim (September 30, 2007). "Design of DOS" (http://dosmandrivel.blogspot.com/2007/09/design-of-dos.html). *DosMan Drivel*. Retrieved 2011-07-04.
20. Seattle Computer Products (August 1980). "86-DOS – 8086 OPERATING SYSTEM – $95" (https://archive.org/stream/byte-magazine-1980-08/1980_08_BYTE_05-08_The_Forth_Language#page/n173/mode/2up). *Byte*. 8. p. 174. Retrieved 2013-08-18. (NB. The SCP advertisement already calls the product *86-DOS*, but does not mention a specific version number. Version 0.3 is known to be called 86-DOS already, so the name change must have taken place either for version 0.2 or immediately afterwards in August 1980.)
21. Seattle Computer Products (1981). "SCP 86-DOS 1.0 Addendum" (http://bitsavers.informatik.uni-stuttgart.de/pdf/seattleComputer/86-DOS_1.0_Addendum.pdf) (PDF). Retrieved 2013-03-10.
22. Wallace, James; Erickson, Jim; (1992); *Hard Drive: Bill Gates and the Making of the Microsoft Empire*, John Wiley & Sons, ISBN 0-471-56886-4
23. Norton, Peter; (1986); *Inside the IBM PC, Revised and Enlarged*, Brady, ISBN 0-89303-583-1, p. 157
24. Jenkinson, Brian; Sammes, A. J. (2000). *Forensic Computing: A Practitioner's Guide (Practitioner Series)*. Berlin: Springer. p. 157. ISBN 1-85233-299-9. "... only 2^12 (that is, 4096) allocation units or clusters can be addressed. In fact, the number is less than this, since 000h and 001h are not used and FF0h to FFFh are reserved or used for other purposes, leaving 002h to FEFh (2 to 4079) as the range of possible clusters."
25. Brouwer, Andries. "FAT under Linux" (http://www.win.tue.nl/~aeb/linux/fs/fat/fat-2.html).
26. Paterson, Tim (1983). "An Inside Look at MS-DOS" (https://web.archive.org/web/20110720115141/http://patersontech.com/Dos/Byte/InsideDos.htm). Byte. Archived from the original (http://www.patersontech.com/dos/Byte/InsideDos.htm#InsideDos_44) on July 20, 2011. Retrieved 2011-07-18. "The numbering starts with 2; the first two numbers, 0 and 1, are reserved."
27. IBM (1984). *IBM PC DOS 3.0 announcement letter*.
28. IBM (1985). *IBM PC DOS Technical Reference*. First Edition, P/N 6024181, dated February 1985.
29. Microsoft Knowledge Base article: "MS-DOS Partitioning Summary" (http://c-bit.org/kb/69912/EN-US/)
30. *FYI – Installing DR DOS on NEC DOS 3.3 Partitions* (http://cd.textfiles.com/netwaresl/NOV_INFO/RNW93/10JAN93.MON), Novell, January 5, 1993, FYI.M.1101, retrieved 2014-08-12
31. Brouwer, Andries. "List of partition identifiers for PCs" (http://www.win.tue.nl/~aeb/partitions/partition_types-1.html).
32. Microsoft (December 17, 2000); *Wyse DOS 3.3 Partitions Incompatible with MS-DOS 5.x and 6.x*, Document Q78407 ([2] (ftp://ftp.microsoft.com/misc1/PEROPSYS/MSDOS/KB/Q78/4/07.TXT))
33. Microsoft; (December 17, 2000); *Upgrading Pre-4.0 Systems with Logical Drive(s) > 32 MB*, Document Q68176 ([3] (ftp://ftp.microsoft.com/misc1/PEROPSYS/MSDOS/KB/Q68/1/76.TXT))
34. Brouwer, Andries. "Properties of partition tables" (http://www.win.tue.nl/~aeb/partitions/partition_types-2.html#ss2.6).
35. Kuhnt, Udo; Georgiev, Luchezar; Davis, Jeremy (2007). "*FAT+*, FATPLUS.TXT, draft revision 2" (http://www.fdos.org/kernel/fatplus.txt). Retrieved 2015-04-20.
36. "Dskprobe Overview: Data Recovery" (https://technet.microsoft.com/en-us/library/cc736327(v=ws.10).aspx). Microsoft TechNet. March 28, 2003. Retrieved 2011-08-03.
37. "Errors Creating Files or Folders in the Root Directory" (http://c-bit.org/kb/120138/EN-US/). Microsoft Help and Support. December 16, 2004. Retrieved 2006-10-14.
38. "mkdosfs man page" (http://www.die.net/doc/linux/man/man8/mkdosfs.8.html).
39. "Windows 98 Resource Kit – Chapter 10 – Disks and File Systems" (https://technet.microsoft.com/en-us/library/cc768180.aspx). Microsoft TechNet. 1998. Retrieved 2012-07-16.

40. "Limitations of FAT32 File System" (http://c-bit.org/kb/184006/EN-US/). Microsoft Knowledge Base. March 26, 2007. Retrieved 2011-08-21. "Clusters cannot be 64 kilobytes (KB) or larger"
41. "Limitations of the FAT32 File System in Windows XP" (http://c-bit.org/kb/314463/EN-US/). Microsoft Knowledge Base. December 1, 2007. Retrieved 2011-08-21.
42. *README.TXT – Caldera DR-DOS FAT32 Enabled Boot Disk (DRFAT32)*. Caldera, Inc. July 24, 1998.
43. *DRFAT32.SYS R1.00 INT 13h Interface for FAT32 Redirector*, Caldera, Inc., September 11, 1998
44. IBM; *4690 OS User's Guide Version 5.2*, IBM document SC30-4134-01, 2008-01-10 ([4] (ftp://ftp.software.ibm.com/software/retail/pubs/sw/opsys/4690/ver5r2/bsf1_UG_mst.pdf))
45. Karpowitz, Christina (September 23, 1998). "PowerQuest PartitionMagic 4.0 now available" (https://web.archive.org/web/19990208204638/http://www.powerquest.com/press/PM4available.html). PowerQuest. Archived from the original (http://www.powerquest.com/press/PM4available.html) on February 8, 1999. Retrieved 2015-04-17.
46. Livingston, Brian (October 28, 1998). "FAT-32X may operate differently than FAT-32 on large hard drives" (http://brianlivingston.com/windowmanager/archive/cgi-bin/new/livingst/981026bl.htm). **20** (43). Infoworld. Retrieved 2015-04-17.
47. Duitz, Neal (July 17, 2001). "Can anyone explain FAT32X?" (https://web.archive.org/web/20040613232209/http://www.win98private.net/fat32x.htm). Win98 Private FAQ, Windows 98 Consumer Preview Program. Archived from the original (http://www.win98private.net/fat32x.htm) on June 13, 2004. Retrieved 2015-04-17.
48. Costanzo, Lance (May 14, 1998). "FAT32X" (https://web.archive.org/web/19980521015015/http://lance.advantweb.com/fat32x/). Archived from the original (http://lance.advantweb.com/fat32x/) on May 21, 1998. Retrieved 2015-04-17.
49. Steinberg, David (May 1, 1998). "What is a FAT32X partition?" (http://www.sysopt.com/showthread.php?88915-What-is-FAT32X&p=532052&viewfull=1#post532052). *Tech Tip / FAQ*. PowerQuest Technical Support. Retrieved 2015-04-17.
50. Eager, Bob; Tavi Systems (October 28, 2000); *Implementation of extended attributes on the FAT file system* ([5] (http://www.tavi.co.uk/os2pages/eadata.html))
51. Kelder, Henk; (2003); *FAT32.TXT for FAT32.IFS version 0.9.13." ([6] (http://svn.netlabs.org/repos/fat32/branches/fat32-0.9/src/fat32.txt)): "This byte [...] is not modified while running Windows 95 and neighter* [sic] *by SCANDISK or DEFRAG. [...] If another program sets the value to `0x00` for a file that has EAs these EAs will no longer be found using DosFindFirst/Next calls only. The other OS/2 calls for retrieving EAs (DosQueryPathInfo, DosQueryFileInfo and DosEnumAttribute) do not rely on this byte. Also the opposite could [...] occur. [...] In this situation only the performance of directory scans will be decreased. Both situations [...] are corrected by CHKDSK".*
52. Kelder, Henk; *FAT32.TXT for FAT32.IFS version 0.74* ("Archived copy" (https://web.archive.org/web/20120330171510/http://macarlo.com/fat32v074.htm). Archived from the original (http://macarlo.com/fat32v074.htm) on March 30, 2012. Retrieved January 14, 2012.). Comment: This older version of the README file still discusses the old `0xEA` and `0xEC` magic values.
53. Eager, Bob (October 28, 2000). "Implementation of extended attributes on the FAT file system" (http://www.tavi.co.uk/os2pages/eadata.html). *Tavi OS/2 pages*. Retrieved 2006-10-14.
54. IBM; *4690 OS Programming Guide Version 5.2*, IBM document SC30-4137-01, 2007-12-06 ([7] (ftp://ftp.software.ibm.com/software/retail/pubs/sw/opsys/4690/ver5r2/bsi1_PG_mst.pdf))
55. "Release notes for v2.5.7" (https://www.kernel.org/pub/linux/kernel/v2.5/ChangeLog-2.5.7). The Linux Kernel archives. March 12, 2002. Retrieved 2006-10-14.
56. "FATX Specification" (http://www.free60.org/wiki/FATX). free60 wiki. Retrieved 2011-08-16.
57. de Quincey, Andrew; Murray-Pitts, Lucien (August 29, 2008). "Xbox partitioning and file system details" (https://web.archive.org/web/20100617020539/http://www.xbox-linux.org/wiki/Xbox_Partitioning_and_Filesystem_Details). 0.13. Xbox-Linux project. Archived from the original (http://www.xbox-linux.org/wiki/Xbox_Partitioning_and_Filesystem_Details) on June 17, 2010. Retrieved 2014-05-25.
58. Steil, Michael (February 26, 2008) [2003]. "Differences between Xbox FATX and MS-DOS FAT" (https://web.archive.org/web/20100617022009/http://www.xbox-linux.org/wiki/Differences_between_Xbox_FATX_and_MS-DOS_FAT). Xbox-Linux project. Archived from the original (http://www.xbox-linux.org/wiki/Differences_between_Xbox_FATX_and_MS-DOS_FAT) on June 17, 2010. Retrieved 2014-05-25.

59. Microsoft. "exFAT File System Intellectual Property licensing program" (https://web.archive.org/web/20130507183540/http://www.microsoft.com/en-us/legal/intellectualproperty/IPLicensing/Programs/exFATFileSystem.aspx). Archived from the original (http://www.microsoft.com/en-us/legal/intellectualproperty/IPLicensing/Programs/exFATFileSystem.aspx) on May 7, 2013. Retrieved 2013-04-23.
60. Udo Kuhnt (July 21, 2011). "DR-DOS/OpenDOS Enhancement Project" (http://www.drdosprojects.de/). Retrieved 2015-04-20.
61. US 5579517 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US5579517), Reynolds, Aaron R.; Dennis R. Adler & Ralph A. Lipe et al., "Common name space for long and short filenames", issued 1996
62. US 5745902 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US5745902), Miller, Thomas J. & Gary D. Kimura, "Method and system for accessing a file using file names having different file name formats", issued 1998
63. US 5758352 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US5758352), Reynolds, Aaron R.; Dennis R. Adler & Ralph A. Lipe et al., "Common name space for long and short filenames", issued 1998
64. US 6286013 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US6286013), Reynolds, Aaron R.; Dennis R. Adler & Ralph A. Lipe et al., "Method and system for providing a common name space for long and short file names in an operating system", issued 1996
65. Microsoft.com (http://www.microsoft.com/presspass/press/2003/dec03/12-03ExpandIPPR.mspx) Archived (https://web.archive.org/web/20090822044026/http://www.microsoft.com/presspass/press/2003/dec03/12-03ExpandIPPR.mspx) August 22, 2009, at the Wayback Machine
66. "FAT File System" (http://webarchive.loc.gov/all/20160921194326/http://www.microsoft.com/iplicensing/productDetail.aspx?product%20title=FAT%20File%20System). *Intellectual Property Licensing*. Microsoft. Archived from the original (http://www.microsoft.com/iplicensing/productDetail.aspx?product%20title=FAT%20File%20System) on 2016-09-21.
67. Garrett, Matthew (January 19, 2012). "EFI and Linux: the future is here, and it's awful" (https://www.youtube.com/watch?v=V2aq5M3Q76U). *linux.conf.au*. YouTube. Retrieved 2014-01-12.
68. US 5367671 (https://worldwide.espacenet.com/textdoc?DB=EPODOC&IDX=US5367671), Feigenbaum, Barry A. & Felix Miro, "System for accessing extended object attribute (EA) data through file name or EA handle linkages in path tables", issued 1994
69. Ravicher, Daniel B. (April 15, 2004). "PUBPAT's Request for Reexamination of Microsoft's FAT Patent" (http://www.pubpat.org/assets/files/MicrosoftFAT/Reynolds_517_Reexam_Request.pdf) (PDF). Public Patent Foundation. Retrieved 2014-01-12.
70. USPTO (September 30, 2004). "Patent Office's Office Action Rejecting Microsoft FAT Patent" (http://www.pubpat.org/assets/files/MicrosoftFAT/Reynolds_517_Rejected_040916.PDF) (PDF). Public Patent Foundation. Retrieved 2014-01-12.
71. Broache, Anne (January 10, 2006). "Microsoft's file system patent upheld" (http://www.cnet.com/news/microsofts-file-system-patent-upheld/). CNET News.
72. Paul, Ryan (February 25, 2009). "Microsoft suit over FAT patents could open OSS Pandora's Box" (https://arstechnica.com/microsoft/news/2009/02/microsoft-sues-tomtom-over-fat-patents-in-linux-based-device.ars). arstechnica.com. Retrieved 2009-02-28.
73. Fried, Ina (March 30, 2009). "Microsoft, TomTom settle patent dispute" (http://news.cnet.com/8301-13860_3-10206988-56.html). cnet.com. Retrieved 2009-08-22.
74. "Microsoft Motorola Patent Suit" (https://www.scribd.com/doc/38550703/Micrsoft-Motorola-Patent-Suit). October 1, 2010. Retrieved 2010-10-02.
75. Protalinski, Emil (October 1, 2010). "Microsoft sues Motorola, citing Android patent infringement" (https://arstechnica.com/microsoft/news/2010/10/microsoft-sues-motorola-citing-android-patent-infringement.ars). arstechnica.com. Retrieved 2010-10-02.
76. Paul, Ryan (July 2, 2009). "New Linux patch could circumvent Microsoft's FAT patents" (https://arstechnica.com/information-technology/2009/07/vfat-linux-patch-could-circumvent-microsofts-patent-claims/). ArsTechnica.com. Retrieved 2013-10-30.
77. Brown, Eric (July 2, 2009). "Can FAT patch avoid Microsoft lawsuits?" (https://web.archive.org/web/20130131034455/http://www.desktoplinux.com/news/NS4980952387.html). DesktopLinux.Com. Archived from the original (http://www.desktoplinux.com/news/NS4980952387.html?kc=rss) on January 31, 2013. Retrieved 2009-08-23.

78. Müller, Florian (December 5, 2013). "Federal Patent Court of Germany invalidates Microsoft FAT patent, appeals court may disagree" (http://www.fosspatents.com/2013/12/federal-patent-court-of-germany.html). FOSS Patents. Retrieved 2014-01-12.

External links

- *Description of the FAT32 File System* (http://support.microsoft.com/kb/154997/): Microsoft Knowledge Base Article 154997
- *MS-DOS: Directory and Subdirectory Limitations* (https://jeffpar.github.io/kbarchive/kb/039/Q39927/): Microsoft Knowledge Base Article 39927
- *Overview of FAT, HPFS, and NTFS File Systems* (http://support.microsoft.com/kb/100108/): Microsoft Knowledge Base Article 100108
- Microsoft Technet; *Volume and file size limits of FAT file systems* (https://web.archive.org/web/20060307082555/http://www.microsoft.com/technet/prodtechnol/winxppro/reskit/c13621675.mspx), copy made by Internet Archive Wayback Machine (https://archive.org/) of an article with summary of limits in FAT32 which is no longer available on Microsoft website.
- Chen, Raymond; *Microsoft TechNet: A Brief and Incomplete History of FAT32* (https://web.archive.org/web/20081118122857/http://www.microsoft.com/technet/technetmag/issues/2006/07/WindowsConfidential/)
- *Fdisk does not recognize full size of hard disks larger than 64 GB* (http://support.microsoft.com/kb/263044): Microsoft Knowledge Base Article 263044
- *Microsoft Windows XP: FAT32 File System* (https://web.archive.org/web/20050319235548/http://www.microsoft.com/resources/documentation/Windows/XP/all/reskit/en-us/prkc_fil_cycz.asp), copy made by Internet Archive Wayback Machine (https://archive.org/) of an article with summary of limits in FAT32 which is no longer available on Microsoft website.