Facultatea de Electronica, Telecomunicatii si Tehnologia Informatiei

Universitatea „Politehnica” , Bucuresti

Arhitectura sistemelui de fisiere FAT

Profesor Coordonator :

Conf.dr.ing. Stefan Stancescu

Student:

Patriche Nicolae - Cristian

Cuprins :

Introducere

1.Privire de ansamblu

File Allocation Table (FAT) este numele unei arhitecturi pentru fisiere de sistem și in momentul de fata o intreaga familie din industria de lucru a fișierelor folosesc aceasta arhitectura.

Sistemul de fișiere FAT este un sistem de fișiere simplu și robust. Oferă performanțe bune chiar și în implementările mai laborioase , dar nu poate oferi aceeași performanță, fiabilitate și scalabilitate ca unele sisteme moderne de fișiere. Totuși, este susținută din motive de compatibilitate de aproximativ toate sistemele de operare existente pentru calculatoare personale și astfel, este un format potrivit pentru schimbul de date între calculatoare și dispozitive de aproape orice tip și vârstă de la începutul anilor 1980 până în prezent.

Inițial conceput la sfarsitul anilor 1970 pentru a fi utilizate pe floppy disk-uri, a fost curând adaptat și folosit aproape universal timp de doua decenii pe hard disk-uri in timpul compatibilitatii erelor DOS si Windows 9x . Odată cu introducerea computerelor si a sistemelor de operare mai performante, precum și dezvoltarea de sisteme de fișiere mai complexe pentru acestea , arhitectura FAT nu mai reprezinta favoritul publicului pentru utilizarea pe hard disk-uri de catre majoritatea sistemelor moderne de operare .

Astăzi, arhitectura FAT este inca frecvent întâlnita pe floppy disk-uri, carduri de memorie solid-state, carduri de memorie flash și pe mai multe dispozitive portabile și integrate. Aceasta este, de asemenea, utilizată în etapa de pornire a calculatoarelor EFI (Extensible Firmware Interface)

Denumirea sistemului de fișiere provine din utilizarea proeminenta a unui tabel index alocat static la momentul de formatare. Tabelul conține intrări pentru fiecare cluster(unitate de alocare pe disk pentru fisiere si directoare), o zonă adiacenta de stocare pe disc. Fiecare intrare conține fie numărul urmatorului cluster sau un marcaj care indică sfârșitul unui fisier , spațiul nefolosit pe disc sau zone special rezervate ale discului. Directorul rădăcină al discului conține numărul primului cluster pentru fiecare fișier în acel director; sistemul de operare poate parcurge apoi tabelul FAT, căutând numărul cluster-ului pentru fiecare parte succesivă a fișierului de pe disc ca si cand avem un lanț de clustere până cand vom ajunge la sfarsitul fisierului.

In conditiile in care unitatile de disc au evoluat, numărul maxim de clustere a crescut în mod semnificativ și astfel numărul de biți folosiți pentru a identifica fiecare cluster a crescut. Versiunile succesive majore ale formatul FAT sunt denumite după numărul de biti ai elementelor tabelului:

12 (FAT12), 16 (FAT16) și 32 (FAT32). Fiecare din aceste variante este încă în uz. Standardul FAT a fost, de asemenea, extins si dezvoltat si pe alte ramuri , păstrând în general compatibilitate cu software-ul existent.

2.Utilizari

Sistemul de fișiere FAT oferă performanțe rezonabile, chiar și în foarte implementări mai complexe. Prin urmare, este adoptat pe scară largă și susținut de aproape toate sistemele de operare existente pentru computere personale, precum și pentru o multitudine de sisteme embedded. Acest lucru dovedeste utilitatea formatului pentru carduri de memorie solid-state și este convenabil sa fie folosit pentru a partaja date între sistemele de operare.

FAT este sistemul de fișiere implicit pentru unitatile media amovibile (cu excepția CD-urilor și a DVD-urilor) și ca atare sunt de obicei găsite pe floppy disk-uri, super-dischete, memorie și carduri de memorie flash sau unitati flash USB și totodata sunt susținute de majoritatea dispozitivelor portabile cum ar fi PDA-uri, camere digitale, camere video, playere media sau telefoane mobile. În timp ce FAT12 este omniprezent pe dischete, FAT16 și FAT32 sunt de obicei găsite in unitatile media mai mari.

Datorită utilizării pe scară largă a unitatilor media formatate cu ajutorul arhitecturii FAT , multe sisteme de operare oferă suport pentru FAT .De exemplu, Linux, FreeBSD, BeOS și JNode ofera suport încorporat pentru FAT, chiar daca susțin, de asemenea, sisteme de fișiere mai sofisticate, cum ar fi Ext4 sau Btrfs.

3.Evolutie tehnica

3.1 Original 8-bit FAT

Structura FAT (așa cum a fost numita inițial), s-a născut în una dintr-o serie de discuții între Marc McDonald și Bill Gates fiind proiectata și codificata de McDonald. Acesta a fost introdus cu elemente de tabel de 8-biți (și numere pentru clustere valabile de la 0x02 la 0xBF .Sistemul de fișiere FAT a fost, de asemenea, utilizat în sistemul de operare de la Microsoft - MDOS / MIDAS, proiectat de McDonald din 1979 pentru platformele 8080/Z80.

3.2 FAT12

Între lunile aprilie și august 1980, în timp ce împrumuta conceptul FAT pentru sistemul de operare QDOS 0.11 al lui 8086, Tim Paterson a extins elementele de tabel în 12 biți, reducand numărul FAT-urilor la doua si redefinind semantica unor valori rezervate ale clusterelor.

Initial conceput ca un sistem de fișiere pentru dischete, FAT12 folosea intrări pe 12-biți pentru adresele clusterelor în FAT, care nu numai că a limitat numarul clusterelor la 4078 (pentru grupurile de date de la 0x002 la 0xFEF) sau în unele cazuri chiar pana la 4084 (pentru grupurile de date de la 0x002 la 0xFF5) dar a făcut manipularea FAT dificila pentru registrele de 8 respectiv 16 biti ale unui PC; În timp ce MS-DOS și PC DOS suportau până la 4084 clustere pe unitatile compatibile FAT12, valoarea clusterului 0xFF0 este tratata ca marker suplimentar pentru sfarsitul lantului de clustere pe orice volum FAT12 .

Dimensiunea discului a fost salvata și calculata ca un număr de 16-biti sectoare, ceea ce a limitat dimensiunea de 32 MB pentru o dimensiune a sectorului logic de 512 octeți. FAT12 a fost folosit de mai mulți producători, cu diferite formate fizice, dar o dimensiune tipica pentru o dischetă la acel moment de timp a fost de 5,25 inch (130 mm) cu o capacitate de 160 KB, atât pentru zonele de sistem cat și pentru fișiere.

În timp ce 86-DOS suporta trei formaturi de disc (250.25 KB, 500.5 KB și 1232 KB cu descriptor media 0xff și 0xFE) pe 8-inch (200 mm) unitati floppy, IBM PC-DOS 1.0, lansat cu originalul IBM Personal Computer în 1981 , suporta doar un format cu 8 sectoare ,de o capacitate formatată de 160 KB (descriptor media 0xFE) pentru unitati floppy de 5.25 inch cu o singura fata și PC-DOS 1.1 a adăugat suport pentru formatul față-verso cu 320 KB (descriptor media 0xFF). PC-DOS 2.0 a introdus suport pentru formatele floppy cu 9 sectoare de 180 KB (descriptor media 0xFC) și 360 KB (descriptor media 0xFD).

PC-XT a fost primul calculator cu un hard disk de la IBM și PC-DOS 2.0 a susținut hard disk-ul cu FAT12 (descriptor media 0xF8). Ipoteza folosirii a 8 sectoare pe clusterele de pe hard disk a limitat practic, dimensiunea maximă a partiției la 16 MB pentru 512 sectoare octet si 4 KB clustere.

În domeniul computerelor, BIOS Paramater Block , de multe ori prescurtat BPB, este o structură de date în sectorul partitiei de bootare(Volume Boot Record – VBR) care descrie aspectul fizic al unui volum de stocare a datelor. Pe dispozitive partiționate, cum ar fi hard disk-urile, BPB descrie partiția unitatii, în timp ce, pe dispozitivele nepartiționate cum ar fi dischetele se descrie intregul mediu. Un BPB de bază poate apărea și fi folosit pe orice partiție, inclusiv pe dischetele unde prezența sa este adesea necesara. Fisierele de sistem care utilizează un BPB sunt FAT12 (cu excepția DOS 1.x), FAT16, FAT32, HPFS și NTFS.Bios Parameter Block (BPB) a fost introdus cu PC-DOS 2.0.

MS-DOS 3.0 a introdus suport pentru densitatea inalta de 1,2 MB pentru dischete de 5,25 inch (descriptor media 0xF9), care a avut în special 15 sectoare si prin urmare mai mult spatiu pentru FAT.

FAT12 rămâne în uz pe toate unitatile floppy disk , inclusiv 1,44 MB și mai târziu, 2.88 MB (descriptor media 0xf0).

Original FAT16

La data de 14 august 1984, IBM a lansat PC-ul AT, care venea cu un hard disk de 20 MB și PC-DOS 3.0. Microsoft a introdus în paralel MS-DOS 3.0. Adresele clusterelor au crescut la 16 biti, permitand până la 65,524 clustere pe volum și prin urmare, dimensiuni mult mai mari pentru fisierele de sistem, cel puțin în teorie. Cu toate acestea, numărul maxim posibil de sectoare și dimensiunea maxima de 32 MB nu s-a modificat. Prin urmare, deși adresele clusterelor au fost pe 16 biți, acest format nu a fost ceea ce azi este considerat a fi FAT16.

Odată cu implementarea inițială a FAT16 care nu asigura dimensiuni mai mari ale partitiei decat FAT12, beneficiul initial al lui FAT16 a fost ca permitea utilizarea de clustere mai mici, făcând utilizarea discului mai eficienta, în special pentru un număr mare de fișiere de doar cateva sute de bytes.

Hard disk-urile din MS-DOS 2.x mai mari de 15 MB vor fi incompatibile cu versiunile ulterioare de MS-DOS. Un hard disk de 20 MB formatat sub MS-DOS 3.0 nu a fost accesibil de către mai vechiul MS-DOS 2.0 deoarece MS-DOS 2.0 nu suporta versiunea 3.0 a FAT16. MS-DOS 3.0 putea accesa în continuare MS-DOS 2.0 cu partitii de 8 KB clustere sub 15 MB.

Sectorizarea logica a FAT-ului

Când hard disk-urile si-au marit considerabil capacitatea de stocare implementarea fisierele de system FAT12 si FAT16 in MS-DOS/PC DOS nu a oferit mijloacele necesare pentru a profita de spațiul de stocare suplimentar, cativa producători creand si dezvoltandu-si propriile variante FAT pentru a aborda problema în MS –DOS OEM.

Unii furnizori (AST și NEC) au suportat opt în loc de patru standard, intrările partițiilor primare în Master Boot Record (MBR), și-au adaptat MS-DOS pentru a folosi mai mult decat o singura partitie primara.

Așa-numitele sectoare logice au fost mai mari (până la 8192 bytes) decât dimensiunea sectorului fizic (de obicei 512 bytes), cum era de așteptat de către ROM-BIOS INT 13h sau de hardware-ul unității de disc. DOS-BIOS-ul sau sistemul BIOS-ul va combina apoi mai multe sectoare fizice în sectoare logice pentru ca sistemul de fisiere sa se foloseasca de ele. Aceste modificări au fost transparente pentru implementarea sistemului de fișiere în kernelul DOS din moment ce in nivelele de abstractizare a fisierelor de sistem volumele sunt văzute ca o serie liniară de sectoare logic adresabile, cunoscute sub numele de sectoare absolute (adresate de numărul lor de sector logic (LSN), începând cu LSN 0) independent de locația fizică a volumului .

Dezavantajul acestei abordări a fost existenta unui sector mai puțin eficient din punct de vedere al memoriei ,punandu-si serios amprenta ipentru structurile de date din DOS. Din moment ce versiunile mai vechi DOS nu au fost suficient de flexibile pentru a lucra cu aceste geometrii logice, producătorii de echipament original (OEM) au trebuit să introducă ID-uri partitie noi pentru variantele lor de FAT cu scopul de a le ascunde de probleme universal ale MS-DOS si PC DOS. ID-urile partitie cunoscute pentru sectoarele logice FAT includ: 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), 0x56 (AT & T MS-DOS 3.x), 0xE5 (Tandy MS-DOS), 0xF2 (Sperry IT MS-DOS 3.x, Unisys MS-DOS 3.3 - folosit de către Digital Research DOS plus 2. Versiuni OEM, cum ar fi Toshiba MS-DOS, Wyse MS-DOS 3.2 și 3.3 precum și Zenith MS-DOS sunt, de asemenea, cunoscute ca folosesc sectorizarea logica.

Final FAT16

În cele din urmă, în noiembrie 1987, Compaq MS-DOS 3.31 (o versiune OEM modificata de MS-DOS 3.3 lansata de Compaq cu mașinile lor) a introdus ceea ce astăzi este pur și simplu cunoscut sub numele de FAT16, cu extinderea sectorului de 16 biti la 32 de biți în BPB. Deși schimbările de pe disc au fost minore, întreagul disc DOS a trebuit să fie convertite pentru a utiliza sectoare pe 32 de biți, o sarcină complicată pentru faptul ca DOS-ul a fost scris intr-un limbaj de asamblare pe 16 biti. Rezultatul a fost inițial numit DOS 3.31 Large File System. Instrument DSKPROBE Microsoft se referă la tipul 0x06 ca BigFAT în timp ce unele versiuni mai vechi ale FDISK l-au descries ca BIGDOS. Tehnic vorbind , este cunoscut sub numele de FAT16B.

Din moment ce versiunile mai vechi de DOS nu a putut face față cu mai mult de 65,535 sectoare, a fost, de asemenea, necesar să se introducă un nou tip de partiție pentru acest format. În timp ce forma originală de FAT16 cu mai puțin de 65536 de sectoare a avut un tip de partiție 0x04, un tip 0x06 tip indică 65536 sau mai multe sectoare. Singura diferență între originalul FAT16 și formatul FAT16B este utilizarea de noi BPB cu intrari ale sectoarelor pe 32 de biți asa incat sistemele de operare mai noi sa poata face față, de asemenea, cu formatul original FAT16 fără modificări necesare. Cu toate acestea,in practica tipul partitiilor primare 0x01 și 0x04 nu trebuie sa se afle fizic în afara primilor 32 MB de pe disc, din cauza altor restricții in MS-DOS 2.x.

În 1988, îmbunătățirea FAT16B a devenit mult mai vizibila prin DR DOS 3.31, PC-DOS 4.0, OS / 2 1.1, și MS-DOS 4.0. Limita pe dimensiunea partiției a fost dictată de numărul de 8-biți sectoare per cluster, care a avut inițial un maxim de putere a doua valori de 64. Cu dimensiunea standard a sectorului de 512 bytes, aceasta oferă un maxim de 32 KB dimensiune pe cluster, fixare astfel limita "definitiva" pentru dimensiunea partiției FAT16 la 2 GB pentru sectoare de 512. Pe medii magneto-optice, care poate avea 1 sau 2 KB sectoare în loc de 0,5 KB, aceasta limita de dimensiune este proporțional mai mare.

Mult mai târziu, Windows NT a crescut dimensiunea maxima a clusterului la 64 KB, luând în considerare numărul de sectoare per cluster ca fiind nedefinit. Cu toate acestea, formatul rezultat nu a fost compatibil cu orice alta implementarea FAT din acel timp și a generat o fragmentare internă mai mare..

Înainte de 1995, versiunile de DOS accesau discul prin adresarea CHS. Când MS-DOS 7.0 / Windows 95 a introdus accesul LBA pe disk, partițiile au putut începe să fie localizate în afara primului cca. 8 GB de pe disc și, prin urmare, nu mai statea la mana traditionalei metode de acces prin CHS. Prin urmare, partițiile parțial sau integral situate dincolo de bariera CHS au trebuit să fie ascunse de sistemele de operare non-LBA prin utilizarea noilor tipuri de partitii 0x0E în tabela de partiții. Partițiile FAT16 care utilizează acest tip de partiție sunt numite FAT16X.

FAT32

Pentru a depăși limita de dimensiune de la FAT16, în timp ce în același timp permițându-I modului real DOS să se ocupe de format, Microsoft a proiectat o noua versiune a sistemului de fișiere, **FAT32**, care a suportat un număr mare de posibile clustere dar care ar putea reutiliza cea mai mare parte a codului existent, astfel încât memoria convențională existenta sa fie redusa cu mai puțin de 5 KB pentru DOS. Valorile cluster sunt reprezentate de numere pe 32 de biți, din care 28 de biti sunt folositi pentru pastrarea numărului de cluster. Sectorul de boot utilizează un câmp pe 32 de biți pentru numărul de sector, limitand mărimea volumului compatibil FAT32 la 2 TB pentru o dimensiune de sector de 512 octeti si 16 TB pentru o dimensiune de sector de 4096 octeti. FAT32 a fost introdusă cu MS-DOS 7.1 / Windows 95 OSR2 în 1996, cu toate că era nevoie de folosirea reformatarii și DriveSpace 3 (varianta care a venit cu Windows 95 OSR2 și Windows 98) nu l-au sprijinit. Windows 98 a introdus o utilitate pentru a converti hard disk-uri existente de la FAT16 la FAT32 fără pierderi de date. În linia de Windows NT, suportul nativ pentru FAT32 a ajuns în Windows 2000.

Dimensiunea maximă posibilă pentru un fișier de pe un volum FAT32 este de 4 GB minus 1 octet sau 4294967295 (232-1) octeti.

Sistemul deschis FAT + isi propune sa stocheze fișiere mai mari de 256 GB minus 1 octet sau 274877906943 (238-1) bytes pe volumele FAT32 ușor modificate dar se impune un risc deoarece implementarile FAT32 care nu au fost anuntate de aceasta extensie pot provoca stergeri de fisiere in cazul depasirii formatului normal FAT32. De asemenea, suportul pentru FAT32 + (și FAT16 +) este limitat la unele versiuni ale DR-DOS și nu este disponibil în sistemele de operare obișnuite de până acum.

Ca și cu sistemele de fișiere anterioare proiectarea sistemului de fișiere FAT32 nu a inclus sprijin direct pentru numele de fișiere lungi, dar volumele FAT32 pot pastra opțional nume de fișiere VFAT lungi exact în același mod in care numele lungi de fisiere VFAT au fost implementate opțional pentru FAT12 și FAT16.

Două tipuri de partiții au fost rezervate pentru partițiile FAT32, 0x0B și 0x0C. Ultimul tip este, de asemenea, numit FAT32X pentru a indica utilizarea LBA in locul CHS-ului.

Extensii

Atribute extinse

OS / 2 depinde foarte mult de atributele extinse (EA) și le stochează într-un fișier ascuns numit "EA $_{SP}$DATA. $_{SP}$ SF" în directorul rădăcină al volumelor FAT12 si FAT 16. Acest fișier este indexat de doi octeți rezervati anterior în intrarea directorului la adresa de offset 0x14. În formatul FAT32, acesti octeti pastreaza partea superioară de 16 biți ale numărului clusterului de pornire din fișier sau director, prin urmare, face imposibilă stocarea atributelor extinse OS/2 (EA) pe FAT32 folosind această metodă.

*Nota: Atributele extinse ale fisierelor sunt niste caracteristici ale sistemului de fișiere care permit utilizatorilor să asocieze fișiere de calculator cu metadate(date despre date) care nu sunt interpretate de sistemul de fișiere, în timp ce atributele normale au un scop strict definit de sistemul de fișiere (cum ar fi permisele sau înregistrările care vizeaza timpul de modificare sau creare).

Cu toate acestea, fisierele de sistem instalabile pe a 3-a partitie FAT32 (IFS) cu FAT32.IFS driver versiunea 0.70 si mai sus realizate de Henk Kelder & Netlabs pentru OS / 2 stocheaza atributele extinse in fișierele suplimentare cu nume fisierelor avand șirul "$_{SP}$EA.$_{SP}$SF" anexat la numele fișierului in fisierul de care aparțin.Driver-ul utilizeaza, de asemenea, octetul la offset 0x0C in intrarile directorului pentru a stoca un octet pentru marcaj care indică prezența atributelor extinse pentru a ajuta la accelerarea lucrurile. (Această extensie este critic incompatibilă cu metoda FAT32 + pentru stocarea fișierelor mai mari mult de 4 GB minus 1 pe volumele FAT32.

Fisierele cu nume lungi - LFN

Unul dintre obiectivele experienței de utilizare pentru designeri de Windows 95 a fost capacitatea de a folosi nume de fișiere lungi (LFNs-până la 255 UCS-2 unități de cod lung), în plus față de clasicul 8.3 filenames cunoscut sub numele de fisiere cu nume scurt – short filename (SFN). Pentru compatibilitatea forward and backward LFN-urile au fost implementate ca o extensie opțională pe partea de sus a structurilor existente ale sistemului de fișiere FAT.

Această metodă transparentă de stocare a numelor de fișiere lungi în sisteme de fișiere FAT existente, fără a modifica structurile lor de date este de obicei cunoscut ca **VFAT** (pentru "FAT Virtual").

Sistemele NON –VFAT pot accesa în continuare SFN-urile din cadrul lor fără restricții;cu toate acestea, fisierele cu nume lung pentru VFAT sunt invizibile pentru OS/2 si fisierele cu nume lung pentru EA-uri sunt invizibile pentru Windows .

OS / 2 a adăugat suport FAT pentru nume lungi de fisiere folosind atributele extinse (EA) înainte de introducerea VFAT; asadar, numele lungi de fisiere atribuite VFAT sunt invizibile pentru OS / 2 și numele lungi de fișiere atribuite EA sunt invizibile la Windows.Prin urmare ,utilizatori icu experiență in ambele sisteme de operare sunt nevoiti sa redenumeasca manual fisierele.

Derivate ale sistemului de fisiere FAT

FATX

FATX este o familie de sisteme de fișiere concepute pentru hard disk-urile consolei de jocuri Xbox produs Microsoft introdus în anul 2001.

În timp ce regasim aceleași idei de baza ca si la designul FAT16 și FAT32, structurile **FATX16 și FATX32** sunt simplificate dar fundamental incompatibile cu sistemele de fișiere FAT32 si FAT16 normale ceea ce face imposibil pentru driverele fiserelor de system FAT normale montarea unor volume compatibile FATX16 si FATX32.

Sectorul superblock non-bootabil are o dimensiune de 4 KB și are o structura BPB(Bios Parameter Block) de 18 octeti complet diferita de structurile normale ale BPB.. Clusterele sunt de obicei de 16 KB în dimensiune și exista doar o copie FAT pe Xbox. Intrărilor directoarelor sunt de 64 bytes in dimensiune în loc de cele normale care au 32 bytes. Fișierele pot avea lungimea numelor de până la 42 de caractere folosind setul de caractere OEM și să fie de până la 4 GB minus 1 byte în dimensiune.

exFAT

exFAT este un sistem de fișiere incompatibil, care a fost introdus cu Windows Embedded CE 6.0 în noiembrie 2006. Se bazează vag pe arhitectura FAT dar apartine si este protejat de patente.

Tipul partiției MBR este 0x07 (la fel ca cele utilizate pentru IFS, HPFS, NTFS, etc). Informația logica localizata în VBR este stocată într-un format care nu se regaseste la BPB. exFAT este destinat utilizării pe memoriile flash (cum ar fi SDXC și Memory Stick XC), unde FAT32 este deasemenea utilizat.

Acesta oferă mai multe avantaje in comparatie cu FAT32, inclusiv depasirea celor 4 GB limita (numai când nu luam in considerare extensia FAT + care permite fișiere mai mari de 4 GB)fiind mult mai eficienta din punct de vedere al spatiului pentru fișiere mai mici de 64 KB existente pe volume mari .

Dispozitivele de stocare formatate ca exFAT nu pot face schimb de date cu un dispozitiv care nu suportă formatul. Multe echipamente nu suporta exFAT care necesită achiziționarea unei licențe comerciale de la Microsoft.

Chiar dacă sistemul de operare Windows 7 are suport FAT32, abilitatea de a folosi asta ca de formatare a hard disk-ului a fost limitata în mod artificial în GUI de Microsoft, care le permite sa aleagă între exFAT și NTFS. Utilizatorii care încearcă folosirea formatului FAT32 trebuie să execute comanda FORMAT intr-un cmd cu autoritate de administrator.

Proiectare si design tehnic

1. Aspect

Privire de ansamblu asupra ordinii structurilor existente intr-un disc sau partitie FAT

Continut	Sector de boot	Sector informatii FS(doar FAT32)	Sectoare rezervate (optional)	File Allocation Table #1	File Allocation Table #2 (conditional)	Root Directory (doar FAT12/FAT16)	Regiunea datelor(pentru fisiere si directoare)
Dimensiunea sectoarelor	(numarul sectoarelor rezervate)			(number FAT-urilor) * (sectoare per FAT)		(numarul intrarilor root*32) / (bytes per sector)	(numarul clusterelor) * (sectoare per cluster)

Un sistem de fișiere FAT este compus din patru sectiuni diferite:

- Sectoarele rezervate, localizate la inceput

Primul sector rezervat (sector logic 0) este sectorul de boot (aka Volum Boot Record (VBR)). Acesta include o zona numita BIOS parameter Block (cu unele informații de bază le sistemului de fișiere, în special tipul lui și indicii de localizare pentru alte sectiuni) și de obicei conține codul de bootare al sistemului de operare.

Informațiile importante din sectorul de boot sunt accesibile printr-o structură a sistemului de operare numita Drive Parameter Block (DPB) în DOS și OS / 2.

Numărul total de sectoare rezervate este indicat printr-un camp in sectorul de boot și este de obicei 32 pentru sistemele de fișiere FAT32.

Pentru sisteme de fișiere FAT32, sectoarele rezervate includ un Sector de informatii pentru sistemul de fisiere(FS Information Sector) la sector 1 logic și un sector de boot pentru back-up la sector logic 6.

În timp ce mulți alți furnizori au continuat să utilizeze o instalare cu un singur sector (sectorul logic 0) pentru bootstrap loader, codul sectorului de boot de la Microsoft a crescut să ruleze pe sectoare logice 0 și 2 de la introducerea FAT32, cu sector logic 0 în funcție de subrutine din sectorul logic 2. Zona Sectorului de Boot pentur Back-up consta in 3 sectoare logice 6, 7 si 8. În unele cazuri, Microsoft utilizează, de asemenea, sectorul 12 al zonei sectoare rezervate pentru extinderea bootarii.

- Regiunea FAT

Aceasta conține de obicei doua copii (pot varia) ale File Allocation Table pentru verificare dar este rar folosit, chiar si de utilitarele pentru reparatii disc.

De obicei copiile suplimentare sunt păstrate în sincronizare strânsă pe modul SCRIERE și la CITIRE ele sunt folosite doar atunci când apar erori în primul FAT. În FAT32, este posibila comutarea de la comportamentul implicit și se poate selecta un singur FAT din cei disponibili pentru a fi utilizate in diagnosticare.

Primele două clustere (0 și 1) de pe hartă conțin valori speciale.

- Regiunea Root Directory

Este vorba despre un tabel director care stochează informații despre fișierele și directoarele aflate în directorul rădăcină. Este folosit doar cu FAT12 și FAT16 și impune pe directorul rădăcină o dimensiune maximă fixă care este prealocata la crearea acestui volum. FAT32 stochează directorul rădăcină din Regiunea de date, împreună cu fișiere și alte directoare, permițându-i să crească fără o astfel de constrângere. Astfel, pentru FAT32, Regiunea de date incepe aici.

- Regiunea datelor

  Aici est locul unde fisierele sau directoarele propriu-zise sunt stocate și ocupă cea mai mare parte a partiție. În mod tradițional, părțile neutilizate ale regiunii de date sunt inițializate cu o valoare de umplere de 0xF6 ca pe INT 1Eh Disk Paramater Table (DPT). Dimensiunea fișierelor și subdirectoarele poate fi crescută arbitrar (atâta timp cât există clustere libere), pur și simplu prin adăugarea de mai multe link-uri in lantul de fisiere al FAT. Rețineți că fișierele sunt alocate în unități de clustere, astfel încât în cazul în care 1 fisier de 1 KB isi ocupa pozitia intr-un cluster de 32 KB, vor rezulta 31 de KB pierduti.

FAT utilizează formatul **little-endian (micul indian)** pentru toate intrările din antet (cu excepție pentru unele intrări de pe sectoarele de boot Atari ST). Este posibil să se aloce mai multe sectoare FAT decât este necesar pentru numărul de clustere. Sfârșitul ultimul sector FAT poate fi neutilizat în cazul în care nu există clustere corespunzătoare. Numărul total de sectoare (așa cum s-a menționat în boot record) poate fi mai mare decât numărul de sectoare folosite pentru date(clustere × sectoare pe cluster), FAT-uri (număr de FAT-uri × sectoare per FAT) si sectoare ascunse inclusiv sectorul de boot : aceasta ar duce la sectoare neutilizate la sfârșitul volumului. Dacă o partiție conține mai multe sectoare decât numărul total de sectoare ocupate de catre fisierele de sistem, va rezulta tot sectoare nefolosite la sfarsitul unui volum.

Sectorul de BOOT

Pe dispozitive non-partiționate cum ar fi dischetele(floppy disks), sectorul de boot (VBR) este primul sector (sectorul logic 0 cu adresa fizica CHs 0/0/1 sau adresa LBA 0). Pentru dispozitive partiționate cum sunt hard disk-urile, primul sector este Master Boot Record în timp ce primul sector de partiții formatate cu un sistem de fișiere FAT este din nou sectorul de boot.

Structura comună a primilor 11 bytes folositi de cele mai multe versiuni de FAT pentru masinile IBM compatibile X86 este prezentata in tabelul 1.

Byte Offset	Lungime (bytes)	Descriere
0x000	3	Instructiune de JUMP. În cazul în care sectorul de boot are o semnătură valabila care gaseste în ultimeii doi octeți ai sectorului de boot (testat de cele mai multe încărcătoare de boot care se gasesc în sistemul BIOS sau MBR) și acest volum este bootat, prioritatea loader-ului de bootare va trece la executarea acestui punct de intrare cu valori concrete ale registrelor iar instrucțiunea de salt va sari peste restul de antet (non-executabil). Începând cu DOS 2.0, discurile valabile pentru bootarea x-86 trebuie să înceapă fie cu un salt scurt urmat de un NOP.Pentru compatibilitate inversă MS-DOS, PC-DOS și DR-DOS se acceptă de asemenea un salt (0x69) pe discuri amovibile. Pe hard disk-uri DR DOS acceptă in plus secvența JMPS începând cu un NOP (0x90 0xEB 0x?? in timp ce DOS MS-DOS/PC nu.) Prezența pentru unul dintre aceste modele (în combinație cu un test pentru o valoare valida a descriptorului la un offset de 0x015) servește ca indicator pentru DOS 3.3).
0x003	8	Denumire OEM . Deși oficial documentate ca fiind destinate pentru folosirea OEM, MS-DOS/PC DOS (din 3.1), Windows 95/98/SE/ME și OS/2 verifica acest domeniu pentru a determina care alte părți ale boot record pot fi invocate și cum să le interpreteze. Prin urmare, stabilirea etichetei OEM pentru valorile arbitrare sau false poate duce la incapacitatea MS-DOS, PC-DOS și OS / 2 să recunoască volumul în mod corespunzător și produce coruperea datelor la scriere.
0x00B	varies	BIOS Parameter Block (octetii 13, 19, 21 sau 25), Extended BIOS Parameter Block (32 sau 51 bytes) or FAT32 Extended BIOS Parameter Block (79 bytes); dimensiunea si continutul variaza intre sistemele de operare si versiuni.
varies	varies	Sistemul de fișiere și codul de bootare specific sistemului de operare; de multe ori începe imediat în spatele [E] BPB, dar uneori datele aditionale pentru loder-ulde bootare sunt stocate între sfârșitul [E] BPB și începutul codul de boot.
0x1FD	1	Numărul unității fizice (numai de la DOS 3.2 la sectoarele de boot din 3.31). Cu OS/2 1.0 si DOS 4.0, această intrare s-a mutat în sectorul offset 0x024 (la offset 0x19 în EBPB). Cele mai multe sectoare de boot IBM si Microsoft mențin valorile pentru 0x00 la offset 0x1FC și 0x1FD , deși nu fac parte din semnatura la 0x1FE. Dacă acest lucru aparține unui volum de boot, DR-DOS 7.07 poate fi configurat (vezi offsetul NEWLDR 0x014) pentru a actualiza dinamic această intrare la valoarea DL furnizata la timpul bootarii sau valoarea stocată în tabelul de partiții. Acest lucru permite bootarea sistemului de pe drive-uri alternative, chiar și atunci când codul VBR ignoră valoarea DL.
0x1FE	2	Semnătură sectorul de boot (0x55 0xAA). Această semnătură indică un cod de boot PC compatibil IBM și este testat de cele mai multe loadere de boot care se gasesc în sistemul BIOS sau MBR înainte de a trece la execuția codului de boot din sectorul de boot. Această semnătură nu indică un anumit sistem de fișiere sau sistem de operare. Deoarece acesta semnătură nu este prezenta pe toate discurile cu format FAT (de exemplu, nu este pe DOS sau pe volumele FAT non-x86-boot), sistemele de operare nu trebuie să se bazeze pe această semnătură pentru a fi prezente atunci când vă conectați la volume (probleme vechi de MS-DOS/PC DOS cu precadere la versiunea 3.3).Instrumentele de formatare nu trebuie să scrie această semnătură în cazul în care sectorul de boot scris nu conține cel puțin un model de boot loader compatibil x86 ; acesta trebuie să oprească CPU într-o buclă fără sfârșit (0xF4 0xEB 0xFD) sau să emită un 19h INT și RETF (0xCD 0x19 0xCB). Această semnătură trebuie să fie situata la offset-ul 0x1FE pentru dimensiuni de sector de 512 sau mai mari. Dacă dimensiunea sectorului fizic este mai mare,aceasta poate fi repetat la sfârșitul sectorului fizic. STs Atari va asuma un disc in a fi comptabil de bootare pentru microprocesorul Motorola 68 K în cazul în care suma de control peste 256 de cuvinte big-endian(marele indian) din sectorul de boot este egal cu 0x1234. În cazul în care codul loaderului de boot este compatibil IBM, este important să se asigure că suma de control asupra sectorului de boot nu se potrivește accidental cu aceasta suma de control. Dacă acest lucru s-ar întâmpla se va schimba un bit

		nefolosit (de exemplu, înainte sau după zona codului de boot) pentru a asigura ca această condiție nu este îndeplinită. În cazuri rare, o semnătură inversata 0xAA 0x55 a fost observat pe imaginile discului. Acest lucru poate fi rezultatul unei implementari defectuoasă în instrumentul de formatare bazat pe o documentație defecta dar poate indica, de asemenea, un ordin de octet schimbat a imaginii discului care ar fi avut loc in transferul între platforme utilizând un algoritm de ordonare diferit al octetilor. Valorile BPB și cele FAT12, FAT16, FAT32 au menirea de a utiliza numai reprezentarea little-endian și nu există implementari cunoscute care sa utilizeze big-endian.

Tabel 1

Deasemenea pentru o mai buna comparare voi analiza in tabele 2 si 3 formatul sectorului de boot pentru volumele MSX – DOS si Atari ST compatibile FAT12 care au un aspect foarte asemanator cu cele studiate mai sus.

Byte Offset	Length (bytes)	Description
0x000	2	Instructiune de salt. Sectoarele de boot originale Atari ST incep cu o instructiune 68000 BRA.S (0x60 0x??).Pentru compatibilitatea cu sisteme de operare, discurile formatate Atari ST odata cu aparitia sistemului de operare TOS 1.4 de la Atari au inceput cu OxE9 ox??.
0x002	6	Denumirea OEM .
0x008	3	Numarul disk-ului (implicit: 0x00 0x00 0x00), utilizat de către Atari ST pentru a detecta o schimbare a disc. (Windows 9x Volumul Tracker va stoca întotdeauna "IHC" aici pe floppy disk-uri non-protejate la scriere). Această valoare trebuie să fie schimbata în cazul în care conținutul de pe disc este schimbat cu exteriorul, altfel e posibil ca Atari să nu recunoască schimbarea la reinsertie.
0x00B	19	DOS 3.0 BIOS Parameter Block (format little-endian – micul indian)
0x01E	variaza	Datele din sectorul privat de boot (format mixt pentru big-endian si little-endian)

variaza	variaza	Sistemul de fișiere și sistemul de operare specific codului de bootare ATARI ST. Nicio ipoteza nu trebuie să se facă în ceea ce privește poziția de încărcare a codului, care trebuie să fie relocatabil. Dacă încărcarea unui sistem de operare nu reușește, codul poate reveni la Atari ST BIOS cu 68000 RTS (cod operational 0x4E75 cu 0x4E 0x75 [nb 8]) instruire și toate registrele nemodificate.
0x1FE	2	Suma de control. Suma de control de format 16-bit peste 256 de cuvinte big-endian al sectorului de boot de 512 octeti trebuie să se potrivească cu valoarea magica 0x1234, în scopul de a indica un cod executabil pentru sectorul de boot valabil Atari 68000 .Această suma de control poate fi folosita pentru a alinia suma de control corespunzător. Dacă dimensiunea sectorului logic este mai mare de 512 octeti, restul nu sunt incluse în suma de control și este de obicei zero .Deoarece unele sisteme de operare pentru PC în mod eronat nu acceptă dischete formatate FAT dacă semnătură 0x55 0xAA nu este prezenta aici, este recomandabil sa se plaseze 0x55 0xAA în acest loc (și sa se adauge un boot loader compatibil IBM) apoi sa se foloseasca un cuvânt neutilizat în sectorul de date sau in zona codului de bootare sau numărul de serie, în scopul de a se asigura că suma de control 0x1234 nu se potriveste (cu excepția cazului de suprapunere a codului FAT de la IBM PC si Atari ST executabile in acelasi timp).

Tabel 2

Byte Offset	Length (bytes)	Description
0x000	3	Instructiune de salt la o adresa fictiva (e.g., `0xEB 0xFE 0x90`).
0x003	8	Denumire OEM
0x00B	19	*DOS 3.0 BPB*
0x01E	variaza (2)	Punct de intrare in codul MSX –DOS 1 pentru procesoare Z80 în codul de bootare MSX. Acest lucru este în cazul în care masinile MSX-DOS 1 sar de la trecerea de control pentru sectorul de boot. Această locație se suprapune cu formate BPB din DOS 3.2 sau codul pentru sectorul de boot compatibil x86 sau pentru sectoarele de boot compatibile IBM PC și va duce la o eroare pe mașina MSX dacă nu au fost luate măsuri de precauție speciale, cum ar fi „prinderea CPU” intr-o bucla stransa.
0x020	6	Semnatura volumului MSX – DOS 1.
0x026	1	MSX-DOS 2 steag care nu permite stergerea (implicit:. 0x00 cazul în care semnătură "VOL_ID" este prezentă în sectorul offset 0x020, acest indicator indică, în cazul în care volumul deține fișierele șterse care nu pot fi sterse.
0x027	4	Numarul discului MSX – DOS 2(default: `0x00000000`).
0x02B	5	rezervat
0x030	variaza (2)	Punct de intrare in codul MSX –DOS 2 pentru procesoare Z80 în codul de bootare MSX Acest lucru este în cazul în care masinile MSX-DOS 2 sar de la trecerea de control pentru sectorul de boot. Această locație se suprapune cu formate EBPB din DOS 4.0/OS/2 1.2 sau codul pentru sectorul de boot compatibil x86 sau pentru sectoarele de boot compatibile IBM PC și va duce la o eroare pe mașina MSX dacă nu au fost luate măsuri de precauție speciale, cum ar fi „prinderea CPU” intr-o bucla stransa.
0x1FE	2	Semnatura

Tabel 3

Sector Offset	BPB Offset	Length (bytes)	Description
0x00B	0x00	2	Pe sectorul logic în avem dimensiuni Bytes in puterile lui doi, iar valoarea cea mai comună este 512. Unele sisteme de operare nu acceptă alte dimensiuni sectoriale. Pentru simplitate și performanță maximă, dimensiunea sectorului logic este adesea identică cu dimensiunea fizica a sectorului unui disc, dar pot fi mai mari sau mai mici, în unele cazuri. Valoarea minimă permisă pentru volume FAT12/FAT16 non-bootabile cu până la 65,535 sectoare logice este de 32 de bytes, sau 64 de bytes, pentru mai mult de 65,535 de sectoare logice. Valoarea minimă practic este 128. Unele versiuni pre-DOS 3.31 ale DOS foloseau dimensiuni de sector logic de pana la 8192 bytes pentru sectoare logice FAT. Atari ST GEMDOS supporta dimensiuni de sector logic între 512 și 4096. [DR-DOS suportă bootare de pe volumele FAT12/FAT16 cu dimensiuni de sector logic de până la 32 KB și implementări INT 13h care suporta sectoare fizice de până la 1024 bytes / sector. Dimensiunea minimă pentru un sector logic FAT32 standard este de 512 bytes dar pot fi redusa pana la 128 bytes fără sprijinul Sectorului de Informatii FS.. Disk-urile Floppy si controlerele folosesc dimensiuni fizice ale sectorului de 128, 256, 512 și 1024 bytes (de exemplu, PC / AX).Portofoliul Atari suporta o dimensiune de sector de 512 pentru volume mai mari de 64 KB, 256 bytes pentru volume mai mari 32 KB si 128 Bytes pentru volume mai mici. Unitățile magneto-optice utilizeaza dimensiuni de sector de 512, 1024 și 2048 bytes..
0x00D	0x02	1	Sectoare logice pentru fiecare cluster. Valorile permise sunt puteri de doi de la 1-128. Unele versiuni de MS-DOS 3.x au suportat o dimensiune maximă a clusterului de doar 4 KB în timp ce sistemele moderne MS-DOS/PC DOS și Windows 95 suporta o dimensiune maximă a clusterului de 32 KB. Windows-urile 98/SE/ME suporta parțial o dimensiune a clusterului de 64 de KB dar unele servicii FCB nu sunt disponibile pe astfel de discuri și diverse aplicații nu funcționeza.Familia Windows NT și unele versiuni alternative DOS, cum ar fi PTS-DOS suporta pe deplin clustere de 64 KB . Pentru sistemele de operare DOS, dimensiunea maximă per cluster rămâne la 32 KB sau 64 KB chiar si in cazul dimensiunilor mai mari de 512 octeți. MS-DOS/PC DOS va închide la pornire în cazul în care această valoare este eronat specificata ca 0.
0x00E	0x03	2	Numarul de sectoare logic rezervate. Numărul de sectoare logice înainte de primul FAT în imaginea sistemului de fișiere. Cel puțin 1 pentru acest sector, de obicei 32 pt FAT32 (pentru tinerea sectorului de boot extins, sectorul de informatii FS și sectoarele de boot pentru backup). Din momentul in care volumele DR-DOS 7.0 x formatate FAT32 utilizează un sector de boot cu un singur sector, sectorul de informatii FS si sectorul de back-up, unele volume formatate sub DR-DOS folosesc o valoare de 4 aici.
0x010	0x05	1	Numărul de File Allocation Tables(FAT). Aproape întotdeauna 2, discurile RAM ar putea folosi 1. Cele mai multe versiuni ale MS-DOS/PC DOS nu acceptă mai mult de 2 FAT-uri. Unele sisteme de operare DOS sprijina doar două FAT-uri în driverul lor incorporat in disc, dar suporta alte calcule pentru blocarea rularii driverlor in dispozitive.
0x011	0x06	2	Numărul maxim de FAT12 sau FAT16 intrări in directorul rădăcină. 0 pentru FAT32, unde directorul rădăcină este stocat în grupuri de date obișnuite; a se vedea offset-ul 0x02C în EBPB-urile FAT32. Această valoare trebuie să fie reglata astfel încât intrările din agendă sa consume întotdeauna sectoarele logice ,

			prin care fiecare intrare director ocupă 32 de octeți. MS-DOS/PC DOS are nevoie ca această valoare să fie un multiplu de 16. Valoarea maxima suportata pe o discheta este 240, valoarea maximă acceptată de MS-DOS/PC DOS pe hard disk-uri este de 512..
0x013	0x08	2	Totalitatea sectoarelor logice (daca sunt zero, se va folosi o valoare de 4 octeti la offset-ul `0x020`)
0x015	0x0A	1	0xE5 8-inch (200 mm) Single sided, 77 tracks per side, 26 sectors per track, 128 bytes per sector (243 KB) (DR-DOS only) 0xED 5.25-inch (130 mm) Double sided, 80 tracks per side, 9 sector, 720 KB 0xF0 3.5-inch (90 mm) Double Sided, 80 tracks per side, 18 or 36 sectors per track (1.44 MB or 2.88 MB). Destinat pentru utilizare cu formate floppy personalizat și superfloppy în care geometria este definita în BPB. Este folosit, de asemenea, pentru alte tipuri de media, cum ar fi benzi . 0xF8 Disc fix (de exemplu, de obicei, o partiție de pe un hard disk). (incepand cu DOS 2.0) Desemnat pentru a fi folosite pentru orice media fixa sau detașabile , în cazul în care geometria este definită în BPB. 3.5-inch Single sided, 80 tracks per side, 9 sectors per track (360 KB) (MSX-DOS only) 5.25-inch Double sided, 80 tracks per side, 9 sectors per track (720 KB) (Sanyo 55x DS-DOS 2.11 only) 0xF9 3.5-inch Double sided, 80 tracks per side, 9 sectors per track (720 KB) (since DOS 3.2)[8] 3.5-inch Double sided, 80 tracks per side, 18 sectors per track (1440 KB) (DOS 3.2 only)[8] 5.25-inch Double sided, 80 tracks per side, 15 sectors per track (1.2 MB) (since DOS 3.0)[8] 0xFA 3.5-inch and 5.25-inch Single sided, 80 tracks per side, 8 sectors per track (320 KB) Folsite deasemenea pentru discurile RAM si ROM (HP 200LX) Hard disk (Tandy MS-DOS only) 0xFB 3.5-inch and 5.25-inch Double sided, 80 tracks per side, 8 sectors per track (640 KB) 0xFC 5.25-inch Single sided, 40 tracks per side, 9 sectors per track (180 KB) (since DOS 2.0) 0xFD 5.25-inch Double sided, 40 tracks per side, 9 sectors per track (360 KB) (since DOS 2.0) 8-inch Double sided, 77 tracks per side, 26 sectors per track, 128 bytes per sector (500.5 KB) (8-inch Double sided, (single and) double density (DOS 1) 0xFE

			5.25-inch Single sided, 40 tracks per side, 8 sectors per track (160 KB) (since DOS 1.0) 8-inch Single sided, 77 tracks per side, 26 sectors per track, 128 bytes per sector (250.25 KB) 8-inch Double sided, 77 tracks per side, 8 sectors per track, 1024 bytes per sector (1232 KB) (8-inch Single sided, (single and) double density (DOS 1) 0xFF 5.25-inch Double sided, 40 tracks per side, 8 sectors per track (320 KB) (since DOS 1.1) Hard disk (Sanyo 55x DS-DOS 2.11 only) Această valoare trebuie să reflecte descriptorul media stocat (în intrarea pentru cluster 0) în primul octet din fiecare copie a FAT. Anumite sisteme de operare înainte de DOS 3.2 (86-DOS, MS-DOS/PC DOS 1.x și MSX-DOS versiunea 1.0) ignorau parametrii sectorul de boot și foloseau valoarea descriptorului media de la primul octet din FAT pentru a alege dintres sabloanele parametrilor predefiniti intern. Trebuie să fie mai mare sau egal cu 0xf0 din DOS 4.0. Pe unitățile detașabile, DR-DOS va presupune prezența unui BPB dacă această valoare este mai mare sau egală cu 0xf0, întrucât pentru discurile fixe, acesta trebuie să fie 0xF8 ca sa se presupună prezența unui BPB. Inițial, aceste valori au fost menite să fie utilizate ca flag-uri pentru orice disc media amovibil fără un format BPB recunoscut și un descriptor media pentru 0xF8 sau 0xFA la 0xFF .MS-DOS/PC DOS tratează bit 1 ca un steag pentru a alege un format cu 9-sectoare per track, mai degrabă decât un format de 8 sectoare și bitul 0 ca un steag care indica un mediu față-verso. Valorile de la 0x00 la 0xEF și de la 0xF1 la 0xF7 sunt rezervate și nu trebuie să fie utilizate.
0x016	0x0B	2	Sectoarele logice per File Allocation Table pentru FAT12/FAT16, 0 pentru FAT32

Tabel 4

BIOS Parameter Block

Structura comuna pentru primii 25 octeti ai BPB(Bios Parameter Block) – **prezentata pe larg in tabelul 4** sunt folositi de versiunile FAT incepand cu DOS 2.0 (octetii din offsetul sectorului 0x00B pana la 0x017 sunt stocati incepand cu DOS 2.0 dar nu intotdeauna folositi inainte de DOS 3.2) precum si alte structuri alaturi de caracteristici , vor fi prezentate in tabelele de mai jos.

DOS 3.0 BPB:

Următoarele extensii au fost documentate incepand cu DOS 3.0,insa cu toate acestea, au fost deja susținute de unele probleme ale DOS 2.13. MS-DOS 3.10 suporta în continuare formatul DOS 2.0, dar ar putea sa foloseasca deasemenea formatul DOS 3.0.

Sector Offset	BPB Offset	Length (bytes)	Description

0x00B	0x00	13	*DOS 2.0 BPB*
0x018	0x0D	2	Sectoare fizice cu INT 13h geometrie CHS, de exemplu, 15 pentru un floppy de 1.20 MB O intrare de zero indică faptul că această intrare este rezervata, dar nu este utilizat.
0x01A	0x0F	2	Numărul de capete pentru discuri cu INT 13h geometrie CHS, de exemplu, 2 pentru un floppy cu două fețe. Un bug în toate versiunile de MS-DOS/PC DOS până la inclusiv 7.10 cauzeaza nefunctionarea acestor sisteme de operare pentru geometrii CHS cu 256 de capete, prin urmare, aproape toate BIOS-urile aleg un maxim de doar 255 de capete. O intrare de zero indică faptul că această intrare este rezervata, dar nu este utilizata.
0x01C	0x11	2	Numarul sectoarelor anterioare ascunse de partiția care conține acest volum FAT. Acest câmp trebuie să fie întotdeauna zero, pe unitatile media nepartitionate. Acesta intrare a DOS 3.0 este incompatibilă cu o intrare similară la offset-ul 0x01C in BPB-uri pana la DOS 3.31. Ea nu trebuie să fie utilizata daca intrarile sectoarelor logice la offset de 0x013 este zero.

DOS 3.2 BPB:

Oficial, MS-DOS 3.20 este folosit încă in formatul DOS 3.0, dar SYS și FORMAT au fost adaptate pentru a suporta un format de 6 bytes (din care nu toate intrările au fost).

Sector Offset	BPB Offset	Length (bytes)	Description
0x00B	0x00	19	*DOS 3.0 BPB*
0x01E	0x13	2	Totalul sectoarelor logice, inclusiv sectoarele ascunse. Intrarea DOS 3.2 este incompatibilă cu o intrare similară la offset 0x020 în BPB-uri pana la DOS 3.31. Ea nu trebuie să fie utilizata daca intrarile sectoarelor logice la un offset de 0x013 este zero.

DOS 3.31 BPB:

Oficial introdus cu DOS 3.31 și nefiind utilizat de către DOS 3.2, unele utilitati DOS 3.2 au fost concepute pentru a ti cont de acest nou format. Documentație oficială recomandă să se aibă încredere în aceste valori numai în cazul în care intrarile sectoarelor logice la un offset de 0x013 este zero.

Sector Offset	BPB Offset	Length (bytes)	Description
0x00B	0x00	13	*DOS 2.0 BPB*
0x018	0x0D	2	Sectoare fizice pe pistă pentru discuri cu INT 13h cu geometrie CHS, de exemplu, 18 pentru un floppy de 1.44 MB.Sunt neutilizate de unitățile care nu suporta accesul CHS. Identic cu o intrare disponibila incepand cu DOS 3.0. O intrare zero indică faptul că acesta intrare este rezervata, dar nu este utilizata. O valoare de 0 poate indica doar acces LBA dar poate provoca o excepție a divizarii cu zero în unele boot loadere care pot fi evitate prin stocarea unei valoari neutre aici, daca nicio geometrie CHS nu poate fi simulata.
0x01A	0x0F	2	Numărul de capete pentru discuri cu INT 13h geometrie CHS, de exemplu, 2 pentru un floppy cu două fețe. Neutilizate de unități, care nu suporta accesul CHS. Identic cu o intrare disponibila din DOS 3.0. Un bug în toate versiunile de MS-DOS/PC DOS până la inclusiv 7.10 cauzeaza prabusirea acestor sisteme de operare pentru geometrii CHS cu 256 de capete, prin urmare, aproape toate BIOS-urile aleg un maxim de doar 255 de capete. O intrare de zero indică faptul că această intrare este rezervata, dar nu este utilizata. O valoare de 0 poate indica doar acces LBA, dar poate provoca o exceptie a divizarii cu zero în unele boot loadere, care pot fi evitate prin stocarea unei valoari neutre , daca nicio geometrie CHS nu poate fi simulata rezonabil.
0x01C	0x11	4	Numarul de sectoare ascunse anterioare partiției care conține acest volum FAT. Acest câmp trebuie să fie întotdeauna zero, pe unitatile media care nu sunt partiționate. Această intrare DOS 3.31 este incompatibilă cu o intrare similară la offset 0x01C în BPB-urile din DOS 3.0-3.3. Cel puțin, putem avea incredere doar dacă retine valoarea zero sau în cazul în care intrarile sectoarele logice la offset-0x013 este zero. Dacă aceasta apartine unui AAP(Advanced Active Partition) selectat la timpul bootarii , intrarea BPB va fi actualizată dinamic de MBR pentru a reflecta valorile "sectoarele relative" în tabelul de partiții, depozitate la offset 0x1B6 în AAP sau NEWLDR MBR , astfel încât acesta devine posibila să booteze sistemul de operare de la EBR-uri(extended boot record)..
0x020	0x15	4	Totalul sectoarelor logice (în cazul în care este mai mare decat 65535; în caz contrar, avem offset 0x013). Această intrare DOS 3.31 este incompatibilă cu o intrare similară la offset 0x01E în BPB-urile din DOS 3.2-3.3. Oficial, acesta trebuie să fie utilizat numai în cazul în care intrarile sectoarelor logice la un offset de 0x013 este zero, dar unele sisteme de operare (unele versiuni mai vechi ale DR DOS), utilizeaza această intrare pentru discuri mai mici.

O formulă simplă transforma numărul CN al cluster-ului unui volum intr-un numar LSN al unui sector logic.

1.Determinați (o dată) SSA = RSC + FN × SF + ceil ((32 × RDE) / SS), în cazul în care numărul RSC al sectorului rezervat este stocat la 0x00E, numărul FAT FN la offset de 0x010 ,sectoarele pe FAT SF la offset 0x016 (FAT12/FAT16) sau 0x024 (FAT32), intrarile RDE ale directorului rădăcină la offset 0x011, dimensiunea sectorului SS la offset 0x00B si ceil (x) .

2.Determina LSN = SSA + (CN-2) × SC, unde sectoare per cluster SC sunt depozitate la offset 0x00D.

Pe unitatile nepartitionate numărul de sectoare ascunse este zero și prin urmare adresele LSN si LBA au devenit aceleași atâta timp cât dimensiunea unui sector logic este identică cu dimensiunea sectorului fizic mediu subiacent lui. În aceste condiții, apar relatia de mai jos :

LSN = SPT x (HN + (NOS x TN)) + SN-1, în cazul în care sectoarele pe pista SPT sunt depozitate la offset 0x018 și numărul de fete NOS la offset 0x01A. Numarul pistei TN, numărul de cap HN și numărul sectorului SN corespund **Cylinder –head-sector (CHS)**: formula ne ajuta la translatia CHS in LBA.

Extended BIOS Parameter Block

Structura folosita de către FAT12 și FAT16 cu OS/2 1.0 si DOS 4.0, de asemenea cunoscuta ca Extended BIOS Parameter Block (EBPB):

Sector Offset	EBPB Offset	Length (bytes)	Description
0x00B	0x00	25	*DOS 3.31 BPB*
0x024	0x19	1	Numărul unitatii fizice (0x00 pentru (prima) unitate media amovibila, 0x80 pentru (primul) disc fix ca pe INT 13h). Valorile permise pentru posibilele unități fizice în funcție de BIOS sunt : 0x00-0x7E și 0x80-0xFE. Valorile 0x7F și 0xFF sunt rezervate pentru scopuri interne, cum ar fi bootarea ROM-ului sau control și nu ar trebui să apară pe disc. Unele boot loadere cum ar fi MS-DOS/PC DOS utilizeaza această valoare atunci când booteaza sistemul de operare, altele se ignora cu totul sau folosesc numărul furnizat in registrul DL de catre loader-ul de bootare subiacent (de exemplu, cu multe BIOS-uri si MBR- uri). Intrarea este uneori schimbata de instrumentele SYS sau poate fi fixata dinamic de către loader-ul bootstrap cu scopul de a forța codul sectorului de boot pentru a boota sistemul de operare de pe discuri fizice alternative decât cel implicit. O intrare similară a existat (numai) în DOS 3.2-3.31 sectorul de boot de la sectorul de offset 0x1FD. Dacă acest lucru aparține unui volum bootabil, DR-DOS 7.07 poate fi configurată pentru a actualiza dinamic intrarea EBPB la valoarea DL furnizata la un anumit timp de boot sau valoarea stocată în tabelul de partiții. Acest lucru permite pornirea sistemului de pe drive-uri alternative, chiar și atunci când codul VBR ignoră valoarea DL.
0x025	0x1A	1	Rezervat; În unele MS-DOS/PC DOS codul de boot utilizat ca un blocnotes pentru octetul superior al INT 13h in cazul cuvântului de 16 biți asumat la un offset de 0x024. Unele sectoare de boot DR-DOS FAT12/FAT16 folosesc intrarea tot ca un blocnotes dar pentru scopuri diferite. VGACOPY stochează un CRC asupra sistemului de ROM-BIOS în această locație. Unii manageri de boot folosesc această intrare pentru a comunica litera de unitate dorită pe care volumul ar trebui să apară la sisteme de operare cum ar fi OS / 2, prin stabilirea bitului 7 și specificarea numărului unitatii în biții 6-0 (C: = valoarea 0, D: = valoarea 1, ...). Deoarece aceasta afectează în mod normal imaginea din memorie a sectorului de boot, acest lucru nu pune probleme de compatibilitate cu alte utilizări; În Windows NT folosit pentru flag-uri CHKDSK (biții 7-2 mereu fara valoare, bit 1: erori de disc I / O cu care se confruntă, este posibil sectoare rele, rularea scanarii de suprafață la următorul boot, bit 0: volumul este "murdar" și nu a fost corect demontat înainte de închidere, se ruleaza CHKDSK la pornirea următoare).Ar trebui să fie setat la 0.

0x026	0x1B	1	Semnatura de bootare extinsa. (Ar trebui să fie 0x29 pentru a indica faptul că există o EBPB cu următoarele 3 intrări (de la OS / 2 1.2 și DOS 4.0). Poate fi 0x28 pe un OS / 2 1.0-1.1 și discurile cu PC DOS 3.4 indică o formă anterioară a formatului EBPB . MS-DOS/PC DOS 4.0 și cele superioare, OS / 2 1.2 și cele superioare precum și familia Windows NT recunosc ambele semnături.)
0x027	0x1C	4	ID-ul volumului (număr de serie) De obicei, numărul de serie "xxxx-xxxx" este creat de un plus de 16-biti a ambelor valori DX returnate de INT 21h/AH = 2Ah (obținere date de sistem) și INT 21h/AH = 2Ch (obținere timp de sistem) pentru cuvântul superior și un alt plus de 16-biti a ambelor valori CX pentru cuvântul inferior din numărul de serie. Alternativ, unele utilități DR-DOS oferă o / opțiune # pentru a genera o ștampilă temporala lizibila "mmdd-hhmm" construita cu ajutorul valorilor BCD-codate pe 8-biți pentru luna, ziua, ora și minutul în loc de un număr de serie.
0x02B	0x20	11	Partiție Etichetă volum, de exemplu, "$NO_{SP}NAME_{SP\ SP\ SP\ SP}$ " Software care schimba eticheta volumului director în sistemul de fișiere ar trebui să actualizeze această intrare, dar nu toate software-urile o fac. Eticheta partiție volumului este de obicei afișata cu instrumentele de partitionare deoarece este accesibila fără a monta volumul. A fost implementata incepand cu OS / 2 1.2 si MS-DOS 4.0 dar si versiuni mai noi. Această zonă a fost folosita de către sectoarele de boot ale DOS 3.2 - 3.3 pentru a stoca o copie privată a Tabelului Parametrilor de pe Disk (DPT) în loc sa utilizeze indicatorul INT 1EH pentru a recupera tabelul ROM-ului ca și în problemele ulterioare ale sectorului de boot.Reutilizarea aceastei locații pentru cele mai multe probleme ale etichetei volumului partitionat în cazul în care unele utilitare de sistem mai vechi ar încerca în continuare să „ patch-uiasca” fostul DPT.
0x036	0x2B	8	Tip sistem de fișiere, căptușite cu spații (0x20), de exemplu, "$FAT12_{SP\ SP\ SP}$ ", "$FAT16_{SP\ SP\ SP}$ ", "$FAT_{SP\ SP\ SP\ SP\ SP}$ ".Această intrare este destinata pentru scopurile de afișare și nu trebuie să fie utilizate de către sistemul de operare pentru a identifica tipul sistemului de fișiere. Cu toate acestea, acesta este folosit uneori în scopuri de identificare de catre a 3-a partitie a software-ului și prin urmare, valorile nu trebuie să difere de cele utilizate în mod oficial. A fost implementat incepand cu OS / 2 1.2 si MS-DOS 4.0.

FAT32 Extended BIOS Parameter Block

În esență FAT32 introduce 28 bytes în EBPB, urmate de cei 26 de bytes EBPB rămasi după cum se arată mai sus pentru FAT12 și FAT16. Sistemele de operare Microsoft și IBM au determina tipul de sistemelor de fișiere FAT utilizate pe un volum independent de numărul de clustere si nu de utilizarea formatului BPB folosit sau tipul sistemului de fisiere indicat care este tehnic posibil sa utilizeze un "FAT32 EBPB"atat pentru volumele FAT12 si FAT16 cat și pentru un DOS 4.0 EBPB in cazul volumelor mici FAT32. Deoarece s-a constatat crearea de astfel de volume de catre sistemele de operare Windows , în anumite condiții ciudate sistemele de operare ar trebui să fie pregătite pentru a lucra cu aceste forme hibride.

Sector Offset	FAT32 EBPB Offset	Length (bytes)	Description
0x00B	0x00	25	*DOS 3.31 BPB*
0x024	0x19	4	Sectoare logice per File Allocation Table Octetul de la offset-ul 0x026 în această intrare nu ar trebui să devină 0x28 sau 0x29 în scopul de a evita orice interpretare greșită cu formatul EBPB sub atentia sistemelor de operare necompatibilie FAT32.
0x028	0x1D	2	Descrierea unitatii / mirroring flags (biții 3-0: zero, nr de baza pentru FAT activ, dacă bitul 7 este setat. Dacă bitul 7 este neinitializat, toate FAT-urile sunt reflectate ca de obicei. Sectoarele de boot DR-DOS 7.07 FAT32 cu suport dual LBA și CHS utilizeaza biți 15-8 pentru a stoca un steag de acces și o parte a unui mesaj. Acesti biți conțin fie un sablon 0110: 1111b (litere mici "o", bitul 13 setat pentru acces CHS) sau 0100:1111 b (majuscule litera "O", bitul 13 neinitializat pentru accesul LBA).Octetul este, de asemenea, utilizat pentru al doilea caracter într-un potențiala eroare "No_{SP} IBMBIO$_{SP\ SP}$ COM" (a se vedea offset 0x034), afișate fie în alternanta litere mari-mici fie doar lite mari indicând astfel ce tip de acces a eșuat). Instrumente de formatare sau instrumentele non-DR SYS pot șterge acesti biți dar și alte instrumente de disc ar trebui să lase biți 15-8 neschimbati.
0x02A	0x1F	2	Versiunea (definita ca 0.0).Octetul superior al numărului versiunii este stocat la offset 0x02B și octetul inferior la un offset de 0x02A. Implementările FAT32 ar trebui să refuze sa monteze volume cu numere de versiuni necunoscute de ei. Caracteristicile FAT + propune sa se schimbe numărul de versiune pentru volumele FAT32 + cu depășirea limitei de dimensiune standard de fișiere FAT32 de 4 GB - 1 la un număr de versiune 0.1 a păstra implementări neconștiente de această extensie pentru montarea volumului. Implementarile cunoscute ar trebui să monteze și sa așteapte intrări ale fisierelor mari FAT + cu un număr de versiune 0.0 pentru compatibilitate maximă .
0x02C	0x21	4	Numărul clusterului de start pentru directorul rădăcină, de obicei 2 (primul cluster), în cazul în care acesta nu conține niciun sector eronat. Implementarea FAT32 Microsoft impune o limită artificială de 65535 intrări pe directoare, în timp ce mai multe implementari ale partitiilor terț nu. O valoare cluster de 0 nu este permisă în mod oficial și nu poate indica un cluster de pornire valabil pentru directorul rădăcină. Unele implementări FAT32 non-standard pot trata acest numar ca un indicator pentru a căuta un director rădăcină cu dimensiune fixa intalnit pe volumele FAT16.
0x030	0x25	2	Numărul sectorului logic al Sectorului de informatii FS, de obicei 1, de exemplu, al doilea din cele trei sectoare de boot FAT32. Unele implementari FAT32 susțin o ușoară variație a caracteristicilor Microsoft în a face Sectorul de informatii FS opțional, specificând o valoare de 0xFFFF (sau 0x0000) în această intrare. De cand sectorul logic 0 nu poate fi un Sector de informatii valid, dar Sectoare de informatii FS folosesc aceeași semnătură ca și cele găsite pe mai multe sectoare de boot, implementarile sistemelor de fișiere nu ar trebui să încerce să utilizeze sectorul logic 0 ca Sector de informatii FS și in schimb sa se stabileasca neacceptarea caracteristicii pe care anumit volum. Fără un Sector de Informatii FS, dimensiunea minima a sectorului logic pe volumele FAT32 poate fi redusa pana la 128 biti pentru scopuri speciale.
0x032	0x27	2	Primul număr al sectorului logic al unei copii a celor trei sectoare de boot FAT32 este de obicei 6.

			Deoarece volumele DR-DOS 7.0 x formatate FAT32 utilizează un sector de boot cu un singur sector, unele volume formatate sub DR-DOS folosesc o valoare de 2 aici. Valorile 0x0000 (și / sau 0xFFFF) sunt rezervate și indică faptul că nici un sector de rezervă nu este disponibil.
0x034	0x29	12	Rezervat (poate fi schimbat la formatul byte-ului de umplere 0xF6 ca un artefact de MS-DOS FDISK, trebuie să fie inițializat la 0 de instrumente de formatare, dar nu trebuie să fie schimbat de implementarile sistemelor de fișiere sau instrumente de disc.) Sectoarele de boot FAT32 DR-DOS 7.07 folosesc acesti 12 bytes pentru a stoca fisierul "IBMBIO$_{SP\ SP}$ COM" care urmeaza sa fie încărcat (până la primii 29,696 bytes sau dimensiunea reala a fișierului , tot ceea ce este mai mic) apoi executate de sectorul de boot, urmată de un caracter terminal NUL (0x00). Aceasta este, de asemenea, parte dintr-un mesaj de eroare, indicând numele real al fișierului de bootare și metoda de acces (a se vedea offset 0x028).
0x040	0x35	1	Cf. `0x024` for FAT12/FAT16 (Numarul unitatii fizice)
0x041	0x36	1	Cf. `0x025` for FAT12/FAT16 (Utilizat pentru diverse scopuri)
0x042	0x37	1	CF. 0x026 pentru FAT12/FAT16 Multe implementari FAT32 nu suporta o semnătură alternativă de 0x28 pentru a indica o formă prescurtată a EBPB FAT32 doar pe următorul număr de serie (și nicio intrare pentru eticheta de volum și sistemle de fisiere) dar din moment ce acesti cei mai nefolositi 19 bytes ar putea servi pentru diferite scopuri în unele scenarii, implementarea ar trebui să accepte 0x28 ca o semnătură alternativă și apoi sa se intoarca sa utilizeze eticheta volumului directorului în sistemul de fișiere în loc de utilizarea in EBPB pentru compatibilitate cu extensiile posibile.
0x043	0x38	4	Cf. `0x027` for FAT12/FAT16 (ID-ul volumului)
0x047	0x3C	11	Cf. `0x02B` for FAT12/FAT16 (Eticheta volumului)
0x052	0x47	8	Cf. `0x036` for FAT12/FAT16 (tipul sistemului de fisiere, e.g. "`FAT32 SP SP SP`")

Exceptii

Versiunile de DOS înainte de 3.2 bazate integral sau parțial pe octetul descriptorului media în octetu BPB sau octetul ID-ului FAT în clusterul 0 al primului FAT cu scopul de a determina formatul dischetelor FAT12, chiar dacă BPB-ul este prezent. In functie de octetul descriptorului media găsit și tipul de unitate detectat în care implicit utilizeaza una dintre următoarele prototipuri BPB în loc sa foloseasca valorile de fapt stocate în BPB.

Descriptorul media (BPB offset 0x0A)	0xFF		0xFE			0xFD			0xFC	0xFB	0xFA	0xF9			0xF8		0xF0		0xED	0xE5
Dimensiune	8"	5.25"	8"	8"	5.25"	8"	8"	5.25"	5.25"	5.25" / 3.5"	5.25" / 3.5"	5.25"	3.5"	3.5"	5.25"	3.5"	3.5"	3.5"	5.25"	8"
Densitate	?	DD	SD	DD	DD	?	SD	DD	DD	?	?	HD 96tpi	DD	HD	QD	DD	HD 135tpi	ED	QD 96tpi	SD

		48tpi			48tpi			48tpi	48tpi				135tpi	135tpi	96tpi					
Modulatie	?	MFM	?	?	MFM	?	?	MFM	MFM	MFM	MFM	MFM	MFM	MFM	MFM	MFM	MFM	MFM	MFM	?
Capacitate formatata (KB)	?	320	250	1200	160	?	500	360	180	640	320	1200	720	1440	720	360	1440	2880	720	243
CHS	77	40	77	77	40	77	77	40	40	80	80	80	80	80	80	80	80	80	80	77
Sectoare fizice / pista (BPB offset 0x0D	?	8	26	8	8	?	26	9	9	8	8	15	9	18	9 (8)	9	18	36	9 (8)	26
Numarul de capete (BPB offset 0x0F)	?	2	1	2 (1)	1	?	2	2	1	2	1	2	2	2	2	1	2	2	2	1
Octeti / sectoare logice (BPB offset 0x00)	?	512	128	1024	512	?	128	512	512	512	512	512	512	512	512	512	512	512	512	128
Sectoare logice / cluster (BPB offset 0x02)	?	2	4	1	1	?	4	2	1	2	1	1	2	1	?	?	1	2	?	4
Sectoare logice rezervate (BPB offset 0x03)	?	1	1	1	1	?	4	1	1	1	1	1	1 (2)	1	1	1	1	1	?	1
Numarul de FAT-uri (BPB offset0x05)	?	2	2	2	2	?	2	2	2	2	2	2	2	2	2	2	2	2	2	2
Intrarile directorului radacina (BPB offset 0x06)	?	112	68	192	64	?	68	112	64	112	112	224	112	224	?	?	224	240	?	64
Totalitatea sectoarelor logice (BPB offset 0x08)	?	640	2002	1232 (616)	320	?	4004	720	360	1280	640	2400	1440	2880	?	?	2880	5760	?	2002

Sectoare lofice / FAT (BPB offset 0x0B)	?	1	6	2	1	?	6	2	2	2	2	7	3	9 (7)	?	?	9	9	?	1
Sectoare ascunse (BPB offset0x11)	?	0	3 (0)	0	0	?	0	0	0	0	0	0	0	0	0	0	0	0	?	0
DRIVER.SYS /F:n	?	0	3	4	0	?	3	0	0	?	?	1	2	7	?	?	7	9	?	3
Prezenta BPB	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?	?
Suport	?	DOS 1.1	DOS 1.0?	DOS 2.0?	DOS 1.0	?	DOS 2.0?	DOS 2.0	DOS 2.0	?	?	DOS 3.0	DOS 3.2	DOS 3.2 doar; (DR-DOS)	Sanyo 55x DS-DOS 2.11 doar	MSX-DOS doar	DOS 3.3	DOS 5.0	Tandy 2000 doar	DR-DOS doar

Sectorul de informatii FS

"Sectorul Informații FS" a fost introdus în FAT32 pentru accelerarea timpilor de acces la anumite operațiuni (în special, obtinerea de spațiu liber). Acesta este situat la un număr al sectorului logic specificat în valoarea de boot FAT32 EBPB la poziția 0x030 (de obicei sectorul 1 logic imediat după valoarea de boot).

Byte Offset	Length (bytes)	Description
0x000	4	Semnătura Sectorul de informatii FS (0x52 0x52 0x61 0x41 = "RRaA")
0x004	480	Rezervat (valorile byte-ului ar trebui să fie setate la 0x00 timpul formatarii, fara sa fie schimbat mai târziu)
0x1E4	4	Semnătura Sectorul de informatii FS (`0x72 0x72 0x41 0x61` = "`rrAa`")
0x1E8	4	Ultimul număr cunoscut de clustere de date de pe volum, sau 0xFFFFFFFF dacă este necunoscut. Ar trebui să fie setat la 0xFFFFFFFF în timpul formatarii și actualizat de către sistemul de operare mai târziu. Nu trebuie să fie absolut invocat pentru a fi corect în toate scenariile. Înainte de a utiliza această valoare, sistemul de operare ar trebui să verifice această valoare să fie de cel puțin mai mica sau egala cu numărul clusterelor din volum.
0x1EC	4	Numărul cel mai utilizat pentru a fi alocate clusterele. Ar trebui să fie setat la 0xFFFFFFFF pe durata formatarii și actualizate de către sistemul de operare mai târziu. Cu 0xFFFFFFFF sistemul ar trebui să înceapă de la clusterul 0x00000002. Nu trebuie să fie absolut invocat pentru a fi corect în toate scenariile. Înainte de a utiliza această valoare, sistemul de operare ar trebui să verifice această valoare să fie un număr de cluster valabil pentru volum.

0x1F0	12	Rezervat (valorile byte-ului ar trebui să fie setat la 0x00 in timpul formatarii, dar sa nu fie schimbat mai târziu)
0x1FC	4	Semnatura Sectorului de informatii FS (`0x00 0x00 0x55 0xAA`)(Toate cei patru octeți trebuie să se potrivească înainte ca si anume conținutul acestui sector sa fie în format valid.)

Datele din acest sector pot fi depășite și nu reflectă conținutul curent al media deoarece nu toate sistemele de operare actualizeaza sau folosesc acest sector, și chiar dacă o fac, conținutul nu este valabil în cazul în care mediul a fost scos fara demontare corespunzătoare volumul sau după o cadere de tensiune. Prin urmare, sisteme de operare ar trebui să verifice în primul rând bitflag-urile optionale de la oprire.

În cazul în care acest sector este prezent pe un volum FAT32, dimensiune minim permisa pentru un sector logic este de 512 bytes, în timp ce altfel ar fi 128 octeti. Unele implementari FAT32 susțin o ușoară variație a caracteristicilor Microsoft de a face Sectorul de Informatii FS opțional, specificând o valoare de 0xFFFF (sau 0x0000) la intrare pentru un offset de 0x030.

File Allocation Table

Zona de date

Un volum este împărțit în clustere de dimensiuni identice , blocuri mici învecinate. Dimensiunile clusterelor variază în funcție de tipul de sistem de fișiere FAT utilizat si deasemenea de dimensiunea partitiei; de obicei dimensiunile de cluster se află undeva între 2 KB si 32 KB.Fiecare fișier poate ocupa unul sau mai multe dintre aceste clustere, în funcție de dimensiunea sa, astfel ca un fișier este reprezentat printr-un lanț de clustere.Totuși aceste clustere nu sunt neapărat stocate adiacente unul de altul pe suprafața discului, dar sunt adesea fragmentate în regiunea de date.

File Allocation Table (FAT) este numărul de sectoare aflate imediat după zona de sectoare rezervate. Aceasta reprezintă o listă de intrări care mapează fiecare cluster pe volum. Fiecare intrare înregistrează una dintre cele patru lucruri:

1.numărul clusterului al clusterului urmator (avem o inlantuire de clustere)

2.un capat special pentru lantul de clustere (EOC) – indica sfarsitul lantului

3.o intrare specială pentru a marca un cluster eronat

4.un zero care indica ca grupul este nefolosit

Fiecare versiune a sistemului de fișiere FAT utilizează un format diferit de intrări FAT. Numerele mai mici dau FAT-ri mai mici, dar se iroseste spațiul în partiții mari in cazul in care au nevoie sa aloce clustere mari. Sistemul de fișiere FAT12 utilizează 12 biți pe intrarea FAT, astfel două intrări cuprind 3 bytes.Este în mod constant little-endian: în cazul în care acesti trei octeti sunt considerati ca fiind un număr compatibil little-indian pe 24 biti, cu 12 biți mai puțin semnificativi reprezintand prima intrare (cluster 0) și ceilalti 12 biți cei mai semnificativ reprezentand a doua (cluster 1).

Primele două intrări într-un FAT au valori speciale:

Prima intrare (cluster 0 în FAT) deține descriptorul media (valori permise 0xf0-0xFF cu 0xF1-0xF7 rezervate) in biti 7-0 care sunt de asemenea copiate în BPB-ul sectorului de boot. Cei 4 biți rămasi (dacă e cazul FAT12), 8 biți (dacă e cazul FAT16) sau 20 biți (dacă e cazul FAT32) vor fi in aceasta intrare întotdeauna 1. Pentru descriptori media in afara de 0xFF și 0x00, este posibil să se determine ordinea corecta pentru nibble si byte pentru a fi utilizati de către driver-ul sistemului de fisiere si cu toate acestea, sistemul de fișiere FAT utilizează în mod oficial o reprezentare little-endia și nu există implementari cunoscute de variante care sa utilizeze valori big-endian.

Cea de a doua intrare (cluster 1 în FAT) stochează nominal printr-un marker sfârșitul lantului de clustere cum este utilizat de către unitatea formatoare, dar are întotdeauna 0xFFF / 0xFFFF / 0x0FFFFFFF setati. Unele sisteme de operare Microsoft, cu toate acestea, seteaza acesti biți în cazul în care volumul nu este cel pe care este pastrat sistemul de operare care rulează (fiind folosita utilizarea 0xFFFFFFFF în loc de 0x0FFFFFFF).

Tabela directoare

Un tabel director este un tip special de fișier care reprezintă un director .Fiecare fișier sau director stocat în acesta este reprezentat de o intrare tip 32-byte în tabel. Fiecare intrare înregistrează numele, extinderea, atributele (arhiva, director, ascuns doar în citire, sistem și volumul), data și ora ultimei modificări, adresa primului cluster date de directorul de fișiere și în cele din urmă de mărimea fișierului / directorului. În afară de directorul rădăcină în FAT12 și sisteme de fișiere FAT16, toate tabelele directoare sunt stocate în regiunea de date. Numărul real de intrări într-un director stocat în regiunea de date poate crește prin adăugarea unui alt cluster al lanțului în FAT.
Sistemul de fișiere FAT în sine nu impune limite cu privire la adâncimea unui arbore subdirector pentru atâta timp cât există grupuri libere disponibile pentru a aloca subdirectoarele.
Rețineți că înainte de fiecare intrare nu poate fi "intrări false" pentru a sprijini un nume de fișier lung (LFN), a se vedea mai jos.

Caracterele oficiale folosite numele de fisiere scurte din DOS includ următoarele:
Majuscule A-Z
Numere 0-9
Spațiu(desi aici apar anumite exceptii si interpretari particulare pentru diferite sisteme de operare).
! # $% & '() - @ ^ _ `{} ~
Valorile 128-255
Următoarele caractere ASCII sunt valabile **doar** in cazul numelor lungi de fisiere (LFN):
"* /: <> \ |
+,. ; = []

Fragmentarea

Sistemul de fișiere FAT nu conține mecanisme built-in care împiedică fișierele nou scrise de la a deveni împrăștiate pe partiție. Pe volumele în care fișierele sunt create și șterse frecvent sau lungimea lor de multe ori este schimbata, mediul va deveni tot mai fragmentat de- a lungul timpului.

În timp ce proiectarea sistemului de fișiere FAT nu produce nici o ingreuna organizațională în structurile de disc sau sa reduca cantitatea de spațiu de stocare libera cu cantitati crescute de fragmentare, așa cum se întâmplă cu fragmentarea externă, timpul necesar pentru a citi și scrie fișiere fragmentate va crește iar sistemul de operare va trebui să urmeze lanțurile de clustere în FAT (cu parti ce trebuie să fie încărcate în memorie în primul rând, în special pe volume mari) și apoi sa citeasca datele corespunzătoare împrăștiate fizic pe tot mediul reducând șansele pentru drive-ul dispozitivului sa efectueze operatii I/O de tip multi-sector sau sa inițieze transferuri DMA mai mari sporind astfel în mod eficient orice suprasolicitare a protocolului. De asemenea, operațiunile cu fisiere vor deveni mai lente cu cat fragmentarea este tot mai mare asa ca este nevoie din ce în ce mai mult ca sistemul de operare sa isi găsească fișiere sau clustere libere.

Alte sisteme de fișiere, de exemplu, HPFS sau exFAT utilizați bitmap-urile libere care indică clusterele libere si utilizate, care ar putea fi apoi repede aranjat in ordine cu scopul de a găsi zonele învecinate libere. O altă soluție este ca legătura dintre toate clusterele libere sa fie într-una sau mai multe liste (așa cum se face în sistemele de fișiere Unix). În schimb, FAT trebuie să fie scanat ca o matrice pentru a găsi clustere libere, ceea ce poate duce la sancțiuni de performanta pentru discuri de mari dimensiuni.

De fapt, căutarea de fișiere în directoarele mari sau calcularea spațiului liber pe volume FAT este unul dintre cele mai mari operații consumatoare de resurse, deoarece necesită citirea tabelelelor directoare sau chiar întreagul FAT in mod liniar.

Odată cu introducerea de FAT32, cautarile lungi si numarul scanarilor au devenit mai evidente, mai ales pe volume foarte mari. O posibilă justificare sugerata de Raymond Chen de la Microsoft pentru a limita dimensiunea maximă a partițiilor FAT32 create pe Windows a fost implementarea unui timp necesar pentru a efectua o operație "DIR", care afișează întotdeauna spațiul liber pe disc ca ultima linie. Afisarea aceastei linii a durat mai mult cu cat numărul de grupuri a crescut. FAT32 a introdus, prin urmare, un sector de informatii special pentru a observa si cazul în care suma calculată anterior pentru determinarea spațiului liber se păstrează pe cicluri de putere, astfel încât contorul de spațiu liber trebuie să fie recalculat doar atunci când un mediu FAT amovibil se scoate fără sa fie demontat mai intai sau în cazul în care sistemul este oprit fără a fi inchis corect.

Diverse optimizări și trucuri pentru punerea în aplicare a driverelor pentru sistemele de fisiere FAT, drivere de dispozitiv bloc și instrumentele de disc au fost concepute pentru a depăși cele mai multe blocaje de performanță din proiectarea sistemului de fișiere fără a fi nevoie să se schimbe aspectul structurilor de pe disc. Ele pot fi împărțite în metode de lucru on-line și off-line prin încercarea de a evita fragmentarea în sistemul de fișiere, în primul rând, implementarea metodelor pentru a face față mai bine cu fragmentarea existentă , reordonarea și optimizarea structurilor de pe disc. Cu optimizări, performanta in volumele FAT se poate castiga cel mai bine odata cu complexitatea sistemelor de fisiere dar în același timp se păstreaza avantajul ca fragmentarile sa fie disponibile chiar si pe sisteme foarte mici sau mai vechi.

DOS 3.0 nu va reutiliza imediat spatiul de pe disc rezultat din cazul fisierelor sterse pentru noi alocări, ci cauta spațiul nefolosit anterior . Acest lucru nu numai că ajută la menținerea integrității fișierelor șterse pentru cât mai mult posibil, dar de asemenea, accelerează alocările de fișiere și evită fragmentarea, deoarece spatiul pe disc alocat este întotdeauna nefragmentat. DOS realizează acest lucru prin păstrarea unui pointer la ultimul cluster alocat pe fiecare volum montat în memorie și începe să caute spațiu liber din această locație pornind in sus, așa cum a fost încă realizat in DOS 2.x. În cazul în care se ajunge la sfârșitul FAT, se reiau căutarile de la începutul FAT până fie spațiu liber a fost găsit sau poziția inițială a fost atinsa din nou, fără a fi găsit spațiu liber. Aceste indicii sunt inițializate la punctul de la începutul FAT dupa bootare. În plus, intrările

director de fișierele șterse vor fi marcate 0xE5 din DOS 3.0. Deoarece DOS 3.3 oferă mijloacele de a îmbunătăți performanța operațiunilor de fișiere cu FASTOPEN pentru urmărirea poziției de fișiere deschise recent sau directoare în diferite forme de liste (MS-DOS/PC DOS) sau tabele (DR-DOS). Înainte de DOS 5.0 s-a avut o grijă deosebită atunci când se utilizează astfel de mecanisme, în combinație cu software de defragmentare pentru a nu aparea erori de sintaxa sau de compatibilitate.S-a insistat cel mai mult pe partea de compatibilitate si comunicare eficienta intre partea software in cazul defragmentarii si mecanismele interne din cadrul sistemelor de fisiere.

Windows NT va aloca în avans spatiu pe disc pentru fișierele de pe FAT dar în cazul unui eșec, fișierele care au fost adăugate vor apărea mai mari decât au fost vreodată scrise.

Numele lung al fisierelor VFAT

Fisierele VFAT cu nume lung (LFN) sunt stocate pe un sistem de fișiere FAT folosind un truc adăugându-se (eventual multiple) intrări suplimentare în director înainte de intrarea fișierelor normal. Intrările suplimentare sunt marcate cu atributele eticheta de volum, sistem, ascuns și Read -Only (rezultând 0x0F) care este o combinație neașteptat în mediul MS-DOS și prin urmare ignorate de programele MS-DOS și utilitatile din partitia a 3-a. În special, un director care conține numai etichetele de volum este considerat ca fiind gol și este permis să fie eliminat o astfel de situație ivindu-se în cazul în care fișierele cu nume lung create sunt șterse din DOS-urile simple. Această metodă este foarte asemănătoare cu metoda DELWATCH de a utiliza atributul volumului pentru a ascunde așteptarea ștergerii fișierelor pentru o posibila viitoare nepermitere a stergerii.

Deoarece versiunile mai vechi ale DOS-ar putea confunda numele LFN în directorul rădăcină cu eticeta de volum, VFAT a fost conceput pentru a crea o etichetă de volum în directorul rădăcină înainte de a adăuga orice nume intreg de LFN (în cazul în care o etichetă de volum nu există deja).

Fiecare intrare poate conține până la 13 caractere UCS-2 (26 bytes) cu ajutorul unui câmp de înregistrare care conține dimensiunea fișierului sau marca de timp (dar nu in campul clusterului de pornire pentru compatibilitate cu utilitățile de disc, câmpul clusterului de inceput fiind stabilit la o valoare de 0.

După ultimele caractere UCS-2 se adaugă un 0x0000. Caracterele neutilizate rămase sunt initializate cu 0xFFFF.

Intrarile LFN folosesc urmatorul format:

Byte Offset	Length (bytes)	Description

0x00	1	Numărul de ordine (bit 6: ultima intrare logic, prima intrare fizica , bit 5: 0; biții 4-0: numerele 0x01 .. 0x14 (0x1F), intrare șterse: 0xE5)
0x01	10	Numele caracterelor (Cinci caractere UCS-2)
0x0B	1	Atribute (`0x0F`)
0x0C	1	Tip (intotdeauna `0x00` pentru VFAT LFN, alte valori rezervate pentru utilizari viitoare; pentru folosire speciala a bitilor 3 si 4 in SFN-uri
0x0D	1	Suma de control pentru numele fisierelor din DOS
0x0E	12	Numele caracterelor (6 caractere UCS-2)
0x1A	2	Primul cluster (mereu `0x0000`)
0x1C	4	Numele caracterelor (two characters UCS-2)

Dacă există înregistrări LFN multe, necesare pentru a reprezenta un nume de fișier vine în primul rând la intrarea LFN ultima parte a fișierului. Numărul de ordine are bitul 6 setat (0x40) (aceasta înseamnă ultima intrare LFN, însă e prima intrare văzuta atunci când citim fisierul). Ultima intrare LFN are cel mai mare număr de ordine, care scade la următoarele intrari. Prima intrare LFN are numărul de ordine 1. O valoare de 0xE5 este utilizata pentru a indica faptul că intrarea este ștearsa

De exemplu, daca avem numele de fisier "File cu very long filename.ext" va fi formatat in felul urmator:

Sequence number	Entry data
0x43	"me.ext"
0x02	"y long filena"
0x01	"File cu ver"
???	Normal 8.3 entry

Un control permite, de asemenea, verificarea dacă un nume de fișier lung se potrivește cu sistemul 8.3 filename. O astfel de nepotrivire ar putea apărea dacă un fișier a fost șters și recreat folosind DOS în aceeași poziție.Controlul este calculat folosind algoritmul de mai jos. (Rețineți că pFCBName este un pointer la nume așa cum apare într-o intrare normala , adică primele opt caractere sunt numele fișierului, iar ultimele trei sunt extensia. Punctul este implicit.

```
unsigned char lfn_checksum(const unsigned char *pFCBName)
{
   int i;
   unsigned char sum = 0;
```

```
    for (i = 11; i; i--)
        sum = ((sum & 1) << 7) + (sum >> 1) + *pFCBName++;

    return sum;
}
```

În computere , Global File System 2 sau GFS2 este un sistem comun de fișiere pe disc pentru grupurile de clustere de la Linux. GFS2 diferă de la sisteme de fișiere distribuite (cum ar fi AFS, Coda sau Intermezzo) deoarece GFS2 permite tuturor nodurile sa aiba acces concurent direct la același bloc de stocare partajat. În plus, GFS sau GFS2 poate fi, de asemenea, utilizat ca un sistem de fișiere local.

GFS nu are un mod de operare discontinuu și nici roluri client sau server. Toate nodurile dintr-un cluster GFS functioneaza ca niste colegi.

NTFS. B – TREE

Pe scurt, sistemul de fisiere NTFS a ajutat alaturi de sistemul FAT la revolutionarea si dezvoltarea industriei de calculatoare si de tehnologie a informatiei.Este un punct de referinta in industria IT.De-a lungul timpului au fost analizate foarte multe tipuri de sisteme de fisiere insa toate aceste sisteme au fost derivate sau au continut notiuni fundamentale ale claselor **FAT si NTFS**.

Un **B – tree** reprezinta un „copac de arie N” cu un numar variabil (in cele mai multe cazuri) mare de „copii per nod”.
Contine „o radacina, noduri interne si frunze”.Radacina poate fi atat frunza cat si nod cu 2 sau mai multi copii.
În informatică, un B - TREE este o structură de date de tip arbore care păstrează datele sortate și permite căutări, acces secvențial, inserții și ștergeri într-un timp foarte scurt.B-TREE este o generalizare a unui arbore binar de căutare în care un nod poate avea mai mult de doi copii.Spre deosebire de arborii de cautare conventionali, B-TREE este optimizat pentru sisteme care citesc si scriu blocuri mari de date. Acesta este frecvent utilizat în bazele de date și sisteme de fișiere.
NTFS foloseste acesta structura pentru indexarea metadatelor (adica clasificarea si repartizarea concreta a „datelor datelor”)
 Factorul de ramificare , b al unui B – TREE măsoară capacitatea de noduri (de exemplu, numărul de noduri copii) pentru nodurile interne din copac.

Tipul Nodului	Tipul copiilor	Nr.min Copii	Nr.max Copii	Exemplu b = 7	Exemplu b = 100

Nodul Radacina (cand e singurul nod din copac)	Inregistrari	1	b	1 - 7	1 - 100
Nodul Radacina	Noduri interne sau noduri frunza	2	b	2 - 7	2 - 100
Nodul intern	Noduri interne sau noduri frunza	$\lceil b/2 \rceil$	b	4 - 7	50 - 100
Nod Frunza	Inregistrari	$\lfloor b/2 \rfloor$	b - 1	3 - 6	50 - 99

Cautam o valoare k în B - TREE. Pornind de la rădăcină, suntem în căutarea frunzei care poate conține valoarea k. La fiecare nod, ne dăm seama care indicator intern ar trebui să-l urmam. Un nod intern din B - TREE are cel mult d ≤ b copii, unde fiecare dintre ei reprezintă un alt sub-interval.

```
Function: tree_search (k, node)
                if node is a leaf
                then return node;
 switch k do
                 case k < k_0
                return tree_search(k, p_0);
                 case k_i ≤ k < k_{i+1}
                return tree_search(k, p_i);
                 case k_d ≤ k
                return tree_search(k, p_d);
```

Acest pseudocod presupune că valorile duplicate nu sunt permise.

Inserare

Efectuați o căutare pentru a determina ceea ce zona(„galeata”) noua inregistrare ar trebui sa fie stocata.
Dacă zona nu este completa (cel mult b - 1 înregistrări după inserție), adăugați înregistrarea.
În caz contrar, împărțit zona(„galeata”).
Aloca noi frunze și muta jumătatea elementelor din găleată in cealalta jumatate.

Introduceți adresa si cheia celei mai putin semnificative frunze la părinte.
În cazul în care părintele este plin, divizati-l si pe acesta.
Adăugați cheia de mijloc la nodul părinte.
Repetați până când un părinte nu mai e nevoie sa fie divizat.
În cazul în care rădăcina se desparte, creati un noua rădăcină care are o cheie și doi indicatori.

Stergerea

Începe de la rădăcină, găsi frunza L care apartine intrarii.
Scoateți intrarea.
Dacă L este cel puțin pe jumătate plin, procesul s-a terminat!
Dacă L are mai puține puncte decât ar trebui,incercați să redistribuiti, imprumutand de la frate (nodul adiacent cu aceeași mamă, L).
Dacă redistribuirea nu merge, unitii L si fratele.
Dacă îmbinare a avut loc, trebuie să ștergeți intrarea.
Unirea s-ar putea propaga pana la rădăcină.

Implementare

Frunzele (cele mai de jos indexate blocuri) ale B –TREE sunt legate intre ele.Aceasta nu mareste in mod semnificativ consumul spatiului sau managementul arborelui.Aceasta ilustreaza unul din avantajele importante ale B –TREE: Intr-un B- TREE din moment ce nu toate cheile sunt prezente in frunze, o lista inlantuita nu poate fi creata, asa ca un B –TREE este util ca un index pentru bazele sistemului, unde pur si simplu data se localizeaza pe disc.Daca un sistem de stocare are un bloc de B octeti si cheile care urmeaza sa fie stocate au o dimensiune de k, cel mai eficient B – TREE este acolo unde b=(B/k) -1.Desi in teorie lucrurile stau perfect, in practica va exista un mic spatiu extra folosit pentru index-ul blocurilor.Avand un index al blocurilor care este usor mai mare decat spatiul blocului de sistem vom avea o scadere semnificativa a performantei.
Daca nodurile B-TREE-ului sunt organizate ca niste elemente de matrice, atunci vom avea nevoie de un timp considerabil pentru a insera sau sterge un element deoarece jumatate din matrice va fi nevoita sa fie shiftata .Pentru a rezolva aceasta problema, elementele din interiorul unui nod pot fi organizate intr-un arbore binar sau B-TREE in loc de matrice.
B-TREE-urile pot fi deasemenea folosite pentru stocarea datelor in RAM.In acest caz o alegere rezonabila pentru dimensiunea blocurilor va fi dimensiunea cache-ului de la procesor. Cu toate acestea, unele studii au demonstrat ca o dimensiune a blocului de câteva ori mai mare decât cache-ul procesorului poate oferi o performanță mai bună în cazul în care este folosit prefetching-ul cache.

Comparatii intre sisteme de fisiere

Pentru o analiza mai eficienta , voi prezenta sub forma de tabel diferite caracteristici si notiuni existente in diferite formate ale sistemelor de fisiere:

Informatii generale:

Sistem de fisiere	Creator	Anul introducerii	Sistemul de operare original
ADFS	Acorn Computers Ltd	1983	Acorn Electron (later Arthur RISC OS)
DFS	Acorn Computers Ltd	1982	Acorn BBC Micro MOS
Romeo	Adaptec	1996	Microsoft Windows
MINIX V1 FS	Andrew S. Tanenbaum	1987	MINIX 1.0
MINIX V2 FS	Andrew S. Tanenbaum	1992	MINIX 1.6 and 2.0
MINIX V3 FS	Andrew S. Tanenbaum	2005	MINIX 3
DOS 3.x	Apple Computer	1978	Apple DOS
HFS	Apple Computer	1985	Mac OS
HFS Plus	Apple Computer	1998	Mac OS 8.1
MFS	Apple Computer	1984	Mac OS
Pascal	Apple Computer	1978	Apple Pascal
ProDOS	Apple Computer	1983	ProDOS 8
Be File System	Be Inc., D. Giampaolo, C. Meurillon	1996	BeOS
Fossil	Bell Labs	2003	Plan 9 from Bell Labs 4
V6FS	Bell Labs	1975	Version 6 Unix
V7FS	Bell Labs	1979	Version 7 Unix
AFS	Carnegie Mellon University	1982	Multiplatform MultoOS
Lustre	Cluster File Systems (later Oracle Corporation)	2002	Linux
Amiga FFS	Commodore	1988	Amiga OS 1.3
CBM DOS	Commodore	1978	Microsoft BASIC (for CBM PET)
FAT16B	Compaq	1987	Compaq MS-DOS 3.31, DR DOS 3.31
Lanyard Filesystem	Dan Luedtke	2012	Linux

Sistem de fisiere	Creator	Anul introducerii	Sistemul de operare original
Reliance Nitro[4]	Datalight	2009	Windows CE, Windows Mobile, VxWorks, Linux, custom ports
Reliance[4]	Datalight	2003	Windows CE, VxWorks, custom ports
AdvFS	DEC	1993[2]	Digital Unix
DECtape	DEC	1964	PDP-6 Monitor
Level-D	DEC	1968	TOPS-10
ODS-1	DEC	1972	RSX-11
ODS-2	DEC	1979	OpenVMS
ODS-5	DEC	1998	OpenVMS 7.2
RT-11 file system	DEC	1973	RT-11
High Sierra	Ecma International	1985	MS-DOS, Mac OS
ISO 9660:1988	Ecma International, Microsoft	1988	MS-DOS, Mac OS, and AmigaOS
ISO 9660:1999	Ecma International, Microsoft	1999	Microsoft Windows, Linux, Mac OS X, FreeBSD, and AmigaOS
CP/M file system	Gary Kildall	1974	CP/M
DOS (GEC)	GEC	1973	Core Operating System
OS4000	GEC	1977	OS4000
Google File System	Google	2003	Linux
GPFS	IBM	1996	AIX, Linux, Windows
HFS (Hierarchical File System)	IBM	1994	MVS/ESA (now z/OS)
JFS	IBM	1999	OS/2 Warp Server for e-business
JFS1	IBM	1990	AIX[1]
LTFS	IBM	2010	Linux, Mac OS X, planned Microsoft Windows,
zFS	IBM	2001	z/OS (backported to OS/390)
HPFS	IBM & Microsoft	1988	OS/2
George 2	ICT (later ICL)	1968	George 2
IlesfayFS	Ilesfay Technology Group	2011	Microsoft Windows, planned Red Hat Enterprise Linux
UDF	ISO/ECMA/OSTA	1995	-
SFS	John Hendrikx	1998	AmigaOS, AROS, MorphOS
FFS	Kirk McKusick	1983	4.2BSD
UFS1	Kirk McKusick	1994	4.4BSD
UFS2	Kirk McKusick	2002	FreeBSD 5.0

Sistem de fisiere	Creator	Anul introducerii	Sistemul de operare original
QFS	LSC Inc, Sun Microsystems	1996	Solaris
FAT (8-bit)	Marc McDonald, Microsoft	1977	Microsoft Disk BASIC
LFS	Margo Seltzer	1993	Berkeley Sprite
HAMMER	Matthew Dillon	2008	Dragonfly BSD
Amiga OFS[15]	Metacomco for Commodore	1985	Amiga OS
PFS	Michiel Pelt	1996	AmigaOS
exFAT	Microsoft	2006, 2009	Windows CE 6.0, Windows XP SP3, Windows Vista SP1
FAT16	Microsoft	1984	MS-DOS 3.0
FAT32	Microsoft	1996	Windows 95b[3]
FATX	Microsoft	2002	Xbox
Joliet ("CDFS")	Microsoft	1995	Microsoft Windows, Linux, Mac OS, and FreeBSD
NTFS Version 3.1	Microsoft	2001	Windows XP
ReFS	Microsoft	2012, 2013	Windows 2012 Server
TexFAT/TFAT	Microsoft	2006	Windows CE 6.0
NTFS Version 1.0	Microsoft, Tom Miller, Gary Kimura	1993	Windows NT 3.1
PramFS	MontaVista	2003	Linux
Reiser4	Namesys	2004	Linux
ReiserFS	Namesys	2001	Linux
WAFL	NetApp	1992	Data ONTAP
UBIFS	Nokia cu help of University of Szeged	2008	Linux
NSS	Novell	1998	NetWare 5
NWFS	Novell	1985	NetWare 286
Elektronika BK tape format	NPO "Scientific centre" (now Sitronics)	1985	Vilnius Basic, BK monitor program
NILFS	NTT	2005	Linux, (ReadOnly for NetBSD)
Btrfs	Oracle Corporation	2007	Linux
OCFS	Oracle Corporation	2002	Linux
OCFS2	Oracle Corporation	2005	Linux
Oracle ACFS	Oracle Corporation	2009	Linux - Red Hat Enterprise Linux 5 and Oracle Enterprise Linux 5 only
Non-Volatile File System	Palm, Inc.	2004	Palm OS Garnet
PolyServe File System (PSFS)	PolyServe	1998	Windows, Linux

Sistem de fisiere	Creator	Anul introducerii	Sistemul de operare original
GFS2	Red Hat	2006	Linux
ext	Rémy Card	1992	Linux
ext2	Rémy Card	1993	Linux, Hurd
Ceph	Sage Weil, Inktank Storage	2007, 2012	Linux
F2FS	Samsung	2012	Linux
Melio FS	Sanbolic	2001	Windows
XFS	SGI	1994	IRIX, Linux, FreeBSD
GFS	Sistina (Red Hat)	2000	Linux
ext3	Stephen Tweedie	1999	Linux
ZFS	Sun Microsystems	2004	Solaris, FreeBSD, PC-BSD, FreeNAS
FAT12	Tim Paterson	1980	QDOS, 86-DOS
ext4	Various	2006	Linux
Tux3	Various	2008	Linux
VxCFS	VERITAS, (now Symantec)	2004	AIX, HP-UX, Solaris, Linux
VxFS	VERITAS, (now Symantec)	1991	AIX, HP-UX, Solaris, Linux
VMFS2	VMware	2002	VMware ESX Server 2.0
VMFS3	VMware	2005	VMware ESX Server 3.0
VMFS5	VMware	2011	VMware ESXi 5.0tux 3 stats
Rock Ridge	Young Minds Inc.	1994	Linux, Mac OS, Amiga OS, and FreeBSD
ext3cow	Zachary Peterson	2003	Linux

SURSA: http://en.wikipedia.org/wiki/Comparison_of_file_systems#cite_ref-2

Limitari

Sistem de fisiere	Lungime maxima pentru numele fisierelor	Caractere permise in intrarile directoarelor	Lungimea maxima pentru numele caii	Dimensiunea maxima a fisierului	Dimensiunea maxima a volumului
Acorn ADFS	10 bytes	Any ISO 8859-1 character cu exceptia: SPACE $ & % @ \ ^ : . #	nicio limita definita	512 MB sau 4 GB	512 MB sau 4 GB
Apple DOS 3.x	30 bytes	Orice byte cu exceptia NUL	30 B, fara subdirectoare	necunoscut	113.75 kB DOS 3.1, 3.2 140 kB DOS 3.3

Sistem de fisiere	Lungime maxima pentru numele fisierelor	Caractere permise in intrarile directoarelor	Lungimea maxima pentru numele caii	Dimensiunea maxima a fisierului	Dimensiunea maxima a volumului
Apple ProDOS	15 bytes	A-Z, a-z, 0-9, and period	Necunoscut	16 MB	32 MB
CP/M file system	8.3	orice octet cu exceptia SPACE < > . , ; : = ? * [ ] % | ( ) / \[9]	16 "zone user", fara subdirectoare	8 MB[10]	8 MB to 512 MB[10]
IBM SFS	8.8	Necunoscut	Non - ierarhica	Necunoscut	Necunoscut
DECtape	6.3	A–Z, 0–9	DTxN:FILNAM.EXT = 15	369,280 B (577 * 640)	369,920 B (578 * 640)
Elektronika BK tape format	16 bytes	Necunoscut	Non - ierarhica	64 kB	Fara limita. Approx. 800 kB (one side) for 90 min cassette
MicroDOSfile system	14 bytes	Necunoscut	Necunoscut	16 MB	32 MB
Level-D	6.3	A–Z, 0–9	DEVICE:FILNAM.EXT[PROJCT,PROGRM] = 7 + 10 + 15 = 32; + 5*7 for SFDs = 67	24 GB (34,359,738,368 cuvinte (2^{35}-1);	12 GB (aprox; 64 * 178 MB)
RT-11	6.3	A–Z, 0–9, $	Non - ierarhica	32 MB (65536 * 512)	32 MB
V6FS	14 bytes[12]	Orice octet exceptand NUL and /	Fara limita stabilita	16 MB[15]	2 TB
DOS (GEC)	8 bytes	A–Z, 0–9	Non - ierarhica	64 MB	64 MB
OS4000	8 bytes	A–Z, 0–9	Fara limita stabilita	2 GB	1 GB (at least)
CBM DOS	16 bytes	Orice octet exceptand NUL	Non - ierarhica	16 MB	16 MB
V7FS	14 bytes[12]	Orice octet exceptand NUL and /[13]	Fara limita stabilita	1 GB[16]	2 TB
exFAT	255 characters[17]	Orice Unicode exceptand NUL	Fara limita stabilita	127 PB	64 ZB, 512 TB recomandat
TexFAT	247 characters	Orice Unicode exceptand NUL	Fara limita stabilita	2 GB	500 GB Testat
FAT12	8.3 (255 UTF-16 code units cu LFN)[12]	Orice octet exceptandfor values 0-31, 127 (DEL) and: " * / : < > ? \ | + , . ; = [] (lowcase a-z are stored as A-Z). With VFAT LFN	Fara limita stabilita[14]	32 MB (256 MB)	32 MB (256 MB)
FAT16	8.3 (255 UTF-16 code units cu LFN)[12]	Orice octet exceptandfor values 0-31, 127 (DEL) and: " * / : < > ? \ | + , . ; = []	Fara limita stabilita[14]	2 GB (4 GB)	2 GB sau 4 GB

Sistem de fisiere	Lungime maxima pentru numele fisierelor	Caractere permise in intrarile directoarelor	Lungimea maxima pentru numele caii	Dimensiunea maxima a fisierului	Dimensiunea maxima a volumului
		(lowcase a-z are stored as A-Z). With VFAT LFN anyUnicode except NUL[12][13]			
FAT32	8.3 (255 UTF-16 code units cu LFN)[12]	Orice octet exceptandfor values 0-31, 127 (DEL) and: " * / : < > ? \ | + , . ; = [] (lowcase a-z are stored as A-Z). With VFAT LFN anyUnicode except NUL[12][13]	Fara limita stabilita[14]	4 GB (256 GB[20])	2 TB[21] (16 TB)
FATX	42 bytes[12]	ASCII. Unicode not permitted.	Fara limita stabilita[14]	2 GB	2 GB
Fossil	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
MFS	255 bytes	Orice octet exceptand:	NU path (flat filesystem)	226 MB	226 MB
HFS	31 bytes	Orice octet exceptand:[22]	Unlimited	2 GB	2 TB
HPFS	255 bytes	Orice octet exceptand NUL[23]	Fara limita stabilita[14]	2 GB	2 TB[24]
NTFS	255 characters[25][26][27]	Depinde de spatiul de nume folosit[25][26][27][28]	32,767 Unicode characters cu each path component (directory sau filename) commonly up to 255 characters long[14]	16 EB[29]	16 EB[29]
ReFS	255 unicode characters [30]	Necunoscut	32 kB	16 EB	Format supports 256ZB cu 16kB cluster size (2^64 * 16 * 2^10). Windows stack addressing allows 16EB
HFS Plus	255 UTF-16 code units[31]	Any valid Unicode[13][32]	Unlimited	8 EB	8 EB[33][34]
FFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita[14]	8 ZB	8 ZB
UFS1	255 bytes	Orice octet exceptand NUL	Fara limita stabilita[14]	226 TB	226 TB
UFS2	255 bytes	Orice octet exceptand NUL	Fara limita stabilita[14]	32 PB	1 YB
ext2	255 bytes	Orice octet exceptand NUL and /	Fara limita stabilita[14]	2 TB[6]	32 TB
ext3	255 bytes	Orice octet exceptand NUL and /	Fara limita stabilita[14]	2 TB[6]	32 TB
ext3cow	255 bytes	Orice octet exceptand NUL, / and @	Fara limita stabilita[14]	2 TB[6]	32 TB
ext4	255 bytes	Orice octet exceptand NUL and /	Fara limita stabilita	16 TB[6][35]	1 EB[36]

Sistem de fisiere	Lungime maxima pentru numele fisierelor	Caractere permise in intrarile directoarelor	Lungimea maxima pentru numele caii	Dimensiunea maxima a fisierului	Dimensiunea maxima a volumului
Lustre	255 bytes	Orice octet exceptand NUL and /	Fara limita stabilita	32 PB (on ext4)	1 YB (on ext4, 20 PB tested)
GPFS	255 UTF-8codepoints	Orice octet exceptand NUL	Fara limita stabilita	512 YB	512 YB (4 PB tested)
GFS	255	Orice octet exceptand NUL	Fara limita stabilita	8 EB	8 EB
ReiserFS	4,032 bytes/226 characters	Orice octet exceptand NUL	Fara limita stabilita	8 TB[38] (v3.6), 2 GB (v3.5)	16 TB
NILFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	8 EB	8 EB
Reiser4	3,976 bytes	Orice octet exceptand/ and NUL	Fara limita stabilita	8 TB on x86	Necunoscut
OCFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	8 TB	8 TB
OCFS2	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	4 PB	4 PB
Reliance	260 bytes	OS specific	260 B	4 GB	2 TB
Reliance Nitro	1,024 bytes	OS specific	1024 bytes	32 TB	32 TB
JFS1	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	8 EB	4 PB
JFS	255 bytes	Any Unicode except NUL	Fara limita stabilita	4 PB	32 PB
QFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	16 EB	4 PB
BFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	260 GB	2 EB
AdvFS	226 characters	Orice octet exceptand NUL	Fara limita stabilita	16 TB	16 TB
NSS	226 characters	Depinde de spatiul de nume folosit	Only limited by client	8 TB	8 TB
NWFS	80 bytes	Depinde de spatiul de nume folosit	Fara limita stabilita	4 GB	1 TB
ODS-5	236 bytes	Necunoscut	4,096 bytes	2 TB	2 TB
VxFS	255 bytes	Orice octet exceptand NUL	Fara limita stabilita	256 TB	256 TB
UDF	255 bytes	Orice Unicode exceptand NUL	1,023 bytes	16 EB	2 TB (hard disk), 8 TB (optical disc)
MINIX V1 FS	14 sau 30 bytes, set at filesystem creation time	Orice octet exceptand NUL	Fara limita stabilita	64 MB	64 MB
MINIX V2 FS	14 sau 30 bytes, set at filesystem creation time	Orice octet exceptand NUL	Fara limita stabilita	4 GB	1 GB, then 2 TB
MINIX V3 FS	60 bytes	Orice octet exceptand NUL	Fara limita stabilita	4 GB	16 TB

Sistem de fisiere	Lungime maxima pentru numele fisierelor	Caractere permise in intrarile directoarelor	Lungimea maxima pentru numele caii	Dimensiunea maxima a fisierului	Dimensiunea maxima a volumului
VMFS2	128	Orice octet exceptand NUL and /	2,048	4 TB	64 TB
VMFS3	128	Orice octet exceptand NUL and /	2,048	2 TB	64 TB
ISO 9660:1988	Level 1: 8.3, Level 2 & 3: ~ 180	Depends on Level	~ 180 bytes?	4 GB (Level 1 & 2) to 8 TB (Level 3)	8 TB
Joliet ("CDFS")	64 Unicodecharacters	Toate codurile UCS-2 exceptand * / \ : ; and ?	Necunoscut	4 GB (same as ISO 9660:1988)	8 TB (same as ISO 9660:1988)
ISO 9660:1999	Necunoscut (207?)	Necunoscut	Necunoscut	Necunoscut	Necunoscut
High Sierra	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
HAMMER	Necunoscut	Necunoscut	Necunoscut	Necunoscut	1 EB
LTFS	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
PramFS	31 bytes	Orice octet exceptand NUL	Necunoscut	1 GB	8 EB
Lanyard Filesystem	255 bytes	Orice octet exceptand NUL and /	Fara limita stabilita	64 ZB	128 kB to 64 ZB
LEAN	4,068 bytes[54]	case sensitive, in UTF-8	Fara limita stabilita	8 EB	8 EB
XFS	255 bytes[55]	Orice octet exceptand NUL	Fara limita stabilita[14]	8 EB	8 EB
ZFS	255 bytes	Orice Unicode exceptand NUL	Fara limita stabilita[14]	16 EB	16 EB
Btrfs	255 bytes	Orice octet exceptand NUL	Necunoscut	16 EB	16 EB

SURSA: http://en.wikipedia.org/wiki/Comparison_of_file_systems#cite_ref-2

Caracteristici si proprietati

Sistem de fisiere	Conexiuni hard	Conexiuni simbolic	Case-sensitive	Case-preserving	File Change Log	Snapshot	XIP	Criptare	COW	LWM integrat	Deduplicarea datelor	Redimensionarea Volumelor
Lanyard Filesystem	NU	NU	DA	DA	NU	NU	NU	NU	NU	NU	NU	Nedisponibil
CBM DOS	NU	NU	DA	DA	NU	NU	NU	NU	NU	NU	NU	NU
CP/M file system	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut
DECtape	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut
Level-D	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
RT-11	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut
DOS (GEC)	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut
OS4000	NU	DA[86]	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut
V6FS	DA	NU	DA	DA	NU	NU	NU	NU	NU	NU	NU	Necunoscut
V7FS	DA	NU[87]	DA	DA	NU	NU	NU	NU	NU	NU	NU	Necunoscut
FAT12	NU	NU	NU	Partial	NU	NU	NU	NU	NU	NU	NU	Nedisponibil
FAT16	NU	NU	NU	Partial	NU	NU	NU	NU	NU	NU	NU	Nedisponibil
FAT32	NU	NU	NU	Partial	NU	NU	NU	NU	NU	NU	NU	Nedisponibil
exFAT	NU	NU	NU	DA	NU	Necunoscut	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
GFS	DA	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Disponibil
GPFS	DA	DA	DA	DA	DA	DA	DA	NU	DA	DA	NU	Disponibil
HAMMER	DA	DA	DA	DA	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	La cerere	Necunoscut
HPFS	NU	NU	NU	DA	NU	Necunoscut	NU	NU	Necunoscut	Necunoscut	NU	Necunoscut
NTFS	DA	DA	DA[93]	DA	DA	Partial	DA	DA	Partial	Necunoscut	DA (Windows Server 2012)	Disponibil
HFS	NU	DA	NU	DA	NU	NU	NU	NU	NU	NU	NU	Necunoscut
HFS Plus	DA	DA	Partial	DA	DA[101]	NU	NU	DA	NU	NU	NU	DA
FFS	DA	DA	DA	DA	NU	NU	NU	NU	NU	NU	NU	Nedisponibil

Sistem de fisiere	Conexiuni hard	Conexiuni simbolic	Case-sensitive	Case-preserving	File Change Log	Snapshot	XIP	Criptare	COW	LWM integrat	Deduplicarea datelor	Redimensionarea Volumelor
UFS1	DA	DA	DA	DA	NU	NU	NU	NU	NU	NU	NU	Necunoscut
UFS2	DA	DA	DA	DA	NU	DA	Necunoscut	NU	NU	NU	NU	Nedisponibil
LFS	DA	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
ext2	DA	DA	DA	DA	NU	NU	DA	NU	NU	NU	NU	Disponibil
ext3	DA	DA	DA	DA	NU	NU	DA	DA	NU	NU	NU	Disponibil
ext3cow	DA	DA	DA	DA	Necunoscut	DA	Necunoscut	DA	DA	NU	NU	Necunoscut
ext4	DA	DA	DA	DA	NU	NU	DA	DA	NU	NU	NU	Disponibil
Lustre	DA	DA	DA	DA	DA in 2.0 and later	NU	NU	NU	NU	NU	NU	Disponibil
NILFS	DA	DA	DA	DA	DA	DA	NU	NU	DA	Necunoscut	Necunoscut	Disponibil
ReiserFS	DA	DA	DA	DA	NU	NU	NU	NU	NU	NU	NU	Disponibil
Reiser4	DA	DA	DA	DA	NU	Necunoscut	NU	DA	DA	NU	Necunoscut	Disponibil
OCFS	NU	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
OCFS2	DA	DA	DA	DA	NU	Partial[116]	NU	NU	Necunoscut	NU	NU	Disponibil
Reliance	NU	NU	NU	DA	NU	NU	NU	NU	DA	NU	NU	Necunoscut
Reliance Nitro	DA	DA	Depends de OS	DA	NU	NU	NU	NU	DA	NU	NU	Necunoscut
XFS	DA	DA	DA	DA	NU	NU	NU	NU	NU	NU	NU	Disponibil
JFS	DA	DA	DA[119]	DA	NU	DA	NU	NU	NU	Necunoscut	Necunoscut	Disponibil [120]
QFS	DA	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
Be File System	DA	DA	DA	DA	Necunoscut	NU	NU	NU	NU	NU	NU	Necunoscut
NSS	DA	DA	DA[121]	DA[121]	DA[122]	DA	NU	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut
NWFS	DA[123]	DA[123]	DA[121]	DA[121]	DA[122]	Necunoscut	NU	NU	NU	DA	Necunoscut	Necunoscut
ODS-2	DA	DA[125]	NU	NU	DA	DA	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
ODS-5	DA	DA[125]	NU	DA	DA	DA	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
UDF	DA	DA	DA	DA	NU	NU	DA	NU	NU	NU	NU	Necunoscut
VxFS	DA	DA	DA	DA	DA	DA	Necunoscut	NU	Necunoscut	Necunoscut	DA	Necunoscut
Fossil	NU	NU	DA	DA	DA	DA	NU	NU	Necunoscut	NU	DA	Necunoscut
ZFS	DA	DA	DA	DA	NU	DA	NU	DA	DA	DA	DA	Disponibil

Sistem de fisiere	Conexiuni hard	Conexiuni simbolic	Case-sensitive	Case-preserving	File Change Log	Snapshot	XIP	Criptare	COW	LWM integrat	Deduplicarea datelor	Redimensionarea Volumelor
VMFS2	DA	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
VMFS3	DA	DA	DA	DA	NU	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
Btrfs	DA	DA	DA	DA	DA	DA	NU	Planned[130]	DA	DA	DA	Disponibil
PramFS	NU	DA	DA	DA	NU	NU	DA	NU	NU	NU	NU	NU

SURSA: http://en.wikipedia.org/wiki/Comparison_of_file_systems#cite_ref-2

Compatibilitate cu sisteme de operare

Sistemul de fisiere	DOS	Windows 9x	Windows NT	Linux	Mac OS	Mac OS X	FreeBSD	BeOS	Solaris	AIX	z/OS	OS/2	Windows CE	Windows Mobile	VxWorks	HP-UX
FAT12	DA	DA	DA	DA	DA	DA	DA	DA	NU	Doar pe dischete prin comenzi dos*	Necunoscut	DA	DA	Necunoscut	DA	Necunoscut
FAT16	DA incepand cu DOS 3.0, FAT16B incepand cu DOS 3.31	DA	DA	DA	DA	DA	DA	DA	DA	Doar pe dischete prin comenzi dos*	Necunoscut	DA	DA	DA	DA	Necunoscut
FAT32	DA incepand cu DOS 7.1	DA incepand cu Windows 95OSR2	DA incepand cu Windows 2000	DA	DA	DA	DA	DA	DA	Doar pe dischete prin comenzi dos*	Necunoscut	DA	DA	DA	DA	Necunoscut
exFAT	NU	DA	DA : Win7, Vista SP1, poate fi adaugat si pe XP SP2	DA	NU	DA 10.6.5+	NU	NU	DA	NU	NU	NU	DA	NU	Necunoscut	Necunoscut
NTFS	cu driver	cu	DA	DA Kernel	cuNTFS-	read-only	cu NTFS-	cuNTFS-3G	cu NTFS-	Necunosc	Necunoscut	read-only	cu 3rd-	NU	Necunoscut	Necunoscut

Sistemul de fisiere	DOS	Windows 9x	Windows NT	Linux	Mac OS	Mac OS X	FreeBSD	BeOS	Solaris	AIX	z/OS	OS/2	Windows CE	Windows Mobile	VxWorks	HP-UX
		driver[150]		2.2 sau newer, sau cuNTFS-3G sauntfsprogs	3GsauMacFUSE	partial(read-write cuNTFS-3G)	3G		3Gon Opensolaris	ut		partial	party driver[152]			
HFS	NU	DA	NU	DA	DA	read-only partialincepand cu OSX 10.6[154]	NU	Necunoscut	Necunoscut	Necunoscut	NU	DA	NU	NU	NU	Necunoscut
HFS Plus	NU	NU	NU	write-only	DA	DA	read-only partial	Necunoscut	Necunoscut	Necunoscut	NU	DA	NU	NU	NU	Necunoscut
HPFS	cu driver	read-only partialdriver[159]	included until v3.51, driver until 4.0[160]	DA	NU	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	DA	NU	Necunoscut	Necunoscut	Necunoscut
FFS	NU	Necunoscut	Necunoscut	DA[161]	NU	DA	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
UFS1	NU	Necunoscut	Necunoscut	Partial - read only	NU	DA	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut
UFS2	NU	Necunoscut	Necunoscut	Partial - read only	NU	NU	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut
ext2	Necunoscut	Necunoscut	cu Ext2Fsd (complete) sau Ext2 IFS (partial, no large inodes)[163] sau Ext2Read (read-only, also on LVM2)	DA	NU	cu fuse-ext2, ExtFS and ext2fsx	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	app[168]	cu 3rd-party app	cu 3rd-party app	Necunoscut	Necunoscut
ext3	Necunoscut	Necunoscut	cu Ext2Fsd (complete) sau Ext2	DA	NU	cu fuse-ext2 and ExtFS	DA	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	cu 3rd-party app	cu 3rd-party app	Necunoscut	Necunoscut

Sistem ul de fisiere	DOS	Windo ws 9x	Windo ws NT	Linux	Mac OS	Mac OS X	FreeBS D	BeOS	Solaris	AIX	z/OS	OS/2	Window s CE	Window s Mobile	VxWork s	HP-UX
			IFS (partial, no large inodes) sa uExt2Read (read-only, also on LVM2)													
ext3cow	Necunosc	Necunosc	Necunosc	DA Kernel	Necunosc	Necunos	Necunosc	Necunosc	Necunosc	Necunosc	Necunosc	Necunoscut	Necunoscut	Necunosc	Necunoscut	Necunoscut
ext4	NU	NU	cu Ext2Fsd (partial, extents limited)[162] sau Ext2Read (read-only, also on LVM2)[164]	DA incepand cu kernel 2.6.28	NU	cu fuse-ext2 (partial)[165] and ExtFS (full read/write)[166]	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut
Btrfs	NU	NU	NU	DA	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut	Necunoscut
ZFS	NU	NU	NU	cu 3rd Party kernel module[170] sau FUSE[171]	NU	cu free 3rd-party software[172]	DA	NU	DA	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
Lustre	NU	NU	Partial - under development[173]	DA[174]	NU	Partial - via FUSE	Partial - via FUSE	NU	Partial - under development[175]	NU	NU	NU	NU	NU	Necunoscut	Necunoscut
GFS	NU	Necunoscut	Necunoscut	DA	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
NILFS	NU	Necunoscut	Necunoscut	DA incepand cu kernel 2.6.30	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
ReiserFS	NU	Necunoscut	Partial cu app	DA	NU	NU	Partial - read only	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	cu 3rd-party app[169]	cu 3rd-party app[169]	Necunoscut	Necunoscut

Sistemul de fisiere	DOS	Windows 9x	Windows NT	Linux	Mac OS	Mac OS X	FreeBSD	BeOS	Solaris	AIX	z/OS	OS/2	Windows CE	Windows Mobile	VxWorks	HP-UX
Reiser4	NU	Necunoscut	Necunoscut	cu a kernel patch	NU	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
OCFS	NU	Necunoscut	Necunoscut	DA	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
OCFS2	NU	Necunoscut	Necunoscut	DA	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
Reliance	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	DA	NU	DA	Necunoscut
Reliance Nitro	NU	NU	NU	DA	NU	NU	NU	NU	NU	NU	NU	NU	DA	DA	DA	Necunoscut
XFS	NU	Necunoscut	Necunoscut	DA	NU	Necunoscut	Partial	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
JFS	NU	Necunoscut	Necunoscut	DA	NU	NU	NU	Necunoscut	Necunoscut	DA	Necunoscut	DA	NU	NU	Necunoscut	
QFS	NU	Necunoscut	Necunoscut	via client software[176]	NU	Necunoscut	NU	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
BFS	NU	Necunoscut	Necunoscut	Partial - read-only	NU	Necunoscut	NU	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
NSS	Necunoscut	Necunoscut	Necunoscut	cu Novell OES2[*citation needed*]	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
NWFS	Necunoscut	Necunoscut	Necunoscut	via ncpfs client software[177]	NU	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
UDF	Necunoscut	read-only partialsupport of UDF 1.02 incepand cu Win98 and WinME	DA[178]	DA	DA incepand cu Mac OS 9	DA	DA	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut	DA	Necunoscut	Necunoscut	Necunoscut
VxFS	NU	Necunoscut	Necunoscut	DA	NU	Necunoscut	NU	Necunoscut	DA	DA	Necunoscut	Necunoscut	NU	NU	Necunoscut	DA
Fossil	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut

Sistem ul de fisiere	DOS	Windo ws 9x	Windo ws NT	Linux	Mac OS	Mac OS X	FreeBS D	BeOS	Solaris	AIX	z/OS	OS/2	Window s CE	Window s Mobile	VxWork s	HP-UX
IBM HFS	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	DA	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
IBM zFS	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	DA	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
IBMGPFS[179]	NU	NU	DA	DA	NU	NU	NU	NU	NU	DA	NU	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut
VMFS2	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	
VMFS3	NU	Necunoscut	Necunoscut	read-only partialwith vmfs[180]	Necunoscut	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
DECtape	NU	Necunoscut	Necunoscut	cu AncientFS[181]	NU	cu AncientFS[181]	cu AncientFS[181]	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
Level-D	NU	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
RT-11	NU	Necunoscut	Necunoscut	Necunoscut	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	DA	Necunoscut
ODS-2	NU	Necunoscut	Necunoscut	read-only partialwith tool sau kernel module[182]	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
ODS-5	NU	Necunoscut	Necunoscut	read-only partialwith kernel module[182]	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut
LFS	NU	Necunoscut	Necunoscut	cu logfs[183] a nd others	NU	Necunoscut	NU	Necunoscut	Necunoscut	Necunoscut	Necunoscut	Necunoscut	NU	NU	Necunoscut	Necunoscut
LTFS	NU	Necunoscut	Necunoscut	DA	NU	DA	NU	NU	NU	NU	NU	NU	NU	NU	Necunoscut	Necunoscut
PramFS	NU	NU	NU	DA	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU	NU

SURSA: http://en.wikipedia.org/wiki/Comparison_of_file_systems#cite_ref-2

Concluzii personale

Oamenii obisnuiti folosesc calculatoarele, laptopurile si ale dispozitive multimedia fara a interactiona intr-un mod anume cu aceste notiuni.Ele lucreaza fizic la un nivel innaccesibil omului si in functie de tehnologie ne ajuta pe noi, utilizatorii sa lucram cu viteze mai mari de procesare, sa gestionam altfel anumite activitati. Diferite sisteme de operare utilizează sisteme diferite de fişiere. Cele două sisteme populare sunt FAT32 şi NTFS. Momentan este de dorit sa se utilizeze NTFS pentru ca celelalte sisteme de fisiere ar limita si incetini activitatea unui sistem de operare.Pentru un calculator cu sistem multi-boot, se poate lua in considerare cazul FAT32.

Bibliografie si referinte

1. "File Systems". Microsoft TechNet. 2001. – Editie imbunatatia in 31.07.2011 - (paginile 7, 8, 28) + tabel 1
2. Udo Kuhnt, Luchezar Georgiev, Jeremy Davis (2007). *FAT+*. FATPLUS.TXT, draft revision 2 ([1],[2]). – (paginile 1, 2, 4,6,8,9, 21, 24, 25, 26)
3. "Volume and File Structure of Disk Cartridges for Information Interchange". *Standard ECMA-107 (2nd ed., June 1995)*. ECMA. 1995.- paginile 1, 2, 3,5, 11, 12, 13,15,18,21,22,24, 28) + tabel 2, 3
4. "Information technology -- Volume and file structure of disk cartridges for information interchange".*ISO/IEC 9293:1994*. ISO catalogue. 1994. Editie imbunatatita 21.06.2010 – paginile 1,2,3,4,5,6,9,10,11,14,17,22)
5. "Information processing -- Volume and file structure of flexible disk cartridges for information interchange". *ISO 9293:1987*. ISO catalogue. 1987. Editie imbunatatita 06.01.2012 – paginile 1, 2, 3, 4....14, 16,22,24,25,26,27) + tabel 4
6. ^ Aaron R. Reynolds, Dennis R. Adler, Ralph A. Lipe, Ray D. Pedrizetti, Jeffrey T. Parsons, Rasipuram V. Arun (1998-05-26). "Common name space for long and short filenames".*US Patent 5758352*. Editie imbunatatita 2012-01-19. – paginile 17,18
7. Ray Duncan (1988). *The MS-DOS Encyclopedia - version 1.0 through 3.2*. Microsoft Press. ISBN 1-55615-049-0. – paginile 1,2.....28
8. Tim Paterson (2007-09-30). "Design of DOS".*DosMan Drivel*. Editie imbunatatita 2011-07-04. – paginile 8,9.10,11,12,15,17,18
9. Seattle Computer Products (1981). "SCP 86-DOS 1.0 Addendum". Editie imbunatatita 2013-03-10. – paginile 1,2,5,6,7

10. "Microsoft Extensible Firmware Initiative FAT32 File System Specification, FAT: General Overview of On-Disk Format". Microsoft. 2000-12-06. Editie imbunatatita 2011-07-03. – paginile 1,2,3,4,5,6,7,11,12,14,15,18,22,23,25,28
11. Andries Brouwer. "FAT under Linux". – paginile 3, 4,5,6,7,8
12. Geoff Chappell (1994). *DOS Internals*. Addison Wesley. ISBN 0-201-60835-9, ISBN 978-0-201-60835-9.- paginile 2,3,4,6,7,11,14,15,16,18,24
13. Tim Paterson (1983). "An Inside Look at MS-DOS". Byte. Editie imbunatatita 2011-07-18. – paginile 8,9,12,15,16,17,21,22 + tabel 1, 2
14. Microsoft TechNet. 1998. Editie imbunatatita 2012-07-16. – paginile 2,3,4,12,13,14,15,16,19,20,21,22,26
15. "Limitations of FAT32 File System". Microsoft Knowledge Base. 2007-03-26. Editie imbunatatita 2011-08-21. – paginile 33,34,35,36,37
16. IBM. *4690 OS User's Guide Version 5.2*, IBM document SC30-4134-01, 2008-01-10 ([6]). – paginile 10 ,11,12
17. Bob Eager, Tavi Systems (2000-10-28).*Implementation of extended attributes on the FAT file system*. ([7]). – paginile 7,8
18. Bob Eager (2000-10-28). "Implementation of extended attributes on the FAT file system". *Tavi OS/2 pages*. Editie imbunatatita 2006-10-14. – paginile 7, 8
19. "FATX Specification". free60 wiki. Editie imbunatatita 2011-08-16. – paginile 8,34,35
20. http://www.seanster.com/BplusTree/BplusTree.html - paginile 32,33,34
21. Microsoft. "exFAT File System Intellectual Property licensing program". Editie imbunatatita 2013-04-23. – paginile 8,32,34,40
22. Daniel B. Sedory. *The Boot Sector of IBM® Personal Computer™ DOS Version 1.00 (1981)*. 2005-08-02 ([11]). – paginile 6,9,10,22
23. "Detailed Explanation of FAT Boot Sector". DEW Associates Corporation. 2002. Editie imbunatatita 2011-10-16. – paginile 9, 10,11
24. Navathe, Ramez Elmasri, Shamkant B. (2010). *Fundamentals of database systems* (6th ed. ed.). Upper Saddle River, N.J.: Pearson Education. pp. 652–660. ISBN 9780136086208. – paginile 32,33,34
25. http://en.wikipedia.org/wiki/Comparison_of_file_systems#cite_ref-2 – paginile 35,36...45