Московский Энергетический Институт (Технический Университет)

Курсовая работа

по предмету «Операционные системы» на тему:

Файловые системы

Группа: **А-13-03**
Студент: **Ясенков Е.М.**

Москва 2007

Содержание

Введение

Файловая система (ФС) является важной частью любой операционной системы, которая отвечает за организацию хранения и доступа к информации на каких-либо носителях. Рассмотрим в качестве примера файловые системы для наиболее распространенных в наше время носителей информации – магнитных дисков. Как известно, информация на жестком диске хранится в секторах (обычно 512 байт) и само устройство может выполнять лишь команды считать/записать информацию в определенный сектор на диске. В отличие от этого файловая система позволяет пользователю оперировать с более удобным для него понятием - файл. Файловая система берет на себя организацию взаимодействия программ с файлами, расположенными на дисках. Для идентификации файлов используются имена. Современные файловые системы предоставляют пользователям возможность давать файлам достаточно длинные мнемонические названия.

Под каталогом в ФС понимается, с одной стороны, группа файлов, объединенных пользователем исходя из некоторых соображений, с другой стороны каталог - это файл, содержащий системную информацию о группе составляющих его файлов. Файловые системы обычно имеют иерархическую структуру, в которой уровни создаются за счет каталогов, содержащих информацию о файлах и каталогах более низкого уровня.

Рассмотрим более подробно структуру жесткого диска. Базовой единицей жесткого диска является раздел, создаваемый во время разметки жесткого диска. Каждый раздел содержит один том, обслуживаемый какой-либо файловой системой и имеющий таблицу оглавления файлов - корневой каталог. Некоторые операционные системы поддерживают создание томов, охватывающих несколько разделов. Жесткий диск может содержать до четырех основных разделов. Это ограничение связано с характером организации данных на жестких дисках IBM-совместимых компьютеров. Многие операционные системы позволяют создавать, так называемый, расширенный (extended) раздел, который по аналогии с разделами может разбиваться на несколько логических дисков.

В первом физическом секторе жесткого диска располагается головная запись загрузки и таблица разделов (табл. 1). Головная запись загрузки (master boot record, MBR) - первая часть данных на жестком диске. Она зарезервирована для программы начальной загрузки BIOS (ROM Bootstrap routine), которая при загрузке с жесткого диска считывает и загружает в память первый физический сектор на активном разделе диска, называемый загрузочным сектором (Boot Sector). Каждая запись в таблице разделов (partition table) содержит начальную позицию и размер раздела на жестком диске, а также информацию о том, первый сектор какого раздела содержит загрузочный сектор.

Размер (байт)	Описание
446	Загрузочная запись (MBR)
16	Запись 1 раздела
16	Запись 2 раздела
16	Запись 3 раздела
16	Запись 4 раздела
2	Сигнатура 055AAh

Табл. 1. Таблица деления диска

В широком смысле понятие "файловая система" включает:

- совокупность всех файлов на диске,
- наборы служебных структур данных, используемых для управления файлами, такие как, например, каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске,
- комплекс системных программных средств, реализующих управление файлами, в частности операции по созданию, уничтожению, чтению, записи, именованию файлов, установке атрибутов и уровней доступа, поиску и т.д.

Различие между файловыми системами заключается, в основном, в способах распределения пространства между файлами на диске и организации на диске служебных областей.

Современные операционные системы стремятся обеспечить пользователя возможностью работать одновременно с несколькими файловыми системами. В этом случае ФС рассматривается как часть подсистемы ввода-вывода. В большинстве операционных систем (Windows 98, 2000, XP, OS/2) реализуется механизм переключения файловых систем (File System Switch, FSS), позволяющий поддерживать различные типы ФС. В соответствии с этим подходом информация о файловых системах и файлах разбивается на две части – зависимую от ФС и не зависимую. FSS обеспечивает интерфейс между ядром и файловой системой, транслируя запросы ядра в операции, зависящие от типа файловой системы. При этом ядро имеет представление только о независимой части ФС.

Файловая система представляет многоуровневую структуру, на верхнем уровне которой располагается так называемый переключатель файловых систем (в Windows, такой переключатель называется устанавливаемым диспетчером файловой системы - installable filesystem manager, IFS). Он обеспечивает интерфейс между приложением и конкретной файловой системой, к которой обращается приложение. Переключатель файловых систем преобразует запросы к файлам в формат, воспринимаемый следующим уровнем - уровнем драйверов файловых систем. Для выполнения своих функций драйверы файловых систем обращаются к драйверам конкретных устройств хранения информации.

Клиент-серверные приложения предъявляют повышенные требования к производительности файловых систем. Современные файловые системы должны обеспечивать эффективный доступ к файлам, поддержку носителей данных достаточно большого объема, защиту от несанкционированного доступа к данным и сохранение целостности данных. Под целостностью данных подразумевается способность ФС обеспечивать отсутствие ошибок и нарушений согласованности в данных, а также восстанавливать поврежденные данные.

FAT

Файловая система FAT (File Allocation Table) была разработана Биллом Гейтсом и Марком МакДональдом в 1977 году и первоначально использовалась в операционной системе 86-DOS. Чтобы добиться переносимости программ из операционной системы CP/M в 86-DOS, в ней были сохранены ранее принятые ограничения на имена файлов. В дальнейшем 86-DOS была приобретена Microsoft и стала основой для ОС MS-DOS 1.0, выпущенной в августе 1981 года. FAT была предназначена для работы с гибкими дисками размером менее 1 Мбайта, и вначале не предусматривала поддержки жестких дисков. В настоящее время FAT поддерживает файлы и разделы размеров до 2 Гбайт.

В FAT применяются следующие соглашения по именам файлов:

- имя должно начинаться с буквы или цифры и может содержать любой символ ASCII, за исключением пробела и символов "/\[]:;|=,^*?
- Длина имени не превышает 8 символов, за ним следует точка и необязательное расширение длиной до 3 символов.

- регистр символов в именах файлов не различается и не сохраняется.

Структура раздела FAT изображена на рисунке 2. В блоке параметров BIOS содержится необходимая BIOS информация о физических характеристиках жесткого диска. Файловая система FAT не может контролировать отдельно каждый сектор, поэтому она объединяет смежные сектора в кластеры (clusters). Таким образом, уменьшается общее количество единиц хранения, за которыми должна следить файловая система. Размер кластера в FAT является степенью двойки и определяется размером тома при форматировании диска (табл. 2). Кластер представляет собой минимальное пространство, которое может занимать файл. Это приводит к тому, что часть пространства диска расходуется впустую. В состав операционной системы входят различные утилиты (DoubleSpace, DriveSpace), предназначенные для уплотнения данных на диске.

Рис. 2

Свое название FAT получила от одноименной таблицы размещения файлов. В таблице размещения файлов хранится информация о кластерах логического диска. Каждому кластеру в FAT соответствует отдельная запись, которая показывает, свободен ли он, занят ли данными файла, или помечен как сбойный (испорченный). Если кластер занят под файл, то в соответствующей записи в таблице размещения файлов указывается адрес кластера, содержащего следующую часть файла. Из-за этого FAT называют файловой системой со связанными списками. Оригинальная версия FAT, разработанная для DOS 1.00, использовала 12-битную таблицу размещения файлов и поддерживала разделы объемом до 16 Мб (в DOS можно создать не более двух разделов FAT). Для поддержки жестких дисков размером более 32 Мб разрядность FAT была увеличена до 16 бит, а размер кластера - до 64 секторов (32 Кб). Так как каждому кластеру может быть присвоен уникальный 16-разрядный номер, то FAT поддерживает максимально 2^{16}, или 65536 кластеров на одном томе.

Размер раздела	Размер кластера	Тип FAT
< 16 Мб	4 Кб	FAT12
16 Мб – 127 Мб	2 Кб	FAT16
128 Мб – 255 Мб	4 Кб	FAT16
256 Мб – 511 Мб	8 Кб	FAT16
512 Мб – 1023 Мб	16 Кб	FAT16
1 Гб – 2 Гб	32 Кб	FAT16

Табл. 2

Поскольку загрузочная запись слишком мала для хранения алгоритма поиска системных файлов на диске, то системные файлы должны находиться в определенном месте, чтобы загрузочная запись могла их найти. Фиксированное положение системных файлов в начале

области данных накладывает жесткое ограничение на размеры корневого каталога и таблицы размещения файлов. Вследствие этого общее число файлов и подкаталогов в корневом каталоге на диске FAT ограничено 512.

Каждому файлу и подкаталогу в FAT соответствует 32-байтный элемент каталога (directory entry), содержащий имя файла, его атрибуты (архивный, скрытый, системный и “только для чтения”), дату и время создания (или внесения в него последних изменений), а также прочую информацию (табл. 3).

Содержание	Размер (байт)
Имя файла	8
Расширение	3
Байт атрибутов	1
Зарезервировано	10
Время	2
Дата	2
Номер начального кластера с данными	2
Размер файла	4

Табл. 3. Элемент каталога

Файловая система FAT всегда заполняет свободное место на диске последовательно от начала к концу. При создании нового файла или увеличении уже существующего она ищет самый первый свободный кластер в таблице размещения файлов. Если в процессе работы одни файлы были удалены, а другие изменились в размере, то появляющиеся в результате пустые кластеры будут рассеяны по диску. Если кластеры, содержащие данные файла, расположены не подряд, то файл оказывается фрагментированным. Сильно фрагментированные файлы значительно снижают эффективность работы, так как головки чтения/записи при поиске очередной записи файла должны будут перемещаться от одной области диска к другой. В состав операционных систем, поддерживающих FAT, обычно входят специальные утилиты дефрагментации диска, предназначенные повысить производительность файловых операций.

Еще один недостаток FAT заключается в том, что ее производительность сильно зависит от количества файлов, хранящихся в одном каталоге. При большом количестве файлов (около тысячи), выполнение операции считывания списка файлов в каталоге может занять несколько минут. Это обусловлено тем, что в FAT каталог имеет линейную неупорядоченную структуру, и имена файлов в каталогах идут в порядке их создания. В результате, чем больше в каталоге записей, тем медленнее работают программы, так как при поиске файла требуется просмотреть последовательно все записи в каталоге.

Поскольку FAT изначально проектировалась для однопользовательской операционной системы DOS, то она не предусматривает хранения такой информации, как сведения о владельце или полномочия доступа к файлу/каталогу.

FAT является наиболее распространенной файловой системой и ее в той или иной степени поддерживают большинство современных ОС. Благодаря своей универсальности FAT может применяться на томах, с которыми работают разные операционные системы.

Хотя нет никаких препятствий использовать при форматировании дискет любую другую файловую систему, большинство ОС для совместимости используют FAT. Отчасти это можно объяснить тем, что простая структура FAT требует меньше места для хранения служебных данных, чем остальные системы. Преимущества других файловых систем становятся заметны только при использовании их на носителях объемом более 100 Мб.

Надо отметить, что FAT - простая файловая система, не предотвращающая порчи файлов из-за ненормального завершения работы компьютера. В состав операционных систем, поддерживающих FAT, входят специальные утилиты проверяющие структуру и корректирующие несоответствия в файловой системе.

HPFS

Высокопроизводительная файловая система HPFS (High Performance File System) была представлена фирмой IBM в 1989 году вместе с операционной системой OS/2 1.20. Файловая система HPFS также поддерживалась ОС Windows NT до версии 3.51 включительно. По производительности эта ФС существенно опережает FAT. HPFS позволяет использовать жесткие диски объемом до 2 Терабайт (первоначально до 4 Гбайт). Кроме того, она поддерживает разделы диска размером до 512 Гб и позволяет использовать имена файлов длиной до 255 символов (на каждый символ при этом отводится 2 байта). В HPFS по сравнению с FAT уменьшено время доступа к файлам в больших каталогах.

HPFS распределяет пространство на диске не кластерами как в FAT, а физическими секторами по 512 байт, что не позволяет ее использовать на жестких дисках, имеющих другой размер сектора. Эти секторы принято называть блоками. Чтобы уменьшить фрагментацию диска, при распределении пространства под файл HPFS стремится, по возможности, размещать файлы в последовательных смежных секторах. Фрагмент файла, располагающийся в смежных секторах, называется экстентом.

Для нумерации единиц распределения пространства диска HPFS использует 32 разряда, что дает 2^{32}, или более 4 миллиардов номеров. Однако HPFS использует числа со знаком, что сокращает число возможных номеров блоков до 2 миллиардов. Помимо стандартных атрибутов файла, HPFS поддерживает расширенные атрибуты файла (Extended Attributes, EA), которые могут содержать до 64 Кб различных дополнительных сведений о файле.

Диск HPFS имеет следующие три базовые структуры (рис. 3): загрузочный блок (BootBlock), дополнительный блок (SuperBlock) и резервный блок (SpareBlock).

Загрузочный блок	Дополнительный блок	Резервный блок	Группа 1	Битовая карта группы 1	Битовая карта группы 2	Группа 2	Группа 3	Битовая карта группы 3	Битовая карта группы 4	Группа 4

Рис. 3. Дисковый раздел HPFS

Загрузочный блок в HPFS аналогичен загрузочному блоку в FAT. Он располагается в секторах с 0 по 15 и занимает на диске 8 Кб. Системные файлы, также как и в FAT, располагаются в корневом каталоге, но при этом физически могут находиться в любом месте на диске.

В 16 секторе размещается дополнительный блок, содержащий указатель на список блоков битовых карт (bitmap block list). В этом списке перечислены все блоки на диске, в которых расположены битовые карты, используемые для обнаружения свободных секторов. Также в дополнительном блоке хранится указатель на список дефектных блоков (bad block list), указатель на группу каталогов (directory band), указатель на файловый узел корневого каталога и дата последней проверки диска. Файловый узел (fnode) – это структура диска HPFS, которая содержит информацию о расположении файла и о его расширенных атрибутах.

В следующем секторе находится резервный блок, содержащий карту аварийного замещения (hotfix map), указатель на список свободных запасных блоков (directory emergency free block list) и ряд системных флагов. Резервный блок обеспечивает высокую отказоустойчивость HPFS и позволяет восстанавливать поврежденные данные на диске.

Остальное пространство диска разделено на группы (band) хранения данных. Каждая группа занимает 8 Мб и имеет свою собственную битовую карту свободного пространства, которая похожа на таблицу размещения файлов FAT. Каждому сектору группы соответствует один бит к ее битовой карте, показывающий занят ли соответствующий сектор. Битовые карты двух групп располагаются на диске рядом, также как располагаются и сами группы. Это дает возможность непрерывно разместить на жестком диске файл размером до 16 Мб.

Одна из групп данных размером 8 Мб, расположенная в середине жесткого диска и называемая группой каталогов, хранит информацию о каталогах диска. В ней наряду с остальными каталогами располагается и корневой каталог. Расположение группы каталогов в центре диска значительно сокращает время позиционирования головок чтения/записи.

В отличие от линейной структуры FAT, структура каталога в HPFS представляет собой сбалансированное дерево (так называемое B-дерево) с записями, расположенными в алфавитном порядке. Сбалансированное дерево состоит из корневого (root block) и оконечных блоков (leaf block). Блоки занимают 4 последовательных сектора и в среднем могут содержать 40 записей. Каждая запись корневого блока указывает на один из оконечных блоков (если только в каталоге не меньше 40 файлов); в свою очередь, каждая запись в оконечном блоке указывает на файловый узел файла или на оконечный блок следующего уровня. Таким образом, двухуровневая структура может содержать 40 оконечных блоков по 40 записей в каждом и описывать до 1600 файлов. При поиске файловая система HPFS просматривает только необходимые ветви дерева.

Файловый узел имеет размер 512 байт и всегда по возможности располагается непосредственно перед первым блоком своего файла. Каждый файл и каталог диска HPFS имеет свой файловый узел. Информация, хранящаяся в файловом узле, включает в себя расширенные атрибуты файла, если они достаточно малы, чтобы поместится в один сектор диска, и сокращенное имя файла в формате 8.3. Если расширенные атрибуты не помещаются в файловый узел, то в него записывается указатель на них. Положение файла на диске описывается в файловом узле двумя 32-битными числами. Первое из чисел представляет собой указатель на первый блок файла, а второе - длину экстента. Если же файл фрагментирован, то его размещение описывается дополнительными парами 32-битных чисел. В файловом узле можно хранить информацию максимум о 8 экстентах файла. Если файл имеет больше число экстентов, то в его файловый узел записывается указатель на блок размещения (allocation block), который может содержать до 40 указателей на экстенты или на другие блоки размещения. Таким образом, двухуровневая структура блоков размещения может хранить информацию о 480 (12*40) секторах, что теоретически, позволяет работать с файлами размером до 7.68 Гб (12*40*16 Мб).

VFAT

Файловая система VFAT (Virtual FAT), реализованная в Windows NT 3.5, Windows 95 (DOS 7.0), - это файловая система FAT, включающая поддержку длинных имен файлов (Long File Name, LFN) в кодировке UNICODE (каждый символ имени кодируется 2 байтами). VFAT использует ту же самую схему распределения дискового пространства, что и файловая система FAT, поэтому размер кластера определяется величиной раздела.

В VFAT ослаблены ограничения, устанавливаемые соглашениями по именам файлов FAT:

- имя может быть длиной до 255 символов.
- в имя можно включать несколько пробелов и точек, однако, текст после последней точки рассматривается как расширение.

- регистр символов в именах не различается, но сохраняется.

Основной задачей при разработке VFAT была необходимость корректной работы старых программ, не поддерживающих длинные имена файлов. Как правило, прикладные программы для доступа к файлам используют функции ОС. Если у элемента каталога установить “нереальную” комбинацию битов атрибутов: “только для чтения”, “скрытый”, “системный”, “метка тома” – то любые файловые функции старых версий DOS и Windows не заметят такого элемента каталога. В итоге для каждого файла и подкаталога в VFAT хранится два имени: длинное и короткое в формате 8.3 для совместимости со старыми программами. Длинные имена (LFN) хранятся в специальных записях каталога, байт атрибутов, у которых равен 0Fh. Для любого файла или подкаталога непосредственно перед единственной записью каталога с его именем в формате 8.3 находится группа из одной или нескольких записей, представляющих длинное имя. Каждая такая запись содержит часть длинного имени файла не более 13 символов, из всех таких записей ОС составляет полное имя файла. Поскольку одно длинное имя файла может занимать до 21 записи, а корневой каталог FAT ограничен 512 записями, желательно ограничить использование длинных имен в корневом каталоге.

Содержание	Размер (байт)
Порядок следования	1
Первые пять символов LFN	10
Байт атрибутов (0Fh)	1
Указатель типа (всегда 0)	1
Контрольная сумма части имени	1
Следующие шесть символов LFN	12
Номер начального кластера (всегда 0)	2
Следующие два символа LFN	4

Табл. 4. Элемент каталога для длинного имени

Короткое имя генерируется файловой системой автоматически в формате 8.3. Для создания коротких имен (псевдонимов) файлов используется следующий алгоритм:

1. Из длинного имени удалить все символы не допустимые в именах FAT. Удалить точки в конце и начале имени. После этого удалить все точки, находящиеся внутри имени кроме последней.

2. Обрезать строку, расположенную перед точкой, до 6 символов и добавить в ее конец "~1". Обрезать строку за точкой до 3 символов.
3. Полученные буквы преобразовать в прописные. Если сгенерированное имя совпадает с уже существующим, то увеличить число в строке "~1".

Данный алгоритм зависит от версии операционной системы и в будущих версиях может меняться.

Редактирование файлов программами, не поддерживающими длинные имена файлов, может приводить к потере длинных имен. Windows обнаруживает подобные элементы каталога, так как их контрольная сумма не соответствует больше тому, что записано в последующей записи каталога в формате 8.3. Однако такие записи не удаляются системой автоматически, они занимают дисковое пространство, до тех пор, пока вы не запустите программу ScanDisk, входящую в состав операционной системы. Большинство старых дисковых утилит воспримут записи, соответствующие длинным именам, как ошибки логической структуры диска. Попытки использовать данные утилиты, в лучшем случае приведет к потере длинных имен, а в худшем - к потере информации на диске.

FAT 32

Система FAT32 - более новая файловая система на основе формата FAT, она поддерживается Windows 95 OSR2, Windows 98 и Windows Millennium Edition. FAT32 использует 32-разрядные идентификаторы кластеров, но при этом резервирует старшие 4 бита, так что эффективный размер идентификатора кластера составляет 28 бит. Поскольку максимальный размер кластеров FAT32 равен 32 Кбайт, теоретически FAT32 может работать с 8-терабайтными томами. Windows 2000 ограничивает размер новых томов FAT32 до 32 Гбайт, хотя поддерживает существующие тома FAT32 большего размера (созданные в других операционных системах). Большее число кластеров, поддерживаемое FAT32, позволяет ей управлять дисками более эффективно, чем FAT 16. FAT32 может использовать 512-байтовые кластеры для томов размером до 128 Мбайт.

Файловая система FAT 32 в Windows 98 используется в качестве основной. С этой операционной системой поставляется специальная программа преобразования диска из FAT 16 в FAT 32. Windows 2000 и Windows XP тоже могут использовать файловую систему FAT, и поэтому можно загрузить компьютер с DOS-диска и иметь полный доступ ко всем файлам. Однако некоторые из самых прогрессивных возможностей Windows 2000 и Windows XP обеспечиваются ее собственной файловой системой NTFS (NT File System). NTFS позволяет создавать на диске разделы объемом до 2 Тбайт (как и FAT 32), но, кроме этого, в нее встроены функции сжатия файлов, безопасности и аудита, необходимые при работе в сетевой среде. А в Windows 2000, как и в Windows XP реализуется поддержка файловой системы FAT 32. Данные этих операционных систем можно хранить на диске FAT, но по желанию пользователя диск может быть конвертирован в формат NTFS.

Для этого можно воспользоваться утилитой Convert.exe, поставляемой вместе с операционной системой. Преобразованный к системе NTFS раздел диска становится недоступным для других операционных систем. Чтобы вернуться в DOS, Windows 95, Windows 98 или Me, нужно удалить раздел NTFS, а вместо него создать раздел FAT. Windows 2000, как и в Windows XP можно устанавливать на диск с файловой системой FAT 32 и NTFS.

Возможности файловых систем FAT32 гораздо шире возможностей FAT16. Самая важная ее особенность в том, что она поддерживает диски объемом до 2 047 Гбайт и работает с

кластерами меньшего размера, благодаря чему существенно сокращает объемы неиспользуемого дискового пространства. Например, жесткий диск объемом 2 Гбайт в FAT16 использует кластеры размером по 32 Кбайт, а в FAT32 - кластеры размером по 4 Кбайт. Чтобы по возможности сохранить совместимость с существующими программами, сетями и драйверами устройств, FAT32 реализована с минимальными изменениями в архитектуре, API-интерфейсах, структурах внутренних данных и дисковом формате. Но, так как размер элементов таблицы FAT32 теперь составляет четыре байта, многие внутренние и дисковые структуры данных, а также API-интерфейсы пришлось пересмотреть или расширить. Отдельные API на FAT32-дисках блокируются, чтобы унаследованные дисковые утилиты не повредили содержимое FAT32-дисков. На большинстве программ эти изменения никак не скажутся. Существующие инструментальные средства и драйверы будут работать и на FAT32-дисках. Однако драйверы блочных устройств MS-DOS (например, Aspidisk.sys) и дисковые утилиты нуждаются в модификации для поддержки FAT32. Все дисковые утилиты, поставляемые Microsoft (Format, Fdisk, Defrag, а также ScanDisk для реального и защищенного режимов), переработаны и полностью поддерживают FAT32. Кроме того, Microsoft помогает ведущим поставщикам дисковых утилит и драйверов устройств в модификации их продуктов для поддержки FAT32. FAT32 эффективнее FAT16 при работе с дисками большего объема и не требует их разбиения на разделы по 2 Гбайт. Windows 98 обязательно поддерживает FAT16, так как именно эта файловая система совместима с другими операционными системами, в том числе сторонних компании. В MS-DOS реального режима и в безопасном режиме Windows 98, файловая система FAT32 работает значительно медленнее, чем FAT16. Поэтому, при запуске программ в режиме MS DOS желательно включить в файл Autoexec.bat или PIF-файл команду для загрузки Smartdrv.exe, что ускорит дисковые операции. Некоторые устаревшие программы, рассчитанные на спецификацию FAT16, могут сообщать неправильную информацию об объеме свободного или общего дискового пространства, если он больше 2 Гбайт. Windows 98 предоставляет новые API-интерфейсы для MS-DOS и Win32, которые позволяют корректно определять эти показатели. В табл. 1 приведены сравнительные характеристики FAT16 и FAT32.

Таблица 1. Сравнение файловых систем FAT16 и FAT32

FAT16	FAT32
Реализована и используется большинством операционных систем (MS-DOS, Windows 95/98/Me, Windows 2000 и Windows XP, OS/2, UNIX).	На данный момент поддерживается только в Windows 95/98/Me, Windows 2000 и Windows XP.
Очень эффективна для логических дисков размером менее 256 Мбайт.	Не работает с дисками объемом менее 512 Мбайт.
Поддерживает сжатие дисков, например по алгоритму DriveSpace.	Не поддерживает сжатие дисков.
Обрабатывает максимум 65 525 кластеров, размер которых зависит от объема логического диска. Так как максимальный размер кластеров равен 32 Кбайт, FAT16 может работать с логическими дисками объемом не более 2 Гбайт.	Способна работать с логическими дисками объемом до 2 047 Гбайт при максимальном размере кластеров в 32 Кбайт.
Чем больше размер логического диска, тем меньше эффективность хранения файлов в FAT'16-системе, так как увеличивается и размер кластеров. Пространство для файлов выделяется кластерами, и поэтому при максимальном объеме логического диска файл	На логических дисках объемом менее 8 Гбайт размер кластеров составляет 4 Кбайт.

размером 10 Кбайт потребует 32 Кбайт, а 22 Кбайт дискового пространства пропадет впустую.	

Максимально возможная длина файла в FAT32 равна 4 Гбайт за вычетом 2 байтов. Win32-приложения могут открывать файлы такой длины без специальной обработки. Остальные приложения должны использовать прерывание Int 21h, функцию 716C (FAT32) с флагом открытия, равным EXTEND-SIZE (1000h).

В файловой системе FAT32 на каждый кластер в таблице размещения файлов отводится по 4 байта, тогда как в FAT16 - по 2, а в FAT12 - по 1,5.

Старшие 4 бита 32-разрядного элемента таблицы FAT32 зарезервированы и не участвуют в формировании номера кластера. Программы, напрямую считывающие FAT32-таблицу, должны маскировать эти биты и предохранять их от изменения при записи новых значений.

Итак, FAT32 обладает следующими преимуществами в сравнении с прежними реализациями файловой системы FAT:

- поддерживает диски объемом до 2 Тбайт;
- эффективнее организует дисковое пространство. FAT32 использует кластеры меньшего размера (4 Кбайт для дисков объемом до 8 Гбайт), что позволяет сэкономить до 10-15% пространства на больших дисках по сравнению с FAT;
- корневой каталог FAT 32, как и все остальные каталоги, теперь не ограничен, он состоит из цепочки кластеров и может быть расположен в любом месте диска;
- имеет более высокую надежность: FAT32 способна перемещать корневой каталог и работать с резервной копией FAT, кроме того, загрузочная запись на FAT32-дисках расширена и теперь включает резервную копию критически важных структур данных, а это означает, что FAT32-диски менее чувствительны к возникновению отдельных сбойных участков, чем существующие FAT-тома;
- программы загружаются на 50% быстрее.

Таблица 2. Сравнение размеров кластеров

Объем диска	Размер кластеров в FAT16, Кбайт	Размер кластеров в FAT32, Кбайт
256 Мбайт-511 Мбайт	8	Не поддерживается
512 Мбайт -1023 Мбайт	16	4
1024 Мбайт - 2 Гбайт	32	4
2 Гбайт - 8 Гбайт	Не поддерживается	4
8 Гбайт-16 Гбайт	Не поддерживается	8
16 Гбайт-32 Гбайт	Не поддерживается	16
Более 32 Гбайт	Не поддерживается	32

Усовершенствованная утилита дефрагментации дисков оптимизирует размещение файлов приложения, загружаемых в момент его запуска. Возможно преобразование диска в FAT32 с помощью утилиты Drive Converter (FAT32), но после этого рекомендуется запустить утилиту Disk Defragmenter, - иначе компьютер будет работать с диском медленнее, чем раньше. Благодаря этому на больших дисках удается высвободить десятки и даже сотни мегабайтов, а в сочетании с усовершенствованной утилитой дефрагментации дисков FAT32 значительно сокращает время загрузки приложений. Процедура преобразования файловой системы на

жестком диске в FAT32 с помощью Drive Converter (FAT32) достаточно проста. Для этого последовательно необходимо открыть меню Start (Пуск), подменю Programs (Программы), Accessories (Стандартные), System Tools (Служебные) и выбрать команду Drive Converter (FAT32) (Преобразование диска в FAT32). Преобразование может повлиять на функции спящего режима (hibernate features) (сохранения состояния компьютера на диск), предусмотренные во многих компьютерах. Системы, в которых режим сна реализован через APM BIOS или ACPI (Advanced Configuration and Power Interface) S4/BIOS, должны поддерживать FAT32, - только тогда они будут корректно работать в Windows 98 и Me.

Большинство изготовителей BIOS включают в нее средства защиты от вирусов, отслеживающие изменения в главной загрузочной записи MBR (Master Boot Record). Кроме того, устаревшие антивирусные утилиты, устанавливаемые как резидентные программы или драйверы реального режима, могут обнаруживать изменение MBR при загрузке MS-DOS. Так как преобразование в FAT32 приводит к неизбежной модификации MBR, некоторые средства проверки на вирусы могут ошибочно счесть это признаком инфицирования системы.
Лучше всего удалить антивирусное программное обеспечение и отключить встроенные в BIOS средства защиты от вирусов перед преобразованием диска в FAT32. Потом можно вновь установить антивирусную утилиту и активизировать встроенные в BIOS средства защиты от вирусов.

Главная загрузочная запись (MBR)

Форматирование жестких дисков выполняется в три этапа:

- низкоуровневое форматирование (физическая разметка диска на цилиндры, дорожки, секторы);
- разбиение диска на разделы (логические устройства):
- высокоуровневое (логическое) форматирование каждого раздела.

На этапе низкоуровневого форматирования процессор, выполняя программу форматирования, поочередно передает в контроллер жесткого диска сначала команду "Поиск" для установки головок накопителя на нужный цилиндр, а затем посылает команду "Форматировать дорожку". Выполняя команду "Форматировать дорожку" контроллер жесткого диска, получив из накопителя импульс "Индекс" (начало дорожки), производит запись служебного формата дорожки, который разбивает ее на секторы. Каждый сектор содержит в себе блок данных (512 байт), обрамленный служебным форматом сектора (содержание к размер служебного формата определяется конкретной фирмой-разработчиком данного устройства).
Служебный формат дорожки и секторов необходим контроллеру жесткого диска при выполнении команд. Читая и расшифровывая поля служебного формата, контроллер находит на диске нужный цилиндр, поверхность, сектор и блок данных внутри сектора. На следующих этапах форматирования в блоки данных ряда секторов записывается системная информация, которая обеспечивает организацию разделов на диске, автоматическую загрузку операционной системы и поддержку файловой системы на диске.

На этапе разбиения диска на разделы в блоке данных первого физического сектора диска (0 цилиндр, 0 поверхность, 1 сектор) с адреса 1BEh формируется таблица разделов (Partition table), состоящая из 4-х шестнадцатибайтных строк. Обычно системную информацию, записанную в блок данных этого сектора в процессе форматирования, называют Master Boot Record (MBR).

С самого начала блока данных этого сектора располагается программа (IPL 1). Переход на программу IPL 1 процессор осуществляет после успешного завершения POST и программы "Начального загрузчика", выполняя которую процессор загружает с диска в память MBR, и передает управление на начало MBR (на программу IPL 1), продолжая действия ведущие к загрузке операционной системы. Программа IPL 1 (загрузчик), находящаяся в MBR просматривает строки таблицы разделов в поисках активного раздела с которого возможна загрузка операционной системы. Если в таблице разделов нет активного раздела, выдается сообщение об ошибке. Если хотя бы один раздел содержит неправильную метку, либо несколько разделов помечены как активные, выдается сообщение об ошибке Invalid partition table, и процесс загрузки останавливается. Если активный раздел обнаружен, то анализируется загрузочный сектор этого раздела. Если найден только один активный раздел, то содержимое блока данных его загрузочного сектора (BOOT) читается в память по адресу 0000:7C00 и управление передается по этому адресу, если загрузочный сектор активного раздела не читается за пять попыток, выдается сообщение об ошибке: Error loading operating system и система останавливается; проверяется сигнатура считанного загрузочного сектора активного раздела и если последних два его байта не соответствуют сигнатуре 55AAh, выдается сообщение об ошибке: Missing operating system и система останавливается). Процессор читает по адресу 0000:7C00 команду JMP, выполняя ее, передает управление на начало программы IPL 2, которая осуществляет проверку, действительно ли раздел активный: IPL 2 проверяет имена и расширения двух файлов в корневом каталоге - это должны быть файлы IO.SYS и MSDOS.SYS (NTLDR для Windows XP), загружает их и. т. д.

Система Windows 9x/Me во многом основана на тех же концепциях, что и DOS, но в ней эти концепции получили дальнейшее логическое развитие. Те же два системных файла IO.SYS и MSDOS.SYS, но теперь вся системная программа находится в IO.SYS , а второй файл MSDOS.SYS содержит ASCII-текст с установками, управляющими поведением системы при загрузке. Эквиваленты программ Himem.sys. Ifshlp.sys и Setver.exe автоматически загружаются программой IO.SYS при запуске системы. Как и прежде, для загрузки в память драйверов и резидентных программ можно использовать файлы Config.sys и Autoexec.bat, но загрузку 32-разрядных драйверов устройств, которые разработаны специально для Windows 9x, теперь обеспечивают записи в системном реестре. Когда вся предварительная работа выполнена, запускается файл Win.com, и Windows 9x/Me загружается и предоставляет свои возможности через графическое меню.

Системный реестр является базой данных, в которой Windows 9x/Me хранит информацию обо всех настройках, конфигурационных установках и параметрах, необходимых для работы ее собственных модулей и отдельных приложений. Системный реестр как бы выполняет функции Config.sys, Autoexec.bat и ini-файлов Windows 3.1 вместе взятых. На диске компьютера реестр хранится в виде двух отдельных файлов: System.dat и User.dat. В первом из них содержатся всевозможные аппаратные установки, а во втором - данные о работающих в системе пользователях и используемых ими конфигурациях. Каждый пользователь может иметь свой файл User.dat, т.е. собственную рабочую среду, которую он настраивает по своему вкусу и потребностям. Системный реестр можно импортировать, экспортировать, а также создавать его резервные копии и, используя их, восстанавливать сохраненные данные - одним словом, это довольно мощный механизм управления системными параметрами и их защиты от потерь и повреждений.

Таблица 3. Компоненты MBR

Область	Описание
Программа IPL 1 (программа загрузчика	Код программы Сообщения об ошибках (Error Messages):

занимает зону от адреса 00h до 1BEh)	• Invalid Partition Table (неправильная таблица разделов). • Error loading operating system (ошибка при загрузке операционной системы) • Missing operating system (операционная система отсутствует).
Таблица разделения физического диска на логические устройства (Partition Tables) (4 строки по 16 байт = 64 байта) занимает зону с адреса 1BEh до 1FDh	1 строка (16 байт): • Флаг загрузки (80h - активный / 00h -обычный раздел) - 1 байт • Начальный физический сектор раздела (головка, сектор и цилиндр) - 3 байта • Тип раздела -1 байт • Конечный физический сектор раздела (головка, сектор и цилиндр) - 3 байта • Число секторов предшествующих разделу - 4 байта • Общее количество секторов в данном разделе - 4 байта
2 последних байта в блоке данных сектора с адреса 1FE по 1FF- концевая сигнатура (Ending Signature)	55AA - отмечает конец MBR. Проверяется программой начального загрузчика

Область MBR, изменившаяся в FAT32 - это Partition Table. Она, как и прежде, состоит из четырех 16-байтных записей. Каждая запись определяет раздел. В FAT32 введено 2 новых типа разделов DOS32 (0B) и DOS32X (OC).

DOS32 (0B).

Определяет основной раздел FAT32 размером до 2 047 Гбайт. Используется, когда для доступа к основному разделу не требуется механизм логической блочной адресации (LBA). LBA базируется на расширениях прерывания Int 13h.
Расширенный дисковый сервис BIOS Enhanced Disk Drive Services (EDD), продвигаемый фирмой Phoenix Technologies LTD, реализуется многими разработчиками BIOS и устройств массовой памяти. Он позволяет работать с устройствами, имеющими объем до 264 секторов. Сервис оперирует линейным логическим адресом сектора (LBA). Вместо традиционных таблиц параметров дисков в нем используются новые, дающие исчерпывающую информацию об устройствах, их физической организации и интерфейсе. Устройства могут иметь сменные носители и сами быть съемными в процессе работы компьютера (например, подключенные к шине USB или IEEE 1394), так что понятие "сменяемость носителя" несколько размывается. Такие устройства должны поддерживать механизм уведомления о смене носителя и программное блокирование смены носителя. Расширения BIOS Int 13h используют ОС Windows 95, Windows 98, Windows 2000 и Windows XP. Правда, использование этих возможностей ограничено лишь начальной загрузкой и процессом установки (FDISK, FORMAT), поскольку в регулярной работе применяются собственные 32-разрядные драйверы. Расширения BIOS Int 13h не используют все версии DOS, Windows 3.1, Windows NT, Novell NetWare, OS/2, Warp, Linux, UNIX.

В настоящее время определены три набора функций:

- доступ к фиксированным дискам - функции 4lh - 44h, 47h и 48h;
- блокировка и смена носителя - функции 41h, 45h, 46h, 48h и 49h;
- поддержка расширенных дисков - функции 41h и 48h;
- для эмуляции дисков на загружаемых CD-ROM имеются расширенные функции 4Ah - 4Dh.

Расширенный сервис, как и традиционный, вызывается программным прерыванием Int 13h с номерами функций свыше 3Fh (в регистре AH), с номерами устройств (в регистре DL) в диапазоне 80h-FFh. Основные параметры вызова - начальный адрес блока, число секторов для передачи и адрес буфера - передаются через адресный пакет. Формат пакета в сравнении с передачей параметров традиционного сервиса через регистры процессора имеет более широкие возможности. Поскольку расширение BIOS может и отсутствовать, имеется функция проверки его наличия (номер 41h). Расширение может действовать избирательно (не для всех устройств), так что проверку надо производить для конкретного устройства, интересующего программу. Проверка дает номер версии расширения и карту поддерживаемых наборов функций. Функции расширенного чтения, записи, верификации и поиска (42h, 43h, 44h и 47h) по смыслу не отличаются от их аналогов из традиционного сервиса. Для работы со сменными носителями введены функции отпирания-запирания, извлечения и проверки факта смены, носителя (45h, 46h и 49h). Сильно отличается от традиционного сервиса функция получения параметров устройства (48h). Она возвращает в ОЗУ буфер с набором параметров и детальным описанием устройства, позволяющим ОС и приложениям работать с ним, минуя BIOS. Функция установка аппаратной конфигурации (4Eh) позволяет управлять режимом передачи (PIO, DMA), а также предварительной выборкой (поиском).

DOS32X (0C). Определяет основной раздел

FAT32 размером до 2 047 Гбайт. Используется, когда для доступа к любой части основного или дополнительного раздела требуется механизм LBA (адрес превышает максимальное значение, возможное в комбинации из 1 024 цилиндров, 63 секторов на дорожку и 16 головок). Этот тип разделов недоступен из MS-DOS версий 6.x или более ранних. Некоторые системные коды (типы разделов) для разделов и логических дисков DOS/Windows 9x/Me приведены в табл. 4.

Таблица 4. Коды и типы разделов жесткого диска

Код	Раздел	ОС, с которой введен	Файловая система	Объем
01	DOS FAT12	MS-DOS 2.0	FAT12	до 16 Мбайт
04	DOS FAT16	MS-DOS 3.0	FAT16	до 32 Мбайт
05	DOS Extended	MS-DOS 3.3	FAT16	до 2 Гбайт
06	DOS FAT16 (Big DOS)	MS-DOS 4.0	FAT16	до 2 Гбайт
07	OS/2 HPFS	Windows NT NTFS	HPFS	512 Мбайт - 2 Тбайт
OB	Win95 FAT32	Windows 95 OSR2	FAT32	512 Мбайт - 2 Тбайт
OC	Win95 FAT32 (LBA)	Windows 95 OSR2	FAT32	512 Мбайт - 2 Тбайт
OE	Win95 FAT16 (LBA)	Windows 95 OSR2	FAT16	32 Мбайт - 2 Гбайт
OF	Win95 Extended (LBA)	Windows 95 OSR2	FAT32	512 Мбайт - 2 Тбайт

Разделы с кодами (01, 04, 06, 0B, OC, OE) являются первичными разделами DOS/Windows. Утилита FDISK из MS-DOS и Windows 9x/Me позволяет создавать не более одного первичного раздела, хотя в принципе их может быть и больше. Первичный раздел содержит один логический диск. В стандартном случае, когда на диске имеется один первичный

раздел, для первого винчестера на нем будет диск C:, для второго - D: и т. д. В операционных системах MS-DOS и Windows 9x/Me на одном диске не должно быть более одного первичного раздела, а также первичный раздел должен быть первым в таблице разделов. Другие операционные системы (ОС), например Linux, не ограничивают жестко количество и расположение разделов. Разные коды первичных разделов указывают на различную разрядность FAT, новые типы вводились по мере роста размеров винчестера. С Windows 95 OSR2 появились новые типы разделов для FAT32 и FAT16 (0Ch, 0=0Eh), специально для дисков, поддерживающих адресацию LBA. Заметим, что в каждом описателе разделов задаются как трехмерные границы раздела [начальные и конечные номера цилиндра, головки и сектора), так и линейные (номер начального сектора и их количество), но долгое время использовали только трехмерные описатели. Среди разделов DOS/Windows 9x/Me активным может быть только первичный раздел. Расширенный раздел (код 05 или 0F) служит для организации произвольного количества логических дисков. Первый сектор расширенного раздела аналогичен MBR (но загрузчик отсутствует) и содержит расширенную таблицу разделов EPR (Extended Partition Record) той же структуры, но первая строка таблицы задает, вторичный (secondary) раздел, отведенный под очередной логический диск; в нем указывается код раздела с файловой системой (для DOS/ Windows это FAT с кодами 04h, 06h, 0Bh, 0Ch или 0Eh, для других ОС -свои). В этом описателе, как обычно, задаются координаты начала и конца раздела с логическим диском (трехмерные и линейные). Если этот логический диск занимает не весь объем расширенного раздела, то второй описатель тоже имеет код 05 или 0F и указывает на положение сектора со следующей расширенной таблицей разделов. Остальные описатели не используются (их коды нулевые). Если свободного места в разделе уже нет, то и второй описатель не используется. В следующей расширенной таблице разделов действуют те же правила. Эта цепочка заканчивается на расширенной таблице, у которой во втором описателе стоит нулевой код раздела. Второй описатель в расширенных таблицах может указывать только на положение следующей расширенной таблицы. Часть пространства расширенного раздела может оставаться не распределенной, в дальнейшем она может быть использована под логические диски. Цепочка расширенных таблиц разделов должна быть непрерывной, неветвящейся (используются только два описателя, и только второй может указывать на следующую таблицу) и не зацикленной (второй описатель не должен ссылаться на ту же таблицу или предыдущую в цепочке). Несоблюдение первых двух условий ведет к потере логических дисков (их система не найдет). Несоблюдение последнего условия может привести к зависанию ОС при загрузке (она зациклится на бесконечном определении повторяющихся логических дисков). Код (05 или 0F) расширенного раздела не несет никакой информации о файловой системе, и данный тип раздела используется как указатель на расширенную таблицу рядом ОС, в том числе и отличных от DOS/Windows. Координаты расширенных таблиц разделов обычно имеют вид N, 0, 1.

По расположению на физическом диске расширенные разделы являются вложенными друг в друга: все они располагаются в области, описанной в главной таблице разделов как расширенный раздел. В главной таблице может быть описан лишь один расширенный раздел.

Если расширенные разделы имеют код 0Fh, то линейные адреса всех элементов таблиц будут указываться относительно начала физического диска (так поступает новая версия утилиты FDISK, и это более естественно, поскольку при этом описатель LBA является эквивалентом описателя CHS).

Каждый логический диск из расширенного раздела имеет ту же структуру, что и первичный раздел. Он также начинается с загрузочного сектора (только загрузчик никогда не исполняется), в котором имеется описание структуры логического диска. Координаты

загрузочных секторов логических дисков обычно имеют вид N, 1, 1. Операционная система назначает логическим дискам расширенных разделов имена (буквы), остающиеся

после дисков первичных разделов. Так, если имеется один жесткий диск и у него есть первичный и
вторичный разделы, причем последний разбит на два логических диска, то мы увидим следующее:

C: - первичный раздел;
D: - первый логический диск расширенного раздела:
E: - второй логический диск расширенного раздела.

Теперь если добавить второй жесткий диск (всего с одним первичным разделом), то картина изменится:

C: - первичный раздел первого диска (остался на месте);
D: - первичный раздел второго диска (новый);
E: - первый логический диск расширенного раздела первого диска (тот, что был D:);
F: - второй логический диск расширенного раздела первого диска (тот, что был E:).

Если у нового диска был бы расширенный раздел со своими логическими дисками, то они бы заняли следующие буквы (G:, H:, ...). О механизме присвоения логических имен следует помнить, устанавливая программы на компьютер, к которому эпизодически подключают дополнительные винчестеры. Незыблемое имя (C:) будет только у первичного раздела винчестера, подключенного ведущим к первому контроллеру ATA.

Загрузочный сектор (BOOT)

На этапе логического форматирования каждого раздела (логического диска) создаются четыре логических области:
· загрузочный сектор (boot sector);
· таблица размещения файлов {ЕКГ1 и FAT2);
· каталог;
· область данных.

Загрузочный сектор на любом логическом диске (разделе) располагается первым. Его блок данных (512 байт) начинается с команды JMP, которая передает управление на программу IPL2, содержит имя операционной системы и ее версию, содержит блок параметров BIOS диска (BPB), программу IPL 2, загружающую операционную систему и заканчивается сигнатурой 55AA.. Ниже в табл.5 поясняются некоторые из его важнейших записей.

Изменения в загрузочном секторе

Число зарезервированных секторов

Число зарезервированных секторов теперь перед первой FAT равно 32.

Новый блок параметров BIOS

Блок параметров BIOS в FAT32 занимает больше места, чем стандартный, и называется Big FAT BIOS Parameter Block (BF_BPB). Из-за этого загрузочный сектор теперь занимает не один, а три физических сектора, причем имеется еще дополнительный и размещается через три физических сектора в седьмом, восьмом и девятом физическом секторе.
BF_BPB - это расширенная версия BPB, присутствовавшего в 12- и 16-разрядной FAT. Он содержит те же структуры, что и стандартный BPB, но включает несколько дополнительных полей, которые нужны для FAT32. Изменения, внесенные в BPB для поддержки FAT32, описаны ниже.

Таблица 5. Важнейшие записи в загрузочном секторе

Длина (в байтах)	Содержимое
3	Команды JMP и NOP
8	Название и версия Windows
2	Количество байтов на сектор
1	Количество секторов на кластер (всегда кратно двум в степени n)
2	Количество зарезервированных секторов перед первой FAT
1	Количество таблиц FAT
2	Количество элементов в корневом каталоге (максимальный предел)
2	Общее число секторов (00 00 - если размер диска больше 32 Мб)
1	Дескриптор среды; в данном случае F8, что идентифицирует диск как жесткий с любой емкостью
2	Количество секторов на элемент таблицы FAT
2	Количество секторов на дорожку
2	Число головок
4	Количество скрытых секторов
4	Общее число секторов, если размер диска больше 32 Мб
1	Номер диска; в данном случае 80, что идентифицирует основной раздел
1	Зарезервирован
1	Расширенная сигнатура (всегда 29h)
4	Серийный номер тома
11	Метка тома
8	Тип файловой системы (12- или 16-разрядная)

Примечание. Эта часть загрузочного сектора известна как BIOS Parameter Block (BPB) (блок параметров BIOS). Она содержит физические характеристики диска, которые MS-DOS и Windows используют при поиске определенного участка. Складывая или перемножая значения этих параметров, операционная система узнает, где находится таблица FAT, корневой каталог, где начинается и кончается область данных.

Поле корневого каталога.

Этот элемент сообщает количество секторов в корневом каталоге. Для жестких дисков это значение всегда было равно 512 (0200h) и означало количество строк каталога размещаемых в тридцати двух секторах. Теперь оно изменено на 0 (0000h) и на FAT32-дисках игнорируется.

Количество секторов на элемент таблицы FAT.

Этот элемент заменен нулем и теперь действует как указатель на соответствующий элемент в BF_BPB, когда в процессе загрузки дело доходит до BF_BPB.

Описание диска.

Новое двухбайтовое поле, используемое как флаг, указывающий количество таблиц FAT на диске - одна или две. Если флаг установлен, на диске только одна FAT, если сброшен - две. FAT32, созданная командой Format, всегда формирует 2 таблицы FAT.

Первый кластер корневого каталога.

Максимальное число элементов в корневом каталоге теперь расширено до 65535, а сам корневой каталог может находиться в любом месте. Данное значение указывает номер первого кластера, занимаемого корневым каталогом на FAT32-диске.

Сектор файловой информации.

Указывает на второй загрузочный сектор. В нем содержится информация о том, сколько на диске всего кластеров, сколько из них свободно и какой кластер был выделен самым последним. Таким образом, чтобы получить эту часто используемую информацию, теперь не нужно считывать всю таблицу FAT.

Резервная копия загрузочного сектора.

Еще одно важное новшество в FAT32. В прежних версиях файловой системы FAT повреждение загрузочного сектора приводило к полной потере всего содержимого диска. FAT32 снимает остроту этой проблемы. Записывая изменения на загрузочный том FAT32, программа FDISK создает резервную копию загрузочного сектора и помещает ее в логический сектор 6 этого тома. Если новая MBR при обращении к загрузочному сектору обнаруживает ошибку чтения или неправильную сигнатуру, она ищет сектор 6 и считывает остальную часть загрузочного кода уже из него.

32-разрядная FAT-таблица

Предназначение FAT не изменилось. Она по-прежнему используется как таблица, связывающая отдельные кластеры файла. Элементы каталожной записи, указывающие на первый кластер файла, теперь состоят из четырех байтов, а содержимое этих байтов является номером (адресом) следующего кластера и элемента таблицы FAT, который содержит (указывает) номер следующего кластера файла, а также является номером элемента таблицы FAT и т. д. до последнего кластера файла. Элементы таблицы FAT теперь в 2 раза длиннее (по 4 байта), так как на FAT32-диске может быть гораздо больше кластеров, чем на FAT16-диске. В 16-разрядной FAT максимальное число кластеров на диске равно 65 525 (2 - за вычетом 10 зарезервированных), а в 32-разрядной FAT старшие 4 бита каждого 32-битного значения зарезервированы и не участвуют в формировании номера кластера, поэтому максимальное число кластеров в 32-разрядной FAT равно 268 435 445. (228 за вычетом 10 зарезервированных).

Каталог в FAT32

Начальный кластер, указанный в 32-х байтной строке каталога, сообщает операционной системе, где на диске искать первую часть файла и где в таблице FAT32 искать следующий номер кластера. В показанной ниже строке каталога адрес начального кластера выделен полужирным шрифтом.

49 4F 20 20 20 20 20 20 - 44 4F 53 07 00 00 00 00	IO SYS.....
00 00 00 00 00 00 80 32 - 3E 1B 02 00 46 9F 00 00	.....

Для указания номера кластера используются 2 дополнительных байта. Они размещаются в зарезервированной области, и в примере, показанном выше, это - 00 00. Объединяя их с обычными (существующими в FAT 16) двумя байтами (02 00), операционная система получает нужное значение (00 00 00 02) и ищет по нему соответствующий элемент таблицы FAT. Ниже показан пример записи с номерами кластеров файла в 32-разрядной таблице FAT:

F8 FF FF 0F FF FF FF 0F - 03 00 00 00 04 00 00 00
05 00 00 00 06 00 00 00 - 07 00 00 00 08 00 00 00
09 00 00 00 0A 00 00 00 - 0B 00 00 00 0C 00 00 00
0D 00 00 00 0E 00 00 00 - 0F 00 00 00 10 00 00 00
11 00 00 00 12 00 00 00 - 13 00 00 00 14 00 00 00
15 00 00 00 16 00 00 00 - 17 00 00 00 18 00 00 00
19 00 00 00 1A 00 00 00 - 1B 00 00 00 FF FF FF F8

Как и раньше в FAT 16, F8 - это байт, содержащий дескриптор носителя. Следующие 7 байтов, FF FF 0F FF FF FF 0F, зарезервированы. Номера кластеров записываются как четырехбайтовые числа. Их следует читать так:

03 00 00 00	04 00 00 00	05 00 00 00	06 00 00 00
00000003	00000004	00000005	00000006

Конец цепочки кластеров для файла помечается новым маркером - FFFFFFF8.

Зеркализация FAT

Исторически сложилось так, что на всех FAT-дисках существуют 2 экземпляра таблицы FAT. Если при чтении исходного экземпляра возникает ошибка, файловая система пытается считать его резервную копию. На дисках с 12-и 16-разрядной FAT первая таблица FAT всегда является основной, и все изменения автоматически записываются в ее копию. Создание резервной копии второй таблицы FAT называется зеркализацией (mirroring). В FAT32 зеркализацию второй таблицы FAT можно отключить. Тогда операции чтения/записи ускоряются, а если первая FAT оказывается поврежденной, используется ее второй экземпляр (он становится основным). На FAT32-дисках таблица FAT может достигать огромных размеров, и отключение зеркализации способно заметно ускорить доступ к файлам. В самой Windows 9x/Me нет механизма, позволяющего это сделать. Зеркализация всегда включена. Но ничто не мешает разработчикам реализовать в своих дисковых утилитах отключение зеркализации на дисках очень большого объема.

Корневой каталог

Корневой каталог в FAT32 может содержать до 65 535 элементов. В загрузочном секторе появился новый элемент, который указывает на первый кластер корневого каталога. Поэтому корневой каталог больше не привязан к строго определенному участка на диске (раньше он должен был находиться непосредственно за второй таблицей FAT) и может расширяться точно так же, как и любой подкаталог. Однако при наличии большого количества элементов в корневом каталоге поиск нужных данных занимает довольно много времени. Производительность файловой системы из-за этого падает. Поэтому лучше ограничивать число элементов в корневом каталоге до какого-то разумного предела.
В любых файловых системах Windows 9x/Me пользователи могут присваивать файлам имена длиной до 255 символов и более чем с одной точкой. Имя файла считается длинным, если оно превышает размеры, допускаемые форматом "8.3", или, если в нем содержатся строчные буквы и другие символы, недопустимые в пространстве имен формата "8.3".

Поддержка длинных имен файлов

С целью поддержки совместимости для каждого длинного имени файла автоматически генерируется псевдоним, удовлетворяющий формату "8.3". Этот псевдоним составляется из первых шести символов имени файла, дополняемых знаками ~n (где n - порядковый номер), и первых трех символов за последней точкой. Таким образом, файл DnisIsAdpg.File.Name получит псевдоним DNISIS~1.NAM. Если в каталоге уже есть такой псевдоним, порядковый номер увеличивается на единицу до тех пор, пока не получится уникальное имя. Ни пользователь, ни приложение не могут повлиять на процесс автоматического формирования псевдонима. В псевдониме используются только допустимые символы, а все буквы должны быть заглавными, чтобы соответствовать правилам формата "8.3". Для имен формата "8.3" (и псевдонимов) допустима любая комбинация букв и цифр, пробел (ASCII-код 20h), символы ASCII с кодами больше 127, а также знаки:
$ %'-_@ ~`!()^#&.
Следующие символы допустимы в длинных именах файлов, но недопустимы в псевдонимах или именах формата "8.3":
+,;=[]
Кроме того, файловые системы Windows 9x/Me подчиняются таким правилам:

- максимальная длина имени файла - 255 знаков, включая символ NULL;
- максимальная длина пути - 260 знаков, включая символ NULL (сравните с 80 знаками для краткого имени);
- набор символов OEM, используемый устанавливаемой файловой системой, определяется реестром и содержимым файла Unicode.bin;
- при хранении длинных имен файлов в записях каталогов на диске используется Unicode.

Имя файла и псевдоним одинаковы, если имя соответствует формату "8.3" (т. е. содержит только допустимые для псевдонима символы, и все буквы заглавные). Отсюда следует, что имя файла, во всем совпадающее с псевдонимом за исключением того, что содержит строчные буквы, все равно считается длинным. В таком случае псевдоним формируется простым преобразованием строчных букв в заглавные - например, Examples.Txt трансформируется в EXAMPLES .TXT. (при поиске в файловой системе Windows 98 регистр букв не учитывается).

Чтобы увидеть псевдоним файла, щелкните имя файла правой кнопкой мыши в любой оболочке типа Windows Explorer и выберите из контекстного меню команду Properties (Свойства). Псевдоним будет показан как параметр MS-DOS Name (Имя MS-DOS) в окне

свойств на вкладке General (Общие). Команда dir, введенная в командной строке, отображает только длинные имена файлов.

Размещение длинных имен в каталожной записи

В начале тома FAT12 или FAT16 заранее выделяется место для корневого каталога, достаточное для хранения 256 записей (элементов), что ограничивает число файлов и каталогов в корневом каталоге (в FAT32 такого ограничения нет). Элемент каталога FAT, размер которого составляет 32 байта, хранит имя файла, его размер, начальный кластер и метку времени (время создания, последнего доступа и т.д.). Если имя файла состоит из Unicode-символов или не соответствует правилам именования по формуле "8.3", принятым в MS-DOS, оно считается длинным, и для его хранения выделяются дополнительные элементы каталога.
Вспомогательные элементы предшествуют главному элементу для файла. На рисунке показан пример элемента каталога для файла с именем "The quick brown fox". Система создала представление этого имени в формате "8.3", THEQUI~1.FOX (в элементе каталога нет "точки", поскольку предполагается, что точка следует после восьмого символа), и использовала два дополнительных 32-х байтных элемента для хранения длинного Unicode-имени. Каждая строка на рисунке состоит из 16 байт.

Второй (и последний) элемент для длинного имени

0x42	w	n	.	f	o	0x01	0x00	Контр. сумма	x
0x0000	0xFFFF	0xFFFF	0xFFFF	0xFFFF	0x0000	0xFFFF	0xFFFF		

Первый элемент для длинного имени

0x01	T	h	e		q	0x0F	0x00	Контр. сумма	u
i	c	k		b	0x0000	r	o		

Элемент для краткого имени

T	H	E	Q	U	I	~	1	F	O	X	0x20	NT	Время создания
Дата создания		Время последнего доступа		0x0000		Время последней модификации		Дата последней модификации			Первый кластер		Размер файла

NTFS

NTFS (New Technology File System) - наиболее предпочтительная файловая система при работе с ОС Windows NT (Windows 2000 и XP также являются NT системами), поскольку она была специально разработана для данной системы. В состав Windows NT входит утилита convert, осуществляющая конвертирование томов с FAT и HPFS в тома NTFS. В NTFS значительно расширены возможности по управлению доступом к отдельным файлам и

каталогам, введено большое число атрибутов, реализована отказоустойчивость, средства динамического сжатия файлов, поддержка требований стандарта POSIX. NTFS позволяет использовать имена файлов длиной до 255 символов, при этом она использует тот же алгоритм для генерации короткого имени, что и VFAT. NTFS обладает возможностью самостоятельного восстановления в случае сбоя ОС или оборудования, так что дисковый том остается доступным, а структура каталогов не нарушается.

Каждый файл на томе NTFS представлен записью в специальном файле – главной файловой таблице MFT (Master File Table). NTFS резервирует первые 16 записей таблицы размером около 1 Мб для специальной информации. Первая запись таблицы описывает непосредственно саму главную файловую таблицу. За ней следует зеркальная запись MFT. Если первая запись MFT разрушена, NTFS считывает вторую запись, чтобы отыскать зеркальный файл MFT, первая запись которого идентична первой записи MFT. Местоположение сегментов данных MFT и зеркального файла MFT хранится в секторе начальной загрузки. Копия сектора начальной загрузки находится в логическом центре диска. Третья запись MFT содержит файл регистрации, применяемый для восстановления файлов. Семнадцатая и последующие записи главной файловой таблицы используются собственно файлами и каталогами на томе.

В журнале транзакций (log file) регистрируются все операции, влияющие на структуру тома, включая создание файла и любые команды, изменяющие структуру каталогов. Журнал транзакций применяется для восстановления тома NTFS после сбоя системы. Запись для корневого каталога содержит список файлов и каталогов, хранящихся в корневом каталоге.

Схема распределения пространства на томе хранится в файле битовой карты (bitmap file). Атрибут данных этого файла содержит битовую карту, каждый бит которой представляет один кластер тома и указывает, свободен ли данный кластер или занят некоторым файлом.

В загрузочном файле (boot file) хранится код начального загрузчика Windows NT.

NTFS также поддерживает файл плохих кластеров (bad cluster file) для регистрации поврежденных участков на томе и файл тома (volume file), содержащий имя тома, версию NTFS и бит, который устанавливается при повреждении тома. Наконец, имеется файл, содержащий таблицу определения атрибутов (attribute definition table), которая задает типы атрибутов, поддерживаемые на томе, и указывает можно ли их индексировать, восстанавливать операцией восстановления системы и т.д.

NTFS распределяет пространство кластерами и использует для их нумерации 64 разряда, что дает возможность иметь 2^{64} кластеров, каждый размером до 64 Кбайт. Как и в FAT размер кластера может меняться, но необязательно возрастает пропорционально размеру диска. Размеры кластеров, устанавливаемые по умолчанию при форматировании раздела, приведены в табл. 6.

Размер раздела	Размер кластера
< 512 Мб	512 байт
513 Мб - 1024 Мб (1 Гб)	1 Кб
1 Гб - 2 Гб	2 Кб
2 Гб - 4 Гб	4 Кб
4 Гб - 8 Гб	8 Кб
8 Гб - 16 Гб	16 Кб
16 Гб - 32 Гб	32 Кб

> 32 Гб	64 Кб

Табл. 6

NTFS позволяет хранить файлы размером до 16 эксабайт (2^{64} байт) и располагает встроенным средством уплотнения файлов в реальном времени. Сжатие является одним из атрибутов файла или каталога и подобно любому атрибуту может быть снято или установлено в любой момент (сжатие возможно на разделах с размером кластера не более 4 Кб). При уплотнении файла, в отличие от схем уплотнения используемых в FAT, применяется пофайловое уплотнение, таким образом, порча небольшого участка диска не приводит к потере информации в других файлах.

Для уменьшения фрагментации NTFS всегда пытается сохранить файлы в непрерывных блоках. Эта система использует структуру каталогов в виде B-дерева, аналогичную высокопроизводительной файловой системе HPFS, а не структуре со связанным списком применяемой в FAT. Благодаря этому поиск файлов в каталоге осуществляется быстрее, поскольку имена файлов хранятся сортированными в лексикографическом порядке.

NTFS была разработана как восстанавливаемая файловая система, использующая модель обработки транзакций. Каждая операция ввода-вывода, изменяющая файл на томе NTFS, рассматривается системой как транзакция и может выполняться как неделимый блок. При модификации файла пользователем сервис файла регистрации фиксирует всю информацию необходимую для повторения или отката транзакции. Если транзакция завершена успешно, производится модификация файла. Если нет, NTFS производит откат транзакции.

Несмотря на наличие защиты от несанкционированного доступа к данным NTFS не обеспечивает необходимую конфиденциальность хранимой информации. Для получения доступа к файлам достаточно загрузить компьютер в DOS с дискеты и воспользоваться каким-нибудь сторонним драйвером NTFS для этой системы.

Начиная с версии Windows 2000 Microsoft поддерживает новую файловую систему NTFS 5.0. В новой версии NTFS были введены дополнительные атрибуты файлов; наряду с правом доступа введено понятие запрета доступа, позволяющее, например, при наследовании пользователем прав группы на какой-нибудь файл, запретить ему возможность изменять его содержимое. Новая система также позволяет:

- вводить ограничения (квоты) на размер дискового пространства, предоставленного пользователям;
- проецировать любой каталог (как на локальном, так и на удаленном компьютере) в подкаталог на локальном диске.

Интересной возможностью новой версии Windows NT является динамическое шифрование файлов и каталогов, повышающее надежность хранения информации. В состав Windows 2000 и Windows XP входит файловая система с шифрованием (Encrypting File System, EFS), использующая алгоритмы шифрования с общим ключом. Если для файла установлен атрибут шифрования, то при обращении пользовательской программы к файлу для записи или чтения происходит прозрачное для программы кодирование и декодирование файла.

UFS (Unix File System)

Так же, как Unix представляет не одну систему, а ряд совместимых, так же UFS - не одна система, а целый ряд. Информации о поддержке разными Unix'ами чужих UFS у меня пока

нет; информацию по поводу поддержки чужих файловых систем для каждого конкретного Unix'а скорее всего можно найти в документации к программе 'mount'.

Основным отличием UFS от других известных мне систем является выделение атрибутов файла в отдельном объекте файловой системе - inode; это позволяет иметь доступ к файлу (к набору данных, хранящихся в файле) более чем по одному имени (так называемый жесткий линк; см.ниже), а заодно повысить эффективность функционирования системы.

Классическая UFS Отводит на файл 16 байт - 14-буквенное имя файла и двухбайтный номер inode; современые UFS позволяют создавать длинные имена (до 255 символов), а имена файлов хранят не подряд, а более разумно - в двоичном дереве или hash-таблице, а номер inode может быть любым - четырехбайтным или восьмибайтным.

Сам блок inode содержит:

- количество ссылок на файл - каждое имя, ссылающееся на файл, а также открытие файла увеличивают этот счетчик на единицу; файл стирается с высвобождением занятого места как только счетчик становится равным нулю (т.е. можно стереть открытый файл, а реально он сотрется когда его закроют);
- размер файла;
- дату и время создания, последнего изменения и последего чтения файла;
- тип файла - в Unix это бывает:
    - обычный файл;
    - директория;
    - файл блочного устройства;
    - файл символьного (последовательного) устройства;
    - поименованный пайп (название происходит от символа "|", называемого "pipe" - см.его значение в shell);
    - символьный линк (алиас);

      обычный файл и директория встречаются во всех файловых системах; файлы устройств являются интерфейсами к драйверам этих устройств;

- UID (идентификатор хозяина файла) и GID (идентификатор группы);
- атрибуты доступа:
    - Unix использует атрибуты 'Read', 'Write' и 'eXecute' для хозяина файла (owner), для одногрупника (group) и для остальных (other) - итого 9 бит; для директории эти атрибуты означают соответственно права на чтение списка файлов, на создание/удаление файлов и на обращение к файлам внутри директории;

  важной особенностью является то, что права доступа для хозяина определяются атрибутами для него, права для одногрупника и остальных для хозяина игнорируются; аналогично для одногркпника не играют никакой роли права для остальных;

- кроме них есть атрибуты SetUID и SetGID - для запускаемого файла (не интерпретируемого) эти атрибуты определяют запуск процесса под правами не запустившего их пользователя, а хозяина и/или группы файла соотвнтственно;
- и еще есть один атрибут - для директории он запрещает стирание файлов, не принадлежащих стирающему;
- расширенный ACL (Access Control List, Список Управления Доступом) или ссылку на ACL, если файловая система поддерживает ACL;

- несколько (в классической UFS - 13) ссылок на кластеры файловой системы (раскладка приведена для классической UFS):
    - первые 10 указывают на первые 10 кластеров файла;
    - 11-й указывает на кластер, содержащий адреса следующих 128-ми кластеров файла (в классической UFS размер кластера - полкилобайта, а адрес кластера - четыре байта);
    - 12-й указывает на кластер, содержащий адреса 128-ми кластеров, в свою очередь содержащих адреса следующих 16`384-рех кластеров файла;
    - последний указывает на кластер, ... вобщем, здесь используется еще на один уровень больше, что позволяет адресовать еще 2`097`152 кластера файла;

  итого получается 2`113`674 кластера по полкилобайта - чуть более гигабайта в файловой системе, способной работать с томами до двух терабайт ($2^{\wedge 32}$ кластеров по полкилобайта).
  В современных UFS многое изменено: можно задавать произвольный размер кластера и использовать 64-битные указатели, так что ограничени классической UFS давно преодолены. Основное преимущество такой адресации в том, что маленькие файлы, к которым часто обращаются, достижимы прямо из inode, и так же быстро происходит обращение к началу большого файла; обращение в середину и конец большого файла происходят медленнее, чем в начало, но я не представляю, как можно обеспечить бОльшую скорость, не налагая жесткого требования заведомой дефрагментированности файла или хотя бы таблицы размещения его кластеров.

Во многих UFS если после создания файла в кластер ничего не писалось (например, после открытия файла переместили указатель далеко-далеко и что-то туда записали), то под этот кластер не отводится место, а ссылке, которая должна на него указывать, присваивается 0 (это особенно актуально в свете использования hash-таблиц, которые обычно имеют внутри себя пустое пространство). То же самое делается если кластер оказывается заполнен нулями (незаполненное место считается залитым нулями, хотя полагаться на это программисту я бы не советовал). По-моему, неплохой способ экономии места.

Часть систем UFS реализованы как отказоустойчивые с журналированием, а часть по традиции обходится без этого - считается, что если на машине хранятся действительно важные данные, то эту машину можно обеспечить бесперебойным питанием и не подпускать к ней ламеров.

Заключение

Развитие файловых систем персональных компьютеров определялось двумя факторами - появлением новых стандартов на носители информации и ростом требований к характеристикам файловой системы со стороны прикладных программ (разграничение уровней доступа, поддержка длинных имен файлов в формате UNICODE). Первоначально, для файловых систем первостепенное значение имело увеличение скорости доступа к данным и минимизация объема хранимой служебной информации. Впоследствии с появлением более быстрых жестких дисков и увеличением их объемов, на первый план вышло требование надежности хранения информации, которое привело к необходимости избыточного хранения данных.

Эволюция файловой системы была напрямую связана с развитием технологий реляционных баз данных. Файловая система использовала последние достижения, разработанные для применения в СУБД: механизмы транзакций, защиты данных, систему самовосстановления в результате сбоя.

Развитие файловых систем привело к изменению самого понятия "файл" от первоначального толкования как упорядоченная последовательность логических записей, до понятия файла, как объекта, имеющего набор характеризующих его атрибутов (включая имя файла, его псевдоним, время создания и собственно данные), реализованного в Структура NTFS.

За свою почти 30 летнюю историю файловая система прошла путь от простой системы, взявшей на себя функции управления файлами, до системы, представляющей собой полноценную СУБД, обладающую встроенным механизмом протоколирования и восстановления данных.

В отличие от попыток ввести стандарт на протокол, описывающий правила доступа к удаленным файловым системам (CIFS, NFS), не стоит ожидать появления подобного стандарта, описывающего файловые системы для жестких дисков. Это можно объяснить тем, что файловая система жестких дисков все еще продолжает оставаться одной из главных частей операционной системы, влияющей на ее производительность. Поэтому каждый производитель операционных систем будет стремиться использовать файловую систему, "родную" для его ОС.

Список использованной литературы

1. Робачевский А.М. "Операционная система UNIX"

2. Соломон Д., Руссинович М., “Внутреннее устройство Microsoft Windows 2000: Мастер-класс: Перевод с английского”, "Питер" - 2004, 717 стр

3. http://www.3dnews.ru/ [Электронный ресурс]