**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 file system 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.

**2 Коротко о 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.На основе ее и были созданы VFAT и FAT 32

**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. 2. 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 FАТ32 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 FАТ32 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 FАТ32-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 реальногорежима 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. 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 716С (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 FАТ12 - по 1,5.

Старшие 4 бита 32-разрядного элемента таблицы FAT32 зарезервированы и не участвуют в формировании номера кластера. Программы, напрямую считывающие РАТ32-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 FАТ32-дисках расширена и теперь включает резервную копию критически важных структур данных, а это означает, что FАТ32-диски менее чувствительны к возникновению отдельных сбойных участков, чем существующие FAT-тома;
* программы загружаются на 50% быстрее.

|  |  |  |
| --- | --- | --- |
| **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 |

Таблица 3. Сравнение размеров кластеров

Усовершенствованная утилита дефрагментации дисков оптимизирует размещение файлов приложения, загружаемых в момент его запуска. Возможно преобразование диска в FАТ32 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 нажестком 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 FАТ32). 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 АРМ 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 0000:7С00 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 0000:7С00 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.

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 (0В) и DOS32X (ОС).

**DOS32 (0В).**

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.

**DOS32X** (0С). 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.х или более ранних.

Разделы с кодами (01, 04, 06, 0В, ОС, ОЕ) являются первичными разделами DOS/Windows.

Утилита FDISK из MS-DOS и Windows 9х/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 С:, для второго - D: и т. д. В операционных системах MS-DOS и Windows 9х/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 9х/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:

С: - первичный раздел;

D: - первый логический диск расширенного раздела:

Е: - второй логический диск расширенного раздела.

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

С: - первичный раздел первого диска (остался на месте);

D: - первичный раздел второго диска (новый);

Е: - первый логический диск расширенного раздела первого диска (тот, что был D:);

F: - второй логический диск расширенного раздела первого диска (тот, что был Е:).

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

**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 (ВРВ), программу IPL 2, загружающую операционную систему и заканчивается сигнатурой 55АА.. Ниже в таблице 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 FАТ32 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 ВРВ, 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.

Таблица 4. 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 п) | | |  | |
| 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.

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 FАТ32-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 FАТ32. 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 FАТ32-диске может быть гораздо больше кластеров, чем на FАТ16-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).

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.

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 0А 00 00 00 - 0В 00 00 00 0С 00 00 00 0D 00 00 00 0Е 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 1А 00 00 00 - 1В 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. Ecли 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 9х/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 9х/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 9х/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.

**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.