Конспекты лекций по курсу
«Введение в информатику и системы программирования», 1 семестр
С.А. Немнюгин, направление «Прикладные математика и физика»)

Лекция 13

Операционные системы

Архитектура операционных систем

Файловая система NTFS

NTFS разработана для:

- быстрого выполнения стандартных файловых операций чтения, записи и поиска;
- быстрого выполнения улучшенных операций типа восстановления файловой системы на очень больших жестких дисках.

NTFS поддерживает управление доступом к данным и привилегии владельца.
NTFS — единственная файловая система в Windows, которая позволяет назначить разрешения для отдельных файлов.

Диск NTFS делится на две части:

1. первые 12% диска отводятся под MFT зону - пространство, в которое растет метафайл MFT. Запись каких-либо данных в эту область невозможна. MFT-зона всегда держится пустой - это делается для того, чтобы самый главный, служебный файл (MFT) не фрагментировался при своем росте;
2. остальные 88% диска представляют собой обычное пространство для хранения файлов.

Master File Table - общая таблица файлов/ MFT - централизованный каталог всех остальных файлов диска, включая себя самого. Размещается в *MFT-зоне*.
MFT поделен на записи фиксированного размера (обычно 1 Кбайт). Каждая запись соответствует какому-либо файлу.

Метафайлы
Первые 16 файлов носят служебный характер, недоступны пользователю и называются *метафайлами*. Самый первый метафайл - сам MFT.

Первые 16 элементов MFT имеют фиксированное положение.
Вторая копия первых трех записей хранится ровно посередине диска.
Остальная часть MFT-файла может располагаться в произвольных местах диска - восстановить его положение можно с помощью его самого, «зацепившись» за первый элемент MFT.
NTFS может сместить, даже фрагментировать по диску, все свои служебные области, обойдя любые неисправности поверхности кроме первых 16 элементов MFT.

Метафайлы находятся корневом каталоге NTFS диска, их имена начинаются с символа $. Получить какую-либо информацию о них обычными средствами нельзя.

- **$MFTmirr -** копия первых 16 записей MFT, размещенная посередине диска;
- **$LogFile -** файл поддержки журналирования;
- **$Volume -** служебная информация - метка тома, версия файловой системы, и т.д.;
- **$AttrDef -** список стандартных атрибутов файлов на томе;
- **$. -** корневой каталог;
- **$Bitmap -** карта свободного места тома;
- **$Boot -** загрузочный сектор (если раздел загрузочный);
- **$Quota -** файл, в котором записаны права пользователей на использование дискового пространства и другие.

Характеристика файловых систем FAT32 и NTFS.

	FAT	FAT32	NTFS
Системы, её поддерживающие	DOS, Windows9X, NT всех версий	Windows98, NT5	NT4, NT5
Максимальный размер тома	2 Гбайт	практически неограничен	практически неограничен
Макс. число файлов на томе	примерно 65 тысяч	практически не ограничено	практически не ограничено
Имя файла	с поддержкой длинных имен - 255 символов, системный набор символов	с поддержкой длинных имен - 255 символов, системный набор символов	255 символов, любые символы любых алфавитов (65 тысяч разных начертаний)
Возможные атрибуты файла	Базовый набор	Базовый набор	всё, что придет в голову производителям программного обеспечения
Безопасность	нет	нет	да (начиная с NT5.0 встроена возможность физически шифровать данные)
Сжатие	нет	нет	да
Устойчивость к сбоям	средняя (система слишком проста и поэтому ломаться особо нечему :))	плохая (средства оптимизации по скорости привели к появлению слабых по надежности мест)	полная - автоматическое восстановление системы при любых сбоях (не считая физические ошибки записи, когда пишется одно, а на самом деле записывается другое)
Экономичность	минимальная (огромные размеры кластеров на больших дисках)	улучшена за счет уменьшения размеров кластеров	максимальна. Очень эффективная и разнообразная система хранения данных
	высокое для малого числа файлов, но	полностью аналогично FAT, но на	система не очень эффективна для малых и простых

Файловые системы UNIX

В ОС Linux используются **Extended 2 File System** и **Extended 3 File System** и другие. В других версиях ОС UNIX используются (или использовались) S5FS, UFS и многие, многие другие.

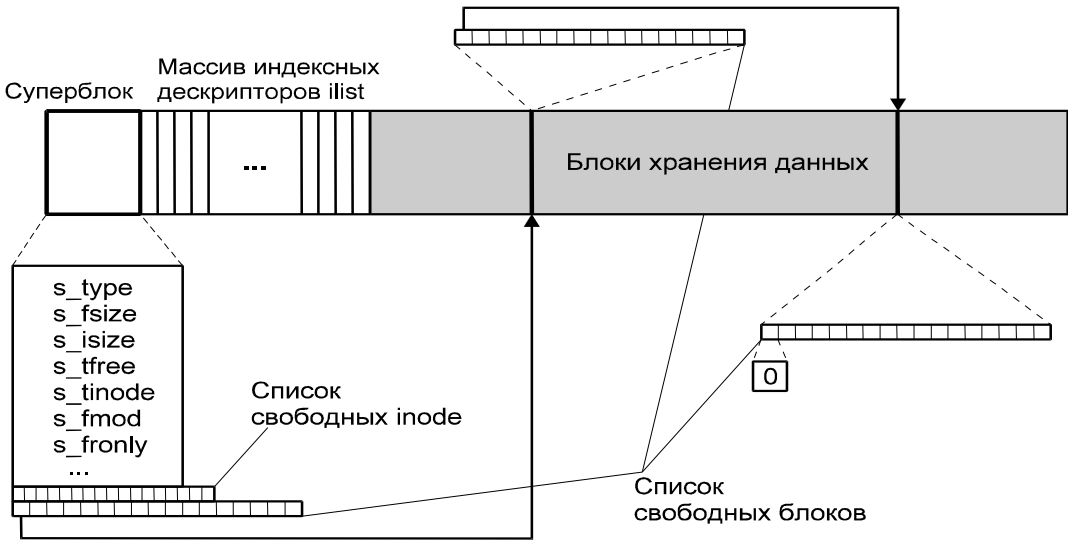
Особенности файловых систем UNIX:

- выполнение основных функций: размещение файлов, их удаление, операции чтения и записи в файлы, изменение атрибутов файлов;
- контроль доступа пользователей к файлам, то есть, реализация многопользовательского характера системы:
  права доступа – триада доступа (права на чтение, запись, исполнение для трех групп пользователей – владельца файла, группы владельца и «всех прочих пользователей»;
- обеспечение доступа к периферийным устройствам компьютера на основе единообразного интерфейса;
- единое иерархическое дерево файловой системы, к которому подключаются (монтируются) другие файловые системы, расположенные, возможно, на других носителях информации, а также на удаленных компьютерах (сетевые файловые системы NFS – Network File System, AFS – Andrew File System). Файловая система UNIX имеет один главный (корневой) каталог.

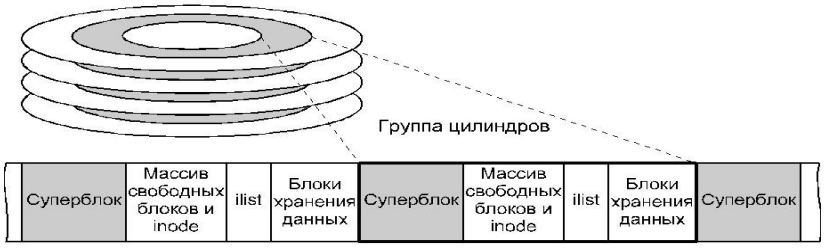
Типы файлов в файловых системах UNIX

1. Обычные
2. Каталоги
3. Специальные файлы устройств
4. Символические ссылки
5. Именованные каналы
6. Сокеты

Файловые системы UNIX s5fs



Файловые системы UNIX ffs



Файловые системы UNIX

Буферный кэш позволяет ускорить операции с файлами и является источником потенциальной опасности для целостности файловой системы.

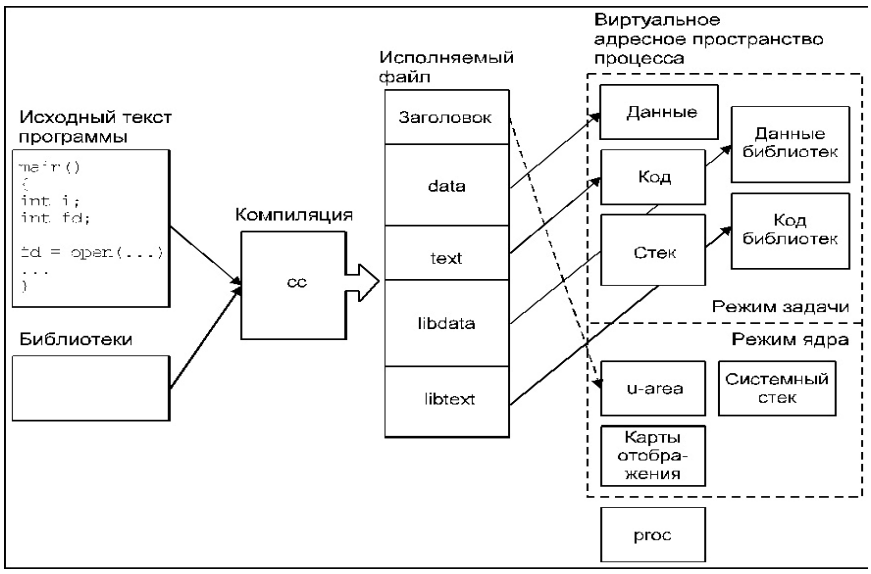
Журналирование и журналируемые файловые системы.

Сравнительная характеристика некоторых файловых систем UNIX

Характеристика	ReiserFS	Journalled File System (JFS)	ext2fs	ext3fs
Максимальный размер файловой системы	1 Тбайт	От 512 Тбайт (при блоке 512 байт) до 4 Пбайт (при блоке 4 Кбайта)	4 Тбайт	4 Тбайт
Максимальный размер файла	1 Тбайт	Определяется возможностями VFS	1 Тбайт	1 Тбайт
Максимальное количество файлов	32 768 каталогов, более 4 миллиардов файлов	Определяется размером файловой системы	Определяется размером файловой системы	Определяется размером файловой системы
Максимальная длина имени файла	255	-	255	255
Минимальный/максимальный размер блока	-	512/4096 байт	1024/4096 байт	1024/4096 байт

Подсистема управления процессами

Процесс представляет собой исполняемый образ программы, включающий отображение в памяти исполняемого файла, полученного в результате компиляции, стек, код и данные библиотек, а также ряд структур данных операционной системы, необходимых для управления процессом.



Планирование выполнения процессов требуется, если операционная система является многозадачной и строится на системе *приоритетов*.

Частью ОС является *планировщик*. В ОС разделения времени процессорное время делится на *кванты* – равновеликие интервалы, в течение которых тот или иной процесс монопольно владеет процессором.

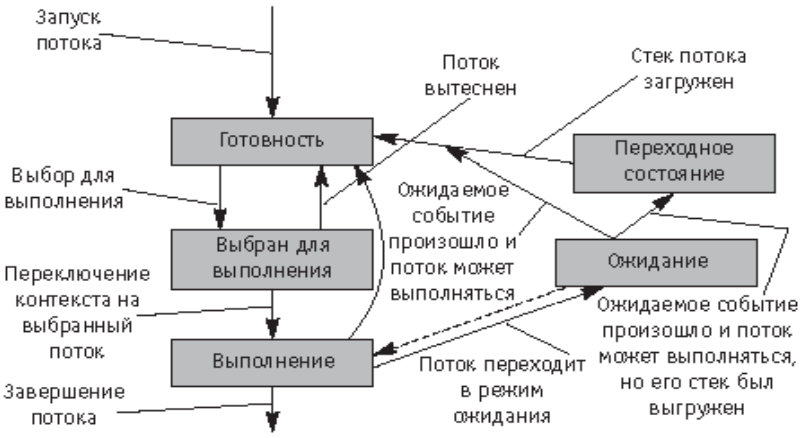
Подсистема управления процессами MS Windows

Вместо термина *процесс* в MS Windows используется термин *поток*. Считается, что процесс является объектом планирования адресного пространства, а поток является объектом планирования распределения процессорного времени.

Три базовых понятия:

1. процесс (process);
2. поток (thread) – связан с ядром ОС и планируется ядром;
3. нить (волокно, fiber) – связано с приложением пользователя и планируется в приложении пользователя.

Поток в процессе выполнения может находиться в разных состояниях:



При выборе потока для выполнения учитываются приоритеты потоков (абсолютные приоритеты) - система начинает выполнять код потока с наибольшим приоритетом из числа готовых к исполнению.

Квантование потоков осуществляется по тикам системного таймера, продолжительность одного тика составляет обычно 10 или 15 мс, больший по продолжительности тик назначают многопроцессорным машинам. Каждый тик системного таймера соответствует 3 условным единицам; величина кванта может варьироваться от 2 до 12 тиков (от 6 до 36 единиц).

В Windows выделяется 32 уровня приоритетов - 0 соответствует самому низкому приоритету, 31 - самому высокому. Этот диапазон делится на три части:

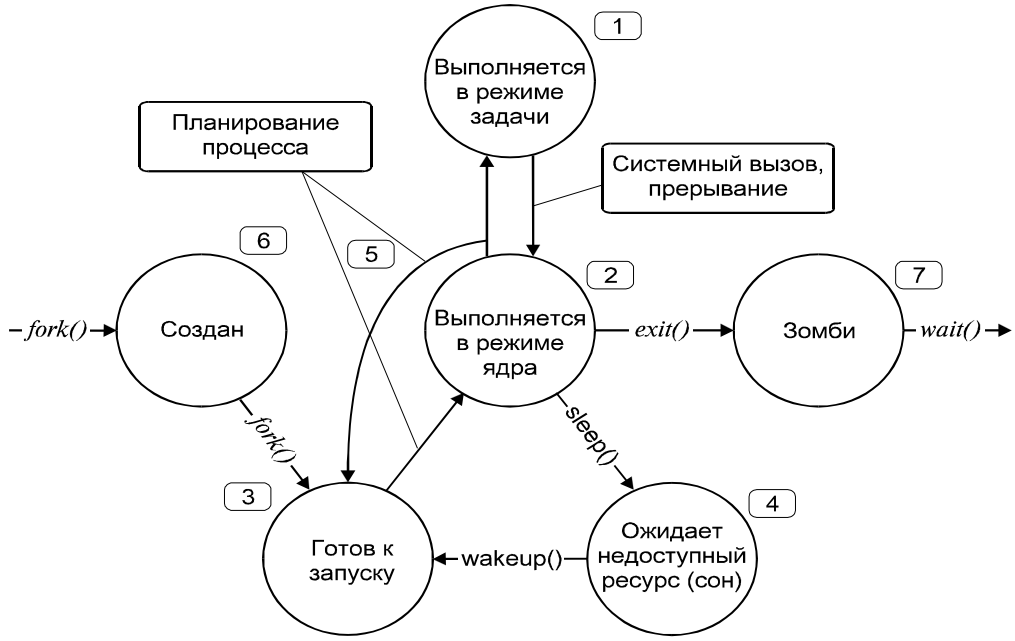
1. приоритет 0 соответствует приоритету потока обнуления страниц;
2. приоритеты с 1 по 15 соответствуют динамическим уровням приоритетов. Большинство потоков работают в этом диапазоне приоритетов, и Windows может корректировать в некоторых случаях приоритеты потоков из этого диапазона;
3. приоритеты с 16 по 31 соответствуют приоритетам «реального времени». Этот уровень достаточно высок для того, чтобы поток, работающий с таким приоритетом, мог реально помешать нормальной работе других потоков в системе – например, помешать обрабатывать сообщения от клавиатуры и мыши. Windows самостоятельно не корректирует приоритеты этого диапазона.

Планировщик операционной системы может корректировать уровень приоритета (из диапазона 1-15). Базовый уровень (класс) не может быть изменен. Такая коррекция приоритета выполняется в случае:

- завершения операции ввода-вывода - в зависимости от устройства, приоритет повышается на 1 - 8 уровней;
- по окончании ожидания события или семафора - на один уровень;
- при пробуждении GUI потоков - на 2 уровня;
- по окончании ожидания потоком активного процесса (определяется по активности интерфейса) - на величину, указанную младшими 2 битами параметра Win32PrioritySeparation.

В случае коррекции приоритета по одной из перечисленных причин, повышенный приоритет начинает постепенно снижаться до начального уровня потока - с каждым тиком таймера на один уровень.
Еще один случай повышения приоритета (вместе с увеличением длительности кванта) - процесс долгое время не получал процессорного времени. В этой ситуации система раз в 3-4 секунды назначает процессу приоритет, равный 15, и квант удвоенной длительности. По истечении этого кванта приоритет возвращается к прежнему значению и восстанавливается рекомендуемая длительность кванта.

Подсистема управления процессами ОС UNIX



Текущий приоритет процесса в режиме задачи p_priuser зависит от двух факторов:

1. значения относительного приоритета;
2. степени использования вычислительных ресурсов p_cpu:

p_priuser = a*p_nice — b*p_cpu,

где p_nice — постоянная составляющая, зависящая от параметра nice.

UNIX версии SVR3, использует следующую формулу:
p_cpu = p_cpu/2

4.3BSD UNIX для пересчета p_cpu используется другая формула:
p_cpu = p_cpu*(2*load)/(2*load+1)