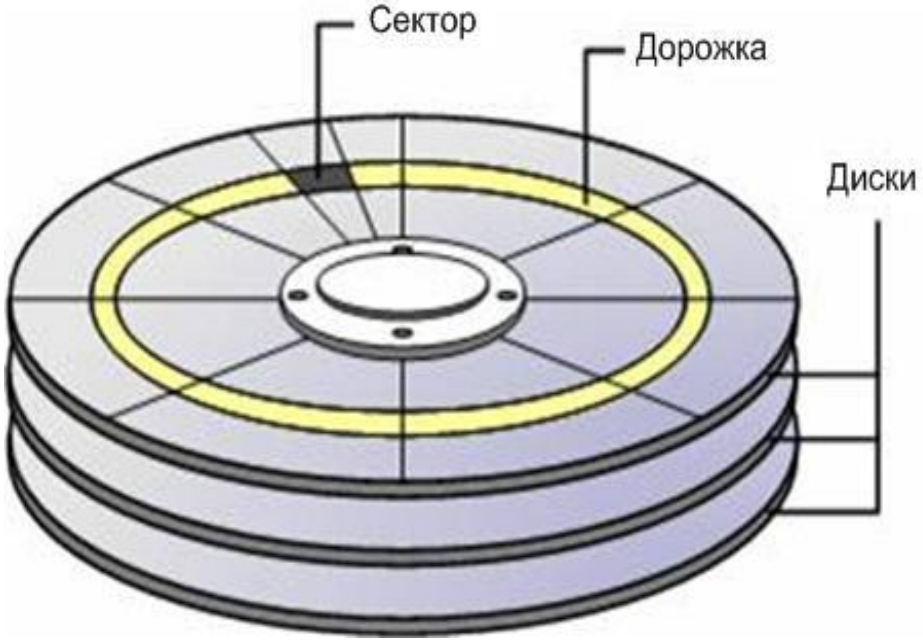
Файловые системы

Геометрия жёсткого диска

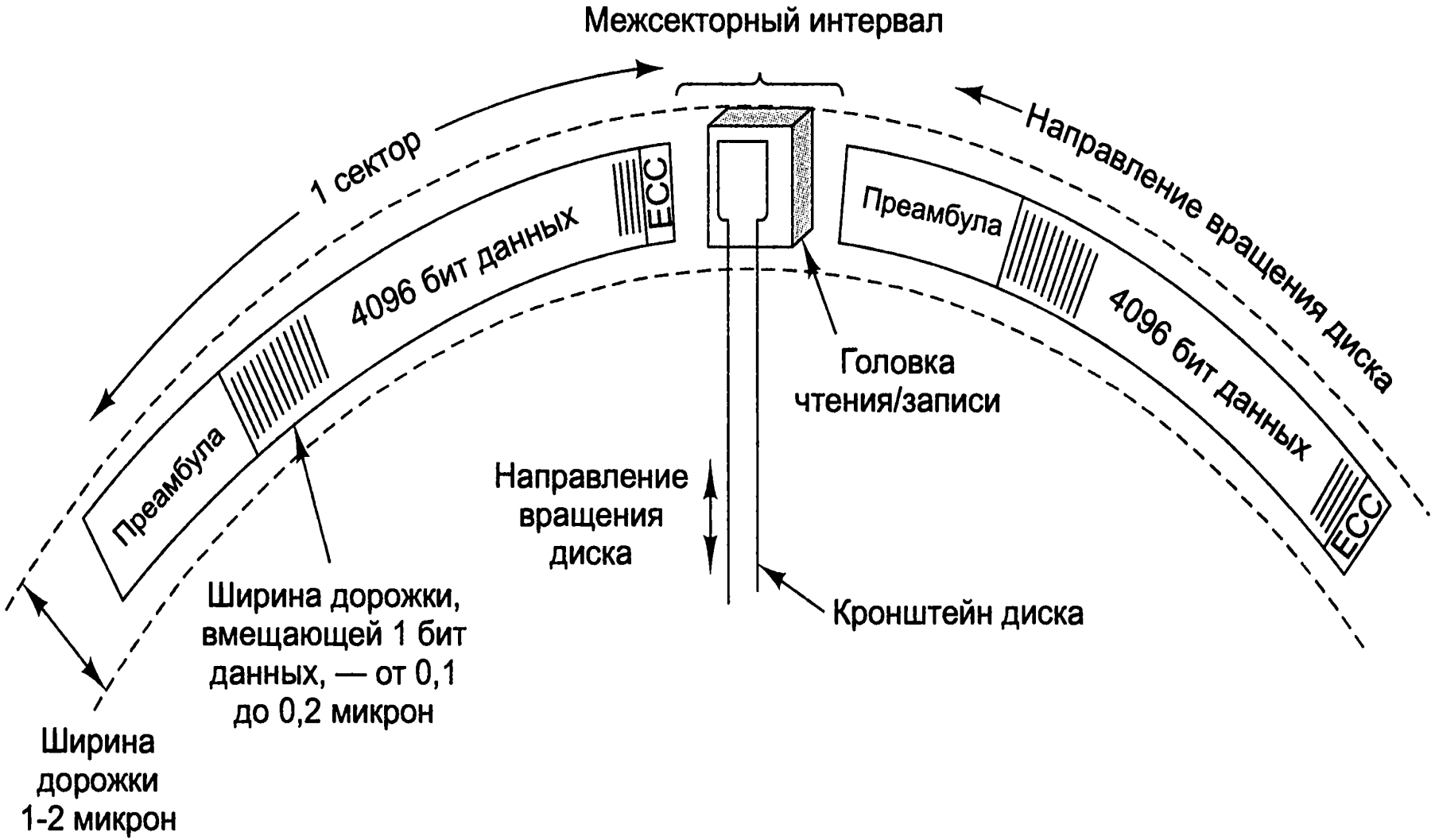
Жёсткий диск хранит информацию блоками фиксированного размера, которые называются ***секторами***. ***Сектор*** (sector) является наименьшей порцией данных, имеющей уникальный адрес на жестком диске. Размер сектора является стандартным для всех жестких дисков и составляет 512 байт.

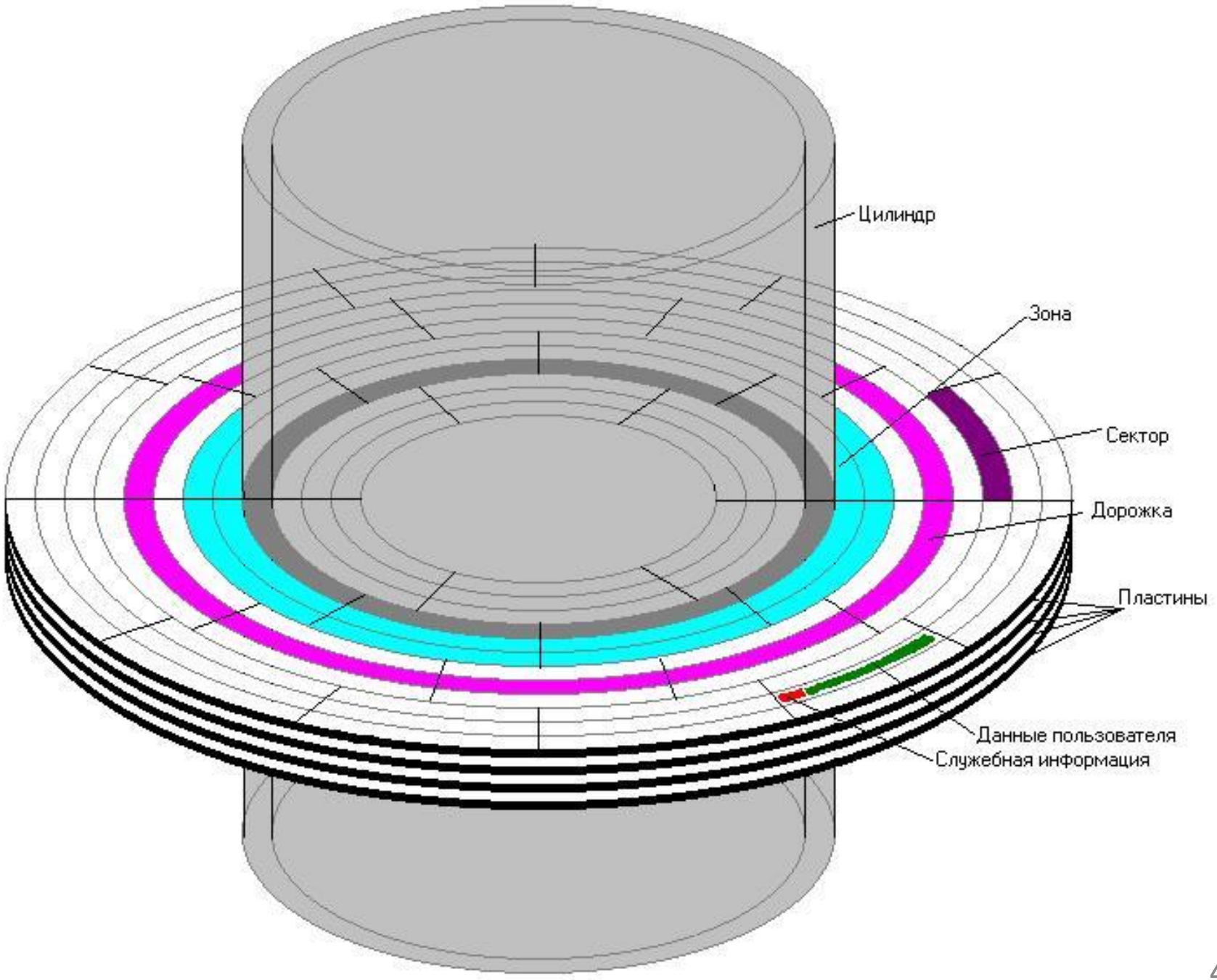
Для ускорения доступа к данным поверхность диска разделена на концентрические ***дорожки*** (track). Сектор является частью дорожки. Совокупность дорожек, одинаково удаленных от центра на всех рабочих поверхностях дисков, образует так называемый ***цилиндр*** (cylinder).



Кластер (cluster) — это минимальный участок памяти на диске, который может быть выделен файловой системой при создании файла. Физически кластер представляет собой несколько смежных секторов, число которых должно быть равно степени 2 (то есть кластер может включать 1, 2, 4, 8, 16, 32 или даже 64 сектора).

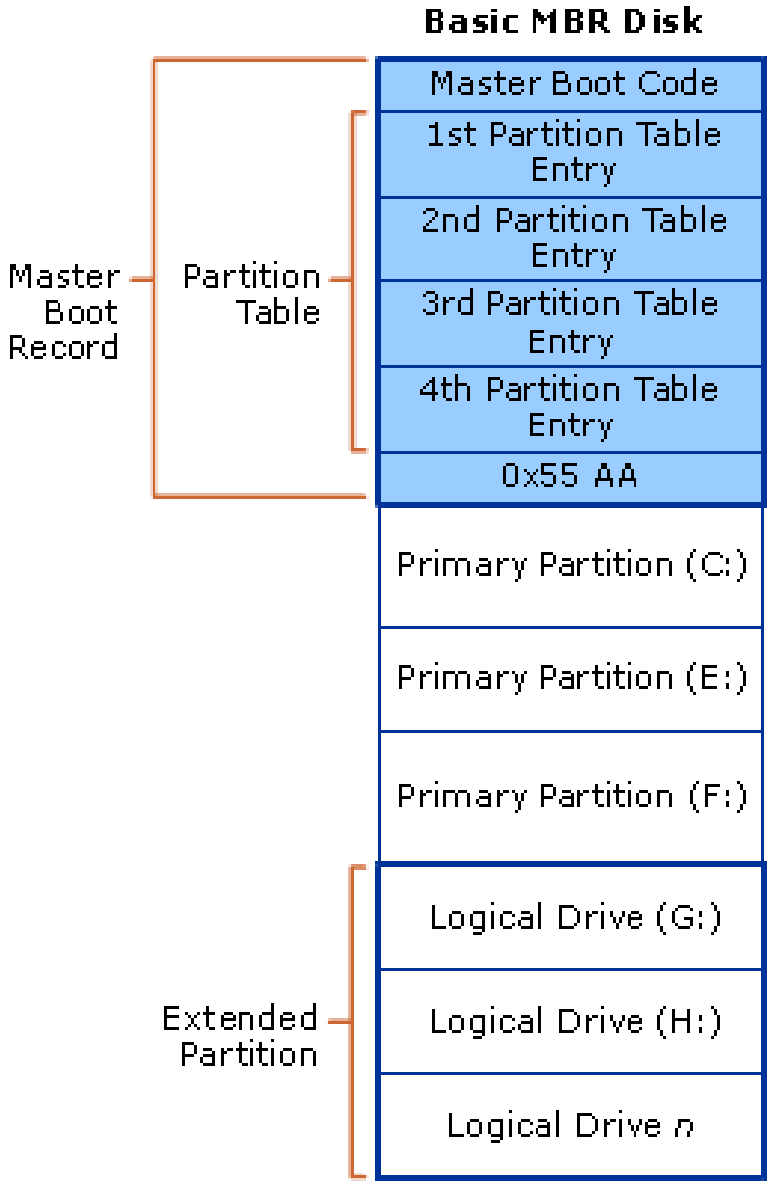
Дорожка жёсткого диска



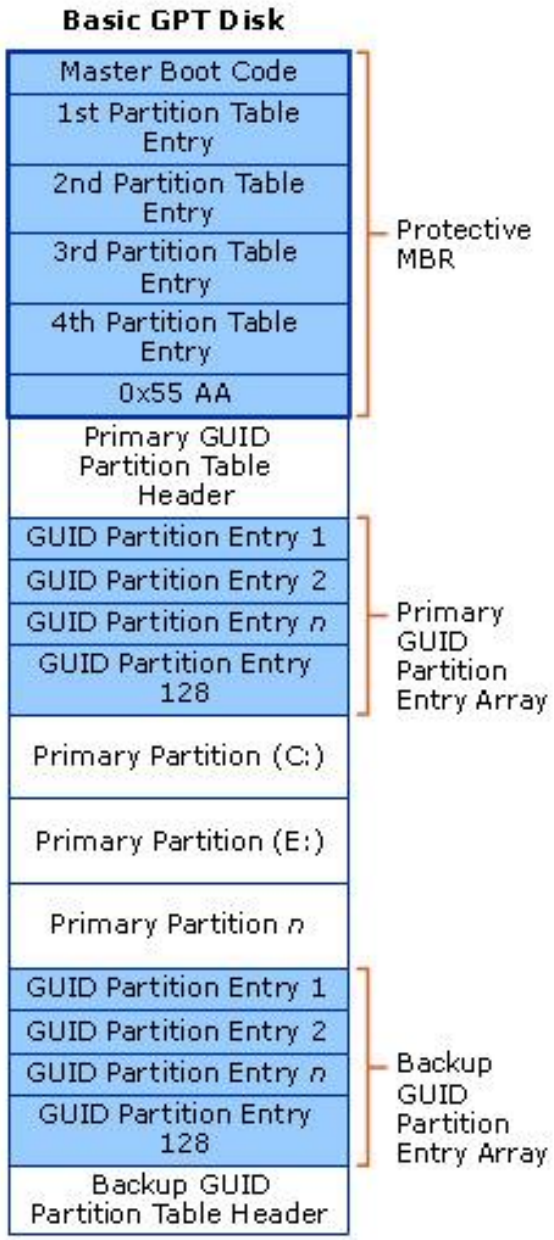

Цилиндр
Зона
Сектор
Дорожка
Пластины
Данные пользователя
Служебная информация

Структура файловой системы

- только 4 первичных раздела
- три первичных раздела и один дополнительный раздел с возможностью создания до 128 логических дисков
- размер диска не более 2,2 Тбайта
- одна копия таблицы разделов



Структура файловой системы

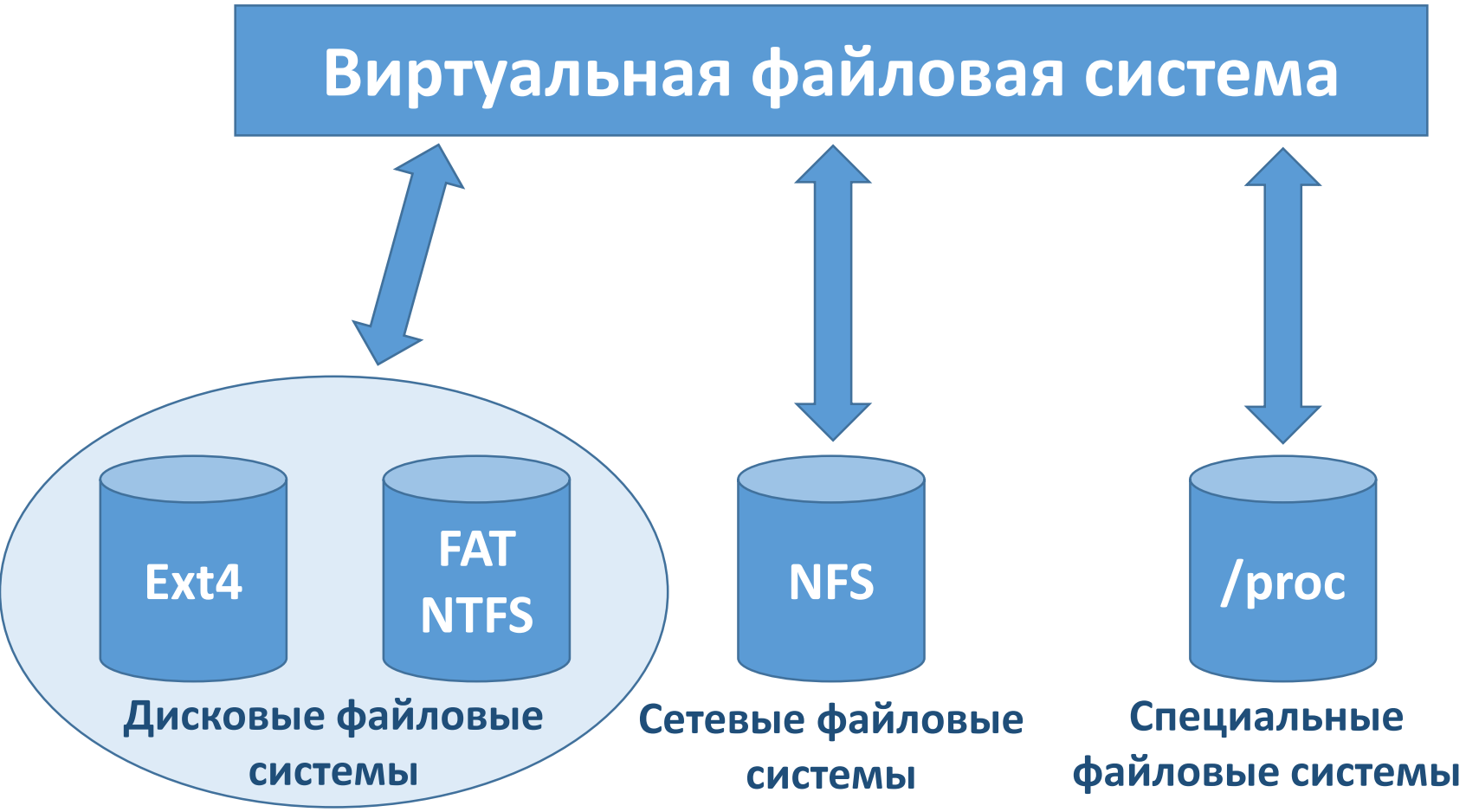


GPT – GUID Partition Table

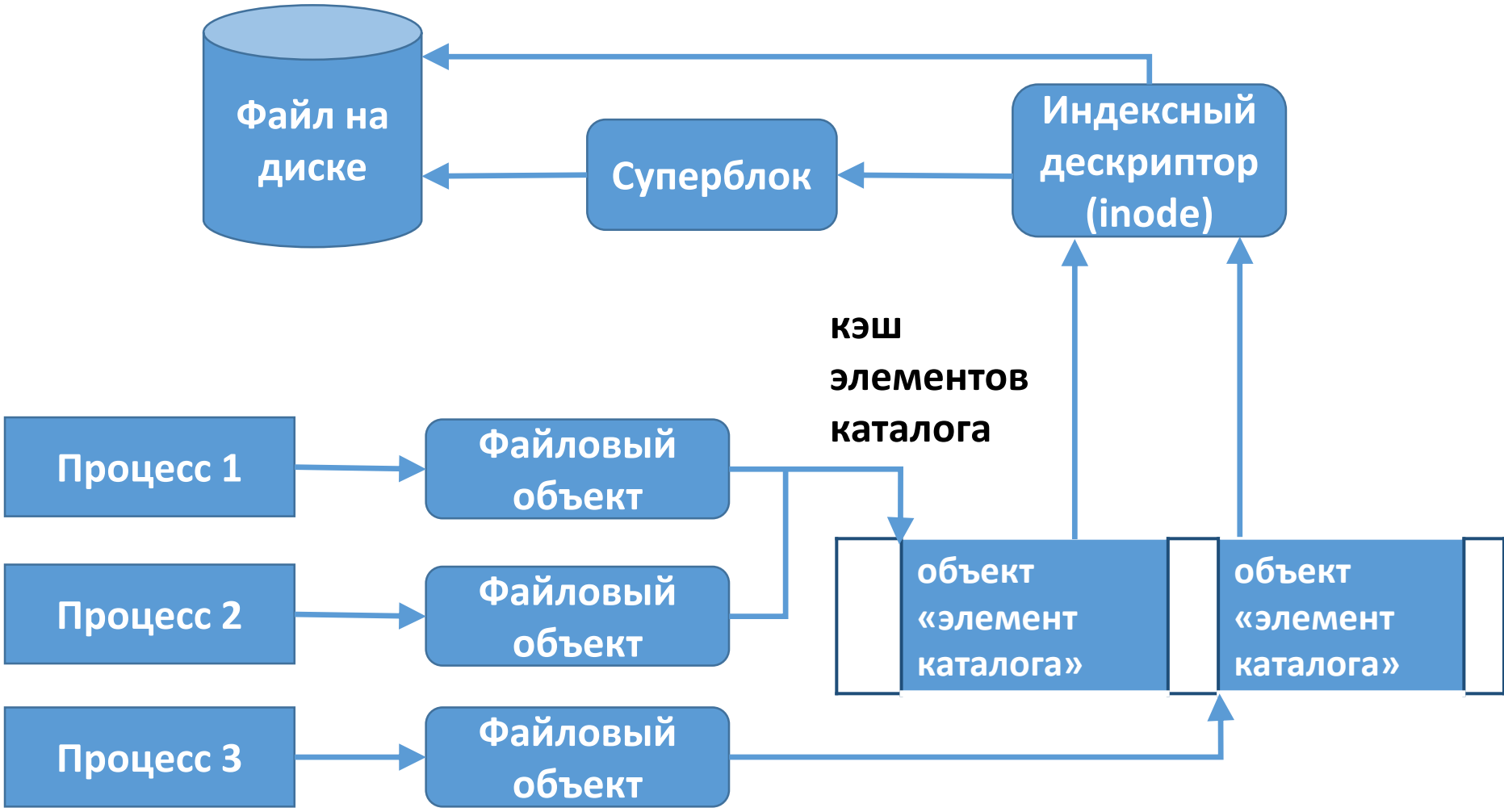
GUID – Globally Unique IDentifier

- 128 первичных разделов
- размер диска до 9,4 Zбайт
- резервная копия таблицы разделов

Виртуальная файловая система (VFS)



Виртуальная файловая система (VFS)



Суперблок

- Суперблок содержит информацию, касающуюся смонтированной файловой системы
- Представлен структурой **`super_block`** в **`linux/fs.h`**
- Все суперблоки объединены в циклический двунаправленный список
- Операции суперблока описаны в структуре **`super_operations`**

Индексный дескриптор (inode)

- Вся информация, необходимая файловой системе для работы с файлом, находится в структуре данных, называемой *индексным дескриптором*
- Индексный дескриптор уникален для каждого файла и остается неизменным, пока существует файл
- Представлен структурой **`inode`** в **`linux/fs.h`**
- Каждый объект "индексный дескриптор" обязательно присутствует в одном из следующих циклических двунаправленных списков:

✓ список допустимых свободных индексных дескрипторов

✓ список допустимых свободных индексных дескрипторов

✓ список "грязных" индексных дескрипторов

Файловый объект

- Файловый объект описывает работу процесса с файлом, который он открыл
- Этот объект создается в момент открытия файла и представлен структурой **`file`** в **`linux/fs.h`**
- Самой важной информацией, хранящейся в объекте, является файловый указатель, текущая позиция в файле, с которой начнется следующая операция чтения/записи
- Используемые файловые объекты собраны в нескольких списках, размещенных в суперблоках файловой системы

Элемент каталога

- Каждый каталог рассматривается как файл, содержащий список файлов и других каталогов
- После того как запись из каталога прочитана в память, VFS преобразует ее в объект "элемент каталога", основанный на структуре **`dentry`** из **`linux/dcache.h`**
- Элемент каталога может находиться в состоянии: «свободен», «не используется», «используется», «отрицательный»

Кэш элементов каталога

- Содержит:
- ✓ набор объектов "элемент каталога", используемых, неиспользуемых или отрицательных
- ✓ хэш-таблица для быстрого нахождения объекта "элемент каталога", связанного с данным именем файла и данным каталогом
- Кэш элементов каталога также служит в качестве управляющего механизма для кэша индексных дескрипторов

Специальные файловые системы

Название	Точка монтирования	Описание
proc	/proc	Общая точка доступа к структурам данных ядра
rootfs	Нет	Предоставляет пустой корневой каталог на этапе загрузки
shm	Нет	Области памяти, совместно используемые при межпроцессорном взаимодействии
sockfs	Нет	Сокеты
sysfs	/sys	Общая точка доступа к системным данным
tmpfs	Любая	Временные файлы (хранятся в оперативной памяти, если не выполняется подкачка)

Регистрация типа файловой системы

- VFS отслеживает все типы файловых систем, код которых включен в ядро
- Каждая зарегистрированная файловая система представлена в виде объекта **`file_system_type`**
- При инициализации системы функция **`register_filesystem()`** вызывается для каждой файловой системы, указанной на этапе компиляции
- Эта функция заносит соответствующий объект **`file_system_type`** в список типов файловых систем