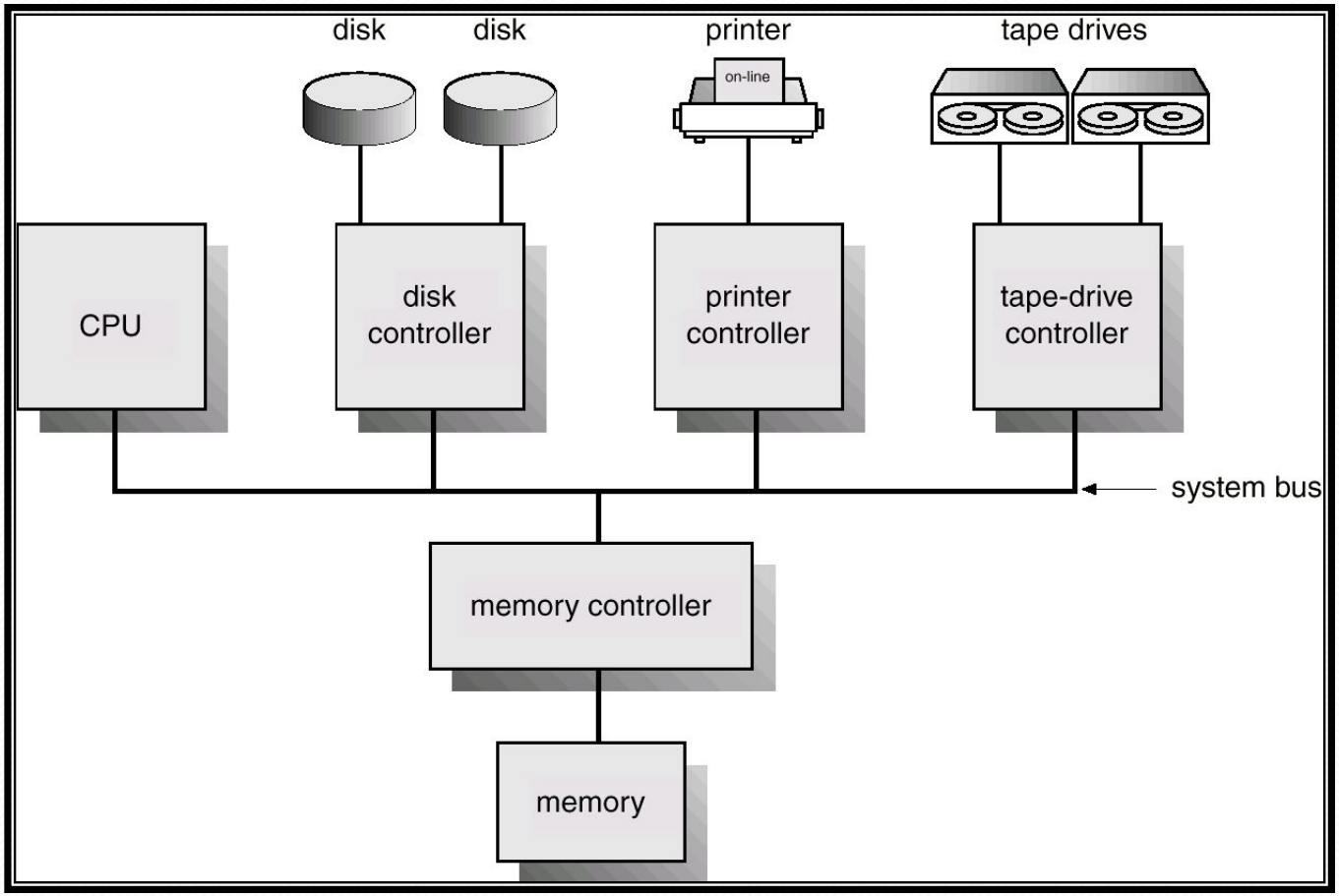
Операционные системы

Архитектура компьютерной системы

Архитектура компьютерных систем



Функционирование компьютерных систем

- **Устройства ввода/вывода и процессор могут функционировать параллельно**
- **Работой каждого устройства управляет специальный контроллер**
- **Каждый контроллер устройства имеет локальный буфер**
- **Процессор перемещает данные из основной памяти в буфера устройства и обратно**
- **Ввод-вывод – перемещение данных из устройства в локальный буфер и обратно.**
- **Контроллер устройства информирует процессор об окончании операции через *прерывание (interrupt)***

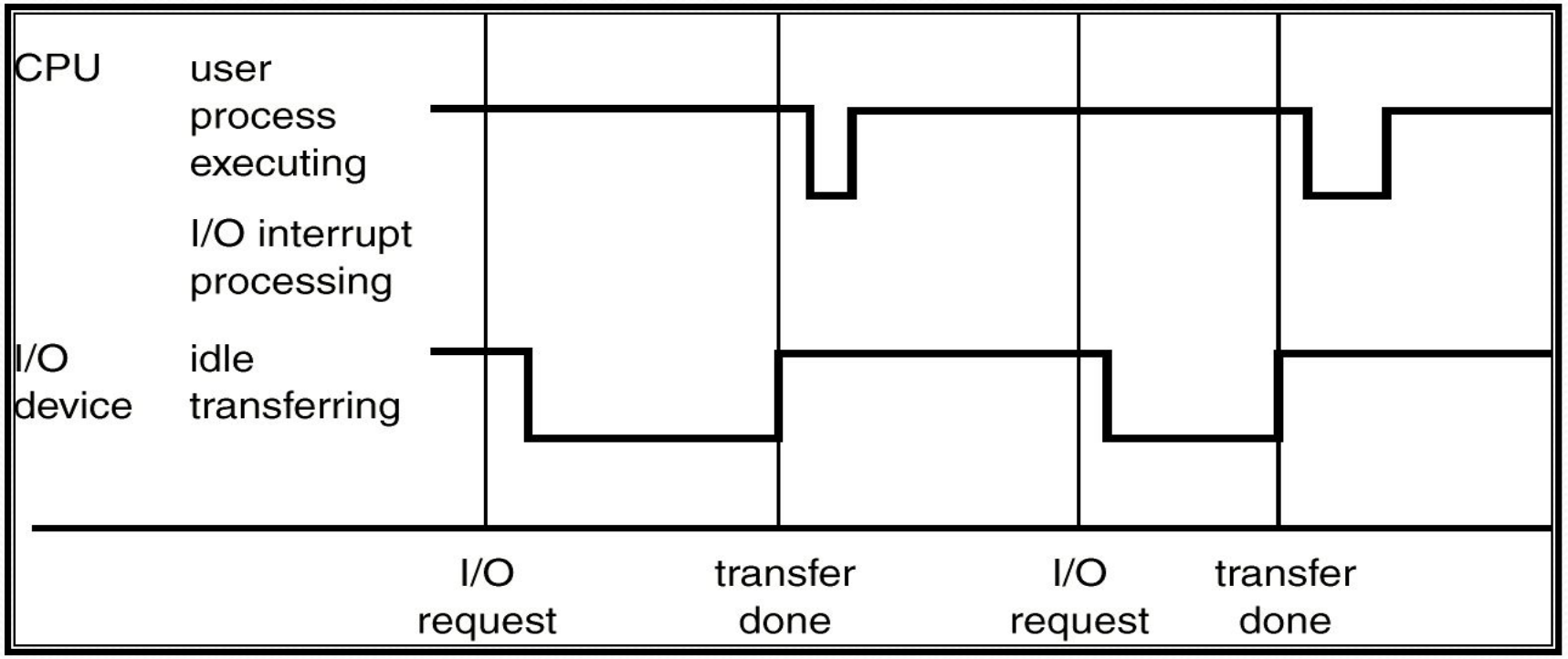
Основные функции прерываний

- **Прерывание передает управление подпрограмме обработке прерываний, как правило, через *вектор прерываний (interrupt vector)*, который содержит адреса всех программ обработки прерываний.**
- **В архитектуре обработки прерываний должно быть предусмотрено сохранение адреса прерванной команды *(instruction)*.**
- **Входящие прерывания задерживаются *(disabled)*, если в данный момент обрабатывается другое прерывание, для предотвращения *потери прерываний (lost interrupt)*.**
- ***Ловушка (trap)* - программно сгенерированное прерывание, либо вызванное ошибкой, либо по запросу пользователя**
- **ОС *управляется прерываниями* (*interrupt driven*).**

Обработка прерываний

- **ОС сохраняет состояние процессора (CPU) – регистры и счетчик команд (program counter – PC)**
- **ОС определяет, какого типа прерывание произошло:**
  - ***Опрос устройств - polling***
  - ***Векторная система прерываний***
- **По содержимому сегмента кода ОС определяет, какого рода действия следует предпринять для соответствующего типа прерывания**

Временной график прерываний процесса, выполняющего вывод



Архитектура ввода/вывода (I/O)

- *Синхронный:* После того, как начинается ввод-вывод, управление возвращается пользовательской программе только после завершения ввода-вывода.
  - Команда ожидания переводит процессор в незагруженный (idle) режим до следующего прерывания
  - Цикл ожидания
  - За один раз обрабатывается не более одного запроса на ввод-вывод; одновременный ввод-вывод отсутствует.
- *Асинхронный:* После того, как начинается ввод-вывод, управление возвращается программе пользователя без ожидания завершения ввода-вывода.
  - *Системный вызов (System call)* – запрос к ОС с целью позволить пользователю ожидать завершения ввода-вывода.
  - *Таблица состояния устройств (Device-status table)* содержит элемент для каждого устройства ввода-вывода, в котором указывается его тип, адрес и состояние.
  - ОС индексирует таблицу устройств с целью определения состояния устройства и модификации элемента таблицы для включения в него информации о прерывании.

Два метода ввода-вывода: синхронный и асинхронный

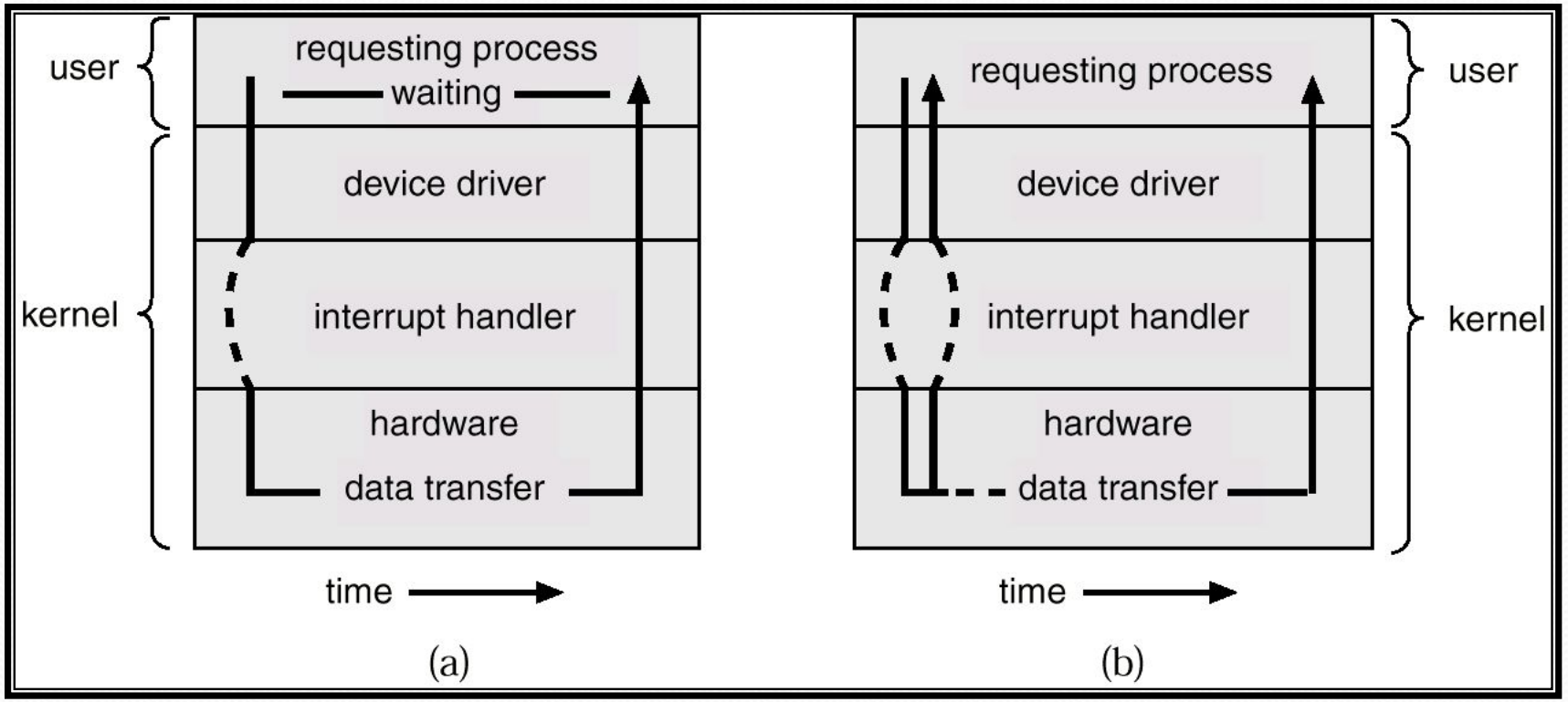
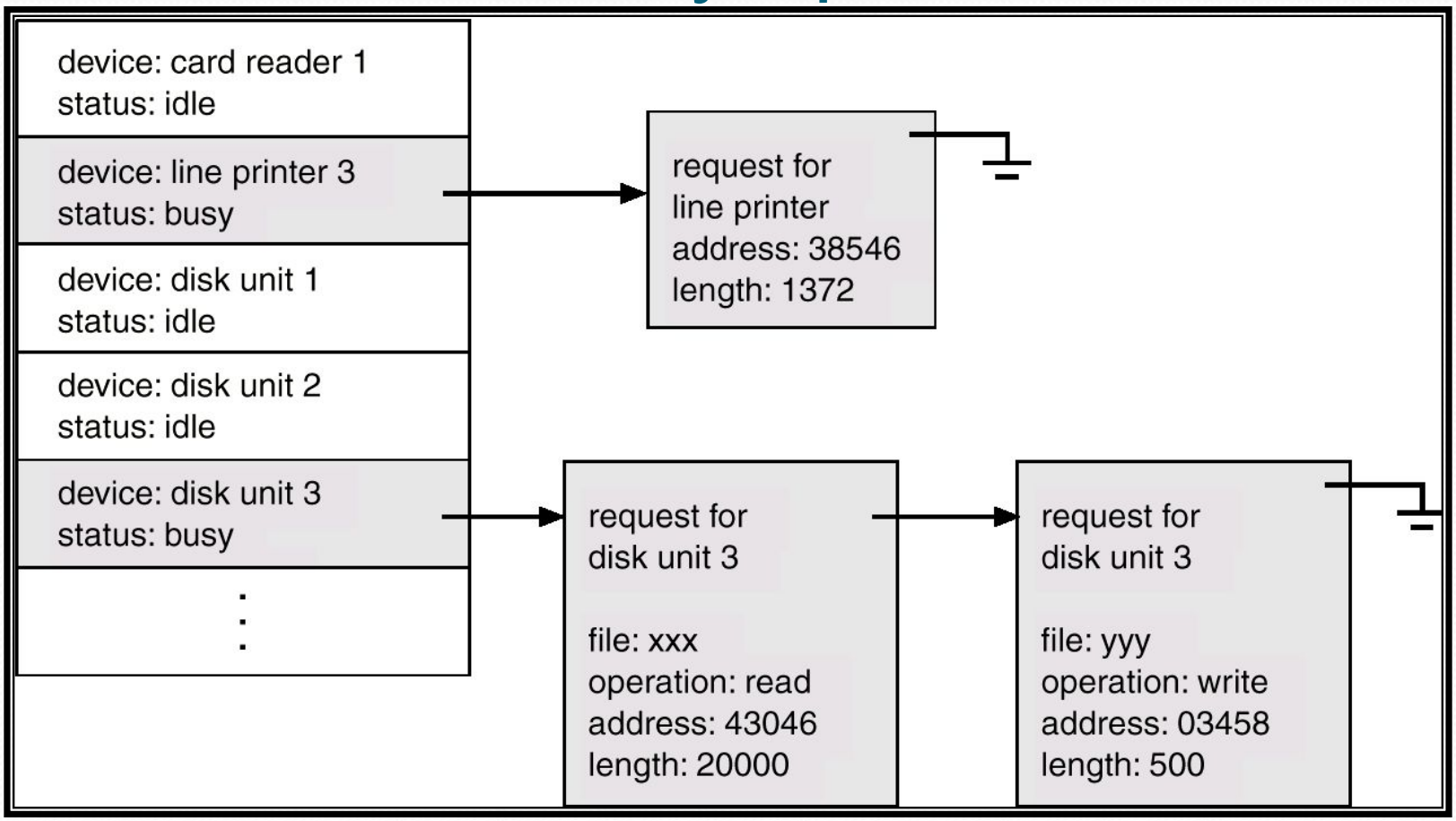


Таблица состояния устройств



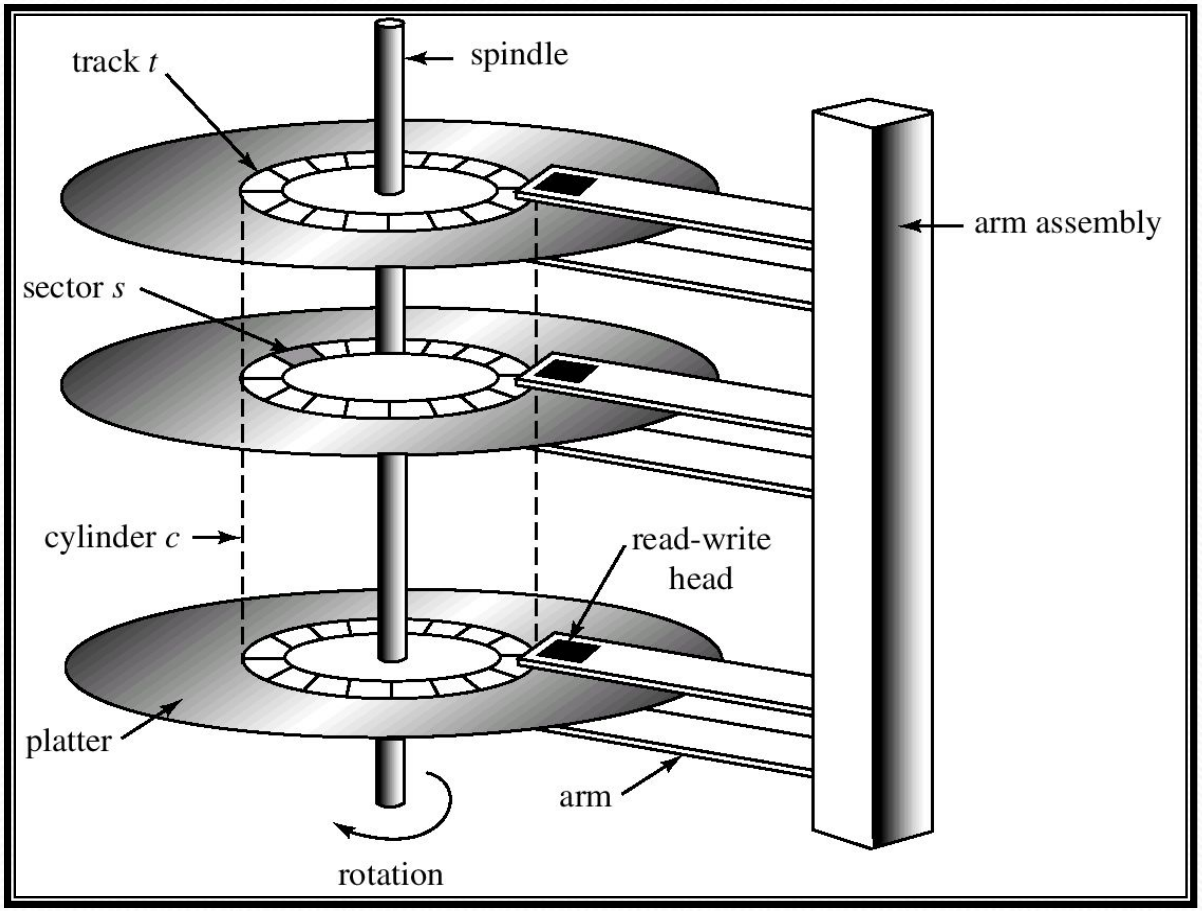
Архитектура прямого доступа к памяти (DMA – Direct Memory Access)

- **Используется для высокоскоростных устройств ввода-вывода, способных передавать информацию со скоростью, близкой к скорости работы памяти.**
- **Контроллер устройства передает блок данных из буферной памяти непосредственно в основную память без участия процессора.**
- **Генерируется только одно прерывание на каждый блок, но не на каждый байт.**

Структура памяти

- **Основная память – единственная большая часть памяти, к которой процессор имеет непосредственный доступ.**
- **Внешняя (вторичная) память – расширение основной памяти , обеспечивающее функциональность устойчивой памяти большого объема**
- **(Магнитные) диски – твердые пластины из металла или стекла, покрытые магнитным слоем для записи**
  - **Поверхность диска логически делится на дорожки (tracks), которые, в свою очередь, делятся на секторы.**
  - **Контроллер диска определяет логику взаимодействия между устройством и компьютером.**

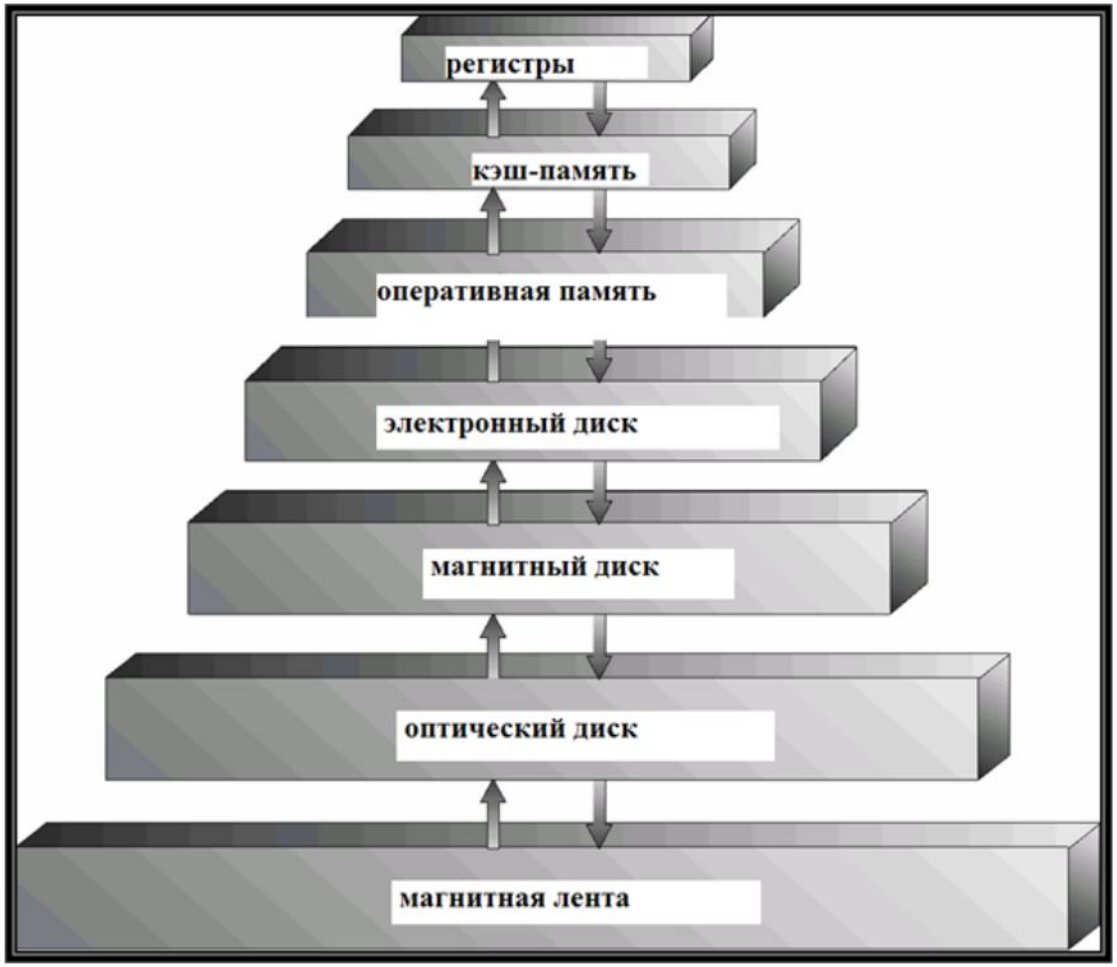
Устройство диска



Иерархия памяти

- **Системы памяти организованы в иерархию по следующим критериям:**
  - **скорость**
  - **цена**
  - **(не)устойчивость (volatility)**
- ***Кеширование (Caching)* – копирование информации в более быструю систему памяти; основная память может рассматриваться как *cache* для внешней памяти.**

Иерархия устройств памяти



Кэширование

- **Использование высокоскоростной памяти для хранения часто используемых (недавно использованных) данных**
- **Требует реализации политики управления кэш-памятью (*cache management*)**
- **Кэширование вводит дополнительный уровень в иерархии памяти. Оно требует согласованности данных, хранимых одновременно на разных уровнях памяти**

Аппаратная защита (hardware protection)

- **Два режима исполнения (Dual-Mode Operation)**
- **Защита ввода-вывода**
- **Защита памяти**
- **Защита процессора**

Два режима исполнения

- **Разделение системных ресурсов требует, чтобы ОС обеспечила невозможность влияния некорректно исполняемой программы на другие программы**
- **Обеспечение аппаратной поддержки для по крайней мере двух режимов исполнения:**
  1. **Пользовательский режим (*User mode*) – для исполнения пользовательских программ**
  2. ***Системный режим (Monitor mode, kernel mode, system mode*) – для исполнения модулей ОС**

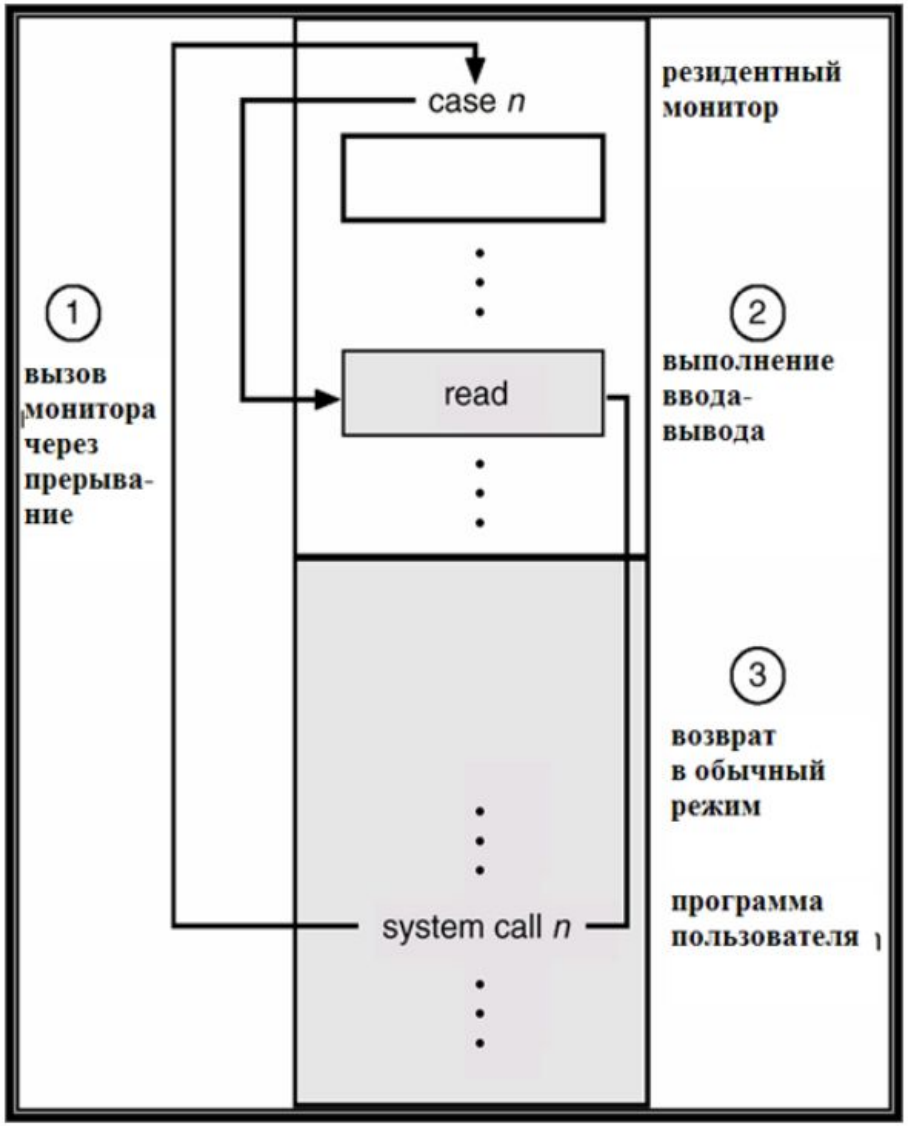
Два режима исполнения (продолжение)

- ***Бит режима (Mode bit)*** **– индикатор режима исполнения: monitor (0); user (1).**
- **При прерывании или сбое аппаратура переключается в системный режим**
- **Привилегированные команды могут исполняться только в системном режиме**

Защита ввода-вывода

- **Все команды ввода-вывода - привилегированные.**
- **Необходимо гарантировать, чтобы пользовательская программа никогда не получила управление в системном режиме (то есть никогда не могла бы записать новый адрес в вектор прерываний)**

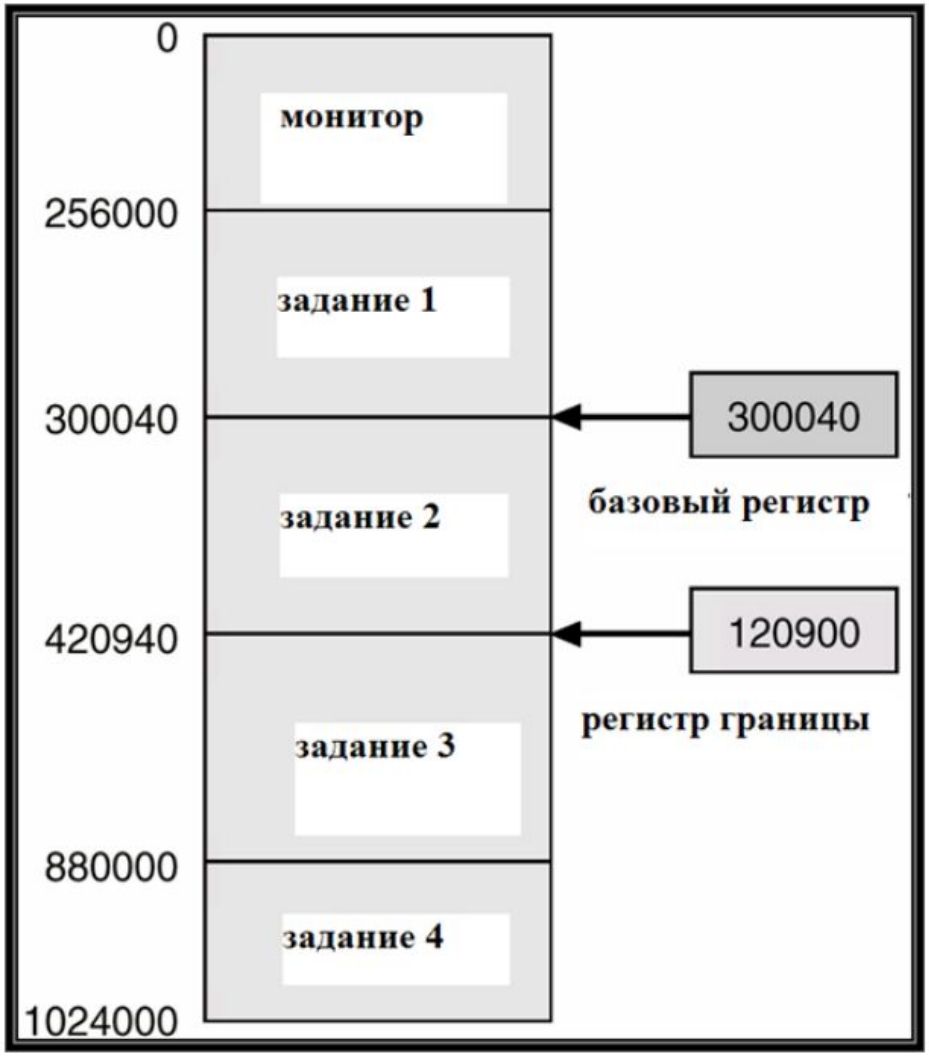
Использование системного вызова для выполнения ввода-вывода



Защита памяти

- **Необходимо обеспечить защиту памяти,**
  по крайней мере для вектора прерываний и подпрограмм обслуживания прерываний
- **Для обеспечения защиты памяти вводятся два регистра, которые отмечают границы допустимой области памяти, выделенной для использования программе**
- **Базовый регистр (Base register) – хранит самый маленький допустимый адрес**
- **Регистр границы (Limit register) – содержит размер диапазона (области)**
- **Память вне отмеченного диапазона защищена**

Использование базового регистра и регистра границы



Аппаратная защита адресов памяти

Аппаратная защита памяти в системах с теговой архитектурой (tagged architecture)

- **МВК “Эльбрус”, Burroughs 6700/7700**
- **Каждое слово памяти имеет *тег* – информацию о типе данных, хранящемся в данном слове**
- **Структура адресного слова (дескриптора – descriptor): тег (“адресная информация”); адрес начала массива; длина массива; биты зашиты**
- **Формирование и изменение дескриптора возможно только средствами ОС в привилегированном режиме**
- **В операции *a[i]* аппаратно проверяется, что индекс *i* не выходит за границы массива *a***

Аппаратная защита

- **При исполнении в привилегированном режиме ОС имеет неограниченный доступ как к памяти монитора, так и к памяти пользователя**
- **Команды записи значений в регистры *base* и *limit* – привилегированные**
- **В системах с теговой архитектурой – только привилегированная команда может сформировать новый дескриптор на область памяти, либо изменить поле в дескрипторе (например, адрес начала или длину)**

Защита процессора

- ***Таймер*** **прерывает процессор через указанный период времени, чтобы убедиться, что ОС сохраняет управление**
  - **Значение таймера уменьшается через каждый квант процессорного времени (clock tick).**
  - **Когда значение таймера становится равным нулю, происходит прерывание.**
- **Таймер обычно используется для реализации режима разделения времени (time sharing).**
- **Таймер используется также для вычисления текущего времени.**
- **Команда записи значения в таймер - привилегированная.**

Операционные системы

Архитектура ОС.

Управление процессами:

Основные понятия.

Семафоры и мониторы

Архитектура ОС

- **Компоненты системы**
- **Сервисы (службы) системы**
- **Системные вызовы**
- **Системные программы**
- **Структура системы**
- **Виртуальные машины**
- **Проектирование и реализация системы**
- **Генерация системы**

Основные компоненты ОС

- **Управление процессами**
- **Управление основной памятью**
- **Управление файлами**
- **Управление системой ввода-вывода**
- **Управление внешней памятью**
- **Поддержка сетей (networking)**
- **Система защиты (protection)**
- **Система поддержки командного интерпретатора (Windows: MS DOS Prompt; UNIX: shells – sh, csh, ksh, bash, etc.)**
- **Графическая оболочка**

Основные компоненты ОС

- **Управление процессами.** Процесс – это программа пользователя в ходе ее выполнения в компьютерной системе.

- **Управление основной памятью.** Основная (оперативная) память может рассматриваться как большой массив. Операционная система распределяет ресурсы памяти между процессами, выделяет память по запросу, освобождает ее при явном запросе или по окончании процесса, хранит списки занятой и свободной памяти в системе.

- **Управление файлами.** Файл – это логическая единица размещения информации на внешнем устройстве, например, на диске. ОС организует работу пользовательских программ с файлами, создает файлы, выполняет их открытие и закрытие и операции над ними (чтение и запись), хранит ссылки на файлы в директориях (папках) и обеспечивает их поиск по символьным именам.

- **Управление системой ввода-вывода.** В компьютерной системе имеется большое число внешних устройств (принтеры, сканеры и др.), управляемых специальными контроллерами (спецпроцессорами) и драйверами – низкоуровневыми программами управления устройствами, выполняемыми в привилегированном режиме. ОС управляет всеми этими аппаратными и программными компонентами, обеспечивая надежность работы внешних устройств, эффективность их использования, диагностику и реконфигурацию в случае их сбоев и отказов.

Основные компоненты ОС

- **Управление внешней памятью.** Внешняя (вторичная) память –это расширение оперативной памяти процессора более медленными, но более емкими и постоянно хранящими информацию видами памяти (диски, ленты и др.). При управлении внешней памятью ОС решает задачи, аналогичные задачам управления основной памятью, - выделение памяти по запросу, освобождение памяти, хранение списков свободной и занятой памяти и др. ОС поддерживает также использование ассоциативной памяти (кэш-памяти) для оптимизации обращения к внешней памяти.

- **Поддержка сетей.** Любая современная компьютерная система постоянно или временно находится в различных локальных и глобальных сетях. Операционная система обеспечивает использование сетевого оборудования (сетевых карт, или адаптеров), вызов соответствующих драйверов, поддержку удаленного взаимодействия с файловыми системами, находящимися на компьютерах сети, удаленный вход на другие компьютеры сети и использование их вычислительных ресурсов, отправку и получение сообщений по сети, защиту от сетевых атак.

- **Система защиты.** Согласно современным принципам надежных и безопасных вычислений (см. "Понятие операционной системы (ОС), цели ее работы. Классификация компьютерных систем"), при работе ОС должны быть обеспечены надежность и безопасность, т.е. защита от внешних атак, конфиденциальность личной и корпоративной информации, диагностика и исправления ошибок и неисправностей и др. ОС обеспечивает защиту компонент компьютерной системы, данных и программ, поддерживает фильтрацию сетевых пакетов, обнаружение и предотвращение внешних атак, хранит информацию обо всех действиях над системными структурами, полезную для анализа атак и борьбы с ними.

Основные компоненты ОС

- **Система поддержки командного интерпретатора.** Любая операционная систем а поддерживает командный язык (или набор командных языков), состоящих из пользовательских команд, выполняемых с пользовательского терминала (из пользовательской консоли). Типичные команды – это получение информации об окружении, установка текущей рабочей директории, пересылка файлов, компиляция и выполнение программ и др. В Windows для выполнения команд используется окно пользовательской консоли MS DOS (MS DOS Prompt),
в системе Linux – специальное окно "Терминал". Наиболее мощные командные процессоры имеются в системах типа UNIX (UNIX, Solaris, Linux и др.). В Windows сравнительно недавно появился мощный командный интерпретатор PowerShell. Кроме того, для Windows имеется система CygWin, позволяющая выполнять команды и командные файлы UNIX в среде Windows.

- **Графическая оболочка** – подсистема ОС, реализующая графический пользовательский интерфейс пользователей и системных администраторов с операционной системой. Использование одного лишь командного языка и системных вызовов неудобно, поэтому простой и наглядный графический пользовательский интерфейс с ОС необходим. Имеется много известных графических оболочек для операционных систем, причем их возможности очень похожи друг на друга - настолько, что подчас не вполне понятно, какая именно ОС используется. Среди графических оболочек, используемых в системах типа UNIX, можно назвать CDE, KDE, GNOME. ОС Windows и MacOS имеют собственные, весьма удобные графические оболочки.

Управление процессами

- ***Процесс*** **(*process*) - это программа при ее исполнении. Для процесса требуется ряд *ресурсов*, включая время процессора, память, файлы, устройства ввода-вывода, сетевые устройства и др.**
- **Обычно при создании процесса для него создается новое пространство виртуальной памяти (*но*: для lightweight process – создается только стек)**
- **ОС отвечает за следующие действия, связанные с управлением процессами:**
  - **Создание и удаление процессов.**
  - **Приостановка и возобновление процессов.**
  - **Обеспечение механизмов для:**
    - **Синхронизации процессов (семафоры, мониторы и др.)**
    - **Взаимодействия процессов (условные переменные, события, рандеву и др.)**

Управление процессами: семафоры (E.W. Dijkstra, 1966)

- **Двоичный семафор – переменная S, которая может находиться в двух состояниях: “открыт” и “закрыт”**
- **Операции над S - “семафорные скобки”:**
  P(S) – закрыть, V(S) – открыть
- **При попытке закрыть уже закрытый семафор происходит прерывание, и ОС добавляет текущий процесс в очередь к закрытому семафору**
- **Операция V(S) активизирует первый стоящий в очереди к S процесс, который успешно завершает операцию P(S)**
- **Синхронизация по ресурсам: *P(S); critical_section; V(S);***
- **Операции P и V – *атомарны (atomic)* для других процессов**

Управление процессами: мониторы (Sir Tony Hoare, 1974)

- **Монитор – многовходовый модуль M, в котором определены общие для процесов данные D и операции P1, … P_N над этими данными (в виде процедур)**
- **В каждый момент не более чем один из параллельных процессов может вызвать какую-либо из операций**
- **Вызов каждой операции монитора – атомарен (как и операции над семафором)**
- **Монитор – еще один механизм синхронизации процессов по ресурсам**
- **Мониторы включены Ч. Хоаром в разработанный им язык Concurrent Pascal (Pascal + конструкция “monitor”) для параллельного программирования и разработки ОС**

Операционные системы

Обзор функций ОС.
Уровни абстракции ОС.
Архитектура UNIX и MS-DOS

Управление основной памятью

- **Память – большой массив слов или байтов *(big endian / little endian),* каждый из которых имеет свой адрес. Это хранилище (repository) данных с быстрым доступом, разделяемое процессором и устройствами ввода-вывода.**
- **Основная память – это неустойчивое (volatile) устройство памяти. Ее содержимое теряется при сбое системы**
- **ОС отвечает за следующие действия, связанные с управлением памятью:**
  - **Отслеживание того, какие части памяти в данный момент используются и какими процессами.**
  - **Стратегия загрузки процессов в основную память, по мере ее освобождения.**
  - **Выделение и освобождение памяти по мере необходимости.**

Управление файлами

- ***Файл (file)*** **– совокупность взаимосвязанной информации, задаваемой его создателем. Как правило, файлы представляют программы (в виде исходного текста или в двоичной форме) или данные.**
- **Другой термин для файла –** ***набор данных (data set)*** **– IBM 360/370**
- **ОС отвечает за следующие действия, связанные с управлением файлами:**
  - **Создание и удаление файлов.**
  - **Создание и удаление директорий.**
  - **Поддержка примитивов (пользовательских команд, APIs) для управления файлами и директориями.**
  - **Отображение файлов на внешнюю память.**
  - **Сброс (*backup*) файлов на устойчивые носители (стример, flash и др.)**
  - **В некоторых ОС реализованы файловые системы с *криптованием* данных при записи в файл**

Управление вторичной памятью

- **Поскольку размер основной памяти недостаточен для постоянного хранения всех программ и данных, в компьютерной системе должна быть предусмотрена вторичная (внешняя) память для сброса, откачки (back up, swapping) части содержамого основной памяти.**
- **В большинстве компьютерных систем в качестве главной вторичной памяти для хранения программ и данных используются диски**
- **ОС отвечает за выполнение следующих действий, связанных с управлением дисками:**
  - **Управление свободной дисковой памятью**
  - **Выделение дисковой памяти**
  - **Диспетчеризация дисков (disk scheduling)**

Сети (распределенные системы)

- **Распределенная система – это совокупность процессоров, которые не используют общую память или часы (такты процессора). Каждый процессор имеет собственную локальную память.**
- **Процессоры в системе соединены в сеть.**
- **Сетевое взаимодействие выполняется по определенному *протоколу* (интерфейсу, набору операций). Наиболее распространенный сетевой протокол – TCP/IP, основанный на IP-адресах машин (hosts); например, 190.100.125.1**
- **Распределенная система обеспечивает доступ пользователей к различным общим сетевым *ресурсам* (файлам, принтерам и т.д.) и удаленный запуск программ (rsh, RPC, RMI, etc.)**
- **Доступ к общему ресурсу (shared resource) позволяет:**
  - Ускорить вычисления
  - Расширить границы доступа к данным
  - Обеспечить более высокую надежность

Система защиты (protection)

- **Термин *защита (protection)* используется для механизма управления доступом программ, процессов и пользователей к системным и пользовательским ресурсам.**
- **Механизм защиты должен:**
  - **Различать авторизованный (санкционированный - authorized) и несанкционированный (unauthorized) доступ.**
  - **Описывать предназначенные для защиты элементы управления (конфигурации).**
  - **Обеспечивать средства выполнения необходимых для защиты действий (сигналы, исключения, блокировка и др.).**

Система поддержки командного интерпретатора

- **Большинство команд для ОС задаются с помощью специальных управляющих операторов, предназначенных для**
  - **создания процессов и управления процессами**
  - **выполнения ввода-вывода**
  - **управления вторичной памятью**
  - **управления основной памятью**
  - **доступа к файловой системе**
  - **защиты**
  - **управления сетью**

Система поддержки командного интерпретатора (продолжение)

- **Программа, которая читает и интерпретирует операторы управления, называется:**
  - ✔ **командным интерпретатором (Windows / MS-DOS prompt: command.com)**
  - ✔ **shell (UNIX, Linux: Start/System tools/Terminal)**

Ее функция состоит в том, чтобы прочесть и исполнить очередной управляющий оператор (команду).

Сервисы (службы) ОС

- ***Исполнение программ*** **– загрузка программы в память и ее исполнение (Windows – execution stub; .NET – execution stub для вызова CLR).**
- ***Поддержка ввода-вывода*** **– обеспечение интерфейса для работы программ с устройствами ввода-вывода.**
- ***Работа с файловой системой*** **– предоставление программам интерфейса для создания, именования, удаления файлов.**
- ***Коммуникация*** **– обмен информацией между процессами, выполняемыми на одном компьютере или на других системах, связанных в сеть. Реализуется с помощью общей памяти (*shared memory*) или передачи сообщений.**
- ***Обнаружение ошибок*** **в работе процессора, памяти, устройств ввода-вывода и программах пользователей.**

Дополнительные функции ОС

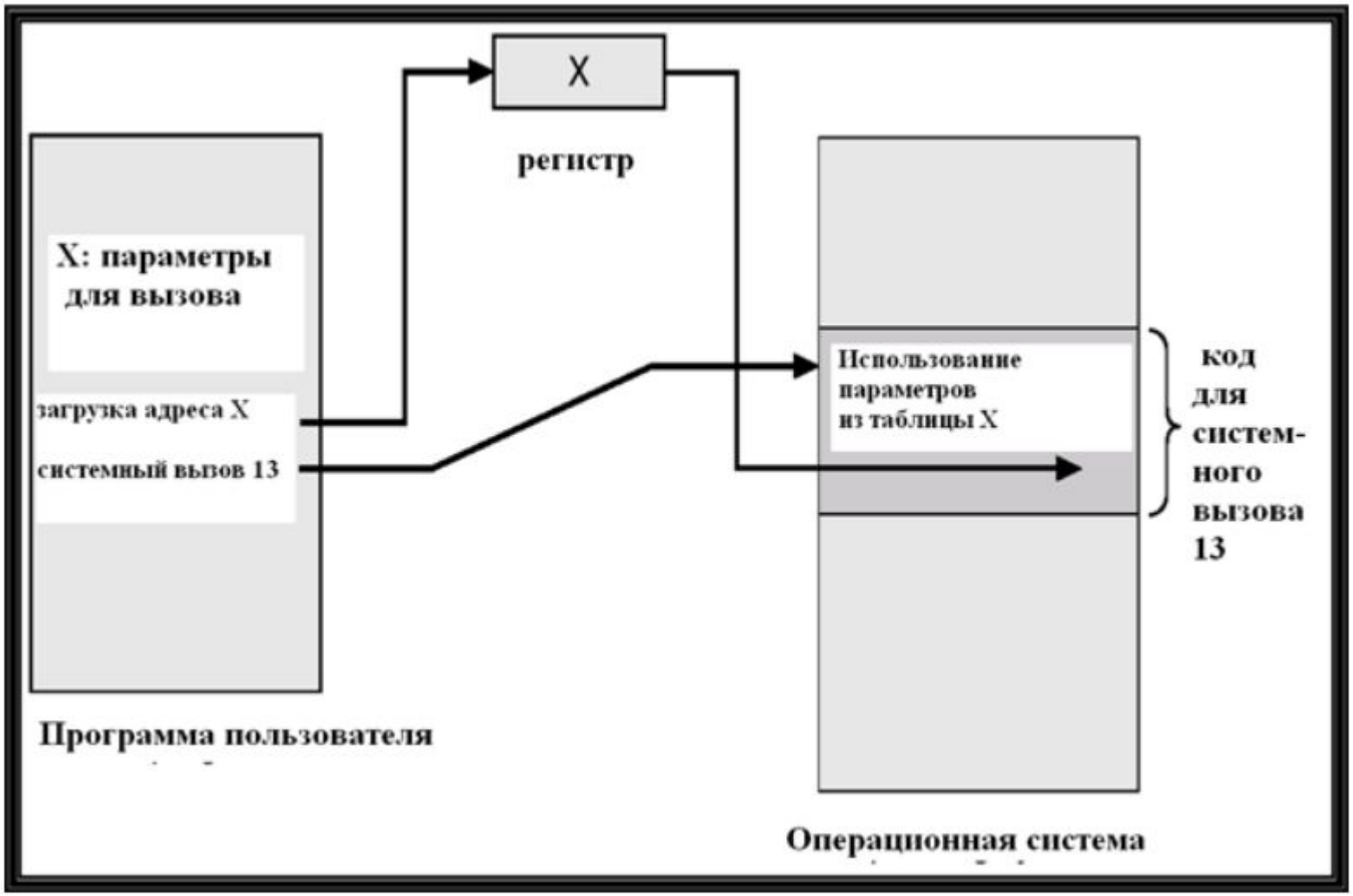
Реализованы не непосредственно для удобства пользователя, а для обеспечения выполнения операций системы.

- ***Распределение ресурсов*** **между пользователями, программами и процессами, работающими одновременно.**
- **Ведение *статистики* использования ресурсов, с целью выставления пользователям счетов (например, за сетевой трафик) или для анализа эффективности работы системы.**
- ***Защита*** **– обеспечение того, чтобы доступ к любым ресурсам был контролируемым.**

Системные вызовы

- **Системные вызовы являются интерфейсом между выполняемой программой и ОС.**
  - **Обычно доступны как специальные ассемблерные команды.**
  - **Некоторые языки (C, C++ и др.) позволяют выполнять системные вызовы непосредственно**
- **Используются три основных способа передачи параметров исполняемой программой операционной системе:**
  - **Передача параметров в регистрах**
  - **Запись параметров в таблицу, расположенную в памяти, и передача адреса этой таблицы в регистре.**
  - **Запись (проталкивание) параметров в стек программой и чтение (выталкивание) их из стека ОС.**

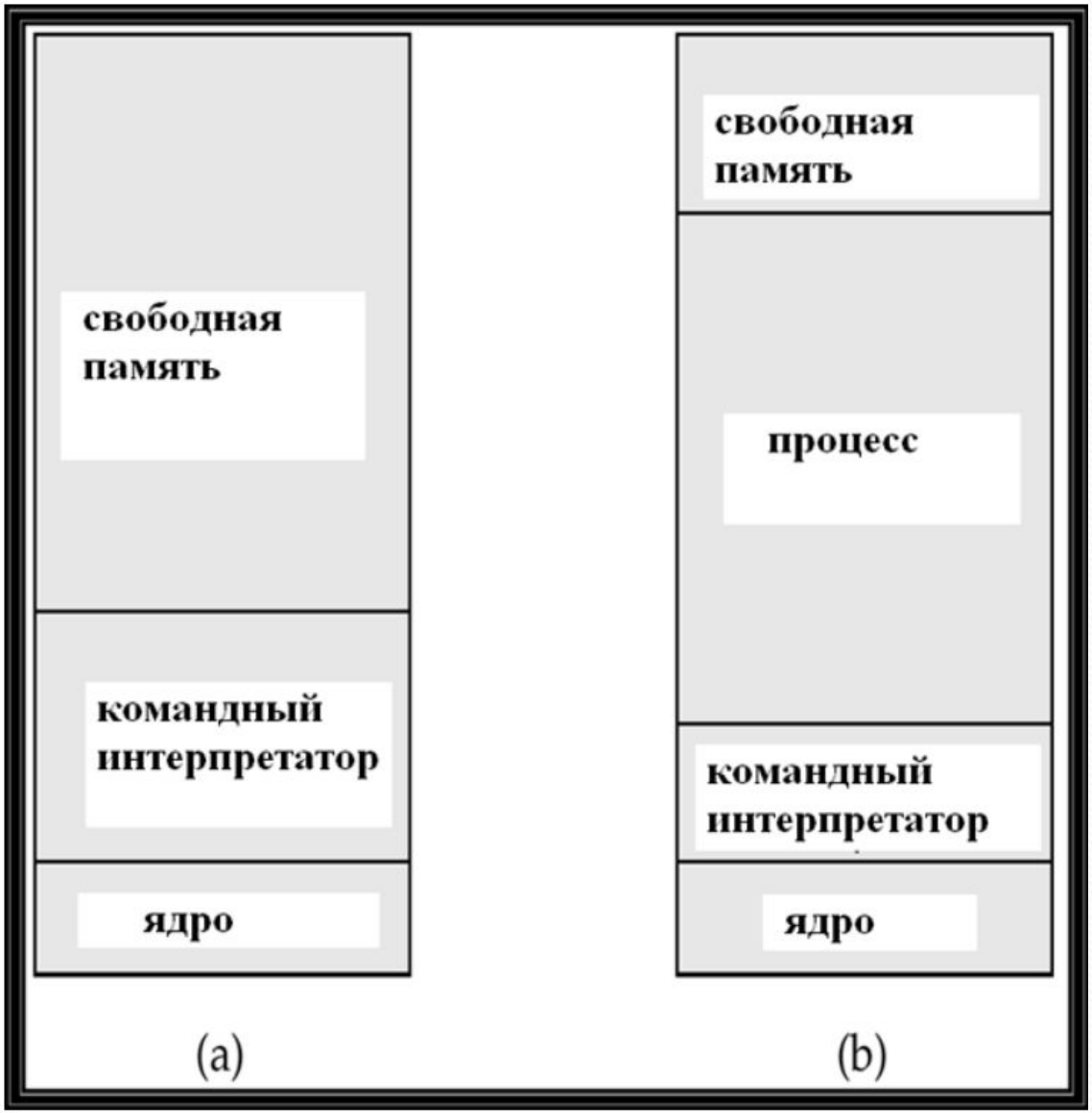
Передача параметров в таблице



Виды системных вызовов

- **Управление процессами**
- **Управление файлами**
- **Управление устройствами**
- **Сопровождение информации**
- **Коммуникации**

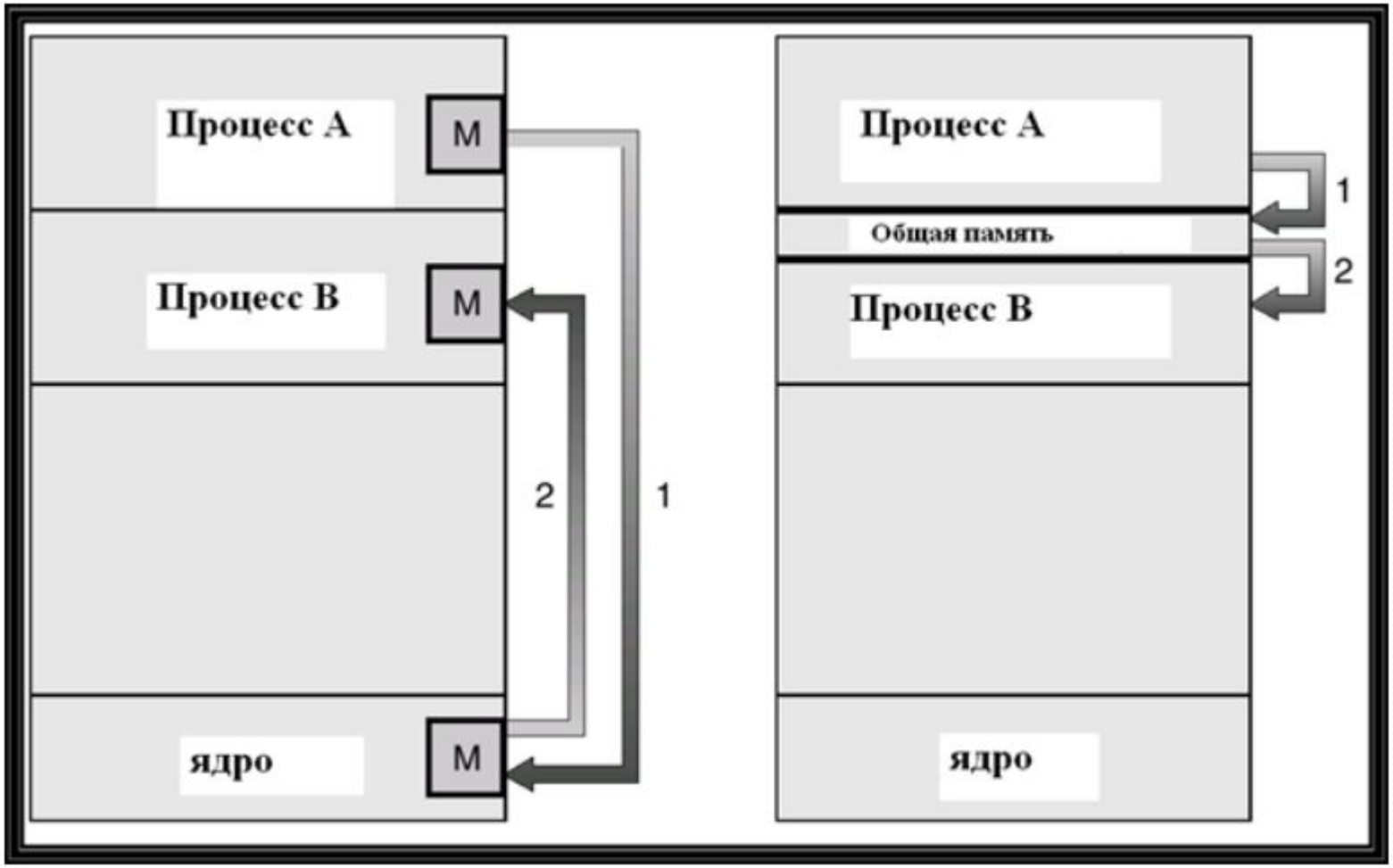
Исполнение программ в MS-DOS



Исполнение нескольких программ в UNIX

Коммуникационные модели

- **Могут реализовываться с помощью общей памяти или передачи сообщений**



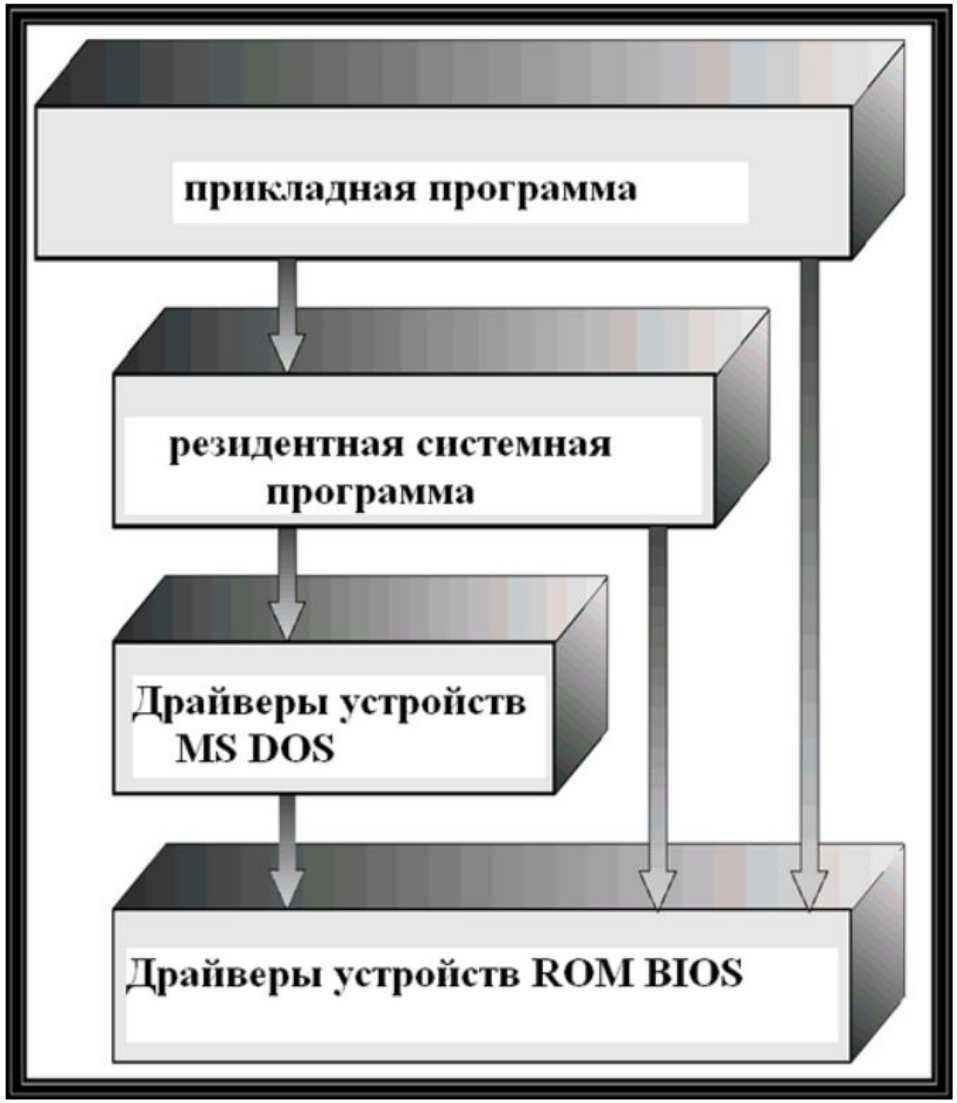
Системные программы

- **Системные программы обеспечивают удобное окружение для разработки и исполнения программ. Они подразделяются на программы:**
  - **Управления файлами**
  - **Получения информации о состоянии**
  - **Изменения файлов**
  - **Поддержки языков программирования**
  - **Загрузки и исполнения программ**
  - **Коммуникации**
- **Использование ОС большинством пользователей основано на использовании системных программ, а не системных вызовов.**

Структура системы MS-DOS

- **MS-DOS – разработана по принципу: обеспечить максимум функциональности, используя минимум памяти (640 К – ограничение на объем памяти для программы в MS-DOS)**
  - **Нет явного разделения на модули**
  - **Хотя MS-DOS и имеет некоторую архитектуру, но уровни функциональности и интерфейсы в ней не отделены четко друг от друга**

Уровни (абстракции) модулей MS-DOS



Структура системы UNIX

- **UNIX – ограничена функциональностью аппаратуры. Первоначальные версии UNIX имели ограниченное структурирование.**
- **Система UNIX состоит из двух частей:**
  - **Системные программы**
  - **Ядро**
    - **Содержит все модули, уровень абстракции которых ниже системных вызовов, но выше непосредственно аппаратных модулей**
    - **Поддержка файловой системы, диспетчеризация процессора, управление памятью и другие функции ОС**

Структура системы UNIX

Подход к созданию ОС на основе уровней абстракции (Э. Дейкстра, операционная система THE, 1968)

- **ОС реализуется в виде набора (иерархии) *уровней абстракции (abstraction layers)*, каждый из которых реализован на основе предыдущего уровня . Уровень 0 (layer 0) - аппаратура (hardware); самый высокий уровень (layer N) - пользовательский интерфейс с ОС**
- **По принципам модульного программирования, при реализации каждого уровня используются только модули предшествующего уровня**
- **“Перескакивание” через уровнь не рекомендуется**