Операционные системы

Введение в операционные системы

Введение в операционные системы

Классификация операционных систем

Признаки классификации

- ОС могут различаться особенностями реализации внутренних алгоритмов управления основными ресурсами компьютера, особенностями использованных методов проектирования, типами аппаратных платформ, областями использования и многими другими свойствами.
- Рассмотрим подробнее классификацию ОС по нескольким наиболее основным признакам:
  - особенности алгоритмов управления ресурсами;
  - область использования;
  - особенности поддерживаемых аппаратных платформ;
  - структурная организация.

Классификация операционных систем

Особенности алгоритмов управления ресурсами

Поддержка многозадачности

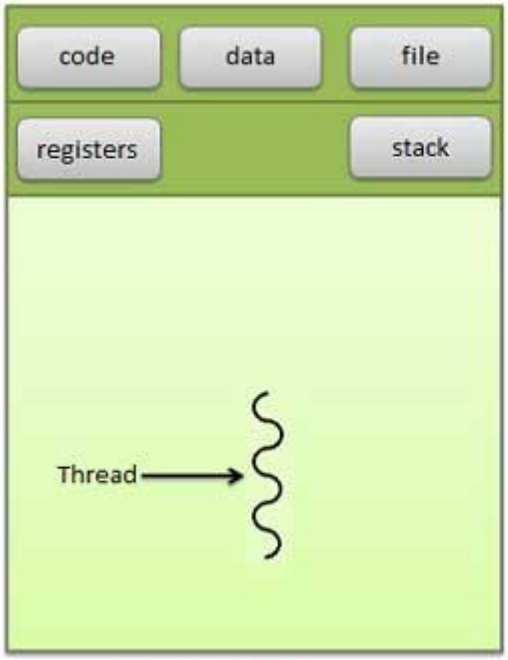
- Однозадачные ОС в основном выполняют функцию предоставления пользователю виртуальной машины.
- Пример: MS-DOS, MSX.
- Многозадачные ОС поддерживают в том или ином виде мультипрограммирование и управляют разделением ресурсов вычислительной системы.
- Пример: Windows, Unix.

Вытесняющая и невытесняющая многозадачность

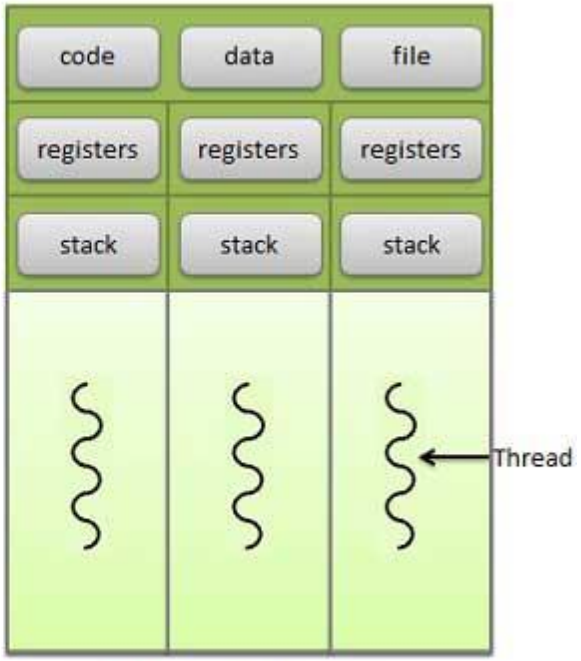
- При невытесняющей многозадачности (nonpreemptive) активная задача выполняется до тех пор, пока она сама, по собственной инициативе, не отдаст управление ОС для того, чтобы та выбрала из очереди другую задачу готовую к выполнению.
- Пример: Novell NetWare, Windows 3.x.
- При вытесняющей многозадачности (preemptive) решение о переключении процессора с одной задачи на другую принимается ОС.
- Пример: Windows 95 и выше, Unix.

Реализация многозадачности

- Многозадачные операционные системы могут быть ограничены одним потоком в рамках каждого процесса или могут поддерживать многопоточные процессы.



многозадачность на уровне процессов

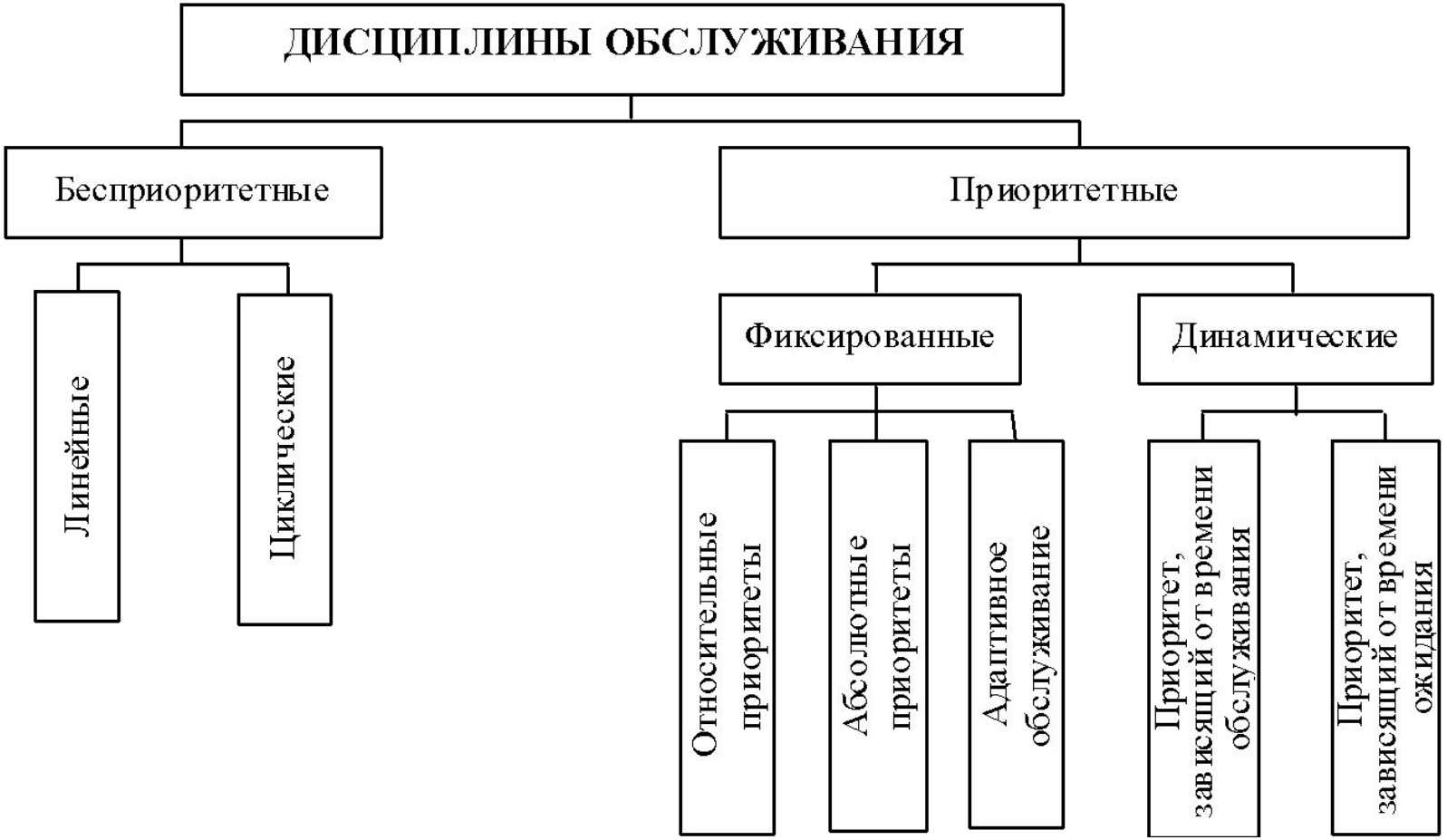


многозадачность на уровне потоков (многопоточность)

Дисциплины обслуживания

- Дисциплина обслуживания – это способ определения того, какое требование в очереди готовых задач (процессов, потоков) должно обслуживаться следующим.
- Решение может основываться на одной из приведенных ниже характеристик или на их совокупности:
  - мера, определяемая относительным временем поступления рассматриваемого требования в очередь;
  - мера требуемого или полученного до сих пор времени обслуживания;
  - функция, определяющая принадлежность задачи к той или иной группе.

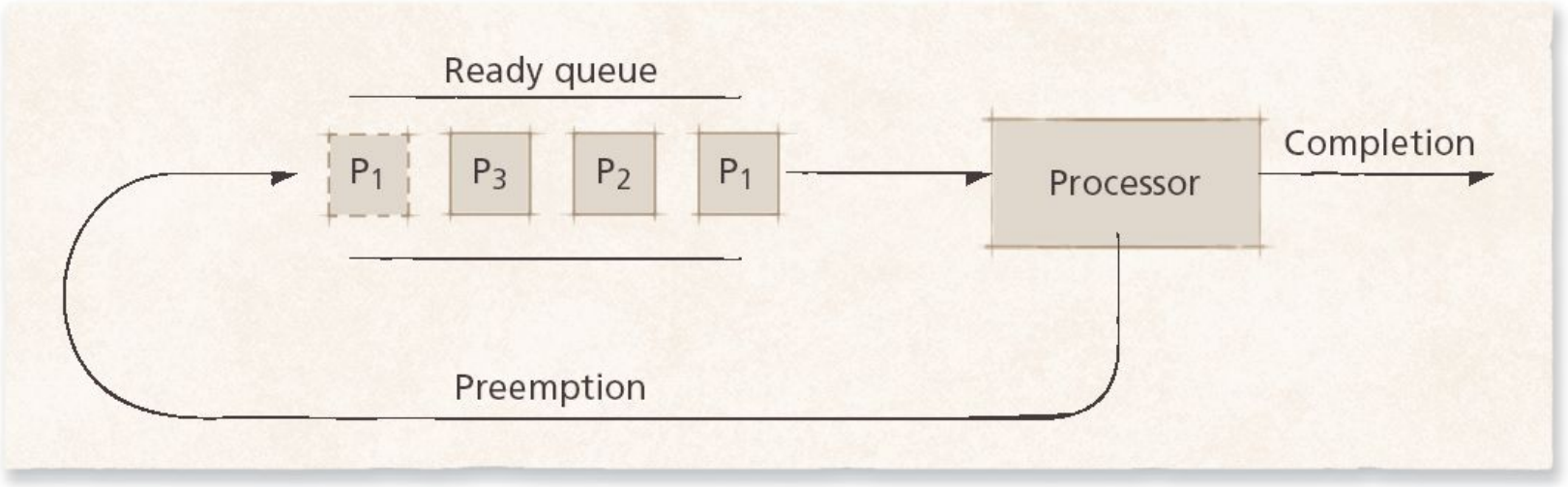
Классификация дисциплин обслуживания



Виды дисциплин обслуживания

- *Бесприоритетные дисциплины* – выбор из очереди производится без учета относительной важности задач и времени их обслуживания.
- *Приоритетное обслуживание* – отдельным задачам предоставляется преимущественное право перейти в состояние ВЫПОЛНЕНИЯ.
- *Фиксированные приоритеты* – являются величиной постоянной на всем жизненном цикле процесса.
- *Динамические приоритеты* – изменяются в зависимости от некоторых условий в соответствии с определенными правилами. Для реализации динамических приоритетов необходимы дополнительные затраты, но их использование предполагает более справедливое распределение процессорного времени между процессами.

Пример бесприоритетной циклической дисциплины



Приоритетное обслуживание

- При данной стратегии с каждым процессом связывается его приоритет (целое число). Процессор выделяется процессу с наивысшим приоритетом (обычно большее число означает более высокий приоритет процесса).
- Приоритетное обслуживание может использовать относительные (без прерывания) и абсолютные приоритеты (с прерыванием).
- Приоритет может быть динамический и фиксированный.
- Часто процессы объединяют по приоритетам в группы, и используют приоритетное планирование среди групп, но внутри группы используют циклическое планирование.

Динамический приоритет

- Динамический приоритет может устанавливаться так:
  - П=1/Т, где Т- часть использованного в последний раз кванта.
  - Если использовано 1/50 кванта, то приоритет 50.
  - Если использован весь квант, то приоритет 1.
- Таким образом, процессы, активно пользующиеся вводом-выводом, будут иметь приоритет над вычислительными процессами.

Поддержка многопользовательского режима

- По числу «одновременно» работающих пользователей ОС делятся на:
  - однопользовательские (MS-DOS, Windows 3.x, Windows 9x);
  - многопользовательские (UNIX, Windows 2000 и выше).
- Главным отличием многопользовательских систем от однопользовательских является **наличие средств защиты информации** каждого пользователя от несанкционированного доступа других пользователей.
- Следует заметить, что не всякая многозадачная система является многопользовательской, и не всякая однопользовательская ОС является однозадачной.

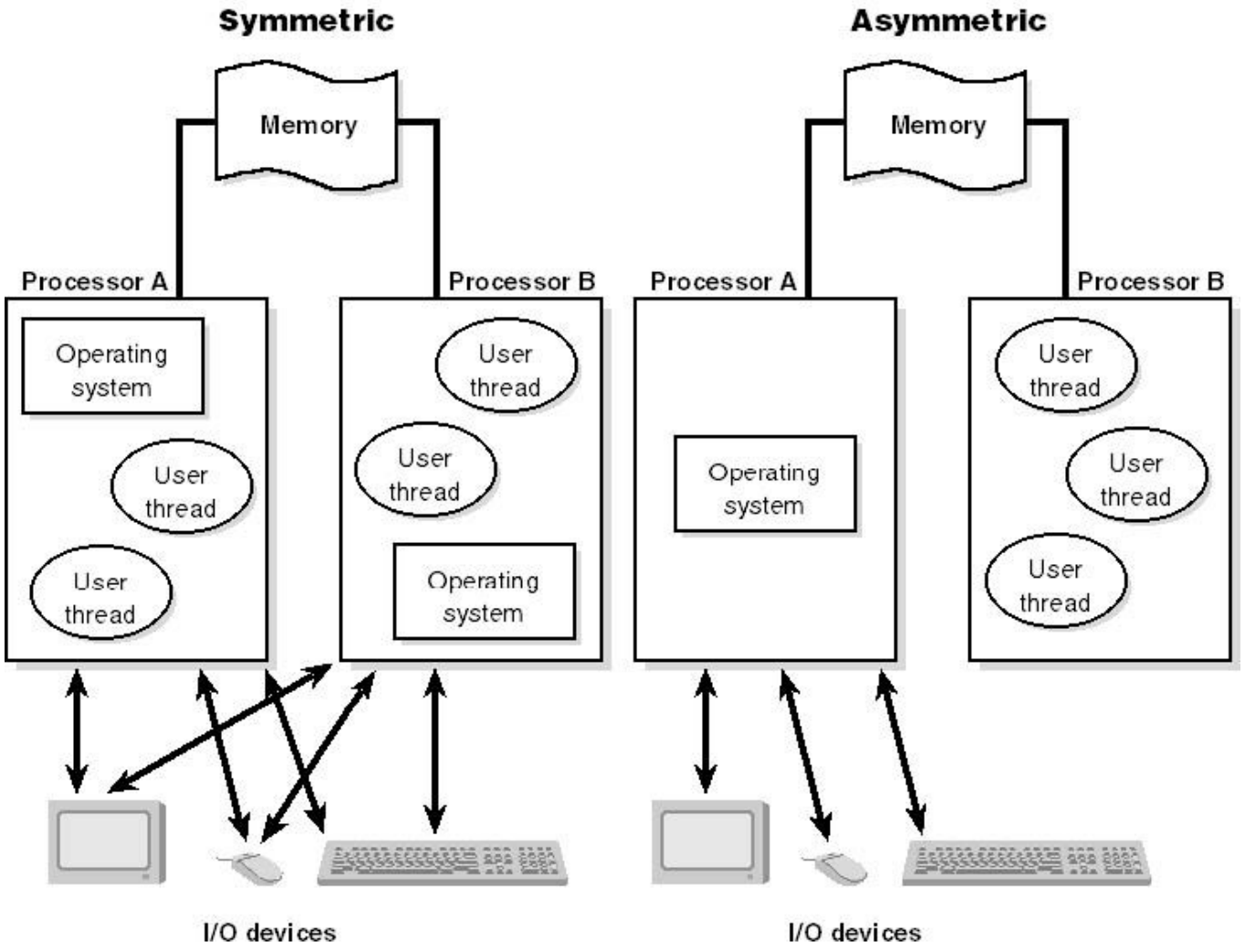
Поддержка многопроцессорной обработки

- Современные операционные системы должны поддерживать многопроцессорные вычислительные системы.
- Поддержка многопроцессорной обработки может приводить к существенному усложнению всех алгоритмов управления ресурсами.
- Многопроцессорные ОС могут классифицироваться по способу организации вычислительного процесса: асимметричные ОС и симметричные ОС.

Виды мультипроцессирования

- Асимметричная ОС целиком выполняется только на одном из процессоров системы, распределяя приложения по остальным процессорам.
- Симметричная ОС полностью децентрализована и использует весь пул процессоров для выполнения как прикладных задач, так и системных.
- Сравнивая асимметричные и симметричны ОС можно сделать следующие выводы:
  - асимметричные ОС обычно более просты с точки зрения внутренней организации и программного кода;
  - симметричные ОС более надежны, т.к. способны поддерживать функционирование n-процессорной системы при выходе из строя (n-1) любых процессоров.

Виды мультипроцессирования

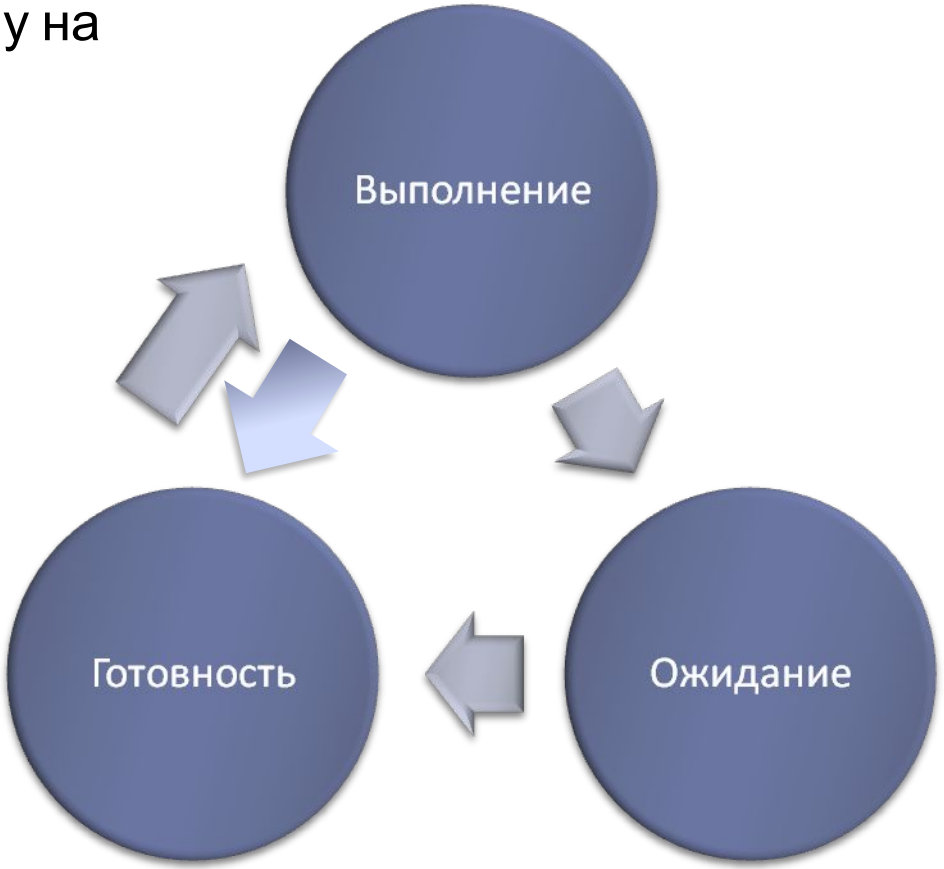


Вопрос

- Как Вы думаете почему ОС с симметричной поддержкой многопроцессорной обработки сложнее, чем асимметричные ОС?

Граф состояний процессов (потоков) при мультипроцессировании

- Несколько состояний «Выполнение» (по одному на каждый процессор).
- Одно или несколько состояний «Готовность» (общая очередь или отдельная на каждый процессор).



Особенности алгоритмов управления ресурсами

- Выше были рассмотрены характеристики ОС, связанные с управлением только основным видом ресурса – процессорным временем.
- Особенности управления другими видами ресурсов (память, файловая система), также оказывающее важное влияние на облик операционной системы в целом, будут рассмотрены в соответствующих разделах дисциплины.

Классификация операционных систем

Особенности областей использования

Типы многозадачных ОС

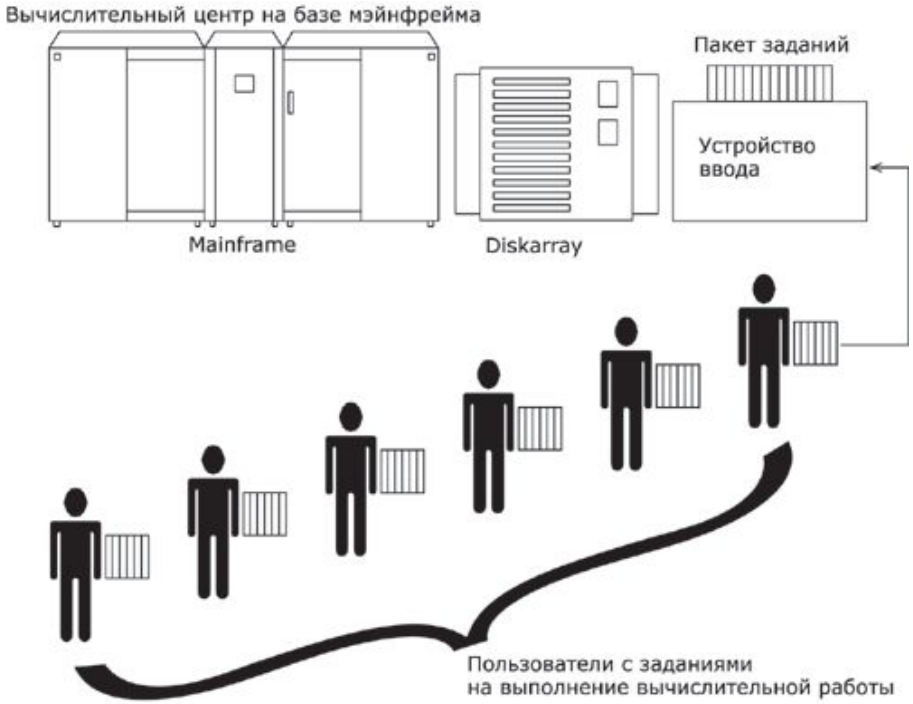
- Многозадачные ОС подразделяются на три типа в соответствии с использованными при их разработке критериями эффективности:
  - системы пакетной обработки (например, ОС ЕС);
  - системы разделения времени (UNIX, MS Windows);
  - системы реального времени (QNX, RT/11).

Системы пакетной обработки

- Системы пакетной обработки (batch processing) предназначались для решения задач в основном вычислительного характера, не требующих быстрого получения результатов.
- Главной целью и критерием эффективности систем пакетной обработки является максимальная пропускная способность.
- Основные недостатки:
  - трудность отладки программ;
  - опасность зацикливания программы;
  - из-за слабой защиты системы одно задание может оказать влияние на выполнение других заданий.

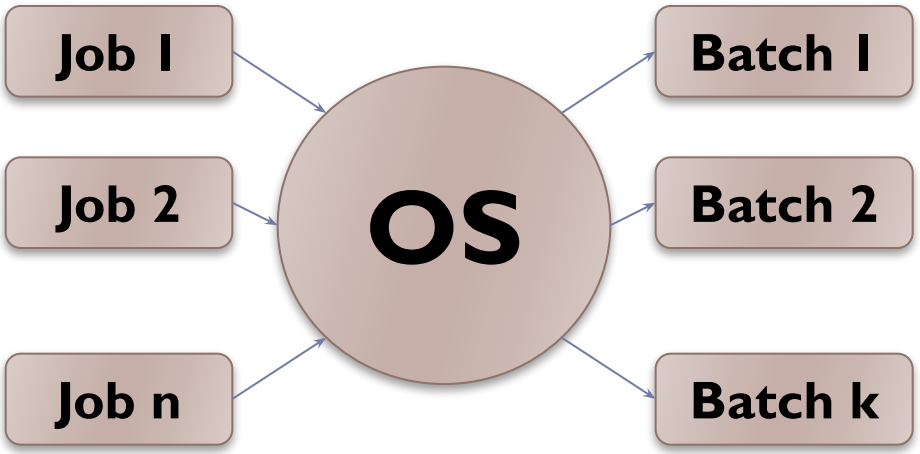
Загрузка системы заданиями

- В системах пакетной обработки в начале работы формируется пакет заданий, каждое задание содержит требования к системным ресурсам.



Множество одновременно выполняемых задач

- Далее из пакета заданий формируется множество одновременно выполняемых задач.



- Для одновременного выполнения выбираются задачи, предъявляющие отличающиеся требования к ресурсам, так, чтобы обеспечивалась сбалансированная загрузка всех устройств ВС; так, например, желательно одновременное присутствие вычислительных задач и задач с интенсивным вводом-выводом.

Диспетчеризация пакетной системы

- Переключение процессора с выполнения одной задачи на выполнение другой происходит только в случае, если активная задача сама отказывается от процессора, например, из-за необходимости выполнить операцию ввода-вывода.
- Из-за применяемой невытесняющей многозадачности в системах пакетной обработки невозможно выполнение интерактивных задач.

Системы разделения времени (1)

- **Системы разделения времени** (time sharing) призваны исправить основной недостаток систем пакетной обработки – отсутствие интерактивности.
- Первоначально системы разделения времени представляли собой многотерминальные ВС на базе мэйнфреймов. Каждому пользователю предоставлялся удаленный терминал, с которого он вел диалог со своей программой. Каждому терминалу выделяется квант процессорного времени, за счет чего достигалась иллюзия «персональной» ЭВМ.
- В настоящее время все многозадачные ОС, которые предоставляют пользователю интерактивный режим работы, считаются системами разделения времени (иногда их называют еще диалоговыми ОС).

Системы разделения времени (2)

- Системы разделения времени обладают меньшей пропускной способностью, чем системы пакетной обработки, так как на выполнение принимается каждая запущенная пользователем задача, а не та, которая «выгодна» системе, и, кроме того, имеются накладные расходы вычислительной мощности на более частое переключение процессора с задачи на задачу.
- Критерием эффективности систем разделения времени является не максимальная пропускная способность, а удобство и эффективность работы пользователя.

Вопрос

- Какой вид многозадачности и дисциплины обслуживания на Ваш взгляд наиболее подходят для операционной системы с разделением времени?

Системы реального времени

- Критерием эффективности для систем реального времени временя реакции системы на события от объекта управления, это свойство системы называется реактивностью.
- Системы реального времени подразделяются:
  - Hard real-time:
    - при нарушении временных ограничений может возникнуть критическая ошибка (отказ);
  - Soft real-time:
    - нарушение временных ограничений не приводит к отказу.

Hard real-time ОС РВ

- Системы типа «hard real-time» применяются для управления различными техническими объектами, такими, (станок, научная экспериментальная установка) или технологическими процессами (гальваническая линия, доменный процесс). Во всех этих случаях существует предельно допустимое время, в течение которого должна быть выполнена та или иная программа, управляющая объектом, в противном случае может произойти авария.
- Обычно характеризуются «облегченной» аппаратной платформой (встроенные системы) благодаря чему ОС может быть упрощена и лишена части функционала. Например, очень часто в таких системах вторичная память ограничена или вообще отсутствует; программный код операционной системы в этом случае хранится в ПЗУ (ROM).

Soft real-time ОС РВ

- Полезны в приложениях (симуляторы, виртуальная реальность), требующих повышенных интерактивных возможностей ОС.
- Ограниченная полезность для промышленных систем управления или в робототехнике.
- Зачастую могут быть реализованы на основе операционных систем разделения времени, поддерживающих абсолютные приоритеты, например, на основе MS Windows 2000+.

Гибридные системы

- Современные операционные системы могут совмещать в себе свойства систем разных типов, например, часть задач может выполняться в режиме пакетной обработки, а часть – в режиме реального времени или в режиме разделения времени. В таких случаях режим пакетной обработки часто называют фоновым режимом.

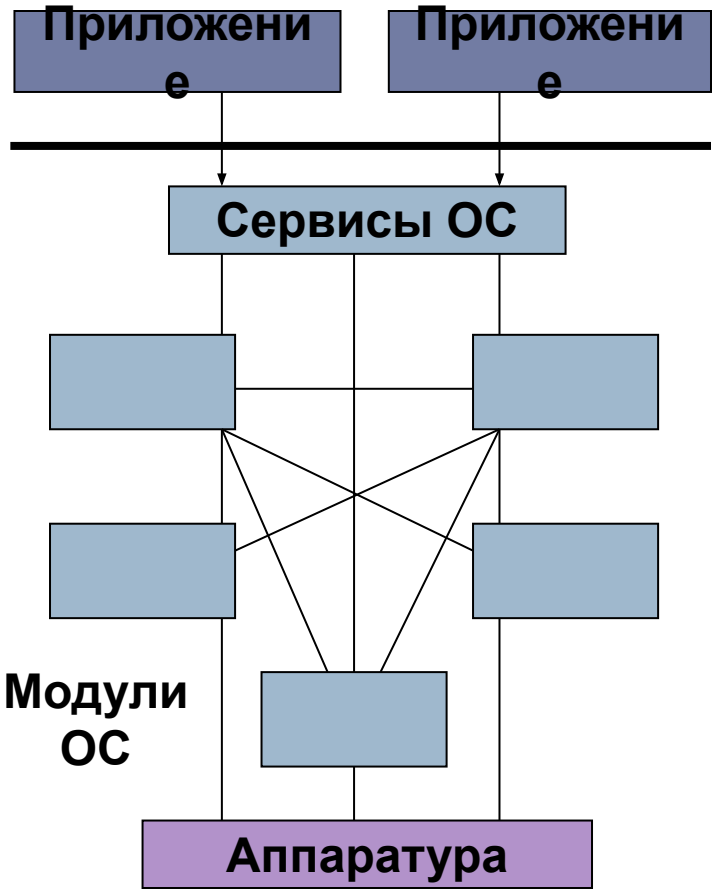
Операционные системы

Структурная организация операционных систем

Основные способы структурной организации

- При описании операционной системы часто указываются особенности ее структурной организации и основные концепции, положенные в ее основу.
- Рассмотрим основные способы структурной организации ОС:
  - монолитную структуру;
  - многоуровневую структуру;
  - ядерную структуру;
  - микроядерную структуру;
  - объектно-ориентированный подход.

Монолитная структура



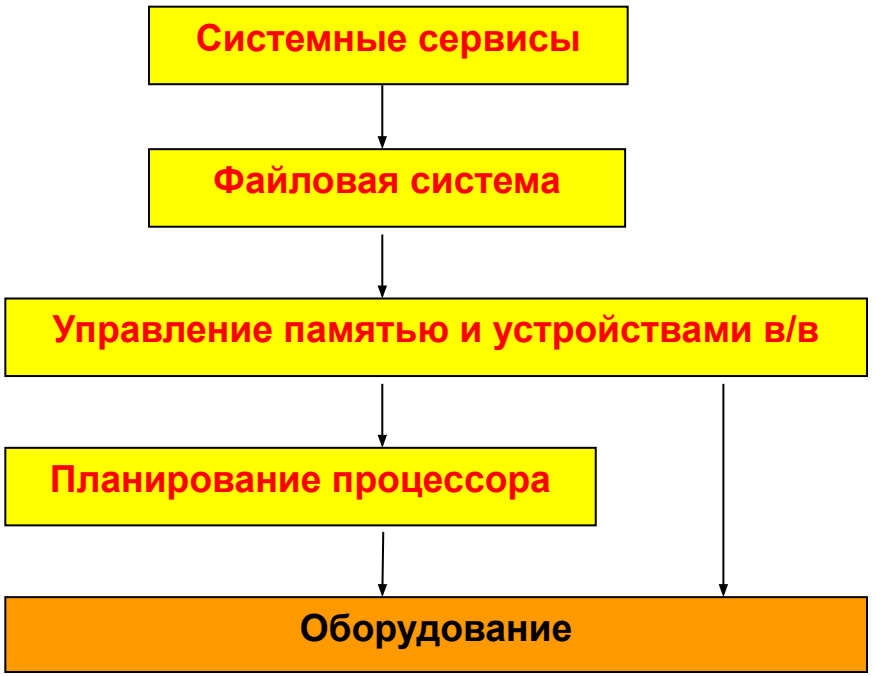
- Наиболее простым и распространенным способом построения ОС является монолитная структура, когда система компонуется как одна программа.
- Для построения монолитной ОС необходимо скомпилировать все отдельные процедуры, а затем связать их вместе в единый объектный файл с помощью компоновщика.
- Примерами могут служить ранние версии ядра UNIX или Novell NetWare.

Вопрос

- Какие недостатки на Ваш взгляд присущи монолитной операционной системы?

Многоуровневая структура

- Развитием монолитного подхода является многоуровневый, когда ОС реализуется как иерархии уровней.
- Уровни образуются группами функций ОС – системные сервисы, файловая система, управление памятью и устройствами и т.п.



- Каждый уровень может взаимодействовать только со своим непосредственным соседом – выше- или нижележащим уровнем.

Многоуровневая структура

- Первой многоуровневой ОС считают систему THE, принцип построения которой заключался в том, что при реализации каждого уровня использовались только модули предшествующего уровня.
- ОС THE была создана в Technische Hogeschool Eindhoven Э. Дейкстрой (E. W. Dijkstra) и его студентами в 1968 году.
- ОС THE была простой пакетной системой для компьютера Electrologica X8, память которого состояла из 32 К 27-разрядных сл

Уровень	Функция
5	Оператор
4	Программы пользователя
3	Управление вводом/выводом
2	Связь оператор-процесс
1	Управление памятью и барабаном
0	Распределение процессора и многозадачность

Понятие ядра

- Развитием многоуровневой концепции стала ядерная архитектура. В общем случае уровни ОС представляют собой серию концентрических колец, где внутренние кольца являлись более привилегированными, чем внешние.
- Ядро – центральная часть ОС, выполняющая основные функции.
- Ядро системы MULTICS, находящееся постоянно в памяти компьютера занимало всего 135 Килобайт кода.

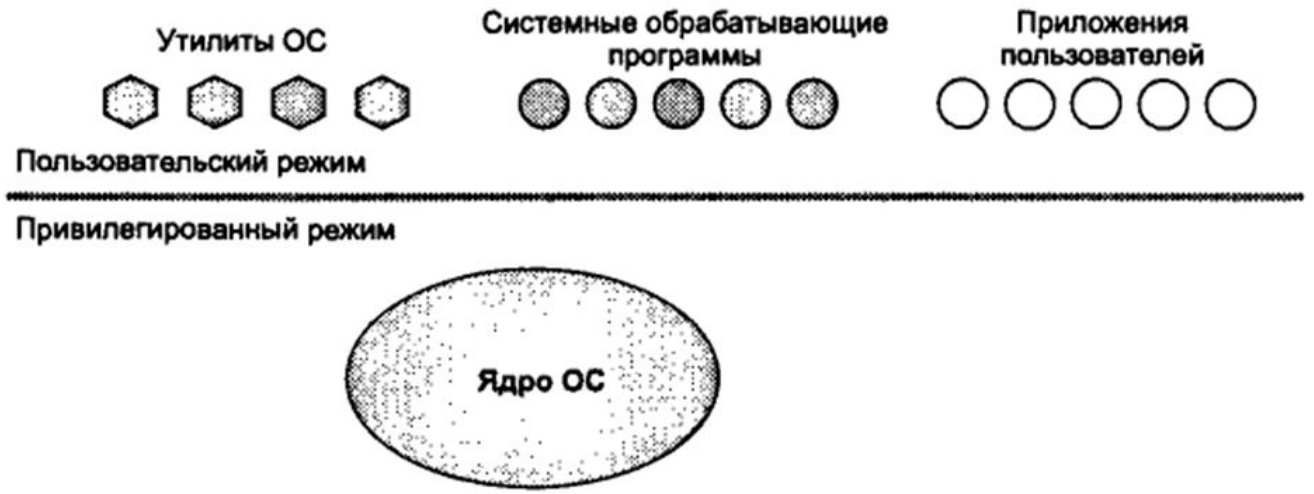
Вопрос

- Какая основная функция ядра ОС?

Уровни привилегий (защиты)

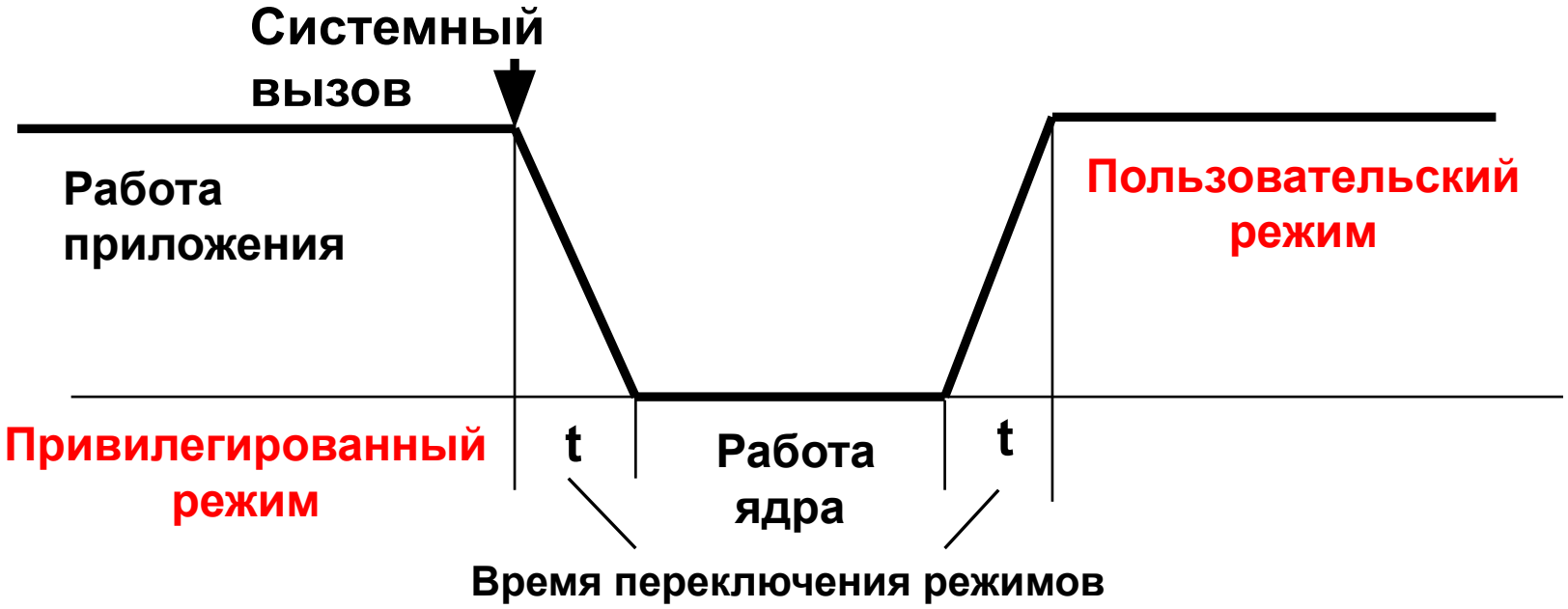
- Для обеспечения привилегий ОС необходима соответствующая аппаратная поддержка.
- Между числом уровней привилегий, поддерживаемых аппаратно, и числом уровней привилегий ОС нет прямого соответствия.
- Для реализации ядра необходимо хотя бы два уровня: основные процедуры ОС выполняются в привилегированном режиме, тогда как пользовательские программы – в непривилегированном.

Ядро в привилегированном (защищенном) режиме



- Повышение устойчивости ОС обеспечиваемое рабой ядра в привилегированном режиме достигается за счет некоторого замедления выполнения системных вызовов.

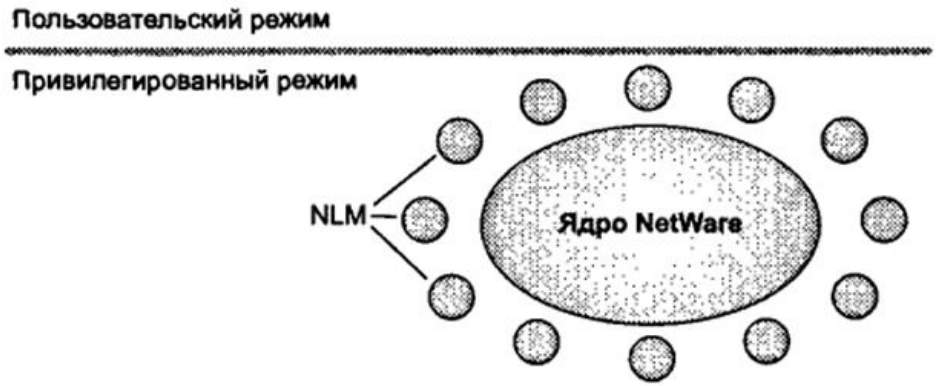
Смена режимов при выполнении вызова функции ядра



- Системный вызов привилегированного ядра инициирует переключение процессора из пользовательского режима в защищенный, а при возврате к приложению – обратно. В результате вызов выполняется медленнее.

Пример ядра в непривилегированном режиме

- В некоторых случаях разработчики ОС отступают от этого классического варианта архитектуры, организуя работу ядра и приложений в одном и том же режиме.
- Так, сетевая ОС Novell NetWare использует привилегированный режим процессоров Intel x86/Pentium как для работы ядра, так и для работы своих специфических приложений – загружаемых модулей NLM.



Монолитное ядро

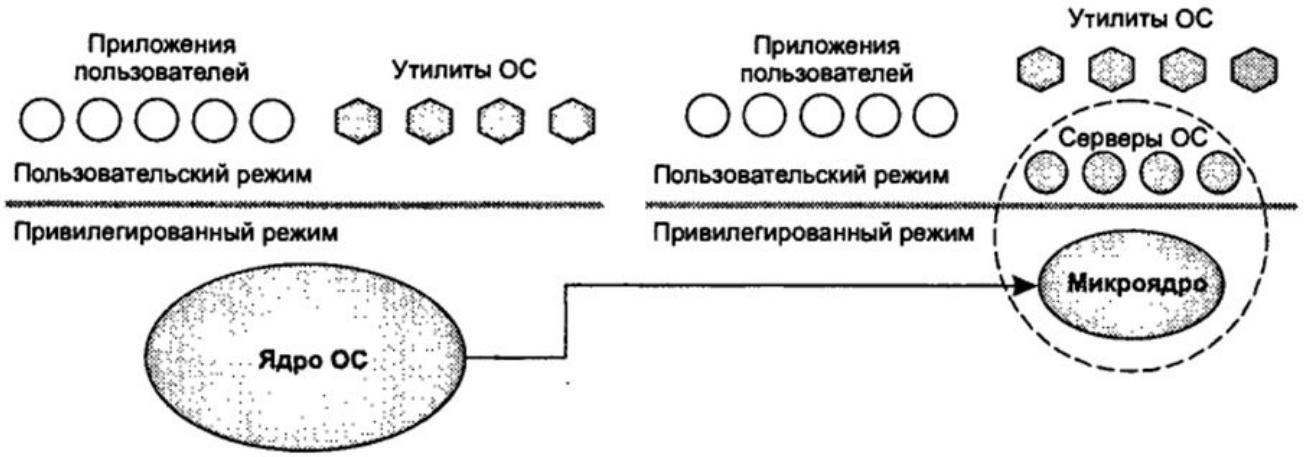
- Наиболее распространенным и классическим вариантом реализации ядерного подхода является **монолитное ядро**.
- Монолитность ядер усложняет их отладку, понимание кода ядра, добавление новых функций и возможностей, удаление «мёртвого», ненужного, унаследованного от предыдущих версий, кода.
- «Разбухание» кода монолитных ядер также повышает требования к объёму оперативной памяти, требуемому для функционирования ядра ОС.
- Это делает монолитные ядерные архитектуры мало пригодными к эксплуатации в системах, сильно ограниченных по объёму ОЗУ, например, встраиваемых системах, производственных микроконтроллерах и т. д.

Модульное ядро

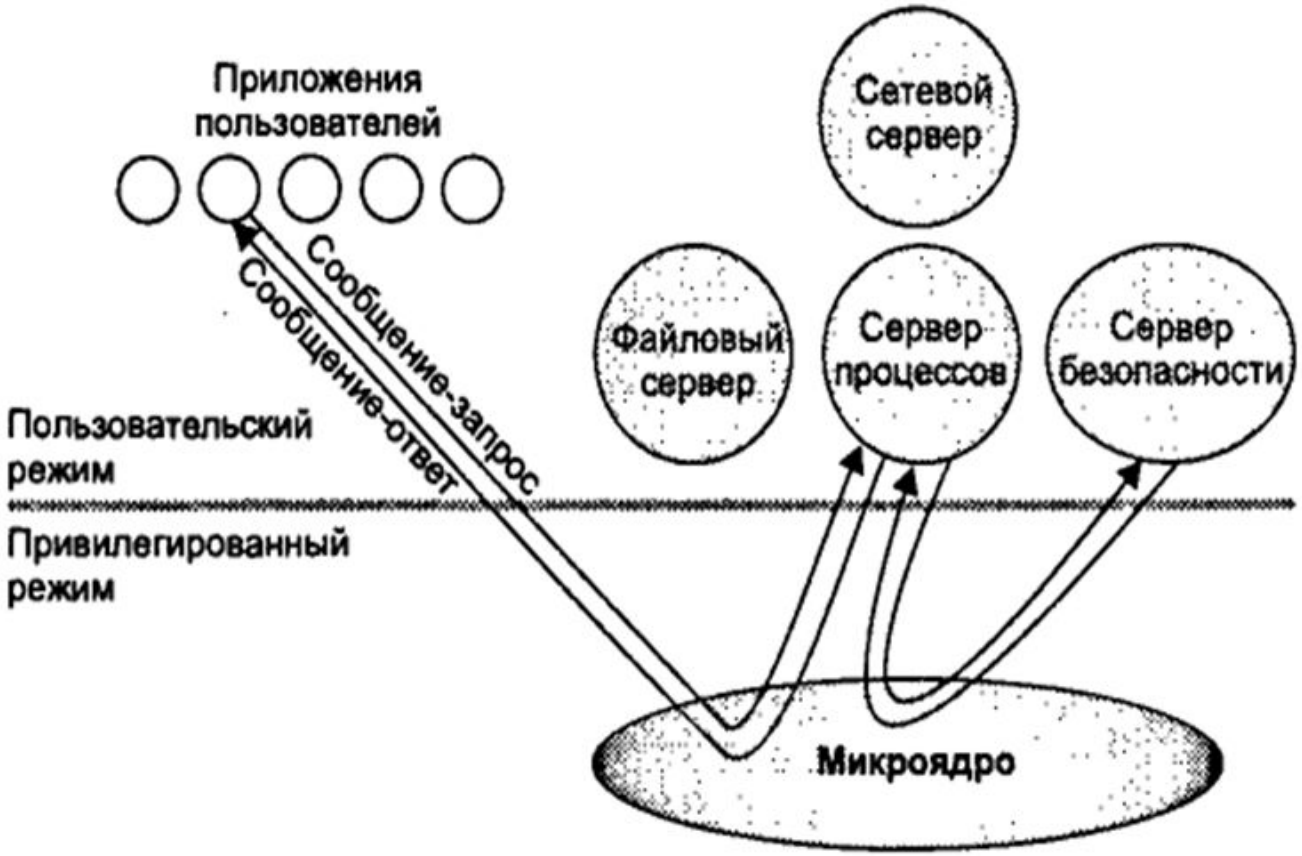
- Современная, усовершенствованная модификация архитектуры монолитных ядер ОС.
- В отличие от «классических» монолитных ядер, считающихся ныне устаревшими, модульные ядра, как правило, не требуют полной перекомпиляции ядра при изменении состава аппаратного обеспечения компьютера.
- Модульные ядра предоставляют тот или иной механизм подгрузки модулей ядра, поддерживающих то или иное аппаратное обеспечение (например, драйверов). При этом подгрузка модулей может быть как динамической (выполняемой «на лету», без перезагрузки ОС, в работающей системе), так и статической (выполняемой при перезагрузке ОС после переконфигурирования системы на загрузку тех или иных модулей).

Микроядро

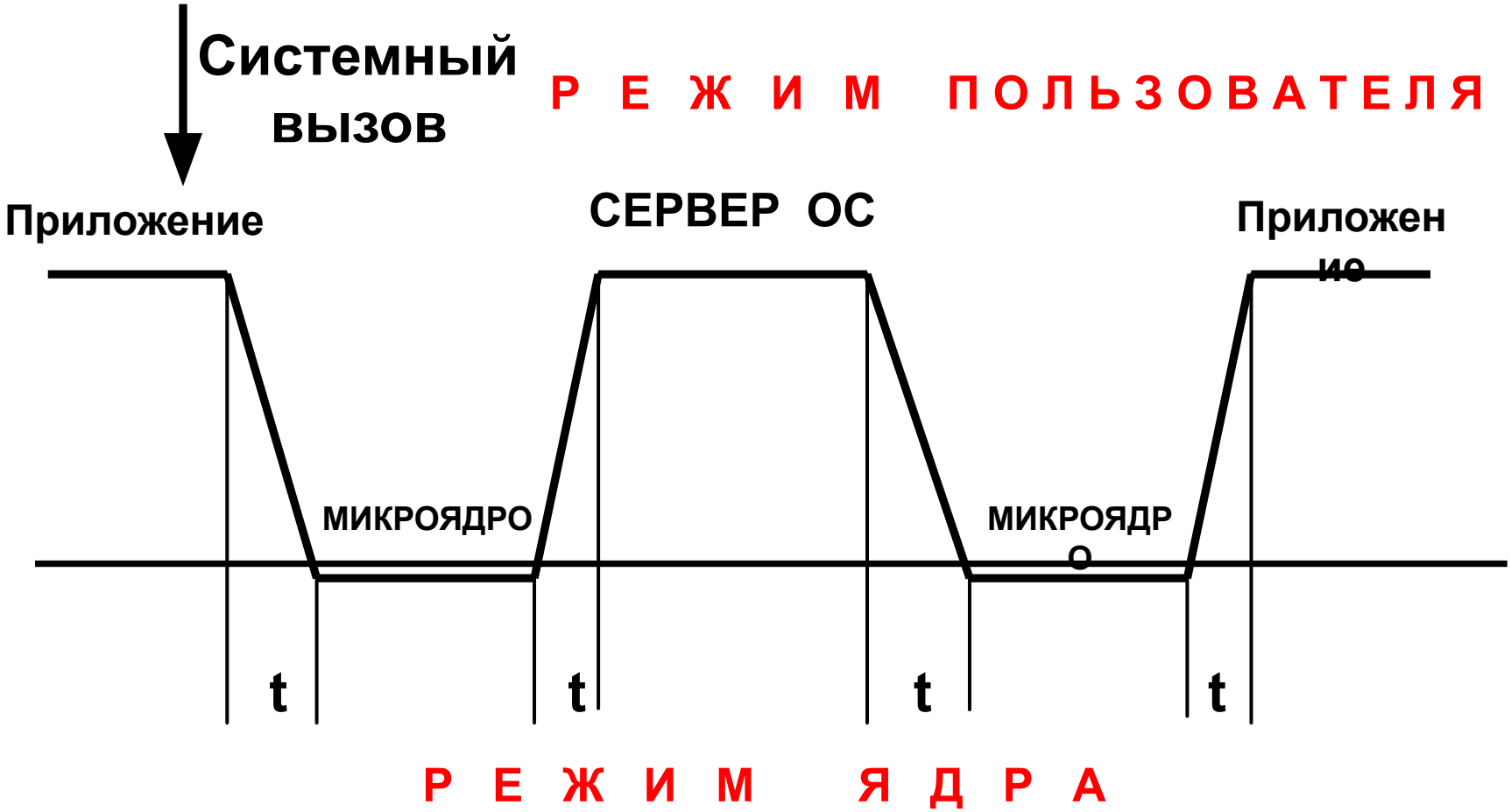
- Развитием ядерного подхода явилась архитектура на основе микроядра.
- Микроядро работает в привилегированном режиме и выполняет только минимум функций по управлению аппаратурой, в то время как функции ОС более высокого уровня выполняют специализированные компоненты ОС – серверы, работающие в пользовательском режиме.



Реализация системного вызова в микроядерной архитектуре



Смена режимов при выполнении вызова функции микроядра



Вопрос

- Сравните ядерную и микроядерную архитектуру операционной системы.

Достоинства и недостатки микроядра

- При микроядерном построении ОС работает более медленно, так как часто выполняются переходы между привилегированным и пользовательским режимом.
- Систему проще функционально развивать, добавляя, модифицируя или исключая серверы пользовательского режима.
- Серверы хорошо защищены друг от друга.

Микроядро Mach

- **Mach** – это микроядро ОС, разработанное в Carnegie Mellon University в исследовательских целях для решения задач с использованием распределенных вычислений. Это одно из микроядер первого поколения.
- Проект разрабатывался с 1985 по 1994 год, закончился на Mach 3.0. Сравнения проведенные в 1997 году показали, что Unix построенный на Mach 3.0 на 50 % медленнее чем традиционный Unix.
- Некоторое число разработчиков продолжило Mach исследования, например, микроядро второго поколения L4.
- Наиболее удачным примером коммерческой реализации можно считать Mac OS X, которая использует сильно модифицированный Mach 3.0 .

Модификации ядерного и микроядерного подходов

- Большинство современных проектов коммерческих ОС используют различные комбинации подходов на основе ядра и микроядра, например:
  - гибридное ядро – модифицированное микроядро, позволяющие для ускорения работы запускать «несущественные» части ОС в пространстве ядра. Примером реализации гибридного ядра можно считать ОС Linux и Windows 2000-2008.
  - наноядро – архитектура ядра ОС компьютеров, в рамках которой крайне упрощённое и минималистичное ядро выполняет лишь одну задачу – обработку аппаратных прерываний, генерируемых устройствами компьютера.

Применение наноядра

- Наиболее часто в современных компьютерах наноядра используются для виртуализации аппаратного обеспечения реальных компьютеров или для реализации механизма гипервизора, с целью позволить нескольким или многим различным ОС работать одновременно и параллельно на одном и том же компьютере.
- Наноядра также могут использоваться для обеспечения переносимости (портабельности) ОС на разное аппаратное обеспечение или для обеспечения возможности запуска «старой» ОС на новом аппаратном обеспечении без ее полного переписывания и портирования.

Поддержка виртуализации

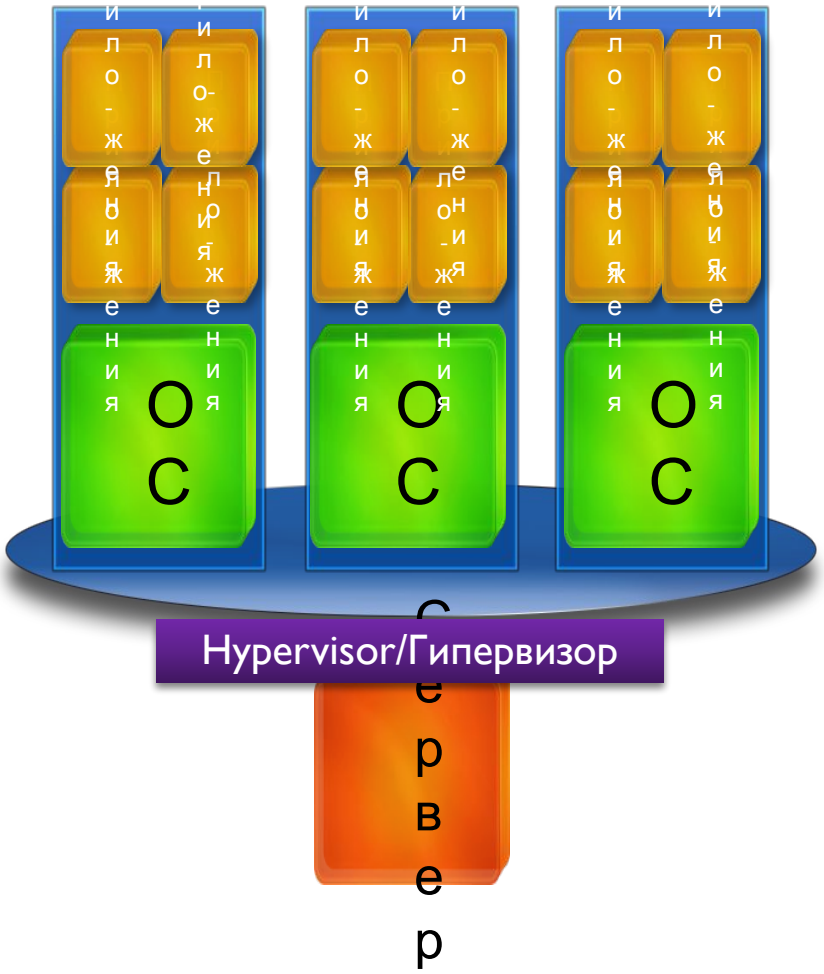
- Виртуализация серверов, говоря обобщенно, позволяет взять одно физическое устройство и установить на нем (и использовать одновременно) две или более среды ОС.
- Все это направлено на уменьшение затрат, улучшение использования ресурсов, оптимизацию инфраструктуры.

Типы виртуализации

- В архитектуре VMM типа 1, уровень монитора виртуальных машин (VMM) работает прямо над оборудованием. Это часто называется уровнем гипервизора. Эта архитектура была первоначально разработана IBM в 1960-е годы для мэйнфреймов и недавно стала доступной на платформах x86/x64.

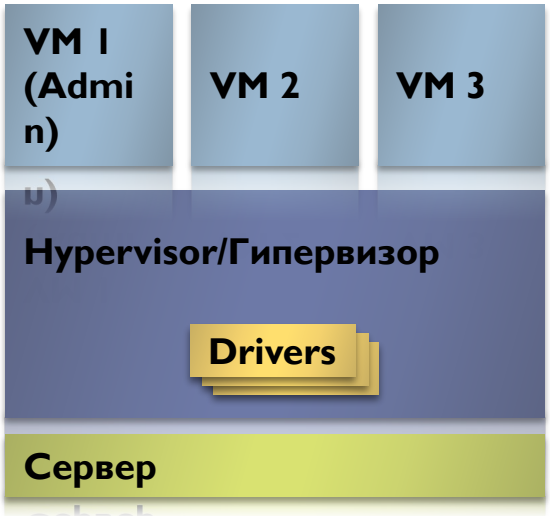
Гипервизор

- **Гипервизор** – программа или аппаратная схема, обеспечивающая или позволяющая одновременное, параллельное выполнение нескольких ОС на одном и том же хост-компьютере.
- Гипервизор также обеспечивает изоляцию операционных систем друг от друга, защиту и безопасность, разделение ресурсов между различными запущенными ОС и управление ресурсами.

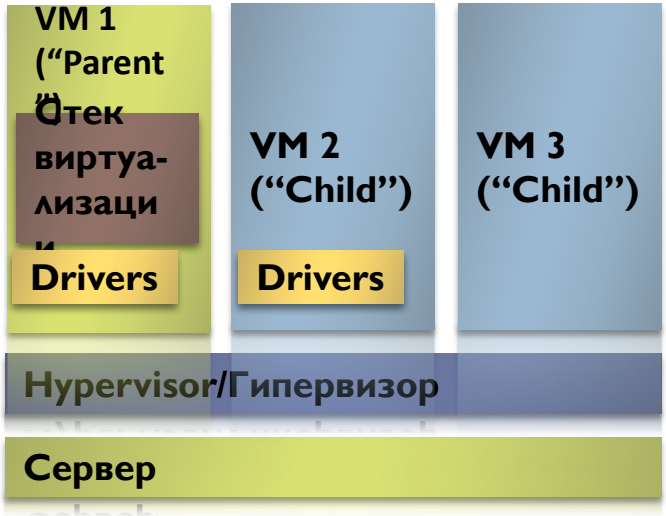


Архитектуры гипервизора

Монолитная



Микроядерная



Типы архитектуры гипервизора

- Монолитный подход размещает гипервизор/VMM в едином уровне, который также включает большинство требуемых компонентов, таких как ядро, драйверы устройств и стек ввода/вывода. Это подход, используемый такими решениями, как VMware ESX и традиционные системы мэйнфреймов.
- Микроядерный подход использует очень тонкий, специализированный гипервизор, выполняющий лишь основные задачи обеспечения изоляции разделов и управления памятью. Этот уровень не включает стека ввода/вывода или драйверов устройств. В этой архитектуре стек виртуализации и драйверы конкретных устройств расположены в специальном разделе, именуемом родительским разделом.

Объектно-ориентированный подход

- Развитием технологии расширяемых модульных систем является **объектно-ориентированный подход**, при котором каждый программный компонент ОС является функционально изолированным от других. Основным понятием этого подхода является «объект».
- *Объект* – это единица программ и данных, взаимодействующая с другими объектам посредством приема и передачи сообщений. Объект может быть представлением как некоторых конкретных вещей – прикладной программы или документа, так и некоторых абстракций – процесса, события.
- Программы (функции) объекта определяют перечень действий, которые могут быть выполнены над данными этого объекта. Объект-клиент может обратиться к другому объекту, послав сообщение с запросом на выполнение какой-либо функции объекта-сервера.

ООП: достоинства и недостатки

- Построение ОС на базе объектно-ориентированного подхода имеет следующие достоинства:
  - аккумуляция удачных решений в форме стандартных объектов и создание новых объектов на их базе с помощью механизма наследования;
  - предотвращение несанкционированного доступа к данным за счет их инкапсуляции во внутренние структуры объекта;
  - структурированность системы, состоящей из набора хорошо определенных объектов.
- В качестве основных недостатков объектно-ориентированного похода следует выделить сложность управления объектами и как следствие более медленную работу системы.

Классификация ОС

Особенности вычислительной среды и аппаратной платформы

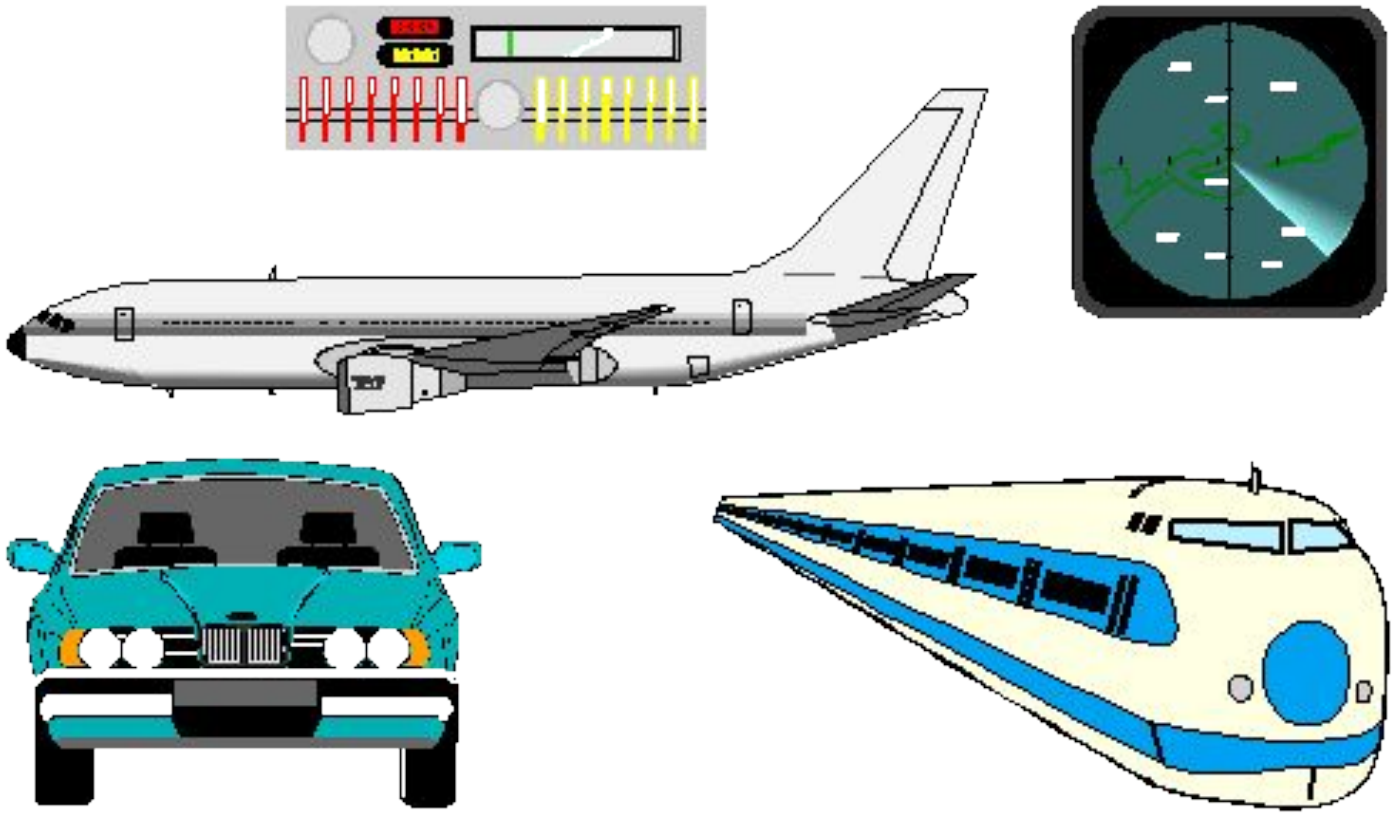
Особенности вычислительной среды и аппаратной платформы

- На свойства ОС непосредственное влияние оказывают особенности вычислительной среды, на которые она ориентирована. Например, различают ОС для встроенных систем, ОС для локальных вычислений, сетевые ОС.
- Наряду с ОС, ориентированными на совершенно определенный тип аппаратной платформы, существуют системы, специально разработанные таким образом, чтобы они могли быть легко перенесены с компьютера одного типа на компьютер другого типа.
- В этих системах аппаратно-зависимые места тщательно локализованы, так что при переносе системы на новую платформу переписываются только они. Средством, облегчающем перенос остальной части ОС, является написание ее на машинно-независимом языке, например, на Си, который и был разработан для программирования ОС.
- Наиболее ярким примером такой ОС является популярная система UNIX.

Вычислительные среды

- Встроенные (Embedded Computing)
- Традиционные (Traditional Computing)
- Сетевые (Web-Based Computing)

Примеры встроенных систем



Примеры встроенных систем

Обработка цифровых изображений, устройства печати

Устройства розничной торговли, банкоматы, кассовые аппараты

Игровые автоматы

Мобильные телефоны, автомобильные системы

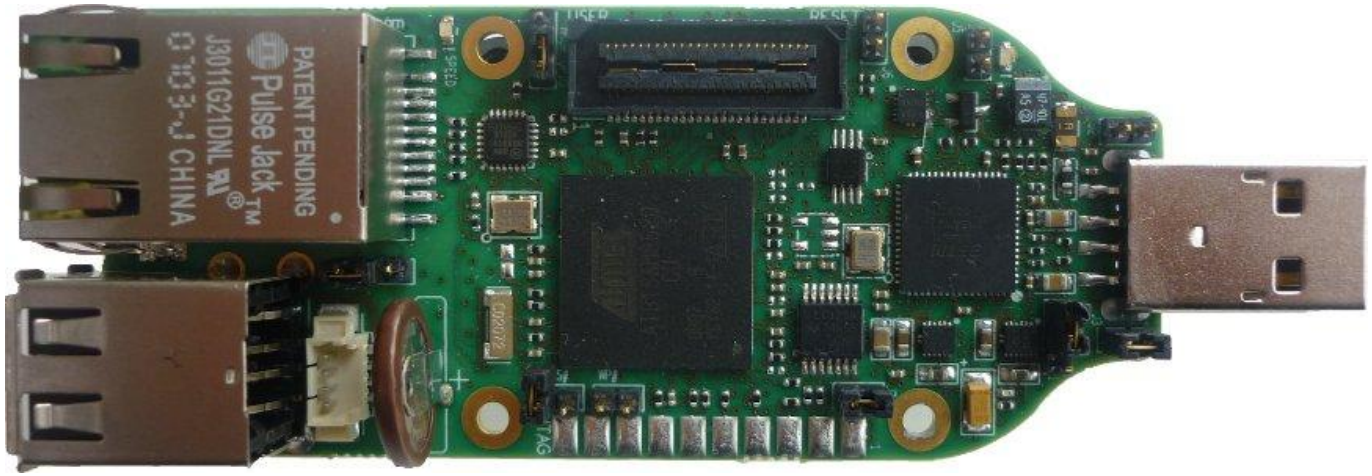
Телевизионные приставки, цифровые видеомагнитофоны, бытовая автоматизация, медиаплееры

Промышленная автоматизация

Медицинские системы

Измерительные приборы

Примеры встроенных систем



- Embedded Computer with USB form-factor (36 x 85 mm), ATMELAT91SAM9G20 @ 400MHz, 256MB NAND Flash (8bits), 64MB SDRAM (32bits @ 133MHz), 1x Ethernet 10/100, 2x USB Host, 1x USB Device, 1x USB Debug, 1x RTC battery backup, 1x Micro-SD, 50 pins expansion Port.

Традиционные вычисления

- «Локальные» вычисления:
  - ноутбук
  - рабочая станция
  - сервер
  - суперкомпьютер

Особенности аппаратных платформ

- На свойства ОС непосредственное влияние оказывают аппаратные средства, на которые она ориентирована. По типу аппаратуры различают ОС персональных компьютеров, мини-компьютеров, мэйнфреймов, кластеров и сетей ЭВМ.
- Наряду с ОС, ориентированными на совершенно определенный тип аппаратной платформы, существуют системы, специально разработанные таким образом, чтобы они могли быть легко перенесены с компьютера одного типа на компьютер другого типа.
- В этих системах аппаратно-зависимые места тщательно локализованы, так что при переносе системы на новую платформу переписываются только они. Средством, облегчающем перенос остальной части ОС, является написание ее на машинно-независимом языке, например, на Си, который и был разработан для программирования ОС.
- Наиболее ярким примером такой ОС является популярная система UNIX.

Высокопроизводительные системы

- Логические процессоры (hyperthreading) – многопоточность в рамках одного ядра
- Многоядерные процессоры
- SMP-системы (оперативная память физически представляет последовательное адресное пространство, доступ к которому имеют одновременно все процессоры системы по единой шине)
- NUMA (Non-Uniform Memory Architecture)

Современные многоядерные процессоры

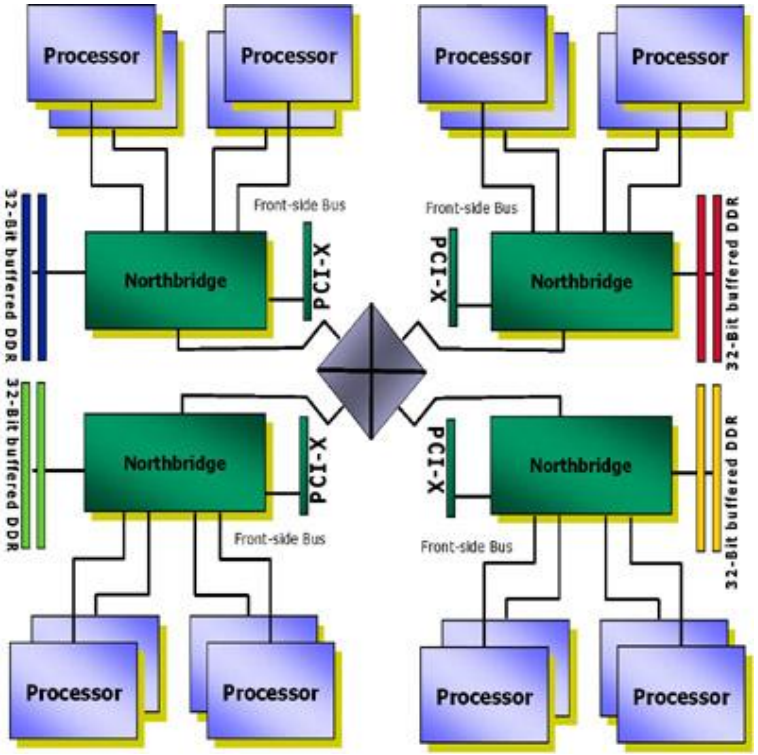
- AMD (Opteron – 16 ядер)
- Intel (Xeon E5, E7 v3 – до 18 ядер и 36 потоков)
- IBM (Power8 – до 12 ядер и 96 потоков)
- Oracle (SPARC T3 – 16 ядер и 128 потоков)

Перспективы многоядерных процессоров

- MIT 110-ядерный процессор построен на архитектуре, благодаря которой удалось в 14 сократить количество данных, перемещающихся по внутренним цепям чипа. Это позволило сократить потребление процессором энергии и выделение тепла.
- Размер кристалла, на котором исследователям удалось разместить 110 ядер, составляет 10 x 10 мм. Чип был изготовлен с применением 45-нм технологической нормы.
- В процессоре, разработанном учеными MIT, кэш был заменен на общий пул памяти, что позволило снизить трафик.
- Более того, процессор способен прогнозировать тенденции перемещения данных по своим цепям. Это позволило сократить количество циклов, необходимых для обработки информации.

NUMA

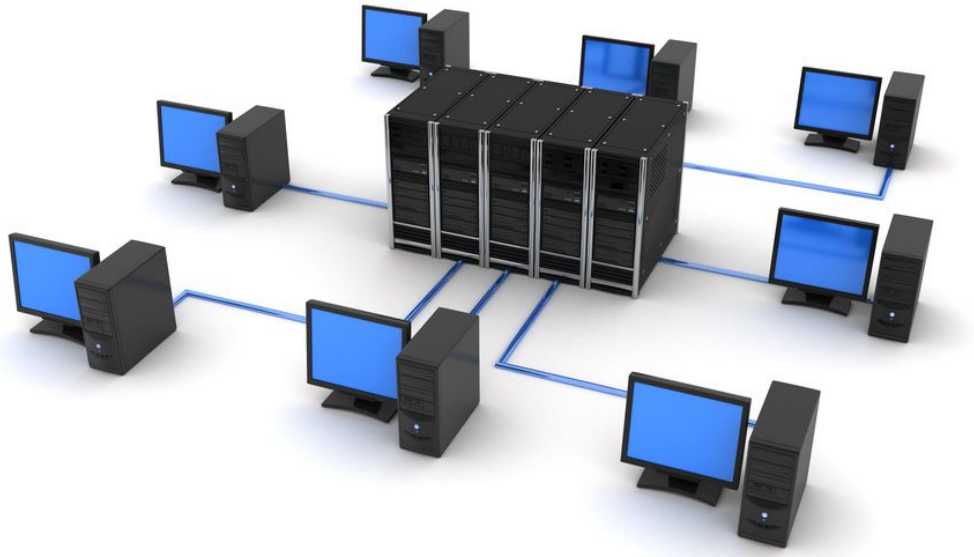
- Процессоры группируются в узлы (Nodes).
- В каждом узле несколько CPU и память (SMP-система, но за счет минимальной компоновки элементов достигается высокая пропускная способность между процессором и локальной памятью модуля).
- Узлы объединяются высокопроизводительной шиной.



Web-Based Computing

- Network Computing
- Grid Computing
- Cloud Computing

Network Computing



- Вычислительные сет
- Распределенная обработка данных
- Клиент-серверная архитектура
- Сетевые операционные системы

Grid Computing как распределенный суперкомпьютер

- Классическая схема Grid Computing основана на использовании распределенных процессорных мощностей и распределенных систем хранения.
- Grid Computing позволяет эффективно задействовать незанятые вычислительные ресурсы, которые могут быть разбросаны по всему миру.

Облачные вычисления

- **Cloud computing** – это парадигма распределенного широкомасштабного компьютинга, где набор абстрактных виртуализованных, динамически масштабируемых, дистанционно управляемых машин, устройств памяти, платформ и сервисов предоставляются по запросу удаленным пользователям через Интернет.

Типы облачных вычислений

- **SaaS**

  (Software as a Service) – программное обеспечение как услуга
- **PaaS**

  (Platform as a Service) – платформа как услуга
- **IaaS**

  (Infrastructure as a Service) – инфраструктура как услуга