

Тема 1.2. Архитектура операционной системы

ЦЕЛЬ УРОКА

ИЗУЧИТЬ КОМПОНЕНТЫ И СТРУКТУРУ ОПЕРАЦИОННОЙ СИСТЕМЫ

План

1. Ядро и вспомогательные модули
2. Ядро в привилегированном режиме
3. Многослойная структура ОС

1. Ядро и вспомогательные модули

Наиболее общим подходом к структуризации операционной системы является разделение всех ее модулей на две группы:

- **ядро** — модули, выполняющие основные функции ОС;
- **модули**, выполняющие вспомогательные функции ОС.

Модули ядра выполняют такие базовые функции ОС:

- управление процессами,
- памятью,
- устройствами ввода-вывода и т.п.

Ядро составляет сердцевину операционной системы, без него ОС является полностью неработоспособной и не сможет выполнить ни одну из своих функций.

[

В состав ядра входят функции, решающие внутрисистемные задачи организации вычислительного процесса:

- переключение контекстов,
- загрузка/выгрузка страниц,
- обработка прерываний.

Эти функции недоступны для приложений.

]

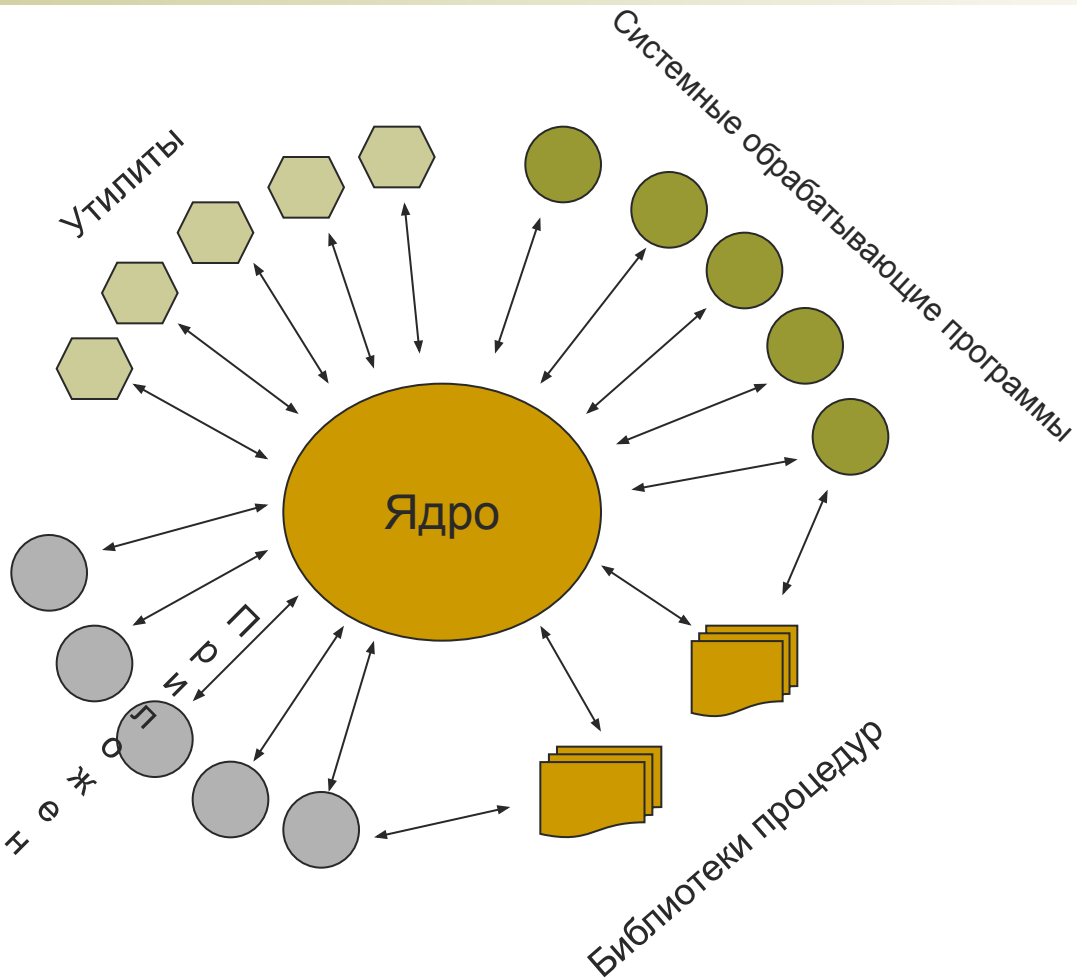
Функции, выполняемые модулями ядра, являются наиболее часто используемыми функциями ОС, поэтому **скорость их выполнения определяет производительность всей системы в целом.**


Для обеспечения высокой скорости работы ОС все модули ядра или большая их часть постоянно находятся в оперативной памяти, то есть являются *резидентными*.

Вспомогательные модули обычно подразделяются на следующие группы:

- *Утилиты* – программы решающие отдельные задачи управления и сопровождения компьютерной системы, такие как программы сжатия дисков, архивирования данных;
- *Системные обрабатывающие программы* – текстовые или графические редакторы, компиляторы, компоновщики, отладчики;
- *Программы предоставления пользователю дополнительных услуг* – специальный вариант пользовательского интерфейса, калькулятор и даже игры;
- *Библиотеки процедур* различного назначения, упрощающие разработку приложений, например библиотека математических функций, функций ввода-вывода и т.п.

Взаимодействие между ядром и вспомогательными модулями



A large black left square bracket and a large yellow right square bracket are positioned at the top of the slide. A thin yellow horizontal line spans the width of the slide, starting from the left edge and ending at the right bracket.

Как и обычные приложения, для выполнения своих задач вспомогательные модули, обращаются к функциям ядра посредством *системных вызовов*.

Модули ОС, оформленные в виде утилит, системных обрабатывающих программ и библиотек, обычно загружаются в оперативную память только на время выполнения своих функций, то есть являются ***транзитными***.

Постоянно в оперативной памяти располагаются только самые необходимые коды ОС, составляющие ее ядро.

2. Ядро в привилегированном режиме

Для надежного управления ходом выполнения приложений операционная система должна иметь по отношению к приложениям определенные привилегии.

Иначе некорректно работающее приложение может вмешаться в работу ОС и разрушить часть ее кодов.

[

Аппаратура компьютера должна поддерживать как минимум два режима работы:

- *пользовательский режим (user mode)*
- *привилегированный режим, который также называют режимом ядра (kernel mode), или режимом супервизора (supervisor mode).*

Архитектура ОС с ядром в привилегированном режиме

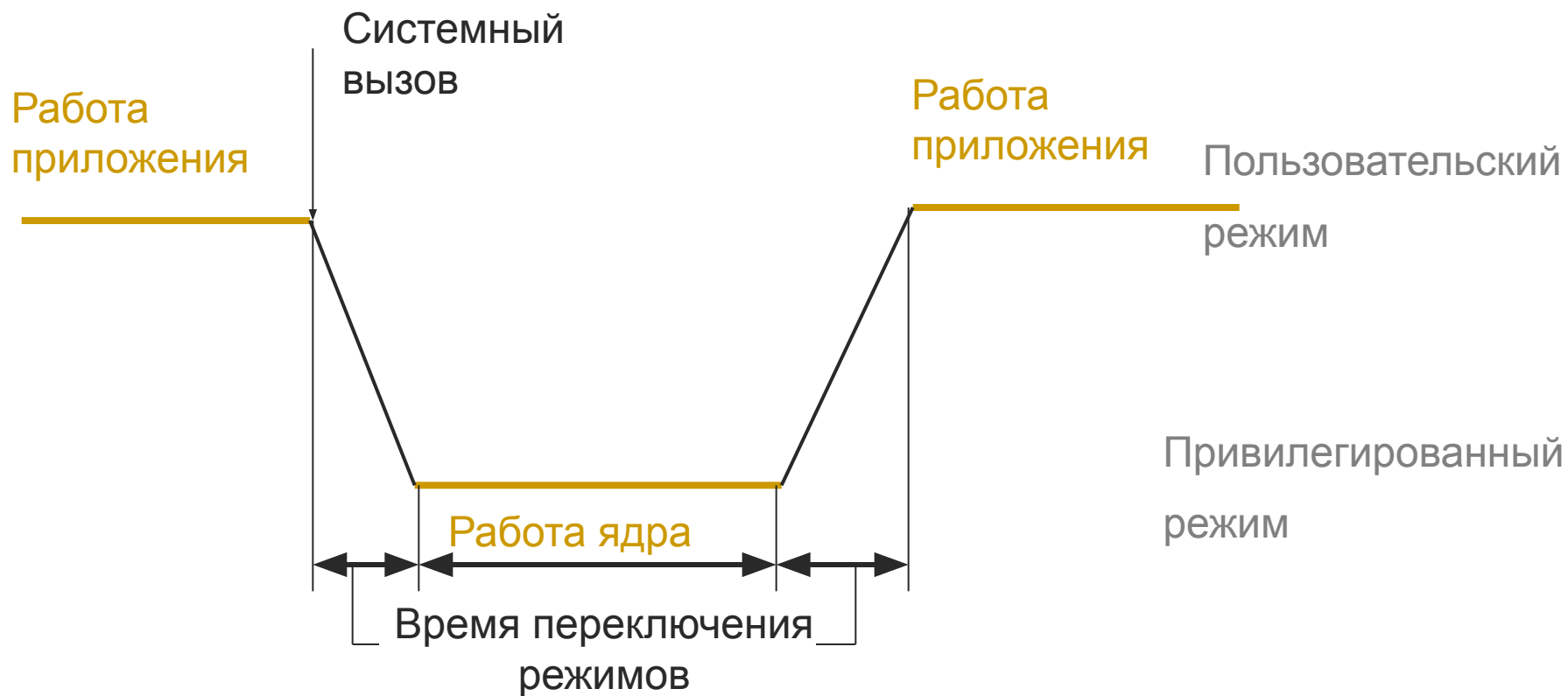


Пользовательский режим (user mode)

Привилегированный режим (kernel mode),
или режим супервизора (supervisor mode)



Смена режимов при выполнении системного вызова к привилегированному ядру



3. Многослойная структура ОС

Вычислительную систему, работающую под управлением ОС на основе ядра, можно рассматривать как систему, состоящую из трех иерархически расположенных слоев: нижний слой образует аппаратура, промежуточный — ядро, а утилиты, обрабатывающие программы и приложения, составляют верхний слой системы

Трёхслойная схема вычислительной машины



Многослойная структура ядра ОС



Ядро может состоять из следующих слоев:

- *Средства аппаратной поддержки ОС.*
Часть функций ОС может выполняться аппаратными средствами.
К операционной системе относят, не все аппаратные устройства компьютера, а только те, которые прямо участвуют в организации вычислительных процессов:
 - ✓ *средства поддержки привилегированного режима,*
 - ✓ *систему прерываний,*
 - ✓ *средства переключения контекстов процессов,*
 - ✓ *средства защиты областей памяти и т. п.*

■ *Машинно-зависимые компоненты ОС.*

Этот слой полностью экранирует вышележащие слои ядра от особенностей аппаратуры.

Это позволяет разрабатывать вышележащие слои на основе машинно-независимых модулей, существующих в единственном экземпляре для всех типов аппаратных платформ, поддерживаемых данной ОС.

■ *Базовые механизмы ядра.*

Этот слой выполняет наиболее примитивные операции ядра:

- ✓ программное переключение контекстов процессов,
- ✓ диспетчеризацию прерываний,
- ✓ перемещение страниц из памяти на диск и обратно и т. п.


Модули данного слоя **не принимают решений о распределении ресурсов** — они только отрабатывают принятые «наверху» решения, что и дает повод называть их исполнительными механизмами для модулей верхних слоев.

■ Менеджеры ресурсов.

Обычно на данном слое работают менеджеры (диспетчеры):

- ✓ процессов,
- ✓ ввода-вывода,
- ✓ файловой системы
- ✓ оперативной памяти.

Каждый из менеджеров ведет учет свободных и используемых ресурсов и планирует их распределение в соответствии с запросами приложений.



Для исполнения принятых решений менеджер обращается к нижележащему слою базовых механизмов с запросами о загрузке (выгрузке) конкретных страниц. Внутри слоя менеджеров существуют тесные взаимные связи, отражающие тот факт, что для выполнения процессу нужен доступ одновременно к нескольким ресурсам — процессору, области памяти, возможно, к определенному файлу или устройству ввода-вывода.

Например, при создании процесса менеджер процессов обращается к менеджеру памяти, который должен выделить процессу определенную область памяти для его кодов и данных.

■ *Интерфейс системных вызовов.*

Этот слой является самым верхним слоем ядра и взаимодействует непосредственно с приложениями и системными утилитами, образуя **прикладной программный интерфейс операционной системы.**

Функции API, обслуживающие системные вызовы, предоставляют доступ к ресурсам системы в удобной и компактной форме, без указания деталей их физического расположения.

Типовые средства аппаратной поддержки ОС

- средства поддержки привилегированного режима;
- средства трансляции адресов;
- средства переключения процессов;
- система прерываний;
- системный таймер;
- средства защиты областей памяти.

Средства поддержки привилегированного режима обычно основаны на системном регистре процессора, часто называемом «словом состояния» машины или процессора. Этот регистр содержит некоторые признаки, определяющие режимы работы процессора, в том числе и признак текущего режима привилегий. Смена режима привилегий выполняется за счет изменения слова состояния машины в результате прерывания или выполнения привилегированной команды. Число градаций привилегированности может быть разным у разных типов процессоров, наиболее часто используются два уровня (ядро-пользователь) или четыре (например, ядро-супервизор-выполнение-пользователь у платформы VAX или 0-1-2-3 у процессоров Intel x86/Pentium). В обязанности средств поддержки привилегированного режима входит выполнение проверки допустимости выполнения активной программой инструкций процессора при текущем уровне привилегированности.

Средства трансляции адресов

выполняют операции преобразования виртуальных адресов, которые содержатся в кодах процесса, в адреса физической памяти. Таблицы, предназначенные при трансляции адресов, обычно имеют большой объем, поэтому для их хранения используются области оперативной памяти, а аппаратура процессора содержит только указатели на эти области. Средства трансляции адресов используют данные указатели для доступа к элементам таблиц и аппаратного выполнения алгоритма преобразования адреса, что значительно ускоряет процедуру трансляции по сравнению с ее чисто программной реализацией.

Средства переключения процессов предназначены для быстрого сохранения контекста приостанавливаемого процесса и восстановления контекста процесса, который становится активным.

Содержимое контекста обычно включает содержимое всех регистров общего назначения процессора, регистра флагов операций (то есть флагов нуля, переноса, переполнения и т. п.), а также тех системных регистров и указателей, которые связаны с отдельным процессом, а не операционной системой, например указателя на таблицу трансляции адресов процесса. Для хранения контекстов приостановленных процессов обычно используются области оперативной памяти, которые поддерживаются указателями процессора. Переключение контекста выполняется по определенным командам процессора, например по команде перехода на новую задачу. Такая команда вызывает автоматическую загрузку данных из сохраненного контекста в регистры процессора, после чего процесс продолжается с прерванного ранее места.

Система прерываний позволяет компьютеру реагировать на внешние события, синхронизировать выполнение процессов и работу устройств ввода-вывода, быстро переходить с одной программы на другую. Механизм прерываний нужен для того, чтобы оповестить процессор о возникновении в вычислительной системе некоторого непредсказуемого события или события, которое не синхронизировано с циклом работы процессора.

Примерами таких событий могут служить завершение операции ввода-вывода внешним устройством (например, запись блока данных контроллером диска), некорректное завершение арифметической операции (например, переполнение регистра), истечение интервала астрономического времени. При возникновении условий прерывания его источник (контроллер внешнего устройства, таймер, арифметический блок процессора и т. п.) выставляет определенный электрический сигнал. Этот сигнал прерывает выполнение процессором последовательности команд, задаваемой исполняемым кодом, и вызывает автоматический переход на заранее определенную процедуру, называемую *процедурой обработки прерываний*. В большинстве моделей процессоров обрабатываемый аппаратурой переход на процедуру обработки прерываний сопровождается заменой слова состояния машины (или даже всего контекста процесса), что позволяет одновременно с переходом по нужному адресу выполнить переход в привилегированный режим. После завершения обработки прерывания обычно происходит возврат к исполнению прерванного кода.

Системный таймер, часто реализуемый в виде быстродействующего регистра-счетчика, необходим операционной системе для выдержки интервалов времени. Для этого в регистр таймера программно загружается значение требуемого интервала в условных единицах, из которого затем автоматически с определенной частотой начинает вычитаться по единице. Частота «тиков» таймера, как правило, тесно связана с частотой тактового генератора процессора. (Не следует путать таймер ни с тактовым генератором, который вырабатывает сигналы, синхронизирующие все операции в компьютере, ни с системными часами — работающей на батареях электронной схеме, — которые ведут независимый отсчет времени и календарной даты.) При достижении нулевого значения счетчика таймер инициирует прерывание, которое обрабатывается процедурой операционной системы. Прерывания от системного таймера используются ОС в первую очередь для слежения за тем, как отдельные процессы расходуют время процессора. Например, в системе разделения времени при обработке очередного прерывания от таймера планировщик процессов может принудительно передать управление другому процессу, если данный процесс исчерпал выделенный ему квант времени.

Средства защиты областей памяти

обеспечивают на аппаратном уровне проверку возможности программного кода осуществлять с данными определенной области памяти такие операции, как чтение, запись или выполнение (при передачах управления). Если аппаратура компьютера поддерживает механизм трансляции адресов, то средства защиты областей памяти встраиваются в этот механизм. Функции аппаратуры по защите памяти обычно состоят в сравнении уровней привилегий текущего кода процессора и сегмента памяти, к которому производится обращение.

Переносимость операционной системы

Если *код операционной системы* может быть сравнительно легко перенесен с процессора одного типа на процессор другого типа и с аппаратной платформы одного типа на аппаратную платформу другого типа, то такую ОС называют *переносимой (portable)*, или *мобильной*.

Свойства мобильности ОС

1. Большая часть кода должна быть написана на языке, трансляторы которого имеются на всех машинах, куда предполагается переносить систему.
2. Объем машинно-зависимых частей кода, которые непосредственно взаимодействуют с аппаратными средствами, должен быть по возможности минимизирован.
3. Аппаратно-зависимый код должен быть надежно изолирован в нескольких модулях, а не быть распределен по всей системе.

Свойства мобильности ОС

- Объем машинно-зависимых частей кода, которые непосредственно взаимодействуют с аппаратными средствами, должен быть по возможности минимизирован. Для уменьшения аппаратной зависимости разработчики ОС должны также исключить возможность использования по умолчанию стандартных конфигураций аппаратуры или их характеристик. Для осуществления всех необходимых действий по управлению аппаратурой, представленной этими параметрами, должен быть написан набор аппаратно-зависимых функций. Каждый раз, когда какому-либо модулю ОС требуется выполнить некоторое действие, связанное с аппаратурой, он манипулирует абстрактными данными, используя соответствующую функцию из имеющегося набора. Когда ОС переносится, то изменяются только эти данные и функции, которые ими манипулируют.

Свойства мобильности ОС

- Аппаратно-зависимый код должен быть надежно изолирован в нескольких модулях, а не быть распределен по всей системе. Изоляции подлежат все части ОС, которые отражают специфику как процессора, так и аппаратной платформы в целом. Низкоуровневые компоненты ОС, имеющие доступ к процессорно-зависимым структурам данных и регистрам, должны быть оформлены в виде компактных модулей, которые могут быть заменены аналогичными модулями для других процессоров. Для снятия платформенной зависимости, возникающей из-за различий между компьютерами разных производителей, построенными на одном и том же процессоре, должен быть введен хорошо локализованный программный слой машинно-зависимых функций.

Машино – независимая часть на алгоритмическом языке

Оперативная система для компьютера А

Машино –
незаменимая
часть ОС (на
языке
компьютера А)

Машино –
зависимая часть
компьютера А

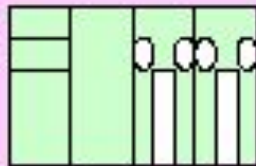


Компьютер типа А

Оперативная система для компьютера В

Машино –
незаменимая
часть ОС (на
языке
компьютера В)

Машино –
зависимая часть
компьютера А

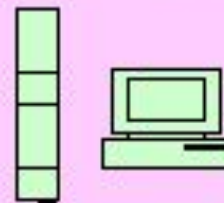


Компьютер типа В

Оперативная система для компьютера С

Машино –
незаменимая
часть ОС (на
языке
компьютера С)

Машино –
зависимая часть
компьютера А



Компьютер типа С

Перенос
операционно
й системы на
разные
аппаратные
платформы