Лекция 3. Ресурсы операционной системы

1. Обзор аппаратного обеспечения компьютера
   1.1.Процессор
   1.2. Память
   1.3. Устройства ввода-вывода
2. Основные виды ресурсов
3. Программные модули и возможности их разделения

Литература

1. **А. В. Гордеев.** Операционные системы: Учебник для вузов. 2-е изд. - СПб.: Питер, 2004. – 416 с.:ил.
2. **Э. Таненбаум.** Современные операционные системы. 2-е изд. – СПб.: Питер, 2006. – 1038 с.: ил.

1. Обзор аппаратного обеспечения компьютера

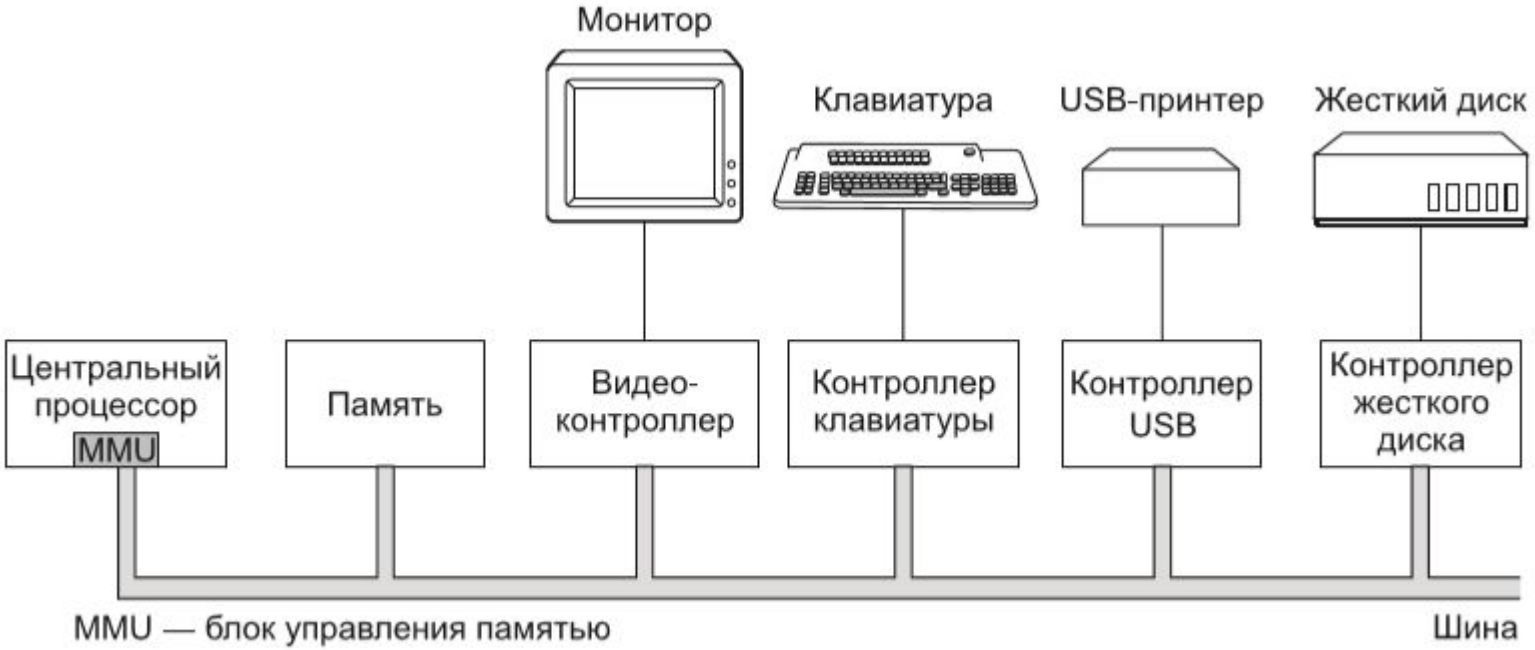
Операционная система тесно связана с аппаратным обеспечением компьютера, на котором она работает. Она расширяет набор команд компьютера и управляет его ресурсами.

Чтобы операционная система заработала, нужны глубокие познания в компьютерном оборудовании, по крайней мере нужно представлять, в каком виде оно предстает перед программистом.

Кратко рассмотрим аппаратное обеспечение, входящее в состав современного персонального компьютера.
После этого мы сможем приступить к подробному изучению того, чем занимаются операционные системы и как они работают.

1. Обзор аппаратного обеспечения компьютера

Простой **персональный компьютер** можно представить в виде следующей модели.



Центральный процессор, память и устройства ввода-вывода соединены системной шиной, по которой они обмениваются информацией друг с другом. Современные персональные компьютеры имеют более сложную структуру и используют несколько шин.

1. Обзор аппаратного обеспечения компьютера. 1.1. Процессор

Центральный процессор — это «мозг» компьютера, выполняющий программы. Он выбирает команды последовательно из памяти и выполняет их.

1. Цикл работы центрального процессора выглядит так:

- выборка из памяти первой команды,
- декодирование команды для определения ее типа и операндов,
- выполнение этой команды,
- а затем выборка, декодирование и выполнение последующих команд.

Этот цикл повторяется до тех пор, пока не закончится программа.

1. Обзор аппаратного обеспечения компьютера.1.1. Процессор

2. **Для каждого типа центрального процессора существует определенный набор команд, которые он может выполнять**. Поэтому x86 не может выполнять программы, написанные для ARM-процессоров, а те, в свою очередь, не в состоянии выполнять программы, написанные для x86.

3. Поскольку доступ к памяти для получения команды или данных занимает намного больше времени, чем выполнение команды, **у всех центральных процессоров есть несколько собственных регистров для хранения основных переменных и промежуточных результатов**.

4. **Процессоры первого поколения выполняли одну команду за один такт**

1. Обзор аппаратного обеспечения компьютера. 1.1. Процессор

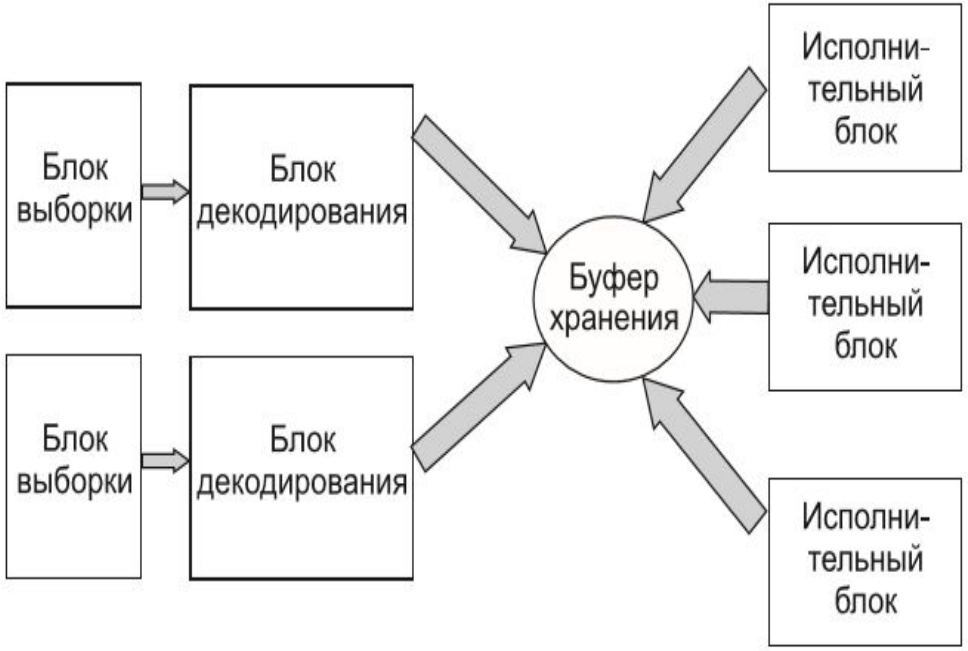
5. **Конвейерная архитектура процессоров.** Для повышения производительности процессоров разработчики давно отказались от простой модели извлечения, декодирования и выполнения одной команды за один цикл. Современные процессоры способны одновременно выполнять более одной команды.

У конвейерного процессора имеются отдельные блоки для выборки, декодирования и выполнения команд, тогда во время выполнения команды n он сможет декодировать команду n + 1 и осуществлять выборку команды n + 2.

В большинстве конструкций конвейеров, как только команда выбрана и помещена в конвейер, она должна быть выполнена, даже если предыдущая выбранная команда была условным ветвлением. Для разработчиков компиляторов и операционных систем конвейеры — это сплошная головная боль, обнажающая перед ними все сложности исходной машины и заставляющая справляться с возникающими проблемами.

1. Обзор аппаратного обеспечения компьютера. 1.1.Процессор

6. **Суперскалярный процессор** – имеет множество выполняющихся узлов



Для целочисленных операций

Для операций с плавающей точкой

Для логических операций

За один такт считываются две или более команды, которые помещаются в буфер хранения, где ожидают обработки на соответствующем процессоре.

1. Обзор аппаратного обеспечения компьютера. 1.1.Процессор

Большинство центральных процессоров имеют два режима работы: **режим ядра** и **режим пользователя**.

При работе **в режиме ядра** процессор может выполнять любые команды из своего набора и использовать любые возможности аппаратуры

.

ОС обычно работает в режиме ядра, что дает ей доступ ко всему оборудованию.

Пользовательские программы всегда работают в режиме пользователя, который допускает выполнение только подмножества команд и дает доступ к определенному подмножеству возможностей аппаратуры.

1. Обзор аппаратного обеспечения компьютера. 1.1.Процессор

В пользовательском режиме запрещены все команды, касающиеся операций ввода-вывода и защиты памяти.

Для получения услуг от операционной системы пользовательская программа должна осуществить **системный вызов**, который осуществляет переключение из пользовательского режима в режим ядра и запускает операционную систему.

Когда обработка вызова будет завершена, управление возвращается пользовательской программе и выполняется команда, которая следует за системным вызовом.

1. Обзор аппаратного обеспечения компьютера. 1.2.Память

Память - вторая основная составляющая компьютера.

В идеале память должна быть

1) максимально быстрой (работать быстрее, чем производится выполнение одной инструкции, чтобы работа центрального процессора не замедлялась обращениями к памяти),
2) довольно большой
3) чрезвычайно дешевой.

Никакая современная технология не в состоянии удовлетворить эти 3 требования, поэтому используется другой подход.

1. Обзор аппаратного обеспечения компьютера. 1.2.Память

Система памяти создается в виде иерархии уровней. Верхние уровни обладают более высоким быстродействием, меньшим объемом и более высокой удельной стоимостью хранения одного бита информации, чем нижние уровни,

1. Обзор аппаратного обеспечения компьютера.
1.2.Память

. Верхний уровень ОП состоит из **внутренних регистров процессора**, выполненых по той же технологии, что и сам процессор, и поэтому не уступающих ему в быстродействии. Следовательно, к ним нет и задержек доступа.

2. **Кэш-память**, которая управляется главным образом аппаратурой. Оперативная память разделяется на кэш-строки, обычно по 64 байт, с адресами от 0 до 63 в кэш-строке 0, адресами от 64 до 127 в кэш-строке 1 и т. д. Наиболее интенсивно используемые кэш-строки оперативной памяти сохраняются в высокоскоростной кэшпамяти, находящейся внутри процессора или очень близко к нему. Когда программе нужно считать слово из памяти, аппаратура кэша проверяет, нет ли нужной строки в кэш-памяти. Если строка в ней имеется, то происходит результативное обращение к кэш-памяти (cache hit — кэш-попадание), запрос удовлетворяется за счет кэш-памяти без отправки запроса по шине к оперативной памяти. Обычно результативное обращение к кэшу занимает по времени два такта. Отсутствие слова в кэш-памяти вынуждает обращаться к оперативной памяти, что приводит к существенной потере времени.

1. Обзор аппаратного обеспечения компьютера. 1.2.Память

Кэш-память из-за своей высокой стоимости ограничена в объеме. Некоторые машины имеют два или даже три уровня кэша, причем каждый из последующих медленнее и объемнее предыдущего.

Кэширование играет существенную роль во многих областях информатики, это относится не только к кэшированию строк оперативной памяти.

Довольно часто для повышения производительности к кэшированию прибегают везде, где есть какой-либо объемный ресурс, который можно поделить на фрагменты, часть из которых используется намного интенсивнее всех остальных.

Операционные системы используют кэширование повсеместно. Большинство ОС держат интенсивно используемые файлы (или фрагменты файлов) в оперативной памяти, избегая их многократного считывания с диска.

Может кэшироваться для дальнейшего использования результат преобразования адресов веб-страниц (URL) в сетевые адреса (IP-адреса). Можно привести массу других примеров использования технологии кэширования.

1. Обзор аппаратного обеспечения компьютера. 1.3. Устройства ввода-вывода

Устройства ввода-вывода обычно состоят из двух компонентов: самого устройства и его контроллера.

Контроллер - микросхема или набор микросхем, управляющих устройством на физическом уровне. Он принимает от ОС команды, например считать данные с помощью устройства, а затем их выполняет. Непосредственное управление устройством сложно и требует высокого уровня детализации, поэтому задачей контроллера является предоставление ОС простого интерфейса.

Поэтому контроллеры содержат маленькие встроенные компьютеры, запрограммированные на конкретные задачи по чтению-записи данных..

Устройства имеют довольно простые интерфейсы, поскольку они, во-первых, обладают весьма скромными возможностями, а во-вторых, должны отвечать общим стандартам. Соблюдение стандарта необходимо для того, чтобы, к примеру, любой контроллер SATA-диска смог работать с любым SATA-диском.

1. Обзор аппаратного обеспечения компьютера. 1.3. Устройства ввода-вывода

Поскольку интерфейс устройства скрыт его контроллером, все операционные системы видят только интерфейс контроллера, который может существенно отличаться от интерфейса самого устройства.

Так как все типы контроллеров отличаются друг от друга, для управления ими требуется различное программное обеспечение. **Программа, предназначенная для общения с контроллером, выдачи ему команды и получения поступающих от него ответов, называется драйвером устройства**.

Каждый производитель контроллеров поставляет вместе с ними драйверы для каждой поддерживаемой операционной системы. Например, сканер может поступить в продажу с драйверами для операционных систем OS X, Windows 7, Windows 8 и Linux.

Для использования драйвер нужно поместить в операционную систему, предоставив ему тем самым возможность работать в режиме ядра.

2. Основные виды ресурсов ОС

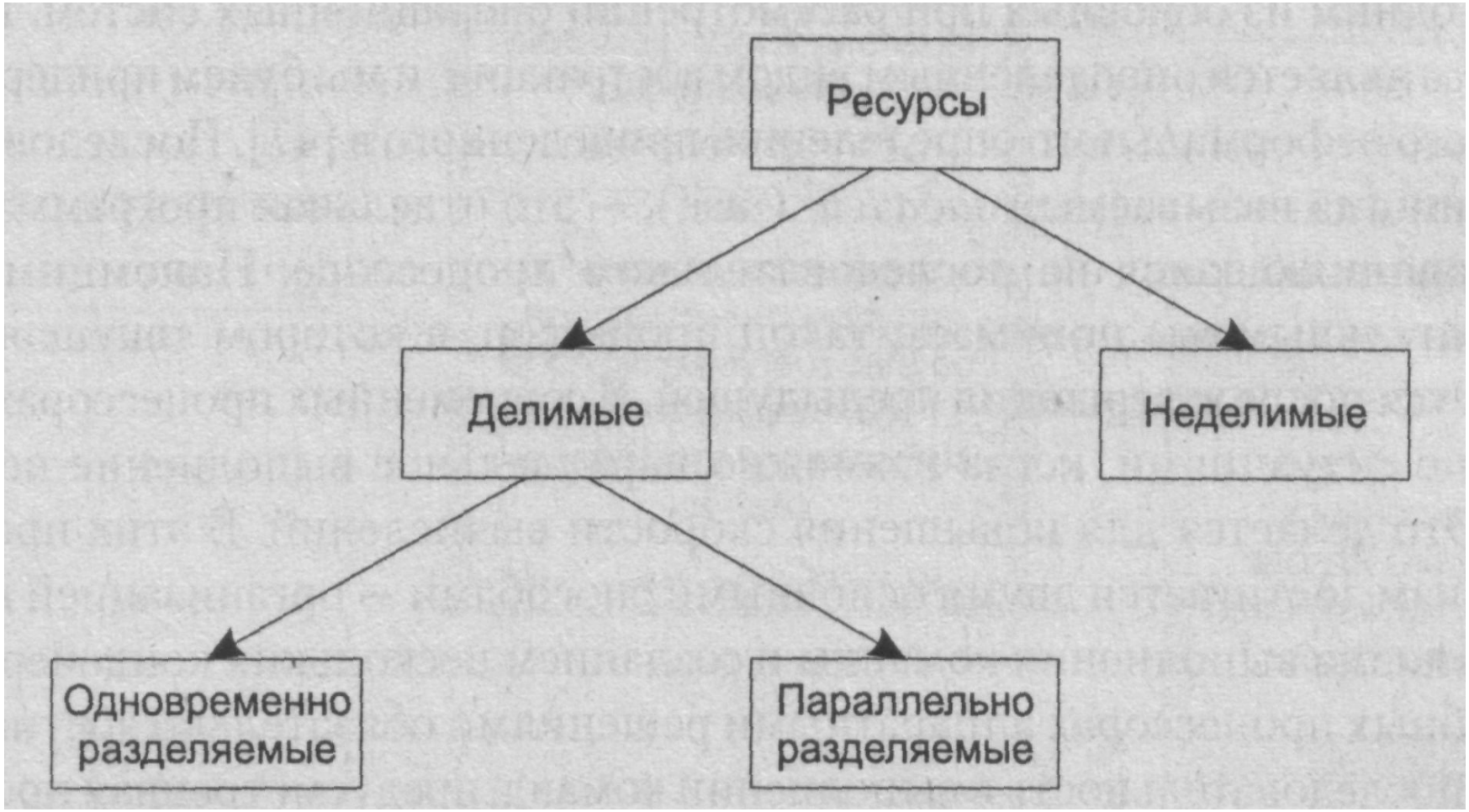
Ресурсом называется всякий объект, который может распределяться внутри системы.

Ресурс это средство вычислительной системы, которое может быть выделено процессу на определенный интервал времени.

Потребители ресурсов – процессы.

2. Основные виды ресурсов ОС

(по характеру использования)



2. Основные виды ресурсов ОС

Основные виды ресурсов в системе:

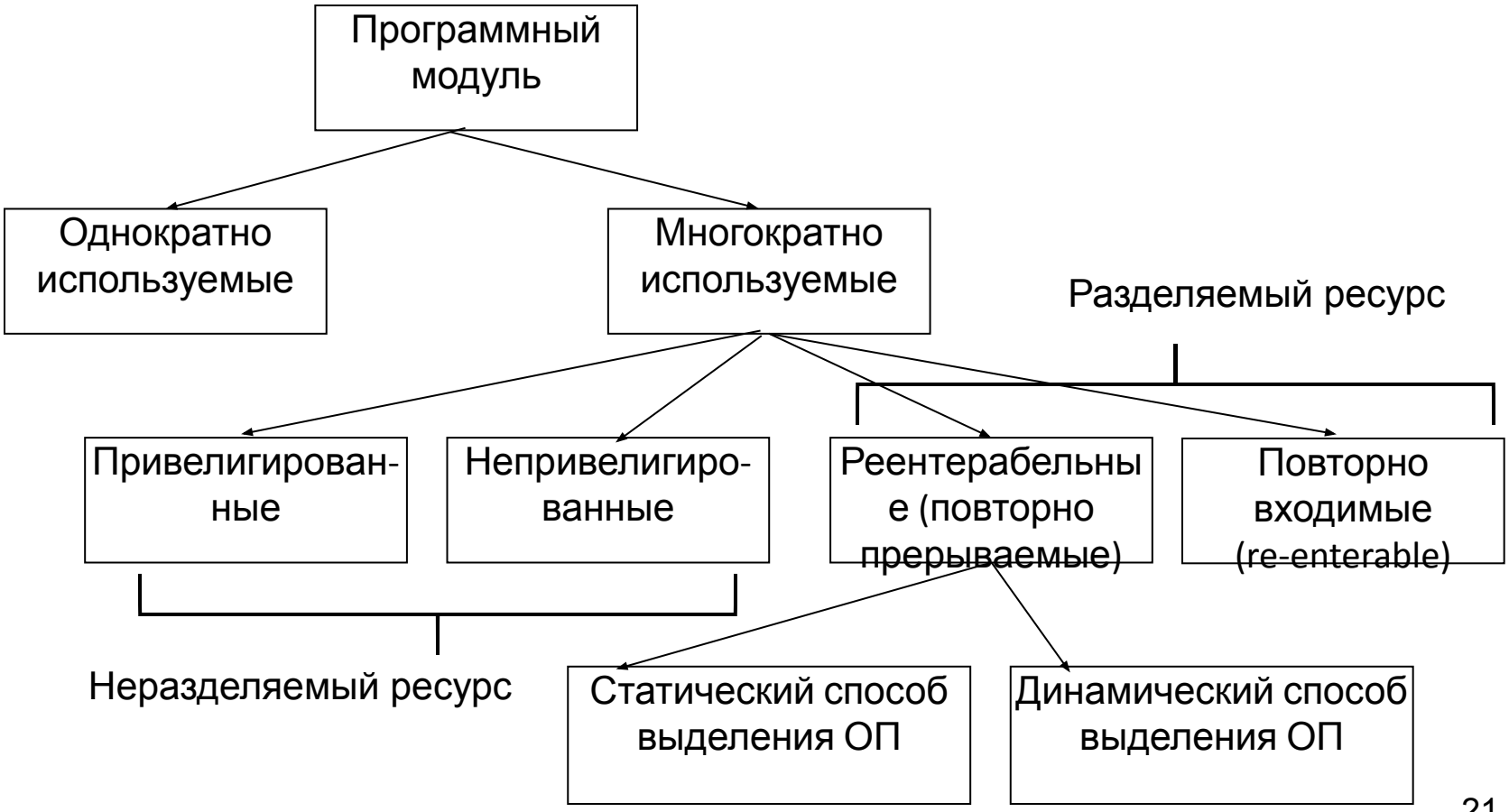
- ***процессорное время*** (*параллельно разделяемый*);
- ***оперативная память*** (делиться и одновременно, и параллельно);
- ***внешние устройства:***
  - механизм *прямого доступа* (HDD, флэш-память): доступ - параллельно разделяемый ресурс, память - одновременно разделяемый ресурс;
  - механизм *последовательного доступа* (принтер, НМЛ) - неделимый ресурс.
- ***программные*** (системные) ***модули*** – могут быть как делимые, так и неделимые;
- ***информационные ресурсы*** (переменные) – могут как делимые, так и неделимые.

3. Программные модули и возможности их разделения

Программный (системный) модуль это функционально законченный элемент программы (системы), выполненный в соответствии с принятыми межмодульными интерфейсами.

3. Программные модули и возможности их разделения

Классификация программных модулей



3. Программные модули и возможности их разделения.

3.1. Классификация программных модулей

Однократно используемыми называют такие программные модули, которые могут быть правильно выполнены только один раз, то есть в процессе своего выполнения они могут изменить себя: повреждается либо часть кода, либо исходные данные, от которых зависит ход вычислений. Это *неделимый ресурс*. Системные однократно используемые программные модули, как правило, задействуются только на этапе загрузки ОС.

Многократно (повторно) используемые допускают корректное повторное выполнение программного кода при обращении к нему из другой программы.

Привилегированные модули

Привилегированными программными модулями называются такие модули, которые работают в привилегированном режиме, то есть при отключенной системе прерываний, когда никакие внешние события не могут нарушить естественный порядок вычислений. Относятся к *неразделяемым* ресурсам.

Отключение прерываний	Собственное тело программного модуля	Включение прерываний

Структура привилегированного программного модуля

Непривилегированные модули

Непривилегированными программными называются такие модули, которые могут быть прерваны во время своей работы. Следовательно, такие модули в общем случае являются *неразделяемыми*, потому что ***если*** после прерывания выполнения такого модуля, исполняемого в рамках одного вычислительного процесса, запустить его еще раз по требованию другого вычислительного процесса, то промежуточные результаты для прерванных вычислений *могут быть потеряны*.

Реентерабельные модули

Реентерабельными *(повторно прерываемыми)* ***программными модулями*** называются такие модули, которые допускают повторное многократное прерывание своего исполнения и повторный их запуск по обращению из других задач (вычислительных процессов) до завершения обработки предыдущего запроса (сл. слайд).

Внутри реентерабельного модуля могут находиться ***несколько текущих точек*** выполнения этой программы, т.е. несколько вычислительных ***процессов*** могут обладать этим модулем.

Порождаемые из реентерабельной программы процессы обладают **общими** сегментами кода, но **разными** сегментами данных.

Пример программ – драйверы в/в.

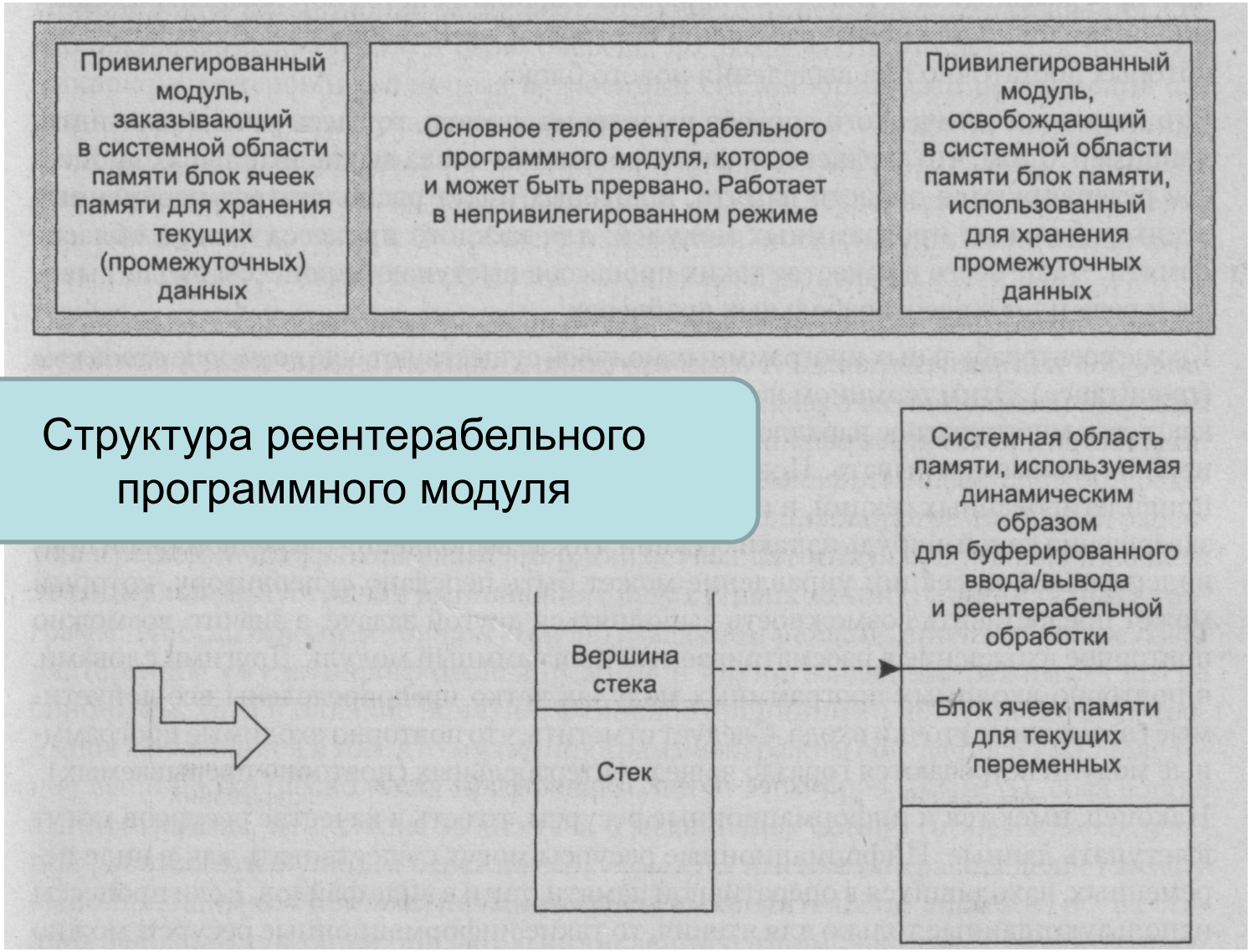
Реентерабельные модули

Для обеспечения реентерабельности процедуры **должны удовлетворять следующим требованиям**:

1. Код процедуры не может быть самомодифицирующимся.
2. Сегмент данных процедуры может содержать только константы.
3. Области памяти под переменные процедура должна получать из вызывающего модуля с помощью заранее оговоренного регистра процессора, указывающего на эту область. При соблюдении этого требования переключение контекстов операционной системой приведет к переустановке указателей между областями памяти, содержащими данные разных входов в процедуру.

Требование реентерабельности программ для организации их совместного использования не является обязательным: достаточно предпринять меры, исключающие повторный вход. Решение этого вопроса - синхронизация (взаимное исключение) работы процессов и потоков.

Реентерабельные модули



Структура реентерабельного программного модуля

Реентерабельные модули

Сегменты данных предоставляются системой двумя способами:

1. *Статический метод выделения памяти.*

Заранее для фиксированного числа вычислительных процессов резервируются области памяти, в которых будут располагаться переменные реентерабельных программных модулей: для каждого процесса — своя область памяти.

2. *Динамический метод выделения памяти.*

Все текущие переменные располагаются в системной области памяти (системный стек). Адресация этих переменных осуществляется относительно вершины стека. При повторном вхождении в реентерабельный программный модуль новые переменные записываются в свободную часть стека.

Повторно входимые модули

Повторно входимыми (re-entrance) называют программные модули, которые допускают свое многократное параллельное использование, но, в отличие от реентерабельных, их нельзя прерывать.

Повторно входимые программные модули **состоят из привилегированных секций**, и повторное обращение к ним возможно только после завершения какой-нибудь из таких секций.

В повторно входимых программных модулях четко предопределены все **допустимые (возможные) точки входа**.

Повторно входимые программные модули встречаются гораздо чаще реентерабельных.

Контрольные вопросы

1. Нарисуйте и поясните модель персонального компьютера
2. Объясните принцип работы современного компьютера. Что такое конвейерные вычисления? Что такое скалярный процессор?
3. Объясните принцип работы современного компьютера. Что такое привелигированный режим работы? Что такое пользовательский режим работы?
4. Нарисуйте и поясните пирамиду памяти компьютера. Что такое кеширование?
5. Что такое контроллер и что такое драйвер внешнего устройства?

Контрольные вопросы

6. Объясните понятие ресурса. Почему понятие ресурса является одним из фундаментальных при рассмотрении операционных систем?
7. Какие виды и типы ресурсов вы знаете? Приведите примеры делимых и неделимых ресурсов
8. Приведите пример внешнего устройства с *прямым* и *последовательным* доступом.
9. Что такое *привилегированный программный модуль*?
10. Что такое *реентерабельный программный модуль*?
11. Что такое *повторно входимый программный модуль?*