

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ  
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ОРЕНБУРГСКИЙ ГОСУДАРСТВЕННЫЙ АГРАРНЫЙ УНИВЕРСИТЕТ»**

**МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ДЛЯ ОБУЧАЮЩИХСЯ  
ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ**

**Б1.В.03 Операционные системы**

**Специальность** 10.05.03 Информационная безопасность автоматизированных систем

**Специализация** Информационная безопасность автоматизированных систем критически  
важных объектов

**Форма обучения** очная

## СОДЕРЖАНИЕ

<b>1. Конспект лекций.....</b>	<b>5</b>
1.1 Лекция №1 Общие сведения об операционных системах .....	5
1.2 Лекция №2 Аппаратное обеспечение вычислительных систем. Управляющие подсистемы ОС.....	6
1.3 Лекция №3 Архитектура операционных систем. Микроядерная архитектура.....	8
1.4 Лекция №4 Аппаратная зависимость ОС. Переносимость операционных систем	11
1.5 Лекция №5 Совместимость ОС. Интерфейс пользователя .....	13
1.6 Лекция №6 Мультипрограммирование. Мультипроцессорная обработка.....	15
1.7 Лекция №7 Обработка прерываний. Планирование процессов.....	16
1.8 Лекция №8 Диспетчеризация процессов. Синхронизация потоков.....	19
1.9 Лекция №9 Организация памяти. Сегментация памяти.....	21
1.10 Лекция №10 Алгоритмы организации памяти.....	23
1.11-1.12 Лекция №11-12 Основные концепции организации ввода-вывода. Организация ввода-вывода.....	26
1.13 Лекция №13 Установка и настройка операционной системы .....	31
1.14 Лекция №14 Использование сетевых возможностей. Обеспечение работоспособности системы.....	35
1.15-1.16 Лекция №15-16 Основные понятия безопасности. Обеспечение безопасности	40
<b>2. Методические указания по проведению лабораторных работ .....</b>	<b>42</b>
2.1 Лабораторная работа №ЛР-1-2 Общие сведения об операционных системах .....	42
2.2 Лабораторная работа №ЛР-3-4 Аппаратное обеспечение вычислительных систем. Управляющие подсистемы ОС.....	43
2.3 Лабораторная работа №ЛР-5-6 Архитектура операционных систем. Микроядерная архитектура .....	44
2.4 Лабораторная работа №ЛР-7-8 Совместимость ОС.....	45
2.5 Лабораторная работа №ЛР-9-10 Мультипрограммирование .....	45
2.6 Лабораторная работа №ЛР-11-12 Обработка прерываний.....	47
2.7 Лабораторная работа №ЛР-13-14 Диспетчеризация процессов.....	49
2.8 Лабораторная работа №ЛР-15-16 Синхронизация потоков .....	50
2.9 Лабораторная работа №ЛР-17-18 Организация памяти.....	50
2.10 Лабораторная работа №ЛР-19-20 Алгоритмы организации памяти.....	50
2.11 Лабораторная работа №ЛР-21-22 Основные концепции организации ввода-вывода	51
2.12 Лабораторная работа №ЛР-23-24 Организация ввода-вывода.....	52

<b>2.13 Лабораторная работа №ЛР-25-26 Установка и настройка операционной системы</b>	<b>52</b>
<b>2.14 Лабораторная работа №ЛР-27-28 Использование системы.....</b>	<b>53</b>
<b>2.15 Лабораторная работа №ЛР-29-30 Основные понятия безопасности .....</b>	<b>54</b>
<b>2.16 Лабораторная работа №ЛР-31-34 Обеспечение безопасности .....</b>	<b>54</b>

## **1. КОНСПЕКТ ЛЕКЦИЙ**

### **1.1 Лекция № 1 (2 часа).**

**Тема:** «Общие сведения об операционных системах»

#### **1.1.1 Вопросы лекции:**

1. Понятие, назначение и основные функции операционных систем
2. Основные принципы построения операционных систем
3. Классификация операционных систем
4. Понятие операционного окружения
5. Требования к операционным системам

#### **1.1.2 Краткое содержание вопросов:**

##### **1.Понятие, назначение и основные функции ОС**

Операционная система — это комплекс взаимосвязанных системных программ, назначение которого — организовать взаимодействие пользователя с компьютером и выполнение всех других программ.

Операционная система выполняет роль связующего звена между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем, с другой стороны.

Операционная система обычно хранится во внешней памяти компьютера — на диске. При включении компьютера она считывается с дисковой памяти и размещается в ОЗУ.

Назначение операционной системы.

Основная цель ОС, обеспечивающей работу ЭВМ в любом из описанных режимов, - динамическое распределение ресурсов и управление ими в соответствии с требованиями вычислительных процессов (задач).

В функции операционной системы входит:

- Осуществление диалога с пользователем;
- Ввод-вывод и управление данными;
- Планирование и организация процесса обработки программ;
- Распределение ресурсов (оперативной памяти и кэша, процессора, внешних устройств);
- Запуск программ на выполнение;
- Всевозможные вспомогательные операции обслуживания;
- Передача информации между различными внутренними устройствами;
- Программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).

## **2.Основные принципы построения ОС**

Среди множества принципов, которые используются при построении ОС, перечислим несколько наиболее важных.

Принцип модульности

Под модулем в общем случае понимают функционально законченный элемент системы, выполненный в соответствии с принятыми межмодульными интерфейсами. Принцип модульности отражает технологические и эксплуатационные свойства системы. Наибольший эффект от его использования достижим в случае, когда принцип распространен одновременно на операционную систему, прикладные программы и аппаратуру.

#### Принцип функциональной избирательности

В ОС выделяется некоторая часть важных модулей, которые должны постоянно находиться в оперативной памяти для более эффективной организации вычислительного процесса. Эту часть в ОС называют ядром, так как это действительно основа системы.

#### Принцип генерируемости ОС

Основное положение этого принципа определяет такой способ исходного представления центральной системной управляющей программы ОС (ее ядра и основных компонентов, которые должны постоянно находиться в оперативной памяти), который позволял бы настраивать эту системную супервизорную часть, исходя из конкретной конфигурации конкретного вычислительного комплекса и круга решаемых задач. Эта процедура проводится редко, перед достаточно протяженным периодом эксплуатации ОС.

#### Принцип функциональной избыточности

Принцип функциональной избыточности: Этот принцип учитывает возможность проведения одной и той же работы различными средствами.

#### Принцип виртуализации

Принцип виртуализации: построение виртуальных ресурсов, их распределение и использование в настоящее время применяется практически в любой ОС. Этот принцип позволяет представить структуру системы в виде определенного набора планировщиков процессов и распределителей ресурсов (мониторов) и использовать единую централизованную схему распределения ресурсов.

#### Принцип совместимости

Одним из аспектов совместимости является способность ОС выполнять программы, написанные для других ОС или для более ранних версий данной ОС, а также для другой аппаратной платформы. Необходимо разделять вопросы двоичной совместимости и совместимости на уровне исходных текстов приложений.

#### Принцип мобильности

Принцип мобильности: операционная система относительно легко должна переноситься с процессора одного типа на процессор другого типа и с аппаратной платформы одного типа, которая включает наряду с типом процессора и способ организации всей аппаратуры компьютера (архитектуру вычислительной системы), на аппаратную платформу другого типа.

#### Принцип обеспечения безопасности вычислений

Принцип обеспечения безопасности вычислений: обеспечение безопасности при выполнении вычислений является желательным свойством для любой многопользовательской системы.

### **3. Классификация операционных систем.**

Операционные системы классифицируются по:

- Количеству одновременно работающих пользователей: однопользовательские, многопользовательские;
- Числу процессов, одновременно выполняемых под управлением системы: однозадачные, многозадачные;
- Количеству поддерживаемых процессоров: однопроцессорные, многопроцессорные;
- Разрядности кода ОС: 8-разрядные, 16-разрядные, 32-разрядные, 64-разрядные;
- Типу интерфейса: командные (текстовые) и объектно-ориентированные (графические);
- Типу доступа пользователя к ЭВМ: с пакетной обработкой, с разделением времени, реального времени;
- Типу использования ресурсов: сетевые, локальные.

**4. Операционное окружение** — среда, в которой пользователь запускает программу. Например, операционное окружение DOS состоит из всех команд DOS, доступных пользователю. С другой стороны, операционное окружение Macintosh включает в себя графический интерфейс пользователя, использующий иконки и меню вместо команд.

В различных операционных системах состав среды окружения может сильно различаться.

К составу среды окружения обычно относят:

- Системные переменные
- Текущие пути на различных дисках (в случае поддержки нескольких дисков операционной системой)
- Точка монтирования каталогов (в том числе корневого), используется в unix-подобных операционных системах для обеспечения режима «тюрьмы» (англ. jail)
- Связь стандартных потоков ввода-вывода с файловыми хэндлерами или устройствами (используется для перенаправления ввода-вывода)
- Ограничения на количество одновременно открытых файлов, стеков и т. д.
- Набор прав (обычно соответствует правам пользователя, запустившего процесс, но может изменяться как в сторону большего набора прав, так и в сторону ужесточения)
- Дисковые квоты, ограничение на максимальный объем оперативной памяти, загрузки процессоров и т. д.
- Значения показателей использования ресурсов, получаемые от родительского процесса (на некоторых системах).

## **5. Требования к операционным системам**

### Расширяемость

Если код ОС написан таким образом, что дополнения и изменения могут вноситься без нарушения целостности системы, то такую ОС называют расширяемой.

### Переносимость или многоплатформенность

Код ОС должен легко переноситься с процессора одного типа на процессор другого типа и с аппаратной платформы одного типа на аппаратную платформу другого типа.

### Совместимость

Если ОС имеет средства для выполнения прикладных программ, написанных для других операционных систем, то про нее говорят, что она обладает совместимостью с этими ОС.

#### Надежность и отказоустойчивость

Система должна быть защищена как от внутренних, так и от внешних ошибок, сбоев и отказов.

#### Безопасность

Современная ОС должна защищать данные и другие ресурсы вычислительной системы от несанкционированного доступа.

#### Производительность

Операционная система должна обладать настолько хорошим быстродействием и временем реакции, насколько это позволяет аппаратная платформа.

### **1.2 Лекция № 2(2 часа).**

**Тема:** «Управляющие подсистемы ОС»

#### **1.1.1 Вопросы лекции:**

1. Состав системы (процессор, память, устройства ввода-вывода, контролеры внешних устройств, системная шина)
2. Организация вычислительных процессов.
3. Управление файлами
4. Управление устройствами
5. Управление пользователями
6. Управление памятью
7. Управление процессами

#### **1.1.2 Краткое содержание вопросов:**

**1. Состав системы (процессор, память, устройства ввода-вывода, контролеры внешних устройств, системная шина).**

##### Процессор

«Мозгом» компьютера является центральный процессор (CPU — CentralProcessingUnit). Он выбирает из памяти команды и выполняет их.

##### Память

Второй основной составляющей любого компьютера является память. В идеале память должна быть максимально быстрой (быстрее, чем обработка одной инструкции, чтобы работа центрального процессора не замедлялась обращениями к памяти), достаточно большой и чрезвычайно дешевой. Системы памяти конструируются в виде иерархии слоев.

##### Устройства ввода-вывода

Устройства ввода-вывода также тесно взаимодействуют с операционной системой. Устройства ввода-вывода обычно состоят из двух частей: контроллера и самого устройства.

##### Контролеры внешних устройств

В составе контроллера можно выделить четыре внутренних регистра, называемых регистрами состояния, управления, входных данных и выходных данных. Для доступа к содержимому этих регистров вычислительная система может использовать один или несколько портов.

##### Системная шина

В основе устройства ЭВМ лежит системная шина, которая служит для обмена командами и данными между компонентами ЭВМ, расположенными на материнской плате.

## **2. Организация вычислительных процессов.**

Процедура организации вычислительного процесса обладает различной функциональной сложностью в зависимости от класса и количества решаемых задач, режимов обработки данных, топологии системы обработки данных. В наиболее полном объеме функции организации вычислительного процесса реализуются при обработке данных на больших универсальных машинах (мейнфреймах), которые, как правило, работают в многопользовательском режиме и обладают большими объемами памяти и высокой производительностью. При обработке данных с помощью ЭВМ в зависимости от конкретного применения информационной технологии, а значит, и решаемых задач различают три основных режима: пакетный, разделения времени, реального времени.

## **3. Управление файлами**

Файл – именованная область внешней памяти, предназначенная для считывания и записи данных.

Файлы хранятся в памяти, не зависящей от энергопитания. Исключением является электронный диск, когда в ОП создается структура, имитирующая файловую систему.

Файловая система (ФС) — это компонент ОС, обеспечивающий организацию создания, хранения и доступа к именованным наборам данных - файлам.

Файловая система включает:

Совокупность всех фалов на диске.

Наборы структур данных, используемых для управления файлами (каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске).

Комплекс системных программных средств, реализующих различные операции над файлами: создание, уничтожение, чтение, запись, именование, поиск.

Задачи, решаемые ФС, зависят от способа организации вычислительного процесса в целом. Самый простой тип – это ФС в однопользовательских и однопрограммных ОС. Основные функции в такой ФС нацелены на решение следующих задач:

Именование файлов.

Программный интерфейс для приложений.

## **4. Управление устройствами**

Подсистема управления внешними устройствами, называемая также подсистемой ввода-вывода, является интерфейсом ко всем устройствам, подключенным к компьютеру. Множество этих устройств очень велико. Номенклатура выпускаемых накопителей на жестких, гибких и оптических дисках принтеров, сканеров, мониторов, плоттеров, модемов, сетевых адаптеров и специализированных устройств ввода-вывода (например, аналого-цифровых преобразователей, устройств считывания штрих-кода и т.д.) насчитывает тысячи моделей.

## **5. Управление пользователями**

Пользователи — это основа вашей конференции. Вам, как администратору конференции, очень важно уметь управлять вашими пользователями. Управление

пользователями и их информационными данными просто и доступно в панели администратора.

Для начала войдите в панель администратора и перейдите в раздел «Пользователи и группы». Также можно щёлкнуть ссылку слева «Управление пользователями» на главной странице панели администрирования.

Для продолжения управления пользователем вам необходимо знать имя пользователя, которым вы хотите управлять. В текстовое поле «Найти пользователя» введите имя пользователя, информацией и настройками которого вы хотите управлять. С другой стороны, если вы хотите найти пользователя, нажмите ссылку «Найти пользователя» (которая находится ниже текстового поля), и следуйте за всеми шагами для поиска и выбора пользователя. Если вы хотите управлять информацией и настройками анонимного пользователя (любой посетитель, не вошедший на конференцию под своим именем, является анонимным посетителем, или гостем), то поставьте флажок «Выбрать учётную запись гостя». После выбора пользователя нажмите кнопку «Отправить».

## **6. Управление памятью**

На подсистему управления памятью возложены очень важные функции, поскольку процесс может выполняться только в том случае, если его коды и данные находятся в оперативной памяти компьютера. Управление памятью включает в себя распределение имеющейся в вычислительной системе физической памяти между всеми существующими в данный момент в системе процессами, загрузку кодов и данных процессов в отведенные им области памяти, настройку адресно-зависимых частей кодов процесса на физические адреса выделенной области, а также защиту областей памяти каждого процесса. Одним из наиболее популярных способов управления памятью в современных ОС является так называемая виртуальная память.

Защита памяти – это избирательная способность предохранять выполняемую задачу от операций записи или чтения памяти, назначенной другой задаче.

## **7. Управление процессами**

Важнейшей частью операционной системы, непосредственно влияющей на функционирование вычислительной машины, является подсистема управления процессами.

Для каждого вновь создаваемого процесса ОС генерирует системные информационные структуры, которые содержат данные о потребностях процесса в ресурсах вычислительной системы, а также о фактически выделенных ему ресурсах. Таким образом, процесс можно также определить как некоторую заявку на потребление системных ресурсов.

### **1.3 Лекция №3 (2 часа).**

**Тема:** «Микроядерная архитектура»»

#### **1.3.1 Вопросы лекции:**

1. Структура операционных систем: монолитная, многоуровневая.
2. Модель экзядра.
3. Основные положения микроядерной архитектуры;
4. Преимущества и недостатки микроядерной архитектуры.

#### **1.3.1 Краткое содержание вопросов:**

- 1. Структура операционных систем: монолитная, многоуровневая.**



В общем случае "структура" монолитной системы представляет собой отсутствие структуры. ОС написана как набор процедур, каждая из которых может вызывать другие, когда ей это нужно. При использовании этой техники каждая процедура системы имеет хорошо определенный интерфейс в терминах параметров и результатов, и каждая вольна вызывать любую другую для выполнения некоторой нужной для нее полезной работы.

Для построения монолитной системы 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). Каждая процедура видит любую другую процедуру (в отличие от структуры, содержащей модули, в которой большая часть информации является локальной для модуля, и процедуры модуля можно вызвать только через специально определенные точки входа).

Однако даже такие монолитные системы могут быть немного структурированными. При обращении к системным вызовам, поддерживаемым ОС, параметры помещаются в строго определенные места, такие, как регистры или стек, а затем выполняется специальная команда прерывания, известная как вызов ядра или вызов супервизора. Эта команда переключает машину из режима пользователя в режим ядра, называемый также режимом супервизора, и передает управление ОС. Затем ОС проверяет параметры вызова для того, чтобы определить, какой системный вызов должен быть выполнен. После этого ОС индексирует таблицу, содержащую ссылки на процедуры, и вызывает соответствующую процедуру. Такая организация ОС предполагает следующую структуру:

- а) главная программа, которая вызывает требуемые сервисные процедуры;
- б) набор сервисных процедур, реализующих системные вызовы.
- в) набор утилит, обслуживающих сервисные процедуры.

Обобщением предыдущего подхода является организация ОС как иерархии уровней. Уровни образуются группами функций операционной системы - файловая система, управление процессами и устройствами и т.п. Каждый уровень может взаимодействовать только со своим непосредственным соседом - выше- или нижележащим уровнем. Прикладные программы или модули самой операционной системы передают запросы вверх и вниз по этим уровням.

Первой системой, построенной таким образом была простая пакетная система THE, которую построил Дейкстра и его студенты в 1968 году. Система имела 6 уровней. Уровень 0 занимался распределением времени процессора, переключая процессы по прерыванию или по истечении времени. Уровень 1 управлял памятью - распределял оперативную память и пространство на магнитном барабане для тех частей процессов (страниц), для которых не было места в ОП, то есть слой 1 выполнял функции виртуальной памяти. Слой 2 управлял связью между консолью оператора и процессами. С помощью этого уровня каждый процесс имел свою собственную консоль оператора. Уровень 3 управлял устройствами ввода-вывода и буферизовал потоки информации к ним и от них. С помощью уровня 3 каждый процесс вместо того, чтобы работать с конкретными устройствами, с их разнообразными особенностями, обращался к абстрактным устройствам ввода-вывода, обладающим удобными для пользователя характеристиками. На уровне 4 работали пользовательские программы, которым не надо было заботиться ни о процессах, ни о памяти, ни о консоли, ни об управлении устройствами ввода-вывода. Процесс системного оператора размещался на уровне 5.

В системе THE многоуровневая схема служила, в основном, целям разработки, так как все части системы компоновались затем в общий объектный модуль.

Дальнейшее обобщение многоуровневой концепции было сделано в ОС MULTICS. В системе MULTICS каждый уровень (называемый кольцом) является

более привилегированным, чем вышележащий. Когда процедура верхнего уровня хочет вызвать процедуру нижележащего, она должна выполнить соответствующий системный вызов, то есть команду TRAP (прерывание), параметры которой тщательно проверяются перед тем, как выполняется вызов. Хотя ОС в MULTICS является частью адресного пространства каждого пользовательского процесса, аппаратура обеспечивает защиту данных на уровне сегментов памяти, разрешая, например, доступ к одним сегментам только для записи, а к другим - для чтения или выполнения. Преимущество подхода MULTICS заключается в том, что он может быть расширен и на структуру пользовательских подсистем. Например, профессор может написать программу для тестирования и оценки студенческих программ и запустить эту программу на уровне  $n$ , в то время как студенческие программы будут работать на уровне  $n+1$ , так что они не смогут изменить свои оценки.

Многоуровневый подход был также использован при реализации различных вариантов ОС UNIX.

Монолитная система – это схема операционной системы, при которой все ее компоненты являются составными частями одной программы, используют общие структуры данных и взаимодействуют друг с другом путем непосредственного вызова процедур.

Многоуровневые системы – организация операционных систем в виде иерархии уровней. Первой многоуровневой системой была система THE, созданная в Technische Hogeschool Eindhoven (Нидерланды) Э. Дейкстроом (E.W. Dijkstra) и его студентами в 1968 г. Она была простой пакетной системой для голландского компьютера Electrologica X8, память которого состояла из 32 Кб 27-разрядных слов.

## **2. Модель экзоядра.**

В задачу экзоядра (exokernel) входит распределение ресурсов для виртуальных машин и проверка их использования (отслеживание попыток машин использовать чужой ресурс).

## **3. Основные положения микроядерной архитектуры**

Микроядерная архитектура является альтернативой рассмотренному выше классическому способу построения ОС. В отличие от традиционной архитектуры, согласно которой ОС представляет собой монолитное ядро, реализующее основные функции по управлению аппаратными ресурсами и организующее среду для выполнения пользовательских процессов, микроядерная архитектура распределяет функции ОС между микроядром и входящими в состав ОС системными сервисами, реализованными в виде процессов, равноправных с пользовательскими приложениями.

## **4. Преимущества и недостатки микроядерной архитектуры**

Операционные системы, основанные на концепции микроядра, в высокой степени удовлетворяют большинству требований, предъявляемых к современным ОС, обладая переносимостью, расширяемостью, надежностью и создавая хорошие предпосылки для поддержки распределенных приложений. За эти достоинства приходится платить снижением производительности, и это является основным недостатком микроядерной архитектуры.

Высокая степень переносимости обусловлена тем, что весь машинно-зависимый код изолирован в микроядре, поэтому для переноса системы на новый процессор требуется меньше изменений и все они логически сгруппированы вместе.

Расширяемость присуща микроядерной ОС в очень высокой степени. В традиционных системах даже при наличии многослойной структуры нелегко удалить один слой и поменять его на другой по причине множественности и размытости интерфейсов между слоями. Добавление новых функций и изменение существующих требует хорошего знания операционной системы и больших затрат времени. В то же время ограниченный набор четко определенных интерфейсов микроядра открывает путь к упорядоченному росту и эволюции ОС. Добавление новой подсистемы требует разработки нового приложения, что никак не затрагивает целостность микроядра. Микроядерная структура позволяет не только добавлять, но и сокращать число компонентов операционной системы, что также бывает очень полезно. Например, не всем пользователям нужны средства безопасности или поддержки распределенных вычислений, а удаление их из традиционного ядра чаще всего невозможно. Обычно традиционные операционные системы позволяют динамически добавлять в ядро или удалять из ядра только драйверы внешних устройств — ввиду частых изменений в конфигурации подключенных к компьютеру внешних устройств подсистема ввода-вывода ядра допускает загрузку и выгрузку драйверов «на ходу», но для этого она разрабатывается особым образом (например, среда STREAMS в UNIX или менеджер ввода-вывода в Windows NT). При микроядерном подходе конфигурируемость ОС не вызывает никаких проблем и не требует особых мер.

Использование микроядерной модели повышает надежность ОС. Каждый сервер выполняется в виде отдельного процесса в своей собственной области памяти и таким образом защищен от других серверов операционной системы, что не наблюдается в традиционной ОС, где все модули ядра могут влиять друг на друга. И если отдельный сервер терпит крах, то он может быть перезапущен без останова или повреждения остальных серверов ОС. Более того, поскольку серверы выполняются в пользовательском режиме, они не имеют непосредственного доступа к аппаратуре и не могут модифицировать память, в которой хранится и работает микроядро. Другим потенциальным источником повышения надежности ОС является уменьшенный объем кода микроядра по сравнению с традиционным ядром — это снижает вероятность появления ошибок программирования.

Производительность. При классической организации ОС выполнение системного вызова сопровождается двумя переключениями режимов, а при микроядерной организации — четырьмя. Таким образом, операционная система на основе микроядра при прочих равных условиях всегда будет менее производительной, чем ОС с классическим ядром. Именно по этой причине микроядерный подход не получил такого широкого распространения, которое ему предрекали.

#### **1.4 Лекция №4 (2 часа).**

**Тема: «Совместимость ОС»**

##### **1.4.1 Вопросы лекции:**

1. Виды совместимости
2. Способы реализации совместимости
3. Понятие программного интерфейса, его назначение
4. Виды интерфейсов (командный интерфейс ОС MSDOS, Linux)
5. Графический интерфейс (ОС Windows)

##### **1.4.2 Краткое содержание вопросов:**

#### **1. Виды совместимости**

Свойство ОС, характеризующее возможность выполнения в ОС приложений, написанных для других ОС, называется совместимостью. Существует 2 принципиально

отличающихся вида совместимости, которые не следует путать: совместимость на двоичном уровне и совместимость на уровне исходных текстов.

Приложения обычно хранятся в компьютере в виде исполняемых файлов, содержащих двоичные образы кодов и данных. Двоичная совместимость достигается в том случае, если можно взять исполняемую программу, работающую в среде одной ОС, и запустить ее на выполнение в среде другой ОС.

Совместимость на уровне исходных текстов требует наличия соответствующих компиляторов в составе программного обеспечения компьютера, на котором предполагается использовать данное приложение, а также совместимости на уровне библиотек и системных вызовов. При этом необходима перекомпиляция исходных текстов программ в новые исполняемые модули.

## **2.Способы реализации совместимости**

Несмотря на то, что реализация на практике полноценной прикладной среды, полностью совместимой со средой другой ОС, является очень сложной задачей, существует несколько типовых подходов к ее решению. Эти варианты отличаются в основном особенностями архитектурных решений и функциональными возможностями, обеспечивающими различную степень переносимости приложений.

## **3. Понятие программного интерфейса, его назначение**

Интерфейс — это внешняя оболочка приложения вместе с программами управления доступом и другими скрытыми от пользователя механизмами управления, дающая возможность работать с документами, данными и другой информацией, хранящейся в компьютере или за его пределами. Главная цель любого приложения — обеспечить максимальное удобство и эффективность работы с информацией: документами, базами данных, графикой или изображениями. Поэтому интерфейс является самой важной частью любого приложения

## **4. Виды интерфейсов (командный интерфейс ОС MSDOS, Linux)**

Интерфейсы различают по таким характеристикам, как структура связей, способ подключения и передачи данных, принципы управления и синхронизации.

Внутримашинный интерфейс — система связи и средств сопряжения узлов и блоков ЭВМ между собой. Внутримашинный интерфейс представляет собой совокупность электрических линий связи (проводов), схем сопряжения с компонентами компьютера, протоколов (алгоритмов) передачи и преобразования сигналов.

Внешний интерфейс — система связи системного блока с периферийными устройствами ЭВМ или с другими ЭВМ.

Пользовательский интерфейс — это способ, которым вы выполняете какую-либо задачу с помощью каких-либо средств (какой-либо программы), а именно совершаемые вами действия и то, что вы получаете в ответ.

Командный интерфейс — взаимодействие человека с компьютером осуществляется путем подачи компьютеру команд, которые он выполняет и выдает результат пользователю.

ООМУ (окно, образ, меню, указатель) WIMP (window, image, menu, pointer) - интерфейс. Характерной чертой этого интерфейса является то, что диалог пользователя с компьютером ведется не с помощью командной строки, а с помощью окон, графических образов меню, курсора и других элементов. Хотя в этом интерфейсе подаются команды машине, но это делается через графические образы.

РОЯЗ (речь, образ, язык, знания) SILK (speech, image, language, knowledge) - интерфейс. Этот интерфейс наиболее приближен к обычной человеческой форме общения. В рамках этого интерфейса идет обычный разговор человека и компьютера. При этом компьютер находит для себя команды, анализируя человеческую речь и находя в ней ключевые фразы. Результаты выполнения команд он также преобразует в понятную человеку форму.

## **5. Графический интерфейс ( OCWindows)**

Элементы графического интерфейса Windows:

### Рабочий стол

Название «Рабочий стол» подобрано удачно. На нем, как и на обычном рабочем столе расположены различные программы и инструменты, представленные в виде значков, или иконки.

### Значки

Значками в Windows обозначаются программы, документы. Запуск производится двойным щелчком кнопки мыши по значку. Программа может быть расположена непосредственно на Рабочем столе, а может быть скрыта глубоко на диске, но и в этом случае представлена на Рабочем столе своим образом – ярлыком.

### Ярлыки

Ярлык программы – это не сама программа, а только ее образ, указание на то место на диске, где она находится. Двойной щелчок по ярлыку также вызывает запуск программы. Ярлыки от значков отличаются наличием небольшой стрелочки внизу слева.

### Панель задач

Располагается в нижней части экрана. На ней находятся: кнопка Пуск, кнопки открытых окон, индикаторы и часы.

### Окно

Окно – один из главных элементов интерфейса Windows.

## **1.5 Лекция №5 (2 часа).**

**Тема:** «Мультипрограммирование»

### **1.5.1 Вопросы лекции:**

1. Мультипрограммирование в системе пакетной обработки
2. Мультипрограммирование в системе разделения времени
3. Мультипрограммирование в системе реального времени.

4. Понятие мультипроцессирования
5. Симметричное, Асимметричное мультипроцессирование

### **1.5.2 Краткое содержание вопросов:**

#### **1. Мультипрограммирование в системе пакетной обработки**

Мультипрограммирование в системах пакетной обработки

Рациональный режим функционирования в системах пакетной обработки достигается благодаря использованию следующей схемы функционирования: в начале работы формируется пакет заданий, каждое задание содержит требование к системным ресурсам; из этого пакета заданий формируется мультипрограммная смесь, т.е. множество одновременно выполняемых задач. Причем смесь строится таким образом, что одновременно должны выполняться задачи, предъявляющие разные требования к ресурсам с целью обеспечения сбалансированной загрузки всех устройств вычислительной машины.

#### **2. Мультипрограммирование в системах разделения времени**

В том случае, когда критерием эффективности работы системы является удобство работы пользователя, то применяется другой способ мультипрограммирования – разделения времени. В системах разделения времени пользователям (или одному пользователю) предоставляется возможность интерактивной работы сразу с несколькими приложениями. Для этого каждое приложение должно регулярно получать возможность “общения” с пользователем. Понятно, что в пакетных системах возможности диалога пользователя с приложением весьма ограничены.

#### **3. Мультипрограммирование в системах реального времени**

Еще одна разновидность мультипрограммирования используется в системах реального времени, предназначенных для управления с помощью компьютера различными техническими объектами (например, станком, спутником, научной экспериментальной установкой и т. д.) или технологическими процессами. Во всех этих случаях существует предельно допустимое время, в течение которого должна быть выполнена та или иная управляющая объектом программа. В противном случае может произойти авария. Таким образом, критерием эффективности здесь является время реакции системы.

#### **4. Понятие мультипроцессирования**

Мультипроцессорная обработка – это способ организации вычислительного процесса в системах с несколькими процессорами, при котором несколько задач (процессов, потоков) могут одновременно выполняться на разных процессорах системы.

Многопроцессорность теперь является одним из необходимых требований, которые предъявляются к компьютерам, используемым в качестве центрального сервера более-менее крупной сети.

#### **5. Симметричное, Асимметричное мультипроцессирование**

Мультипроцессорные системы можно разделить на два основных типа: симметричные и асимметричные.

Симметричная архитектура мультимикропроцессорной системы предполагает однородность всех процессоров и единообразие их включения в общую схему мультимикропроцессорной системы.

В асимметричной архитектуре разные процессоры могут отличаться как своими характеристиками (производительностью, надежностью, системой команд и т.д., вплоть до модели микропроцессора), так и функциональной ролью, которая поручается им в системе.

## **1.6. Лекция №6 (2 часа).**

**Тема: «Обработка прерываний»**

### **1.6.1. Вопросы лекции:**

1. Понятие прерывания. Последовательность действий при обработке прерываний.
2. Вектор прерывания. Функции диспетчера прерываний.
3. Процедуры обработки прерываний.
4. Системные вызовы
5. Понятия процесс, поток. Модель процесса, состояния процессов.
6. Поток (нити, облегченный процесс). Модель потока
7. Планирование потоков.

### **1.6.2 Краткое содержание вопросов:**

**1. Понятие прерывания. Последовательность действий при обработке прерываний.**

Прерывания являются основной движущей силой любой операционной системы. В зависимости от источника прерывания делятся на три больших класса: внешние, внутренние, программные.

### **2. Вектор прерывания. Функции диспетчера прерываний.**

Адреса программ, соответствующих различным прерываниям, собраны в таблицу, которая называется таблицей векторов прерываний.

### **3. Процедуры обработки прерываний.**

Механизм прерываний поддерживается аппаратными средствами компьютера и программными средствами операционной системы.

Существуют два основных способа, с помощью которых шины выполняют прерывания: векторный (vectored) и опрашиваемый (polled). В обоих способах процессору предоставляется информация об уровне приоритета прерывания на шине подключения внешних устройств. В случае векторных прерываний в процессор передается также информация о начальном адресе программы обработки возникшего прерывания – обработчика прерываний.

### **4. Системные вызовы**

Интерфейс между ОС и программами пользователя определяется набором системных вызовов (CB), предоставляемых ОС. Системные вызовы, доступные в интерфейсе, меняются от одной ОС к другой (хотя лежащая в их основе концепция практически одинакова).

### **5. Понятия процесс, поток. Модель процесса, состояния процессов.**

Создать процесс – это прежде всего означает создать описатель процесса, в качестве которого выступает одна или несколько информационных структур, содержащих все сведения о процессе, необходимые операционной системе для управления им.

Создание процесса включает загрузку кодов и данных исполняемой программы данного процесса с диска в оперативную память.

В многопоточной системе при создании процесса ОС создает для каждого процесса как минимум один поток выполнения. При создании потока так же, как и при создании процесса, операционная система генерирует специальную информационную структуру – **описатель потока**, который содержит идентификатор потока, данные о правах доступа и приоритете, о состоянии потока и другую информацию.

## **6. Потоки (нити, облегчённый процесс). Модель потока**

Каждому процессу соответствует адресное пространство и одиночный **поток** исполняемых команд. В многопользовательских системах, при каждом обращении к одному и тому же сервису, приходится создавать новый процесс для обслуживания клиента. Это менее выгодно, чем создать квазипараллельный поток внутри этого процесса с одним адресным пространством.

## **7. Планирование потоков**

Выбор текущего потока из нескольких активных потоков, пытающихся получить *доступ к процессору* называется *планированием*. Планирование - очень важная и критичная для производительности операция, поэтому система предоставляет много рычагов для ее гибкой настройки

### **1.7. Лекция №7 (2 часа).**

**Тема:** «Диспетчеризация процессов»

#### **1.7.1. Вопросы лекции:**

1. Диспетчеризация процесса.
2. Блок состояния процесса.
3. Алгоритмы диспетчеризации
4. Синхронизация потоков.
5. Методы синхронизации: взаимное исключение, блокирующие переменные.
6. Взаимоблокировка процессов. Моделирование взаимоблокировок.
7. Методы борьбы с взаимоблокировками

#### **1.7.2 Краткое содержание вопросов:**

1. Диспетчеризация процесса.

Процесс – это объект, которому выделен процессор, - это задача.

Фактический процессор (микропроцессор) – реально существующий процессор.

Виртуальный процессор – это процессор связанный с выполнением некоторого процесса.

Фактический процессор используется мультипрограммно, у пользователя создается иллюзия, что каждый процесс использует процессор независимо друг от друга. Считают, что с каждым процессом связан его собственный виртуальный процессор.

Распределение процессора снижает скорость выполнения работы. Скорость каждого виртуального процессора меньше скорости реального процессора.

### **Состояния процессов.**



1 состояние: исполнение, в нем находится только один процесс.

2 состояние: состояние готовности. Процессу выделены все ресурсы кроме фактического процессора.

3 состояние: заблокированное, процесс не готов к выполнению, он ждет какого-либо события.

Любой процесс может находиться в одном из состояний. Процессу, готовому к исполнению, выделены все ресурсы, в том числе и ОП. Он ждет только одного ресурса – процессора.

В заблокированном состоянии находится процесс, ожидающий некоторого события (отсутствия сегмента, завершение в/выв). Заблокированный процесс м.б откачен во внешнюю память.

Причины выхода процесса из состояния исполнения:

1) процесс завершился, он покидает виртуальный процессор.

2) процесс блокирует сам себя. Причина: запрос на ввод или вывод. Процесс перейдет из заблокированного состояния в состояние ожидания после завершения запрошенных действий.

3) процесс снимается с процессора управляющей программой и возвращается в состояние готовности.

Распределение и перераспределение фактического процессора и выделение виртуальных процессоров – это функция управления процессором.

Выбор работ – планирование, выполняется спец программой, которая называется планировщик (программа инициирования, инициатор). Он создает процесс для выбранной работы, переводит его в состояние готовности, выделяет ОП, внешнюю память, действия по завершению работы. Выделение фактического процессора готовому процессу – называется диспетчеризацией, выполняется спец программой – диспетчер (монитор). Диспетчеризация выполняется многократно. Планирование осуществляется один раз для каждой задачи, диспетчеризация – многократно.

Для исполнения диспетчер выбирает процесс, находящийся в состоянии готовности и имеющий наивысший приоритет.

Для определения приоритета используют следующие факторы:

1) время создания процесса

2) время появления работы, вызвавшей образование процесса

3) заказное время обслуживания

4) использованное время обслуживания

5) время, в течении которого процесс не обслуживался

6) объем и вид других ресурсов, необходимых процессу.

Общие стратегии диспетчеризации.

#### **диспетчеризация с одной очередью:**

предоставление всем пользователям одинаковых услуг. Процессы обслуживаются в порядке их появления, позволяя каждому процессу обслуживаться до конца (FIFO). Если позволить процессу переходить в заблокированное состояние, то стратегия будет называться FCFS (первый пришел, первый обслужен).

1. процесс обслуживается в порядке появления и каждому процессу позволено выполняться до конца (первый пришел, первый ушел - FIFO). Здесь процессы не блокируются.

2. если допускается блокирование процессов, то используется стратегия FCFS (первый пришел, первый обслужен). Очередь готовых к выполнению процессов

необходимо поддерживать в порядке возрастания времени процесса. Перераспределение процессора не производится.

НЕДОСТАТОК: короткие процессы ждут столько же, сколько и длинные.

3. стратегия на основе минимизации общего среднего времени ожидания, приоритет основан на времени выполнения (стратегия – следующий с кратчайшим ожиданием заданием SJN). Стратегия НЕ перераспределяет процессор, она изменяет общее среднее время ожидания: для коротких процессов оно становится меньше, увеличивая время ожидания длинных процессов.

4. следующая с минимальным остановившимся временем SRT. Перераспределение процессора осуществляется с использованием интервалов времени, которые выделялись процессору. Оставшееся время – разность между запрошенным и полученным временем. В этой стратегии достигается минимально возможное общее время ожидания. НЕДОСТАТКИ 1-4: задержка выполнения длинных процессов (некоторые процессы могут теряться навсегда). Требования: гарантия завершения всех процессов.

5. наиболее популярна циклическая стратегия перераспределения процессора – RR. Каждому процессу по очереди выделяется **фиксированный квант времени**, в конце которого, если процесс не закончился или не заблокировался, он снимается с процессора, считается только что поступившим в систему, и ставится в конец очереди.

6. стратегия гарантирует выделение процессора длинным процессам путем увеличения приоритета с течением времени. Эта стратегия учитывает старение.

(+) – низкая стоимость реализации

(-) – короткие процессы ждут столько же, сколько и длинные, - увеличивается среднее время ожидания.

#### Диспетчеризация с несколькими очередями

Используется информация пользователя и информация, получаемая в процессе выполнения. Существуют процессы интерактивные, пакетные, реального времени.

Для каждого вида процессов создается своя очередь. Существует связь очереди и процесса, она м.б. постоянной или меняться во времени.

Процесс реального времени должен быть обслужен к конкретному моменту времени.

Интерактивный д.б обслужен за приемлемый интервал времени.

Пакетный процесс не имеет жестких ограничений во времени.

Для каждой очереди используется своя стратегия, с различными управляющими параметрами, связь между процессом и очередью м.б. постоянной, а может изменяться со временем (статическая, динамическая).

#### 2. Блок состояния процесса.

Несмотря на то, что процесс является независимым объектом, со своим счетчиком команд и внутренним состоянием, существует необходимость взаимодействия с другими процессами. **Например**, выходные данные одного процесса могут служить входными данными для другого процесса.

В зависимости от относительных скоростей процессов (скорости зависят от относительной сложности программ и процессорного времени, предоставляемого каждому процессу), может получиться, что процесс уже готов к запуску, но входных данных для него еще нет. В этом случае процесс блокируется до поступления входных данных.

Процесс блокируется, поскольку с точки зрения логики он не может продолжать свою работу (обычно это связано с отсутствием входных данных, ожидаемых процессом). Также возможна ситуация, когда процесс, готовый и способный работать,

останавливается, поскольку операционная система решила предоставить на время процессор другому процессу.

Представлена диаграмма состояний, показывающая три возможных состояния процесса:

1. **Работающий** (в этот конкретный момент использующий процессор).

2. **Готовый к работе** (процесс временно приостановлен, чтобы позволить выполняться другому процессу).

3. **Заблокированный** (процесс не может быть запущен прежде, чем произойдет некое внешнее событие).

С точки зрения логики первые два состояния одинаковы. В обоих случаях процесс может быть запущен, только во втором случае недоступен процессор. Третье состояние отличается тем, что запустить процесс невозможно, независимо от загрузки процессора.

### 3. Алгоритмы диспетчеризации

*Алгоритм диспетчеризации*, или дисциплина обслуживания заявок, также оказывает большое влияние на длительность пребывания заявок в памяти ЦВМ. Ниже предполагается, что процессы поступления и обслуживания заявок являются независимыми, и поступившая заявка немедленно начинает обслуживаться, если в этот момент ЦВМ свободна от выполнения других подпрограмм, вызванных поступившими ранее заявками. Если же в момент поступления очередной заявки ЦВМ занята обслуживанием некоторой поступившей ранее заявки, то в зависимости от типа поступившей и обслуживаемой заявки, а также от вида используемого алгоритма диспетчеризации, поступившая заявка может прервать выполняемую подпрограмму или же ожидать начала своего обслуживания в памяти ЦВМ. Выбранная заявка покидает очередь, но все еще продолжает оставаться в памяти ЦВМ вплоть до момента завершения процесса своего обслуживания.

#### 1.8. Лекция №8 (2 часа).

**Тема:** «Синхронизация потоков»

##### 1.8.1. Вопросы лекции:

1. Синхронизация потоков.
2. Методы синхронизации: взаимное исключение, блокирующие переменные.
3. Взаимоблокировка процессов. Моделирование взаимоблокировок.
4. Методы борьбы с взаимоблокировками

#### 1.7.2 Краткое содержание вопросов:

##### 1. Синхронизация потоков.

Синхронизация потоков.

При выполнении нескольких потоков они будут неизменно взаимодействовать друг с другом, чтобы синхронизировать свое выполнение. Существует несколько методов синхронизации потоков:

- Блоки взаимного исключения (Mutex);
- Переменные состояния;
- Семафоры.

Объекты синхронизации являются переменными в памяти, к которым можно обратиться так же, как к данным. Потоки в различных процессах могут связаться друг с

другом через объекты синхронизации, помещенные в разделяемую память потоков, даже в случае, когда потоки в различных процессах вообще невидимы друг для друга.

Объекты синхронизации можно разместить в файлах, где они будут существовать независимо от создавшего их процесса.

## **2. Методы синхронизации: взаимное исключение, блокирующие переменные.**

Блокирующие переменные. Для синхронизации потоков одного процесса прикладной программист может использовать глобальные блокирующие переменные. С этими переменными, к которым все потоки процесса имеют прямой доступ, программист работает, не обращаясь к системным вызовам ОС. Каждому набору критических данных ставится в соответствие двоичная переменная, которой поток присваивает значение 0, когда он входит в критическую секцию, и значение 1, когда он ее покидает.

Блокирующие переменные могут использоваться не только при доступе к разделяемым данным, но и при доступе к разделяемым ресурсам любого вида.

## **3. Взаимоблокировка процессов. Моделирование взаимоблокировок.**

Взаимоблокировка процессов может происходить, когда несколько процессов борются за один ресурс.

Ресурсы бывают выгружаемые и невыгружаемые, аппаратные и программные.

Выгружаемый ресурс - этот ресурс безболезненно можно забрать у процесса (например: память).

Невыгружаемый ресурс - этот ресурс нельзя забрать у процесса без потери данных (например: принтер).

Проблема взаимоблокировок процессов возникает при борьбе за невыгружаемый ресурс.

Условия необходимые для взаимоблокировки:

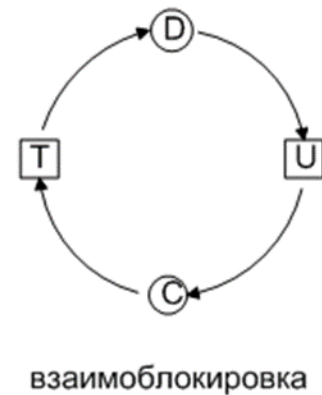
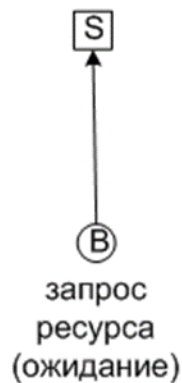
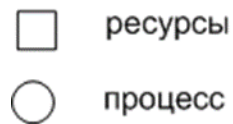
1) Условие взаимного исключения - в какой-то момент времени, ресурс занят только одним процессом или свободен.

2) Условие удержания и ожидания - процесс, удерживающий ресурс может запрашивать новые ресурсы.

3) Условие отсутствия принудительной выгрузки ресурса.

4) Условие циклического ожидания - должна существовать круговая последовательность из процессов, каждый, из которого ждет доступа к ресурсу, удерживаемому следующим членом последовательности.

## **4. Методы борьбы с взаимоблокировками**



Условные обозначения

На такой модели очень хорошо проверить возникает ли взаимоблокировка. Если есть цикл, значит, есть и взаимоблокировка

### 1.9. Лекция №9 (2 часа).

Тема: «Организация памяти»

#### 1.9.1. Вопросы лекции:

1. Управление памятью.
2. Методы без использования внешней памяти.
3. Методы с использованием внешней памяти (свопинг и виртуальная память).

#### 1.9.2 Краткое содержание вопросов:

##### 1. Управление памятью.

Под памятью (*memory*) в данном случае подразумевается оперативная (основная) *память* компьютера. В однопрограммных операционных системах основная *память* разделяется на две части. Одна часть для операционной системы (резидентный *монитор*, *ядро*), а вторая – для выполняющейся в текущий момент времени программы. В многопрограммных ОС "пользовательская" часть памяти – важнейший *ресурс* вычислительной системы – должна быть распределена для размещения нескольких процессов, в том числе процессов ОС. Эта задача распределения выполняется операционной системой динамической специальной подсистемой управления памятью (*memory management*). Эффективное *управление памятью* жизненно важно для многозадачных систем. Если в памяти будет находиться небольшое число процессов, то значительную часть времени процессы будут находиться в состоянии ожидания ввода-вывода и *загрузка* процессора будет низкой.

В ранних ОС *управление памятью* сводилось просто к загрузке программы и ее данных из некоторого внешнего накопителя (перфоленты, магнитной ленты или магнитного диска) в ОЗУ. При этом *память* разделялась между программой и ОС. На [рис. 6.3](#) показаны три варианта такой схемы. Первая модель раньше применялась на мэйнфреймах и мини-компьютерах. Вторая схема сейчас используется на некоторых карманных компьютерах и встроенных системах, третья модель была характерна для ранних персональных компьютеров с *MS-DOS*.

С появлением мультипрограммирования задачи ОС, связанные с распределением имеющейся памяти между несколькими одновременно выполняющимися программами, существенно усложнились.

Функциями ОС по управлению памятью в мультипрограммных системах являются:

- отслеживание (учет) свободной и занятой памяти;
- первоначальное и *динамическое выделение памяти* процессам приложений и самой операционной системе и освобождение памяти по завершении процессов;
- настройка адресов программы на конкретную область физической памяти;
- полное или частичное вытеснение кодов и данных процессов из ОП на диск, когда размеры ОП недостаточны для размещения всех процессов, и возвращение их в ОП;
- защита памяти, выделенной процессу, от возможных вмешательств со стороны других процессов;
- *дефрагментация памяти*.

Перечисленные функции особого пояснения не требуют, остановимся только на задаче преобразования адресов программы при ее загрузке в ОП.

В каждом компьютере есть оперативная память, используемая для хранения выполняющихся программ. В очень простых операционных системах в конкретный момент времени в памяти может находиться только одна программа. Для запуска второй программы сначала нужно удалить из памяти первую и загрузить на ее место вторую.

Более изощренные системы позволяют одновременно находиться в памяти нескольким программам. Для того чтобы они не мешали друг другу (и операционной системе), необходим некий защитный механизм. Хотя этот механизм располагается в аппаратуре, он управляется операционной системой.

Все алгоритмы распределения памяти можно разделить на два класса алгоритмы, в которых используется перемещение сегментов процессов между оперативной памятью и диском, и алгоритмы, в которых внешняя память не привлекается. *Одиночное непрерывное распределение памяти*

## **2. Методы без использования внешней памяти.**

Все алгоритмы распределения памяти можно разделить на два класса алгоритмы, в которых используется перемещение сегментов процессов между оперативной памятью и диском, и алгоритмы, в которых внешняя память не привлекается. *Одиночное непрерывное распределение памяти*

### **Некоторые понятия связанные с виртуальными адресами**

Максимально возможное виртуальное адресное пространство (ВАП) – определяется разрядностью процессора. Для 32 – разрядного Intel Pentium эта величина составляет 4Гб.

Назначенное ВАП – действительно необходимое процессу для работы. Его также называют образом процесса. Назначенное ВАП может превышать физический объем памяти. На этом основан механизм виртуальной памяти.

ВАП и виртуальная память – это разные механизмы для ОС. ОС может поддерживать ВАП, но механизм виртуальной памяти может при этом отсутствовать. Например, в случае превышения физической памяти над ВАП любого процесса. **Системная и пользовательская части ВАП (по 2 Гб каждая)**

При использовании данного метода память в начальный момент времени считается свободной (за исключением памяти отведенной под ОС). Каждому процессу

отводится вся необходимая память. Если ее не хватает, то процесс не создается. В произвольный момент времени память представляет собой случайную последовательность занятых и свободных участков.

Функции ОС:

Ведение таблиц свободных и занятых участков

Поиск участка

Загрузка и корректировка таблиц

После завершения процесса корректировка таблиц

Достоинство: большая гибкость, по сравнению с методом распределения фиксированными разделами. Недостаток – высокий уровень фрагментации.

#### **Распределение памяти перемещаемыми разделами**

В этом методе разработчики попытались учесть достоинства и недостатки предыдущего.

Один из способов борьбы с фрагментацией – сжатие, таким образом, чтобы вся свободная память образовала непрерывную область – дефрагментация. Такой метод был применен в ранних версиях OS/2. Недостаток этого метода – низкая производительность.

### **3. Методы с использованием внешней памяти (свопинг и виртуальная память).**

Виртуальным называется ресурс, который пользователю или пользовательской программе представляется обладающим свойствами, которыми он в действительности не обладает.

Виртуализация памяти может быть осуществлена на основе двух различных подходов: свопинг и виртуальная память.

#### **1.10. Лекция № 10 ( 2 часа).**

**Тема:** «Алгоритмы организации памяти»

##### **1.10.1. Вопросы лекции:**

1. Алгоритмы замещения страниц.
2. Распределение памяти.
3. Особенности реализации в UNIX и Windows
4. Сегментация памяти.
5. Реализация сегментации.

##### **1.10.2 Краткое содержание вопросов:**

#### **1. Алгоритмы замещения страниц.**

Обычно рассматривают три стратегии:

Стратегия выборки (fetchpolicy) - в какой момент следует переписать страницу из вторичной памяти в первичную. Выборка бывает по запросу и с упреждением. Алгоритм выборки вступает в действие в тот момент, когда процесс обращается к не

присутствующей странице, содержимое которой в данный момент находится на диске (в swap файле или отображенном файле), и потому является ключевым алгоритмом свопинга.

Стратегия размещения (placement policy) - определить в какое место первичной памяти поместить поступающую страницу. В системах со страничной организацией в любой свободный страничный кадр (в системах с сегментной организацией - нужна стратегия, аналогичная стратегии с переменными разделами).

Стратегия замещения (replacement policy) - какую страницу нужно вытолкнуть во внешнюю память, чтобы освободить место. Разумная стратегия замещения позволяет оптимизировать хранение в памяти самой крайне важной информации и тем самым снизить частоту страничных нарушений.

## **2. Распределение памяти.**

### *Распределение памяти разделами с фиксированными границами*

Простейший способ управления оперативной памятью состоит в том, что память разбивается на несколько областей фиксированной величины, называемых разделами. Такое разбиение может быть выполнено вручную оператором во время старта системы или во время ее установки. После этого границы разделов не изменяются.

### *Распределение памяти разделами с динамическими границами*

Сначала вся память, отводимая для приложений, свободна. Каждому вновь поступающему на выполнение приложению на этапе создания процесса выделяется вся необходимая ему память (если достаточный объем памяти отсутствует, то приложение не принимается на выполнение и процесс для него не создается). После завершения процесса память освобождается, и на это место может быть загружен другой процесс.

### *Распределение памяти подвижными разделами*

Одним из методов борьбы с фрагментацией является перемещение всех занятых участков в сторону старших или младших адресов, так, чтобы вся свободная память образовала единую свободную область. В дополнение к функциям, которые выполняет ОС при распределении памяти динамическими разделами, в данном случае она должна еще время от времени копировать содержимое разделов из одного места памяти в другое, корректируя таблицы свободных и занятых областей. Эта процедура называется сжатием.

### *Страничное распределение памяти*

Виртуальное адресное пространство каждого процесса делится на части одинакового, фиксированного для данной системы размера, называемые виртуальными страницами (virtual pages). В общем случае размер виртуального адресного пространства процесса не кратен размеру страницы, поэтому последняя страница каждого процесса дополняется фиктивной областью.

### *Сегментное распределение памяти*

Виртуальное адресное пространство процесса делится на части – сегменты, размер которых определяется с учетом смыслового значения содержащейся в них



информации. Отдельный сегмент может представлять собой подпрограмму, массив данных и т.п.

### *Сегментно-страничное распределение памяти*

Данный метод представляет собой комбинацию страничного и сегментного механизмов управления памятью и направлен на реализацию достоинств обоих подходов.

## **3. Особенности реализации в UNIX и Windows**

**В LUNIX системе** на 32-разрядной машине каждый процесс получает 3Гбайта виртуального пространства для себя, и 1Гбайт для страничных таблиц и других данных ядра.

## **4. Сегментация памяти.**

Недостатки системы, в которой используется одно адресное пространство:

Один участок может полностью заполниться, но при этом останутся свободные участки. Можно конечно перемещать участки, но это очень сложно.

Эти проблемы можно решить, если дать каждому участку независимое адресное пространство, называемое **сегментом**.

Сегмент - это логический объект.

В системах с сегментацией памяти каждое слово в адресном пространстве пользователя определяется виртуальным адресом, состоящим из двух частей: старшие разряды адреса рассматриваются как номер сегмента, а младшие - как номер слова внутри сегмента. Наряду с сегментацией может также использоваться страничная организация памяти. В этом случае виртуальный адрес слова состоит из трех частей: старшие разряды адреса определяют номер сегмента, средние - номер страницы внутри сегмента, а младшие - номер слова внутри страницы.

Как и в случае страничной организации, необходимо обеспечить преобразование виртуального адреса в реальный физический адрес основной памяти. С этой целью для каждого пользователя операционная система должна сформировать таблицу сегментов. Каждый элемент таблицы сегментов содержит дескриптор сегмента (поля базы, границы и индикаторов режима доступа). При отсутствии страничной организации поле базы определяет адрес начала сегмента в основной памяти, а граница - длину сегмента. При наличии страничной организации поле базы определяет адрес начала таблицы страниц данного сегмента, а граница - число страниц в сегменте. Поле индикаторов режима доступа представляет собой некоторую комбинацию признаков блокировки чтения, записи и выполнения.

Таблицы сегментов различных пользователей операционная система хранит в основной памяти. Для определения расположения таблицы сегментов выполняющейся программы используется специальный регистр защиты, который загружается операционной системой перед началом ее выполнения. Этот регистр содержит дескриптор таблицы сегментов (базу и границу), причем база содержит адрес начала таблицы сегментов выполняющейся программы, а граница - длину этой таблицы сегментов. Разряды номера сегмента виртуального адреса используются в качестве индекса для поиска в таблице сегментов. Таким образом, наличие базово-граничных пар в дескрипторе таблицы сегментов и элементах таблицы сегментов предотвращает возможность

обращения программы пользователя к таблицам сегментов и страниц, с которыми она не связана. Наличие в элементах таблицы сегментов индикаторов режима доступа позволяет осуществить необходимый режим доступа к сегменту со стороны данной программы. Для повышения эффективности схемы используется ассоциативная кэш-память.

Отметим, что в описанной схеме сегментации таблица сегментов с индикаторами доступа предоставляет всем программам, являющимся частями некоторой задачи, одинаковые возможности доступа, т. е. она определяет единственную область (домен) защиты. Однако для создания защищенных подсистем в рамках одной задачи для того, чтобы изменять возможности доступа, когда точка выполнения переходит через различные программы, управляющие ее решением, необходимо связать с каждой задачей множество доменов защиты. Реализация защищенных подсистем требует разработки некоторых специальных аппаратных средств. Рассмотрение таких систем, которые включают в себя кольцевые схемы защиты, а также различного рода мандатные схемы защиты, выходит за рамки данного обзора.

## **5. Реализация сегментации.**

Реализация сегментации существенно отличается от страничной организации памяти: страницы имеют фиксированный размер, а сегменты — нет.

### **1.11. Лекция № 11-12 (4 часа).**

**Тема:** «Основные концепции организации ввода-вывода. Организация ввода-вывода»

#### **1.11.1. Вопросы лекции:**

1. Устройства ввода–вывода
2. Реализации доступа к управляющим регистрам и буферам.
3. Принципы программного обеспечения ввода–вывода. \
4. Программный ввод–вывод.
5. Управляемый прерываниями ввод–вывод.
6. Ввод–вывод с использованием DMA.
7. Аппаратная часть различных видов накопителей.
8. Форматирование
9. Структура и типы файлов
10. Примеры файловых систем
11. Режимы хранения информации.
12. Управление общими дисковыми ресурсами
13. Сжатие информации. Архивирование данных.

#### **1.11.2 Краткое содержание вопросов:**

##### **1. Устройства ввода–вывода**

Внешние устройства (ВУ) — это важнейшая составная часть любого вычислительного комплекса, они составляют 50—80 % всего ПК. От состава и характеристик ВУ во многом зависят возможность и эффективность применения ПК.

Так, по назначению можно выделить следующие виды ВУ:

- внешние запоминающие устройства (ВЗУ), или внешняя память ПК;

- диалоговые средства пользователя;
- устройства ввода информации;
- устройства вывода информации;
- средства связи и телекоммуникации.

К устройствам **ввода информации** относятся: клавиатура, манипуляторы, сканеры, графические планшеты.

К устройствам **вывода информации** относятся принтеры и плоттеры (графопостроители).

## 2. Реализации доступа к управляющим регистрам и буферам.

Существуют два способа реализации доступа к управляющим регистрам и буферам данных устройств ввода-вывода.

1. Каждому управляющему регистру назначается номер порта ввода-вывода, 8-или 16-разрядное целое число. Таким образом работали самые древние компьютеры. И при такой схеме адресные пространства ОЗУ и устройств ввода-вывода не пересекаются.
2. Отображение всех управляющих регистров периферийных устройств на адресное пространство памяти.

## 3. Принципы программного обеспечения ввода–вывода

Разные специалисты рассматривают аппаратуру ввода/вывода, аппаратное обеспечение ввода/вывода с различных точек зрения. Инженеры-электронщики видят в ней микросхемы, проводники, источники питания, двигатели и прочие физические компоненты. Программисты, в первую очередь, обращают внимание на интерфейс, предоставляемый программному обеспечению, — команды, принимаемые аппаратурой, выполняемые ею функции, ошибки, о которых аппаратура может сообщить. В то же время программирование многих устройств ввода/вывода часто оказывается тесно связанным с их внутренним функционированием.

Устройства ввода/вывода можно грубо разделить на две категории: блочное устройство и символьное устройство. Блочными называются устройства, хранящие информацию в виде адресуемых блоков фиксированного размера. Обычно размеры блоков варьируются от 512 байт до 32 768 байт. Важное свойство блочного устройства состоит в том, что каждый блок может быть прочитан независимо от остальных блоков. Наиболее распространенными блочными устройствами являются диски. Если приглядеться внимательнее, то окажется, что граница между блочно адресуемыми устройствами и устройствами, к отдельным составляющим которых нельзя адресоваться напрямую, не определена строго. Все согласны с тем, что диск является блочно адресуемым устройством, так как вне зависимости от текущего положения головки дисководов всегда можно переместить ее на определенный цилиндр и затем считать или записать отдельный блок с нужной дорожки. Рассмотрим теперь накопитель на магнитной ленте (магнитофон), применяемый для хранения резервных копий диска. На ленте хранится последовательность блоков. Если магнитофону дать команду прочитать некоторый блок, ему потребуется перемотать ленту и начать читать данные, пока процесс не дойдет до запрашиваемого блока. Эта операция подобна поиску блока на диске с той лишь разницей, что она занимает значительно больше времени. Кроме того, в зависимости от накопителя и формата хранящихся на нем данных не гарантирована запись отдельного

произвольного блока в середине ленты. Попытка использовать магнитные ленты в качестве блочных устройств произвольного доступа явилась бы в какой-то степени натяжкой: никто их не использует таким образом.

#### **4. Программный ввод–вывод.**

Алгоритм печати:

- Строка для печати собирается в пространстве пользователя.
- Обращаясь к системному вызову, процесс получает принтер.
- Обращаясь к системному вызову, процесс просит распечатать строку на принтере.
- Операционная система копирует строку в массив, расположенный в режиме ядра.
- ОС копирует первый символ в регистр данных принтера, который отображен на памяти.
- Символ печатается на бумаге.
- Указатель устанавливается на следующий символ.
- Процессор ждет, когда бит готовности принтера выставится в готовность.
- Все повторяется.

#### **5. Управляемый прерываниями ввод–вывод.**

Алгоритм печати:

- До пункта 8 тоже самое.
- Процессор не ждет готовности принтера, а вызывает планировщик и переключается на другую задачу. Печатающий процесс блокируется.
- Когда принтер будет готов, он посылает прерывание процессору.

Процессор переключается на печатающий процесс.

#### **6. Ввод–вывод с использованием DMA.**

Независимо от того, отображаются ли регистры или буферы ввода-вывода на память или нет, CPU необходимо как-то адресоваться к контроллерам устройств для обмена данными с ними. CPU может запрашивать данные от контроллера ввода-вывода по одному байту, но подобная организация обмена данными крайне неэффективна, так как расходует огромное количество времени CPU. Поэтому на практике часто применяется другая схема, называемая доступом к памяти (DMA - direct memory access). ОС может

воспользоваться DMA только при наличии аппаратного DMA-контроллера, который есть у большинства систем. Иногда DMA-контроллер интегрируется в другие контроллеры, например в дисковый контроллер, но такой дизайн требует оснащения DMA-контроллерами каждого периферийного устройства. Как правило, контроллер, устанавливаемый на материнской плате, обслуживает запросы по передаче данных нескольких различных устройств ввода-вывода, часто конкурентной основе. Где бы он ни располагался физически, DMA-контроллер может получать доступ к системной шине независимо от CPU, как показано на Рис. 4. Он содержит несколько регистров, доступных CPU для чтения и записи. К ним относятся регистр адреса памяти, счетчик байтов и один или более управляющих регистров. Управляющие регистры задают, какой порт ввода-вывода должен быть использован, направление переноса данных (чтение из устройства ввода-вывода или запись в него), единицу переноса (осуществлять перенос данных побайтно или пословно), а также число байтов, которые следует перенести за одну операцию.

Чтобы понять, как работает DMA, познакомимся сначала с тем, как происходит чтение с диска при отсутствии DMA. Сначала контроллер считывает с диска блок (один или несколько секторов) последовательно, бит за битом, пока весь блок не окажется во внутреннем буфере контроллера. Затем контроллер проверяет контрольную сумму, чтобы убедиться, что при чтении не произошло ошибки. После этого контроллер инициирует прерывание. Когда ОС начинает работу, она может прочитать блок диска побайтно или пословно, в цикле сохраняя считанное слово или байт в RAM.

#### **7. Недостаток предыдущего метода в том, что прерывание происходит при печати каждого символа.**

Алгоритм не отличается, но всю работу на себя берет контроллер DMA, а не ЦП.

Программа аналогичная, ее выполняет контроллер DMA.

#### **8. Аппаратная часть различных видов накопителей.**

Конфигурация – это набор комплектующих, определяющий мощность и возможности компьютера: материнская плата, процессор, жесткий диск, монитор и т. д. Правильный выбор конфигурации позволяет добиться требуемого быстродействия и поддерживать его на протяжении длительного периода времени.

#### **9. Форматирование**

Каким образом организовать хранение файлов в системе, имеющей несколько устройств внешней памяти?

Первое решение состоит в том, что на каждом из устройств размещается автономная файловая система, т. е. файлы, находящиеся на этом устройстве, описываются деревом каталогов, никак не связанным с деревьями каталогов на других устройствах.

Другим вариантом является такая организация хранения файлов, при которой пользователю предоставляется возможность объединять файловые системы, находящиеся на разных устройствах, в единую файловую систему, описываемую единым деревом каталогов. Такая операция называется **монтированием**.

#### **10. Структура и типы файлов**

Файловые системы поддерживают несколько функционально различных типов файлов, в число которых, как правило, входят обычные файлы, файлы-каталоги, специальные файлы, именованные конвейеры, отображаемые в память файлы и другие

## **11. Примеры файловых систем**

### Файловая система ISO 9660

Стандарт принят в 1988 г.

По стандарту диски могут быть разбиты на логические разделы, но мы будем рассматривать диски с одним разделом.

### Joliet расширения для Windows

Это расширение было создано, чтобы файловая система ОС Windows 95 была представлена на CD-ROM.

### Romeo расширения для Windows

Стандарт Romeo предоставляет другую возможность записи файлов с длинными именами на компакт-диск. Длина имени может составлять 128 символов, однако использование кодировки Unicode не предусмотрено.

### HFS расширения для Macintosh

Иерархическая файловая система компьютеров Macintosh, не совместима ни с какими другими файловыми системами и называется Hierarchical File System (HFS).

### Файловая система UDF (Universal Disk Format)

Изначально созданная для DVD, с версии 1.50 добавили поддержку CD-RW и CD-R.

Сейчас последняя версия 2.01. Эта файловая система позволяет отформатировать CD-диск и работать с ним, как с обычным диском, копируя, удаляя и создавая файлы.

Недостаток, еще меньший объем диска (из 700 Мбайт остается 550 Мбайт).

### Файловая система CP/M

CP/M (Control Program for Microcomputers) - операционная система, предшественник MS-DOS.

В ее файловой системе только один каталог, с фиксированными записями по 32 байта.

Имена файлов - 8+3 символов верхнего регистра.

### Файловая система MS-DOS (FAT-12,16,32)

В первых версиях был только один каталог (MS-DOS 1.0).

С версии MS-DOS 2.0 применили иерархическую структуру.

Каталоговые записи, фиксированны по 32 байта.

Имена файлов - 8+3 символов верхнего регистра.

## **12. Режимы хранения информации.**

Хранение является одной из основных операций, осуществляемых над информацией, с целью обеспечения её доступности в течение некоторого промежутка времени. Под хранением информации понимают её запись в запоминающее устройство (ЗУ) для последующего использования. Запоминающее устройство (память) – устройство, способное принимать данные и сохранять их для последующего считывания. В компьютерных системах обработки информации выделяют следующие основные Типы памяти: регистровая память; основная память; кэш-память; внешняя память.

## **13. Управление общими дисковыми ресурсами**

В системах Windows для управления общими папками традиционно используется программа Windows Explorer и окно свойств папки. В системах Windows 2000, Windows XP и Windows Server 2003 для этих целей можно также применять оснастку Shared Folders. Кроме того, для просмотра и подключения общих папок можно использовать стандартные команды netshare и netuse, выполняющиеся в окне консоли. Могут быть применены и другие инструменты, использующие Win32 API, а также сценарии.

#### **14. Сжатие информации. Архивирование данных**

Характерной особенностью большинства типов данных является их избыточность. Степень избыточности данных зависит от типа данных. Другим фактором, влияющим на степень избыточности является принятая система кодирования.

В зависимости от того, в каком объекте размещены данные, подлежащие сжатию различают: сжатие (архивация) файлов, сжатие (архивация) папок, сжатие (уплотнение) дисков.

#### **1.13. Лекция №13 (2 часа).**

**Тема:** «Установка и настройка операционной системы»

##### **1.13.1 Вопросы лекции:**

1. Планирование и установка операционной системы.
2. Поддержка аппаратных средств.
3. Файловые системы, диски и тома.
4. Управление общими дисковыми ресурсами.
5. Средства управления оснасткой.
6. Консоль управления Microsoft (MMC).
7. Конфигурирование консолей MMC.
8. Управление учетными записями и рабочей средой пользователя. Аудит локальной системы.
9. Выполнение заданий по расписанию
10. Управление безопасностью.
11. Защита системы и данных.
12. Шифрующая файловая система EFS.
13. Средства мониторинга системы.
14. Просмотр системных событий.
15. Мониторинг производительности.

##### **1.13.2 Краткое содержание вопросов:**

#### **1. Планирование и установка операционной системы.**

Основная задача, которую необходимо решить перед тем, как приступить к установке Windows, заключается в том, чтобы определить, будет ли оборудование компьютера, на который вы собираетесь произвести установку, работать с данной операционной системой.

#### **2. Поддержка аппаратных средств.**

Четкой границы между программной и аппаратной реализацией функций ОС не существует — решение о том, какие функции ОС будут выполняться программно, а какие аппаратно, принимается разработчиками аппаратного и программного обеспечения компьютера. Тем не менее практически все современные аппаратные платформы имеют некоторый типичный набор средств аппаратной поддержки ОС, в который входят следующие компоненты:

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

Средства трансляции адресов выполняют операции преобразования виртуальных адресов, которые содержатся в кодах процесса, в адреса физической памяти.

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

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).

### **3. Файловые системы, диски и тома**

Файловая система (ФС) – это часть операционной системы, включающая:

- совокупность всех файлов на диске;
- наборы структур данных, используемых для управления файлами, такие, например, как каталоги файлов, дескрипторы файлов, таблицы распределения свободного и занятого пространства на диске;
- комплекс системных программных средств, реализующих различные операции над файлами, такие, как создание, уничтожение, чтение, запись, именование и поиск файлов.

Задачи, решаемые ФС, зависят от способа организации вычислительного процесса в целом. Самый простой тип – это ФС в однопользовательских и однопрограммных ОС, к числу которых относится, например, MS-DOS. Основные функции в такой ФС нацелены на решение следующих задач:

- именование файлов;
- программный интерфейс для приложений;
- отображение логической модели файловой системы на физическую организацию хранилища данных;
- устойчивость файловой системы к сбоям питания, ошибкам аппаратных и программных средств.

### **4. Управление общими дисковыми ресурсами**

Локальное и удаленное администрирование общих ресурсов в Windows 2000 осуществляется с помощью оснастки Общие папки (SharedFolders). (В Windows 4.0 аналогичные функции выполняла утилита Server панели управления.) С ее помощью можно также управлять сеансами и открытыми файлами. Она входит в стандартный инструмент администрирования – Управление компьютером (ComputerManagement).

### **5. Средства управления оснасткой.**



У администраторов системы и ее предшественников был целый набор административных инструментов, часть из которых была вынесена в панель управления, а другая часть находилась в меню Администрирование. Каждый инструмент представлял собой отдельное приложение пусть и с удобным, но своим интерфейсом. К тому же, приложение могло выполнять сразу несколько функций. По этим причинам начинающим администраторам (а данными утилитами пользуются, в основном, администраторы) было трудно управлять системой, часто они просто терялись в многообразии функций отдельного приложения, в то же время так и не находя нужного им параметра.

## **6. Консоль управления Microsoft (MMC)**

MMC является общей расширяемой средой для управляющих приложений, удовлетворяющей вышеперечисленным условиям. Реализована она в виде обычного MDI-приложения (многооконного), широко использующего интернет-технологии. Сама по себе MMC не представляет управляющих функций — это лишь среда для оснасток (snap-in). Оснастка — управляющий компонент, интегрирующийся в MMC.

Диспетчер оснасток (Snap-inmanager) — основа консоли управления — позволяет добавлять, удалять и модифицировать оснастки, а также разрешает указать, как будет работать данная оснастка: автономно или с дополнительными расширениями.

## **7. Конфигурирование консолей MMC.**

В WindowsServer 2003 управлять компьютером можно с помощью стандартных оснасток, поставляемых с системой.

Ряд оснасток с заданными функциональными возможностями могут быть объединены администратором в инструмент (также называемый документом), который сохраняется в файле с расширением .msc (ManagementSavedConsole). Файл \*.msc можно затем передать другому администратору (например, по электронной почте), который сможет использовать содержащийся в нем инструмент на своем рабочем месте.

## **8. Управление учетными записями и рабочей средой пользователя. Аудит локальной системы.**

Рабочая среда пользователя состоит из настроек рабочего стола, например, цвета экрана, настроек мыши, размера и расположения окон, из настроек процесса обмена информацией по сети и с устройством печати, переменных среды, параметров реестра и набора доступных приложений. Для управления средой пользователя предназначены следующие средства систем Windows 2000/XP и Windows Server 2003.

- Профили пользователей. В профиле пользователя хранятся все настройки рабочей среды системы, определенные самим пользователем. Это могут быть, например, настройки экрана и соединения с сетью. Все настройки, выполняемые самим пользователем, автоматически сохраняются в папке, имя которой для вновь установленной системы выглядит следующим образом: %SystemDrive%\Documents and settings\<имя\_пользователя>.

- Сценарий входа в систему (сценарий регистрации) представляет собой командный файл, имеющий расширение bat или cmd, исполняемый файл с расширением exe или сценарий VBScript, который запускается при каждой регистрации пользователя в системе или выходе из нее. Сценарий может содержать команды операционной системы,

предназначенные, например, для создания соединения с сетью или для запуска приложения. Кроме того, с помощью сценария можно устанавливать значения переменных среды, указывающих пути поиска, каталоги для временных файлов и другую подобную информацию.

- Сервер сценариев Windows (Windows Scripting Host, WSH). Сервер сценариев независим от языка и предназначен для работы на 32-разрядных платформах Windows. Он включает в себя как ядро сценариев Visual Basic Scripting Edition (VBScript), так и JScript. Сервер сценариев Windows предназначен для выполнения сценариев прямо на рабочем столе Windows или в окне консоли команд. При этом сценарии не нужно встраивать в документ HTML.

## **9. Выполнение заданий по расписанию**

С помощью планировщика заданий можно составить расписание запуска командных файлов, документов, обычных приложений или различных утилит для обслуживания системы. Программы могут запускаться однократно, ежедневно, еженедельно или ежемесячно в заданные дни, при загрузке системы или регистрации в ней, а также при бездействии системы (idle state). Планировщик позволяет задавать достаточно сложное расписание для выполнения заданий, в котором задаются продолжительность задания, время его окончания, количество повторов, зависимость от состояния источника питания (работа от сети или от батарей) и т. п.

Задание сохраняется как файл с расширением job, что позволяет перемещать его с одного компьютера на другой. Администраторы могут создавать файлы заданий для обслуживания систем и переносить их в нужное место. К папке заданий можно обращаться удаленно, кроме того, задания можно пересылать по электронной почте.

Служба планировщика заданий (Task Scheduler Service, MStask.exe) устанавливается вместе с системой и автоматически запускается при ее загрузке. При помощи меню **Дополнительно** (Advanced) планировщика заданий можно приостанавливать или запускать снова эту службу. Данное меню позволяет также обращаться к журналу регистрации запланированных и выполненных заданий.

## **10. Управление безопасностью.**

Управление безопасностью система сбора и обработки входной информации об угрозах для безопасности с упреждением во времени (прогнозирование угрозы), зависящим от потребного времени на реализацию контрмер; У.б. включает: систему оценки риска эксплуатации объекта на момент  $t$  при прогнозируемых угрозах; критерий принятия решения (решающее правило) для лица, принимающего решения; управляющие воздействия, влияющие на безопасность.

## **11. Защита системы и данных.**

Технические меры защиты информации предполагают использование программно - аппаратных средств защиты информации. При обработке ПДн с использованием средств автоматизации применение технических мер защиты является обязательным условием, а их количество и степень защиты определяется в процессе предпроектного обследования информационных ресурсов предприятия.

## **12. Шифрующая файловая система EFS.**

Шифрующая файловая система это тесно интегрированная с NTFS служба, располагающаяся в ядре Windows 2000. Ее назначение: защита данных, хранящихся на диске, от несанкционированного доступа путем их шифрования.

### **13. Средства мониторинга системы.**

Для мониторинга и оптимизации работы компьютера в системах Windows Server 2003 имеется несколько инструментов, позволяющих администратору следить за работой любых компонентов системы и конфигурировать ее оптимальным образом. Эти инструменты перечислены ниже.

1. Диспетчер задач (Task Manager) служит для просмотра текущих данных о производительности системы. В этой утилите основными являются три индикатора: использование процессора, использование виртуальной памяти и запущенные процессы и программы.

2. Оснастка Просмотр событий (Event Viewer) позволяет просматривать журналы событий, генерируемых приложениями, службой безопасности и системой.

3. Производительность (Performance) — обновленная оснастка систем Windows XP и Windows Server 2003, аналог утилиты Performance Monitor в Windows NT 4.0.

### **14. Просмотр системных событий.**

Событиями являются сообщения межсетевого экрана более высокого уровня, т.е. не связанные напрямую с пакетами (как в случае статистики). К событиям относятся сообщения, которые генерируются одним из трех главных модулей (пакетным фильтром, транслятором сетевых адресов или модулем шифрования/аутентификации) или каким-либо другим компонентом межсетевого экрана, таким, например, как ргоху сервер или каким-нибудь процессом (при выполнении специфических задач).

### **15. Мониторинг производительности.**

Два основных инструмента мониторинга производительности систем Windows Server — программу *"Диспетчер задач"*, которая предназначена для мониторинга работы приложений и служб сервера в реальном времени, и консоль *"Производительность"*, которая может осуществлять мониторинг производительности как в реальном времени, так и путем накопления статистики о работе системы за определенный период времени, причем консоль *"Производительность"* может показывать и собирать данные одновременно с нескольких систем.

#### **1.14 Лекция №14 (2 часа).**

**Тема:** «Использование системы»

##### **1.14.1 Вопросы лекции:**

1. Сетевые протоколы.
2. Типы сетевых подключений.
3. Подключение компьютера к Интернету. Совместное использование интернет-подключения.
4. Службы удаленного доступа
5. Восстановление системы и данных. Верификация цифровой подписи. Возможности отката драйвера.

6. Процедуры резервного копирования и восстановления. Консоль восстановления. Устранение неисправностей. Диагностика проблем, возникающих на этапе загрузки системы.
7. Системные сообщения. Назначение и структура системного реестра.

### **1.14.2 Краткое содержание вопросов:**

#### **1. Сетевые протоколы.**

Протокол – набор правил и процедур, регулирующих порядок осуществления некоторой связи (например, дипломатический протокол).

Сетевой протокол – правила и технические процедуры, позволяющие компьютерам, объединенным в сеть, осуществлять соединение и обмен данными. Три основных момента, касающиеся протоколов:

Существует множество протоколов. И хотя все они участвуют в реализации связи, каждый протокол имеет различные цели, выполняет различные задачи.

Протоколы работают на разных уровнях модели OSI (см. ниже). Функции протокола определяются уровнем, на котором он работает.

Несколько протоколов могут работать совместно. В этом случае они образуют так называемый стек протоколов или набор протоколов.

#### **2. Типы сетевых подключений.**

Типы сетевых подключений. Существует 5 типов сетевых подключений. Тип подключения

1. Телефонное подключение  
(Dial-upconnection)
2. ЛВС  
(LAN, Local Area connection)
3. Виртуальная частная сеть  
(VPNconnection,  
Virtual private network)
4. Прямоеподключение  
(Direct Connection)
5. Входящееподключение  
(Incoming connection)

#### **3. Подключение компьютера к Интернету. Совместное использование интернет-подключения.**

Возможность совместного использования интернет-подключения (InternetConnectionSharing, ICS) позволяет настроить Windows XP для подключения домашней или малой офисной сети к Интернету.

Услуги, предоставляемые домашней сетью, можно настроить так, чтобы к ним могли получить доступ пользователи Интернета.

#### **4. Службы удаленного доступа**

Приложения, позволяющие осуществить заход удаленным терминалом, очень популярны в Internet. TelNet – одна из самых старых информационных технологий Интернета, относящаяся к сервисам прямого обращения. Под TelNet понимают триаду, состоящую из TelNet-интерфейса пользователя, TelNet-процесса и TelNet-протокола.

Эта триада обеспечивает описание и реализацию сетевого терминала для доступа к ресурсам удаленного компьютера.

TelNet строится как протокол приложения над транспортным протоколом TCP. В основу TelNet положены три фундаментальные идеи:

- Концепция сетевого виртуального терминала NVT (NetworkVirtualTerminal).
- Принцип договорных опций (согласование параметров взаимодействия).
- Симметрия связи «терминал-процесс».

## **5. Восстановление системы и данных. Верификация цифровой подписи. Возможности отката драйвера.**

Несмотря на то, что WindowsServer 2003 по надежности и отказоустойчивости превосходит все предшествующие версии Windows, давайте посмотрим на вещи реально: поскольку проблемы с загрузкой возможны при работе с любой операционной системой, то и WindowsServer 2003 не является исключением из общего правила. Несмотря на новое название, эта операционная система является логическим продолжением семейства операционных систем Windows NT/2000/XP, и по этой причине в процессе загрузки WindowsServer 2003 могут возникать некоторые проблемы, с которыми сталкивались пользователи этих операционных систем. В частности, возникновение проблем может быть вызвано:

- сбоями в работе жесткого диска или ошибками в работе контроллера жесткого диска;
- сбоями в подаче электропитания;
- некорректно работающими приложениями и плохо написанными драйверами устройств;
- вирусной атакой;
- ошибками пользователей (человеческий фактор никогда не следует сбрасывать со счетов).

Довольно часто такие проблемы, как аппаратные конфликты, нестабильное поведение системы, неправильная работа устройств и даже ошибки STOP бывают вызваны некорректным драйвером. В таких случаях желательно было бы быстро заменить проблемный драйвер предыдущей версией без переустановки системы. Функция отката драйвера оказывается незаменимой при устранении неполадок, при отладке бета-версий драйверов и в других ситуациях.

## **6. Процедуры резервного копирования и восстановления. Консоль восстановления. Устранение неисправностей. Диагностика проблем, возникающих на этапе загрузки системы.**

Процедуры резервного копирования и восстановления. Консоль восстановления. Устранение неисправностей. Диагностика проблем, возникающих на этапе загрузки системы.

## **7. Системные сообщения. Назначение и структура системного реестра.**

*Системный реестр* – база данных, которая сохраняет параметры настройки для 32 разрядных версий Microsoft Windows, включая Windows 95, 98 и NT. Он содержит информацию и параметры настройки для всех аппаратных средств, программ, пользователей, и свойств РС. Каждый раз, когда пользователь делает изменения в параметрах настройки Панели управления, или в ассоциациях файлов, системной

настройке, или в установленном программном обеспечении, изменения отражаются и сохраняются в системном реестре.

Вся база системного реестра разделена на шесть основных разделов, которые принято называть ветвями. Каждая ветвь содержит в себе параметры, относящиеся к определенному набору ключей. Ниже кратко описано назначение этих разделов.

- **HKEY\_CLASSES\_ROOT:** Эта ветвь содержит все типы Ваших ассоциаций к файлам, информацию об OLE и данные по ярлыкам.

- **HKEY\_CURRENT\_USER:** Эта ветвь связана с ветвью **HKEY\_USERS**, и соответствует пользователю, работающему в настоящее время на PC. Содержит настройки системы и программ, относящиеся к текущему пользователю. Он создается при регистрации пользователя в системе на основе информации из соответствующего ключа [**HKEY\_USERS**]. Именно здесь хранится информация о том, как данный пользователь сконфигурировал рабочую станцию.

- **HKEY\_LOCAL\_MACHINE:** Эта ветвь содержит определенную информацию о типах аппаратных средств, программного обеспечения, и других настройках на данном PC, эта информация используется для всех пользователей, которые работают на этом компьютере.

- **HKEY\_USERS:** Эта ветвь содержит индивидуальные настройки каждого пользователя компьютера, каждый пользователь представлен под ключом **SID**, расположенном под главной ветвью. Содержит информацию обо всех пользователях данной рабочей станции. Здесь хранятся данные о каждом пользователе, а также типовые настройки, служащие шаблоном для новых ключей, создаваемых пользователем. Типовые настройки включают различные значения по умолчанию для программ, событий, конфигураций рабочего стола и т.д.

- **HKEY\_CURRENT\_CONFIG:** Эта ветвь связана с ветвью **HKEY\_LOCAL\_MACHINE**, и соответствует текущей аппаратной конфигурации. Содержит информацию о текущей конфигурации аппаратуры компьютера, используется в основном на компьютерах с несколькими аппаратными конфигурациями, например, при подключении портативного ПК к стыковочной станции и отключении от нее. Информация, содержащаяся в этом ключе, копируется из ключа [**HKEY\_LOCAL\_MACHINE**].

- **HKEY\_DYN\_DATA:** Эта ветвь связана с частью **HKEY\_LOCAL\_MACHINE**, и служит для использования особенностей Plug-&Play в Windows, этот раздел динамически изменяется, когда устройства добавляются и удаляются из системы.

#### 6. Состав основных разделов

Каждый из вышеперечисленных разделов содержит в себе другие разделы — как и файловая система, Registry имеет структуру дерева. Каждый узел (раздел или подраздел) называется ключом. Вы можете открывать новые ветви до тех пор, пока не доберетесь до уровня, на котором находятся только параметры.

##### 1. Hkey\_Classes\_Root

Структура раздела несколько отличается от всех остальных. Для каждого зарегистрированного расширения файла имеется подключ (например, .bmp).

Значение этого ключа "По умолчанию" указывает на подключ описания документа ("ACDC\_BMP"), который расположен в той же ветви основного раздела. В подключе описания документа и содержится цепочка ключей, хранящих информацию об ассоциациях, OLE, DDE.

##### 2. Hkey\_Local\_Machine

Информация, сохраненная здесь, используется приложениями, устройствами и системой, и не зависит от того, кто был заявлен в качестве пользователя. Устройства могут помещать информацию в системный реестр с помощью Plug&Play-интерфейса, программные средства — посредством стандартного API. Hkey\_Local\_Machine содержит ряд подразделов, описанных в табл. 1.

### 3. Подраздел *Config*

- Содержит информацию о различных конфигурациях аппаратных средств.
- Каждая конфигурация имеет уникальное обозначение и хранится в отдельном подразделе с соответствующим именем.
- Конфигурации перечислены в списке в окне утилиты Система. Здесь же их можно обрабатывать.

• При запуске Windows проводится проверка конфигурации аппаратных средств. При этом может произойти следующее:

- В большинстве случаев конфигурационные данные позволяют Windows автоматически выбрать соответствующую конфигурацию.

• При первом после изменения оборудования запуске компьютера Windows создает новый элемент конфигурации для новых конфигурационных данных. В результате создается и новый Config-элемент в системном реестре.

• Когда конфигурационные данные не позволяют системе Windows однозначно решить, какую из описанных конфигураций следует выбрать, пользователю при загрузке системы предлагается меню, посредством которого он может выбрать подходящую конфигурацию.

### 4. Подраздел *Enum*

• Windows располагает специальными программами, которые отвечают за построение дерева аппаратуры в системном реестре (например, Диспетчер устройств, вызываемый через Панель управления - Система- Устройства).

- Каждому устройству присваивается уникальный идентификационный код.

• В системном реестре хранится идентификационная информация о каждом устройстве, например, тип устройства, идентификационный код (ID) устройства, информация об изготовителе и информация, о драйвере.

Информация о составе данного раздела приведена в табл.2.

### 5. Подраздел *Software*

• Содержит информацию о каждом программном средстве, установленном на компьютере.

• Содержимое этого раздела является общим для всех пользователей данного компьютера.

• Hkey\_Local\_Machine\Software содержит ряд подразделов и сведения о различных подразделах (их описание), которые могут появиться в системном реестре (см. табл.3).

### 6. Подраздел *System*

• Данные в подразделе System содержат все параметры драйверов устройств и служб, используемые при запуске Windows.

• Вся информация хранится в подразделе CurrentControlSet. Он содержит два следующих подраздела:

• *Control*: Подраздел включает информацию, используемую, при запуске системы, например, сетевое имя компьютера и запускаемые подсистемы.

• *Services*: Подраздел включает информацию, необходимую для контроля загрузки и конфигурирования драйверов, файловой системы, и др. Здесь также определяется, как отдельные службы вызывают одна другую.

Состав двух вышеназванных подразделов приведен в табл.4 и 5.

### 7. Hkey\_Current\_User и Hkey\_Users

• Содержит Default-подраздел и подразделы для всех пользователей, заявленных в системе.

• Информация из подраздела Default используется для того, чтобы создать конфигурацию для нового пользователя.

• Hkey\_Current\_User содержит информацию о пользователе, работающем на компьютере в текущем сеансе (см. табл.6).

Если существуют одинаковые параметры в Hkey\_Local\_Machine и Hkey\_Current\_User, то используются значения параметров, взятые из Hkey\_Current\_User.

#### 8. Hkey\_Current\_Config и Hkey\_Dyn\_Data

- Hkey\_Current\_Config указывает на текущую системную конфигурацию, которая сохранена в Hkey\_Local\_Machine\Config.

- Часть системной информации в Windows должна постоянно присутствовать в оперативной памяти, поскольку системе необходим быстрый доступ к этой информации и Windows не может ожидать, пока нужные данные будут прочитаны с жесткого диска. Вся эта информация находится в Hkey\_Dyn\_Data.

- Подраздел Hkey\_Dyn\_Data\ConfigurationManager, называемый также деревом аппаратуры, представляет собой хранящееся в оперативной памяти описание текущей системной конфигурации.

- Дерево аппаратуры создается заново при каждом запуске системы и адаптируется, если в состав или конфигурацию аппаратуры были внесены изменения. Присутствующие в этом разделе данные можно просмотреть с помощью Редактора реестра, они всегда соответствуют текущему состоянию аппаратуры компьютера.

- Hkey\_Dyn\_Data содержит статистическую информацию о различных сетевых компонентах в системе. Она находится в подразделе PerfStats.

### 1.15 Лекция №15-16 (4 часа).

**Тема:** «Основные понятия безопасности. Обеспечение безопасности»

#### 1.15.1 Вопросы лекции:

1. Основные понятия безопасности.
2. Классификация угроз
3. Системный подход к обеспечению безопасности.
4. Политика безопасности.

#### 1.15.2 Краткое содержание вопросов:

##### 1. Основные понятия безопасности.

Безопасная информационная система - это система, которая, во-первых, защищает данные от несанкционированного доступа, во-вторых, всегда готова предоставить их своим пользователям, а в-третьих, надежно хранит информацию и гарантирует неизменность данных. Таким образом, безопасная система по определению обладает свойствами конфиденциальности, доступности и целостности.

Конфиденциальность (confidentiality) - гарантия того, что секретные данные будут доступны только тем пользователям, которым этот доступ разрешен (такие пользователи называются авторизованными).

Доступность (availability) - гарантия того, что авторизованные пользователи всегда получают доступ к данным.

Целостность (integrity) - гарантия сохранности данными правильных значений, которая обеспечивается запретом для неавторизованных пользователей каким-либо образом изменять, модифицировать, разрушать или создавать данные.

##### 2. Классификация угроз

- 1) По характеру воздействия на ОС:
- 2) По цели:
- 3) По условию начала осуществления воздействия:
- 4) По наличию обратной связи с атакуемым объектом:
- 5) По уровню эталонной модели OSI:



- 6) По принципу взаимодействия с ОС:
- 7) По типу угроз используемой злоумышленником слабости защиты:
- 8) По способу воздействия на объект атаки:
- 9) По способу действия злоумышленника:
- 10) По объекту атаки:
- 11) По используемым средствам атаки:
- 12) По состоянию атакуемого объекта ОС на момент атаки:

### **3. Системный подход к обеспечению безопасности.**

Неумышленные угрозы вызываются ошибочными действиями лояльных сотрудников, становятся следствием их низкой квалификации или безответственности. Кроме того, к такому роду угроз относятся последствия ненадежной работы программных и аппаратных средств системы.

Умышленные угрозы могут ограничиваться либо пассивным чтением данных или мониторингом системы, либо включать в себя активные действия, например, нарушение целостности и доступности информации, приведение в нерабочее состояние приложений и устройств.

Программа - «троянский конь» всегда маскируется под какую-нибудь полезную утилиту или игру, а производит действия, разрушающие систему.

Административные меры - это действия, предпринимаемые руководством предприятия или организации для обеспечения информационной безопасности. К таким мерам относятся конкретные правила работы сотрудников предприятия, например режим работы сотрудников, их должностные инструкции, строго определяющие порядок работы с конфиденциальной информацией на компьютере.

### **4. Политика безопасности.**

Политика безопасности предполагает:

- Минимальный уровень привилегий для каждого сотрудника, необходимый для выполнения его должностных обязанностей
- Комплексный подход (начиная с организационно-административных и заканчивая встроенными средствами аппаратуры)
- Средства безопасности, в случае отказа, должны переходить в состояние максимальной защиты (блокировать доступ)
- Принцип единого контрольно-пропускного пункта между внешней и внутренней сетями
- Принцип баланса возможного ущерба от реализации угроз и затрат на ее предотвращение
- Политика доступа к внутренним ресурсам определяется двумя принципами:
  - запрещать все, что явно не разрешено (высокая степень безопасности, большие неудобства для пользователей, дорогой)
  - разрешать все, что явно не запрещено (меньшая защищенность, однако требует меньше затрат)

## 2. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ПО ВЫПОЛНЕНИЮ

### ЛАБОРАТОРНЫХ РАБОТ

#### 2.1 Лабораторная работа №1-2 (4 часа).

**Тема:** «Общие сведения об операционных системах»

##### 2.1.1 Цель работы:

Ознакомиться с MS-DOS

##### 2.1.2 Задачи работы:

1. Создать на диске D: каталог WORK. Убедиться в выполнении операции.
2. Создать на диске D: в корневом каталоге текстовые файлы rb1 и rb2.  
Записать в них следующие тексты: Записать в них следующие тексты:
  - а. rb1: Компьютер является однопользовательским, если за ним в каждый момент времени может работать только один человек.
  - б. rb2: Основным периферийным устройством, обеспечивающим ввод в компьютер информации, является клавиатура.Убедиться в выполнении операции.
3. Скопировать файл rb1 в каталог WORK. Убедиться в выполнении операции.
4. Сделать каталог WORK текущим и просмотреть текстовый файл rb1.
5. Создать в каталоге WORK два подкаталога NEW1 и NEW2.
6. Скопировать текстовый файл rb2 в каталог WORK и подкаталог NEW2.
7. Переименовать файл rb2 в подкаталоге NEW2 в rb3. Убедиться в выполнении операции.
8. Очистить экран.
9. Вывести содержимое файла rb3.
10. Показать созданные файлы и каталоги преподавателю.
11. Удалить созданные файлы и каталоги. Показать преподавателю.

##### 2.1.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

##### 2.1.4 Описание (ход) работы:

1. Сделать текущим корневой каталог диска D: `CD D:\`.  
Создать в корневом каталоге диска D: каталог WORK командой `MD WORK`.  
Убедиться в выполнении операций командой `DIR`.
2. Командой `COPY CON имя_файла` создать файл rb1 в корневом каталоге диска C:, ввести текст. После ввода текста нажать F6 и Enter для завершения набора.  
Законспектировать экран.  
Повторить то же для файла rb2.
3. Скопировать файл rb1 в каталог WORK командой `COPY rb1 \WORK`.  
Убедиться в том, что файл скопирован с помощью команды `DIR \WORK`.  
Законспектировать экран.
4. Командой `CD` изменить текущий каталог на `D:\WORK`. Просмотреть командой `TYPE имя_файла` содержимое файла rb1 в каталоге WORK.
5. Командой `MD NEW1` создать сначала подкаталог NEW1, а затем NEW2 в каталоге WORK. Командой `DIR` убедиться в создании каталогов, законспектировать экран.

6. Из корневого каталога диска D: командой COPY D:\rb2 скопировать файл rb2 в каталог WORK, а затем скопировать в подкаталог D:\WORK\NEW2 командой COPY D:\rb2 NEW2. Убедиться в выполнении первой операции копирования командой DIR и законспектировать экран, затем проверить выполнение копирования в каталог NEW2 командой DIR NEW2.

7. Командой RENAME NEW2\rb2 rb3 переименовать файл rb2 в подкаталоге NEW2 в файл rb3. Командой DIR NEW2 убедиться в выполнении операции и законспектировать экран.

8. Очистить экран командой CLS.

9. Командой TYPE вывести на экран содержимое файла rb3 подкаталога NEW2.

10. Командами CD и DIR продемонстрировать наличие созданных каталогов и файлов.

11. Удалить поочередно сначала файлы, а затем каталоги командой DEL. Продемонстрировать отсутствие файлов и каталогов командой DIR.

## **2.2 Лабораторная работа №3-4 (4 часа).**

**Тема:** «Управляющие подсистемы ОС»

### **2.2.1 Цель работы:**

Ознакомиться с командами MS-DOS

### **2.2.2 Задачи работы:**

Выполнить конфигурацию ОС, используя файлы AUTOEXEC.BAT и CONFIG.SYS, чтобы она отвечала следующим требованиям:

1. Установить режим прерывания программ.
2. Освободить часть оперативной памяти выше 640Кбайт.
3. Установить число одновременно открытых файлов 30.
4. Выделить 20 буферов для промежуточного хранения информации.
5. Подключить драйвер мыши.
6. Установить нужный формат даты и времени.

### **2.2.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

### **2.2.4 Описание (ход) работы:**

1. Задать приглашение в командной строке в стандартном виде.
2. Установить размещение временных файлов на диске C:\Tmp.
3. Включить режим контроля корректности записи данных на диск.
4. Задать путь поиска файлов в каталогах C:\WINDOWS, C:\NC.
5. Запустить NortonCommander.
6. Для конфигурирования системы нужно написать файлы Autoexec.bat и Config.sys.
7. Для создания файла Config.sys напишите команду: COPY CON CONFIG.SYS

Наберите в файле CONFIG.SYS следующие команды:

Break=on

Dos=HIGH, UMB

Files=30

Buffers=20

Devicehigh=C:\Windows\MOUSE.SYS

Country=07, 866, C:\DOS\Country.sys

Таким же образом создайте файл Autoexec.bat и наберите в нем следующие команды:

```
@ECHO OFF
PROMPT $P$G
SET temp=c:\tmp
VERIFY ON
PATH C:\WINDOWS;C:\NC
NC
```

Выполнить перезагрузку машины.

### **2.3 Лабораторная работа №5-6 (4 часа).**

**Тема:** «Микроядерная архитектура»

#### **2.3.1 Цель работы:**

Научиться настраивать конфигурацию DOS

#### **2.3.2 Задачи работы:**

1. Установить режим прерывания программ.
2. Освободить часть оперативной памяти выше 640Кбайт.
3. Установить число одновременно открытых файлов 30.
4. Выделить 20 буферов для промежуточного хранения информации.
5. Подключить драйвер мыши.
6. Установить нужный формат даты и времени.

#### **2.3.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### **2.3.4 Описание (ход) работы:**

1. Для конфигурирования системы нужно написать файлы Autoexec.bat и Config.sys.
2. Для создания файла Config.sys напишите команду: COPY CON CONFIG.SYS

Наберите в файле CONFIG.SYS следующие команды:

```
Break=on
Dos=HIGH, UMB
Files=30
Buffers=20
Devicehigh=C:\Windows\MOUSE.SYS
Country=07, 866, C:\DOS\Country.sys
```

Таким же образом создайте файл Autoexec.bat и наберите в нем следующие команды:

```
@ECHO OFF
PROMPT $P$G
SET temp=c:\tmp
VERIFY ON
```

PATH C:\WINDOWS;C:\NC  
NC

Выполнить перезагрузку машины.

## **2.4 Лабораторная работа №7-8 (4 часа).**

**Тема:** «Совместимость ОС»

### **2.4.1 Цель работы:**

Изучить структуру операционной системы Windows

### **2.4.2 Задачи работы:**

Изучить

1. Изучить теоретический материал;
2. На примере Windows 7 выполнить упражнения № 1-17;
3. Ответить на контрольные вопросы.

### **2.4.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

### **2.4.4 Описание (ход) работы:**

1. Изучить основные системные объекты и приемы управления ОС *Windows 7* и порядок работы с ними;
2. Изучить назначение и использование активных и пассивных элементов управления *Windows7*;
3. Научиться выполнять основные команды работы с файлами, папками и другими объектами *Windows7*.

## **2.5 Лабораторная работа №9-10 (4 часа).**

**Тема:** «Мультипрограммирование»

### **2.5.1 Цель работы:**

Ознакомиться с ОС Linux Ubuntu

### **2.5.2 Задачи работы:**

1. Обзор Linux
2. Запуск системы
3. Вход в систему и аутентификация пользователя
4. Структура рабочего стола
5. Работа с домашней директорией
6. Работа с папкой StartHere
7. Работа из командной строки. Утилита Terminal
8. Соединение в сеть с Windows-компьютером. Сервер Samba
9. Работа на удаленных компьютерах
10. Выход из системы

### **2.5.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### 2.5.4 Описание (ход) работы:

Linux является UNIX-подобной ОС, созданной по образцу влиятельной UNIX OS, разработанной в свою очередь дочерней компанией Bell Labs в начале 1970-х.

- Система названа в честь Линуса Торвальдса, финского студента, который в 1991 году написал первое ядро Linux. При поддержке широкого спектра сообществ и корпоративных разработчиков, Торвальдс и по сей день руководит разработкой Linux.

- Linux распространяется по лицензии GNU General Public License. Лицензия GPL требует, чтобы исходный код был доступен всем пользователям ПО, допускается изменение, распространение и продажа кода скомпилированных версий, разрешается частное использование модифицированного кода и скомпилированных версий без каких-либо ограничений. Согласно GPL, в случае если модифицированные и скомпилированные версии выставляются на продажу или передаются в общественное пользование, требуется открытая публикация видоизмененного кода.

- Linux сам по себе представляет собой только ядро - основной компонент ОС, который контролирует память и другие функции компьютерного оборудования. Для того, чтобы использовать ОС, ее нужно упаковать совместно с системными библиотеками GNU и утилитами. Ядро может также сочетаться с оконной системой X Window System, которая обеспечивает графическую среду, схожую с Windows или Mac OS.

- Готовые к использованию пакеты Linux называются **дистрибутивами**. Дистрибутивы развиваются, сохраняются и продаются коммерческим компаниям и НКО. Одними из самых популярных дистрибутивов являются: Red Hat, Debian, и Ubuntu. Хотя Linux сам по себе не предполагает никаких лицензионных сборов, коммерческие компании-разработчики Linux получают прибыль за счет продажи сервисов технической поддержки, главным образом, корпоративным клиентам.

В то время как Windows царит на рынке корпоративных компьютеров, а Apple по-прежнему завоевывает сердца писателей, художников и музыкантов, Linux стала излюбленной средой работы для программистов, специалистов в области IT и пользователей с высокой технической компетенцией. Некоторые из их приоритетов могут пересекаться с вашими. Приведем несколько примеров:

**Цена:** Linux можно получить бесплатно. Популярное сообщество Linux-пользователей гласит "Free as in freedom" или "Free as in beer." Linux соответствует обоим высказываниям. Это означает, что Вы можете загрузить дистрибутив Linux и изменить его так, как вы пожелаете. Сайты с дистрибутивами Linux позволяют загрузить их абсолютно бесплатно, записать на диск и установить на неограниченное количество компьютеров, а также сделать копии для друзей (Для корпоративных пользователей есть некоторые ограничения, в большинстве случаев потребуется контракт поддержки).

**Внешний вид:** Также как и Windows и OS X загружаются и работают несколько иначе (хотя они скорее, похожи, чем серьезно отличаются), различия присутствуют и у дистрибутивов Linux. Хотя каждый пользователь Linux может иметь свой видоизмененный пользовательский интерфейс, большинство дистрибутивов также позволяют свободно выбирать альтернативные дизайны рабочего стола.

Упрощенно, можно выделить три основных концепции интерфейсов Linux:

- **клоны Microsoft и Apple:** Macintosh произвела Windows, а Windows произвела GNOME и KDE, а также десятки других интерфейсов на их основе. По существу, все они напоминают концепцию пользовательского интерфейса Windows до выхода ОС Windows 8.

- **традиция нативного дизайна UNIX-систем:** до Linux данная группа интерфейсов использовалась администраторами UNIX-серверов и редко появлялась на публике. Наполненные виджетами, данные интерфейсы предлагали полезные идеи (как

например виртуальный рабочий стол), которые Microsoft и Apple посчитали запутанными для своих пользователей.

- **кроссплатформенные рабочие столы с поддержкой сенсорного ввода, оптимизированные для мобильных устройств:** данный интерфейс стал девизом Windows 8, но впервые он был использован именно Linux (Apple также была первопроходцем в данной области).

## **2.6 Лабораторная работа №11-12 (4 часа).**

**Тема:** «Обработка прерываний»

### **2.6.1 Цель работы:**

Ознакомиться с командами ОС Linux, структурой операционной системы Linux

### **2.6.2 Задачи работы:**

1. Изучить теоретический материал
2. Создать текстовый файл
3. Переименовать файл
4. Создать каталог и два подкаталога
5. Удалить текстовый файл
6. Создать 3 текстовых файла и объединить их.

### **2.6.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

### **2.6.4 Описание (ход) работы:**

Linux команды, которым не нашлось место ни в каком другом разделе.

# uname -a

# Показать версию ядра Linux;

# lsb\_release -a

# На экран выводится информация о версии операционной системы Debian и основанных на ней дистрибутивах;

# cat /etc/SuSE-release

# На экран выводится информация о версии операционной системы SuSE;

# man hier

# Описание иерархии файловой системы (для чего нужна каждая директория в linux);

# clear

# Очистка экрана терминала;

# wall Привет

# Отправляет на терминалы других пользователей сообщение "Привет";

# date

# Показывает текущую дату и время;

# cal -3

# Показывает в удобной форме предыдущий, текущий и последующий месяц (тип календарь);

# uptime

# Показать текущее время и работу системы без перезагрузки и выключения;

```

# hostname
# Показать сетевое имя компьютера;
# whois linux.org
# Показать информацию о домене имени linux.org;
# export http_proxy=http://your.proxy:port
# Изменить переменной окружения http_proxy, для использования интернета через
proxy-сервер;
# wget http://itshaman.ru/images/logo_white.png
# Скачать файл http://itshaman.ru/images/logo_white.png в текущую папку;
# wget --convert-links -r http://www.linux.org/
# Копирование сайта целиком и конвертирование ссылок для автономной работы.
Копирование происходит на 5 уровней в глубину;
# rppconfig
# Создание и настройка Dial-Up соединения для выхода в Интернет по модему;
# rpproconf
# Создание и настройка выхода в Интернет через ADSL-модем;
# !!
# Выполнить последнюю команду;
# history | tail -50
# Показать последние 50 набранных команд;
# exit
# Завершить сеанс текущего пользователя;
# passwd
# Меняет пароль текущего пользователя;
# shutdown -h now
# Выход из Linux;
# poweroff
# Выход из Linux;
# reboot
# Перезагрузка системы;
# last reboot
# Статистика перезагрузок;
# winecfg
# Настройка Wine - НЕэмулятор WinAPI;
# host itshaman.ru
# Показывает IP-адрес введенного сайта;

```

## **2.7 Лабораторная работа №13-14 (4 часа).**

**Тема:** «Диспетчеризация процессов»

### **2.7.1 Цель работы:**

Изучить управление процессами Linux

### **2.7.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Выполнить задания
3. Ответить на вопросы

### **2.7.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player



#### **2.7.4 Описание (ход) работы:**

Процесс в Linux (как и в UNIX) - это программа, которая выполняется в отдельном виртуальном адресном пространстве. Когда пользователь регистрируется в системе, автоматически создается процесс, в котором выполняется оболочка (shell), например, /bin/bash.

В Linux поддерживается классическая схема мультипрограммирования. Linux поддерживает параллельное (или квазипараллельное при наличии только одного процессора) выполнение процессов пользователя. Каждый процесс выполняется в собственном виртуальном адресном пространстве, т.е. процессы защищены друг от друга и крах одного процесса никак не повлияет на другие выполняющиеся процессы и на всю систему в целом. Один процесс не может прочитать что-либо из памяти (или записать в нее) другого процесса без "разрешения" на то другого процесса. Санкционированные взаимодействия между процессами допускаются системой.

Ядро предоставляет системные вызовы для создания новых процессов и для управления порожденными процессами. Любая программа может начать выполняться только если другой процесс ее запустит или произойдет какое-то прерывание (например, прерывание внешнего устройства).

В связи с развитием SMP (Symmetric Multiprocessor Architectures) в ядро Linux был внедрен механизм нитей или потоков управления (threads). Нить - это процесс, который выполняется в виртуальной памяти, используемой вместе с другими нитями процесса, который обладает отдельной виртуальной памятью.

Если интерпретатору (shell) встречается команда, соответствующая выполняемому файлу, интерпретатор выполняет ее, начиная с точки входа (entry point). Для C-программ entry point - это функция main. Запущенная программа тоже может создать процесс, т.е. запустить какую-то программу и ее выполнение тоже начнется с функции main.

Для создания процессов используются два системных вызова: fork() и exec. fork() создает новое адресное пространство, которое полностью идентично адресному пространству основного процесса. После выполнения этого системного вызова мы получаем два абсолютно одинаковых процесса - основной и порожденный. Функция fork() возвращает 0 в порожденном процессе и PID (Process ID - идентификатор порожденного процесса) - в основном. PID - это целое число. Теперь, когда мы уже создали процесс, мы можем запустить программу с помощью вызова exec. Параметрами функции exec является имя выполняемого файла и, если нужно, параметры, которые будут переданы этой программе. В адресное пространство порожденного с помощью fork() процесса будет загружена новая программа и ее выполнение начнется с точки входа (адрес функции main).

### **2.8 Лабораторная работа №15-16 (4 часа).**

**Тема:** «Синхронизация потоков»

#### **2.8.1 Цель работы:**

Изучить многозадачность и многопоточность в ОС Windows 7

#### **2.8.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Выполнить практическую часть
3. Сделать вывод

#### **2.8.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;

4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### **2.8.4 Описание (ход) работы:**

Изучили возможности поддержки многозадачного и многопоточного режимов в среде Windows7. Получили навыки разработки многозадачных и многопоточных приложений

### **2.9 Лабораторная работа №17-18 (4 часа).**

**Тема:** «Организация памяти»

#### **2.9.1 Цель работы:**

Изучить симметричную мультипроцессорную обработку, систему прерываний.

Таймер/счётчик

#### **2.9.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Ответить на контрольные вопросы.

#### **2.9.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### **2.9.4 Описание (ход) работы:**

Знакомство с особенностями многопоточной обработки информации на многоядерных процессорах под управлением ОС MS Windows и методом оценки трудоемкости алгоритмов

### **2.10 Лабораторная работа №19-20 (4 часа).**

**Тема:** «Алгоритмы организации памяти»

#### **2.10.1 Цель работы:**

Научиться работать с виртуальной машиной

#### **2.10.2 Задачи работы:**

1. Установить виртуальную машину.
2. Установить на виртуальном компьютере операционную систему WindowsXPили WindowsVista.
3. Установить на следующем виртуальном компьютере операционную систему WindowsServer2003 или WindowsServer2008.
4. Настроить сетевое соединение компьютеров.
5. Проверить взаимодействие компьютеров через сеть.
6. Создать копии файлов виртуальных машин на мобильном запоминающем устройстве - DVD-диске или Flash-носителе.

#### **2.10.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### 2.10.4 Описание (ход) работы:

Создание виртуальных компьютеров для лабораторного комплекса необходимо выполнить в следующей последовательности.

1. Реализуйте действия, необходимые для установки MicrosoftVirtualPC2004/2007 на компьютере учебного класса с операционной системой WindowsXили WindowsVista. Установка выполняется путем запуска установочного файла *setup.exe* из дистрибутива MicrosoftVirtualPC 2004/2007, являющегося бесплатным и свободно распространяемым продуктом.

При запуске виртуальной машины появляется консоль управления виртуальными компьютерами, предоставляющая возможность установки любых операционных систем и работы с ними после установки как по отдельности, так и в составе компьютерной сети.

2. Для установки операционной системы на виртуальной платформе необходимо выбрать пункт *New* и далее *Create a virtual machine*. При переносе уже имеющейся машины на другой компьютер выбирают пункт *Add an existing virtual machine*, позволяющий добавить в данный контейнер ранее созданный виртуальный компьютер.

#### 2.11 Лабораторная работа №21-22 (4 часа).

**Тема:** «Основные концепции организации ввода-вывода»

##### 2.11.1 Цель работы:

Научиться работать с виртуальной машиной

##### 2.11.2 Задачи работы:

1. Ознакомиться с теоретической частью:
  - Настройка сетевых интерфейсов
  - Расположение конфигурационных файлов
  - Команда **ifconfig**
  - Настройка локального интерфейса **lo**
  - Настройка интерфейса платы **Ethernet** локальной сети (**eth0**)
  - Интерфейс для последовательного порта
  - Настройка маршрутизации
  - Настройка службы имен
  - Тестирование сетевого соединения
  - Утилита **netconf**
2. Протестировать данный теоретический материал средствами ОС Linux

##### 2.11.3 Перечень приборов, материалов, используемых в лабораторной работе:

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

##### 2.11.4 Описание (ход) работы:

- Настройка сетевых интерфейсов
- Расположение конфигурационных файлов
- Команда **ifconfig**
- Настройка локального интерфейса **lo**
- Настройка интерфейса платы **Ethernet** локальной сети (**eth0**)
- Интерфейс для последовательного порта
- Настройка маршрутизации
- Настройка службы имен

- Тестирование сетевого соединения
- Утилита **netconf**

## **2.12 Лабораторная работа №23-24 (4 часа).**

**Тема:** «Организация ввода-вывода»

### **2.12.1 Цель работы:**

Научиться работать с программой администрирования

### **2.12.2 Задачи работы:**

1. через справку выписать назначение данной программы
2. задать объект- процессор и какие-либо 3 счетчика для него
3. снять показания в режиме покоя
4. загрузить 3 процесса, снять средние показания в режиме нагрузки
5. для объекта процессы сделать тоже самое
6. с любым объектом провести те же операции
7. составить результирующую таблицу

### **2.12.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

### **2.12.4 Описание (ход) работы:**

Получить результаты работы различных объектов компьютера в режимах покоя и нагрузки

## **2.13 Лабораторная работа №25-26 (4 часа).**

**Тема:** «Установка и настройка операционной системы»

### **2.13.1 Цель работы:**

Научиться работать с процессами Linux.

### **2.13.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Выполнить задания
3. Ответить на вопросы

### **2.13.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

### **2.13.4 Описание (ход) работы:**

Изучение процессов в операционной системе Linux.

## **2.14 Лабораторная работа №27-28 (4 часа).**

**Тема:** «Использование системы»

### **2.14.1 Цель работы:**

Изучить симметричную мультипроцессорную обработку

#### **2.14.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Ответить на контрольные вопросы.

#### **2.14.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### **2.14.4 Описание (ход) работы:**

Знакомство с особенностями многопоточной обработки информации на многоядерных процессорах под управлением OCMSWindowsи методом оценки трудоемкости алгоритмов

### **2.15 Лабораторная работа №29-30 (4 часа).**

**Тема:** «Основные понятия безопасности»

#### **2.15.1 Цель работы:**

Изучить мультипрограммные вычислительные процессы

#### **2.15.2 Задачи работы:**

1. Ознакомиться с теоретической частью:
  - Получение обобщенной информации о компонентах вычислительного процесса.
  - Получение и анализ информации о процессах и потоках.
  - Детальное исследование вычислительного процесса.
  - Управление журналами счетчиков и оповещений.
2. Выполнить задания
3. Ответить на контрольные вопросы

#### **2.15.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

#### **2.15.4 Описание (ход) работы:**

Изучение и исследование вычислительного процесса, его компонентов; анализ процессов и потоков; управление журналами.

### **2.16 Лабораторная работа №31-34 (8 часа).**

**Тема:** «Обеспечение безопасности»

#### **2.16.1 Цель работы:**

Изучить систему прерываний. Таймер/счётчик

#### **2.16.2 Задачи работы:**

1. Ознакомиться с теоретической частью
2. Выполнить задание

**2.16.3 Перечень приборов, материалов, используемых в лабораторной работе:**

1. Операционная система Linux Ubuntu 14.04
2. Офисный пакет OpenOffice.org
3. Операционные системы Windows 7, XP;
4. Интегрированный пакет MSOffice\$
5. Виртуальная машина VMware Player

**2.16.4 Описание (ход) работы:**

На практическом примере познакомиться с системой прерывания