Министерство образования и науки Российской Федерации

ТОМСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
СИСТЕМ УПРАВЛЕНИЯ И РАДИОЭЛЕКТРОНИКИ (ТУСУР)

ФАКУЛЬТЕТ ДИСТАНЦИОННОГО ОБУЧЕНИЯ (ФДО)

Ю. Б. Гриценко

ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ, СЕТИ И ТЕЛЕКОММУНИКАЦИИ

Учебное пособие

Томск
2015

УДК [004.7 + 621.391.1:519.8](075.8)
ББК 32.973.1я73
Г 858

Рецензенты:
Тарасенко В. Ф., докт. техн. наук, профессор кафедры теоретической кибернетики Национального исследовательского Томского государственного университета;
Сенченко П. В., канд. техн. наук, доцент, декан факультета систем управления ТУСУР.

Гриценко Ю. Б.

Г 858 Вычислительные системы, сети и телекоммуникации : учебное пособие / Ю. Б. Гриценко. — Томск : ФДО, ТУСУР, 2015. — 134 с.

В пособии рассмотрены понятия системы, вычислительной системы, архитектуры электронных вычислительных машин, приведено описание организации памяти и процесса управления устройствами ввода-вывода, а также уделено внимание принципам построения вычислительных сетей и телекоммуникаций.

Учебное пособие содержит теоретическую составляющую дисциплины «Вычислительные системы, сети и телекоммуникации», изучаемой студентами направлений 080500.62 «Бизнес-информатика» и 231000.62 «Программная инженерия».

УДК [004.7 + 621.391.1:519.8](075.8)
ББК 32.973.1я73

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Современное общество живет в век информации. Умение качественно управлять информационными ресурсами — одно из важнейших направлений деятельности человека. В настоящий момент идет бурное развитие систем управления информацией. Управление информацией строится на основе вычислительных систем с использованием всевозможных сетей и телекоммуникаций. В свою очередь, вычислительная система состоит из двух основных компонент — аппаратного (электронные и механические части) и программного обеспечения (программы, процедуры, правила и документация системы обработки информации).

Изучение дисциплины «Вычислительные системы, сети и телекоммуникации» представляет собой основу для изучения всего процесса управления информационными ресурсами и является базовым курсом, который предшествует таким дисциплинам, как «Операционные системы и сети», «Архитектура вычислительных систем».

Учебное пособие состоит из четырех глав.

В первой главе рассмотрены основные принципы построения вычислительных систем, включая историю развития вычислительных систем, обзор архитектур электронных вычислительных машин и процессоров.

Вторая глава содержит описание организации памяти, применение единиц измерения информации, способы адресации в различных режимах и демонстрацию процесса управления памятью в ОС Windows на платформе NT.

В третьей главе приведено описание и классификация устройств ввода-вывода, представлена организация дисковых устройств и файловых систем, изложен процесс управления устройствами ввода-вывода и файловыми системами.

Четвертая глава посвящена принципам построения вычислительных сетей и телекоммуникаций с использованием модели OSI, в главе рассматривается как физическая, так и логическая организация сети, а также использование протокола TCP/IP четвертой версии для организации сетевого взаимодействия.

Описание протокола TCP/IP шестой версии будет дано в курсе «Операционные системы и сети», который является логическим продолжением курса «Вычислительные системы, сети и телекоммуникации».

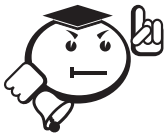
Соглашения, принятые в книге

Для улучшения восприятия материала в данной книге используются пиктограммы и специальное выделение важной информации.

Эта пиктограмма означает определение или новое понятие.

Пример

Эта пиктограмма означает пример. В данном блоке автор может привести практический пример для пояснения и разбора основных моментов, отраженных в теоретическом материале.



В блоке «На заметку» автор может указать дополнительные сведения или другой взгляд на изучаемый предмет, чтобы помочь читателю лучше понять основные идеи.

Контрольные вопросы по главе

Глава 1

ПРИНЦИПЫ ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СИСТЕМ

1.1 Общее представление о вычислительной системе

Согласно Большому Российскому энциклопедическому словарю происхождение слова «*система*» имеет греческие корни и означает множество элементов, находящихся в отношениях и связях друг с другом, которое образует определённую целостность, единство.

Вообще существует несколько десятков различных определений понятия «система», используемых в зависимости от контекста, области знаний и целей исследования [1].

Перед тем, как перейти к рассмотрению понятия «вычислительная система» (ВС), приведем некоторые понятия, часто использующиеся для характеристики системы [2].

Элемент системы *— часть системы, имеющая определенное функциональное назначение. Сложные элементы систем, в свою очередь состоящие из более простых взаимосвязанных элементов, часто называют подсистемами.*

Организация системы *— внутренняя упорядоченность, согласованность взаимодействия элементов системы, проявляющаяся, в частности, в ограничении разнообразия состояний элементов в рамках системы.*

Структура системы *— состав, порядок и принципы взаимодействия элементов системы, определяющие основные свойства системы.*

Если отдельные элементы системы разнесены по разным уровням и внутренние связи между элементами организованы только от вышестоящих к нижестоящим уровням, и наоборот, то говорят об иерархической структуре системы.

Архитектура системы *— совокупность свойств системы, существенных для пользователя.*

Целостность системы *— принципиальная несводимость свойств системы к сумме свойств отдельных ее элементов и, в то же время, зависимость свойств каждого элемента от его места и функции внутри системы.*

Вычислительная система *представляет собой совокупность аппаратных и программных средств, в окружении которых выполняется результирующая программа, порождаемая системой программирования на основании кода исходной программы, созданного разработчиком, а также объектных модулей и библиотек, входящих в состав системы программирования* [3].

Как видно из определения, вычислительная система имеет два типа средств[1]: аппаратные и программные. Взаимодействие средств обоих типов обеспечивается через аппаратно-программный интерфейс (рис. 1.1).

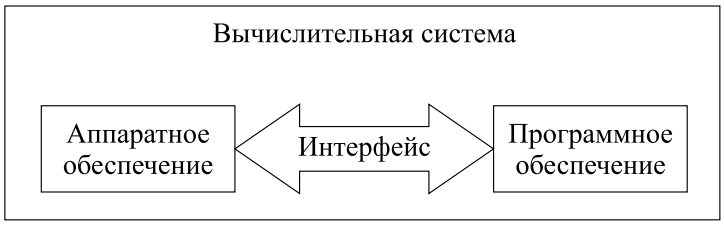


Рис. 1.1 – Структура вычислительной системы

Под ***интерфейсом*** *понимают совокупность способов и методов взаимодействия двух систем, устройств или программ для обмена информацией между ними.*

При использовании понятия аппаратно-программного интерфейса применительно к вычислительной машине оно может быть заменено на понятие «*внутримашинный системный интерфейс*», под которым понимают совокупность унифицированных технических средств, разъёмов и прочего оборудования, используемых для сопряжения устройств в вычислительной системе, и программных средств, таких как операционная система (ОС), драйверы, утилиты и т. п.

Центральным устройством большинства вычислительных систем является электронная вычислительная машина (ЭВМ) или компьютер.

[1]В англоязычной литературе, а часто теперь уже и российской, под терминами «аппаратные и программные средства» используются слова Hard&Soft.

Рассмотрим определения аппаратного и программного обеспечения.

Аппаратное обеспечение (аппаратные средства) *— это электронные и механические части вычислительного устройства, входящие в состав системы или сети.*

Аппаратное обеспечение включает: компьютеры и логические устройства, внешние устройства и диагностическую аппаратуру, энергетическое оборудование, батареи и аккумуляторы.

Программное обеспечение *— это совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ (ГОСТ 19781—90).*

Помимо аппаратного и программного обеспечения при функционировании вычислительной системы могут быть выделены еще несколько видов обеспечения: математическое, информационное, лингвистическое, организационное и методическое, правовое и т. п. Определения этих видов обеспечения приведены в учебнике В. Л. Бройдо [2].

Вычислительные системы могут строиться на основе нескольких процессоров или на основе нескольких самостоятельных компьютеров. В первом случае говорят о *многопроцессорной ВС*, а во втором — о *многомашинной ВС*.

В ***многопроцессорной ВС*** *имеется несколько процессоров, информационно взаимодействующих между собой либо на уровне регистров процессорной памяти, либо на уровне оперативной памяти.*

Многомашинная ВС *содержит некоторое число компьютеров, информационно взаимодействующих между собой. В многомашинных ВС каждый компьютер работает под управлением своей операционной системы.*

Информационное взаимодействие компьютеров в многомашинной ВС может быть организовано на уровне процессоров, оперативной памяти (ОП) или каналов связи.

1.2 История развития вычислительных систем

Одними из первых простейших приспособлений для вычислений были счётные палочки, которые и сегодня используются для обучения счёту. Постепенно из простейших приспособлений для счёта рождались всё более сложные устройства: счёты, логарифмическая линейка, механический арифмометр, электронный компьютер.

Эру появления электронных компьютеров предваряет период механических калькуляторов (1930–1960 годы) (рис. 1.2). Тогда же и появляется в обиходе слово «computer» (буквально — «вычислитель»). Так называлась должность людей, которые использовали калькуляторы для выполнения математических вычислений.

Рис. 1.2 – Счётная машинка Феликс-М[1]

Перед Второй мировой войной начались разработки первых электрических аналоговых компьютеров. На тот момент механические и электрические аналоговые компьютеры считались наиболее современными машинами, и многие считали, что это будущее вычислительной техники. Основные разработки велись параллельно в трех странах: Германии, Великобритании и США.

Сначала первые компьютеры строились на основе релейных переключателей, но впоследствии элементная база стала строиться на электровакуумных лампах.

В 1939 году Джон Винсент Атанасов и Клиффорд Берри из Университета штата Айова разработали *ABC (Atanasoff–Berry Computer)*. Это был первый в мире электронный цифровой компьютер. Конструкция насчитывала более 300 электровакуумных ламп, в качестве памяти использовался вращающийся барабан. Несмотря на то, что машина ABC не была программируемой, она была первой, использующей электронные лампы в сумматоре (рис. 1.3).

Первым работающим компьютером, управляемым программой, считается *Z3*, который был разработан в 1941 году немецким инженером Конрадом Цузе (рис. 1.4). Во многих отношениях *Z3* была подобна современным машинам, в ней впервые был представлен ряд новшеств, таких как арифметика с плавающей запятой. Замена сложной в реализации десятичной системы на двоичную сделала машины Цузе более простыми, а значит, более надёжными.

Программы для *Z3* хранились на перфорированной плёнке. В двух патентах 1936 года Конрад Цузе упоминал, что машинные команды могут храниться в той же памяти, что и данные, — предугадав тем самым то, что позже стало известно как архитектура фон Неймана и было впервые реализовано только в 1949 году в британском EDSAC.

В 1945 году на основе десятичной логики был разработан американский компьютер *ENIAC (Electronic Numerical Integrator and Computer)*, который часто называют первым электронным компьютером общего назначения (рис. 1.5). Эта раз-

[1]Источник: Wikimedia Commons (Музей Воды, Санкт-Петербург; Автор: George Shuklin). — URL: https://commons.wikimedia.org/wiki/File:Счётная_машинка_Феликс-M.jpg?uselang=ru (дата обращения: 16.03.2015).

работка публично доказала применимость электроники для масштабных вычислений. Созданная под руководством Джона Мокли и Джона Преспера Эккерта, эта машина была в 1000 раз быстрее, чем все другие машины того времени.

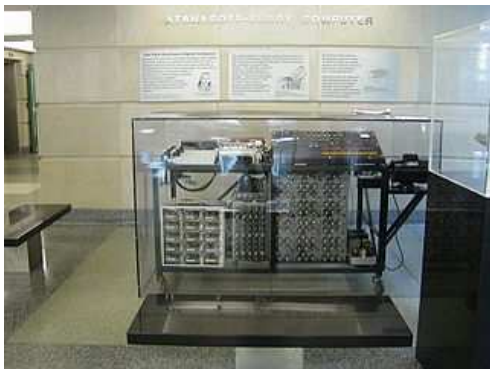


Рис. 1.3 – Компьютер Атанасова—Берри[1]

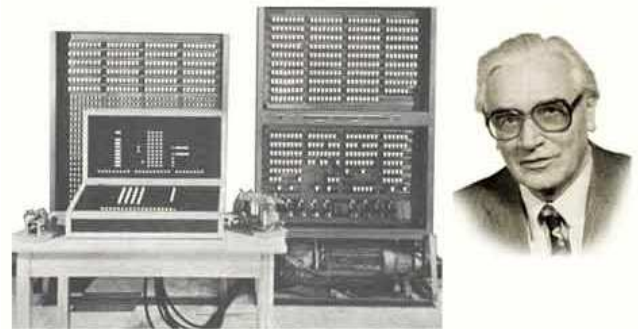


Рис. 1.4 – Конрад Цузе с *Z3*[2]

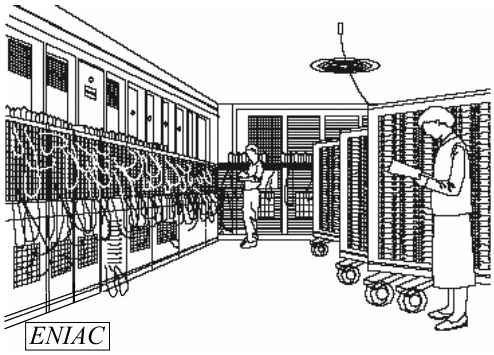


Рис. 1.5 – Компьютер ENIAC[3]

[1]Источник: Wikimedia Commons (Автор: User:Manop). – URL: https://commons.wikimedia.org/wiki/File:Atanasoff-Berry_Computer_at_Durhum_Center.jpg (дата обращения: 16.03.2015).

[2]Источник: URL: http://www.at-mix.de/zuse.htm (дата обращения: 16.03.2015).

[3]Источник: URL: http://physinfo.ulb.ac.be/divers_html/powerpc_programming_info/intro_to_risc/irt2_history2.html (дата обращения: 16.03.2015).

Первой работающей ЭВМ с архитектурой фон Неймана стал манчестерский *Baby*, созданный в Манчестерском университете в 1948 году.

В 1949 году был выпущен компьютер Манчестерский *Марк I*, который уже был полной системой, с запоминающими устройствами на основе электронно-лучевой трубки и магнитного барабана, а также с индексными регистрами (рис. 1.6).

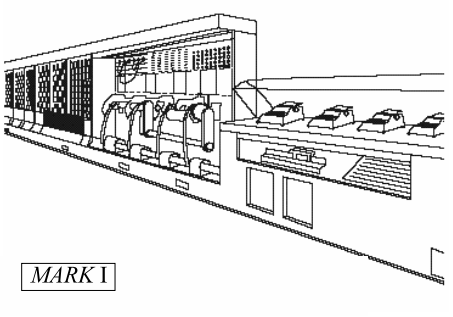


Рис. 1.6 – Компьютер MARK I[1]

Другим претендентом на звание «первый цифровой компьютер с хранимой программой» стал *EDSAC (Electronic Delay Storage Automatic Computer)*, разработанный и сконструированный в Кембриджском университете. В его основе была архитектура американского компьютера *EDVAC (Electronic Discrete Variable Automatic Computer)*, который являлся наследником архитектуры ENIAC. В отличие от своего предшественника ЭНИАКа это был компьютер на двоичной основе и располагающий единственным обрабатывающим блоком. Как и ЭНИАК, EDVAC был разработан командой инженеров и ученых во главе с Джоном Преспером Экертом и Джоном Уильямом Мокли при активной помощи математиков фон Неймана и Германа Голдстайна.

Первый универсальный программируемый компьютер в континентальной Европе был *Z4* Конрада Цузе, завершенный в сентябре 1950 года.

В ноябре того же года командой учёных под руководством Сергея Алексеевича Лебедева из Киевского института электротехники, УССР, была создана так называемая *МЭСМ (Малая электронная счётная машина)*. Она содержала около 6000 электровакуумных ламп и потребляла 15 кВт. Машина могла выполнять около 3000 операций в секунду.

Компьютеры на основе электронных ламп принято называть *первым поколением* развития вычислительной техники.

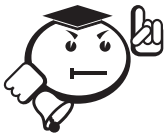
Следующим крупным шагом в истории компьютерной техники стало изобретение транзисторов в 1947 году. Они стали заменой хрупким и энергоёмким лампам. О компьютерах на транзисторах обычно говорят как о *втором поколении*, которое доминировало в 1950-х и начале 1960-х годах. Благо-

[1]Источник: URL: http://physinfo.ulb.ac.be/divers_html/powerpc_programming_info /intro_to_risc/irt2_history2.html (дата обращения: 16.03.2015).

даря транзисторам и печатным платам было достигнуто значительное уменьшение размеров и объёмов потребляемой энергии, а также повышение надёжности.

Бурный рост использования компьютеров начался с *третьего поколения* вычислительных машин. Начало этому положило изобретение интегральных схем, которые независимо друг от друга сделали лауреат Нобелевской премии Джек Килби и Роберт Нойс.

Появление микропроцессоров (изобретатель Тэд Хофф) привело к разработке микрокомпьютеров, которыми могли владеть небольшие компании или отдельные люди. Микрокомпьютеры, представители *четвёртого поколения*, первый из которых появился в 1970-х годах, стали повсеместным явлением в 1980-х годах и позже. Стив Возняк, один из основателей Apple Computer, стал известен как разработчик первого массового домашнего компьютера, а позже — первого персонального компьютера.



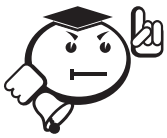
Настоящее время связывают с *пятым поколением* — поколением компьютеров со многими десятками параллельно работающих микропроцессоров, позволяющих строить эффективные системы обработки знаний; компьютеры на сверхсложных микропроцессорах с параллельно-векторной структурой, одновременно выполняющих десятки последовательных инструкций программы [2].

На начало 2014 года самый быстрый компьютер в мире создали китайские ученые (эксплуатируется в Национальном суперкомпьютерном центре в Гуанчжоу; разработчик Оборонный научно-технический университет Народно-освободительной армии Китая). Новый суперкомпьютер *Тяньхэ-2* побил рекорд вычислительной машины *Титан*, ранее созданной в США (эксплуатируется в Национальной лаборатории Оук-Ридж, Теннесси; разработчик Cray Inc.).

Компьютер Тяньхэ-2 (в переводе с китайского языка «Млечный путь-2») работает со скоростью 33.86 petaflop/s[1] [4]. Большинство деталей суперкомпьютера изготовлено в самом Китае, и лишь некоторые компоненты центрального процессора выпущены компанией Intel. Помимо Тяньхэ-2, в первую десятку нового рейтинга вошел еще один компьютер китайского производства, пять суперкомпьютеров из США, два из Германии и один из Японии. В России работают восемь крупнейших суперкомпьютеров мира из TOP-500. Самый производительный из них — «*Ломоносов*», установлен в Московском государственном университете, занимает 31-е место в рейтинге [5].

Подобные аппараты используются, в частности, для анализа процессов изменения климата, прогнозирования стихийных катаклизмов и космических исследований. Среди других сфер их применения также называют моделирование ядерных испытаний и разработку прототипов авиалайнеров [5].

[1] Квадриллионов операций с плавающей точкой в секунду.



Будущее связывают с *шестым поколением вычислительных систем*, основанным на оптоэлектронных компьютерах с массовым параллелизмом и нейронной структурой, с распределенной сетью большого числа несложных микропроцессоров, моделирующих архитектуру нейронных биологических систем.

1.3 Электронные вычислительные машины

ЭВМ в вычислительной системе может быть одна, но агрегирована с многофункциональным периферийным оборудованием. В качестве распространенного примера одномашинной ВС можно привести систему телеобработки информации [2]. Но в современное время, как правило, ВС строятся как многомашинные и/или многопроцессорные комплексы.

*Под **электронной вычислительной машиной** будем понимать комплекс технических средств, предназначенных для автоматической обработки информации в процессе решения вычислительных и информационных задач* [6].

Понятие «ЭВМ», как правило, является синонимом компьютера, хотя это не совсем верно. ЭВМ подразумевает использование электронных компонентов в качестве её функциональных узлов, однако компьютер может быть устроен и на других принципах — он может быть механическим, биологическим, оптическим, квантовым и т. п.

По принципу действия ЭВМ можно разделить на два основных типа: *аналоговые* и *цифровые*.

***Аналоговые вычислительные машины (АВМ)** — вычислительные машины, работающие с информацией в виде непрерывного ряда значений какой-либо физической величины.*

Чаще всего информация представляется электрическим напряжением (рис. 1.7). АВМ имеют точность вычисления (0,001–0,01) и, как правило, используются в основном в научно-исследовательских учреждениях, в составе различных стендов.

***Цифровые вычислительные машины (ЦВМ)** — вычислительные машины, работающие с информацией, представленной в дискретном (цифровом) виде (рис. 1.8).*

Рынок современных ЦВМ весьма динамичен. Каждый год стоимость вычислений сокращается на 15–20%, стоимость хранения единицы информации — на 40%.

Каждое десятилетие меняется поколение машин. Каждые год-два — основные типы микропроцессоров.

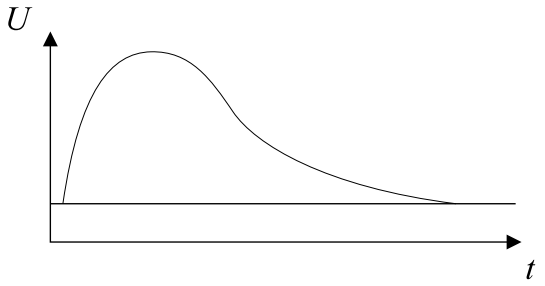


Рис. 1.7 – Аналоговая форма представления информации

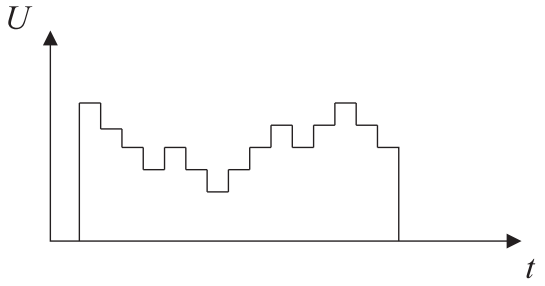
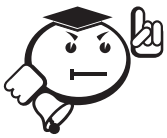


Рис. 1.8 – Цифровая форма представления информации

Применительно к ЦВМ широко используются два способа физического представления сигналов: *импульсный* и *потенциальный*.



При *импульсном способе* представления сигналов единичному значению некоторой двоичной переменной ставится в соответствие наличие импульса (тока или напряжения), нулевому — отсутствие импульса (рис. 1.9).

При *потенциальном способе* представления сигналов единичное значение двоичной переменной отображается высоким уровнем напряжения, а нулевое значение — низким уровнем (рис. 1.10).

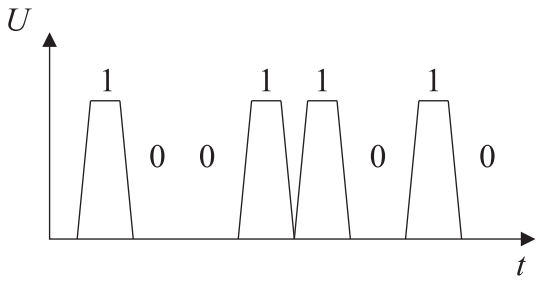


Рис. 1.9 – Импульсный способ представления сигнала

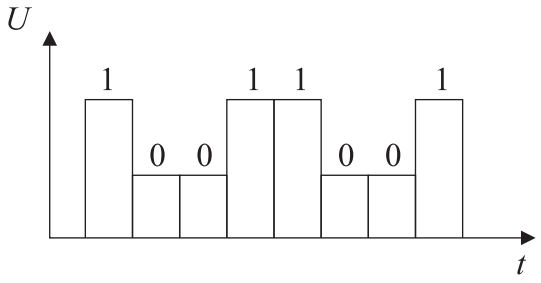


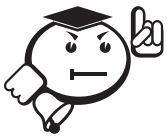
Рис. 1.10 – Потенциальный способ представления сигнала

Иногда в литературе выделяется третий тип ЭВМ, классифицируемый по принципу действия — *гибридные вычислительные машины*.

***Гибридные вычислительные машины (ГВМ)**, или вычислительные машины комбинированного действия, работают с информацией, представленной и в цифровой, и в аналоговой форме; они совмещают в себе достоинства АВМ и ЦВМ. ГВМ целесообразно использовать для решения задач управления сложными быстродействующими техническими комплексами* [2].

В настоящее время в обществе получили широкое распространение ЦВМ, обычно называемые просто электронными вычислительными машинами (ЭВМ), без упоминания об их цифровом характере.

Независимо от видов сигналов различают *последовательный* и *параллельный* коды передачи информации в ЭВМ.



При *последовательном коде* представления данных используются одиночные шины или линии передачи, в которых сигналы разнесены во времени. Обработка такой информации производится разряд за разрядом.

При *параллельном коде* представления данных предполагается параллельная и одновременная фиксация всех разрядов данных на различных шинах. Такая форма передачи информации позволяет ускорить обработку данных во времени, но увеличивает затраты на аппаратные средства пропорционально числу обрабатываемых разрядов.

В ЭВМ используют параллельно-последовательные коды представления информации. Данные поступают на обработку последовательно, но каждая часть данных представляется параллельным кодом.

Иногда ЭВМ по размерам и быстродействию делят также на несколько классов (это деление весьма условное): *супер-ЭВМ, большие ЭВМ, средние ЭВМ, мини-ЭВМ, микро-ЭВМ*.

Супер-ЭВМ — вычислительные системы, из категории больших ЭВМ, превосходящие по вычислительным характеристикам остальные системы. Отнесем их к списку TOP-500 [7]. Как правило, используются для решения крупномасштабных задач, обслуживание крупнейших информационных банков данных. Согласно *ГОСТу 15971—90* Супер-ЭВМ — это ЭВМ, относящаяся к классу вычислительных машин, имеющих самую высокую производительность, которая может быть достигнута на данном этапе развития технологии, и в основном предназначенных для решения сложных научно-технических задач.

Большие ЭВМ (мэйнфреймы, mainframe) появились как класс ЭВМ одними из первых и реализуются в виде нескольких стоек. Данный класс используется для решения научно-технических задач, для работы с большими базами данных, для управления вычислительными сетями и их ресурсами. На их основе могут быть реализованы территориальные центры обработки информации.

Очень часто в настоящее время технологии построения больших и суперкомпьютеров строятся на базе *кластерных решений*. Под кластером понимают объединение нескольких однородных элементов, которое может рассматриваться как самостоятельная единица, обладающая определёнными свойствами. На смену отдельным, независимым суперкомпьютерам приходят группы высокопроизводительных серверов, объединенных в кластер.

Средние ЭВМ используются для управления сложными техническими процессами. Данный класс возник в результате компромисса цена/быстродействие. В настоящее время трудно определить четкую грань между средними ЭВМ и большими, с одной стороны, и малыми — с другой.

Мини-ЭВМ, согласно *ГОСТу 15971—90*, — это ЭВМ, относящаяся к классу вычислительных машин, разрабатываемых из требования минимизации стоимости и предназначенных для решения достаточно простых задач. Миникомпьютеры составляют многочисленный и быстроразвивающийся класс ЭВМ. Их популярность объясняется малыми размерами, низкой стоимостью (по сравнению с большими и средними ЭВМ) и универсальными возможностями. Мини-ЭВМ ориентированы на использование в качестве управляющих вычислительных комплексов. Наряду с использованием мини-ЭВМ для управления технологическими процессами, они успешно применяются для вычислений в многопользовательских вычислительных системах, в системах автоматизированного проектирования, в системах моделирования объектов.

Класс ***Микро-ЭВМ*** возник с появлением микропроцессора. Определяющим признаком микро-ЭВМ является наличие одного или нескольких микропроцессоров. Микро-ЭВМ, благодаря малым размерам, высокой производительности, повышенной надежности и небольшой стоимости, нашли самое широкое распространение. У Микро-ЭВМ выделяют свои подклассы:

- *Многопользовательские микрокомпьютеры* — это мощные компьютеры, оборудованные несколькими видеотерминалами и функционирующие в режиме разделения времени, что позволяет эффективно работать на них сразу нескольким пользователям.
- *Персональные компьютеры* — однопользовательские компьютеры, удовлетворяющие требованиям общедоступности и универсальности применения. Делятся на стационарные и переносные (планшеты, ноутбуки, смартфоны и т. п.).

- *Рабочие станции* представляют собой однопользовательские компьютеры, часто специализированные для выполнения определенного вида работ (графических, инженерных, издательских и т. д.).
- *Серверы* — многопользовательские мощные компьютеры в вычислительных сетях, выделенные для обработки запросов от всех рабочих станций сети.
- *Сетевые компьютеры* — упрощенные компьютеры, обеспечивающие работу в сети и доступ к сетевым ресурсам, часто специализированные на выполнение определенного вида работ (защита сети от несанкционированного доступа, организация просмотра сетевых ресурсов, электронной почты и т. д.).

1.4 Архитектура ЭВМ

1.4.1 Определение архитектуры ЭВМ

В процессе создания вычислительных машин вместо независимой разработки аппаратуры, математического и программного обеспечения стала проектироваться система, состоящая из совокупности аппаратных и программных средств с концепцией их взаимодействия. Так возникло принципиально новое понятие — архитектура ВС.

Архитектура вычислительной системы *— это совокупность характеристик и параметров, определяющих функционально-логичную и структурно-организованную систему и затрагивающих в основном уровень параллельно работающих вычислителей.*

Понятие архитектуры охватывает общие принципы построения и функционирования, наиболее существенные для пользователя.

Основными компонентами архитектуры вычислительных машин и систем принято считать следующие компоненты:

- вычислительные и логические возможности: система команд, формат команд, способы адресации, назначение и состав регистров;
- аппаратные средства: структура, организация памяти, организация ввода-вывода, принципы управления;
- программное обеспечение (ПО): ОС, языки программирования, прикладное ПО.

В рамках понятия архитектуры ВС выделим понятие архитектуры ЭВМ.

Архитектура ЭВМ *— концептуальная структура вычислительной машины, определяющая проведение обработки информации и включающая методы преобразования информации в данные и принципы взаимодействия технических средств и программного обеспечения (ГОСТ 15971—90).*

Архитектура ЭВМ охватывает широкий круг проблем, связанных с построением комплекса аппаратных и программных средств и учитывающих множество факторов. Среди этих факторов важнейшими являются: стоимость, сфера применения, функциональные возможности, удобство эксплуатации, а *одним из главных компонентов архитектуры являются аппаратные средства*.

Следует различать понятия архитектуры вычислительного средства от понятия структуры. ***Структура*** определяет конкретный состав на некотором уровне и описывает связи. ***Архитектура*** определяет правила взаимодействия составных частей. Она регламентирует не все связи, а наиболее важные, которые должны быть известны для более эффективного использования вычислительного средства.

Наиболее известны два классических типа архитектур ЭВМ: *принстонская архитектура (архитектура фон Неймана)* и *гарвардская архитектура*.

В 30-х годах Гарвардский и Принстонский университеты разрабатывали архитектуру ЭВМ для военно-морской артиллерии. В Гарвардском университете разработчиком архитектуры являлся Говард Эйкен. В Принстонском университете над разработкой работал фон Нейман. Министерство обороны США выбрало разработку Принстонского университета (более известная как архитектура фон Неймана), так как она была проще в реализации. Гарвардская архитектура использовалась советским учёным А. И. Китовым в первом вычислительном центре министерства обороны СССР.

1.4.2 Принстонская архитектура (архитектура фон Неймана)

Джон фон Нейман — венгеро-американский математик, сделавший важный вклад в квантовую физику, квантовую логику, функциональный анализ, теорию множеств, информатику, экономику и другие отрасли науки. Наиболее известен как человек, с именем которого связывают архитектуру большинства современных компьютеров.

Архитектуру фон Неймана ассоциируют со схематическим изображением машины фон Неймана (состоящей из *блока управления*, *арифметико-логического устройства*, *памяти и устройств ввода-вывода*) и принципами организации ЭВМ. В ней реализуются следующие принципы организации архитектуры.

Принцип однородности памяти. Команды и данные хранятся в одной и той же памяти и внешне в памяти неразличимы. Распознать их можно только по способу использования; то есть одно и то же значение в ячейке памяти может использоваться и как данные, и как команда, и как адрес в зависимости лишь от способа обращения к нему. Это позволяет производить над командами те же операции, что и над числами, и, соответственно, открывает ряд дополнительных возможностей.

Принцип адресности. Структурно основная память состоит из пронумерованных ячеек, причем процессору в произвольный момент доступна любая ячейка. Двоичные коды команд и данных разделяются на единицы информации, называемые словами, и хранятся в ячейках памяти, а для доступа к ним используются номера соответствующих ячеек — адреса.

Принцип программного управления. Все вычисления, предусмотренные алгоритмом решения задачи, должны быть представлены в виде программы, состо-

ящей из последовательности управляющих слов — команд. Каждая команда предписывает некоторую операцию из набора операций, реализуемых вычислительной машиной. Команды программы хранятся в последовательных ячейках памяти вычислительной машины и выполняются в естественной последовательности, то есть в порядке их положения в программе. При необходимости, с помощью специальных команд, эта последовательность может быть изменена.

Принцип двоичного кодирования. Согласно этому принципу вся информация, как данные, так и команды, кодируется двоичными цифрами: 0 и 1. Каждый тип информации представляется двоичной последовательностью и имеет свой формат.

Со временем Джон Бэкус в архитектуре фон Неймана выделил «узкое место»[1]. Использование общей шины для команд и данных приводит к ограничению пропускной способности между процессором и оперативной памятью. Из-за того, что код программы и данные не могут быть доступны в одно и то же время, пропускная способность шины является значительно меньшей, чем скорость, с которой процессор может работать. Процессор постоянно вынужден ждать необходимых данных, которые будут переданы в память или из памяти. Так как скорость процессора и объём памяти увеличивались гораздо быстрее, чем пропускная способность между ними, это стало большой проблемой.

1.4.3 Гарвардская архитектура

Гарвардская архитектура была разработана Говардом Эйкеном в конце 1930-х годов в Гарвардском университете. В должности инженера IBM Говард Эйкен руководил работами по созданию первого американского компьютера «Марк I».

Отличительные признаки гарвардской архитектуры:

- *хранилище инструкций и хранилище данных представляют собой разные физические устройства*;
- *канал инструкций и канал данных физически разделены*.

В первом компьютере Эйкена «Марк I» для хранения инструкций использовалась перфорированная лента, а для работы с данными — электромеханические регистры. Это позволяло одновременно пересылать и обрабатывать команды и данные, благодаря чему значительно повышалось общее быстродействие компьютера.

Компьютер с гарвардской архитектурой может работать быстрее компьютера с принстонской архитектурой, поскольку доставка инструкций и доступ к данным используют разные каналы памяти. В гарвардской архитектуре характеристики устройств памяти для инструкций и памяти для данных не обязательно должны быть одинаковыми. Но такая архитектура имеет и недостаток — это высокая стоимость. При разделении каналов передачи команд и данных на кристалле процессора последний должен иметь почти вдвое больше выводов, так как шина адреса и шина данных составляют основную часть выводов микропроцессора.

[1] Backus, John W. «Can Programming Be Liberated from the von Neumann Style? A Functional Style and Its Algebra of Programs».

Способом решения этой проблемы стала идея использовать общие шину данных и шину адреса для всех внешних данных, а внутри процессора использовать шину данных, шину команд и две шины адреса. Такую концепцию стали называть *модифицированной гарвардской архитектурой*.

Существуют *гибридные архитектуры*, сочетающие достоинства как гарвардской, так и принстонской архитектур. Современные CISC-процессоры[1] обладают раздельной кэш-памятью 1-го уровня для инструкций и данных, что позволяет им за один рабочий такт получать одновременно и команду, и данные для её выполнения. То есть процессорное ядро формально является гарвардским, но программно оно принстонское, что упрощает написание программ. Обычно в данных процессорах одна шина используется и для передачи команд, и для передачи данных, что упрощает конструкцию системы. Современные варианты таких процессоров могут иногда содержать встроенные контроллеры сразу нескольких разнотипных шин для работы с различными типами памяти. Тем не менее и в этом случае шины, как правило, используются и для передачи команд, и для передачи данных без разделения, что делает данные процессоры ещё более близкими к принстонской архитектуре при сохранении плюсов гарвардской архитектуры.

1.4.4 Архитектурные свойства ЭВМ

Современные ЭВМ обладают некоторыми общими и индивидуальными свойствами архитектуры. К числу *общих архитектурных свойств и принципов* можно отнести [8]:

- Принцип хранимой программы. Согласно ему код программы и ее данные находятся в одном адресном пространстве в оперативной памяти.
- Принцип микропрограммирования. В состав процессора входит блок микропрограммного управления. Этот блок для каждой машинной команды имеет набор действий-сигналов, которые нужно сгенерировать для физического выполнения требуемой машинной команды.
- Линейное пространство памяти — совокупность ячеек памяти, которым последовательно присваиваются номера (адреса) 0, 1, 2, ...
- Последовательное выполнение программ. Процессор выбирает из памяти команды строго последовательно. Для изменения прямолинейного хода выполнения программ необходимо использовать специальные команды (команды условного и безусловного перехода). Процессор не различает команды и данные, поэтому важно в программе всегда четко разделять пространство данных и команд.
- Безразличие к целевому назначению данных. Машине все равно, какую логическую нагрузку несут обрабатываемые ею данные.

Перечень *индивидуальных свойств и принципов* ЭВМ весьма велик, наиболее значимыми являются [8]:

1. Суперскалярная архитектура. Важным элементом архитектуры, появившимся в микропроцессоре Intel 486 (i486), стал конвейер — специальное устрой-

[1] См. пункт 1.5.1. CISC-процессоры.

ство, при котором выполнение команд в микропроцессоре разбивается на несколько этапов. Преимуществом такого подхода является то, что очередная команда после ее выборки попадает в блок декодирования. Таким образом, блок выборки свободен и может выбрать следующую команду. В результате на конвейере могут находиться в различной стадии выполнения пять команд. Микропроцессоры, имеющие один конвейер, называются скалярными, а два и более — суперскалярными. В i486 имеются следующие этапы выполнения команды:

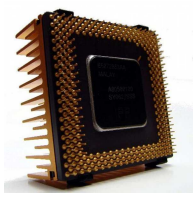
- выборка команд из оперативной или кэш-памяти;
- декодирование команды;
- генерация адреса;
- выполнение операции с помощью АЛУ (арифметико-логическое устройство);
- запись результата.

2. Раздельное кэширование кода и данных. Кэширование — это способ увеличения быстродействия системы за счет хранения часто используемых данных и кодов в так называемой «кэш-памяти первого уровня», находящейся внутри микропроцессора.
3. Предсказание правильного адреса перехода. Под переходом понимают запланированное алгоритмом изменение последовательного характера выполнения программы. Типичная программа на каждые 6–8 команд содержит одну команду перехода. Последствия этого предсказать нетрудно: при наличии конвейера через каждые 6–8 команд его нужно очищать и заполнять заново в соответствии с адресом перехода. Все преимущества конвейеризации теряются. Поэтому в архитектуру Pentium был введен блок предсказания переходов. Вероятность правильного предсказания перехода составляет около 80%.

1.5 Архитектуры процессоров

1.5.1 CISC-процессоры

Микропроцессор *— это устройство, представляющее собой одну или несколько больших интегральных схем, выполняющих функции процессора ЭВМ.*



Существуют процессоры различных архитектур. Среди них можно выделить два основных типа построения архитектуры: CISC и RISC.

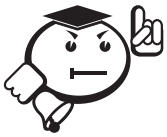
CISC-архитектура (Complex Instruction Set Computing — компьютер с полным набором команд) — концепция проектирования процессоров, которая обладает следующими свойствами:

- имеет большое число различных по формату и длине команд, длины команды имеют нефиксированное значение;
- поддерживается кодировка арифметических действий в одной команде и большое количество режимов адресации;
- небольшое число регистров, каждый из которых выполняет строго определённую функцию.

На основе этой архитектуры разработаны процессоры Intel x86[1] и процессоры Motorola MC680x0. Формально все x86-процессоры являлись CISC-процессорами, однако процессоры, начиная с Intel Pentium Pro, являются CISC-процессорами с RISC-ядром. Они непосредственно перед исполнением преобразуют CISC-инструкции процессоров x86 в более простой набор внутренних инструкций RISC. При этом одна команда x86 может порождать несколько RISC-команд. Исполнение команд происходит на суперскалярном конвейере одновременно по несколько штук. Необходимость таких преобразований обуславливается потребностью увеличения скорости обработки CISC-команд. В итоге, такой подход и позволил поднять производительность CPU.

К недостаткам CISC-архитектуры можно отнести высокую стоимость аппаратной части, сложность с распараллеливанием вычислений, отставание от RISC-процессоров по количеству выполняемых операций в секунду.

1.5.2 RISC-процессоры



RISC-архитектура (Restricted Instruction Set Computer — компьютер с сокращённым набором команд) — архитектура процессора, в которой быстродействие увеличивается за счёт упрощения инструкций, чтобы их декодирование было более простым, а время выполнения — меньшим.

Первые RISC-процессоры даже не имели инструкций умножения и деления. Это также облегчает повышение тактовой частоты и делает более эффективной суперскалярность.

Нередко слова «сокращённый набор команд» понимаются как минимизация количества инструкций в системе команд. В действительности, инструкций у многих RISC-процессоров больше, чем у CISC-процессоров [9].

Термин «сокращённый» в названии описывает тот факт, что сокращён объём (и время) работы, выполняемый каждой отдельной инструкцией, — как максимум один цикл доступа к памяти, — тогда как сложные инструкции CISC-процессоров могут требовать сотен циклов доступа к памяти для своего выполнения [10].

Можно выделить характерные особенности RISC-процессоров:

- Фиксированная длина машинных инструкций и простой формат команды.

[1] Исключая современные Intel Pentium 4, Pentium D, Core, AMD Athlon, Phenom, которые являются гибридными.

- Специализированные команды для операций с памятью — чтения или записи. Операции вида Read-Modify-Write («прочитать-изменить-записать») отсутствуют. Любые операции «изменить» выполняются только над содержимым регистров.
- Большое количество регистров общего назначения (32 и более).
- Отсутствует поддержка операций вида «изменить» над укороченными типами данных — байт, 16-битное слово. Так, например, система команд DEC Alpha содержала только операции над 64-битными словами и требовала разработки и последующего вызова процедур для выполнения операций над байтами, 16- и 32-битными словами.
- Отсутствие микропрограмм внутри самого процессора. То, что в CISC-процессоре исполняется микропрограммами, в RISC-процессоре исполняется как обыкновенный машинный код, не отличающийся принципиально от кода ядра ОС и приложений.

В настоящее время многие архитектуры процессоров являются RISC-подобными, к примеру, ARM, DEC Alpha, SPARC, AVR, MIPS, POWER и PowerPC.

Методика построения системы команд RISC противоположна методике CISC. Различие этих концепций состоит в методах программирования, а не в реальной архитектуре процессора. Практически все современные процессоры эмулируют наборы команд как RISC, так и CISC типа.

Существуют и другие альтернативные архитектуры процессоров: Minimal Instruction Set Computer (MISC), Zero Instruction Set Computer (ZISC), Ultimate RISC (другое название OISC), Transport Triggered Architecture (TTA), Very Long Instruction Word (VLIW), Systolic array, Reconfigurable computing, Dataflow architecture и т. п.

1.5.3 Микропроцессоры семейства x86–64

x86 (Intel 80x86) — архитектура процессора с одноимённым набором команд, впервые реализованная в процессорах компании Intel.

Название образовано от двух цифр, которыми заканчивались названия процессоров Intel ранних моделей — 8086, 80186, 80286, 80386, 80486. За время своего существования набор команд постоянно расширялся, сохраняя совместимость с предыдущими поколениями.

Помимо Intel, архитектура также была реализована в процессорах других производителей: AMD, VIA, Transmeta, IDT и др. В настоящее время для этой архитектуры существует ещё одно название — *IA-32 (Intel Architecture-32)*.

x86 — это CISC-архитектура. Доступ к памяти происходит по «словам». Современные процессоры включают в себя декодеры команд x86 для преобразования их в упрощённый внутренний формат с последующим их выполнением.

Опишем историю развития микропроцессоров семейства x86.

8086 (1978 г.) — 16-разрядный процессор сначала работал на частотах 4,77 МГц, затем на 8 и 10 МГц. Изготавливался по технологии 3 мкм и имел 29 000 транзисторов.

8088 (1979 г.) — процессор работал на тех же частотах, что и 8086, но использовал 8-разрядную шину данных (внутренняя шина процессора осталась 16-раз-

рядной) для обеспечения большей совместимости с имевшейся в то время в ходу периферией. Благодаря более низкой цене широко использовался в ранних системах IBM PC вместо 8086.

80186/80188 (1982 г.) — процессоры первоначально не получили широкого распространения, но оказались чрезвычайно удачными для использования во встроенных системах и в различных модификациях выпускаются до настоящего времени. В эти процессоры было первоначально добавлено несколько новых команд, повышена тактовая частота.

80286 (1982 г.) — процессор работал на частотах 6, а затем 8, 10, 12, 16, 20 МГц. Производился по техпроцессу 1,5 мкм и содержал около 134 тыс. транзисторов. С его появлением появилось такое понятие, как защищённый режим (protected mode) и виртуальная память. Производительность процессора по сравнению с 8086 увеличилась в несколько раз.

80386 (1985 г.) — первый 32-разрядный процессор, работал на частотах 16–40 МГц. Произвел кардинальные изменения в семействе процессоров x86. Основные принципы, заложенные в этом чипе, сохранились и до наших дней. Первые 386-е процессоры содержали серьёзную ошибку, приводящую к невозможности функционирования в защищенном режиме. Исправленная версия называлась i386DX. Также выпускались более дешевые процессоры i386SX с урезанной до 16 бит внешней шиной данных и 24-битной шиной адреса. i386 — первый процессор, который мог использовать кэш-память (расположенную на внешнем чипе).

80486 (1989 г.) — процессор является усовершенствованным 80386 процессором и первым скалярным процессором Intel. Имел встроенный блок вычислений с плавающей запятой (*FPU — Floating Point Unit*) и, впервые, встроенную кэш-память (8 Кбайт). 80486 — первый процессор Intel, для которого была применена технология умножения частоты шины FSB — Front Side Bus.

i586 или Pentium[1] (1993 г.) — первый суперскалярный и суперконвейерный процессор Intel. У Pentium два конвейера, что позволяет ему при одинаковых частотах в идеале быть вдвое производительней 80486, выполняя сразу 2 инструкции за такт.

Кроме того, особенностью процессора Pentium являлся полностью переработанный и очень мощный на то время блок FPU, производительность которого оставалась недостижимой для конкурентов вплоть до конца 1990-х годов.

Pentium Pro (1995 г.) — первый процессор шестого поколения (P6). Идеи и технологии, заложенные в данный чип, определили архитектуры всех современных x86-процессоров: блоки предсказания ветвлений, переименование регистров, RISC-ядро, интегрированная в один корпус с ядром кэш-память второго уровня. Однако технологическая сложность ядра данного процессора привела к сравнительно невысокому выходу годных чипов при технологиях того времени, что сказалось на высокой цене Pentium Pro. При этом процессор обладал достаточно низкой производительностью при исполнении 16-разрядного кода.

[1] Intel отказалась от номерных названий типа 8086, 80286 и др., потому что не могла запатентовать числа.

Pentium MMX[1] (1997 г.) — процессор пятого поколения, модификация ядра Pentium. Был добавлен новый блок целочисленных матричных вычислений MMX и увеличен до 32 Кбайт объём кэш-памяти первого уровня.

i686 или Pentium II (1997 г.) — модификация ядра Pentium Pro с целью сделать его более доступным. Интегрированный кэш был вынесен на отдельную микросхему с пониженной в два раза частотой. Это упростило и удешевило процессор, хотя и сделало его более медленным, чем Pentium Pro. Первые процессоры Pentium II выпускались с кэш-памятью второго уровня емкостью 256 Кбайт, затем её объём был увеличен до 512 Кбайт. В ядро Pentium II был добавлен блок MMX.

Celeron (1998 г.) — семейство низкобюджетных x86-совместимых процессоров компании Intel, имеющее большое количество модификаций. Одной из причин невысокой цены является их более низкая по сравнению со старшими моделями производительность, что достигается двумя основными методами: искусственным снижением частоты шины процессора и блокировкой части кэш-памяти второго уровня.

Pentium III (1999 г.) — процессор, изготовленный изначально по технологическому процессу 0,18 мкм, отличается от Pentium II главным образом добавлением инструкций *SSE*[2] *(Streaming SIMD Extensions)*. *SIMD (Single Instruction, Multiple Data)* набор инструкций, разработанный Intel и впервые представленный в процессорах серии Pentium III. Технология SSE позволяла преодолеть 2 основные проблемы MMX — при использовании MMX невозможно было одновременно использовать инструкции сопроцессора, так как его регистры были общими с регистрами MMX. Поздние процессоры этой серии изготавливались по технологическому процессу 0,13 мкм, получили интегрированную в кристалл ядра полночастотную кэш-память (сначала 256 Кбайт, затем — 512 Кбайт) и послужили прообразом процессоров архитектуры Pentium M.

Pentium M (2003 г.) — очень сильно доработанная версия процессора Pentium III на ядре Tualatin, разработанная для использования в мобильных компьютерах.

Pentium 4 (2000 г.) — принципиально новый процессор с гиперконвейеризацией (hyperpipelining) — с конвейером, состоящим из 20 ступеней. Согласно заявлениям процессоры Intel, основанные на данной технологии, позволяют добиться увеличения частоты примерно на 40% относительно семейства P6 при одинаковом технологическом процессе (при «правильной» загрузке процессора). На практике же, первое поколение процессоров работало даже медленнее, чем Pentium III. Позже были дополнены поддержкой Hyper-threading и 64-битного кода.

Core (2006 г.) — процессоры Core являются преемниками процессоров предыдущего поколения, представленных моделями Pentium и Celeron. Для серверов имеются более «продвинутые» версии процессоров Core под маркой Xeon. В основе процессоров лежит переработанное ядро Pentium M. Таким образом, ядро P6, использованное ещё в процессорах Pentium Pro, продолжило свою эволюцию, нарастив частоту со 150 МГц до 3,2 ГГц и обзаведясь новой системной шиной,

[1] MMX (Multimedia Extensions) — мультимедийные расширения. Коммерческое название дополнительного набора инструкций, выполняющих характерные для процессов кодирования/декодирования потоковых аудио/видеоданных действия за одну машинную инструкцию.

[2] Набор инструкций как альтернатива набору инструкций 3DNow! фирмы AMD, который был представлен годом раньше.

поддержкой многоядерности, мультимедийных инструкций. Процессоры Core являются решением для ноутбуков, одно- и двухъядерное, исполняющее 32-битный код. Процессоры Core 2 выпускаются как в настольном, так и мобильном исполнении, включают ряд микроархитектурных улучшений и способны исполнять 64-битный код. Количество ядер варьируется от одного до четырёх.

Core i7/Core i5/Core i3 (2009 г.) — дальнейшее развитие идей, заложенных в процессорах Core 2. Сохранив основную конструкцию процессорных ядер, появившийся первым Core i7 получил модульную структуру, позволяющую легко варьировать их количество, встроенный контроллер памяти и новую шину, соединяющую процессор с чипсетом. Микроархитектурные улучшения позволяют Core i7 показывать повышенную производительность в сравнении с Core 2 на равных частотах. Большое внимание было уделено вопросу энергоэффективности нового процессора. Позже появились более дешевые Core i5/i7 с двухканальным контроллером памяти и четырьмя ядрами, затем был разработан Core i3/i5 с двумя ядрами и встроенным видеоядром. В секторе наиболее производительных решений выпускаются также процессоры Core i7 с трехканальным контроллером памяти и шестью ядрами. Благодаря использованию технологии Hyper-threading эти процессоры способны одновременно исполнять до 12 потоков команд. Данные процессоры изготавливаются по технологиям 65–45 нм.

Atom (2011 г.) — недорогие сверхэкономичные одно- и двухядерные процессоры, предназначенные для использования в так называемых интернет-компьютерах — нетбуках и неттопах (компьютерах, в которых вычислительная мощность пожертвована в пользу экономичности, бесшумности и малогабаритности). В основе — модифицированное ядро от первых Pentium, которое адаптировали под новый техпроцесс, добавили возможность исполнения 64-битного кода и мультимедийных инструкций, а также кэш-память второго уровня и поддержку многопоточного. Для упрощения конструкции было решено отказаться от внеочередного исполнения команд, что не лучшим образом сказалось на производительности. С 2013 года разрабатываются по технологии 22 нм.

Xeon — семейство процессоров, ориентированных на серверы и многопоточные вычисления. Первый представитель этого семейства базировался на архитектуре Pentium II, представлял собой картридж с печатной платой, на которой монтировались ядро, кэш-память второго уровня и тег кэша. Современные Xeon базируются на архитектуре Core 2/Core i7. Данные процессоры изготавливались по технологиям 250–32 нм. В 2013 году Intel представила Xeon-процессоры, основанные на микроархитектуре Ivy Bridge (кодовое название 22-нм версии микроархитектуры Sandy Bridge, поддерживающей видеоускоритель).

Параллельно с семейством микропроцессоров фирмы Intel выпускались и x86-совместимые процессоры сторонних фирм. Наиболее выделяется семейство ***процессоров AMD*** (Advanced Micro Devices, дословный перевод «Передовые микроустройства» [11]). Advanced Micro Devices, Inc. — компания, являющаяся вторым по величине производителем x86- и x64-совместимых процессоров, а также одним из крупнейших поставщиков графических процессоров, чипсетов (наборов микросхем, спроектированных для совместной работы с целью выполнения каких-либо функций) для материнских плат и флеш-памяти.

Ранние процессоры AMD обычно выпускались с максимальной частотой чуть выше, чем у оригиналов (микропроцессоров Intel). Впоследствии характеристики

процессоров AMD стали приобретать кардинальные отличия от процессоров Intel. Приведем некоторые из них:

- интегрированная кэш-память второго уровня, работающая на полной частоте ядра (в ранних Pentium III кэш-память работала на половине частоты ядра), а кэш-память, установленная на материнской плате, рассматривается как кэш-память третьего уровня;
- три конвейера для целочисленных вычислений и три конвейера для операций с плавающей точкой;
- новые команды в блок 3DNow! (дополнительное расширение MMX для процессоров AMD, начиная с AMD K6 3D).

В 2003 году компания AMD выпустила первые 64-битные процессоры, которые были совместимы с процессорами x86: процессоры ***Opteron***, предназначавшиеся для серверов и рабочих станций, и процессоры ***Athlon 64*** для персональных компьютеров.

Компания AMD была не единственной, кто создал архитектуру 64-разрядного процессора. Специалисты Intel сконструировали 64-разрядный процессор и назвали его ***IA64***. Первый современный 64-разрядный релиз ОС Windows был создан для запуска на процессорах, реализующих архитектуру IA64. Единственными видами процессоров, которые реализуют IA64, являются ***Itanium*** и ***Itanium 2*** от Intel. На сегодня только платформы ОС Windows Server поддерживают IA64. Хотя изначально ОС Windows XP поддерживала архитектуру IA64, проблемы с поддержкой унаследованных приложений на системах с IA64 привели к тому, что настольные системы на этой базе были весьма редки. Компания Microsoft прекратила поддерживать ОС Windows XP на IA64 в 2005 году.

Когда Intel решила не поддерживать совместимость с процессорами x86 в архитектуре IA64, AMD начала работу над новым дизайном 64-разрядного процессора, который бы расширял возможности x86 старой версии. AMD изначально описывала в спецификации такие процессоры, как x86–64, затем они были переименованы в ***AMD64***. 64-разрядные версии ОС Windows от Microsoft, построенные для запуска на этой архитектуре, также использовали название AMD64. В 2003 году была выпущена ОС Windows XP для AMD64, и каждая последующая версия ОС Windows поддерживала архитектуру AMD64.

Фирмы VIA Technologies (тайваньская компания, производитель электронных схем, чипсетов для материнских плат, микропроцессоров и микросхем памяти) и Intel продают процессоры, которые используют архитектуру AMD64. В маркетинговых целях Intel называет эту технологию Intel 64, скрывая, что ее процессоры реализуют инструкции набора AMD64. Microsoft тоже начала использовать имя x64 для такой архитектуры, чтобы не возникало сомнений относительно архитектуры ОС Windows. Каким бы ни было маркетинговое имя, процессоры Intel 64 используют те же сборки ОС Windows, что и другие процессоры, реализующие архитектуру AMD64. В терминологии ОС Windows можно оперировать названиями AMD64, Intel 64, x86–64, x64, EM64T, как грубыми эквивалентами.

1.5.4 Режимы работы микропроцессоров семейства x86–64

Все 32-разрядные и более поздние процессоры Intel, а также совместимые с ними могут выполнять программы в нескольких режимах. Режимы процессора предназначены для выполнения программ в различных средах. В зависимости от режима процессора изменяется схема управления памятью системы и задачами.

Режимы работы 32-разрядных и 64-разрядных процессоров несколько различны. У 32-разрядных микропроцессоров выделяют несколько режимов работы:

- ***Реальный режим работы*** — данный режим предназначен для совместимости с младшими моделями процессоров (16-разрядными микропроцессорами). Также этот режим первым начинает работу при включении компьютера, в нем выполняется процедура самотестирования оборудования *POST (Power-On Self-Test)*. Данная функция выполняется программами, хранящимися в BIOS (Basic Input/Output System — «базовая система ввода-вывода») материнской платы компьютера.
- ***Защищенный режим*** — основной режим работы процессоров. Именно в нем доступны все особенности 32-разрядных моделей процессоров, такие, как многозадачность, защита программ пользователей, возможность работы с большим объемом памяти, виртуальная память и т. п. Механизм сегментации позволяет поддерживать виртуальную память объемом до 64 Тбайта. Но, как правило, используется только страничная трансляция, при которой каждой задаче предоставляется только 4 Гбайта.
- ***Режим системного управления (SMM — System Management Mode)***. В этом режиме приостанавливается исполнение другого кода, включая код ОС, и запускается специальная программа, хранящаяся в оперативной памяти системы в наиболее привилегированном режиме. Возможностей применения SMM много, приведем некоторые из них: обработка системных ошибок, таких как ошибки памяти и чипсета; выполнение функций защиты, например выключение процессоров при сильном перегреве; управление питанием, например схемами изменения напряжения и т. п.
- ***Режим Virtual-86***. Этот режим схож с реальным режимом, однако может быть включен только в защищенном режиме. В этом режиме возможно выполнение нескольких приложений реального режима.
- ***«Нереальный» режим*** — это неофициальный режим, который поддерживают все 32-битные микропроцессоры. Он поддерживает адресацию к 4 Гбайтам памяти. В этом режиме команды исполняются так же, как и в реальном режиме, с использованием дополнительных сегментных регистров [12].

В 64-разрядных микропроцессорах архитектуры AMD64 введен дополнительный *режим Long Mode* («расширенный режим»), включающий два подрежима:

- ***64-разрядный режим*** (***64-bit Mode***) позволяет 64-разрядной ОС выполнять 64-разрядное ПО. В этом режиме поддерживается только «плоская» модель памяти (один общий сегмент для кода и данных);
- ***режим совместимости*** (***Compatibility Mode***) позволяет 64-разрядной ОС выполнять 32-разрядное ПО.

Контрольные вопросы по главе 1

1. Дайте определение термину «система». Какие характеристики используют для описания систем?
2. Что понимают под определением «вычислительная система»?
3. Коротко расскажите историю развития вычислительной техники.
4. Опишите электронные вычислительные машины и приведите их классификацию.
5. Сравните определения архитектуры вычислительной системы и архитектуры электронной вычислительной машины.
6. В чем отличие принстонской архитектуры ЭВМ от гарвардской архитектуры ЭВМ?
7. Расскажите, какие свойства ЭВМ относятся к общим, а какие — к индивидуальным свойствам?
8. Какие свойства имеет CISC-архитектура процессора?
9. Какие свойства имеет RISC-архитектура процессора?
10. Опишите основные «вехи» развития микропроцессоров семейства x86–64.
11. Какие режимы работы имеют микропроцессоры семейства x86–64 и что они собой представляют?

Глава 2

ОРГАНИЗАЦИЯ ПАМЯТИ

2.1 Единицы измерения информации и их представление в ЭВМ

Единицы измерения информации. В повседневной деятельности мы привыкли к счету с применением десяти цифр: 1, 2, 3 и т. д. Вследствие чисто технических особенностей компьютер, как правило, применяет другой метод счета, в котором задействованы только две цифры: 0 и 1. Например, до пяти он считает следующим образом: 1, 10, 11, 100, 101. Данные числа являются двоичными. Таким образом, двоичное число 10 соответствует десятичному 2. Для обозначения двоичного числа к ним принято приписывать символ «b», например 1010b.

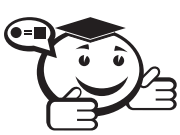
Увеличивая число элементов, можно увеличивать число значений до необходимой величины. Таким образом, мы получили возможность хранить числа в любом диапазоне.

Элемент памяти, содержащий один двоичный разряд, называется ***битом***.

Общее число значений зависит от числа разрядов по следующему закону:

$$\text{Число значений} = 2^{\text{число разрядов}}.$$

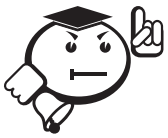
Один бит является слишком маленькой единицей информации. Он содержит только два значения: 0 или 1, что слишком мало для использования в одной операции. Обычно биты группируют по восемь.



Единица информации, состоящая из восьми битов, называется ***байтом***.

Для большего значения используются следующие единицы информации:

- *слово* — 2 байта или 16 бит, максимальное значение 65 535 ($2^{16} - 1$);
- *двойное слово* — 4 байта или 32 бита, максимальное значение $2^{32} - 1$;
- *учетверенное слово* — 8 байт или 64 бита, максимальное значение $2^{64} - 1$;
- *параграф* — шестнадцать байт, максимальное значение $2^{128} - 1$.



Двоичные числа обладают существенным недостатком — выглядят они длинно и громоздко. Для этого было введены *шестнадцатеричные числа*. Они предназначены для более компактного записывания двоичных чисел.

В технической документации, электрических схемах и текстах программ могут применяться разные способы представления чисел:

- *двоичные (binary) числа* — каждая цифра означает значение одного бита (0 или 1), старший бит всегда пишется слева, после числа ставится буква «b». Для удобства восприятия тетрады могут быть разделены пробелами. Например, 1010 0101b;
- *шестнадцатеричные (hexadecimal) числа* — каждая тетрада представляется одним символом 0,…, 9, A, B,…, F. Обозначаться такое представление может по-разному. В данном пособии используется символ «h» после последней шестнадцатеричной цифры (например, A5h). В текстах программ это же число может обозначаться и как 0xA5, и как 0A5h, в зависимости от синтаксиса языка программирования. Незначащий ноль (0) добавляется слева от старшей шестнадцатеричной цифры, изображаемой буквой, чтобы различать числа и символические имена;
- *десятичные (decimal) числа* — каждый байт (слово, двойное слово) представляется обычным числом, а признак десятичного представления (букву «d») обычно опускают. Байт из предыдущих примеров имеет десятичное значение 165. В отличие от двоичной и шестнадцатеричной форм записи по десятичной трудно в уме определить значение каждого бита, что иногда приходится делать;
- *восьмеричные (octal) числа* — каждая тройка бит (разделение начинается с младшего) записывается в виде цифры 0–7, в конце ставится признак «o». То же самое число будет записано как 245o. Восьмеричная система неудобна тем, что байт нельзя разделить поровну, но зато все цифры — привычные.

В таблице 2.1 приведены представления одной тетрады (4 бит) в различных системах исчисления.

В таблице 2.2 приведем *примеры работы шестнадцатеричной арифметики*, в которой единица информации представлена двумя байтами — словом.

С первой и четвертой строкой все ясно, шестнадцатеричная арифметика ведет себя, так же как и десятичная. Если внимательно рассмотреть таблицу 2.1, то приведенные в третьей строке примеры тоже становятся понятны, десятичное число «14» заменяется шестнадцатеричным «Eh». А вот вторая и пятая строки на первый взгляд ведут себя странно. Шестнадцатеричное число FFFF соответствует едини-

це со знаком минус «–1». Пример в стоке 6 показывает, что при сложении чисел «F» и «3» происходит перенос «1» в следующий разряд.

Таблица 2.1 – Представление двоичных чисел в разных системах счисления

Двоичное	Шестнадцатеричное	Десятичное	Восьмеричное
0000	0	0	0
0001	1	1	1
0010	2	2	2
0011	3	3	3
0100	4	4	4
0101	5	5	5
0110	6	6	6
0111	7	7	7
1000	8	8	10
1001	9	9	11
1010	A	10	12
1011	B	11	13
1100	C	12	14
1101	D	13	15
1110	E	14	16
1111	F	15	17

Таблица 2.2 – Примеры работы шестнадцатеричной арифметики

№	Десятичная арифметика	Шестнадцатеричная арифметика[1]
1	2 + 3 = 5	2h + 3h = 5h
2	2 – 3 = –1	2h – 3h = FFFFh
3	9 + 5 = 14	9h + 5h = Eh
4	9 – 5 = 4	9h – 5h = 4h
5	–1 + 5 = 4 65 535 + 5 = 4	FFFFh + 5h = 4h
6	15 + 3 = 18	Fh + 3 = 12h

Пример

Теперь посмотрим, что происходит, когда мы складываем «5h» и «FFFFh»:

$$\begin{array}{r} 0005h \\ +\ FFFFh \\ \hline 10004h \end{array}$$

[1] Для того чтобы отличать шестнадцатеричные числа от других чисел, к шестнадцатеричному числу принято приписывать символ «h».

Происходит последовательный перенос единицы в крайнюю левую позицию. Если мы игнорируем эту единицу, то получаем правильный ответ, а именно «4». Данная ситуация называется переполнением, так как теперь число пятизначное, а в слове хранится четыре цифры.

Допустим FFFFh равным 65 535. Это положительное число, и оно максимальное из тех, что возможно записать при помощи шестнадцатеричных цифр. Число FFFFh — это число без знака («unsigned»). Если принимаем, что FFFFh — знаковое число, то оно будет соответствовать «–1». Фактически, все числа от FFFFh до 8000h ведут прекрасно себя как отрицательные числа. Для чисел без знака переполнение не является ошибкой, а для чисел со знаком переполнение — ошибка.

Рассмотрим *перевод шестнадцатеричных чисел в десятичные числа*.

Пример

$$\begin{array}{rr} & E \longrightarrow \quad 14 \times 16 = 224 \\ + & 6 \longrightarrow \quad\ \ 6 \times 1 = \ \ 6 \\ \hline & E6h \qquad\qquad\ = 230 \end{array}$$

Возьмем число «E6h», «E» — это шестнадцатеричная цифра 14, и там 16 таких цифр (по аналогии с 10 для десятичного числа). Таким образом, «E6h» — это четырнадцать раз по шестнадцать и шесть единиц. Для отделения разрядов шестнадцатеричного числа можно применять следующие четыре числа.

Пример

$$16^3 = 4096$$
$$16^2 = 256$$
$$16^1 = 16$$
$$16^0 = 1$$

Теперь научимся *переводить десятичные числа в шестнадцатеричную форму*.

Пример

$$\begin{array}{r} 1069/16 = 66\ (1069 - 66 \times 16 = 13) \\ 66/16 = 4\ (66 - 4 \times 16 = 2) \\ 4/16 = 0\ (4 - 0 \times 16 = 4) \\ \hline 1069 = 42Dh \end{array}$$

Для этого 1069 разделим на 16, остаток от деления будет давать нам первую шестнадцатеричную цифру, далее целую часть от деления разделим еще раз на шестнадцать, на этот раз остаток даст нам вторую цифру и т. д. Итоговое число собирается последовательно от последнего остатка отделения к первому.

В двухбайтном слове принят *LH-порядок следования байт*: адрес слова указывает на младший байт L (Low), а старший байт H (High) размещается по адресу, на единицу большему. В двойном слове порядок будет аналогичным — адрес укажет на самый младший байт, после которого будут размещены следующие по старшинству. Этот порядок, естественный для процессоров Intel, применяется не во всех микропроцессорных семействах. Байт (8 бит) делится на пару тетрад (nible): старшую тетраду — биты (7:4) и младшую тетраду — биты (3:0).

2.2 Иерархия памяти

Память компьютера предназначена для кратковременного и долговременного хранения информации — кодов команд и данных. В памяти информация хранится в массиве ячеек. Минимальной адресуемой единицей является байт — каждый байт памяти имеет свой уникальный адрес. Память можно рассматривать как иерархическую систему (рис. 2.1). Входящие в ее состав запоминающие устройства различаются емкостью, быстродействием и скоростью.

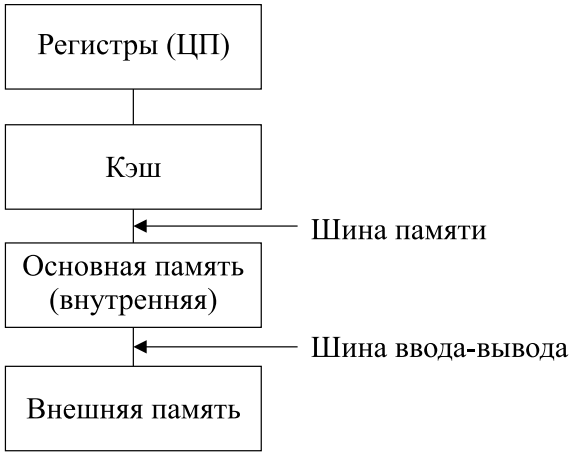


Рис. 2.1 – Иерархия памяти

Самыми быстродействующими являются *регистры* центрального процессора (ЦП), и они же имеют наименьшую емкость. Обращение к ним происходит при выполнении почти каждой машинной команды.

Кэш-память процессора в целом отличается меньшим объемом, но более высоким быстродействием по сравнению с основной памятью. Она функционирует как буфер между процессором и основной памятью и содержит копии областей памяти, которые использовались последними, — их адреса и расположенные по этим адресам данные. Каждый раз, когда процессору требуется обратиться к памяти по заданному адресу, он сначала обращается к кэшу. Если происходит промах кэша, то есть этот адрес в кэше отсутствует, производится обращение к основной памяти. Данные считываются из памяти в указанный в команде регистр процессора,

а их копия записывается в кэш. Для большей части программного обеспечения характерна временная локализация доступа, то есть значительная вероятность повторного обращения по одним тем же адресам в течение короткого промежутка времени. Если повторное обращение произойдет достаточно скоро, нужные данные все еще будут в кэше и их не придется считывать из основной памяти. Такая ситуация называется «попаданием в кэш».

Передача информации между основной памятью и кэш-памятью осуществляется по шине памяти, которая состоит из шины данных и шины адреса.

Со времени появления больших по размерам компьютеров сложилось деление памяти на внутреннюю и внешнюю. Под внутренней подразумевалась память, расположенная внутри процессорного «шкафа» (или плотно к нему примыкающая). Внешняя память представляла собой отдельные устройства с подвижными носителями — накопителями на магнитных дисках (а ранее, барабанах) и ленте. Со временем все устройства компьютера удалось поместить в один небольшой корпус, и прежнюю классификацию памяти применительно к персональным компьютерам можно переформулировать следующим образом [8].

Внутренняя память *— электронная (полупроводниковая) память, устанавливаемая на системной плате или на платах расширения.*

Внешняя память *— память, реализованная в виде устройств с различными принципами хранения информации, чаще всего с подвижными носителями. В настоящее время сюда входят устройства магнитной (дисковой и ленточной) памяти, оптической и магнитооптической памяти; устройства внешней памяти могут размещаться как в системном блоке компьютера, так и в отдельных корпусах, достигающих иногда размеров небольшого шкафа.*

Информация из внешней памяти во внутреннюю память попадает по шине ввода-вывода.

Для процессора непосредственно доступной является внутренняя память, доступ к которой осуществляется по адресу, заданному программой. Для внутренней памяти характерен одномерный (линейный) адрес, который представляет собой одно двоичное число определенной разрядности. Внутренняя память подразделяется на *оперативную* (ОЗУ — оперативно запоминающее устройство), информация в которой может изменяться процессором в любой момент времени, и *постоянную* (ПЗУ — постоянно запоминающее устройство), информацию, которую процессор может только считывать. Обращение к ячейкам оперативной памяти может происходить в любом порядке как по чтению, так и по записи, поэтому оперативную память называют памятью с произвольным доступом RAM (Random Access Memory) — в отличие от постоянной памяти ROM (Read Only Memory).

2.3 Адресация и распределение памяти в реальном режиме работы микропроцессора Intel x86

Адресация ячеек памяти в реальном режиме. В реальном режиме адрес имеет размер 20 бит. Максимальный объем адресуемой памяти составляет 1 Мбайт (2^{20} = 1 Мбайт). Для формирования 20-битового адреса в памяти используются два 16-битовых регистра: сегментный регистр и регистр смещения (рис. 2.2).

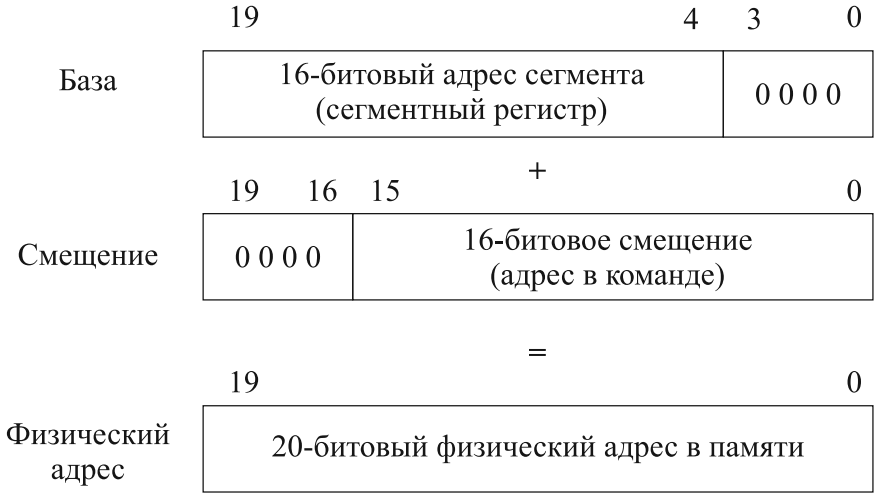


Рис. 2.2 – Преобразование логического адреса в физический адрес в реальном режиме

Сегментный регистр определяет базовый адрес сегмента. В нем содержатся старшие 16 бит базового адреса сегмента, младшие четыре бита предполагаются равными 0. В регистре смещения находится смещение внутри сегмента (относительно базового адреса сегмента). Смещение может также задаваться непосредственной константой. Смещение добавляется к базовому адресу сегмента, и, таким образом, получается 20-битовый адрес в памяти.

Более просто можно описать вычисление следующим образом — 16-битовое значение сегментного регистра сдвигается на 4 бита влево и к нему добавляется значение смещения, в результате получается 20-битовое значение. Или это же действие можно представить математической формулой:

Физический адрес = ***Сегментный адрес*** · ***10h*** + ***Смещение***.

Такая схема применялась в 8086 и 8088 микропроцессорах, и такое же представление 20-битного адреса двумя 16-битными числами поддерживается в реальном режиме всех последующих микропроцессоров x86.

С использованием реального режима функционировала ОС *MS DOS (Microsoft Disk Operating System)* — дисковая операционная система от фирмы Microsoft.

Распределение оперативной памяти в MS DOS. Согласно сегментной модели физический адрес вычисляется по формуле, приведенной на рисунке 2.2. Таким образом, обеспечивался доступ к адресному пространству от 00000h до FFFFFh при помощи пары 16-битных регистров.

Обратите внимание, что при сегментном адресе, равном 0FFFFh, и смещении, равном 0FFFFh, данная формула дает адрес 10FFEFh, но ввиду 20-битного

ограничения на шину адреса эта комбинация в физической памяти указывает на 0FFEFh. Таким образом, адресное пространство как бы сворачивается в кольцо с небольшим «нахлестом». Начиная с процессора 80286, шина адреса была расширена до 24 бит, а впоследствии (в процессорах 386DX, 486 и выше) до 32 бит и даже 36 бит. В реальном режиме процессора, используемом в MS DOS, применяется та же сегментная модель памяти, и формально был доступен лишь 1 Мбайт памяти. Однако выяснилось, что процессоры 80286 в реальном режиме эмулируют 8086 с ошибкой: та самая единица в бите A20, которая отбрасывалась в процессорах 8086/88, теперь попадает на шину адреса, и в результате максимально доступный линейный адрес в реальном режиме достиг 10FFEFh. Дополнительные байты оперативной памяти (64 Кбайт минус 16 байт), адресуемой в реальном режиме, позволили освободить дефицитное пространство оперативной памяти для прикладных программ. В эту область от 100 000h до 10FFEFh, названную высокой памятью *HMA (High Memory Area)*, стали помещать часть операционной системы и небольшие резидентные программы.

Распределение оперативной памяти, непосредственно адресуемой микропроцессором в реальном режиме, представлено на рисунке 2.3 [12].

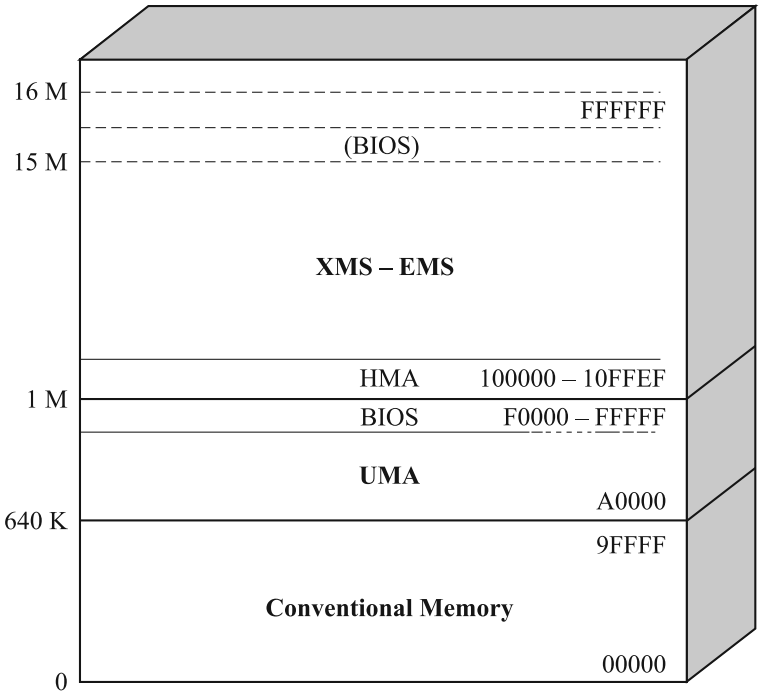


Рис. 2.3 – Распределение памяти в MS DOS

Стандартная память – Conventional memory. При работе в среде ОС типа MS DOS стандартная память является самой дефицитной в персональном компьютере. На ее небольшой объем (типовое значение 640 Кбайт) претендуют и BIOS, и ОС, а оставшаяся часть предназначена для прикладного ПО. Стандартная память распределяется следующим образом:

- 00000h–003FFh — Interrupt Vectors — таблица векторов прерываний[1] (256 двойных слов);
- 00400h–004FFh — BIOS Data Area — область переменных BIOS;
- 00500h–00xxxh — DOS Area — область DOS;
- 00xxxh–9FFFFh — User RAM — память, предоставляемая пользователю.

В процессорах семейства Intel предусмотрено 256 прерываний и соответственно 256 векторов прерываний. Все вектора прерываний объединяются в таблицу, состоящую из 256 4-байтовых элементов и занимающую 1 Кбайт. В реальном режиме эта таблица находится в самом начале памяти по адресу 0000:0000 или просто по 0-му физическому адресу.

Каждый элемент таблицы состоит из двух полей. Первые два байта представляют собой значение, заносимое в регистр, в котором хранится адрес смещения, последние два байта содержат значение, заносимое в сегментный регистр.

Управление пользовательской памятью осуществляется с использованием специализированных структур[2]: таблицы таблиц — list of list (табл. 2.3) и управляющих блоков памяти — memory control block (MCB) (табл. 2.4).

Таблица 2.3 – Структура таблицы таблиц

Смещение	Длина	Содержимое
−2	2	Сегментный адрес первого блока управления памятью MCB
0	4	Указатель на первый DPB (disk parameter blockout)
+4	4	Указатель на список таблиц открытых файлов
+8	4	Указатель на первый драйвер DOS (CLOCK$)
. . .	. . .	. . .

Таблица 2.4 – Структура блока управления памятью

Смещение	Длина	Содержимое
+0	1	Тип блока: 'M' (4dH) — за этим блоком есть еще блоки. 'Z' (5aH) — данный блок является последним (свободная память)
+1	2	Сегментный адрес префикса программного сегмента (PSP — program segment prefix)
+3	2	Размер блока в параграфах (1 параграф = 16 байт = 10h байт)
+5	0Bh	Зарезервировано
+10h	?	Начало блока памяти длинной: (размер блока в параграфах · 10h) байт

[1] Прерывания представляют собой механизм, позволяющий координировать параллельное функционирование отдельных устройств вычислительной системы и реагировать на особые состояния, возникающие при работе процессора [13].

[2] Данные структуры являются недокументированными.

Замечания:

- блоки памяти всегда выровнены на границу параграфа («сегмент блока»);
- блоки M-типа: следующий блок находится по адресу (Сегментный адрес блока + Размер + 1):0000;
- блоки Z-типа: (Сегментный адрес блока + Размер + 1):0000 = конец памяти (a000H = 640K).
- +1 при вычислении адреса следующего блока — это шапка блока, равная 16 байтам или 10h (см. табл. 2.3).

В любом MCB указан его владелец — сегментный адрес префикса программного сегмента — PSP программы владельца данного блока памяти. А в PSP есть ссылка на окружение данной программы, в котором можно найти имя программы — путь ее запуска.

Следует помнить, что сама программа (и PSP в том числе) и ее окружение сами располагаются в блоках памяти. Поэтому в MCB блока памяти самой программы в качестве хозяина указан собственный адрес самого себя.

Верхняя память — UMA (Upper Memory Area). Верхняя память имеет области различного назначения, которые могут быть заполнены буферной памятью адаптеров, постоянной памятью или оставаться незаполненными. Первоначально эти пространства не использовали из-за сложности адресации. С появлением механизма страничной переадресации их стали по возможности заполнять «островками» оперативной памяти, названными блоками верхней памяти UMB (Upper Memory Block). Эти области доступны DOS для размещения резидентных программ[1] и драйверов через драйвер EMM386, который отображает в них доступную дополнительную память.

XMS – EMS память. Дополнительная память за пределами первого мегабайта адресного пространства IBM PC-совместимого компьютера с процессором Intel 80286 или более поздним.

XMS (Extended Memory Specification) — спецификация дополнительной памяти, предполагает использование дополнительной памяти в реальном режиме только для хранения данных. Память становится доступной благодаря использованию менеджера дополнительной памяти XMM (eXtended Memory Manager) — HIMEM.SYS.

EMS (Expanded Memory Specification) — спецификация расширенной памяти, предоставляющая доступ DOS-приложениям к памяти, недоступной через адресное пространство основной памяти. Расширенная память адресуется постранично через «окно», находящееся в области памяти UMA. С конца 1980-х до середины 1990-х годов EMS активно использовалась в приложениях, однако с появлением спецификации дополнительной памяти (XMS) стала использоваться реже.

[1] В операционной системе MS DOS резидентная программа возвращает управление оболочке операционной системы (command.com) либо надстройке над операционной системой (Norton Commander и т. п.), но остается в оперативной памяти персонального компьютера. Резидентная программа активизируется каждый раз при возникновении прерывания, вектор которого эта программа изменила на адрес одной из своих процедур.

2.4 Адресация и распределение памяти в защищенном режиме работы микропроцессора Intel x86

Управление памятью в архитектуре Intel делится на две части: сегментацию и трансляцию страниц.

Сегментация *предоставляет механизм для изолирования индивидуального кода, данных и стека (специальная область памяти, предназначенная для временного хранения данных и организованная по принципу «первым вошел, последним вышел»).*

Таким образом, несколько программ могут выполняться одновременно на одном процессоре, не перекрывая друг друга.

Трансляция страниц *предоставляет механизм для реализации виртуальной памяти с подкачкой страниц по запросу, где части программы отображаются, как необходимо, на физическую память.*

Трансляция страниц также может использоваться для изоляции между несколькими задачами.

Все сегменты содержатся внутри линейного адресного пространства процессора. Для доступа к любому байту информации в каком-либо сегменте необходим логический адрес (иногда его называют дальним указателем). Логический адрес состоит из селектора сегмента и смещения. Селектор сегмента — это уникальный идентификатор сегмента. Селектор сегмента также является смещением в таблице дескрипторов. Каждый сегмент имеет дескриптор сегмента, который определяет размер, права доступа, уровень привилегий, тип сегмента и расположение его первого байта внутри линейного пространства. Смещение добавляется к базовому адресу сегмента для доступа к конкретному байту внутри сегмента. Базовый адрес плюс смещение формируют линейный адрес внутри линейного адресного пространства процессора [14].

Если трансляция страниц не используется, то линейное адресное пространство непосредственно отображается на физическое адресное пространство процессора. Физическое адресное пространство определяется как диапазон адресов, которые процессор может генерировать на шине адреса.

В многозадачных операционных системах линейное пространство гораздо больше, чем пространство физической памяти, поэтому необходим метод виртуализации линейного адресного пространства. Эта виртуализация линейного адресного пространства производится через механизм трансляции страниц.

Трансляция страниц поддерживает среду с виртуальной памятью, где большое линейное пространство эмулируется за счет маленького количества физической памяти и некоторого количества дисковой памяти. При этом все сегменты делятся

на 4-килобайтовые страницы, которые могут находиться либо в памяти, либо на диске. Когда задача пытается обратиться к странице, которой нет в памяти, происходит исключение и операционная система загружает недостающую страницу в память, после чего выполнение задачи возобновляется.

Для более экономного использования памяти используется двухуровневая таблица схемы трансляции страниц. На верхнем уровне находится каталог страниц, который содержит ссылки на таблицы страниц. Таблицы страниц содержат ссылки на сами страницы. Трансляция страниц — это единственный способ, при котором возможно выполнение нескольких задач реального режима.

Упрощенно схема по формированию физического адреса памяти на 32-разрядных процессорах в защищенном режиме представлена на рисунке 2.4.

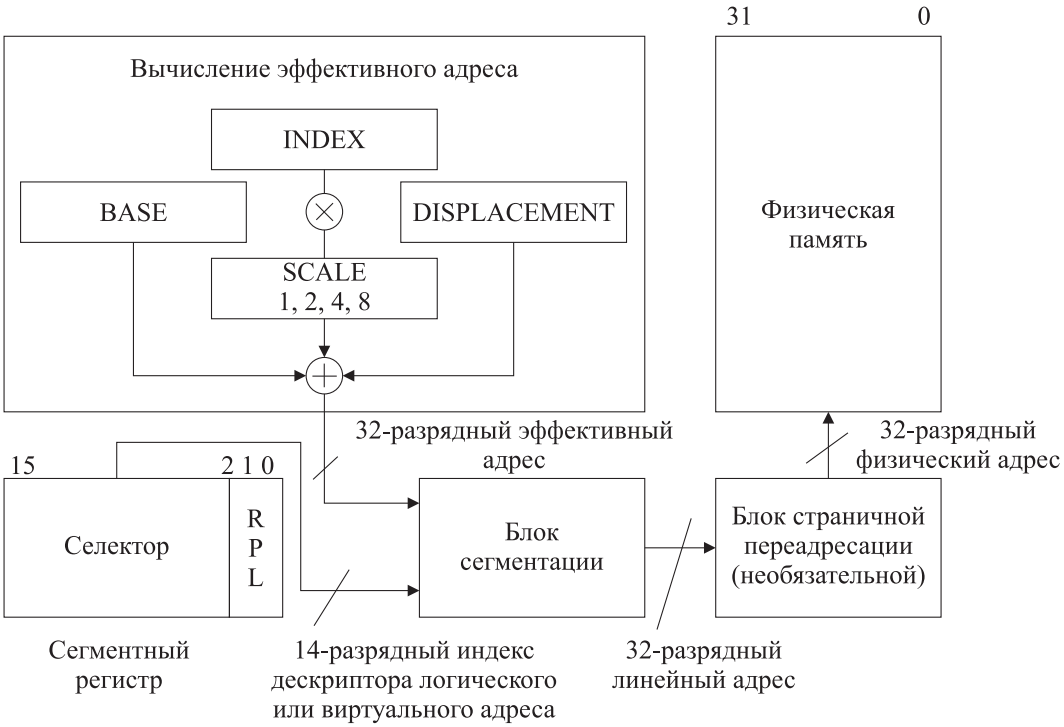


Рис. 2.4 – Формирование адреса памяти 32-разрядных процессоров в защищенном режиме

В сегментном регистре в старших 14 битах находится селектор сегмента. Младший бит селектора (2 бит сегментного регистра при нумерации от нуля) указывает, какая таблица дескрипторов должна использоваться. Нуль соответствует глобальной таблице дескрипторов, единица — локальной таблице дескрипторов. Поле RPL (0 и 1 бит сегментного регистра) используется для контроля прав доступа программы к сегменту.

Эффективный адрес формируется суммированием компонентов base (базы), index (индекса), displacement (смещения) с учетом scale (масштаба). Поскольку каждая задача может иметь до 16 Кбайт селекторов (2^{14}), а смещение, ограниченное размером сегмента, может достигать 4 Гбайт, логическое адресное пространство для каждой задачи может достигать 64 Тбайт. Все это пространство виртуальной памяти в принципе доступно программисту при условии поддержки со стороны операционной системы.

Блок сегментации транслирует логическое адресное пространство в 32-битное пространство линейных адресов.

32-битный физический адрес памяти образуется после преобразования линейного адреса блоком страничной переадресации. Он выводится на внешнюю шину адреса процессора. В простейшем случае при отключенном блоке страничной переадресации физический адрес совпадает с линейным. Включенный блок страничной переадресации осуществляет трансляцию линейного адреса в физический страницами размером 4 Кбайт (для последних поколений процессоров возможны страницы размером 2–4 Мбайт). Блок обеспечивает расширение разрядности физического адреса процессоров шестого поколения до 36 бит. Блок переадресации может включаться только в защищенном режиме.

Для обращения к памяти процессор совместно с внешними схемами формирует шинные сигналы для операций записи и чтения. Шина адреса разрядностью 32/36 бит позволяет адресовать 4/64 Гбайт физической памяти, но в реальном режиме доступен только 1 Мбайт, начинающийся с младших адресов.

Распределение оперативной памяти в ОС Windows. За историю существования семейства ОС Windows выделялись ОС на платформе Windows 9x и на платформе Windows NT. В обеих платформах использовалась «плоская» модель памяти. Однако схема распределения возможного виртуального адресного пространства в системах Windows NT разительно отличается от модели памяти Windows 9x. Прежде всего, в отличие от Windows 9x в Windows NT в гораздо большей степени используется ряд серьезных аппаратных средств защиты, имеющихся в микропроцессорах, а также применено принципиально другое логическое распределение адресного пространства.

В связи с тем, что платформа Windows 9x перестала поддерживаться фирмой Microsoft более 10 лет назад, оставим вопрос рассмотрения только распределения оперативной памяти на платформе Windows NT.

В ОС на платформе Windows NT все системные программные модули находятся в своих собственных виртуальных адресных пространствах и доступ к ним со стороны прикладных программ невозможен. Ядро системы и несколько драйверов работают в нулевом кольце защиты в отдельном адресном пространстве.

Остальные программные модули самой операционной системы, которые выступают как серверные процессы по отношению к прикладным программам (клиентам), функционируют также в своем собственном системном виртуальном адресном пространстве, невидимом для прикладных процессов. Логическое распределение адресных пространств приведено на рисунке 2.5.

Прикладным программам выделяется 2 Гбайта локального (собственного) линейного (неструктурированного) адресного пространства от границы 64 Кбайт до 2 Гбайт.

Нижние 64 Кбайта каждого виртуального адресного пространства в обычном состоянии не отображаются на физическую память. Это делается для облегчения перехвата программных ошибок (выявления недействительных указателей).

Прикладные программы изолированы друг от друга, хотя могут общаться через буфер обмена (clipboard) и механизмы: универсальные механизмы динамического обмена данными DDE (Dynamic Data Exchange); технологию документно-ориентированной архитектуры приложений OLE (Object Linking and Embedding).

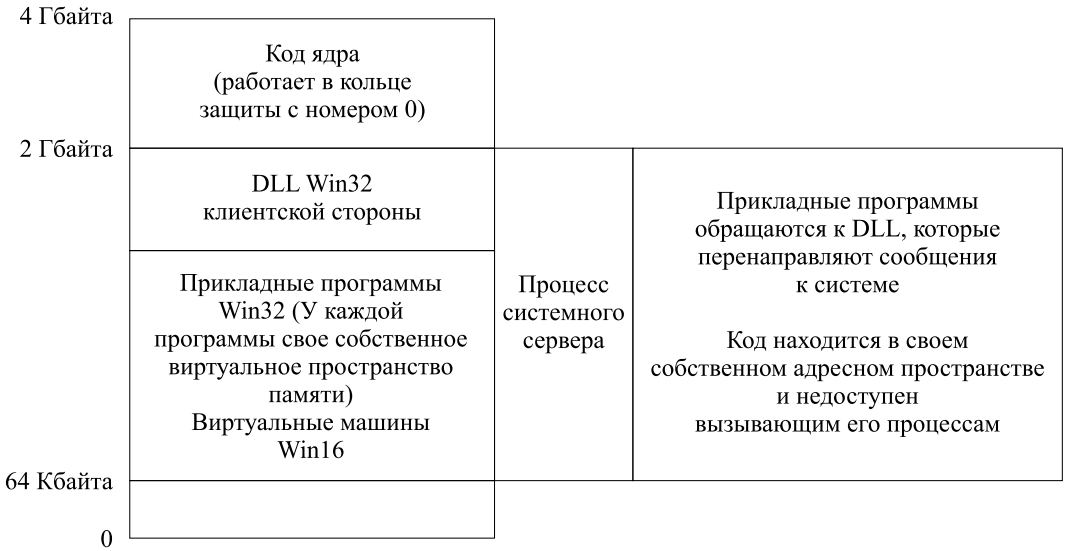


Рис. 2.5 – Модель распределения памяти в ОС Windows NT

В верхней части каждой 2-гигабайтной области прикладной программы размещен код системных DLL (Dynamic Link Library — динамически подключаемая библиотека) кольца 3, который выполняет перенаправление вызовов в совершенно изолированное адресное пространство, где содержится уже собственно системный код, выступающий как сервер-процесс (server process), который проверяет значения параметров, исполняет запрошенную функцию и пересылает результаты назад в адресное пространство прикладной программы. Хотя сервер-процесс сам по себе остается процессом прикладного уровня, он полностью защищен от вызывающей его прикладной программы и изолирован от нее. Между отметками 2 и 4 Гбайта расположены низкоуровневые системные компоненты Windows NT кольца 0, в том числе ядро, планировщик потоков и диспетчер виртуальной памяти. Системные страницы в этой области наделены привилегиями супервизора, которые задаются физическими схемами кольцевой защиты процессора. Это делает низкоуровневый системный код невидимым и недоступным для записи программ прикладного уровня, но приводит к падению производительности во время переходов между кольцами.

Для 16-разрядных прикладных Windows-программ ОС Windows NT реализует сеансы Windows on Windows (WOW). В отличие от Windows 95/98 ОС Windows NT дает возможность выполнять 16-разрядные программы Windows индивидуально в собственных пространствах памяти или совместно в разделяемом адресном пространстве. Почти во всех случаях 16- и 32-разрядные прикладные программы Windows могут свободно взаимодействовать, используя OLE, независимо от того, выполняются они в отдельной или общей памяти. Собственные прикладные программы и сеансы WOW выполняются в режиме вытесняющей многозадачности, основанной на управлении отдельными потоками. Множественные 16-разрядные прикладные программы Windows в одном сеансе WOW выполняются в соответствии с кооперативной моделью многозадачности. Windows NT может также выполнять в многозадачном режиме несколько сеансов DOS.

2.5 Адресация и распределение памяти в архитектуре AMD64

Переход к 64-битной архитектуре был обусловлен необходимостью выделения программе больше 2 Гбайт памяти, то есть преодолеть ограничение 32-битных систем. В 32-битных системах на платформе Windows шина адреса позволяет адресовать 4 Гбайта памяти (2^{32}). Верхние 2 Гбайта адресов зарезервированы для ядра и API-функций. Реально Windows-приложениям доступно всего 2 Гбайта адресов (3 Гбайта в некоторых версиях Windows Advanced Server при указании специальной опции). Потребности же многих приложений (баз данных, программ обработки растровой графики и видео, систем автоматизированного проектирования), могут превышать этот лимит, так как оперируют большими наборами данных.

Процессоры фирмы Intel с Pentium Pro могут иметь и более 4 Гбайт физической памяти (2^{36} = 64 Гбайта), но только через механизм PAE (Physical Address Extension — расширение физического адреса). Но приложения по-прежнему ограничены двумя гигабайтами адресного пространства. Предоставить доступ к большему пространству можно, но лишь через механизм AWE (Address Windowing Extensions — расширение адреса методом окна). Программирование через этот механизм затруднительно, и производительность при случайном доступе к памяти оказывается заметно ниже, чем при использовании плоской модели адресации.

Процессоры *Athlon 64* и *Opteron* не полностью 64-битные. Для указателей в программах используются новые 64-битные регистры, но объем адресуемой памяти в 16 эксабайт (экзабайт)[1] для 64-битной адресации в AMD посчитали излишним. Поэтому процессоры с архитектурой AMD64 используют 48-битную адресацию, и позволяют адресовать до 256 Тбайт виртуальной памяти. Старшие 16 бит 64-битного адреса не используются. Для задания физического адреса в адресной шине отведено 40 бит, что позволяет иметь до 1 Тбайт физической оперативной памяти.

Скорость компьютера сильно зависит от нагрузки, приходящейся на каналы памяти. В типичном коде около половины объема приходится на адреса операндов и связанные с ними константы, следовательно, двукратное увеличение разрядности адреса должно привести к увеличению объема кода примерно в полтора раза. Что, в свою очередь, означало бы пропорциональное увеличение нагрузки на шину данных и снижение эффективности кэша. Иначе говоря, переход Athlon'а в 64-битный режим мог бы сопровождаться заметным падением производительности [15].

Для предотвращения данной проблемы инженеры AMD добавили новый режим адресации относительно 64-битного указателя инструкций — RIP-relative addressing. Эффективный адрес в этом случае получается суммированием 32-битного адреса операнда, который играет роль смещения, и регистра RIP[2]. Адреса, используемые в командах перехода, имеют закон распределения, близкий к нормальному, с вершиной в команде перехода. То есть чем дальше адрес отстоит от текущего значения указателя инструкций, тем меньше вероятность его применения. Это

[1] 1 эксабайт = 2^{60} байт.

[2] 64-разрядный регистр, обозначающий смещение следующей команды относительно кодового сегмента.

позволяет снизить разрядность адресов в командах. Когда 32-х бит все же недостаточно, компилятор формирует полный 64-битный адрес, после чего, по возможности, снова продолжает генерировать 32-битные адреса. Такой подход практически ликвидировал главный недостаток 64-битного кода — большой размер.

Большинство 64-битных операционных систем поддерживают режим совместимости *Compatibility Mode*. Но вызов системной функции, сделанный в 64-битной среде 32-битным приложением, нуждается в преобразовании некоторых аргументов, в основном, указателей адреса. ОС Windows реализует эти преобразования через подсистему «Windows on Windows» (рис. 2.6). Специальная библиотека, wow.dll, динамически подключается ко всем 32-битным приложениям и выполняет следующие функции [15]:

- приводит аргументы к 64-битному виду;
- передает управление (вызов) 64-битному ядру;
- приводит результаты вызова обратно к 32-битному виду;
- возвращает их приложению.

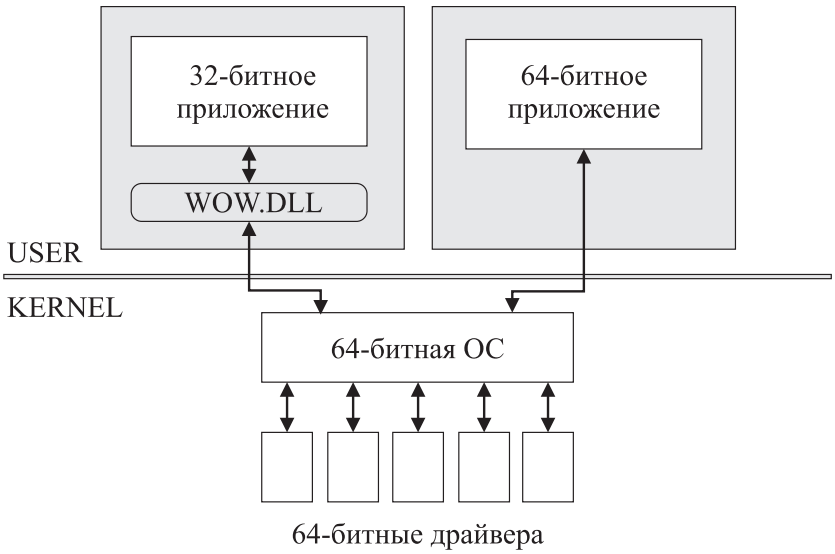


Рис. 2.6 – Взаимодействие приложений с ядром в архитектуре AMD64

Большинство преобразований при вызове системных функций сводятся к добавлению нулей к аргументам функции и к их последующему удалению. Весь остальной код работает без посредников.

Способность выполнять одновременно 32-битные и 64-битные приложения особенно ценна для программистов. При разработке 64-битных приложений они могут пользоваться 32-битным инструментарием на той же машине, на которой отлаживают 64-битный код. Можно оставить 32-битные редакторы/утилиты, которые, возможно, никогда не будут перенесены на новую ОС. 32-разрядная совместимость позволит разрабатывать и проверять код на той же машине, без перезагрузки [15].

В таблице 2.5 сравниваются увеличенные максимальные ресурсы памяти компьютеров под управлением 64-разрядных версий Windows на базе 64-разрядных процессоров Intel с существующими максимальными ресурсами 32-разрядных версий.

Таблица 2.5 – Сравнение распределения памяти
в 64- и 32-разрядных версиях Windows по данным корпорации Microsoft

Компонент архитектуры	64-разрядная версия Windows	32-разрядная версия Windows
Виртуальная память	16 ТБ	4 ГБ
Размер файла подкачки	512 ТБ	16 ТБ
Гиперпространство	8 ГБ	4 МБ
Выгружаемый пул	128 ГБ	470 МБ
Невыгружаемый пул	128 ГБ	256 МБ
Системный кэш	1 ТБ	1 ГБ
PTE системы	128 ГБ	660 МБ

***Виртуальная память**. Технология расширения доступной физической памяти компьютера. В системе виртуальной памяти операционная система создает файл подкачки и делит память на единицы, называемые страницами. Страницы, к которым недавно обращались, хранятся в физической памяти (ОЗУ). Если какое-то время к странице памяти не обращаются, она записывается в файл подкачки (процесс, называемый «подкачкой памяти» или «откачкой страницы»). Если позже к этому участку памяти обращается программа, операционная система считывает страницу памяти из файла подкачки и помещает ее в физическую память (процесс, называемый «подкачкой памяти» или «подкачкой страницы»). Общий объем памяти, доступной программам, равен сумме физической памяти компьютера и размера файла подкачки.*

***Файл подкачки** — файл, хранящийся на диске, который используется компьютером для увеличения объема физического хранилища виртуальной памяти.*

Важный вывод из вышесказанного заключается в том, что даже 32-разрядные приложения будут работать эффективнее благодаря увеличенному виртуальному адресному пространству, если они используются в 64-разрядных версиях Windows.

***Гиперпространство** — специальная область, используемая для управления страницами процессов с применением виртуальной памяти.*

***Выгружаемый пул** — область виртуальной памяти в системном пространстве, которая может выгружаться из рабочего множества системных процессов и загружаться в него.*

***Невыгружаемый пул** — область памяти, состоящая из ячеек виртуальных системных адресов, которые всегда находятся в физи-*

ческой памяти, и поэтому доступ к ним возможен из любого адресного пространства без загрузки/выгрузки при подкачке страниц.

Системный кэш *— страницы памяти, используемые для распределения открытых файлов в системном кэше.*

PTE (Page Table Entry) системы *— пул системных элементов таблицы страниц, используемый для сопоставления системных страниц, таких как загружаемое/выгружаемое пространство, стеки ядра и списки дескрипторов памяти.*

. . . . . .

В 64-разрядных программах используется линейное пространство памяти в 16 терабайт (8 терабайт для пользователя и 8 терабайт для ядра). В 32-разрядных программах все еще используется 4 ГБ (2 ГБ для пользователя и 2 ГБ для ядра).

2.6 Управление памятью в ОС Windows

2.6.1 Получение общей информации об использовании памяти

Диспетчер задач Windows позволяет просматривать общее использование памяти на вкладке Быстродействие[1] (рис. 2.7). Здесь к информации о памяти относятся три раздела:

- Система.
- Физическая память.
- Память ядра.

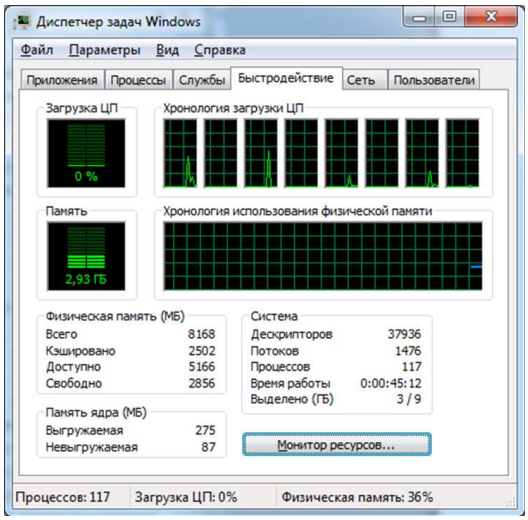


Рис. 2.7 – Вид вкладки «Быстродействие» в Диспетчере задач Windows

В первом разделе содержится пять полей.

1. Дескрипторы — число уникальных идентификаторов объектов, используемых процессами.

[1] В различных версиях операционных систем семейства Windows интерфейс может иметь различия.

2. Потоки — число объектов или процессов, выполняющихся внутри более крупных процессов или программ.
3. Процессы — число отдельных процессов, исполняемых на компьютере.
4. Время работы — время, прошедшее после загрузки компьютера.
5. Выделение (ГБ) — первое число представляет собой объем используемой в данное время оперативной и виртуальной памяти, а второе — объем доступной оперативной и виртуальной памяти.

Во втором разделе содержатся параметры, несущие информацию о текущем состоянии физической памяти машины. Эта статистика не имеет никакого отношения к файлу подкачки, следовательно, может являться хорошим индикатором ситуаций, когда его увеличение не даст эффекта. Параметр «Всего» — это объем памяти, обнаруженный операционной системой на компьютере. Параметр «Доступно» отражает память, доступную для использования процессами. Эта величина не включает в себя память, доступную приложениям за счет файла подкачки. Каждое приложение требует определенного объема физической памяти и не может использовать только ресурсы файла подкачки. Параметр «Кэшировано» сообщает объем, доступный кэш-памяти системы.

В третьем разделе отображается информация о потребностях компонентов операционной системы, обладающих наивысшим приоритетом. Эти компоненты обычно работают с сервисом низкого уровня, типа прямого доступа к жесткому диску. Параметр «Выгружаемая» несет информацию об общем объеме памяти, использованной системой за счет файла подкачки. Параметр «Невыгружаемая» — объем физической памяти, потребляемой операционной системой.

Все приведенные параметры относятся лишь к привилегированным службам, а не ко всему сервису системы в целом. Многие компоненты ОС работают как приложения. В большинстве случаев параметры вкладки «Память ядра» должны оставаться без изменений, если не меняется что-либо в ядре операционной системы (например, устанавливается новое устройство в компьютер).

С помощью Диспетчера задач можно также узнать объемы памяти, используемые процессами. Для этого нужно перейти на вкладку «Процессы», которая показывает список исполняемых процессов и занимаемую ими память, в том числе физическую память, пиковое, максимальное, использование памяти и виртуальную память.

Однако конкретное размещение процесса в виртуальной памяти с помощью Диспетчера задач узнать невозможно, нельзя также увидеть свободные, занятые страницы и блоки памяти, их размер и атрибуты защиты.

Информация, которую способен вывести Диспетчер задач, не является полной. В ряде случаев ее достаточно для оптимизации системы, но есть несколько ограничений, характерных для Диспетчера задач [16].

- Список процессов не полон. В окне Диспетчера задач представлены только процессы, зарегистрированные в Windows. В частности, в этот список не включаются драйверы устройств и некоторые системные службы.
- Требования к памяти отражают текущее состояние процесса. В списке отражены объемы памяти, занимаемые приложениями в текущий момент времени, а не их максимальные значения.

- Отсутствуют статистические данные. Поскольку в Диспетчере задач не выводятся временные характеристики, а только мгновенная картина потребления памяти, нет возможности отследить ее изменение.

Более обширную информацию по сравнению с Диспетчером задач можно получить утилитой TaskList (рис. 2.8).

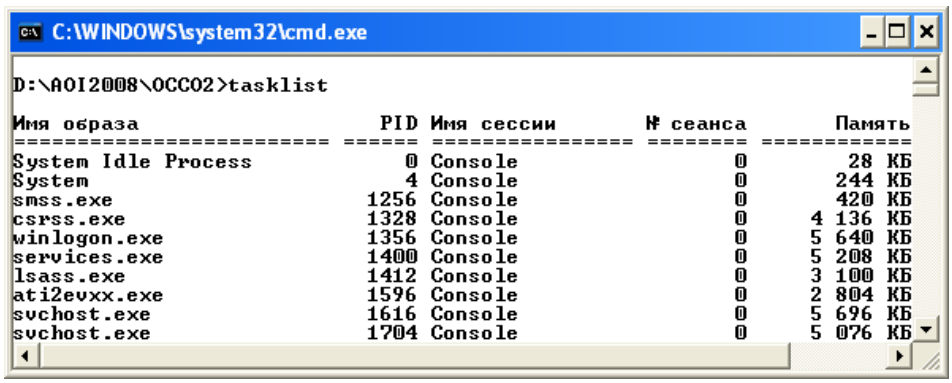


Рис. 2.8 – Запуск утилиты TaskList в окне командной строки

Получить информацию по возможностям утилиты TaskList можно так же, как и по любой команде, используемой в CMD (рис. 2.9).

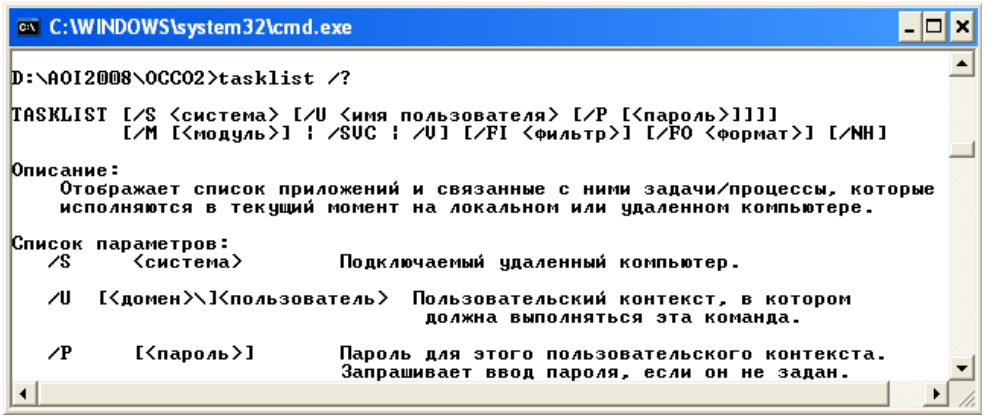


Рис. 2.9 – Вызов помощи по утилите TaskList

В операционной системе Windows через меню *Пуск\Все программы\Стандартные\Служебные* можно вызвать приложение «Сведения о системе», в котором можно получить сведения об основных характеристиках организации памяти в компьютере (рис. 2.10).

В этом приложении можно узнать полный объем физической, виртуальной и свободной в данный момент памяти, размещение и объем файла подкачки и т. п.

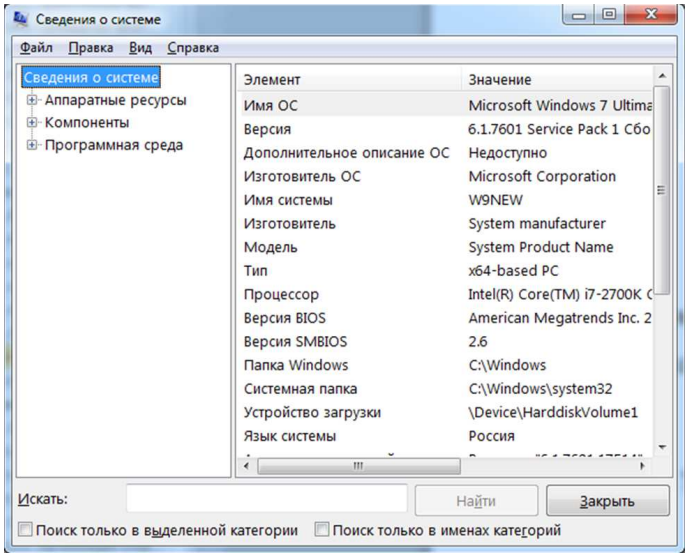


Рис. 2.10 – Окно приложения «Сведения о системе»

2.6.2 Управление файлом подкачки на платформе Microsoft Windows NT

Как уже говорилось ранее, файл подкачки — это область жесткого диска, используемая Windows для хранения данных оперативной памяти. Он создает иллюзию, что система располагает большим объемом оперативной памяти, чем это есть на самом деле. Единой стратегии работы с файлом подкачки не существует. Многое определяется назначением и настройкой компьютера.

По умолчанию Windows удаляет файл подкачки после каждого сеанса работы и создает его в процессе загрузки операционной системы. Размер файла постоянно меняется по мере выполнения приложений и контролируется операционной системой. Обычно используется единственный файл подкачки, расположенный на том же диске, что и операционная система. Такой подход не является лучшим и, более того, практически всегда плох. В этом случае возникает несколько проблем [16].

1. Память может неожиданно оказаться исчерпанной из-за того, что приложение создало на жестком диске файл большого объема или операционная система без предупреждения увеличила потребности файла подкачки. Большой файл подкачки приводит к дефициту дискового пространства и к увеличению непроизводительных затрат на организацию страничного обмена.
2. Файл подкачки фрагментируется, что приводит не только к медленному считыванию жесткого диска, но и к дополнительным перемещениям считывающей головки диска, а в итоге — к существенному снижению производительности.
3. Файл подкачки фрагментируется сам по себе и очень быстро, причем так, что одна и та же область памяти может оказаться в разных местах жесткого диска. В этом случае даже отдельные приложения не могут получить доступ к памяти без нескольких обращений к диску.

4. Производительность системы падает и в том случае, если операционная система установлена не на самом быстром из жестких дисков, имеющихся в компьютере.

Наличие двух жестких дисков может дать значительное преимущество при настройке файла подкачки. Для максимально эффективного использования файла подкачки нужно так его настроить, чтобы он располагался на жестком диске в виде достаточно протяженных фрагментов (это уменьшает количество перемещений считывающей головки, радикально влияющих на производительность). Кроме того, файл подкачки необходимо периодически удалять, чтобы избежать его фрагментации.

Для установки размера файла подкачки нужно выполнить следующую последовательность действий. Щелкнуть правой клавишей мыши по значку «Мой компьютер» и выбрать в контекстном меню строку «Свойства». На экране появится окно «Свойства системы». Перейти на вкладку «Дополнительно» и нажать кнопку «Параметры» в рамке «Быстродействие» (рис. 2.11). В появившемся окне «Параметры быстродействия» на вкладке «Дополнительно» нажать кнопку «Изменить». Предварительно следует выбрать принцип распределения времени процессора (для оптимизации работы программ, если это пользовательский компьютер, или служб, работающих в фоновом режиме, если это сервер). Кроме того, следует задать режим использования памяти. Для пользовательского компьютера — оптимизировать работу программ, для сервера — системного кэша.

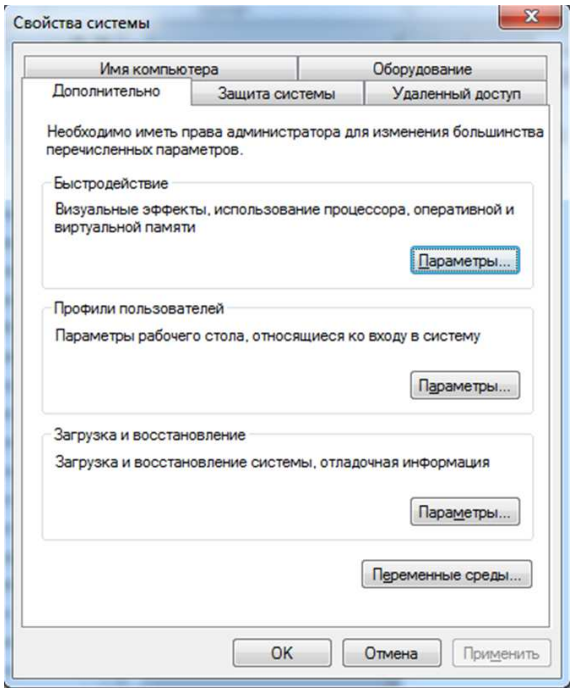


Рис. 2.11 – Вид окна «Свойства системы» на вкладке «Быстродействие»

Определение размера файла подкачки до сих пор вызывает многочисленные дискуссии. Основное правило заключается в том, что при небольшом объеме оперативной памяти файл подкачки должен быть достаточно большим. При большом объеме оперативной памяти файл подкачки можно уменьшить. Существует

возможность вообще ликвидировать файл подкачки, выполнив определенную настройку реестра. Однако в этом случае объем оперативной памяти должен быть достаточно большим, лишь немногие системы обладают подобными ресурсами.

По рекомендациям некоторых авторов можно установить исходный размер файла подкачки, равный размеру физической памяти, а максимальный размер не более двух размеров физической памяти. После этого следует нажать кнопку «Задать» (рис. 2.12) и убедиться в том, что новое значение файла подкачки установлено. Далее щелкнуть на кнопке «ОК».

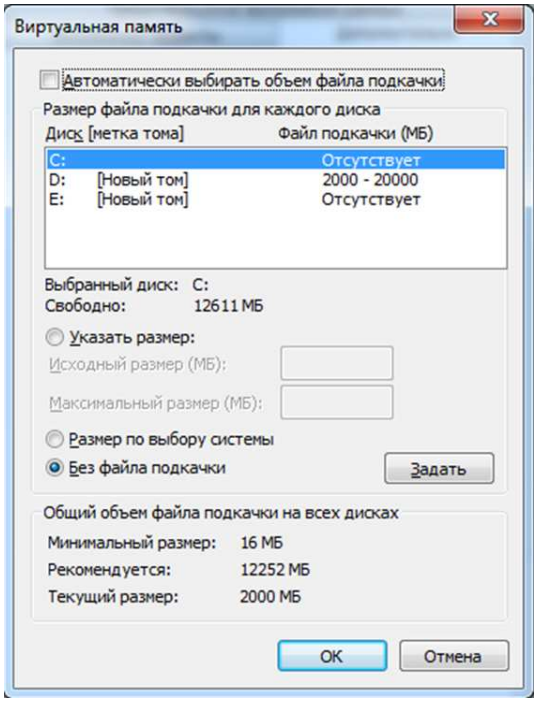


Рис. 2.12 – Вид окна «Виртуальная память»

Следует иметь в виду, что при первом создании файла подкачки жесткий диск, как правило, не готов к его размещению. Это обусловлено фрагментацией жесткого диска.

Поэтому нужно вначале выполнить дефрагментацию диска и лишь затем создать файл подкачки, чтобы поместить его в единственную область диска. Последовательность действий может быть, например, такой [16]:

1) если в компьютере имеется единственный жесткий диск, установить минимальный размер файла подкачки (2 Мбайт);
2) если имеется два жестких диска, переместить файл подкачки на более медленный диск;
3) провести дефрагментацию диска (во втором случае — быстрого). Для полной дефрагментации нужно выполнить несколько проходов;
4) присвоить файлу подкачки желаемый размер.

В результате работа с файлом подкачки станет максимально быстрой, а процессорная мощность и дисковое пространство будут использоваться эффективно.

Существует несколько приемов, позволяющих оптимизировать использование файла подкачки для повышения производительности. Файл подкачки следует по возможности размещать на отдельном жестком диске.

При наличии нескольких жестких дисков файл подкачки следует разделить, т. к. это повышает скорость работы с ним. Если при наличии двух жестких дисков разделить файл подкачки, то поскольку доступ к данным на обоих жестких дисках осуществляется одновременно, это значительно повысит производительность. Однако если имеются два жестких диска, из которых один быстрее другого, возможно, более эффективным решением будет размещение файла подкачки только на более быстром жестком диске. Определить наиболее производительную конфигурацию для данной системы можно экспериментальным путем.

Контрольные вопросы по главе 2

1. Опишите существующие единицы информации и их представление.
2. Дайте определения видам памяти и их месту в иерархии.
3. Что собой представляет адресация и распределение памяти в реальном режиме работы микропроцессора Intel x86?
4. Что собой представляет адресация и распределение памяти в защищенном режиме работы микропроцессора Intel x86?
5. Что собой представляет адресация и распределение памяти в архитектуре AMD64?
6. Как в ОС Windows на платформе NT можно узнать такие параметры, как общий размер физической памяти и размер памяти ядра ОС?
7. Что такое файл подкачки и как им можно управлять в ОС Windows на платформе NT?

Глава 3

УПРАВЛЕНИЕ УСТРОЙСТВАМИ ВВОДА-ВЫВОДА

3.1 Описание устройств ввода-вывода

3.1.1 Классификация устройств ввода-вывода

Внешние устройства, выполняющие операции ввода-вывода, можно разделить на три группы [17]:

1) *устройства, работающие с пользователем*. Используются для связи пользователя с компьютером. Сюда относятся принтеры, дисплеи, клавиатура, манипуляторы (мышь, трекболы, джойстики) и т. п.;
2) *устройства, работающие с компьютером*. Используются для связи с электронным оборудованием. К ним можно отнести дисковые устройства и устройства с магнитными лентами, датчики, контроллеры, преобразователи;
3) *коммуникации*. Используются для связи с удаленными устройствами. К ним относятся модемы и адаптеры цифровых линий.

По другому признаку устройства ввода-вывода можно разделить на два типа [17]: *блочные устройства* и *символьные устройства*.

Блочные устройства *— устройства, которые используются для хранения информации в виде блоков фиксированного размера, причем у каждого блока есть адрес и каждый блок может быть прочитан независимо от остальных блоков.*

Блоки обычно имеют фиксированный размер, кратный степени числа 2. Блок может быть переписан из внутренней памяти во внешнюю или обратно только це-

ликом, и для выполнения любой операции обмена с внешней памятью требуется специальная процедура (подпрограмма). Процедуры обмена с устройствами внешней памяти привязаны к типу устройства, его контроллеру и способу подключения устройства к системе (интерфейсу).

Символьные устройства *— устройства, которые используются для приема или передачи потока символов без какой-либо блочной структуры (принтеры, сетевые карты, мыши и т. д.).*

Однако некоторые из устройств не попадают ни в одну из этих категорий, например часы, мониторы и др. И все же модель блочных и символьных устройств является настолько общей, что может использоваться в качестве основы для достижения независимости от устройств некоторого программного обеспечения операционных систем, имеющего дело с вводом-выводом. Например, файловая система имеет дело с абстрактными блочными устройствами, а зависимую от устройств часть оставляет программному обеспечению низкого уровня.

По методу доступа к информации устройства внешней памяти разделяются на устройства с *прямым* (или непосредственным) и *последовательным* доступом [12].

Прямой доступ (direct access) подразумевает возможность обращения к блокам по их адресам в произвольном порядке. Традиционными устройствами с прямым доступом являются дисковые накопители, и часто в понятие «диск», или «дисковое устройство» (disk device), вкладывают значение «устройство внешней памяти прямого доступа». Так, например, виртуальный диск в ОЗУ и электронный диск на флэш-памяти отнюдь не имеют круглых, а тем более вращающихся деталей.

Традиционными устройствами с *последовательным доступом* являются накопители на магнитной ленте (tape device), они же стримеры. Здесь каждый блок информации тоже может иметь свой адрес, но для обращения к нему устройство хранения должно сначала найти некоторый маркер начала ленты (тома), после чего последовательным холостым чтением блока за блоком дойти до требуемого места и только тогда производить собственно операции обмена данными. Конечно, каждый раз возвращаться на начало ленты необязательно, однако необходимость последовательного сканирования блоков (вперед или назад) — неотъемлемое свойство устройств последовательного доступа. Несмотря на очевидный проигрыш во времени доступа к требуемым данным, ленточные устройства последовательного доступа в качестве внешней памяти находят применение для хранения очень больших массивов информации. В отличие от них устройства прямого доступа — диски самой различной природы — являются обязательной принадлежностью подавляющего большинства компьютеров.

3.1.2 Основные характеристики устройств внешней памяти

Главная характеристика устройств — *емкость хранения*, измеряемая в килобайтах (Кбайт), мегабайтах (Мбайт), гигабайтах (Гбайт) и терабайтах (Тбайт), или в английской транскрипции — KB, MB, GB, TB соответственно. Здесь, как правило, приставки кило-, мега-, гига-, тера- имеют десятичные значения — 10^3, 10^6, 10^9

и 10^{12} соответственно. В других подсистемах компьютера, например при определении объема ОЗУ, ПЗУ и другой внутренней памяти, эти же приставки чаще применяют в двоичных значениях 2^{10}, 2^{20}, 2^{30} и 2^{40}, соответственно при этом 1 Кбайт равен 1024 байтам, 1 Мбайт — 1024 Кбайт, 1 Гбайт — 1024 Мбайт, 1 Тбайт — 1024 Гбайт [12]. Этими разночтениями объясняются различия значений емкости одного и того же устройства, полученные из разных источников. «Двоичные» кило-, мега-, гига-, тера- более «увесисты», поэтому емкость устройства, отраженная в десятичных единицах, будет выглядеть внушительнее.

Важнейшими общими параметрами устройств являются время доступа, скорость передачи данных, удельная стоимость хранения информации, скорость записи и считывания.

Время доступа *(Access time) определяется как усредненный интервал времени от выдачи запроса на передачу блока данных до фактического начала передачи.*

Дисковые устройства имеют время доступа от единиц до сотен миллисекунд. Для электронных устройств внешней памяти время доступа определяется быстродействием используемых микросхем памяти и при чтении составляет доли микросекунд, причем запись может продолжаться значительно дольше, что объясняется природой энергонезависимой электронной памяти. Для устройств с подвижными носителями основной расход времени имеет место в процессе позиционирования головок (seek time — время поиска) и ожидания подхода к ним требуемого участка носителей (latency — скрытый период). Для дисковых и ленточных устройств принципы позиционирования различны, и различные составляющие процесса поиска будут подробнее рассмотрены в описании соответствующих устройств.

Скорость записи и считывания *определяется как отношение объема записываемых или считываемых данных ко времени, затрачиваемому на эту операцию.*

В затраты времени входит и время доступа, и время передачи данных. При этом оговаривается характер запросов (линейный или случайный), что сильно сказывается на величине скорости из-за влияния времени доступа. При определении скорости линейных запросов чтения-записи (linear transfer rate read/write) производится обращение к длинной цепочке блоков с последовательным нарастанием адреса. При определении скорости случайных запросов чтения-записи (random transfer rate read/write) соседние запросы разбросаны по всему носителю, что увеличивает время записи или считывания. Для современных многозадачных ОС характерно чередующееся выполнение нескольких потоков запросов, и в каждом потоке высока вероятность последовательного нарастания адреса.

Скорость передачи данных *(Transfer Speed, Transfer Rate) определяется как производительность обмена данными после выполнения поиска данных.*

Однако в способе измерения этого параметра возможны разночтения, поскольку современные устройства имеют в своем составе буферную память[1] существенных размеров. Скорости обмена буферной памяти с собственно носителем (внутренняя скорость) и внешним интерфейсом могут существенно различаться. Если скорость работы внешнего интерфейса ограничивается быстродействием электронных схем и достижимой частотой передаваемых сигналов, то внутренняя скорость более жестко ограничивается возможностями электромеханических устройств (скоростью движения носителя и плотностью записи). При измерениях скорости передачи на небольших объемах пересылок проявится ограничение внешнего интерфейса буферной памяти, при средних объемах — ограничение внутренней скорости, а при больших объемах проявится еще и время поиска последующих блоков информации. Бывает, что в качестве скорости передачи данных указывают лишь максимальную скорость интерфейса, а о внутренней скорости можно судить по частоте вращения дисковых носителей и числу секторов на треке, но об этих понятиях будет сказано чуть позже.

Определение *удельной стоимости хранения информации* для накопителей с фиксированными носителями пояснения не требует. В случае сменных носителей этот показатель интересен для собственно носителей, но не следует забывать и о цене самих приводов, которую тоже можно приводить к их емкости.

3.1.3 Характеристики накопителей на жестких магнитных дисках

Накопители на жестких магнитных дисках (НЖМД), они же HDD (Hard Disk Drive), являются главными устройствами дисковой памяти большинства компьютеров. По случайному совпадению цифр первые модели НЖМД назвали «винчестером» (просто игра слов), и это неофициальное название закрепилось в качестве синонима HDD и НЖМД [12]. Винчестер определяет мощность компьютера наряду с процессором и оперативной памятью. Мощность винчестера характеризуется большим объемом хранимой информации, малым временем доступа, большой скоростью передачи данных, высокой надежностью, умеренной стоимостью и рядом других полезных свойств. Прогресс в области производства винчестеров устойчив и стремителен: от 10-мегабайтного диска уже пришли к терабайтам.

В энциклопедии М. Гука [12] приводятся следующие ***общие параметры диска***:

- *форматированная емкость (formatted capacity)*, измеряемая в терабайтах (гигабайтах, мегабайтах), представляет собой объем хранимой полезной информации (сумму полей данных всех доступных секторов). Неформа-

[1] Буферной памятью называется объект памяти, которым владеют одновременно два объекта, не связанные между собой.

тированная емкость представляет собой максимальное количество битов, записываемых на всех треках диска, включая и служебную информацию (заголовки секторов, контрольные коды полей данных). Соотношение форматированной и неформатированной емкостей определяется форматом трека (размером сектора). Для обычных пользователей практический интерес представляет только форматированная емкость диска, которая указывается для стандартного размера сектора (512 байт);

- *скорость вращения шпинделя (spindle speed)*, измеряемая в оборотах в минуту RPM (Revolutions Per Minute), позволяет косвенно судить о производительности (внутренней скорости). Для жестких дисков широкого применения значение 3600 об/мин было стандартным несколько лет назад; сейчас обычной считается 5400 об/мин, а 7200 — более высокой скоростью. Там, где производительность особо критична, используют диски со скоростью 10 000 и 15 000 об/мин;
- *интерфейс (interface)* определяет способ подключения накопителя. Для накопителей со встроенным контроллером распространен интерфейс SATA, для устройств внешнего исполнения применяют шины USB, FireWire и Fibre Channel.

К группе параметров ***внутренней организации диска*** относятся [12]:

- *количество физических дисков (disks) или рабочих поверхностей (data surfaces)*, используемых для хранения данных. Современные накопители с небольшой высотой имеют малое (1–2) количество дисков для облегчения блока головок. Большее число дисков характерно для старых накопителей и современных накопителей большой емкости;
- *количество физических головок чтения-записи (read/write heads)*, естественно, совпадающее с числом рабочих поверхностей;
- *физическое количество цилиндров (cylinders)*, возросшее от нескольких сотен, характерных для первых винчестеров, до десятков тысяч;
- *размер сектора (Bytes Per Sector)*, составляющий обычно 512 байт. Количество зон и количество секторов на треке (Sectors Per Track) в крайних зонах;
- *расположение сервометок или сервоголовок (servo head)*. Используются для позиционирования головок диска;
- *метод кодирования-декодирования данных (recording method или data encoding sheme*). Информация на магнитном носителе хранится в аналоговой форме, а сами данные представлены в цифровом виде. При выполнении записи необходимо производить преобразование информации. Одним из последних методов можно назвать EPRML (Extended PRML) — улучшенный вариант PRML-кодирования (partial response, maximum likelihood) — частичный ответ, максимальное правдоподобие.

Быстродействие и производительность диска характеризуются следующими параметрами [12]:

- *временем перехода на соседний трек (track-to-track seek)*, измеряемым в миллисекундах, показывающим быстродействие системы позиционирования;

- *средним временем поиска (average seek time)*, определяемым по набору обращений к случайным цилиндрам. Чем больше объем накопителя, тем сложнее достичь снижения времени поиска: большее число головок труднее быстро перемещать; большее число цилиндров или увеличивает длину перемещения головок, или повышает требования к точности позиционирования;
- *максимальным или полным временем поиска (maximum seek time, full seek time)*, определяемым для самых удаленных переходов между крайними цилиндрами. Оно примерно в два раза превышает среднее время поиска;
- *внутренней скоростью передачи данных (internal transfer rate)* между носителем и буферной памятью контроллера, задающей физический предел производительности накопителя;
- *внешней скоростью передачи данных (external transfer rate)*, измеряемой в килобайтах (мегабайтах) полезных данных в секунду, передаваемых по шине внешнего интерфейса, и зависящей от быстродействия электроники контролера, типа интерфейсной шины и режима обмена;
- *длительной производительностью (sustained throughput)*, определяемой при последовательном чтении большого количества секторов. На этот параметр влияют все составляющие: внутренняя и внешняя скорости, время позицирования, задержка подхода сектора, количество ошибок позицирования и чтения.

Группа параметров ***надежности устройств и достоверности хранения*** включает следующие показатели [12]:

- *ожидаемое время до отказа MTBF (Mean Time Before Failur)*, измеряемое в сотнях тысяч часов, — среднестатистический показатель для данного устройства. Реально столько часов (100 000 часов — это более 10 лет) испытания проводить, естественно, невозможно. На самом деле делается выборка из большой группы устройств, из которых за вполне обозримое время испытаний какая-то часть выйдет из строя;
- наиболее значимый для пользователя параметр — *гарантийный срок (limited warranty)*, в течение которого изготовитель (или поставщик) обеспечивает ремонт или замену отказавшего устройства. Примечательно, что даже при MTBF, равном 800 000 часам (91 год), изготовитель дает гарантию всего на 3–5 лет;
- *вероятность неисправимых ошибок чтения (nonrecoverable read errors per bits read)*. Для современных винчестеров это приблизительно один раз в 115 дней при непрерывной работе. Вполне вероятно, что повторное считывание сектора пройдет без ошибок;
- *вероятность исправимых ошибок (recoverable read errors per bits read)*. При отсутствии контроллера или неисправной схеме контроля этот поток ошибок сделал бы работу с таким накопителем просто невыносимой (ошибки будут появляться чаще, чем один раз в три часа);
- *вероятность ошибок поиска (seek errors per seek)*, характеризующая качество сервосистемы. Для современных винчестеров характерна вероятность одной ошибки на 108 операций поиска. Эти ошибки при малом их числе

вполне безобидны, поскольку наличие номера цилиндра в заголовке каждого сектора не позволяет «промахнуться» при выполнении операций чтения или записи. Повторение операции поиска только слегка снижает среднее время доступа.

В отдельную группу параметров выделяется *уровень акустического шума*, который характеризуется *звуковой мощностью (sound power)*, излучаемой винчестером. На холостом ходу для винчестеров со скоростью вращения 5400 об/мин предпочтителен уровень до 30 дБ, при позиционировании желательно, чтобы он возрастал не более чем на 3–4 дБ. Для высокопроизводительных винчестеров (7200 об/мин) желательно, чтобы уровень шума не превышал 35 дБ на холостом ходу; для винчестеров, предназначенных для работы в устройствах бытовой электроники, — 25 дБ.

3.2 Организация дисковых устройств

3.2.1 Физическая структура магнитного диска

Загрузка собственно операционной системы и организация с ее помощью работы той или иной файловой системы осуществляется с магнитного диска. Были приняты специальные системные соглашения о структуре диска. Структура данных, содержащая информацию о логической организации диска, и простейшая программа, с помощью которой можно находить и загружать загрузочные программы той или иной ОС, — это первый (загрузочный) сектор магнитного диска.

Как известно, информация на магнитных дисках размещается и передается блоками. Каждый такой блок называется *сектором* (sector), сектора расположены на концентрических дорожках поверхности диска. Каждая дорожка (track) образуется при вращении магнитного диска под зафиксированной в некотором предопределенном положении головкой чтения/записи. *Накопитель на жестких магнитных дисках (НЖМД)* содержит один или более дисков (в современных распространенных НЖМД часто — два или три). Однако обычно под термином «жесткий диск» понимают весь пакет магнитных дисков.

Группы дорожек (треков) одного радиуса, расположенных на поверхностях магнитных дисков, образуют так называемые *цилиндры (cylinder)*. Современные жесткие диски могут иметь по несколько десятков тысяч цилиндров, в то время как на поверхности дискеты число дорожек (число цилиндров), как правило, составляет всего восемьдесят.

Каждый сектор состоит из *поля данных* и *поля служебной информации*, ограничивающей и идентифицирующей его. Размер сектора (точнее — емкость поля данных) устанавливается контроллером или драйвером. Пользовательский интерфейс операционных систем, как правило, поддерживает размер сектора — 512 байт.

Физический адрес сектора на диске определяется с помощью трех «координат», то есть представляется триадой [$c - h - s$], где c — номер цилиндра (дорожки на поверхности диска, cylinder), h — номер рабочей поверхности диска (магнитной головки, head), а s — номер сектора на дорожке. Номер цилиндра [c] лежит в диапазоне $0, \ldots, c - 1$, где c — количество цилиндров. Номер рабочей поверхности диска [h] принадлежит диапазону $0, \ldots, h - 1$, где h — число магнитных головок

в накопителе. Номер сектора на дорожке [*s*] указывается в диапазоне 1,…, *s*, где *s* — количество секторов на дорожке. Например, триада [1 – 0 – 2] адресует сектор 2 на дорожке 0 (обычно верхняя рабочая поверхность) цилиндра 1.

Обмен информацией между ОЗУ и дисками физически осуществляется только секторами. Вся совокупность физических секторов на винчестере представляет его неформатированную емкость.

3.2.2 Логическая структура магнитного диска

Жесткий диск может быть разбит на несколько *разделов* (partition), которые могут использоваться либо одной ОС, либо различными ОС. Причем главным является то, что на каждом разделе может быть организована своя файловая система. Однако для организации даже одной-единственной файловой системы необходимо определить, по крайней мере, один раздел. Разделы диска могут быть двух типов — *primary* (обычно этот термин переводят как *первичный*) и *extended* (*расширенный*). Максимальное число primary-разделов равно четырем. При этом на диске обязательно должен быть, по крайней мере, один primary-раздел. Если primary-разделов несколько, то только один из них может быть активным. Именно загрузчику, расположенному в активном разделе, передается управление при включении компьютера и загрузке операционной системы. Остальные primary-разделы в этом случае считаются «невидимыми, скрытыми» (*hidden*).

Согласно спецификациям на одном жестком диске может быть только один *extended-раздел*, который, в свою очередь, может быть разделен на большое количество подразделов — *логических дисков* (*logical*). В этом смысле термин «первичный» следует признать не совсем удачным переводом слова primary; можно это слово перевести и как «простейший, примитивный». В этом случае становится понятным и логичным термин «*extended*».

Один из primary-разделов должен быть *активным*, именно с него должна загружаться программа загрузки операционной системы, или так называемый *менеджер загрузки*, назначение которого — загрузить программу загрузки ОС из какого-нибудь другого раздела и уже с ее помощью загружать операционную систему. Поскольку до загрузки ОС система управления файлами работать не может, то следует использовать для указания упомянутых загрузчиков исключительно абсолютные адреса в формате [*c* – *h* – *s*].

Операционная система назначает логическим дискам расширенных разделов имена (буквы), остающиеся после дисков первичных разделов. Так, если имеется один жесткий диск и у него есть первичные и вторичный разделы, причем последний разбит на два логических диска, мы увидим следующее:

- C: — первичный раздел;
- D: — первый логический диск расширенного раздела;
- E: — второй логический диск расширенного раздела.

Теперь если добавить второй жесткий диск (всего с одним первичным разделом), то картина изменится:

- C: — первичный раздел первого диска (остался на месте);
- D: — первичный раздел второго диска (новый);

- E: — первый логический диск расширенного раздела первого диска (тот, что был D:);
- F: — второй логический диск расширенного раздела первого диска (тот, что был E:).

Если бы у нового диска был расширенный раздел со своими логическими дисками, то они бы заняли следующие буквы (G:, H:,…).

О механизме присвоения логических имен следует помнить, устанавливая программы на компьютер, которому эпизодически подключают дополнительные винчестеры. Незыблемое имя (C:) будет только у первичного раздела винчестера, подключенного к первому контроллеру.

По физическому адресу [0 – 0 – 1] на винчестере располагается *главная загрузочная запись MBR (Master Boot Record)*, содержащая *внесистемный загрузчик NSB (Non-System Bootstrap)* и *таблицу разделов PT (Partition Table)* [13]. Эта запись занимает ровно один сектор и размещается в памяти, начиная с адреса 0:7C00h, после чего управление передается коду, содержащемуся в первом секторе диска. Таким образом, в первом (стартовом) секторе физического жесткого диска находится не обычная запись boot record, как на дискете, а master boot record.

MBR является основным средством загрузки с жесткого диска, поддерживаемым BIOS. В MBR находятся три важных элемента:

1) *программа начальной загрузки (внесистемный загрузчик)*. Именно она запускается BIOS после успешной загрузки в память первого сектора с MBR. Она не превышает 512 байт, и ее хватает только для загрузки следующей, чуть более сложной программы — стартового сектора операционной системы — и передачи ей управления;
2) *таблица описания разделов диска*, располагающаяся в MBR по смещению lBEh и занимающая 64 байта;
3) *сигнатура MBR*. Последние два байта MBR должны содержать число AA55h. По наличию этой сигнатуры BIOS проверяет, что первый блок был загружен успешно. Сигнатура эта выбрана не случайно. Ее успешная проверка позволяет установить, что все линии передачи данных могут передавать и нули, и единицы.

Упрощенно структура MBR представлена в таблице 3.1.

*Таблица описания разделов **partition table** описывает размещение и характеристики имеющихся на винчестере разделов.*

Можно сказать, что таблица разделов — одна из наиболее важных структур данных на жестком диске. Если эта таблица повреждена, то не только не будет загружаться операционная система (или одна из операционных систем, установленных на винчестере), но перестанут быть доступными и данные, расположенные на винчестере, особенно если жесткий диск был разбит на несколько разделов.

В таблице 3.1 показано, что в начале загрузочного сектора располагается программа анализа таблицы разделов и чтения первого сектора из активного раздела диска. Сама таблица partition table располагается в конце MBR, и для описания каждого раздела в этой таблице отводится по 16 байтов.

Таблица 3.1 – Структура MBR

Смещение (Offset)	Размер (Sise), байт	Содержимое (Contents)
0	446	Программа анализа Partition Table и загрузки System Bootstrap с активного раздела жесткого диска
+1BEh	16	Partition 1 entry (Описатель раздела)
+1CEh	16	Partition 2 entry
+1DEh	16	Partition 3 entry
+1EEh	16	Partition 4 entry
+1FEh	2	Сигнатура (AA55h)

Первым байтом в элементе раздела идет флаг активности раздела boot indicator (0 — не активен, 128 (80h) — активен). Он определяет, является ли раздел системным загрузочным и есть ли необходимость производить загрузку операционной системы с него при старте компьютера. Активным может быть только один раздел. За флагом активности раздела следует байт номера головки, с которой начинается раздел. За ним следуют два байта, означающие соответственно номер сектора и номер цилиндра загрузочного сектора, где располагается первый сектор загрузчика операционной системы. Затем следует кодовый идентификатор System ID длиной в один байт, указывающий на принадлежность данного раздела к той или иной операционной системе и тип установленной на нем файловой системы.

За байтом кода операционной системы расположен байт номера головки конца раздела, за которым идут два байта — номер сектора и номер цилиндра последнего сектора данного раздела. Ниже представлен формат элемента таблицы разделов (табл. 3.2).

Таблица 3.2 – Формат элемента таблицы разделов

Название записи элемента Partition Table	Длина, байт
Флаг активности раздела	1
Номер головки начала раздела	1
Номер сектора и номер цилиндра загрузочного сектора раздела	2
Кодовый идентификатор операционной системы	1
Номер головки конца раздела	1
Номер сектора и цилиндра последнего сектора раздела	2
Младшее и старшее двухбайтовое слово относительного номера начального сектора	4
Младшее и старшее двухбайтовое слово размера раздела в секторах	4

В таблице 3.3 приведены наиболее известные идентификаторы.

Таблица 3.3 – Типы разделов жесткого диска

System ID, идентификатор	Тип раздела	System ID, идентификатор	Тип раздела
00	Empty («пустой» раздел)	41	PPC PreP Boot
01	FAT12	42	SFS
02	XENIX root	4D	QNX 4.x
03	XENIX usr	4E	QNX 4.x 2nd part
04	FAT16 (< 32 Мбайт)	4F	QNX 4.x 3nd part
05	Extended	50	OnTrack DM
06	FAT16	51	OnTrack DM6 Aux
07	HPFS/NTFS	52	CP/M
08	AIX	53	OnTrack DM6
09	AIX bootable	54	OnTrack DM6
0A	OS/2 Boot Manager	55	EZ Drive
0B	Win95 FAT32	56	Golden Bou
0C	Win95 FAT32 LBA	5C	Priam Edisk
0E	Win95 FAT16 LBA	61	Speed Stor
0F	Win95 Extended	64	Novell Netware
10	OPUS	65	Novell Netware
11	Hidden FAT12	75	PC/IX
12	Compaq diagnost	80	Old Minix
14	Hidden FAT16 (< 32 Мбайт)	82	Linux swap
16	Hidden FAT16	83	Linux native
17	Hidden HPFS/NTFS	84	OS/2 hidden C:
18	AST Windows swap	85	Linux Extended
1B	Hidden Win95 Fat	86	NTFS volume set
1C	Hidden Win95 Fat	A5	BSD/386
1E	Hidden Win95 Fat	A6	Open BSD
24	NEC DOS	A7	Next Step
3C	Partition Magic	EB	Be OS
40	Venix 80286		

Вслед за сектором MBR размещаются собственно сами разделы. В процессе начальной загрузки сектора MBR, содержащего таблицу partition table, работают программные модули BIOS. Начальная загрузка считается выполненной корректно только в том случае, когда таблица разделов содержит допустимую информацию.

В MS DOS в первичном разделе может быть сформирован только один логический диск, а в расширенном — любое их количество. Каждый логический диск

«управляется» своим логическим приводом. Каждому логическому диску на винчестере соответствует своя (относительная) *логическая* нумерация. Физическая же адресация жесткого диска сквозная.

Первичный раздел DOS включает только *системный логический диск* без каких-либо дополнительных информационных структур.

Расширенный раздел DOS содержит вторичную запись *SMBR (Secondary MBR)*, в состав которой вместо partition table входит таблица логического диска *LDT (Logical Disk Table)*, ей аналогичная.

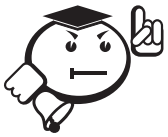


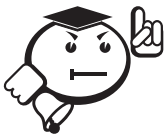
Таблица LDT описывает размещение и характеристики раздела, содержащего единственный логический диск, а также может специфицировать следующую запись SMBR. Следовательно, если в расширенном разделе DOS создано K логических дисков, то он содержит K экземпляров SMBR, связанных в список. Каждый элемент этого списка описывает соответствующий логический диск и ссылается (кроме последнего) на следующий элемент списка.

Для работы с разделами жесткого диска в MS DOS, Windows 9x используется утилита fdisk.exe, а в Windows на платформе NT используется «Панель управления», далее «Администрирование», «Управление компьютером». Существует также очень удобная, но не входящая в состав операционных систем утилита Partition Magic (фирмы PowerQuest), предназначенная для работы с разделами жестких дисков. Она выполняет такие функции с разделами, как форматирование под различные файловые системы, перемещение, изменение размеров, активизацию, скрытие и т. д.

3.3 Обзор файловых систем

3.3.1 Файловая система FAT

FAT16. Файловая система FAT16 была разработана еще до создания MS DOS и в настоящее время поддерживается всеми операционными системами Microsoft для обеспечения совместимости, а так же большим количеством операционных систем других производителей. Ее название — таблица расположения файлов (File Allocation Table) — отлично отражает физическую организацию файловой системы, к основным характеристикам которой можно отнести максимальный размер поддерживаемого тома (жесткого диска или раздела на жестком диске), не превышающий 4095 Мбайт. В период эксплуатации MS DOS 4-гигабайтные жесткие диски казались несбыточной мечтой (роскошью были диски объемом 20–40 Мбайт), поэтому такой запас был вполне оправданным.



FAT может состоять из 12- или 16-разрядных элементов. Для дисков с объемом менее 384 Кбайт очень эффективны 12-разрядные элементы. Файловая система такого диска в полном объеме помещается в один сектор (512 байтов). FAT12 использовалась для работы с гибкими магнитными дисками.

Том, отформатированный для использования FAT16, разделяется на кластеры. Размер кластера по умолчанию зависит от размера тома и может колебаться от 512 байт до 64 Кбайт. В таблице 3.4 показана зависимость размера кластера от размера тома. Размер кластера может отличаться от значения по умолчанию, но должен иметь одно из значений, указанных в таблице 3.4.

Таблица 3.4 – Зависимость размера тома от размера кластера в FAT16

Размер тома, Мбайт	**Число секторов в кластере**	**Размер кластера, Кбайт**
0–31	1	0,5 (512 байт)
32–63	2	1
64–127	4	2
128–255	8	4
256–511	16	8
512–1023	32	16
1024–2047	64	32
2048–4095	128	64

Не рекомендуется применять файловую систему FAT16 на томах размером больше 511 Мбайт, так как для относительно небольших по объему файлов дисковое пространство будет использоваться крайне неэффективно: файл размером в 1 байт будет занимать 64 Кбайт. Независимо от размера кластера файловая система FAT16 не поддерживается для томов размером больше 4 Гбайт.

На рисунке 3.1 показано, как организован том при использовании файловой системы FAT16.

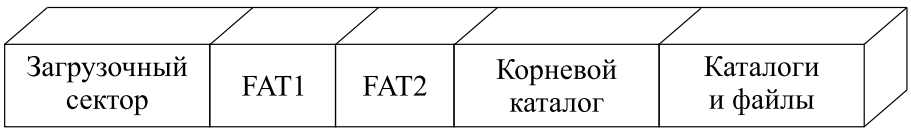


Рис. 3.1 – Структура тома при использовании FAT16

Первый сектор тома является загрузочным сектором. Далее за ним идут таблицы FAT1 и FAT2. Таблица FAT — это часть файловой системы FAT. Она содержит элементы, описывающие состояния кластеров в томе. FAT2 является копией FAT1. При использовании файловой системы FAT16 за второй копией таблицы FAT всегда располагается корневой каталог. Единственным различием между корневым каталогом и другими является то, что корневой располагается в определенном месте и имеет фиксированное число вхождений. Каждый каталог и файл используют

одно или более вхождений. Например, если число фиксированных вхождений для корневого каталога равно 512 и создано 100 подкаталогов, в корневом каталоге можно создать не более 412 файлов (512–100).

Для каждого файла и каталога в файловой системе хранится информация в соответствии со структурой, изображенной в таблице 3.5.

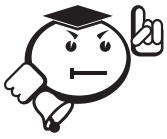
Таблица 3.5 – Структура элемента корневого каталога

Размер поля данных, байт	Содержание поля
11	Имя файла или каталога
1	Атрибуты файла
1	Резервное поле
3	Время создания
2	Дата создания
2	Дата последнего доступа
2	Зарезервированное
2	Время последней модификации
2	Дата последней модификации
2	Номер начального кластера в FAT
4	Размер файла

Каждый элемент каталога содержит номер начального кластера файла, описываемого данным элементом. Этот номер является указателем в FAT, где содержится информация об остальных кластерах файла, организованная в связный список.

В FAT16 кластеры могут иметь различное значение:

- (0)000h — свободный кластер;
- (F)FF0h — (F)FF6h — зарезервированный кластер;
- (F)FF7h — дефектный кластер;
- (F)FF8h — (F)FFFh — конец файла;
- (0)002h — (F)FEFh — номер следующего кластера файла.



Старшая тетрада, заключенная в скобки, относится к 16-разрядным элементам. Например, дефектный кластер помечается FF7h в 12-разрядный FAT и FFF7h — в 16-разрядный FAT.

Расположение файлов по кластерам показано на рисунке 3.2. В папке расположены три файла; первый из них — File1 — занимает три кластера (файл не фрагментирован, кластеры 2, 3 и 4 расположены последовательно); второй файл — File2 — фрагментирован и располагается в кластерах 5, 6 и 8; третий — File3 — занимает всего один кластер. Вхождение для каждого файла содержит адрес его начального кластера (2, 5 и 7 соответственно). Последний кластер каждого файла (4, 8 и 7) в качестве адреса следующего кластера содержит значение FFFF, указывающее на то, что это последний кластер для данного файла.

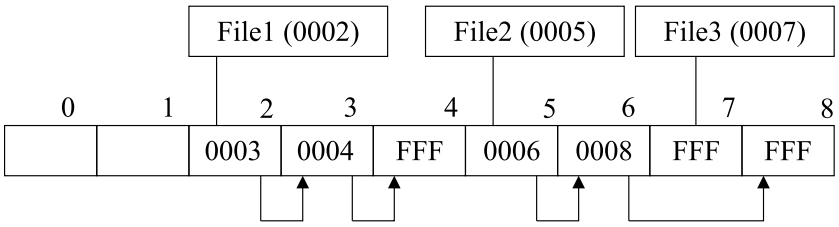


Рис. 3.2 – Пример расположения файлов по кластерам в FAT16

Так как все вхождения имеют одинаковый размер информационного блока, они различаются по байту атрибутов. Один из битов в данном байте может указывать, что это каталог, другой — что это метка тома. Для пользователей доступны четыре бита, позволяющие управлять атрибутами файла: архивный (archive), системный (system), скрытый (hidden), доступный только для чтения (read-only).

Среди *преимуществ FAT16* можно отметить следующие:

- файловая система поддерживается операционными системами MS DOS, Windows 95, Windows 98, Windows NT, Windows 2000, а также некоторыми операционными системами UNIX;
- существует большое число программ, позволяющих исправлять ошибки в этой файловой системе и восстанавливать данные;
- при возникновении проблем с загрузкой с жесткого диска система может быть загружена с флоппи-диска;
- данная файловая система достаточно эффективна для томов объемом менее 256 Мбайт.

К *основным недостаткам FAT16* относятся следующие:

- корневой каталог не может содержать более 512 элементов. Использование длинных имен файлов существенно сокращает число этих элементов;
- FAT16 поддерживает не более 65 536 кластеров, а так как некоторые кластеры зарезервированы операционной системой, то число доступных кластеров составляет 65 524. Каждый кластер имеет фиксированный размер для данного логического устройства. При достижении максимального числа кластеров с максимальным размером в 32 килобайта максимальный объем поддерживаемого тома ограничивается 4-мя гигабайтами под управлением Windows 2000. Для поддержания совместимости с MS DOS, Windows 95 и Windows 98 объем тома под FAT16 не должен превышать 2 Гбайт;
- не поддерживается резервная копия загрузочного сектора;
- в FAT16 не поддерживается встроенная защита файлов и их сжатие;
- на дисках большого объема теряется много места за счет того, что используется максимальный размер кластера. Место под файл выделяется исходя из размера не файла, а кластера.

FAT32. В версии Microsoft Windows 95 OEM Service Release 2 (OSR2) в Windows появилась поддержка 32-битной FAT. Для систем на базе Windows NT эта файловая система впервые стала поддерживаться в Microsoft Windows 2000. Если FAT16 может поддерживать тома объемом до 4 Гбайт, то FAT32 способна обслуживать тома объемом до 4 Тбайт. Размер кластера в FAT32 может изменять-

ся от 1 (512 байт) до 64 секторов (32 Кбайт). Для хранения значений кластеров FAT32 требуется 4 байта (32 бит, а не 16, как в FAT16). Это означает, в частности, что некоторые файловые утилиты, рассчитанные на FAT16, не могут работать с FAT32.

Основным отличием FAT32 от FAT16 является изменение размера логического раздела диска. При этом если при использовании FAT16 с 2-гигабайтными дисками требовался кластер размером в 32 Кбайт, то в FAT32 кластер размером в 4 Кбайта подходит для дисков объемом от 512 Мбайт до 8 Гбайт (табл. 3.6). Это соответственно означает более эффективное использование дискового пространства — чем меньше кластер, тем меньше места требуется для хранения файла и, как следствие, диск реже становится фрагментированным.

Таблица 3.6 – Зависимость размера тома от размера кластера в FAT32

Размер раздела, Гбайт	**Размер кластера по умолчанию, Кбайт**
Менее 8	4
От 8 до 16	8
От 16 до 32	16
32 и более	32

При применении FAT32 максимальный размер файла может достигать 4 Гбайт минус 2 байта. Если при использовании FAT16 максимальное число вхождений в корневой каталог ограничивалось 512, то FAT32 позволяет увеличить это число до 65 535.

FAT32 накладывает ограничения на минимальный размер тома — не менее 65 527 кластеров. При этом размер кластера не может быть таким, при котором бы FAT занимала более 16 Мбайт — 64 Кбайт/4 или 4 млн кластеров.

При использовании длинных имен файлов данные, необходимые для доступа из FAT16 и FAT32, не перекрываются. При создании файла с длинным именем Windows создает соответствующее имя в формате 8.3 (восемь символов — имя файла, три — расширение файла) и одно или более вхождений в каталог для хранения длинного имени (по 13 символов из длинного имени файла на каждое вхождение). Каждое последующее вхождение хранит соответствующую часть имени файла в формате Unicode. Такие вхождения имеют атрибуты «идентификатор тома», «только чтение», «системный» и «скрытый». Атрибут вхождения «скрытый» характеризует набор, игнорируемый MS DOS, в которой доступ к файлу осуществляется по его «псевдониму» в формате 8.3.

Среди *преимуществ FAT32* можно отметить следующие:

- выделение дискового пространства выполняется более эффективно, особенно для дисков большого объема;
- корневой каталог в FAT32 представляет собой обычную цепочку кластеров и может находиться в любом месте диска;
- за счет использования кластеров меньшего размера (4 Кбайта на дисках объемом до 8 Гбайт) занятое дисковое пространство обычно на 10–15% меньше, чем под FAT16;

- FAT32 является более надежной файловой системой. В частности, она поддерживает возможность перемещения корневого каталога и использования резервной копии FAT. Кроме того, загрузочная запись содержит ряд критичных для файловой системы данных.

Основные недостатки FAT32:

- размер тома при использовании FAT32 под Windows 2000 ограничен 32 Гбайт;
- тома FAT32 недоступны из многих операционных систем, которые поддерживают FAT;
- не поддерживается резервная копия загрузочного сектора;
- в FAT32 не поддерживается встроенная защита файлов и их сжатие.

3.3.2 Файловая система NTFS

В названии файловой системы *NTFS (New Technology File System)* содержатся слова «новая технология». Действительно, NTFS характеризуется рядом значительных усовершенствований и изменений, существенно отличающих ее от других файловых систем. С точки зрения пользователей, файлы по-прежнему хранятся в каталогах, часто называемых «папками», или фолдерами, в среде Windows. Однако в NTFS, в отличие от FAT, работа на дисках большого объема происходит намного эффективнее:

- имеются средства для ограничения в доступе к файлам и каталогам;
- введены механизмы, существенно повышающие надежность файловой системы;
- сняты многие ограничения на максимальное количество дисковых секторов и/или кластеров.

При проектировании системы NTFS особое внимание было обращено на следующие характеристики:

- *надежность*. Высокопроизводительные компьютеры и системы совместного пользования (серверы) должны обладать повышенной надежностью, которая является ключевым элементом структуры и поведения NTFS. Одним из способов увеличения надежности является введение механизма транзакций, при котором осуществляется *журналирование файловых операций*;
- *расширенную функциональность*. NTFS проектировалась с учетом возможного расширения. В ней были реализованы многие дополнительные возможности: усовершенствованная отказоустойчивость; эмуляция других файловых систем; мощная модель безопасности; параллельная обработка потоков данных; создание файловых атрибутов, определяемых пользователем;
- *поддержку платформенно-независимого системного интерфейса для компьютерного окружения POSIX* (Portable Operating System Interface for Computer Environments). Поскольку правительство США требовало, чтобы все закупаемые им системы хотя бы в минимальной степени соответствовали стандарту POSIX, такая возможность была предусмотрена и в NTFS. К числу базовых средств файловой системы POSIX относится необязательное использование имен файлов с учетом регистра, хранение времени послед-

него обращения к файлу и механизм так называемых «жестких ссылок» (альтернативных имен, позволяющих ссылаться на один и тот же файл по двум и более именам);

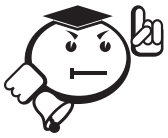
- *гибкость*. Модель распределения дискового пространства в NTFS отличается чрезвычайной гибкостью. Размер кластера может изменяться от 512 байт до 64 Кбайт; он представляет собой число, кратное внутреннему кванту распределения дискового пространства. NTFS также поддерживает длинные имена файлов, набор символов Unicode и альтернативные имена формата 8.3 для совместимости с FAT.

Как и при использовании FAT, основной информационной единицей в NTFS является кластер. В таблице 3.7 показаны размеры кластеров по умолчанию для томов различной емкости.

Таблица 3.7 – Зависимость размера тома от размера кластера в NTFS

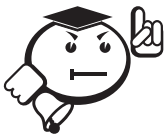
Размер тома, Мбайт	Число секторов в кластере	Размер кластера, Кбайт
512 и менее	1	0,5 (512 байт)
513–1024 (1 Гбайт)	2	1
1025–2048 (2 Гбайт)	4	2
Более 2049	8	4

Теоретически NTFS поддерживает тома с числом кластеров до 2^{32}. Индустриальные стандарты ограничивают размер таблицы разделов 2^{32} секторами. Другим ограничением является размер сектора, который обычно равен 512 байтам. Поскольку размер сектора может измениться в будущем, текущий размер дает ограничение на размер одного тома — 2 Тбайта ($2^{32} \times 512$ байт = 2^{41}).



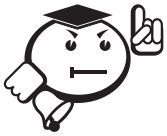
Таким образом, размер тома в 2 Тбайта является практическим пределом для физических и логических томов NTFS.

Управление доступом к файлам и каталогам. При использовании томов NTFS можно устанавливать права доступа к файлам и каталогам. Эти права доступа указывают, какие пользователи и группы имеют доступ к ним и какой уровень доступа допустим. Такие права доступа распространяются как на пользователей, работающих за компьютером, на котором располагаются файлы, так и на пользователей, обращающихся к файлам через сеть, когда файл располагается в каталоге, открытом для удаленного доступа. Под NTFS можно также устанавливать разрешения на удаленный доступ, объединяемые с разрешениями на доступ к файлам и каталогам. Помимо этого, файловые атрибуты (только чтение, скрытый, системный) также ограничивают доступ к файлу. Под управлением FAT16 и FAT32 тоже можно устанавливать атрибуты файлов, но они не обеспечивают права доступа к файлам.



В версии NTFS, используемой в Windows 2000, появился новый тип разрешения на доступ — наследуемые разрешения.

Сжатие файлов и каталогов. В Windows 2000 поддерживается сжатие файлов и каталогов, расположенных на NTFS-томах. Сжатые файлы доступны для чтения и записи любыми Windows-приложениями. Для этого нет необходимости в их предварительной распаковке. Используемый алгоритм сжатия схож с тем, который используется в Double-Space (MS DOS 6.0) и DriveSpace (MS DOS 6.22), но имеет одно существенное отличие — под управлением MS DOS выполняется сжатие целого первичного раздела или логического устройства, тогда как под NTFS можно упаковывать отдельные файлы и каталоги.



Алгоритм сжатия в NTFS разработан с учетом поддержки кластеров размером до 4 Кбайт. Если величина кластера больше 4 Кбайт, функции сжатия NTFS становятся недоступными.

Самовосстановление NTFS. Файловая система NTFS обладает способностью самовосстановления и может поддерживать свою целостность за счет использования протокола выполняемых действий и ряда других механизмов. NTFS рассматривает каждую операцию, модифицирующую системные файлы на NTFS-томах, как транзакцию и сохраняет информацию о такой транзакции в протоколе. Начатая транзакция может быть либо полностью завершена (commit), либо откатывается (rollback). В последнем случае NTFS-том возвращается в состояние, предшествующее началу транзакции. Для того чтобы управлять транзакциями, перед тем как осуществить запись на диск, NTFS записывает все операции, входящие в транзакцию, в файл протокола. После того как транзакция завершена, все операции выполняются. Таким образом, под управлением NTFS не может быть незавершенных операций. В случае дисковых сбоев незавершенные операции просто отменяются. Под управлением NTFS также выполняются операции, позволяющие «на лету» определять дефектные кластеры и отводить новые кластеры для файловых операций. Этот механизм называется «*cluster remapping*».

При формировании файловой системы NTFS программа форматирования создает файл MFT (Master File Table) и другие области для хранения метаданных. Метаданные используются NTFS для реализации файловой структуры. Первые 16 записей в MFT зарезервированы самой NTFS. Местоположение файлов метаданных $Mft и $MftMirr записано в загрузочном секторе диска. Если первая запись в MFT повреждена, NTFS считывает вторую запись для нахождения копии первой. Полная копия загрузочного сектора располагается в конце тома. Основные метаданные, хранимые в MFT, перечислены в таблице 3.8.

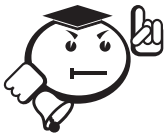
Остальные записи MFT содержат записи для каждого файла и каталога, расположенных на данном томе.

Таблица 3.8 – Основные метаданные MFT

Системные файлы	Имя файла	Запись MFT	Содержание записи MFT
Master file table	$Mft	0	Одна базовая файловая запись для каждого файла или каталога на томе NTFS
Master file table2	$MftMirr	1	Копия первых четырех записей MFT. Гарантирует доступ к MFT в случае, если первый сектор поврежден
Log file	$LogFile	2	Список действий, необходимых для восстановления NTFS
Volume	$Volume	3	Информация о томе — метка и номер версии
Attribute definitions	$AttrDef	4	Таблица имен атрибутов и описание
Root file name index	$	5	Корневой каталог
Cluster bitmap	$Bitmap	6	Информация о занятых кластерах
Boot sector	$Boot	7	Код загрузки для загрузочных томов
Bad cluster file	$BadClus	8	Информация о дефектных кластерах
Security file	$Secure	9	Уникальные дескрипторы для всех файлов
Upcase table	$Upcase	10	Информация для преобразования символов нижнего регистра в соответствующие Unicode-символы верхнего регистра
NTFS extension file	$Extend	11	Информация для различных служб ОС: службы квот, службы пересчета и идентификаторы объектов
		12–15	Зарезервированные записи для будущих версий

Обычно один файл использует одну запись в MFT, но если у файла большой набор атрибутов или он становится слишком фрагментированным, то для хранения информации о нем могут потребоваться дополнительные записи. В этом случае первая запись о файле, называемая базовой записью, хранит местоположение других записей. Данные о файлах и каталогах небольшого размера (до 1500 байт) полностью содержатся в первой записи.

Каждый занятый сектор на NTFS-томе принадлежит тому или иному файлу. Даже метаданные файловой системы являются частью файла. NTFS рассматривает каждый файл (или каталог) как набор файловых атрибутов. Такие элементы, как имя файла, информация о его защите и даже данные в нем, являются атрибутами файла. Каждый атрибут идентифицируется кодом определенного типа и именем атрибута.



Если атрибуты файла вмещаются в файловую запись, они называются резидентными атрибутами.

Такими атрибутами всегда являются имя файла и дата его создания. В тех случаях, когда информация о файле слишком велика, чтобы вместиться в одну MFT-запись, некоторые атрибуты файла становятся нерезидентными. Резидентные атрибуты хранятся в одном или более кластерах и представляют собой поток альтернативных данных для текущего тома. Для описания местонахождения резидентных и нерезидентных атрибутов NTFS создает атрибут Attribute List.

Возможности файловой системы NTFS по ограничению доступа к файлам и каталогам. Благодаря наличию механизма расширенных атрибутов в NTFS реализованы ограничения в доступе к файлам и каталогам. Эти дополнительные атрибуты, использованные для ограничения в доступе к файловым объектам, назвали атрибутами безопасности. При каждом обращении к такому объекту сравнивается специальный список дискреционных прав доступа, приписанный ему, со специальным системным идентификатором, несущим информацию об имени пользователя, осуществляющего текущий запрос к файлу или каталогу. Если имеется в списке необходимое разрешение, то действие выполняется, в противном случае система сообщает об отказе.

Файловая система NTFS имеет так называемые индивидуальные разрешения, которые могут быть приписаны любому файлу и/или каталогу: ***Read*** (*прочитать*), ***Write*** (*записать*), ***eXecute*** (*выполнить*), ***Delete*** (*удалить*), ***Change Permissions*** (*изменить разрешения*) и ***Take Ownership*** (*стать владельцем*). Соответствующие этим разрешениям действия можно выполнять только в случаях, когда для данного пользователя или группы, к которой он принадлежит, имеется одноименное разрешение. Другими словами, если для некоторого файла указано, что все пользователи могут его читать и исполнять, то только эти действия и можно с ним сделать, если при этом не указано, что для какой-нибудь другой группы пользователей (отдельного пользователя) имеются другие разрешения. Комбинации индивидуальных разрешений и определяют действия, которые могут быть выполнены с файлом или каталогом.

Изначально всему диску, а значит, и файлам, которые на нем создаются, присвоены все индивидуальные разрешения для группы Everyone (все). Это означает, что любой пользователь, имея полный набор индивидуальных разрешений на файлы и каталоги, может изменять их по своему усмотрению, т. е. ограничивать других пользователей в правах доступа к тому или иному объекту. Если изменить разрешения на каталог, то новые файлы, создаваемые в нем, будут получать и соответствующие разрешения: они будут наследовать разрешения своего родительского каталога.

Каталоги обычно обладают теми же разрешениями, что и находящиеся в них файлы и папки, хотя у каждого файла могут быть свои разрешения. Разрешения, которые имеются у файла, имеют приоритет над разрешениями, которые установлены на каталог, в котором находится этот файл. Например, если вы создаете каталог внутри другого каталога, для которого администраторы обладают правом полного доступа, а пользователи — правом чтения, то новый каталог унаследует

эти права. То же относится и к файлам, копируемым из другого каталога или перемещаемым из другого раздела NTFS.

Если каталог или файл перемещается в другой каталог того же раздела NTFS, то атрибуты безопасности не наследуются от нового каталога. Дело в том, что при перемещении файлов в границах одного раздела NTFS изменяется только указатель местонахождения объекта, а все остальные атрибуты (включая атрибуты безопасности) остаются без изменений.

Существует три важных правила, которые помогут определить состояние прав доступа при перемещении или копировании объектов NTFS:

1) при перемещении файлов в границах раздела NTFS сохраняются исходные права доступа;
2) при выполнении других операций (создании или копировании файлов, а также их перемещении между разделами NTFS) наследуются права доступа родительского каталога;
3) при перемещении файлов из раздела NTFS в раздел FAT все права NTFS теряются.

Основные отличия FAT и NTFS. Если говорить о дополнительных расходах на хранение служебной информации, то можно отметить, что FAT отличается от NTFS большей компактностью и меньшей сложностью. В большинстве томов FAT на хранение таблицы размещения, содержащей информацию обо всех файлах тома, расходуется менее 1 Мбайт. Столь низкие дополнительные расходы позволяют форматировать в FAT жесткие диски малого объема и флоппи-диски. В NTFS служебные данные занимают больше места, чем в FAT. Так, каждый элемент каталога занимает 2 Кбайта. Однако это имеет и свои преимущества, так как содержимое файлов объемом 1500 байт и менее может полностью храниться в элементе каталога.

Система NTFS не может использоваться для форматирования флоппи-дисков. Не стоит пользоваться ею для форматирования разделов объемом менее 50–100 Мбайт. Относительно высокие дополнительные расходы приводят к тому, что для малых разделов служебные данные могут занимать до 25% объема носителя.

Следующий критерий сравнения — размер файлов. Разделы FAT имеют объем до 2 Гбайт, FAT32 — до 4 Тбайт. Тем не менее из-за особенностей своего внутреннего строения разделы FAT лучше всего работают для разделов объемом 200 Мбайт и менее. В NTFS в настоящее время из-за аппаратных и других системных причин размер файлов ограничивается 2 терабайтами.

Разделы FAT могут использоваться практически во всех операционных системах. За редкими исключениями, с разделами NTFS можно работать напрямую только из Windows NT, хотя и имеются для ряда ОС соответствующие реализации систем управления файлами для чтения файлов из томов NTFS. Так, например, утилита (драйвер) NTFSDOS позволяет читать данные NTFS на компьютере, загруженном в режиме MS DOS. Однако полноценных реализаций для работы с NTFS вне системы Windows NT пока нет.

Разделы FAT не обеспечивают локальной безопасности, в то время как разделы NTFS обеспечивают локальную безопасность как файлов, так и каталогов. Для разделов FAT могут устанавливаться общие права, связанные с общим доступом к каталогам в сети. Однако такая защита не помешает пользователю с локальным

входом получить доступ к файлам своего компьютера. В вопросе организации системы безопасности NTFS оказываются более предпочтительным вариантом. Разделы NTFS могут запрещать или ограничивать доступ как удаленных, так и локальных пользователей. Следовательно, к защищенным файлам смогут обратиться лишь те пользователи, которым были предоставлены соответствующие права.

Windows NT содержит специальную утилиту CONVERT.EXE, которая преобразует тома FAT в эквивалентные тома NTFS, однако для обратного преобразования из NTFS в FAT подобных утилит не существует. Чтобы выполнить обратное преобразование, необходимо создать раздел FAT, скопировать в него файлы из раздела NTFS и затем удалить оригиналы. Важно при этом не забывать и о том, что при копировании файлов из NTFS в FAT теряются все атрибуты безопасности NTFS. Напомним, что в FAT не предусмотрены средства для определения и последующего хранения этих атрибутов.

3.3.3 Файловая система HPFS

HPFS (High Performance File System) — высокопроизводительная файловая система — впервые появилась в OS/2 1.2 и LAN Manager [13]. HPFS была разработана совместными усилиями лучших специалистов компании IBM и Microsoft на основе опыта IBM по созданию файловых систем MVS, VM/CMS и виртуального метода доступа. Архитектура HPFS начала создаваться как файловая система, которая сможет использовать преимущества многозадачного режима и обеспечит в будущем более эффективную и надежную работу с файлами на дисках большого объема.

HPFS стала первой файловой системой для персонального компьютера, в которой была реализована поддержка длинных имен. HPFS, как FAT и многие другие файловые системы, обладает структурой каталогов, но в ней также предусмотрены автоматическая сортировка каталогов и специальные расширенные атрибуты, упрощающие реализацию безопасности файлового уровня и создание множественных имен. HPFS поддерживает те же самые атрибуты, что и файловая система FAT, а также и новую форму file-associated, то есть информацию, называемую *расширенными атрибутами*. Каждый расширенный атрибут концептуально подобен переменной окружения. Но самым главным отличием систем FAT и HPFS являются базовые принципы хранения информации о местоположении файлов.

Принципы размещения файлов на диске, положенные в основу HPFS, увеличивают как производительность файловой системы, так и ее надежность и отказоустойчивость. Для достижения этих целей предложено несколько способов:

- размещение каталогов в середине дискового пространства;
- использование методов бинарных сбалансированных деревьев для ускорения поиска информации о файле;
- рассредоточение информации о местоположении записей файлов по всему диску, при том что записи каждого конкретного файла размещаются по возможности в смежных секторах и поблизости от данных об их местоположении.

Действительно, система HPFS стремится, прежде всего, к тому, чтобы расположить файл в смежных кластерах или, если такой возможности нет, разместить

его на диске таким образом, чтобы *экстенты* (фрагменты) файла физически были как можно ближе друг к другу. Такой подход существенно уменьшает *время позиционирования* головок записи/чтения жесткого диска и *время ожидания* (rotational latency), т. е. время задержки между установкой головки чтения/записи на нужную дорожку диска и началом чтения данных с диска. Можно сказать, что файловая система HPFS обладает по сравнению с FAT следующими основными преимуществами:

- высокой производительностью;
- надежностью;
- возможностью работы с расширенными атрибутами, что позволяет управлять доступом к файлам и каталогам;
- возможностью эффективного использования дискового пространства.

Все эти преимущества обусловлены структурой диска HPFS. В начале диска расположено несколько управляющих блоков. Все остальное дисковое пространство в HPFS разбито на части: полосы, ленты из смежных секторов (band). Каждая такая группа данных занимает на диске пространство в 8 Мбайт и имеет свою собственную *битовую карту* распределения секторов, показывающую, какие секторы данной полосы заняты, а какие — свободны. Каждому сектору ленты данных соответствует один бит в ее битовой карте. Если бит имеет значение 1, то соответствующий сектор занят, а если 0 — свободен.

Битовые карты двух полос располагаются на диске рядом, так же располагаются и сами полосы, то есть последовательность полос и карт выглядит следующим образом: битовая карта, битовая карта, лента с данными, лента с данными, битовая карта, битовая карта и т. д. Такое расположение лент позволяет непрерывно разместить на жестком диске файл размером до 16 Мбайт и в то же время не удалять от самих файлов информацию об их местонахождении.

Дисковое пространство в HPFS выделяется не кластерами, как в FAT, а *блоками*. В современной реализации размер блока взят равным одному сектору, но в принципе он мог бы быть и иного размера. По сути дела, блок — это и есть кластер. Размещение файлов в таких небольших блоках позволяет более эффективно использовать пространство диска, так как непроизводительные потери свободного места составляют в среднем всего 256 байт на каждый файл. Вспомните, что чем больше размер кластера, тем больше места на диске расходуется напрасно. Например, кластер на отформатированном под FAT диске объемом от 512 до 1024 Мбайт имеет размер 16 Кбайт. Следовательно, непродуктивные потери свободного пространства на таком разделе в среднем составляют 8 Кбайт (8192 байт) на один файл, в то время как на разделе HPFS эти потери всегда будут составлять всего 256 байт на файл. Таким образом, на каждый файл экономится почти 8 Кбайт.

Помимо лент с записями файлов и битовых карт, в томе с HPFS имеются еще три информационные структуры. Это так называемый *загрузочный блок (boot block), дополнительный блок (super block) и запасной (резервный) блок (spare block)*. Загрузочный блок (boot block) располагается в секторах с 0 по 15; он содержит имя тома, его серийный номер, блок параметров BIOS и программу начальной загрузки. Программа начальной загрузки находит файл OS2LDR, считывает его в память и передает управление этой программе загрузки ОС, которая, в свою очередь, загружает с диска в память ядро OS/2 — OS2KRNL, и уже OS2KRNL с помощью

сведений из файла CONFIG.SYS загружает в память все остальные необходимые программные модули и блоки данных.

В блоке (super block) содержится указатель на список битовых карт (bitmap block list). В этом списке перечислены все блоки на диске, в которых расположены битовые карты, используемые для обнаружения свободных секторов. Также в дополнительном блоке хранится указатель на список дефектных блоков (bad block list), указатель на группу каталогов (directory band), указатель на файловый узел (F-node) корневого каталога, а также дата последней проверки раздела программой CHKDSK. В списке дефектных блоков перечислены все поврежденные сектора (блоки) диска. Когда система обнаруживает поврежденный блок, он вносится в этот список и для хранения информации больше не используется. Кроме этого, в структуре super block содержится информация о размере полосы. В текущей реализации HPFS размер полосы равен 8 Мбайт. Блок super block размещается в секторе с номером 16 логического диска, на котором установлена файловая система HPFS.

Резервный блок (spare block) содержит указатель на карту аварийного замещения (hotfix map или hotfix-areas), указатель на список свободных запасных блоков (directory emergency free block list), используемых для операций на почти переполненном диске, и ряд системных флагов и дескрипторов. Этот блок размещается в 17-ом секторе диска. Резервный блок обеспечивает высокую отказоустойчивость файловой системы HPFS и восстановление поврежденных данных на диске.

Файлы и каталоги в HPFS базируются на фундаментальном объекте, называемом F-Node [13]. Эта структура характерна для HPFS и аналога в файловой системе FAT не имеет. Каждый файл и каталог диска имеет свой файловый узел F-Node. Каждый объект F-Node занимает один сектор и всегда располагается поблизости от своего файла или каталога (обычно непосредственно перед файлом или каталогом). Объект F-Node содержит информацию о длине и первых 15 символах имени файла, специальную служебную информацию, статистику по доступу к файлу, расширенные атрибуты файла и список прав доступа (или только часть этого списка в случае большого размера), ассоциативную информацию о расположении и подчинении файла и т. д. Структура распределения в F-node может принимать несколько форм в зависимости от размера каталога или файлов. HPFS просматривает файл как совокупность одного или более секторов. Из прикладной программы это не видно; файл появляется как непрерывный поток байтов. Если расширенные атрибуты слишком велики для файлового узла, то в него записывается указатель на них.

В HPFS структура каталога представляет собой сбалансированное дерево с записями, расположенными в алфавитном порядке. Каждая запись, входящая в состав двоичного дерева (B-Tree), содержит: атрибуты файла; указатель на соответствующий файловый узел; информацию о времени и дате создания файла, времени и дате последнего обновления и обращения, длине данных, содержащих расширенные атрибуты; счетчик обращений к файлу; информацию о длине имени файла и само имя и другую информацию.

Файловая система HPFS при поиске файла в каталоге просматривает только необходимые ветви двоичного дерева (B-Tree). Такой метод во много раз эффективнее, чем последовательное чтение всех записей в каталоге, что имеет место в системе FAT. Для того чтобы найти искомый файл в каталоге (точнее, указатель на его информационную структуру F-node), организованном на принципах сбалан-

сированных двоичных деревьев, большинство записей вообще читать не нужно. В результате для поиска информации о файле необходимо выполнить существенно меньшее количество операций чтения диска.

Действительно, если, например, каталог содержит 4096 файлов, то файловая система FAT потребует чтения в среднем 64 секторов для поиска нужного файла внутри такого каталога, в то время как HPFS осуществит чтение всего только 2–4 секторов (в среднем) и найдет искомый файл. Несложные расчеты позволяют увидеть явные преимущества HPFS над FAT. Так, например, при использовании 40 входов на блок блоки каталога дерева с двумя уровнями могут содержать 1640 входов, а блоки каталога дерева с тремя уровнями — уже 65 640 входов. Другими словами, некоторый файл может быть найден в типичном каталоге из 65 640 файлов максимум за три обращения. Это намного лучше файловой системы FAT, где для нахождения файла нужно прочитать в худшем случае более 4000 секторов.

Размер каждого из блоков, в терминах которых выделяются каталоги в текущей реализации HPFS, равен 2 Кбайтам. Размер записи, описывающей файл, зависит от размера имени файла. Если имя занимает 13 байтов (для формата 8.3), то блок из 2 Кбайт вмещает до 40 описателей файлов. Блоки связаны друг с другом посредством списковой структуры, как и описатели экстентов, для облегчения последовательного обхода.

При переименовании файлов может возникнуть так называемая перебалансировка дерева. Создание файла, переименование или стирание может приводить к каскадированию блоков каталогов. Фактически, переименование может потерпеть неудачу из-за недостатка дискового пространства, даже если файл непосредственно в размерах не увеличился. Во избежание такой ситуации HPFS поддерживает небольшой пул свободных блоков, которые могут использоваться при «аварии». Эта операция может потребовать выделения дополнительных блоков на заполненном диске. Указатель на пул свободных блоков сохраняется в SpareBlock.

Важное значение для повышения скорости работы с файлами имеет уменьшение их фрагментации. *В HPFS считается, что файл является фрагментированным, если он содержит больше одного экстента*. Снижение фрагментации файлов сокращает время позиционирования и время ожидания за счет уменьшения количества перемещений головок, необходимых для доступа к данным файла. Алгоритмы работы файловой системы HPFS работают таким образом, чтобы по возможности размещать файлы в последовательных смежных секторах диска, что обеспечивает максимально быстрый доступ к данным впоследствии. В системе FAT, наоборот, запись следующей порции данных в первый же свободный кластер неизбежно приводит к фрагментации файлов. В системе HPFS также, если это предоставляется возможным, данные записываются в смежные секторы диска, а не в первый попавшийся. Это позволяет несколько снизить число перемещений головок чтения/записи от дорожки к дорожке. При этом когда данные дописываются в существующий файл, HPFS сразу же резервирует как минимум 4 Кбайта непрерывного пространства на диске. Если же часть этого пространства не потребовалась, то после закрытия файла она высвобождается для дальнейшего использования. Файловая система HPFS равномерно размещает непрерывные файлы по всему диску для того, чтобы впоследствии без фрагментации обеспечить их возможное увеличение. Если же файл не может быть увеличен без нарушения его непрерывности, HPFS вновь

резервирует 4 Кбайта смежных блоков как можно ближе к основной части файла с целью сокращения времени позиционирования головок чтения/записи и времени ожидания соответствующего сектора.

Очевидно, что степень фрагментации файлов на диске зависит как от числа и размеров, расположенных на нем файлов и размеров самого диска, так и от характера и интенсивности самих дисковых операций. Незначительная фрагментация файлов практически не сказывается на быстродействии операций с файлами. Файлы, состоящие из двух-трех экстентов, практически не снижают производительность HPFS, так как эта файловая система следит за тем, чтобы области данных, принадлежащие одному и тому же файлу, располагались как можно ближе друг к другу. Файл из трех экстентов имеет только два нарушения непрерывности, и, следовательно, для его чтения потребуется всего лишь два небольших перемещения головки диска. Программы (утилиты) дефрагментации, имеющиеся для этой файловой системы, по умолчанию считают наличие двух-трех экстентов у файла нормой. Например, программа HPFSOPT из набора утилит Gamma-Tech по умолчанию не дефрагментирует файлы, состоящие из трех и менее экстентов, а файлы, которые имеют большее количество экстентов, по возможности приводятся к 2 или 3 экстентам. Файлы объемом в несколько десятков мегабайт всегда будут фрагментированы, ибо максимально возможный размер экстента равен 8 Мбайтам. Практика показывает, что в среднем на диске имеется не более 2 процентов файлов, имеющих три и более экстентов. Даже общее количество фрагментированных файлов, как правило, не превышает 3%. Такая ничтожная фрагментация оказывает пренебрежимо малое влияние на общую производительность системы. Кратко рассмотрим вопрос надежности хранения данных в HPFS. Любая файловая система должна обладать средствами исправления ошибок, возникающих при записи информации на диск. Система HPFS для этого использует *механизм аварийного замещения* (hotfix) [13].

Если файловая система HPFS сталкивается с проблемой в процессе записи данных на диск, то происходят следующие действия.

1. На экран выводится соответствующее сообщение об ошибке.
2. HPFS сохраняет информацию, которая должна быть записана в дефектный сектор в одном из запасных секторов, заранее зарезервированных на этот случай. Список свободных запасных блоков хранится в резервном блоке HPFS. При обнаружении ошибки во время записи данных в нормальный блок HPFS выбирает один из свободных запасных блоков и сохраняет эти данные в нем.
3. Файловая система обновляет карту аварийного замещения в резервном блоке. Эта карта представляет собой просто пары двойных слов, каждое из которых является 32-битным номером сектора. Первый номер указывает на дефектный сектор, а второй — на тот сектор среди имеющихся запасных секторов, который был выбран для его замены.
4. После замены дефектного сектора запасным карта аварийного замещения записывается на диск, и на экране появляется всплывающее окно, информирующее пользователя о произошедшей ошибке записи на диск. Каждый раз, когда система выполняет запись или чтение сектора диска, она просматривает карту аварийного замещения и подменяет все номера дефект-

ных секторов номерами запасных секторов с соответствующими данными. Следует заметить, что это преобразование номеров существенно не влияет на производительность системы, так как оно выполняется только при физическом обращении к диску, но не при чтении данных из дискового кэша. Очистка карты аварийного замещения автоматически выполняется программой CHKDSK при проверке диска HPFS. Для каждого замещенного блока (сектора) эта программа выделяет новый сектор для файла, которому принадлежат данные, в наиболее подходящем месте жесткого диска.

5. Программа перемещает данные из запасного блока в этот сектор и обновляет информацию о положении файла, что может потребовать новой балансировки дерева блоков размещения. После этого CHKDSK вносит поврежденный сектор в список дефектных блоков, который хранится в дополнительном блоке HPFS, и возвращает освобожденный сектор в список свободных запасных секторов резервного блока.
6. Затем удаляет запись из карты аварийного замещения и записывает отредактированную карту на диск.

Все основные файловые объекты в HPFS, в том числе файловые узлы, блоки размещения и блоки каталогов, имеют уникальные 32-битные идентификаторы и указатели на свои родительские и дочерние блоки. Файловые узлы, кроме того, содержат сокращенное имя своего файла или каталога. Избыточность и взаимосвязь файловых структур HPFS позволяют программе CHKDSK полностью восстанавливать файловую структуру диска, последовательно анализируя все файловые узлы, блоки размещения и блоки каталогов. Руководствуясь собранной информацией, CHKDSK реконструирует файлы и каталоги, а затем заново создает битовые карты свободных секторов диска. Запуск программы CHKDSK следует осуществлять с соответствующими ключами. Так, например, один из вариантов работы этой программы позволяет найти и восстановить удаленные файлы. *HPFS относится к так называемым монтируемым файловым системам. Это означает, что она не встроена в операционную систему, а добавляется к ней при необходимости*.

3.3.4 Файловая система ОС UNIX

Файловая система UNIX имеет иерархическую структуру каталогов и файлов, включая корневой каталог. Файловая система располагается на устройстве, которое обычно является магнитным диском того или иного типа. Если диск достаточно велик, он может быть разбит на несколько логических дисков; тогда на каждом логическом диске может быть размещена отдельная файловая система. Каждая файловая система, прежде чем стать доступной, должна быть смонтирована.

Рассмотрим одну из *ранних реализаций файловой системы UNIX*, основные идеи которой сохранились до сих пор. Каждая файловая система имеет четыре основные части:

1) загрузочный блок — это первый блок диска (блок 0), зарезервированный для системной загрузочной программы;
2) суперблок — это первый блок собственно файловой системы (блок 1), содержащий основные данные о файловой системе и ее размещении на диске, в том числе о списках свободных i-узлов и блоков;

3) i-узлы — это последовательность блоков, следующих за суперблоком. I-узел содержит ссылки на блоки. Имеется ровно один i-узел для каждого каталога или файла в файловой системе;
4) блоки — это блоки, которые занимают оставшееся пространство диска и содержат либо действительные данные каталогов и файлов (блоки данных), либо ссылки на блоки (косвенные блоки).

Суперблок содержит следующие данные:

- размер дискового пространства, доступного файловой системе (в блоках);
- число блоков, зарегистрированных для i-узлов;
- имя файловой системы;
- имя тома;
- время последнего изменения;
- время последнего копирования (back up);
- ссылку на список свободных блоков;
- ссылку на список свободных i-узлов.

Структура файловой системы UNIX представлена на рисунке 3.3.

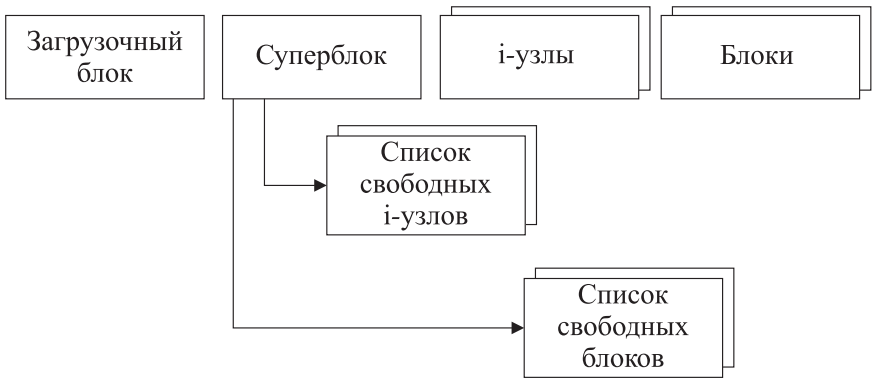


Рис. 3.3 – Структура файловой системы UNIX

Каждый файл и каждый каталог в файловой системе представлен i-узлом, содержащим указатели на блоки, составляющие файл.

В i-узле содержится также информация о правах доступа к файлу, число ссылок на файл из каталогов и другие данные. Каждый i-узел содержит 13 указателей.

Первые 10 указателей непосредственно ссылаются на блоки данных файла. Поскольку блок содержит 512 байтов, то этого достаточно для обработки файлов до 5120 байтов (512×10).

Если длина файла больше 5120 байт, используется 11-й указатель i-узла, который ссылается на косвенный блок из 128 ссылок на блоки данных. Использование косвенного блока позволяет увеличить длину файла до величины 70 656 байт $(512 \times (10 + 128))$.

Если и этого недостаточно, то используется 12-й указатель i-узла, ссылающийся на дважды косвенный блок, содержащий 128 ссылок на косвенные блоки. Тогда максимальный размер файла увеличивается до величины 8 459 264 байта $(512 \times (10 + 128^2))$. Наконец, использование последнего 13-го указателя на трижды

косвенный блок из 128 ссылок на дважды косвенные блоки дает предельную длину в файловой системе — 1 082 201 088 байтов ($512 \times (10 + 128 + 128^2 + 128^3)$). Другие версии системы UNIX могут отличаться количеством ссылок в i-узле, косвенных блоках и размером блока данных.

Когда система загружается, имеется только одна из файловых систем, называемая корневой. В ней находятся все важнейшие каталоги (/dev, /etc, /bin и т. п.). Все остальные файловые системы должны быть созданы и смонтированы.

Команда mkfs создает новую файловую систему. Она расположена в каталоге /etc и имеет два параметра:

```
/etc/mkfs <имя> <размер>
```

Первый параметр является именем специального файла и указывает устройство, на котором создается файловая система.

Второй параметр — размер пространства файловой системы в блоках — используется для определения по некоторым правилам числа блоков после того, как размещены i-узлы.

Пример создания файловой системы на флоппи-диске:

```
/etc/mkfs /dev/flo 2000
isize = 230
```

Ответное сообщение указывает число блоков, выделенных для размещения i-узлов.

Монтирование файловой системы UNIX для какой-либо ОС осуществляется посредством команды mount. Эта команда подключает корневой каталог монтируемой файловой системы в один из каталогов корневой файловой системы. Команда расположена в каталоге /etc и имеет два параметра:

```
/etc/mount <устройство> <каталог>
```

Первый параметр является именем спецфайла для монтируемого логического устройства, содержащего подключаемую файловую систему. Второй — имя уже существующего каталога, под которым монтируется файловая система.

Чтобы выяснить, какие файловые системы смонтированы в данный момент, надо выполнить *команду mount* без параметров:

```
mount
/dev/fl0 on /floppy0
```

Ответом является сообщение об этих системах (в данном случае об одной системе). Оно формируется на основе данных о монтаже файловых систем, хранимых в файле /etc/mnttab.

Права доступа корневого каталога монтируемой файловой системы и каталога, под которым производится монтаж, должны быть одинаковыми во избежание ошибок операционной системы.

Если файловая система на съемном устройстве больше не используется, ее можно демонтировать *командой umount*, расположенной в каталоге /etc и имеющей один параметр:

```
umount <устройство>
```

Например, демонтаж файловой системы на гибком диске из предыдущего примера выполняется командой:

```
umount /dev/fl0
```

Результатом демонтажа является разрыв связи между корневым каталогом демонтируемой файловой системы и каталогом корневой файловой системы, в котором производился монтаж. При выполнении команды демонтажа текущий каталог должен быть вне демонтируемой файловой системы, иначе будет выдано сообщение о том, что устройство занято (`umount : device busy`) и команда не будет выполнена.

Структура файловой системы, описанная выше в терминах i-узлов, блоков, косвенных блоков и суперблока, может быть нарушена. В этом случае возникает необходимость в ее восстановлении. При разрушении информации в трижды косвенном блоке могут появиться следующие проблемы:

- блок может оказаться вне системы, т. е. перестать быть частью файла, и отсутствовать в списке свободных блоков;
- могут появиться дубли i-узлов, описывающие один и тот же файл дважды;
- могут возникнуть ситуации, когда некоторый блок является частью файла и одновременно присутствует в списке свободных блоков;
- могут существовать некоторые файлы, которые не включены ни в один каталог.

Эти проблемы могут быть ликвидированы, поскольку структура файловой системы обладает некоторой избыточностью (наличием дополнительной информации), позволяющей восстанавливать отдельные поломки. Вот некоторые виды избыточности:

- блок данных, являющийся каталогом, содержит имена файлов и номера i-узлов; где-то имеется i-узел, соответствующий этому каталогу, и этот i-узел должен быть каталогом, а не обычным файлом;
- блок, включенный в список свободных блоков, теоретически не может быть частью какого-либо файла; для проверки этого достаточно сканировать все i-узлы для просмотра всех блоков, занятых файлами, и сканировать список свободных блоков;
- аналогично, блок, принадлежащий файлу, должен принадлежать только одному файлу; это легко проверить.

Эти и другие виды избыточности использует программа проверки файловой системы, запускаемая *командой fsck* (file system check). В различных реализациях существуют разные команды проверки целостности файловой системы: icheck, dcheck, ncheck. Однако все они в большей или меньшей степени перекрываются командой fsck.

Команда fsck выполняется в несколько фаз, на которых производится следующая работа:

- проверка целостности i-узлов (счетчик связи, тип и формат i-узла);
- проверка каталогов, указывающих на i-узлы, содержащие ошибки;
- проверка каталогов, на которые нет ссылок;
- проверка счетчиков связей в каталогах и файлах;

- проверка неверных блоков и дублированных блоков в списке свободных блоков; неиспользуемых блоков, которые должны быть включены, но не являются таковыми, в список свободных блоков; счетчика общего числа свободных блоков.

Команда по умолчанию всегда проверяет корневую файловую систему: все другие файловые системы проверяются, если их имена занесены в файл /etc/checklist.

После выполнения fsck, связанного с «починкой» файловой системы, может появиться сообщение `***** BOOT UNIX (NO SYNC!) *****`, требующее перезагрузки системы без выполнения команды sync.

Если этого не сделать, работа по восстановлению списка свободных блоков будет утрачена, так как копии управляющих таблиц и буфера в оперативной памяти остались старыми. Для их обновления требуется перезагрузка без выгрузки буферов на диск командой sync.

Необходимым условием правильной работы fsck является также наличие пустого каталога /lost+found в корневом каталоге.

Если при выполнении fsck будут найдены каталоги, на которые никто не ссылается в проверяемой файловой системе, они будут подключены в каталог /lost+found для дальнейшего изучения их принадлежности.

3.3.5 Файловые системы для CD-ROM

Файловая система CDFS. В Windows 2000 обеспечивается поддержка файловой системы CDFS (Compact Disk File System), отвечающей стандарту ISO 9660, описывающему расположение информации на CD-ROM. Поддерживаются длинные имена файлов в соответствии с ISO 9660 Level 2.

При создании CD-ROM под управлением Windows 2000 следует иметь в виду следующее:

- все имена каталогов и файлов должны содержать менее 32 символов;
- все имена каталогов и файлов должны состоять только из символов верхнего регистра;
- глубина каталогов не должна превышать 8 уровней от корня;
- использование расширений имен файлов не обязательно.

Файловая система UDF. UDF (Universal Disk Format — универсальный дисковый формат) — спецификация формата файловой системы, независимой от операционной системы (ОС) для хранения файлов на оптических носителях (оптических дисках). UDF является реализацией стандарта ISO/IEC 13346 (известного также, как ECMA-167).

Формат UDF призван заменить ISO 9660. ISO 9660 имеет некоторые ограничения, которые делают его несовместимым с DVD, CD-RW и другими новыми форматами дисков. UDF разработан так, чтобы избавить от этих ограничений. UDF позволяет дозаписывать файлы на CD-R- или CD-RW-дисках, один файл одновременно, без существенных потерь дискового пространства, используя метод пакетной записи. Также UDF учитывает возможность выборочного стирания некоторых файлов на перезаписываемых носителях CD-RW, освобождая место на диске. В стандарте ISO 9660 такое не предусмотрено. UDF также лучше подходит для

DVD, так как имеет лучшую поддержку для дисков большого объёма — в частности, ISO 9660 не поддерживает файлы размером более 2 Гб.

UDF разработан и развивается Optical Storage Technology Association (OSTA). Существует несколько версий формата UDF:

- 1.02 (поддерживается Windows 98, многими версиями ОС корпорации Apple, возможно использовать для DVD-RAM и магнитооптических дисков);
- 1.50 (поддерживается Windows 2000, Windows XP и Windows Server 2003; может быть несовместим с Windows 98 или Apple);
- 2.01 (поддерживается Windows XP и Windows Server 2003; может быть несовместим с Windows 98, Windows 2000 или Apple);
- 2.50 (поддерживается Windows Vista; может быть несовместим с более ранними версиями Windows и др. платформами);
- 2.60 (поддерживается Windows Vista, Mac OS X 10.5, NetBSD).

3.4 Управление устройствами ввода-вывода и файловыми системами в ОС Windows

3.4.1 Диспетчер устройств и драйвера устройств

Задача системы ввода-вывода ОС Windows заключается в предоставлении основных средств (каркаса) для эффективного управления широким спектром устройств ввода-вывода. Основу этих средств образует набор независимых от устройств процедур для определенных аспектов ввода-вывода и набор загруженных драйверов для общения с устройствами. Формирует этот каркас *менеджер ввода-вывода*, который предоставляет остальной операционной системе независимый от устройств ввод-вывод, вызывая для выполнения физического ввода-вывода соответствующий драйвер. Файловые системы формально являются драйверами устройств, работающих под управлением менеджера ввода-вывода [16].

В операционной системе Windows имеется программа «*Диспетчер устройств*», которую используют для обновления драйверов (или программного обеспечения) оборудования, изменения настроек оборудования, а также для устранения неполадок. В окне Диспетчера устройств представлено графическое отображение оборудования, установленного на компьютер. Для открытия окна Диспетчера устройств нужно щелкнуть правой клавишей мыши по значку «Мой компьютер» и выбрать в контекстном меню строку «Свойства». В открывшемся окне «Свойства системы» следует перейти на вкладку «Оборудование» и нажать кнопку Диспетчер устройств.

В окне Диспетчера устройств (рис. 3.4) можно, раскрывая соответствующие узлы, видеть устройства, которые либо подключены и работают, либо отключены. Диспетчер устройств обычно используется для проверки состояния оборудования, подключения-отключения оборудования и обновления драйверов устройств, установленных на компьютере.

В ходе процесса настройки Windows назначает устанавливаемому устройству уникальный набор системных ресурсов. Эти ресурсы могут включать в себя один или несколько из следующих параметров:

- номера строк запросов на прерывание (IRQ);
- каналы прямого доступа к памяти (DMA);
- адреса портов ввода/вывода (I/O);
- диапазоны адресов памяти.

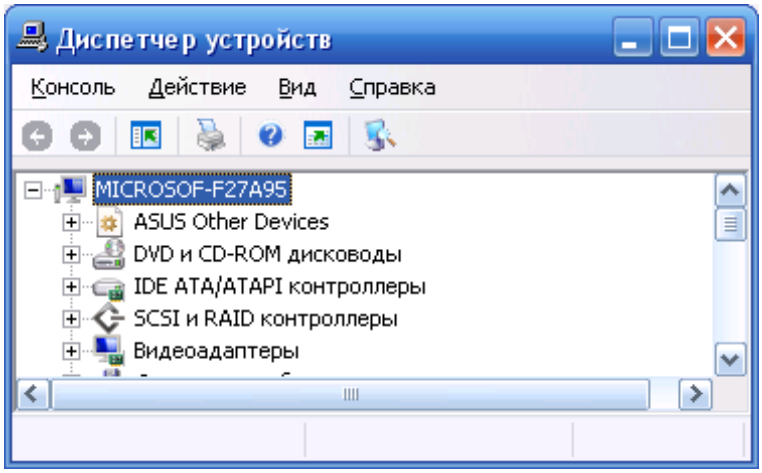


Рис. 3.4 – Окно программы «Диспетчер устройств»

Используя Диспетчер устройств, можно отключать подсоединенные к компьютеру устройства и удалять их из конфигурации компьютера.

Не обязательно удалять устройство, которое требуется отключить, не отсоединяя от компьютера. Не отменяя установку самонастраиваемого устройства, его можно просто отключить. При отключении такого устройства оно физически остается подключенным к компьютеру, но Windows обновляет системный реестр таким образом, что драйверы отключенного устройства не загружаются при запуске компьютера. При включении устройства драйверы снова становятся доступными. Эта возможность полезна при необходимости переключения между двумя устройствами, например сетевым адаптером и модемом, или при устранении неполадок в оборудовании.

3.4.2 Диски и файловая система

Для получения доступа к просмотру состояния и управлению дисками нужно щелкнуть правой клавишей мыши по значку «Мой компьютер», выбрать строку «Управление» и щелкнуть по ней. В открывшемся окне щелкнуть по строке «Управление дисками» (рис. 3.5). В правой части окна будут отображены все дисковые устройства компьютера и основные параметры их состояния.

В окне можно управлять разделами дисковых устройств. Можно создать или удалить раздел или логический диск, можно сделать первичный раздел активным, чтобы при перезагрузке операционной системы обращение к загрузочной записи осуществлялось с указанного раздела. Активный раздел может быть только один. Здесь же можно отформатировать диск и изменить букву или путь диска. Все эти действия вызываются щелчком правой кнопки мыши по выбранному разделу в окне, представленном на рисунке 3.6.

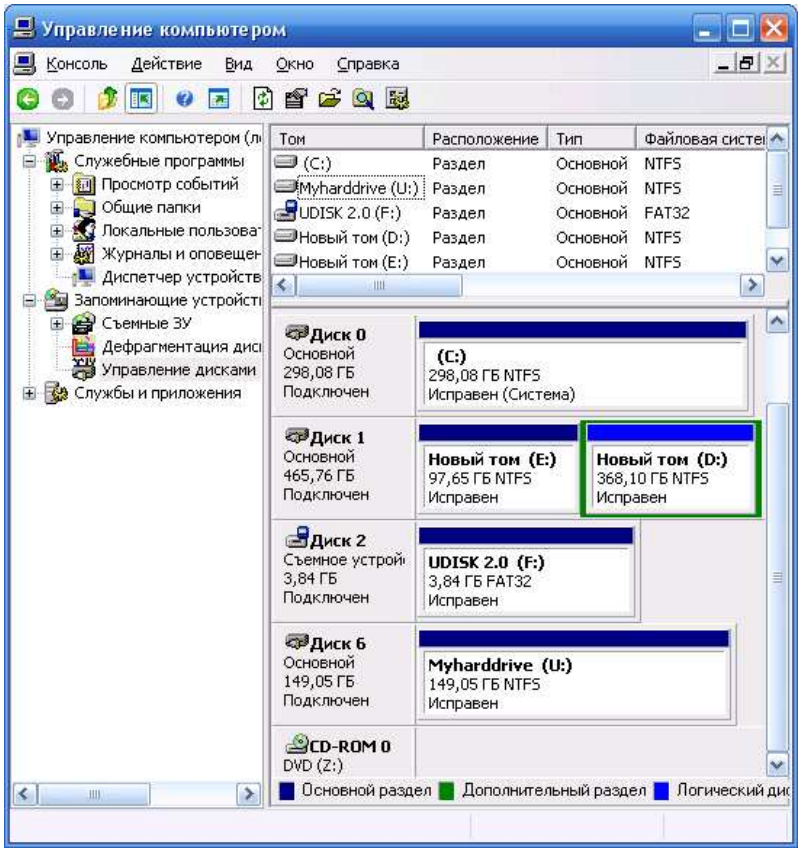


Рис. 3.5 – Вид окна «Управление компьютером» на вкладке «Управление дисками»

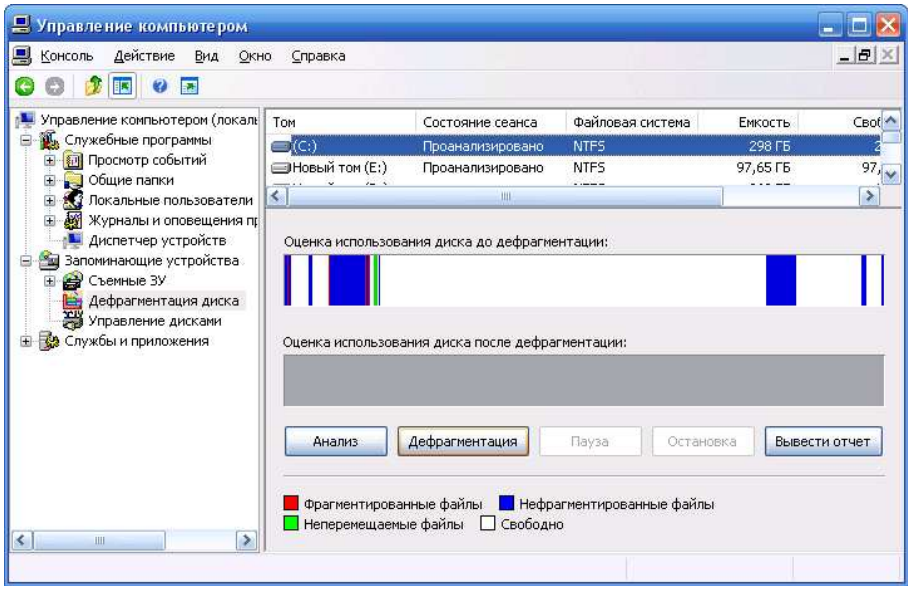


Рис. 3.6 – Вид окна «Управление компьютером» на вкладке «Дефрагментация диска»

При работе с жестким диском всегда имеет место фрагментация. С течением времени после установки программ диск заполняется, а после их удаления файлы

фрагментируются, и операционной системе приходится искать свободные фрагменты на диске для размещения файлов. Это может привести к заметному снижению быстродействия компьютера. Негативный эффект фрагментации устраняется с помощью встроенной в Windows программы дефрагментации, запустить которую можно, указав предварительно имя диска, в левой панели оснастки «Управление компьютером».

Результаты дефрагментации можно просмотреть, нажав на кнопку «Вывести отчет», которая становится доступной после завершения дефрагментации.

3.4.3 Дисковые квоты

При совместном использовании дисковой памяти несколькими пользователями, работающими на одном компьютере, необходим контроль расходования дискового пространства. В Windows на платформе NT эта проблема решается квотированием дискового пространства по каждому тому (независимо от количества физических дисков) и для каждого пользователя.

После установки квот дискового пространства пользователь сможет хранить на томе ограниченный объем данных, в то время как на этом томе может оставаться свободное пространство. Если пользователь превышает выданную ему квоту, в журнал событий вносится соответствующая запись. Затем, в зависимости от конфигурации системы, пользователь либо сможет записать информацию на том (более мягкий режим), либо ему будет отказано в записи.

Устанавливать и просматривать квоты на диске можно только в разделе NTFS 5.0 и при наличии необходимых полномочий (задаваемых с помощью локальных или доменных групповых политик) у пользователя, устанавливающего квоты.

Чтобы установить квоты, нужно выполнить следующие действия.

1. Щелкнуть правой кнопкой мыши по конфигурируемому тому и выбрать в контекстном меню команду «Свойства». В появившемся окне перейти на вкладку «Квота» (рис. 3.7).
2. Установить флажок «Включить управление квотами». В этом случае будет установлен мягкий режим контроля используемого дискового пространства. Для задания жесткого режима контроля нужно установить флажок «Не выделять место на диске при превышении квоты». На этой же вкладке устанавливается размер выделяемой квоты и порог, превышение которого вызовет запись предупреждений в журнале событий.

Чтобы узнать, какие пользователи превысили выделенную им квоту (в мягком режиме), нужно нажать кнопку «Записи квот», где будет отражен список пользователей с параметрами квот и объемом используемого ими пространства диска.

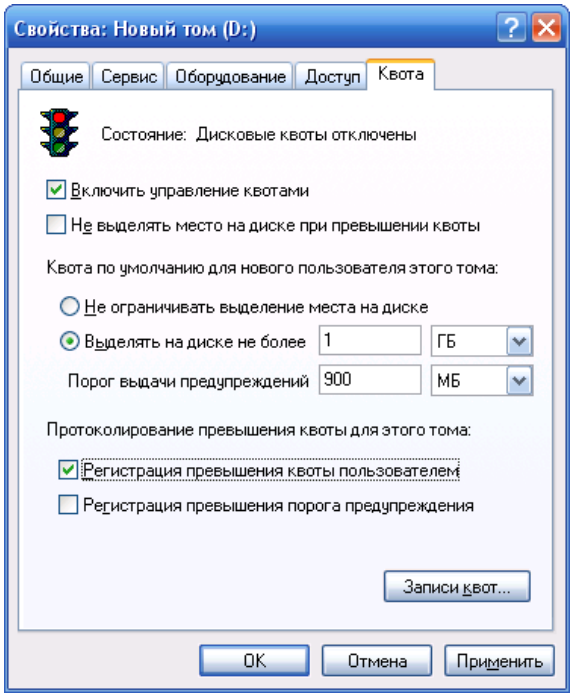


Рис. 3.7 – Вид окна по просмотру свойств диска на вкладке «Квота»

3.4.4 Обеспечение надежности хранения данных на дисковых накопителях с файловой системой NTFS 5.0

Устанавливая пользователям определенные *разрешения для файлов и каталогов (папок)*, администраторы системы могут защищать конфиденциальную информацию от несанкционированного доступа[1]. Каждый пользователь имеет определенный набор разрешений на доступ к конкретному объекту файловой системы (рис. 3.8). Администратор может назначить себя владельцем любого объекта файловой системы.

Действующие разрешения в отношении конкретного файла или каталога образуются из всех прямых и косвенных разрешений, назначенных пользователю для данного объекта с помощью логической функции ИЛИ.

Пользователь может назначить себя владельцем какого-либо объекта файловой системы, если у него есть необходимые права, а также передать права владельца другому пользователю.

Точки соединения (аналог монтирования в UNIX) позволяют отображать целевую папку (диск) в пустую папку, находящуюся в пространстве имен файловой системы NTFS 5.0 локального компьютера. Целевой папкой может служить любой допустимый путь Windows 2000[2] или выше. Точки соединений прозрачны для приложений, это означает, что приложение или пользователь, осуществляющий доступ к локальной папке NTFS, автоматически перенаправляется к другой папке.

[1] Для практического исследования данных возможностей необходимо использовать Windows Server на платформе 2000 или выше.

[2] Поддерживаются только в NTFS 5.0.

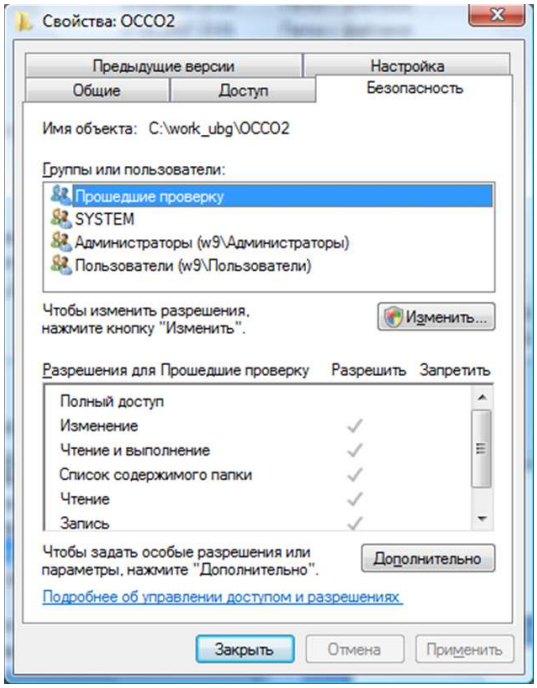


Рис. 3.8 – Вид окна установки разрешений на доступ к конкретному объекту файловой системы

Для работы с точками соединения на уровне томов можно использовать стандартные средства системы — *утилиту* Mountvol (рис. 3.9) и оснастку «Управление дисками». Для монтирования папок нужна утилита Linkd (из Windows 2000 Resource Kit).

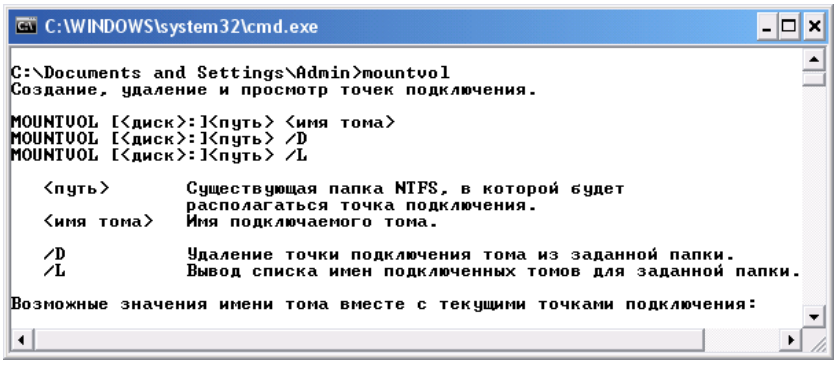


Рис. 3.9 – Вызов утилиты Mountvol

С помощью утилиты Mountvol можно выполнить следующие действия:

- отобразить корневую папку локального тома в некоторую целевую папку NTFS, т. е. подключить или монтировать том;
- вывести на экран информацию о целевой папке точки соединения NTFS, использованной при подключении тома;
- просмотреть список доступных для использования томов файловой системы;
- уничтожить точки подключения томов.

Оснастка «Управление дисками» позволяет также создать соединения для дисков компьютера.

Шифрующая файловая система EFS (Encrypting File System). Поскольку шифрование и дешифрование выполняются автоматически, пользователь может работать с файлом так же, как и до установки его криптозащиты. Все остальные пользователи, которые попытаются получить доступ к зашифрованному файлу, получат сообщение об ошибке доступа, поскольку они не владеют необходимым личным ключом, позволяющим им расшифровать файл [16].

Шифрование информации задается в окне свойств файла или папки. В окне свойств файла на вкладке «Общие» нужно нажать кнопку «Другие». Появится окно диалога «Дополнительные атрибуты». В группе «Атрибуты сжатия и шифрования» необходимо установить флажок «Шифровать содержимое для защиты данных».

При шифровании папки можно указать следующие режимы применения нового атрибута: «Только к этой папке» или «К этой папке и ко всем вложенным папкам и файлам». Для дешифрования файла или папки на вкладке «Общие» окна свойств соответствующего объекта нажать кнопку «Другие» и в открывшемся окне сбросить флажок «Шифровать содержимое для защиты данных».

В процессе шифрования файлов и папок система EFS формирует специальные атрибуты (Data Decryption Field — Поле дешифрования данных), содержащие список зашифрованных ключей *(FEK — File Encryption Key)*, что позволяет организовать доступ к файлу со стороны нескольких пользователей. Для шифрования набора FEK используется открытая часть пары ключей каждого пользователя. Информация, требуемая для дешифрования, привязывается к самому файлу. Секретная часть ключа пользователя используется при дешифровании FEK. Она хранится в безопасном месте, например на смарт-карте или устройстве высокой степени защищенности [16]. FEK применяется для создания ключей восстановления, которые хранятся в другом специальном атрибуте — *DRF (Data Recovery Field — Поле восстановления данных)*. Сама процедура восстановления выполняется довольно редко (при уходе пользователя из организации или забывании секретной части ключа) [16].

Система EFS имеет встроенные средства восстановления зашифрованных данных в условиях, когда неизвестен личный ключ пользователя. Пользователи, которые могут восстанавливать зашифрованные данные в условиях утраты личного ключа, называются агентами восстановления данных. Они обладают сертификатом (X.509 v3) на восстановление данных и личным ключом, с помощью которого выполняется операция восстановления зашифрованных данных [16].

Контрольные вопросы по главе 3

1. Приведите классификацию устройств ввода-вывода.
2. Опишите основные характеристики устройств внешней памяти.
3. Опишите основные характеристики накопителей на жестком магнитном диске.
4. Чем отличается физическая организация магнитного диска от логической?
5. Приведите достоинства и недостатки различных файловых систем.
6. Как в ОС Windows на платформе NT можно управлять дисками и файловыми системами?
7. Какие преимущества в ОС Windows на платформе NT дает использование файловой системы NTFS пятой версии?

Глава 4

ПРИНЦИПЫ ПОСТРОЕНИЯ ВЫЧИСЛИТЕЛЬНЫХ СЕТЕЙ И ТЕЛЕКОММУНИКАЦИЙ

4.1 Сетевая модель OSI

Международной организацией по стандартизации ISO (International Standards Organization) выпущен наиболее известный стандарт в области телекоммуникаций — *сетевая модель OSI (Open Systems Interconnection Basic Reference Model)* [18] — семиуровневая модель взаимодействия открытых систем.

Сетевая модель OSI определяет основные задачи, которые необходимо решать для организации сетевых коммуникаций. В таблице 4.1 представлена модель OSI в виде семи уровней. Каждый уровень определяет свои сетевые задачи и взаимодействует с уровнями выше и ниже. Например, уровень 7 предоставляет услуги программам для получения доступа к сети, а уровни 1 и 2 определяют физическую среду сети и связанные с ней задачи.

Таблица 4.1 – Модель OSI

Тип данных	**Уровень**	**Функции**
Данные	7. Прикладной	Доступ к сетевым службам
Поток	6. Представлений	Представление и шифрование данных
Сеансы	5. Сеансовый	Управление сеансом связи
Сегменты	4. Транспортный	Прямая связь между конечными пунктами и надежность
Пакеты/Дата-граммы	3. Сетевой	Определение маршрута и логическая адресация
продолжение на следующей странице		

Таблица 4.1 — Продолжение

Тип данных	Уровень	Функции
Кадры	2. Канальный	Физическая адресация
Биты	1. Физический	Работа со средой передачи, сигналами и двоичными данными

На каждом уровне можно выделить операнд (логически неделимый элемент данных), которым можно оперировать в рамках модели и используемых протоколов: на физическом уровне — бит, на канальном уровне — кадр, на сетевом — пакет (датаграмма), на транспортном — сегмент. Фрагмент данных, логически объединённый для передачи (кадр, пакет, датаграмма), считается сообщением.

Ценность модели OSI заключается в том, что эта сетевая модель служит базовой точкой при обсуждении различных сетевых протоколов и устройств. Модель OSI является основой для обсуждения сетевого взаимодействия, которая используется для описания устройств и функций протоколов, например протокол Ethernet, стек протоколов TCP/IP[1], протокол Hypertext Transfer Protocol (HTTP). Наличие сетевых стандартов позволяет различным поставщикам выпускать совместимые продукты и снижает затраты на производство.

Существует еще ряд организаций, определяющий сетевые стандарты:

- *IETF* (Internet Engineering Task Force — комитет по проектированию Интернета) — *RFC* (Request for Comments — запросы на комментарии в Интернете), протоколы TCP/IP;
- *W3C* (World Wide Web Consortium — консорциум Всемирной паутины) — развитие Всемирной сети (HTML, HTTP);
- *IEEE* (Institute of Electrical and Electronics Engineers[2] — Институт инженеров по электротехнике и электронике) — технология Ethernet 802.3, беспроводная технология Wi-Fi 802.11;
- *ITU* (International Telecommunication Union — Международный союз электросвязи) — H.323 (система видеоконференций), модемы (V.92).

4.2 Физическая инфраструктура сети

4.2.1 Перечень компонентов сети

На рисунке 4.1 представлен пример структуры сети, из которого можно выделить следующие типичные компоненты сети:

1) рабочие станции;
2) серверы;
3) среда («кабельная» система);
4) коммутатор;
5) маршрутизатор;
6) межсетевой экран.

[1] TCP — Transmission Control Protocol и IP — Internet Protocol.

[2] Принято читать как «I triple E» (рус. «Ай трипл и»).

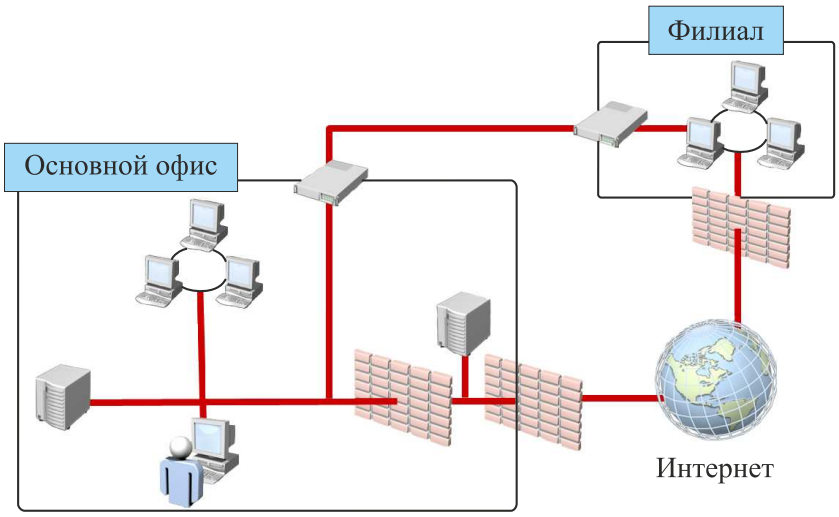
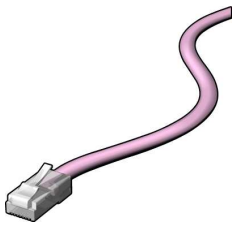


Рис. 4.1 – Пример структуры сети

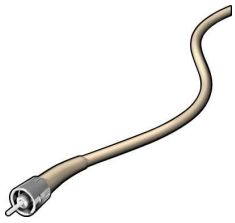
Практически все сети имеют: 1) рабочие станции, 2) серверы, 3) «кабельную» систему и 4) коммутаторы. Существуют одноранговые сети, в них может и не быть выделенного сервера или рабочая станция будет выполнять роль сервера. Маршрутизаторы обычно используют в больших сетях с многочисленными филиалами или с большим количеством компьютеров. Межсетевые экраны используют в любой сети, имеющей подключение к Интернету. Рассмотрим компоненты сети, начиная с третьего пункта списка, более подробно.

4.2.2 «Кабельная» система

На физическом уровне модели OSI используется такой компонент сети, как *среда* (*«кабельная» система*). В современных сетях выделяют три типа «кабельной» системы: витая пара, оптоволоконный кабель, беспроводная связь.



1. Витая пара обычно используется для локальных сетей (*LANs — Local Area Networks*). Представляет собой вид кабеля связи, имеющий одну или несколько пар изолированных проводников, скрученных между собой (с небольшим числом витков на единицу длины), покрытых пластиковой оболочкой. Скручивание проводов уменьшает электромагнитные наводки. Существует несколько категорий кабеля витая пара, которые нумеруются от CAT1 до CAT7. Кабель более высокой категории обычно содержит больше пар проводов, каждая пара имеет больше витков на единицу длины, увеличенную полосу частот и скорость передачи данных.



2. Оптоволоконный кабель используется на больших расстояниях, чем витая пара. Представляет собой нить из оптически прозрачного материала (стекло, пластик), используемую для переноса света внутри себя посредством полного внутреннего отражения. Оптические волокна делятся на две категории: одномодовые и многомодовые.

Одномодовые, имеющие диаметр сердцевины волокон от 7 до 10 микрон. Благодаря малому диаметру достигает-

ся передача по волокну лишь одной моды электромагнитного излучения, за счёт чего исключается влияние дисперсионных искажений.

Многомодовые волокна отличаются от одномодовых диаметром сердцевины, который составляет 50 микрон в европейском стандарте и 62,5 микрон в североамериканском и японском стандартах. Из-за большого диаметра сердцевины по многомодовому волокну распространяется несколько мод излучения — каждая под своим углом, из-за чего импульс света испытывает дисперсионные искажения и из прямоугольного превращается в колоколоподобный.

3. Беспроводная связь (WLAN — Wireless Local Area Network) — используется для мобильности и имеет больший радиус действия, чем витая пара. При таком способе построения сетей передача данных осуществляется через радиоэфир и объединение устройств в сеть происходит без использования кабельных соединений. Существует набор стандартов связи IEEE 802.11 для коммуникации в беспроводной связи в зоне частотных диапазонов 0.9, 2.4, 3.6 и 5 ГГц.

4.2.3 Коммутатор

Следующий компонент сети — это коммутатор (рис. 4.2).

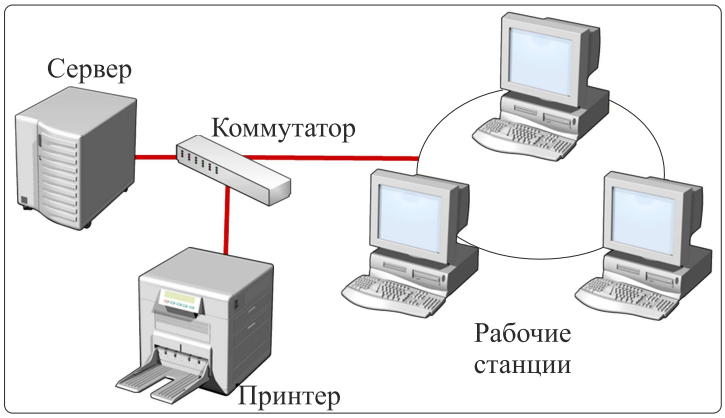


Рис. 4.2 – Пример использования коммутатора

Коммутатор *— многозначный термин, но применительно к компьютерным сетям, это устройство для соединения нескольких узлов или сегментов вычислительной сети.*

Коммутатор относится к устройствам *второго (канального) уровня модели OSI*. Сетевой коммутатор (switch) пришел на смену сетевому концентратору (hub). Концентратор работает на *первом (физическом) уровне сетевой модели OSI*, ретранслируя входящий сигнал с одного из портов в сигнал на все остальные (подключённые) порты, реализуя, таким образом, свойственную сети Ethernet топологию «общая шина», с разделением пропускной способности сети между всеми устройствами.

Коммутатор хранит в памяти таблицу коммутации, в которой указывается соответствие MAC-адреса[1] узла порту коммутатора. При включении коммутатор начинает работать в режиме обучения, заполняя таблицу коммутации (первоначально поступающие данные передаются на все порты коммутатора, коммутатор анализирует фреймы (кадры), и определив MAC-адрес отправителя, заносит его в таблицу). Со временем коммутатор строит таблицу для всех активных MAC-адресов. Впоследствии при поступлении кадра коммутатор отправляет данные только через ассоциированный порт.

Коммутаторы подразделяют на управляемые и неуправляемые. Более сложные управляемые коммутаторы могут управлять коммутацией на сетевом (третьем) уровне, а иногда и на транспортном (четвертом) уровне модели OSI. Управление коммутатором может осуществляться посредством веб-интерфейса, интерфейса командной строки, различных протоколов и т. п.

Многие управляемые коммутаторы позволяют настраивать дополнительные функции и обладают расширенными возможностями: сегментацией трафика между портами, контролем трафика на предмет штормов, обнаружением петель, ограничением количества изучаемых MAC-адресов, ограничением входящей/исходящей скорости на портах, функциями списков доступа и т. п. С целью увеличения числа портов сложные коммутаторы могут объединять в одно логическое устройство — стек. Также с помощью коммутаторов могут быть организованы логические (виртуальные) локальные компьютерные сети VLAN — Virtual Local Area Network. VLAN выглядит так же, как и физическая локальная сеть, но позволяет конечным станциям группироваться вместе, даже если они не находятся в одной физической сети.

4.2.4 Маршрутизатор

Маршрутизатор (шлюз, роутер) *— это специализированный сетевой компьютер, имеющий как минимум один сетевой интерфейс, который передает пакеты между сетями и отслеживает сети, а не компьютеры (рис. 4.3).*

Маршрутизатор часто устанавливают между офисами, и он работает на более высоком *третьем (сетевом) уровне* сетевой модели OSI, чем коммутатор.

Обычно маршрутизатор использует адрес получателя, указанный в пакетных данных, и определяет по таблице маршрутизации путь, по которому следует передать данные. Если в таблице маршрутизации для адреса нет описанного маршрута, пакет отбрасывается.

Таблица маршрутизации содержит информацию, на основе которой маршрутизатор принимает решение о дальнейшей пересылке пакетов. Таблица состоит из некоторого числа записей — маршрутов, в каждой из которых содержится адрес

[1] MAC-адрес (Media Access Control — управление доступом к среде, также Hardware Address) — это уникальный идентификатор, присваиваемый каждой единице активного оборудования компьютерных сетей Ethernet.

сети получателя, адрес следующего узла, которому следует передавать пакеты, административное расстояние — степень доверия к источнику маршрута и некоторый вес записи — метрика. Метрики записей в таблице играют роль в вычислении кратчайших маршрутов к различным получателям. В зависимости от модели маршрутизатора и используемых протоколов маршрутизации в таблице может содержаться некоторая дополнительная служебная информация.

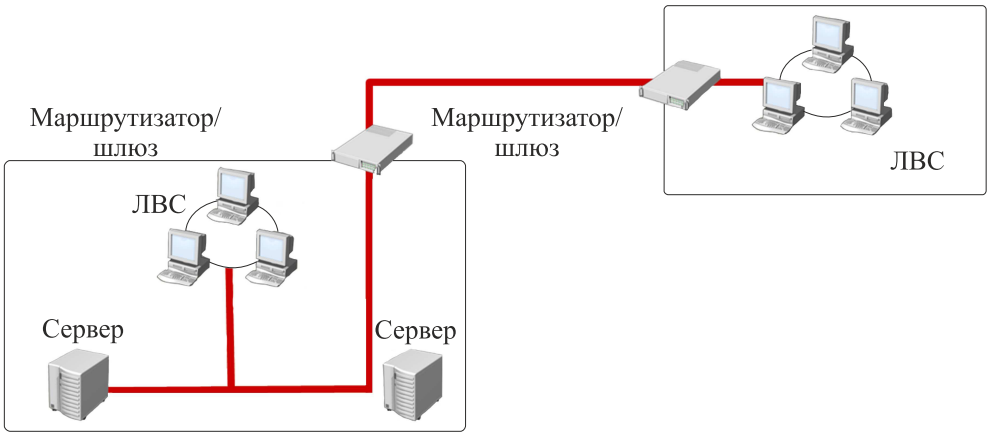


Рис. 4.3 – Пример использования маршрутизатора

В основном маршрутизаторы применяют для объединения сетей разных типов. Нередко маршрутизатор используется для обеспечения доступа из локальной сети в глобальную сеть Интернет, осуществляя функции трансляции адресов и межсетевого экрана. Например, маршрутизатор может обеспечивать подключение домашней сети компьютеров к каналу связи провайдера Интернета. Современный маршрутизатор имеет ряд вспомогательных функций и встроенных возможностей: точка доступа Wi-Fi для подключения мобильных устройств, межсетевой экран для защиты сети от внешних атак, резервирование Интернета от нескольких провайдеров, веб-интерфейс для упрощения настройки устройства, USB-порт для подключения принтера или дискового хранилища и т. п.

В качестве маршрутизатора может выступать как специализированное (аппаратное) устройство, так и обычный компьютер, выполняющий функции маршрутизатора. Существуют программные пакеты, с помощью которых можно превратить персональный компьютер в высокопроизводительный и многофункциональный маршрутизатор.

4.2.5 Межсетевой экран

Межсетевой экран *(Firewall, Brandmauer) — комплекс аппаратных или программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами (рис. 4.4).*

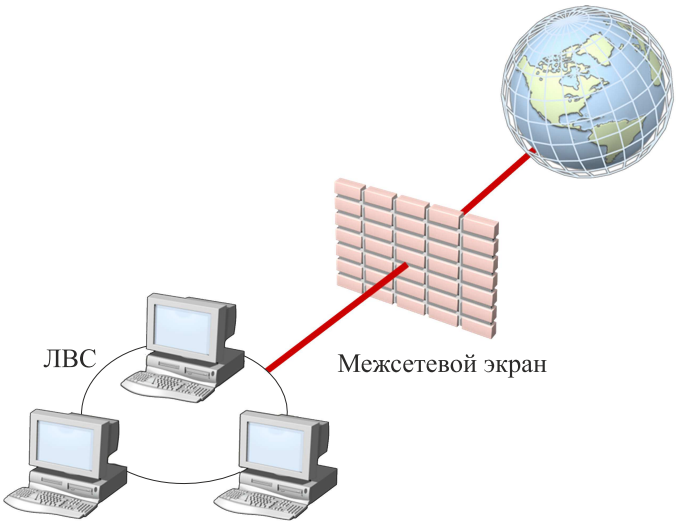


Рис. 4.4 – Пример использования межсетевого экрана

Основной задачей сетевого экрана является защита компьютерных сетей или отдельных узлов от несанкционированного доступа. Также сетевые экраны часто называют фильтрами, так как их основная задача — не пропускать пакеты, не подходящие под критерии, определённые в конфигурации.

Некоторые сетевые экраны также позволяют осуществлять трансляцию адресов — динамическую замену внутрисетевых (серых) адресов или портов на внешние, используемые за пределами ЛВС.

Вследствие защитных ограничений могут быть заблокированы некоторые необходимые пользователю службы, такие как Telnet[1], FTP[2], SMB[3], NFS[4] и так далее. Также следует отметить, что использование межсетевого экрана увеличивает время отклика и снижает пропускную способность, поскольку фильтрация происходит не мгновенно.

Между тем межсетевой экран не защищает узлы сети от проникновения через «люки» (англ. *back doors*) или уязвимости программного обеспечения, не обеспечивает защиту от многих внутренних угроз, в первую очередь — утечки данных, не защищает от загрузки пользователями вредоносных программ, в том числе вирусов.

Для решения последних двух проблем используются соответствующие дополнительные средства, в частности антивирусы. Обычно они подключаются к межсетевому экрану и пропускают через себя соответствующую часть сетевого трафика, работая как прозрачный для прочих сетевых узлов прокси-сервер[5], или же получают с межсетевого экрана копию всех пересылаемых данных. Однако такой анализ

[1] Telnet — сетевой протокол для организации интерфейса командной строки по сети.

[2] FTP (File Transfer Protocol) — стандартный протокол, предназначенный для передачи файлов по TCP-сетям.

[3] SMB (Server Message Block) — сетевой протокол прикладного уровня для удалённого доступа к файлам, принтерам и другим сетевым ресурсам, а также для межпроцессного взаимодействия.

[4] NFS (Network File System) — протокол сетевого доступа к файловым системам.

[5] Прокси-сервер (от англ. *proxy* — «представитель, уполномоченный») — служба (комплекс программ) в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам.

требует значительных аппаратных ресурсов, поэтому обычно проводится на каждом узле сети самостоятельно.

Различают следующие типы межсетевых экранов.

1. Управляемые коммутаторы (канальный уровень).
2. Сетевые фильтры сетевого уровня. Фильтрация статическая осуществляется путём анализа IP-адреса источника и приёмника, протокола, портов отправителя и получателя.
3. Шлюзы сеансового уровня. В сетевой модели TCP/IP нет уровня, однозначно соответствующего сеансовому уровню OSI, поэтому к шлюзам сеансового уровня относят фильтры, которые невозможно отождествить ни с сетевым, ни с транспортным, ни с прикладным уровнем.
4. Шлюз прикладного уровня, часто называемый прокси-сервером. Делятся на прозрачные и непрозрачные.
5. Брандмауэр SPI (Stateful Packet Inspection), или, иначе, брандмауэры с динамической фильтрацией пакетов (Dynamic Packet Filtering), являются по сути шлюзами сеансового уровня с расширенными возможностями. Инспекторы состояния оперируют на сеансовом уровне, но «понимают» протоколы прикладного и сетевого уровней. В отличие от шлюза прикладного уровня, открывающего два виртуальных канала TCP (один — для клиента, другой — для сервера) для каждого соединения, инспектор состояния не препятствует организации прямого соединения между клиентом и сервером.

Существует также понятие «межсетевой экран экспертного уровня». Сетевой экран данного типа базируется на посредниках прикладного уровня или инспекторах состояния, но обязательно комплектуется шлюзами сеансового уровня и сетевыми фильтрами, иногда понимая и сетевой уровень. Зачастую имеет систему протоколирования событий и оповещения администраторов, средства поддержки удаленных пользователей, средства построения виртуальных частных сетей и т. д. К нему относятся почти все имеющиеся на рынке брандмауэры.

4.3 Логическая организация сети

Компьютерную сеть, покрывающую относительно небольшую территорию или небольшую группу зданий (дом, офис, фирму, институт), обычно называют ***локальной вычислительной сетью (LAN — Local Area Network)****.*

Также могут быть локальные вычислительные сети, узлы которых разнесены на большие расстояния (центры управления орбитальными станциями).

При классификации сетей важным является ее способ администрирования. То есть в зависимости от того, как организована сеть и как она управляется, её можно отнести к локальной или какой-либо иной.

В локальной сети обычно используется минимальный контроль сетевого трафика. Коммутаторы автоматически строят таблицу коммутации к каждому из ком-

пьютеров и не контролируют сетевой трафик. Все *хосты (компьютеры или устройства)* на внутренних сетях считаются надежными. Если необходим контроль, то он может осуществляться на основе виртуальных сетей и с помощью ограничения подключения к портам только определенных компьютеров на основе контроля по физическим (MAC) адресам.

В локальных сетях, основанных на протоколе IPv4[1], могут использоваться специальные адреса: 10.0.0.0–10.255.255.255; 172.16.0.0–172.31.255.255; 192.168.0.0–192.168.255.255.

4.3.1 Глобальная компьютерная сеть

Компьютерную сеть, охватывающую большие территории и включающую в себя большое число компьютеров, называют ***глобальной компьютерной сетью (WAN — Wide Area Network)****.*

WAN служат для объединения разрозненных сетей так, чтобы пользователи и компьютеры, где бы они ни находились, могли взаимодействовать со всеми остальными участниками глобальной сети.

Некоторые WAN построены исключительно для частных организаций, другие являются средством коммуникации корпоративных LAN с сетью Интернет или посредством Интернет с удалёнными сетями, входящими в состав корпоративных. Чаще всего WAN опирается на выделенные линии, на одном конце которых маршрутизатор подключается к LAN, а на другом коммутатор связывается с остальными частями LAN.

Глобальные сети отличаются от локальных тем, что рассчитаны на неограниченное число абонентов и механизм управления обменом у них не гарантирует большую скорость.

В глобальных сетях намного более важно не качество связи, а сам факт ее существования. В настоящий момент времени нельзя провести четкий и однозначный предел между локальными и глобальными сетями. Большинство локальных сетей имеют выход в глобальную сеть, но характер переданной информации, принципы организации обмена, режимы доступа к ресурсам внутри локальной сети, как правило, сильно отличаются от тех, что приняты в глобальной сети. И хотя все компьютеры локальной сети в данном случае включены также и в глобальную сеть, специфику локальной сети это не отменяет. Возможность выхода в глобальную сеть остается всего лишь одним из ресурсов, поделенным пользователями локальной сети.

Приведем несколько примеров глобальной сети.

1. Соединение, арендованное у телекоммуникационной компании для соединения офисов в разных городах.

[1] IPv4 — Интернет протокол 4-й версии. Существует Интерент-протокол 6-й версии IPv6, но он еще недостаточно распространен и полностью не заменил 4-ю версию.

2. Соединение, арендованное у телекоммуникационной компании для соединения офисов в одном городе.
3. Беспроводное соединение между двумя зданиями на расстоянии более 10 км друг от друга.

4.3.2 Сеть периметра

В организациях при подключении корпоративных сетей к сетям общего пользования часто используется термин «*защита периметра*» или иначе «*сеть периметра*».

Сеть периметра *используется для изоляции внешних ресурсов компании, которые должны быть доступны и из локальной сети, и из сети Интернет.*

Сеть периметра рекомендуется формировать с помощью двух брандмауэров разных производителей (рис. 4.5). Один брандмауэр расположен между Интернетом и сетью периметра, второй расположен между сетью периметра и локальной сетью. Построение сети периметра с использованием одного межсетевого экрана менее надежно, чем с использованием двух межсетевых экранов. Сеть периметра изолирует ресурсы локальной сети и повышает безопасность работы всех компьютеров и других устройств сети.

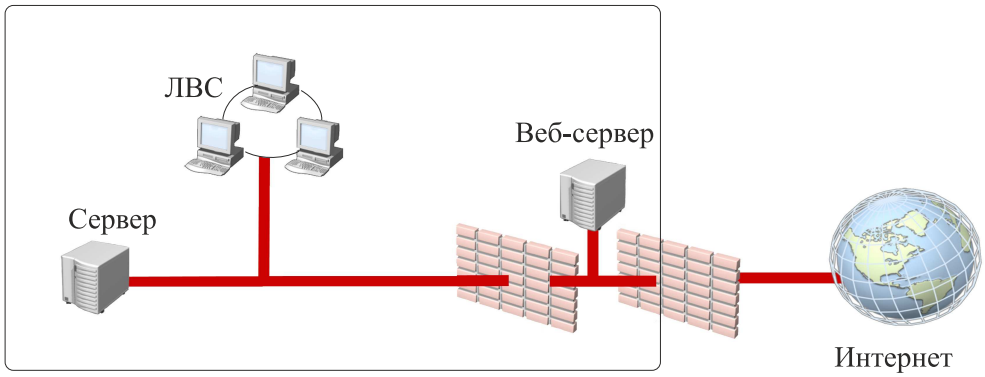


Рис. 4.5 – Организация периметра сети с использованием двух брандмауэров

Если брандмауэр один и через него произошло вторжение, то это даёт возможность атакующим получить доступ к внутренним сетевым ресурсам. Когда используются два межсетевых экрана, то необходимо взломать их одновременно, чтобы получить доступ в локальную сеть.

4.3.3 Удаленный доступ

Организация удаленного доступа обеспечивает доступ к ресурсам локальной сети за пределами офиса и может быть с использованием виртуальной частной се-

ти (VPN — Virtual Private Network) или коммутируемого удалённого доступа (Dial-up) (рис. 4.6).

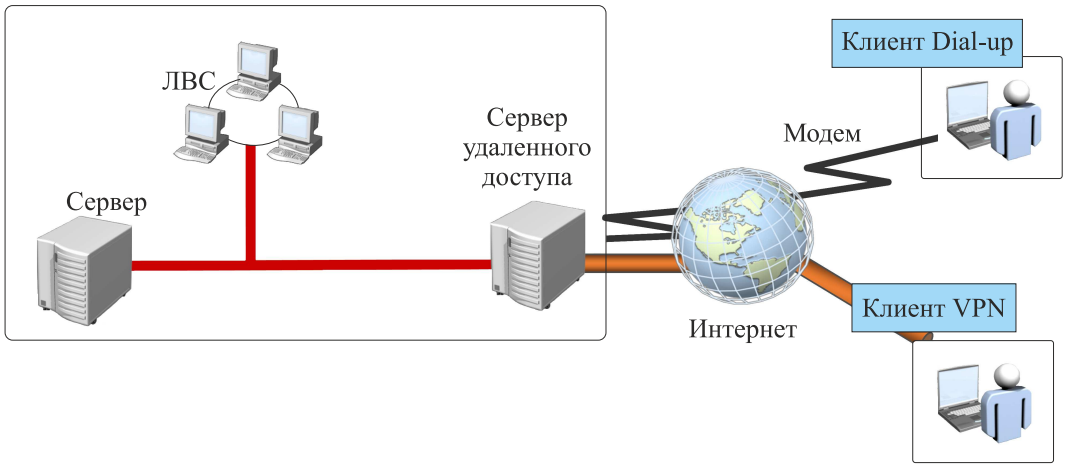
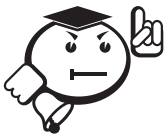
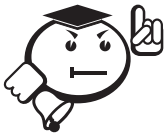


Рис. 4.6 – Организация удаленного доступа



VPN — технология, позволяющая обеспечить одно или несколько сетевых соединений (логическую сеть) поверх другой сети (глобальной компьютерной сети).

Несмотря на то, что коммуникации могут осуществляться по публичным сетям, уровень доверия к построенной логической сети не зависит от уровня доверия к базовым сетям благодаря использованию средств криптографии (шифрования, аутентификации, инфраструктуры открытых ключей, средств для защиты от повторов и изменений, передаваемых по логической сети сообщений). В зависимости от применяемых протоколов и назначения VPN может обеспечивать соединения трёх видов: узел-узел, узел-сеть и сеть-сеть.



Dial-up — сервис, позволяющий компьютеру, используя модем и телефонную сеть общего пользования, подключаться к другому компьютеру для инициализации сеанса передачи данных. Такое подключение может быть использовано как для обеспечения удаленного доступа в корпоративную сеть, так и для доступа в Интернет на домашнем компьютере.

В настоящее время пользователи, использующие домашнее подключение, как правило предпочитают использовать VPN или *службу терминалов*, которая может работать и через коммутируемый удаленный доступ. Для большинства приложений использование службы терминалов ускоряет работу при удаленном доступе, потому что приложение выполняется на сервере и данные не передаются через Интернет. Некоторые организации для повышения безопасности требуют, чтобы связь со службой терминалов производилась с использованием VPN.

Службы терминалов обеспечивают удаленный доступ к рабочему столу Microsoft Windows посредством программного обеспечения «тонкий клиент», позволяющего компьютеру клиента работать в качестве эмулятора терминала. Службы терминалов передают клиенту только пользовательский интерфейс программы. Затем клиент возвращает нажатия клавиш клавиатуры и кнопок мыши для выполнения на сервере. При входе каждый клиент видит только свои сеансы, которые управляются операционной системой сервера и являются независимыми от сеансов других пользователей. Для подключения к компьютеру через службы терминалов в операционной системе Microsoft Windows следует использовать программу «Подключение к удаленному рабочему столу».

4.3.4 Служба каталогов

Служба каталогов *в контексте компьютерных сетей — программный комплекс, позволяющий администратору работать с упорядоченным по ряду признаков массивом информации о сетевых ресурсах (общие папки, серверы печати, принтеры, пользователи и т. д.), хранящимся в едином месте, что обеспечивает централизованное управление как самими ресурсами, так и информацией о них, а также позволяет контролировать использование их третьими лицами.*

Можно выделить основные преимущества использования службы каталогов в сравнении с рабочей группой: простое управление безопасностью, надежное хранение секретной информации, групповая политика, масштабируемость.

Служба каталогов требует внедрения групповой политики, позволяющей управлять компьютерами пользователей: централизованная установка программного обеспечения, стандартизация настроек компьютеров, настройки веб-браузера.

Корпорация Microsoft в составе операционной системы Windows выпускает свою службу каталогов Active Directory. Active Directory позволяет администраторам использовать групповые политики для обеспечения единообразия настройки пользовательской рабочей среды, разворачивать программное обеспечение на множестве компьютеров через групповые политики или посредством System Center Configuration Manager (ранее Microsoft Systems Management Server), устанавливать обновления операционной системы, прикладного и серверного программного обеспечения на всех компьютерах в сети, используя службу обновления Windows Server. Active Directory хранит данные и настройки среды в централизованной базе данных. Масштабируемость Службы каталогов позволяет приложениям хранить в ее базе свою информацию. В рабочей группе такой возможности не существует. Сети Active Directory могут быть различного размера: от нескольких десятков до нескольких миллионов объектов.

Службы каталогов, помимо Microsoft Active Directory, имеют целый ряд коммерческих (Novell eDirectory, iPlanet Directory) и свободных программных реализаций (OpenLDAP, Apache Directory Server, 389 Directory Server, Samba).

Служба каталогов не содержит такой информации о сетевых устройствах, как таблицы маршрутизации. За это отвечают сами сетевые устройства.

4.3.5 Контроллеры доменов

Мастер установки Active Directory позволяет создавать *контроллеры домена*.

Контроллер домена *в компьютерных сетях, построенных на Microsoft Windows Servers, — это сервер, контролирующий область компьютерной сети (домен).*

Домен *— минимальная структурная единица организации Active Directory.*

Контроллеры домена хранят данные каталога и управляют взаимодействиями пользователя и домена, включая процессы входа пользователя в систему, проверку подлинности и поиск.

Приведем общие характеристики домена [19]:

- один или несколько компьютеров являются серверами. Администраторы сети используют серверы для контроля безопасности и разрешений для всех компьютеров домена. Это позволяет легко изменять настройки, так как изменения автоматически производятся для всех компьютеров. Пользователи домена должны указывать пароль или другие учетные данные при каждом доступе к домену;
- если пользователь имеет учетную запись в домене, он может войти в систему на любом компьютере. Для этого не требуется иметь учетную запись на самом компьютере;
- права изменения параметров компьютера могут быть ограничены, так как администраторы сети хотят быть уверены в единообразии настроек компьютеров;
- в домене могут быть тысячи компьютеров;
- компьютеры могут принадлежать к различным локальным сетям.

Домены могут объединяться в иерархическую систему, имеющую один корень. Такое объединение называется деревом доменов. Множество деревьев доменов, находящихся в различных формах доверительных отношений, называются лесом доменов. Все домены леса имеют одну и ту же схему и конфигурацию.

Но нужно помнить, что даже когда доверительные отношения настраиваются автоматически, пользователям все равно необходимо назначать разрешения для доступа к ресурсам в другом домене.

Помимо доменов сетевые технологии поддерживают понятия рабочей и домашней группы.

О рабочей группе можно сказать следующее [19]:

1. Все компьютеры являются одноранговыми узлами сети; ни один компьютер не может контролировать другой.

2. На каждом компьютере находится несколько учетных записей пользователя. Чтобы войти в систему любого компьютера, принадлежащего к рабочей группе, необходимо иметь на этом компьютере учетную запись.
3. В составе рабочей группы обычно насчитывается не больше двадцати компьютеров.
4. Рабочая группа не защищена паролем.
5. Все компьютеры должны находиться в одной локальной сети или подсети.

Домашняя группа имеет следующие характеристики [19]:

1. Компьютеры в домашней сети должны принадлежать рабочей группе, но они также могут состоять в домашней группе. С помощью домашней группы предоставлять доступ к рисункам, музыке, видео, документам и принтерам другим пользователям намного проще.
2. Домашняя группа защищена паролем, но он вводится только один раз при добавлении компьютера в домашнюю группу.

4.4 Основы TCP/IPv4

4.4.1 Обзор семейства протоколов TCP/IP

Во многих сетях, включая Интернет, используется набор сетевых протоколов передачи данных TCP/IP. Название TCP/IP происходит из двух наиважнейших протоколов семейства — Transmission Control Protocol (TCP) и Internet Protocol (IP), которые были разработаны и описаны первыми в данном стандарте. Сетевые протоколы организованы в виде стека, аналогично представлению модели открытых систем OSI.

Стек протоколов TCP/IP включает в себя четыре уровня (рис. 4.7):

1) прикладной уровень;
2) транспортный уровень;
3) сетевой уровень;
4) канальный уровень.

Протоколы этих уровней полностью реализуют функциональные возможности модели OSI. В стеке TCP/IP верхние три уровня модели OSI (прикладной, представительский и сеансовый) объединяют в один — прикладной. Стек является независимым от физической среды передачи данных. Понимание процесса взаимодействия протоколов семейства TCP/IP может быть полезно при поиске неисправностей в сетях и дает представление о взаимодействии приложений. А соотношение уровней модели TCP/IP с соответствующими уровнями модели OSI позволяет понимать функции протоколов TCP/IP и сравнивать их с другими протоколами и технологиями.

В настоящее время первичной публикацией информационных документов, содержащих описание протоколов Интернета, занимается открытое международное сообщество IETF (см. п. 4.1), публикуя документы из серии RFC (см. п. 4.1). Все стандарты TCP/IP публикуются в RFC. Обобщенно процесс разработки выглядит

так: кто угодно может представить документ RFC; документ рассматривается и ему присваивается статус (обязательный, рекомендуемый, факультативный, ограниченное использование, нерекомендуемый); если документ принимается как стандарт, то ему присваивается уровень завершенности (предложенный стандарт, черновой стандарт, стандарт Интернета).

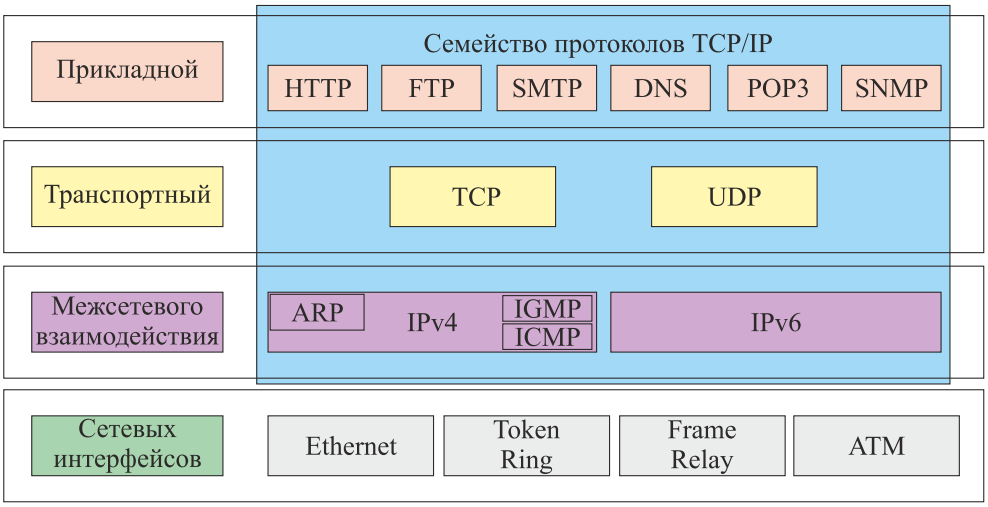


Рис. 4.7 – Архитектура TCP/IP

RFC никогда не изменяется, а заменяется новым RFC с внесенными изменениями. RFC2026 описывает процесс утверждения RFC. Любой специалист может ознакомиться с RFC, когда ему требуется более глубокое понимание протоколов для устранения неполадок в сети.

4.4.2 Протоколы транспортного уровня

TCP — Transmission Control Protocol[1]. Протокол TCP предоставляет поток данных с предварительной установкой соединения, осуществляет повторный запрос данных в случае потери данных и устраняет дублирование при получении двух копий одного пакета, гарантируя тем самым целостность передаваемых данных и уведомление отправителя о результатах передачи.

При осуществлении передачи в сети Интернет TCP работает на верхнем уровне между двумя конечными системами (например, браузером и веб-сервером). TCP осуществляет надежную передачу потока байтов от одной программы на некотором компьютере к другой программе на другом компьютере (например, программы для электронной почты, для обмена файлами). TCP контролирует длину сообщения, скорость обмена сообщениями, сетевой трафик.

Основные характеристики TCP [20]:

- *Надёжность* — TCP управляет подтверждением, повторной передачей и тайм-аутом сообщений. Производятся многочисленные попытки доставить сообщение. Если оно потеряется на пути, сервер вновь запросит по-

[1]RFC 793. URL: http://tools.ietf.org/html/rfc793 (дата обращения: 18.03.2015), на русском языке — URL: http://rfc.com.ru/rfc793.htm (дата обращения: 18.03.2015).

терянную часть. В TCP нет ни пропавших данных, ни разорванных соединений.

- *Упорядоченность* — если два сообщения последовательно отправлены, первое сообщение достигнет приложения-получателя первым. Если участки данных прибывают в неверном порядке, TCP отправляет неупорядоченные данные в буфер до тех пор, пока все данные не могут быть упорядочены и переданы приложению.
- *Тяжеловесность* — TCP необходимо три пакета для установки сокет-соединения перед тем, как отправить данные. TCP следит за надёжностью и перегрузками.
- *Потоковость* — данные читаются как поток байтов, не передается никаких особых обозначений для границ сообщения или сегментов.

UDP — User Datagram Protocol[1]. С UDP программные приложения могут посылать сообщения (датаграммы) другим хостам по сети без необходимости предварительного сообщения для установки специальных каналов передачи. UDP не обеспечивает надежность, упорядоченность и целостность данных. UDP подразумевает, что проверка ошибок и исправление либо не нужны, либо должны исполняться в приложении. В приложениях, относящихся к системам реального времени, часто предпочтительнее сбросить пакеты, чем ждать задержавшиеся пакеты. UDP-протокол, работающий без сохранения состояния, также полезен для серверов, отвечающих на небольшие запросы от огромного числа клиентов (например, потоковые мультимедийные приложения, онлайн-игры, система доменных имен).

Основные характеристики UDP [20]:

- *Ненадёжный* — когда сообщение посылается, неизвестно, достигнет ли оно своего назначения, — оно может потеряться по пути. Нет таких понятий, как подтверждение, повторная передача, тайм-аут.
- *Неупорядоченность* — если два сообщения отправлены одному получателю, то порядок их достижения цели не может быть предугадан.
- *Легковесность* — никакого упорядочивания сообщений, никакого отслеживания соединений и т. д. Это небольшой транспортный уровень, разработанный на IP.
- *Датаграммы* — пакеты посылаются по отдельности и проверяются на целостность, только если они прибыли. Пакеты имеют определенные границы, которые соблюдаются после получения, то есть операция чтения на сокете-получателе выдаст сообщение таким, каким оно было изначально послано.
- *Нет контроля перегрузок* — UDP сам по себе не избегает перегрузок. Для приложений с большой пропускной способностью возможно вызвать коллапс перегрузок, если только они не реализуют меры контроля на прикладном уровне.

[1] RFC 768. URL: http://tools.ietf.org/html/rfc768 (дата обращения: 18.03.2015), на русском языке — URL: http://rfc.com.ru/rfc768.htm (дата обращения: 18.03.2015).

4.4.3 Протоколы прикладного уровня

HTTP — HyperText Transfer Protocol[1]. HTTP — протокол передачи гипертекста, первоначально использовался для передачи текста в формате HTML[2], в настоящее время используется для передачи произвольных данных. HTTP используется в Интернете для получения информации с веб-сайтов.

Программное обеспечение для работы с протоколом HTTP разделяется на три категории: серверы, клиенты и прокси.

Серверы являются основными поставщиками услуг хранения и обработки информации. Из большого перечня существующих веб-серверов выделим два, на взгляд автора, наиболее популярных сервера: Internet Information Services (Microsoft), Apache HTTP Server (Apache Software Foundation).

Клиенты — это конечные потребители услуг сервера. Основными реализациями клиентов являются браузеры (программы для просмотра содержания веб-документов, компьютерных файлов и управления веб-приложениями). Перечень популярных браузеров также велик, выделим некоторые из них: Internet Explorer, Mozilla Firefox, Google Chrome, Safari, Opera, Opera Mini, Netscape Navigator.

Прокси[3] используются для выполнения транспортных служб. Прокси-сервер — это удаленный компьютер/служба/комплекс программ, который является посредником для выхода абонента в Интернет. Прокси позволяет клиентам выполнять косвенные запросы к другим сетевым службам. Прокси-сервер позволяет защищать компьютер клиента от некоторых сетевых атак и помогает сохранять анонимность клиента. В качестве примеров прокси-серверов приведем: для ОС Windows — CoolProxy, Microsoft Forefront Threat Management Gateway, WinGate, Kerio Control; для ОС Linux — Ideco ICS, Kerio Control; для ОС BSD — TOR, 3proxy.

HTTPS — HyperText Transfer Protocol Secure. HTTPS — не является отдельным протоколом, это расширение протокола HTTP, поддерживающее шифрование. HTTPS широко используется в Интернете и поддерживается всеми популярными браузерами. Он обеспечивает защиту от атак, основанных на прослушивании сетевого соединения.

RPC — Remote Procedure Call over HTTP. RPC — удаленный вызов процедур. Эта технология позволяет программам вызывать функции или процедуры в другом адресном пространстве на удаленных компьютерах. RPC использует на транспортном уровне протоколы TCP и UDP, однако некоторые могут использовать и протокол HTTP. Это нарушает архитектуру ISO/OSI, так как HTTP — протокол прикладного уровня.

FTP — File Transfer Protocol[4]. FTP — стандартный протокол, предназначенный для передачи файлов по сетям. FTP часто используется для загрузки сетевых страниц и других документов с частного устройства разработки на открытые сервера хостинга.

Особенностью протокола FTP является использование множественного подключения. Один канал является управляющим, через который поступают команды

[1] RFC 1945. URL: http://tools.ietf.org/html/rfc1945 (дата обращения: 18.03.2015).

[2] HTML — HyperText Markup Language (язык гипертекстовой разметки).

[3] От английского языка *proxy* — представитель.

[4] RFC 959. URL: http://tools.ietf.org/html/rfc959 (дата обращения: 18.03.2015).

серверу и возвращаются его ответы, а через остальные происходит собственно передача данных, по одному каналу на каждую передачу. Благодаря этому в рамках одной сессии по протоколу FTP можно передавать одновременно несколько файлов.

SMTP — Simple Mail Transfer Protocol[1]. SMTP — сетевой протокол, предназначенный для передачи электронной почты в сетях TCP/IP. Протокол SMTP предназначен для передачи исходящей почты с использованием порта TCP 25. Клиентские почтовые приложения обычно используют протокол SMTP только для отправки сообщений на почтовый сервер для ретрансляции. Для получения сообщений клиентские приложения используют либо протокол POP (Post Office Protocol), либо протокол IMAP (Internet Message Access Protocol), либо патентованные системы Microsoft Exchange, Lotus Notes/Domino.

POP3 — Post Office Protocol Version 3[2]. POP3 — стандартный протокол прикладного уровня, используемый клиентами электронной почты для получения почты с удаленного сервера по TCP/IP-соединению. POP поддерживает простые требования по загрузке и удалению для доступа к удаленным почтовым ящикам. POP-клиенты после загрузки почты могут выбирать оставить ее на сервере или удалить.

IMAP — Internet Message Access Protocol[3]. IMAP — протокол прикладного уровня для доступа к электронной почте. IMAP предоставляет пользователю обширные возможности для работы с почтовыми ящиками, находящимися на центральном сервере. Почтовая программа, использующая этот протокол, получает доступ к хранилищу корреспонденции на сервере так, как будто эта корреспонденция расположена на компьютере получателя. Электронными письмами можно манипулировать с компьютера пользователя (клиента) без постоянной пересылки с сервера и обратно файлов с полным содержанием писем.

SNMP — Simple Network Management Protocol[4]. SNMP — стандартный интернет-протокол для управления устройствами в IP-сетях на основе архитектур TCP/UDP. К поддерживающим SNMP-устройствам относятся маршрутизаторы, коммутаторы, серверы, рабочие станции, принтеры, модемные стойки и другие. Протокол обычно используется в системах сетевого управления для контроля подключенных к сети устройств на предмет условий, которые требуют внимания администратора. SNMP предоставляет данные для управления в виде переменных, описывающих конфигурацию управляемой системы.

DHCP — Dynamic Host Configuration Protocol[5]. Протокол динамической конфигурации хостов DHCP применяют для автоматического присвоения клиентам адресной информации протокола IP. DHCP позволяет компьютерам автоматически получать IP-адрес и другие параметры, необходимые для работы в сети TCP/IP. Для автоматической конфигурации компьютер-клиент на этапе конфигурации сетевого

[1] Представлен большим количеством RFC, которые также выложены на сайте организации IETF. SMTP впервые был описан в RFC 821 (1982 год); одно из последних обновлений RFC 5321.

[2] POP (POP1) определен в RFC 918 (1984 год), Первоначальная спецификация POP3 была представлена в RFC 1081 (1988 год). Нынешняя же описана в RFC 1939, обновлена механизмом расширения (RFC 2449) и механизмом аутентификации (RFC 1734).

[3] RFC 3501. URL: https://tools.ietf.org/html/rfc3501 (дата обращения: 18.03.2015).

[4] Представлен большим количеством RFC, которые также выложены на сайте организации IETF. С 2004 года IETF признает SNMPv3, определенный в RFC 3411, RFC 3418 (также известный как STD0062) в качестве текущей стандартной версии SNMP.

[5] RFC 1231. URL: https://tools.ietf.org/html/rfc1231 (дата обращения: 18.03.2015).

устройства обращается к так называемому серверу DHCP и получает от него нужные параметры. Сетевой администратор может задать диапазон адресов, распределяемых сервером среди компьютеров. Это позволяет избежать ручной настройки компьютеров сети и уменьшает количество ошибок. Протокол DHCP используется в большинстве сетей TCP/IP.

Приведем преимущества использования протокола DHCP:

- Позволяет избежать ручной настройки клиентов.
- Клиенты могут автоматически переходить из одной подсети в другую.
- Автоматически освобождает адрес IPv4, когда компьютер удален из сети.
- Снижает вероятность дублирования адресов IPv4.

С помощью протокола DHCP назначают адреса только клиентским компьютерам. Устройства, которым не назначают адреса с помощью DHCP, — это серверы, коммутаторы и принтеры.

4.4.4 Адресация TCP/IPv4

Адресация TCP/IP версии 4 строится из двух составных частей: номера сети (подсети) и номера хоста. IP-адрес имеет длину 4 байта. Удобной формой записи IP-адреса (IPv4) является запись в виде четырёх десятичных чисел значением от 0 до 255, разделённых точками, например 192.168.2.180 (рис. 4.8). В случае локально закрытой сети её адрес может быть выбран администратором из специально зарезервированных для таких сетей блоков адресов: 10.0.0.0/8, 172.16.0.0/12 или 192.168.0.0/16 (см. 4.3.1). Если же сеть должна работать как составная часть Интернета, то адрес сети выдаётся провайдером либо региональным интернет-регистратором. Есть два способа определения того, сколько бит отводится на маску подсети, а сколько — на номер хоста.

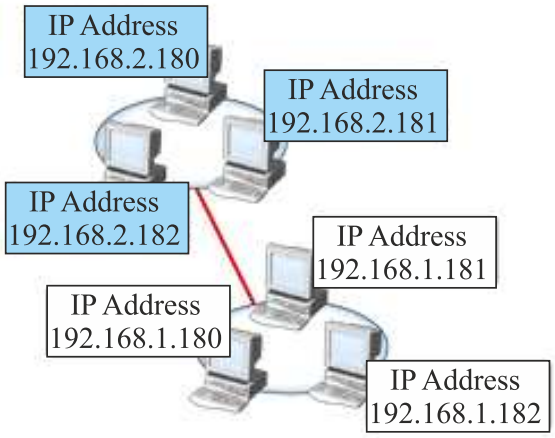


Рис. 4.8 – Пример адресации сети

Изначально использовалась классовая адресация *(INET)*, но со второй половины 90-х годов XX века она была вытеснена бесклассовой адресацией *(CIDR)*, при которой количество адресов в сети определяется *маской подсети*, которая также имеет длину 4 байта и накладывается на IP-адрес. Число 255 в маске подсети по-

казывает, что октет является частью сетевого номера, и правильные маски подсети выглядят так, как показано на рисунке 4.9.

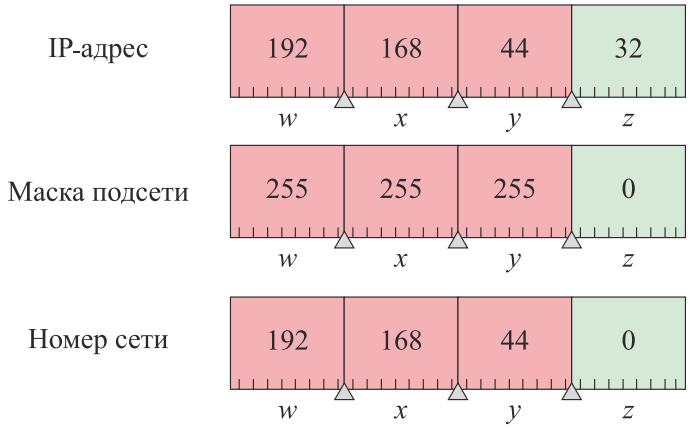


Рис. 4.9 – Выделение номера сети из IP-адреса по маске подсети

Приведем еще пример, если у компьютера IP-адрес 192.32.09.220 и маска подсети 255.255.0.0, то у него номер сети будет 192.32.0.0, а номер хоста будет 0.0.09.220.

Когда маска подсети введена неправильно, то компьютер не будет взаимодействовать с другими компьютерами в сети.

IP-адреса могут быть статическими и динамическими. Адрес является статическим, если он прописан в настройках устройства или если он всегда один и тот же выдается сервером распределения адресов. IP-адрес может быть динамическим если он назначается при подключении устройства к сети автоматически из некоторого диапазона и используется в течение ограниченного промежутка времени. Динамическая выдача адресов используется провайдерами в целях экономии адресов и в локальных сетях для удобства администрирования.

Для подключения компьютера к внешней сети помимо настройки IP-адреса необходимо также обеспечить и настройку шлюза (маршрутизатора). Если шлюз по умолчанию настроен неверно: компьютер может использовать только внутренние ресурсы сети и никакие другие; компьютер может взаимодействовать с другими компьютерами внутри сети и Вы не можете выйти в Интернет.

4.4.5 Система доменных имен DNS

DNS — Domain Name System. DNS — компьютерная распределённая система для получения информации о доменах. Чаще всего используется для получения IP-адреса по имени хоста.

Распределённая база данных DNS поддерживается с помощью иерархии DNS-серверов, взаимодействующих по определённому протоколу. Основой DNS является представление об иерархической структуре доменного имени и зонах. Каждый сервер, отвечающий за имя, может делегировать ответственность за дальнейшую часть домена другому серверу, что позволяет возложить ответственность за актуальность информации на серверы различных организаций, отвечающих только за свою часть доменного имени.

В системе доменных имен зона представляет собой определенную часть пространства имен DNS, разделенного точками, которое может содержать DNS-записи

(например: microsoft.com). Целью выделения части дерева в отдельную зону является передача ответственности за соответствующий домен другому лицу или организации. Зона DNS содержит записи DNS нескольких типов, приведем некоторые из них:

- A — для связывания имени хоста с адресом протокола IPv4;
- SRV — для поиска контроллера домена;
- MX — для поиска почтового сервера;
- CNAME — каноническая запись имени (псевдоним) используется для перенаправления на другое имя;
- PTR — для связывания IP-адрес хоста с его каноническим именем.

В большинстве случаев DNS-записи, необходимые для службы каталогов Active Directory, добавляются к необходимой зоне автоматически контроллерами домена и серверами глобального каталога. Кроме того, рабочие станции и сервера автоматически создают свои собственные A- и PTR-записи.

Приведем пример процесса разрешения DNS-имени www.microsoft.com в IP-адрес (рис. 4.10).

1. Рабочая станция запрашивает у локального DNS-сервера IP-адрес для www.microsoft.com.
2. Если локальный DNS-сервер не располагает информацией, то он запрашивает корневой DNS-сервер о размещении DNS-сервера домена .com.
3. Локальный сервер DNS запрашивает DNS-сервер домена .com о размещении DNS-сервера домена microsoft.com.
4. Локальный сервер DNS запрашивает у DNS-сервера домена microsoft.com IP-адрес для www.microsoft.com.
5. Локальный сервер DNS возвращает рабочей станции IP-адрес www.microsoft.com.

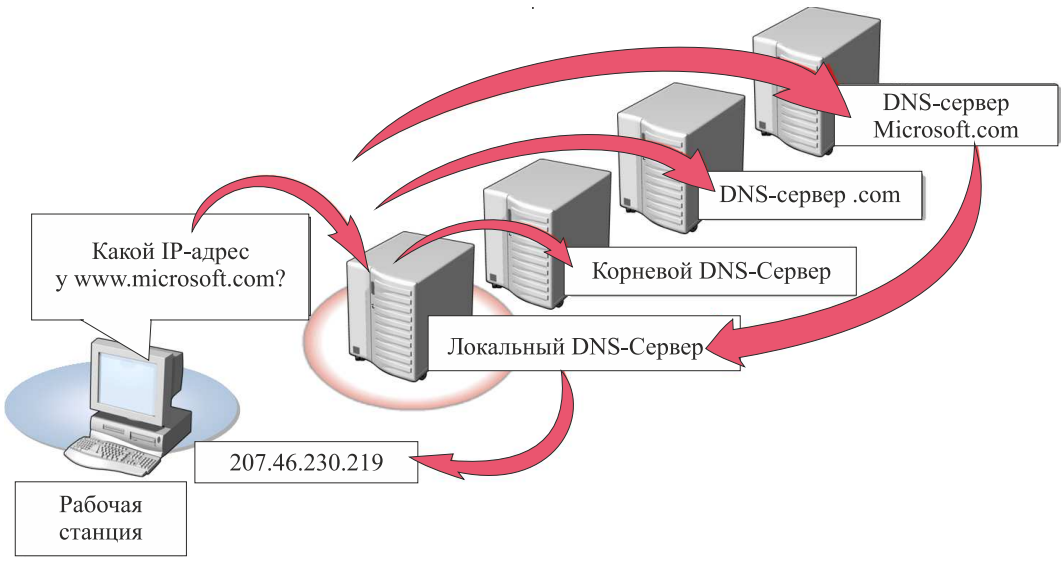


Рис. 4.10 – Процесс разрешения DNS-имени www.microsoft.com в IP-адрес

Понимание этого процесса имеет важное значение при устранении неполадок разрешения имен как для клиентов так и для серверов. Например, когда клиент не может получить доступ к веб-приложению или файловому серверу.

DNS обладает следующими характеристиками [21].

- *Распределённость администрирования*. Ответственность за разные части иерархической структуры несут разные люди или организации.
- *Распределённость хранения информации*. Каждый узел сети в обязательном порядке должен хранить только те данные, которые входят в его зону ответственности, и (возможно) адреса корневых DNS-серверов.
- *Кеширование информации*. Узел может хранить некоторое количество данных не из своей зоны ответственности для уменьшения нагрузки на сеть.
- *Иерархическая структура*, в которой все узлы объединены в дерево и каждый узел может или самостоятельно определять работу нижестоящих узлов, или делегировать (передавать) их другим узлам.
- *Резервирование*. За хранение и обслуживание своих узлов (зон) отвечают (обычно) несколько серверов, разделённых как физически, так и логически, что обеспечивает сохранность данных и продолжение работы даже в случае сбоя одного из узлов.

Для людей проще запоминать буквенные адреса, чем последовательность цифр IP-адреса, поэтому система доменных имен очень важна для работы в Интернете, так как для соединения с узлом необходима информация о его IP-адресе.

4.5 Диагностика сети

4.5.1 Просмотр свойств сетевого окружения

Получить информацию о свойствах сетевого окружения возможно с использованием следующих действий: нажмите кнопку «Пуск» и в появившемся окне щелкните правой кнопкой мыши по пункту «Сетевое окружение». В появившемся контекстном меню выберите пункт «Свойства». Перед вами появится окно, показанное на рисунке 4.11.

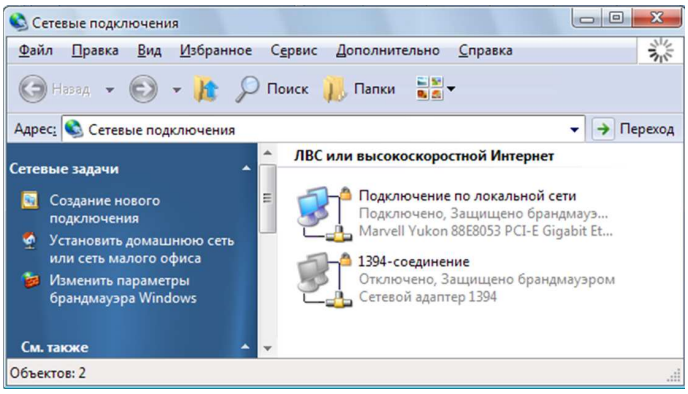


Рис. 4.11 – Свойства сетевого окружения

Чтобы получить информацию о свойствах подключения по локальной сети, щелкните по надписи «Подключение по локальной сети» правой кнопкой мыши и также в появившемся меню выберите пункт «Свойства». В появившемся окне (рис. 4.12) вы можете настраивать протоколы сетевых взаимодействий.

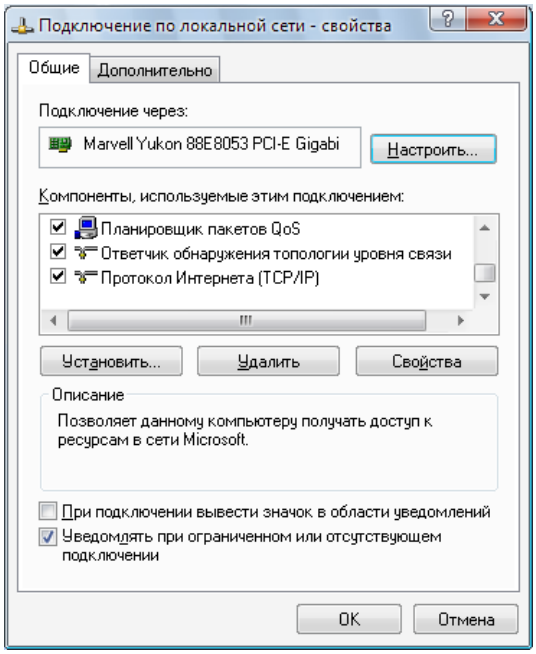


Рис. 4.12 – Свойства подключения по локальной сети

Выбрав этот компонент и нажав кнопку «Свойства», вы откроете окно (рис. 4.13), где можно устанавливать настройки сетевого подключения по протоколу TCP/IP.

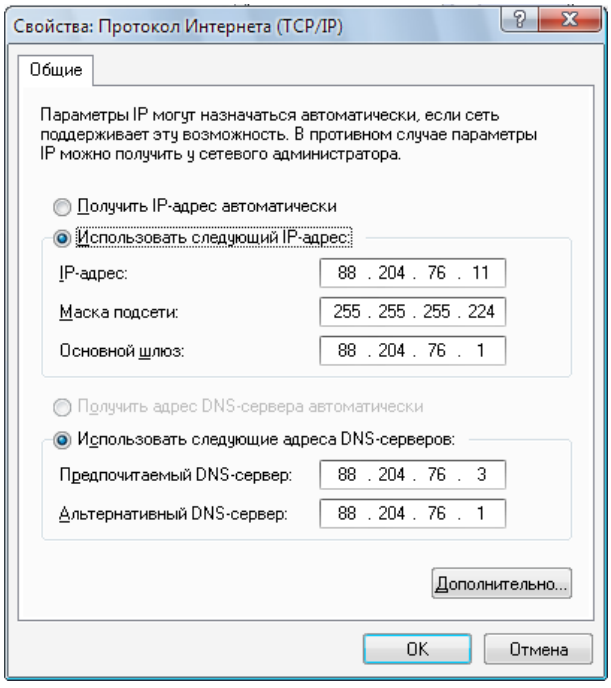


Рис. 4.13 – Свойства протокола Интернета (TCP/IP)

4.5.2 Утилиты диагностики сети

Существуют различные утилиты, позволяющие быстро протестировать IP-подключение. Однако большинство операций легко может быть выполнено с использованием команд самой операционной системы через интерфейс командной строки или пользователи могут воспользоваться специальным мастером с графическим интерфейсом.

Рассмотрим набор сетевых утилит, входящих в состав ОС Windows.

Утилита ipconfig. Для отображения параметров IP-протокола в ОС на платформе Windows NT используется утилита ipconfig. Эта утилита выводит на экран основные параметры настройки протокола TCP/IP: значения адреса, маски, шлюза [3].

1. Нажмите кнопку «Пуск», выберите строку меню «Выполнить», наберите символы cmd (запуск консоли командной строки) и нажмите клавишу Enter на клавиатуре.
2. Введите команду: `ipconfig /all`. При нормальной работе компьютера на экран должен вывестись примерно такой листинг:

```
Windows IP Configuration
  Host Name . . . . . . . . . . . . : w9
  Primary Dns Suffix . . . . . . . : aoi.tusur.ru
  Node Type . . . . . . . . . . . . : Hybrid
  IP Routing Enabled. . . . . . . . : No
  WINS Proxy Enabled. . . . . . . . : No
  DNS Suffix Search List. . . . . . : aoi.tusur.ru
                                      tomsk.ru
Ethernet adapter Local Area Connection:
  Connection-specific DNS Suffix . . : aoi.tusur.ru
  Description . . . . . . . . . . . : Intel(R) PRO/100 S
                                      Desktop Adapter
  Physical Address. . . . . . . . . : 00-03-BA-8D-42-5B
  Dhcp Enabled. . . . . . . . . . . : Yes
  Autoconfiguration Enabled . . . . : Yes
  IP Address. . . . . . . . . . . . : 83.192.12.54
  Subnet Mask . . . . . . . . . . . : 255.255.255.0
  Default Gateway . . . . . . . . . : 83.192.12.254
  DHCP Server . . . . . . . . . . . : 83.192.12.2
  DNS Servers . . . . . . . . . . . : 192.168.0.1
                                      83.192.12.2
  Primary WINS Server . . . . . . . : 83.192.12.2
  Secondary WINS Server . . . . . . : 213.183.109.8
  Lease Obtained. . . . . . . . . . : 27 августа 2012 г.
                                      19:20:22
  Lease Expires . . . . . . . . . . : 13 октября 2012 г.
                                      19:20:22
```

Отключите сетевое подключение, повторите команду. При отсутствующем соединении на экран выводится примерно такой листинг:

```
Windows IP Configuration
  Host Name . . . . . . . . . . . . : w9
  Primary Dns Suffix . . . . . . . : aoi.tusur.ru
  Node Type . . . . . . . . . . . . : Hybrid
  IP Routing Enabled. . . . . . . . : No
  WINS Proxy Enabled. . . . . . . . : No
  DNS Suffix Search List. . . . . . : aoi.tusur.ru
                                      tusur.ru
Ethernet adapter Local Area Connection:
  Media State . . . . . . . . . . . : Media disconnected
  Description . . . . . . . . . . . : Intel(R) PRO/100 S
                                       Desktop Adapter
  Physical Address. . . . . . . . . : 00-03-BA-8D-42-5
```

Обратите внимание, что программа вывела на экран только данные о «физических» параметрах сетевой карты и указала, что отсутствует подключение сетевого кабеля (Media disconnected).

Утилита ping. Утилита ping используется для проверки протокола TCP/IP и достижимости удаленного компьютера. Она выводит на экран время, за которое пакеты данных достигают заданного в ее параметрах компьютера.

1. Проверка правильности установки протокола TCP/IP. Откройте командную строку и выполните команду:

```
ping 127.0.0.1
```

Адрес 127.0.0.1 — это личный адрес любого компьютера. Таким образом, эта команда проверяет прохождение сигнала «на самого себя». Она может быть выполнена без наличия какого-либо сетевого подключения. Вы должны увидеть приблизительно следующие строки:

```
Pinging 127.0.0.1 with 32 bytes of data:
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Reply from 127.0.0.1: bytes=32 time<1ms TTL=128
Ping statistics for 127.0.0.1:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
  Minimum = 0ms, Maximum = 0ms, Average = 0ms
```

По умолчанию команда посылает пакет 32 байта. Размер пакета может быть увеличен до 65 Кбайт. Так можно обнаружить ошибки при пересылке пакетов больших размеров. За размером тестового пакета отображается время отклика удаленной системы (в нашем случае — меньше 1 миллисекунды). Потом показывается еще один параметр протокола — значение TTL. TTL — «время жизни» пакета. На практике это число маршрутизаторов, через которые может пройти пакет. Каждый маршрутизатор уменьшает значение TTL на единицу. При достижении нулевого

значения пакет уничтожается. Такой механизм введен для исключения случаев зацикливания пакетов.

Если будет показано сообщение о недостижимости адресата, то это означает ошибку установки протокола IP. В этом случае целесообразно удалить протокол из системы, перезагрузить компьютер и вновь установить поддержку протокола TCP/IP.

2. Проверка видимости локального компьютера и ближайшего компьютера сети. Выполните команду:

```
ping 192.168.0.19
```

На экран должны быть выведены примерно такие строки:

```
Pinging 212.73.124.100 with 32 bytes of data:
Reply from 192.168.0.19: bytes=32 time=5ms TTL=60
Reply from 192.168.0.19: bytes=32 time=5ms TTL=60
Reply from 192.168.0.19: bytes=32 time=4ms TTL=60
Reply from 192.168.0.19: bytes=32 time=4ms TTL=60
Ping statistics for 212.73.124.100:
  Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
  Minimum = 4ms, Maximum = 5ms, Average = 4ms
```

Наличие отклика свидетельствует о том, что канал связи установлен и работает.

Утилита tracert. При работе в сети одни информационные серверы откликаются быстрее, другие — медленнее, бывают случаи недостижимости желаемого хоста. Для выяснения причин подобных ситуаций можно использовать специальные утилиты.

Например, команда tracert, которая обычно используется для показа пути прохождения сигнала до желаемого хоста. Зачастую это позволяет выяснить причины плохой работоспособности канала. Точка, после которой время отклика резко увеличено, свидетельствует о наличии в этом месте узла, не справляющегося с нагрузкой.

В командной строке введите команду:

```
tracert 192.168.0.19
```

Вы должны увидеть примерно такой листинг:

```
Tracing route to 192.168.0.19
over a maximum of 30 hops:
  1 <1 ms <1 ms <1 ms 192.168.0.19
Trace complete.
```

Утилита route. Команда route позволяет просматривать маршруты прохождения сетевых пакетов при передаче информации.

Выведите на экран таблицу маршрутов TCP/IP, для этого в командной строке введите команду `route print`.

Утилита net view. Выводит список доменов, компьютеров или общих ресурсов на данном компьютере. Вызванная без параметров, команда net view выводит список компьютеров в текущем домене.

1. Введите `net view`, и вы увидите список компьютеров своей рабочей группы.
2. Введите `net view 92.165.0.12` для просмотра общих ресурсов, расположенных на компьютере 192.165.0.12.

Утилита net send. Служит для отправки сообщений другому пользователю, компьютеру или псевдониму, доступному в сети.

1. Введите `net send 192.168.0.1 Привет. Проверка связи.`
   Ваше сообщение получит пользователь 192.168.0.1.
2. Введите `net send * Привет. Проверка связи.`
   Ваше сообщение получат все пользователи рабочей группы.

Контрольные вопросы по главе 4

1. Назовите последовательно уровни сетевой модели OSI.
2. Что входит в состав физической инфраструктуры сети?
3. В чет состоит отличие локальной вычислительной сети от глобальной компьютерной сети?
4. Что такое сеть периметра?
5. В чем состоит отличие технологии VPN от Dial-up сервиса?
6. Что Вы знаете о Microsoft Active Directory?
7. В чем состоит преимущество использования доменов?
8. Приведите обзор семейства протоколов TCP/IP.
9. Приведите примеры протоколов транспортного и прикладного уровня TCP/IP.
10. Как строится адресация TCP/IP четвертой версии?
11. Назовите номер сети и номер хоста, если у компьютера IP-адрес 192.32.9.220 и маска подсети 255.255.255.0.
12. Как строится процесс разрешения DNS?
13. Какие утилиты диагностики сети Вы знаете?

ЗАКЛЮЧЕНИЕ

По окончанию изучения дисциплины «Вычислительные системы, сети и телекоммуникации» у студента должны сформироваться профессиональные знания по теоретическим основам архитектурной и программной организации вычислительных и информационных систем, по основным стандартам информационно-коммуникационных систем и технологий. Студент должен уметь настраивать конкретные конфигурации операционных систем, инсталлировать, тестировать, испытывать и использовать программные средства. Студент также должен владеть начальными навыками работы в среде различных операционных систем и способами их администрирования, а также методом рационального выбора вычислительных систем и информационно-коммуникационных технологий для управления бизнесом.

Изучение дисциплины «Вычислительные системы, сети и телекоммуникации» должно способствовать формированию у студентов следующих компетенций:

- готовность обосновать принимаемые проектные решения, осуществлять постановку и выполнение экспериментов по проверке их корректности и эффективности;
- знакомство с архитектурой ЭВМ и систем;
- навыки использования операционных систем, сетевых технологий, средств разработки программного интерфейса, применения языков и методов формальных спецификаций, систем управления базами данных;
- проводить анализ архитектуры предприятия;
- проводить исследование и анализ рынка ИС и ИКТ;
- выбирать рациональные ИС и ИКТ-решения для управления бизнесом;
- проводить анализ инноваций в экономике, управлении и ИКТ;
- проектировать архитектуру электронного предприятия;
- консультировать заказчиков по рациональному выбору ИС и ИКТ управления бизнесом.

ЛИТЕРАТУРА

[1] Кориков А. М. Теория систем и системный анализ : учеб. пособие / А. М. Кориков, С. Н. Павлов. — Томск : Томс. гос. ун-т систем управления и радиоэлектроники. — 2008. — 264 с.

[2] Бройдо В. Л. Вычислительные системы, сети и телекоммуникации / В. Л. Бройдо. — СПб. : Питер. — 2004. — 703 с.

[3] Гордеев А. В. Операционные системы : учебник для вузов / А. В. Гордеев. — 2-е изд. — СПб. : Питер. — 2004. — 416 с.

[4] China's Tianhe-2 Supercomputer Maintains Top Spot on 42nd TOP500 List / TOP500 [Электронный ресурс]. — URL: http://top500.org/blog/lists/2013/11/press-release/ (дата обращения: 04.02.2014).

[5] Создан самый мощный компьютер в мире / ДНИ.РУ [Электронный ресурс]. — URL: http://www.dni.ru/tech/2013/6/18/254597.html (дата обращения: 04.02.2014).

[6] Першиков В. И. Толковый словарь по информатике / В. И. Першиков, В. М. Савинков. — М. : Финансы и статистика. — 1991. — 543 с.

[7] Sublist Generator / TOP500 [Электронный ресурс]. — URL: http://top500.org/statistics/sublist/ (дата обращения: 04.02.2014).

[8] Юров В. Assembler : учебник / В. Юров. — СПб., 2001. — 624 с.

[9] Jon «Hannibal» Stokes. RISC and CISC, Side by Side? RISC vs. CISC: the Post-RISC Era / Ars Technica [Электронный ресурс]. — URL: http://archive.arstechnica.com/cpu/4q99/risc-cisc/rvc-5.html#Branch (дата обращения: 06.02.2014).

[10] Sivarama P. Dandamudi. Chapter 3: RISC Principles // Guide to RISC Processors for Programmers and Engineers. — Springer New York, 2005. — P. 39–44.

[11] Кондратьев Г. Г. Железо ПК. Популярный самоучитель / Г. Г. Кондратьев, В. С. Пташинский. — 2-е изд. — СПб. : Питер. — 2008. — 224 с.

[12] Гук М. Аппаратные средства IBM PC. Энциклопедия / М. Гук. — 3-е изд. — СПб. : Питер. — 2006. — 1072 с.

[13] Гордеев А. В. Системное программное обеспечение / А. В. Гордеев, А. Ю. Молчанов. — СПб. : Питер. — 2002. — 736 с.

[14] IA-32 Intel Architecture Software. Developer's Manual, Volume 1–4.

[15] Мухамедов Р. 64 «кусочка» из Калифорнии: обзор архитектуры AMD64 / www.overclockers.ru [Электронный ресурс] / Р. Мухаммедов. — URL : http://www.overclockers.ru/lab/15539.shtml (дата обращения: 30.03.2014).

[16] Назаров С. В. Операционные системы. Практикум / С. В. Назаров, Л. П. Гудыно, А. А. Кириченко ; под ред. С. В. Назарова. — М. : КУДИЦ-ПРЕСС. — 2008. — 464 с.

[17] Назаров С. В. Операционные среды, системы и оболочки. Основы структурной и функциональной организации : учеб. пособие / С. В. Назаров. — М. : КУДИЦ-ПРЕСС. — 2007. — 504 с. : ил.

[18] ISO/IEC 10731:1994 Information technology — Open Systems Interconnection — Basic Reference Model — Conventions for the definition of OSI services / ISO (International Organization for Standardization) [Электронный ресурс]. — URL: http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=18824, платный (дата обращения: 17.11.2014).

[19] В чем разница между доменом, рабочей группой и домашней группой? / Microsoft [Электронный ресурс]. — URL: http://windows.microsoft.com/ru-ru/windows7/what-is-the-difference-between-a-domain-a-workgroup-and-a-homegroup (дата обращения: 29.11.2014).

[20] UDP / Свободная энциклопедия Википедия [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/UDP (дата обращения: 30.11.2014).

[21] DNS / Свободная энциклопедия Википедия [Электронный ресурс]. — URL: https://ru.wikipedia.org/wiki/DNS (дата обращения: 30.11.2014).

[22] Олифер В. Г. Компьютерные сети: Принципы, технологии, протоколы : учеб. пособие для вузов / В. Г. Олифер, Н. А. Олифер. — 3-е изд. — СПб. : Питер, 2006. — 960 с.

СПИСОК УСЛОВНЫХ ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

АВМ — аналоговые вычислительные машины

ВС — вычислительная система

ГВМ — гибридные вычислительные машины

МЭСМ — малая электронная счётная машина

ОЗУ — оперативное запоминающее устройство

ОП — оперативная память

ОС — операционная система

ПЗУ — постоянное запоминающее устройство

ПО — программное обеспечение

ЦВМ — цифровые вычислительные машины

ЭВМ — электронная вычислительная машина

ABC — Atanasoff-Berry Computer

AMD — Advanced Micro Devices

AWE — Address Windowing Extensions

BIOS — Basic Input/Output System

CDFS — Compact Disk File System

CISC — Complex Instruction Set Computing

DDE — Dynamic Data Exchange

DHCP — Dynamic Host Configuration Protocol

DLL — Dynamic Link Library

DPB — Disk Parameter Blockout

DRF — Data Recovery Field

EDSAC — Electronic Delay Storage Automatic Computer

EDVAC — Electronic Discrete Variable Automatic Computer

EFS — Encrypting File System

EMS — Expanded Memory Specification

ENIAC — Electronic Numerical Integrator and Computer

EPRML — Extended PRML

FAT — File Allocation Table

FEK — File Encryption Key

FPU — Floating Point Unit

FSB — Front Side Bus

FTP — File Transfer Protocol

HMA — High Memory Area

HPFS — High Performance File System

HTML — HyperText Markup Language

HTTP — Hypertext Transfer Protocol

HTTPS — HyperText Transfer Protocol Secure

IEEE — Institute of Electrical and Electronics Engineers

IETF — Internet Engineering Task Force

IMAP — Internet Message Access Protocol

IP — Internet Protocol

ISO — International Standards Organization

ITU — International Telecommunication Union

LAN — Local Area Network

LDT — Logical Disk Table

MAC-адрес — Media Access Control

MBR — Master Boot Record

MCB — Memory Control Block

MISC — Minimal Instruction Set Computer

MMX — Multimedia Extensions

MS DOS — MicroSoft Disk Operating System

MTBF — Mean Time Before Failure

NFS — Network File System

NSB — Non-System Bootstrap

NTFS — New Technology File System

OLE — Object Linking and Embedding

OSTA — Optical Storage Technology Association

OSI — Open Systems Interconnection Basic Reference Model

PAE — Physical Address Extension

POP — Post Office Protocol

POST — Power-On Self-Test

PRML — Partial Response, Maximum Likelihood

PSP — Program Segment Prefix

PT — Partition Table

PTE — Page Table Entry

RAM — Random Access Memory

RFC — Request for Comments

RISC — Restricted Instruction Set Computer

ROM — Read Only Memory

RPC — Remote Procedure Call

SIMD — Single Instruction, Multiple Data

SMB — Server Message Block

SMBR — Secondary MBR

SMM — System Management Mode

SMTP — Simple Mail Transfer Protocol

SNMP — Simple Network Management Protocol

SSE — Streaming SIMD Extensions

SPI — Stateful Packet Inspection

TCP — Transmission Control Protocol

TTA — Transport Triggered Architecture

UDF — Universal Disk Format

UMA — Upper Memory Area

UMB — Upper Memory Block

VLAN — Virtual Local Area Network

VLIW — Very Long Instruction Word

VPN — Virtual Private Networ

W3C — World Wide Web Consortium

WAN — Wide Area Network

WLAN — Wireless Local Area Network

XMM — eXtended Memory Manager

XMS — Extended Memory Specification

ГЛОССАРИЙ

Аналоговые вычислительные машины — вычислительные машины, работающие с информацией в виде непрерывного ряда значений какой-либо физической величины. Чаще всего информация представляется электрическим напряжением.

Аппаратное обеспечение (аппаратные средства) — это электронные и механические части вычислительного устройства, входящие в состав системы или сети.

Архитектура вычислительной системы — это совокупность характеристик и параметров, определяющих функционально-логичную и структурно-организованную систему и затрагивающих в основном уровень параллельно работающих вычислителей.

Архитектура системы — совокупность свойств системы, существенных для пользователя.

Архитектура ЭВМ — концептуальная структура вычислительной машины, определяющая проведение обработки информации и включающая методы преобразования информации в данные и принципы взаимодействия технических средств и программного обеспечения.

Байт — единица информации, состоящая из восьми битов.

Бит — элемент памяти, содержащий один двоичный разряд.

Блочные устройства — устройства, которые используются для хранения информации в виде блоков фиксированного размера, причем у каждого блока есть адрес и каждый блок может быть прочитан независимо от остальных блоков.

Большие ЭВМ (мэйнфреймы, mainframe) — класс ЭВМ, который используется для решения научно-технических задач, для работы с большими базами данных, для управления вычислительными сетями и их ресурсами. На их основе могут быть реализованы территориальные центры обработки информации.

Виртуальная память — технология расширения доступной физической памяти компьютера. В системе виртуальной памяти операционная система создает файл подкачки и делит память на единицы, называемые страницами. Страницы, к которым недавно обращались, хранятся в физической памяти (ОЗУ). Если какое-то время к странице памяти не обращаются, она записывается в файл подкачки (процесс, называемый «подкачкой памяти» или «откачкой страницы»). Если позже

к этому участку памяти обращается программа, операционная система считывает страницу памяти из файла подкачки и помещает ее в физическую память (процесс, называемый «подкачкой памяти» или «подкачкой страницы»). Общий объем памяти, доступной программам, равен сумме физической памяти компьютера и размера файла подкачки.

Внешняя память — память, реализованная в виде устройств с различными принципами хранения информации, чаще всего с подвижными носителями. В настоящее время сюда входят устройства магнитной (дисковой и ленточной) памяти, оптической и магнитооптической памяти; устройства внешней памяти могут размещаться как в системном блоке компьютера, так и в отдельных корпусах, достигающих иногда размеров небольшого шкафа.

Внутренняя память — электронная (полупроводниковая) память, устанавливаемая на системной плате или на платах расширения.

Внутримашинный системный интерфейс — совокупность унифицированных технических средств, разъёмов и прочего оборудования, используемых для сопряжения устройств в вычислительной системе и программных средств, таких как операционная система, драйверы, утилиты и т. п.

Время доступа — время, которое определяется как усредненный интервал времени от выдачи запроса на передачу блока данных до фактического начала передачи.

Вычислительная система — совокупность аппаратных и программных средств, в окружении которых выполняется результирующая программа, порождаемая системой программирования на основании кода исходной программы, созданного разработчиком, а также объектных модулей и библиотек, входящих в состав системы программирования.

Гибридные вычислительные машины, вычислительные машины комбинированного действия — вычислительные машины, которые работают с информацией, представленной и в цифровой, и в аналоговой форме.

Глобальная компьютерная сеть — компьютерная сеть, охватывающая большие территории и включающая в себя большое число компьютеров.

Домен (в компьютерных сетях, построенных на Microsoft Windows Servers) — минимальная структурная единица организации Active Directory.

Импульсный способ представления сигналов — способ представления, когда единичному значению некоторой двоичной переменной ставится в соответствие наличие импульса (тока или напряжения), нулевому — отсутствие импульса.

Интерфейс — совокупность способов и методов взаимодействия двух систем, устройств или программ для обмена информацией между ними.

Коммутатор — многозначный термин, но применительно к компьютерным сетям это устройство для соединения нескольких узлов или сегментов вычислительной сети.

Контроллер Домена (в компьютерных сетях, построенных на Microsoft Windows Servers) — это сервер, контролирующий область компьютерной сети (домен).

Кэш-память процессора — память процессора, которая в целом отличается меньшим объемом, но более высоким быстродействием по сравнению с основной памятью. Она функционирует как буфер между процессором и основной памятью и содержит копии областей памяти, которые использовались последними, — их адреса и расположенные по этим адресам данные.

Локальная вычислительная сеть — компьютерная сеть, покрывающая относительно небольшую территорию или небольшую группу зданий (дом, офис, фирму, институт).

Маршрутизатор (шлюз, роутер) — это специализированный сетевой компьютер, имеющий как минимум один сетевой интерфейс, который передает пакеты между сетями и отслеживает сети, а не компьютеры.

Межсетевой экран (Firewall, Brandmauer) — комплекс аппаратных или программных средств, осуществляющий контроль и фильтрацию проходящих через него сетевых пакетов в соответствии с заданными правилами.

Микропроцессор — это устройство, представляющее собой одну или несколько больших интегральных схем, выполняющих функции процессора ЭВМ.

Микро-ЭВМ — класс ЭВМ, который возник с появлением микропроцессора. Определяющим признаком микро-ЭВМ является наличие одного или нескольких микропроцессоров.

Мини-ЭВМ — класс ЭВМ, разрабатываемых из требования минимизации стоимости и предназначенных для решения достаточно простых задач.

Многомашинная вычислительная система — вычислительная система, которая содержит некоторое число компьютеров, информационно взаимодействующих между собой. В многомашинных вычислительных системах каждый компьютер работает под управлением своей операционной системы.

Многопользовательские микрокомпьютеры — это мощные компьютеры, оборудованные несколькими видеотерминалами и функционирующие в режиме разделения времени, что позволяет эффективно работать на них сразу нескольким пользователям.

Многопроцессорная вычислительная система — вычислительная система в которой имеется несколько процессоров, информационно взаимодействующих между собой либо на уровне регистров процессорной памяти, либо на уровне оперативной памяти.

Организация системы — внутренняя упорядоченность, согласованность взаимодействия элементов системы, проявляющаяся, в частности, в ограничении разнообразия состояний элементов в рамках системы.

Параграф — единица информации, которая содержит шестнадцать байт, максимальное значение $2^{128} - 1$.

Персональные компьютеры — однопользовательские компьютеры, удовлетворяющие требованиям общедоступности и универсальности применения. Делятся на стационарные и переносные (планшеты, ноутбуки, смартфоны и т. п.).

Последовательный доступ — доступ к внешней памяти, который подразумевает, что каждый блок информации тоже может иметь свой адрес, но для обращения к нему устройство хранения должно сначала найти некоторый маркер начала ленты (тома), после чего последовательным холостым чтением блока за блоком дойти до требуемого места и только тогда производить собственно операции обмена данными.

Потенциальный способ представления сигналов — способ представления, когда единичное значение двоичной переменной отображается высоким уровнем напряжения, а нулевое значение — низким уровнем.

Программное обеспечение — это совокупность программ системы обработки информации и программных документов, необходимых для эксплуатации этих программ.

Прокси-сервер (от англ. proxy — «представитель, уполномоченный») — служба (комплекс программ) в компьютерных сетях, позволяющая клиентам выполнять косвенные запросы к другим сетевым службам.

Прямой доступ — доступ к внешней памяти, который подразумевает возможность обращения к блокам по их адресам в произвольном порядке. Традиционными устройствами с прямым доступом являются дисковые накопители, и часто в понятие «диск», или «дисковое устройство» (disk device), вкладывают значение «устройство внешней памяти прямого доступа».

Рабочие станции — однопользовательские компьютеры, часто специализированные для выполнения определенного вида работ (графических, инженерных, издательских и т. д.).

Сегментация памяти — механизм управления памятью, когда память делится на сегменты для изолирования индивидуального кода, данных и стека. Таким образом, несколько программ могут выполняться одновременно на одном процессоре, не перекрывая друг друга.

Серверы — многопользовательские мощные компьютеры в вычислительных сетях, выделенные для обработки запросов от всех рабочих станций сети.

Сетевая модель OSI (Open Systems Interconnection Basic Reference Model) — семиуровневая модель взаимодействия открытых систем.

Сетевые компьютеры — упрощенные компьютеры, обеспечивающие работу в сети и доступ к сетевым ресурсам, часто специализированные на выполнение определенного вида работ (защита сети от несанкционированного доступа, организация просмотра сетевых ресурсов, электронной почты и т. д.).

Сеть периметра — термин, используемый для изоляции внешних ресурсов компании, которые должны быть доступны и из локальной сети, и из сети Интернет.

Символьные устройства — устройства, которые используются для приема или передачи потока символов без какой-либо блочной структуры (принтеры, сетевые карты, мыши и т. д.).

Система — множество элементов, находящихся в отношениях и связях друг с другом, которое образует определённую целостность, единство.

Скорость записи и считывания — скорость, которая определяется как отношение объема записываемых или считываемых данных ко времени, затрачиваемому на эту операцию.

Скорость передачи данных — скорость, которая определяется как производительность обмена данными после выполнения поиска данных.

Слово — единица информации, которая содержит 2 байта или 16 бит, максимальное значение 65 535 ($2^{16} - 1$).

Служба каталогов (в контексте компьютерных сетей) — программный комплекс, позволяющий администратору работать с упорядоченным по ряду признаков массивом информации о сетевых ресурсах (общие папки, серверы печати, принтеры, пользователи и т. д.), хранящимся в едином месте, что обеспечивает централизованное управление как самими ресурсами, так и информацией о них, а также позволяющий контролировать использование их третьими лицами.

Средние ЭВМ — класс ЭВМ, который возник в результате компромисса цена/быстродействие. В настоящее время трудно определить четкую грань между средними ЭВМ и большими, с одной стороны, и малыми — с другой.

Стек — специальная область памяти, предназначенная для временного хранения данных и организованная по принципу «первым вошел, последним вышел».

Структура системы — состав, порядок и принципы взаимодействия элементов системы, определяющие основные свойства системы.

Супер-ЭВМ — вычислительные системы, из категории больших ЭВМ, превосходящие по вычислительным характеристикам остальные системы.

Таблица описания разделов (partition table) — таблица, в которой содержится описание размещения и характеристики имеющихся на винчестере разделов.

Трансляция страниц — механизм для реализации виртуальной памяти с подкачкой страниц по запросу, где части программы отображаются, как необходимо, на физическую память.

Файл подкачки — файл, хранящийся на диске, который используется компьютером для увеличения объема физического хранилища виртуальной памяти.

Целостность системы — принципиальная несводимость свойств системы к сумме свойств отдельных ее элементов и, в то же время, зависимость свойств каждого элемента от его места и функции внутри системы.

Цифровые вычислительные машины — вычислительные машины, работающие с информацией, представленной в дискретном (цифровом) виде.

Электронная вычислительная машина — комплекс технических средств, предназначенных для автоматической обработки информации в процессе решения вычислительных и информационных задач.

Элемент системы — часть системы, имеющая определенное функциональное назначение. Сложные элементы систем, в свою очередь состоящие из более простых взаимосвязанных элементов, часто называют подсистемами.

AMD64, Intel 64, x86-64, x64, EM64T — разные названия 64-разрядной архитектуры компьютеров, разработанной фирмой AMD.

CISC-архитектура (Complex Instruction Set Computing — компьютер с полным набором команд) — концепция проектирования процессоров, которая обладает следующими свойствами: имеет большое число различных по формату и длине команд, длины команды имеют нефиксированное значение; поддерживается кодировка арифметических действий в одной команде и большое количество режимов адресации; небольшое число регистров, каждый из которых выполняет строго определённую функцию.

IA64 — 64-разрядная архитектура компьютеров, разработанная фирмой Intel.

LH-порядок следования байт — адресация памяти, когда адрес слова указывает на младший байт L (Low), а старший байт H (High) размещается по адресу, на единицу большему.

RISC-архитектура (Restricted Instruction Set Computer — компьютер с сокращённым набором команд) — архитектура процессора, в которой быстродействие увеличивается за счёт упрощения инструкций, чтобы их декодирование было более простым, а время выполнения — меньшим.

x86 (Intel 80x86, IA-32, Intel Architecture-32) — архитектура процессора с одноимённым набором команд, впервые реализованная в процессорах компании Intel.

Учебное издание

Гриценко Юрий Борисович

ВЫЧИСЛИТЕЛЬНЫЕ СИСТЕМЫ,
СЕТИ И ТЕЛЕКОММУНИКАЦИИ

Учебное пособие

Корректор Осипова Е. А.
Компьютерная верстка Табикенова С. Е.

Издано в Томском государственном университете
систем управления и радиоэлектроники.
634050, г. Томск, пр. Ленина, 40
Тел. (3822) 533018.