



Б А К А Л А В Р И А Т

Н.И. ИОПА

ИНФОРМАТИКА

(ДЛЯ ТЕХНИЧЕСКИХ НАПРАВЛЕНИЙ)

Рекомендовано Учебно-методическим объединением вузов
по университетскому политехническому образованию
в качестве **учебного пособия**

Второе издание, стереотипное



МОСКВА
2012

УДК 004(075.8)
ББК 32.973я73
И75

Рецензенты:

В.Ф. Корнюшко, заведующий кафедрой информационных технологий Московской государственной академии тонких химических технологий, засл. деят. науки РФ, д-р техн. наук, проф.,

А.М. Губарь, доц. кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана, ст. науч. сотр., канд. техн. наук,

Г.С. Иванова, доц. кафедры «Компьютерные системы и сети» МГТУ им. Н.Э. Баумана, д-р техн. наук

Иопа Н.И.

И75 Информатика (для технических направлений) : учебное пособие / Н.И. Иопа. — 2-е изд., стер. — М. : КНОРУС, 2012. — 472 с. — (Бакалавриат).

ISBN 978-5-406-02408-9

Излагаются основные понятия и определения, составляющие основу индустрии информатики, рассматриваются основные свойства информации, оценка ее количества и качества, задачи и методы обработки с использованием новых технологий, теоретические вопросы, связанные с передачей информации, а также технические средства передачи и их особенности.

Пособие включает 16 лабораторных работ, посвященных офисным средствам обработки информации, работе в Интернете, а также представлению и обращению различных видов численных данных на компьютере.

В основу пособия положен курс лекций по данной дисциплине, читаемый автором в течение нескольких лет в Рязанском государственном радиотехническом университете для студентов, обучающихся по направлению «Информатика и вычислительная техника».

Для студентов дневного и заочного отделений бакалавриата и магистратуры, обучающихся по направлениям 230100 «Информатика и вычислительная техника», 090100 «Информационная безопасность». Может быть полезно студентам других направлений и специальностей, а также пользователям, занимающимся компьютерной обработкой информации.

УДК 004(075.8)

ББК 32.973я73

Иопа Николай Иванович

**ИНФОРМАТИКА
(ДЛЯ ТЕХНИЧЕСКИХ НАПРАВЛЕНИЙ)**

Сертификат соответствия № РОСС RU. АЕ51. Н 15407 от 31.05.2011 г.

Изд. № 5111. Подписано в печать 28.03.2012. Формат 60×90/16.

Гарнитура «NewtonС». Печать офсетная.

Усл. печ. л. 29,5. Уч.-изд. л. 17,7. Тираж 1000 экз. Заказ №

ООО «КноРус».

129085, Москва, проспект Мира, д. 105, стр. 1.

Тел.: (495) 741-46-28.

E-mail: office@knorus.ru <http://www.knorus.ru>

Отпечатано в ОАО «Первая Образцовая типография»,
филиал «УЛЬЯНОВСКИЙ ДОМ ПЕЧАТИ».

432980, г. Ульяновск, ул. Гончарова, 14.

ISBN 978-5-406-02408-9

© Иопа Н.И., 2012

© ООО «КноРус», 2012

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

1. ЧТО ТАКОЕ ИНФОРМАТИКА

- 1.1. Информатика — фундамент общественного п.....
- 1.2. Определение информатики
- 1.3. Роль информатики в развитии общества

2. ИНФОРМАЦИЯ

- 2.1. Свойства информации
- 2.2. Виды информации и соответствующие им мер.....
- 2.3. Информация и данные
- 2.4. Общая характеристика процесса сбора, переда.....
и накопления информации
- 2.4.1. Операции с данными
- 2.4.2. Задачи и методы обработки информац.....
- 2.5. Информационные системы
- 2.6. Новые информационные технологии
- 2.7. Оценка информации
- 2.7.1. Общая постановка задачи
- 2.7.2. Количественная оценка синтаксическо.....
- 2.7.3. Особенности представления различных.....
- 2.7.4. Вероятностная оценка количества инф.....
- 2.7.5. Связь «энтропия — информация»
- 2.8. Подходы к определению прагматических пок.....
- 2.8.1. Оценка качества информации
- 2.8.2. Оценка важности информации
- 2.8.3. Оценка адекватности информации

3.2.4.	Периодическая последовательность ра
3.2.5.	Последовательность радиолокационн
3.2.6.	Спектр непериодического колебания
3.2.7.	О значении частотного представления
3.3.	Системы передачи данных
3.3.1.	Простейшая модель системы передачи Основные задачи передачи информац
3.3.2.	Данные и сигналы
3.4.	Передача данных на расстояние
3.4.1.	Проблемы передачи. Прямая и косвен
3.4.2.	Способы передачи данных
3.4.3.	О выборе способа передачи данных ..
3.4.4.	Задачи транспортировки сигнала по л
3.4.5.	Типы линий связи
3.4.6.	Беспроводная связь
3.4.7.	Проводная связь
3.5.	Модуляция. Общая модель процесса
3.5.1.	Амплитудная модуляция
3.5.2.	Частотная модуляция
3.5.3.	Фазовая модуляция
3.5.4.	Цифровое кодирование (передача циф с помощью аналоговых сигналов)
3.6.	Кодирование информации
3.6.1.	Типы кодов
3.6.2.	Кодирование как средство контроля п информации
3.6.3.	Систематические коды
3.6.4.	Оценка корректирующей способности
3.6.5.	Декодирование данных
3.6.6.	Методы эффективного кодирования ..
3.7.	Сжатие данных

3.10.	Сетевые технологии передачи данных	144
3.10.1.	Определение и примеры	144
3.10.2.	Об использовании сетевых технологий	145
3.10.3.	Глобальные сети	146
3.10.4.	Локальные сети	147
3.10.5.	Области применения локальных сетей	150
3.10.6.	Многоуровневая организация сетей	150
4.	СРЕДСТВА И МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ	
4.1.	Общая характеристика вычислительных средств обработки информации	158
4.2.	Этапы решения задачи на ЭВМ	165
4.3.	Алгоритмы. Основные требования к алгоритмам	167
4.4.	Компьютер как средство автоматизированной обработки информации	169
4.4.1.	Основные понятия и определения	169
4.4.2.	Принципы организации ЭВМ	170
4.4.3.	Представление информации в ЭВМ	172
4.4.4.	Команды в ЭВМ	197
4.5.	Логические основы построения ЭВМ	198
4.5.1.	Алгебра логики. Основные понятия	198
4.5.2.	Представление функций алгебры логики	200
4.5.3.	Комбинационные схемы	201
4.5.4.	Пример практического применения алгебры логики	204
4.5.5.	Цифровые автоматы	205
4.6.	Структурная организация ЭВМ	209
4.6.1.	Структурная схема фон неймановской ЭВМ	209
4.6.2.	Функционально-регистрационная организация ЭВМ	212
4.7.	Хранение информации	221
4.7.1.	Общая характеристика устройств хранения информации	221
4.7.2.	Основные характеристики памяти	223
4.7.3.	Классификация основной памяти	224
4.7.4.	Логическая организация основной памяти	227
4.7.5.	Память — основной ресурс ЭВМ	229
4.8.	Организация ввода-вывода информации в ЭВМ	230
4.9.	Основные характеристики ЭВМ	233
4.10.	Структурная организация ПК (IBM PC)	234
4.11.	Программное обеспечение	241
4.11.1.	Системное программное обеспечение ЭВМ	242
4.11.2.	Инструментарий технологии программирования	248
4.11.3.	Прикладное программное обеспечение (пакеты прикладных программ)	252
4.12.	Выполнение арифметических и логических операций в ЭВМ	254
4.12.1.	Сложение целых двоичных чисел	255
4.12.2.	Вычитание целых двоичных чисел	257

4.12.3. Умножение целых двоичных чисел	258
4.12.4. Деление целых двоичных чисел	262
4.12.5. Выполнение в ЭВМ логических операций	267
ЛАБОРАТОРНЫЙ ПРАКТИКУМ	
Основные требования к выполнению лабораторных работ	271
Лабораторная работа 1. Основы работы в Windows 2000/XP:	
изучение интерфейсных элементов и работа с ними	
1.1. Цель работы	273
1.2. Порядок выполнения работы	273
1.3. Содержание отчета	274
1.4. Компьютер и пользователь. Место и роль операционной системы	274
1.5. Основные сведения об операционной системе	276
1.6. Операционная система Windows 2000/XP	278
1.7. Интерфейсные элементы Windows: меню, окна, ярлыки, курсор мыши, списки, переключатели	281
1.8. Практическая работа на ПК	285
Контрольные вопросы	289
Лабораторная работа 2. Работа в Windows с файлами и каталогами	
в режиме командной строки	
2.1. Цель работы	290
2.2. Порядок выполнения работы	290
2.3. Содержание отчета	291
2.4. Организация данных. Основные понятия	291
2.5. Режим командной строки ОС Windows 2000/XP	294
2.6. Практическая работа на ПК	295
Контрольные вопросы	301
Лабораторная работа 3. Основы работы в Windows 2000/XP:	
файлы и папки	
3.1. Цель работы	302
3.2. Порядок выполнения работы	302
3.3. Содержание отчета	303
3.4. Файлы и папки	303
3.5. Работа с файлами и папками	304
3.6. Практическая работа на ПК	310
Контрольные вопросы	311
Лабораторная работа 4. Основы работы в Windows 2000/XP:	
стандартные программы Блокнот и Paint	
4.1. Цель работы	312
4.2. Порядок выполнения работы	312
4.3. Содержание отчета	313
4.4. Стандартные программы Блокнот и Paint	313
4.5. Текстовый редактор Блокнот	314
4.6. Графический редактор Paint	316

4.7. Практическая работа на ПК	318
Контрольные вопросы	320
Лабораторная работа 5. Основы работы на ПК под управлением операционной оболочки Far Manager	
5.1. Цель работы	321
5.2. Порядок выполнения работы	321
5.3. Содержание отчета	321
5.4. Основные сведения об операционной оболочке FAR Manager	322
5.5. Практическая работа на ПК	324
Контрольные вопросы	337
Лабораторная работа 6. Защита от компьютерных вирусов	
6.1. Цель работы	338
6.2. Порядок выполнения работы	338
6.3. Содержание отчета	338
6.4. Общие сведения о вирусах	339
6.5. Описание программы Kaspersky Anti-Virus Scanner	342
6.6. Практическая часть	347
Контрольные вопросы	349
Лабораторная работа 7. Текстовый редактор Microsoft Word 2002 (основы работы)	
7.1. Цель работы	349
7.2. Порядок выполнения работы	349
7.3. Содержание отчета	350
7.4. Общие сведения о текстовом редакторе Microsoft Word	350
7.5. Запуск текстового редактора MS Word	351
7.6. Среда редактирования MS Word	351
7.7. Основные операции с документами	354
7.8. Практическая работа на ПК	362
Контрольные вопросы	366
Лабораторная работа 8. Сжатие информации в ПК	
8.1. Цель работы	367
8.2. Порядок выполнения работы	367
8.3. Содержание отчета	367
8.4. Теоретическая часть	368
8.5. Программа-архиватор WinRAR	371
8.6. Практическая работа на ПК	375
Контрольные вопросы	377
Лабораторная работа 9. Табличный процессор Microsoft Excel (основы работы)	
9.1. Цель работы	377
9.2. Порядок выполнения работы	378
9.3. Содержание отчета	378
9.4. Общие сведения о табличном процессоре MS Excel	379

9.5. Работа с процессором MS Excel	380
9.6. Создание таблиц	382
9.7. Аппарат формул и функций MS Excel	386
9.8. Форматирование данных	388
9.9. Построение диаграмм	389
9.10. Сохранение файла	393
9.11. Пример выполнения задания	393
9.12. Практическая работа на ПК	398
Контрольные вопросы	400
Лабораторная работа 10. Работа в Интернете (Поиск информации.	
Электронная почта)	
10.1. Цель работы	400
10.2. Порядок выполнения работы	400
10.3. Содержание отчета	401
10.4. Основные сведения об Интернете	401
10.5. Практическая работа на ПК	404
10.6. Индивидуальные задания	408
Контрольные вопросы	408
Лабораторные работы 11—12. Целые числа: способы представления	
и хранения в ЭВМ, основные операции обращения с числами	
11.1. Цель работы	409
11.2. Порядок выполнения работы	409
11.3. Содержание отчета	409
11.4. Методические указания	410
11.5. Практическая работа на ПК	416
Контрольные вопросы	426
Лабораторные работы 13—14. Вещественные числа: способы	
представления и хранения в ЭВМ	
13.1. Цель работы	427
13.2. Порядок выполнения работы	427
13.3. Содержание отчета	428
13.4. Методические указания	428
13.5. Практическая работа на ПК	433
13.6. Порядок выполнения заданий	435
Контрольные вопросы	439
Лабораторная работа 15. Форматы команд. Способы	
адресации данных	
15.1. Цель работы	440
15.2. Порядок выполнения работы	440
15.3. Содержание отчета	441
15.4. Методические указания	441
15.5. Команды программы Far Manager и отладчика TD персональной ЭВМ ...	445
15.6. Практическая работа на ПК (работа с командами)	449
Контрольные вопросы	452

Лабораторная работа 16. Структура компьютера, принцип функционирования. Выполнение арифметических операций на компьютере	
16.1. Цель работы	452
16.2. Порядок выполнения работы	452
16.3. Содержание отчета	453
16.4. Методические указания	454
16.5. Программная модель МП	458
16.6. Логическая организация памяти	460
16.7. Состав системного ПО ПК	460
16.8. Практическая работа на ПК	461
16.9. Отладчик TD	464
16.10. Варианты заданий	464
16.11. Порядок выполнения работы	465
Контрольные вопросы	468
ЛИТЕРАТУРА	470

ВВЕДЕНИЕ

Вся наша жизнь так или иначе связана с накоплением и обработкой информации, которую мы получаем из окружающего мира. Благодаря научно-техническому прогрессу наши возможности познавать и изучать этот мир постоянно расширяются: появляются новые носители информации и средства ее обработки, новые технологии, растет не только количество информации, но и ее роль.

На современном этапе развития общества в системах обработки информации широко используются персональные компьютеры, микропроцессоры и принципиально новая организация обработки информации — *распределенная*, максимально приближающая вычислительные ресурсы к пользователю.

Будучи аккумулятивной и обработанной, информация дает новые сведения, приводит к новому знанию.

Без информации не может быть жизни в любой форме и не могут функционировать созданные человеком любые информационные системы.

Сегодня для нормального функционирования организации любого масштаба уже недостаточно только традиционных для индустриального общества ресурсов, таких как:

- ♦ природные — объекты, процессы, природные условия, используемые для удовлетворения материальных и духовных потребностей людей;
- ♦ трудовые — люди, обладающие навыками и знаниями для работы в обществе;
- ♦ финансовые — денежные средства, находящиеся в распоряжении государственных и коммерческих структур;
- ♦ энергетические — носители энергии (уголь, нефть, газ, электричество и т.д.) и др.

Чтобы знать, как наиболее эффективно использовать эти ресурсы, необходимо иметь информацию о технологиях. Поэтому информация стала существенным ресурсом. *Информационные ресурсы* — это знания, накопленные людьми для социального использования в обществе. Эти знания зафиксированы и материализованы в виде докумен-

тов, баз данных, баз знаний, алгоритмов, компьютерных программ, произведений литературы, науки, искусства. Информационные ресурсы в настоящее время рассматриваются как отдельная экономическая категория, важнейший стратегический ресурс общества. Их основная особенность состоит в том, что по мере развития общества и роста потребления информации в отличие от других видов ресурсов (материальных, природных и др.) их запасы не убывают, а увеличиваются.

Резкое возрастание роли информации вызвало сегодня неизбежность процесса *информатизации общества* — процесса, обуславливающего переход от индустриального общества к информационному.

Информатизацию можно рассматривать как процесс преобразования производственно-хозяйственных, научных и социально-бытовых структур путем производства информации, необходимой для выработки и реализации решений, направленных на достижение качественно новых результатов деятельности человека, на базе внедрения и использования средств вычислительной техники и информатики, средств связи и информационных технологий. В процессе информатизации общества происходит проникновение информационных технологий во все сферы жизнедеятельности общества, в том числе и связанные с принятием ответственных решений. Вычислительная техника все более широко внедряется в системы управления такими технологическими процессами, как системы управления вооружением, атомными реакторами и т.п. Отличительная особенность таких систем управления — необходимость выполнения сложных видов обработки больших объемов информации в крайне ограниченные сроки. В результате сложность систем неуклонно растет. Информационные технологии уже сейчас оказывают решающее воздействие на многие сферы деятельности людей, человеческих коллективов и общество в целом.

Базовой технической составляющей процесса информатизации общества является *компьютеризация*, под которой понимают развитие и широкое внедрение *компьютеров*, обеспечивающих оперативное получение результатов переработки информации и ее накопление.

Научный фундамент процесса информатизации общества составляет информатика, призванная создавать новые информационные технологии и системы для решения задач информатизации.

Настоящая книга представляет собой учебное пособие по курсу «Информатика» — одному из важнейших компонентов государственного образовательного стандарта бакалавров, инженеров и магистров по специальности «Информатика и вычислительная техника». Курс

охватывает все основные разделы информатики и направлен на формирование знаний и умений, которые образуют теоретический фундамент, необходимый для корректной постановки и решения проблем в области информатики.

В книге рассмотрены теоретическая (разделы 1, 2) и компьютерная (разделы 3, 4 и лабораторный практикум) составляющие информатики. В *теоретической* информатике описываются основные понятия и определения, свойства и характеристики информации, количественная и качественная оценки, информационные системы и новые информационные технологии. В *компьютерной* информатике неразрывно связаны между собой средства передачи и преобразования информации, алгоритмические средства обработки информации и вычислительные средства.

Разделы 3, 4 посвящены средствам и способам передачи информации, задачам и методам ее обработки с помощью средств вычислительной техники. Рассматриваются подходы к решению задач на компьютере, его структурно-функциональная организация, классификация, характеристики, организация памяти, программное обеспечение, выполнение арифметических и логических операций, теоретические и практические вопросы, связанные с передачей информации.

В лабораторном практикуме приведены 16 лабораторных работ по операционным системам и оболочкам, компьютерным вирусам и средствам защиты от них, а также по текстовым, табличным и графическим редакторам, архивированию данных, по работе в Интернете, обработке и хранению различных видов данных на компьютере. Каждая из этих работ содержит теоретическую часть, описывающую основу изучаемого вопроса, конкретные практические рекомендации по выполнению, примеры выполнения практических заданий. В заключительной части работы имеется индивидуальное задание, позволяющее пользователю убедиться, насколько глубоко он освоил тот или иной раздел. В ходе выполнения лабораторной работы пользователь получает навыки практической работы на компьютере. Индивидуальные задания и практические рекомендации составлены так, чтобы пользователь смог максимально быстро освоить изучаемый раздел.

Для выполнения лабораторных работ, связанных с представлением, преобразованием различных типов числовых данных и отладкой ассемблерных программ (ЛР 11—16), необходимо воспользоваться компьютером, подключенным к сети Интернет, и посетить сайт Рязанского государственного радиотехнического университета по ссылке www.rsreu.ru/content/view/1729/1288.

Содержание пособия соответствует курсу лекций, которые автор читает для студентов дневного отделения по специальности «Информатика и вычислительная техника». Усвоение материала курса должно обеспечить основу, позволяющую читателю воспринимать и усваивать многие другие общетехнические и специальные дисциплины по информационным технологиям и вычислительным средствам, входящим в учебные планы по таким специальностям, как «Организация ЭВМ и систем», «Основы информационных технологий», «Сети ЭВМ и телекоммуникации» и др.

1. ЧТО ТАКОЕ ИНФОРМАТИКА

1.1. ИНФОРМАТИКА — ФУНДАМЕНТ ОБЩЕСТВЕННОГО ПРОГРЕССА

Информатику можно рассматривать как многоаспектную, всеохватывающую среду человеческой деятельности, связанную с получением, обработкой, хранением и использованием различных типов информации.

Она объединяет разнообразные по природе и сущности технические процессы и средства, связанные с удовлетворением информационных потребностей общества в настоящем и будущем. Области применения информатики постоянно расширяются — появились экономическая, медицинская, социальная и другие информатики. Большая часть информации сегодня обрабатывается с помощью компьютеров и хранится в их памяти.

По последним данным, в мире работает около полумиллиарда компьютеров, и число их продолжает удваиваться в среднем каждые три года. При этом в среднем 1 раз в полтора года в 2 раза повышаются основные характеристики аппаратных средств (быстродействие, объем памяти и др.), 1 раз в два-три года обновляются поколения компьютеров и программное обеспечение и 1 раз в пять—семь лет меняется база стандартов интерфейсов и протоколов. Под *протоколом* понимают специальные средства для работающего в вычислительной сети компьютера, которые обеспечивают «взаимопонимание» всех компьютеров, включенных в сеть передачи и обработки данных.

Растет число домашних персональных компьютеров (ПК), которые могут быть подключены к широко используемой информационно-вычислительной сети Интернет. Сетевая инфотехнология обеспечивает доступ к обширным *информационным ресурсам* (ИР), находящимся в компьютерах различных классов и типов во всем мире, подключенных к Интернету. Каждый десятый взрослый человек планеты использует компьютер постоянно.

Во многих технически развитых странах умение работать с компьютером считается уже не компьютерной грамотностью, а просто грамотностью. Сбылись пророческие слова академика В.М. Глушкова (1923—1982), бывшего директора Киевского института кибернетики,

который более 40 лет назад писал: «Развитие сети ЭВМ и терминального доступа к ним приводит к тому, что часть информации научно-технической, экономической, социальной и другой помещается в память машины. К началу нового тысячелетия в технически развитых странах основная масса информации будет храниться в безбумажном виде, в памяти ЭВМ. Тем самым человек, который в начале XXI в. не будет уметь пользоваться этой информацией, уподобится человеку XX в., не умеющему ни читать, ни писать».

Сегодня нет ни одной отрасли науки и техники, которая развивалась бы так же стремительно, как информатика.

Фактически в последние годы на наших глазах произошла компьютерная революция, затронувшая все сферы социальной, культурной, научной и производственной деятельности людей. Созданы предпосылки для перехода от технологического общества к *информационному*, в котором:

- ♦ объектом и результатом труда большинства занятого населения являются не материальные ценности, а главным образом информация и научные знания;
- ♦ информация выступает *товаром* и наряду с информационными технологиями занимает ключевые позиции в экономике страны и в определении темпов ее экономического роста;
- ♦ любой гражданин может получить любую информацию, кроме государственных тайн, коммерческих и личных секретов;
- ♦ существует сеть гигантских взаимосвязанных накопителей информации, т.е. общедоступных банков знаний и данных, а также необходимых технологических средств, таких как компьютеры, программы, интерактивные системы связи и др.;
- ♦ особую роль играет компьютер.

Ключевую роль на этапе формирования информационного общества играет *информатика*. Члены информационного общества в той или иной мере связаны с процессами переработки информации и поэтому вынуждены осваивать и практически использовать соответствующие этим процессам *новые информационные технологии* — современные способы сбора, хранения, обработки и передачи информации на основе средств вычислительной техники (ВТ) и связи. Так, на большинстве крупных предприятий в Рязани (ЗИЛ, Приборный завод, Управление распределением ресурсов и др.) используется информационно-поисковая система для складских помещений, позволяющая собирать, хранить и обрабатывать информацию о громадной номенклатуре деталей, храня-

щихся на складе. Движением крана-штабелера управляет компьютер, вычисляющий оптимальную траекторию его движения и выдающий соответствующие сигналы на привод. Информация обо всей номенклатуре деталей и их перемещении также хранится на машиночитаемых носителях и обрабатывается компьютером.

Еще одним примером компьютерной обработки информации является система «Сирена» для резервирования мест на авиарейсы, обеспечивающая хранение большого объема данных о наличии свободных и проданных мест, быстрый поиск необходимой информации о проданных местах, обмен информацией между центральным компьютером и удаленными от него терминалами, установленными на рабочих местах кассиров-операторов.

1.2. ОПРЕДЕЛЕНИЕ ИНФОРМАТИКИ

Известны две концепции в подходе к определению информатики. Сторонники *первой* — академики Н.Н. Моисеев, А.А. Дородницын, А.П. Ершов и другие — исходят из определяющей роли ВТ и *сущность информатики* определяют как повсеместное внедрение ВТ во все сферы деятельности общества, отождествляя ее со всеобщей компьютеризацией. За отправную точку *информатики* они принимают ЭВМ и построенные на их основе автоматизированные системы обработки информации.

Согласно *второй* концепции — концепции академика В.М. Глушкова — такое понимание информатики является узким и отражает только одну, и не самую главную, сторону данного понятия. Вычислительная техника в информатике служит лишь высокоэффективным и в современных условиях незаменимым *средством обработки* информации, т.е. ни появление ЭВМ, ни их тотальное использование не влияют на сущность понятия информатики, а являются лишь его следствием. Задача накопления, обработки и распространения информации стояла перед человечеством на всех этапах его развития, начиная с эпохи, когда основным носителем информации была бумага, до появления ЭВМ и систем обработки информации на их основе. *Информатика* и до появления ЭВМ, и после была и остается одним и тем же атрибутом деятельности общества. Только до появления ЭВМ она была бумажной, а с появлением ЭВМ становится безбумажной (*компьютерной*).

По В.М. Глушкову, информатика есть некоторая *методология* работы с информацией, присущая любому сколько-нибудь цивилизованному обществу независимо от того, какими средствами обработки оно

располагает. И *сущность информатики* заключается в формировании такой информационной среды, в которой бы имелись все объективные предпосылки, необходимые для наиболее рационального информационного обеспечения деятельности во всех сферах. Естественно, что создание такой среды в современных условиях предполагает всеобщую компьютеризацию, но она рассматривается лишь как компонент, хотя и очень важный, общей инфраструктуры информационной среды.

Развитие и эффективное использование ВТ оказали сильное влияние на формирование информатики и выделение ее в самостоятельное направление. Но это только внешние проявления скрытого процесса. Действительной причиной появления не только ВТ, но и информатики послужили *информационные потребности* общества и объективная необходимость резкого повышения эффективности обработки информации.

Таким образом, *информатика* не сводится только к проблемам автоматизации обработки информации с помощью средств вычислительной техники, а *представляет собой научно-методологический базис рационального решения информационных проблем общества.*

Из анализа обеих концепций становится ясно, что *информатика* — это комплекс научных, технических, социальных и других проблем, связанных с методами и средствами получения и обработки информации в различных областях человеческой деятельности, включающий различные разделы современных знаний и технологий:

- ♦ создание новых средств накопления информации на машиночитаемых носителях — магнитных дисках (МД), магнитных лентах (МЛ), оптических дисках;
- ♦ развитие средств связи, обеспечивающих доставку информации в любую точку системы без существенных ограничений во времени и расстоянии;
- ♦ возможность автоматизированной обработки информации с помощью ЭВМ по заданным алгоритмам (сортировка, классификация, представление в нужной форме, преобразование).

Под понятием «информатика» объединяют ряд научных направлений, исследующих разные стороны одного и того же объекта — информации:

- ♦ теоретическая информатика;
- ♦ кибернетика;
- ♦ программирование;
- ♦ искусственный интеллект;
- ♦ информационные системы;
- ♦ вычислительная техника.

Предметом изучения *теоретической информатики* являются математические методы построения моделей обработки, передачи и использования информации. Эти методы опираются на математическую логику, что служит основой для теории алгоритмов, теории автоматов, теории сетей.

В информатике как в области деятельности можно выделить три направления:

- ♦ научное;
- ♦ промышленное;
- ♦ социальное.

Научное занимается фундаментальным исследованием процессов получения, передачи, хранения и обработки информации, а также разработкой технических и программных средств для оперирования информацией; *промышленное* — массовым производством компьютеров и других технических средств информатики, а *социальное* изучает и решает социальные вопросы информатизации общества, в том числе и вопросы компьютеризации учебного процесса.

В настоящее время бурно развивается один из разделов информатики — *компьютерная информатика*. Это естественно-научная дисциплина, занимающаяся вопросами сбора, хранения, обработки и отображения информации с использованием средств ВТ.

Сегодня это одно из стратегических направлений развития общества, в котором информация является таким же стратегическим ресурсом, как полезные ископаемые, электроэнергия и другие, а информационные технологии обеспечивают ее эффективное использование в различных сферах человеческой деятельности.

Вместе с тем широкая компьютеризация породила целый ряд важных проблем — технических, экономических, социальных и других, требующих обязательного решения. Прежде всего следует выделить проблему обучения широких масс пользователей основам компьютерной грамотности, учитывая разносторонний характер как компьютерных приложений, так и самих пользователей.

Информационная политика будущего представляется сегодня как подключение коллективных и индивидуальных абонентов, находящихся в самых разных точках мира, к разветвленным компьютерным системам, основанным на гигантских банках данных или базах знаний и способным выдавать информацию в удобном для абонентов виде.

Сложная система, называемая *банком данных* или системой баз данных (СБД), включает в себя собственно *базу данных* (БД) — средство накопления и организации больших массивов информации об объектах

некоторой предметной области, а также программные, технические, языковые и организационно-методические средства, обеспечивающие централизованное накопление и коллективное многоцелевое использование данных. Одна из компонент СБД — *система управления БД (СУБД)* — совокупность языковых и программных средств, с помощью которых БД создается и поддерживается в процессе эксплуатации. Типичным примером БД в инженерных приложениях являются, например, каталоги интегральных схем.

В отличие от БД *база знаний (БЗ)* не только хранит множество данных, характеризующих элементы некоторой предметной области, но и отражает отношение между ними. *Знания* в отличие от данных хранят не сведения о каком-либо единичном конкретном факте, а информацию о том, как устроены все факты определенного типа.

1.3. РОЛЬ ИНФОРМАТИКИ В РАЗВИТИИ ОБЩЕСТВА

Информатика как широкая область научных знаний возникла на стыке нескольких фундаментальных и прикладных наук, в том числе:

- ♦ *философии и психологии* (учение об информации и теория познания);
- ♦ *математики* (теория математического моделирования, дискретная математика, математическая логика и теория алгоритмов);
- ♦ *кибернетики* (теория информации и управления);
- ♦ *лингвистики* (учение о формальных языках и о знаковых системах).

Можно выделить задачи, которые являются общими для всех наук при обработке информации о проводимых экспериментах и исследованиях с помощью ЭВМ:

- ♦ сбор данных и запись их в запоминающее устройство в таком виде, который позволяет находить их повторно, считывать и анализировать;
- ♦ обработка данных по разработанным алгоритмам и программам на ЭВМ;
- ♦ преобразование данных к виду, удобному для исследования (графики, таблицы, номограммы и др.);
- ♦ отображение информации;
- ♦ передача результатов другим участникам эксперимента, находящимся на расстоянии.

Научным фундаментом для решения перечисленных задач является *информатика*. Чрезвычайно велика ее роль и в развитии общества. Информатика служит научным фундаментом процесса его информатизации. С ней связывают прогрессивное увеличение возможностей компьютерной техники, развитие информационных сетей, создание новых информационных технологий, которые приводят к значительным изменениям в различных сферах общества — производстве, науке, образовании, медицине и т.д.

2. ИНФОРМАЦИЯ

2.1. СВОЙСТВА ИНФОРМАЦИИ

Информация в широком смысле — это отражение реального мира, а в узком — любые сведения, являющиеся объектом хранения, передачи и преобразования.

Информацию получают из самых разных источников: лекции, доклады, лабораторные работы, практические занятия, научные журналы, книги, радио, телевидение, глобальная вычислительная сеть Интернет, устные сообщения, записки, фотографии и т.д.

Источником информации могут быть как люди, так и техническая система, генерирующая информацию о состоянии x , некоторого физического объекта. Под *генерированием* информации понимают всю совокупность процессов, в результате которых возникает новая информация. Эта информация отсутствует в системе и ее нельзя получить на основе уже имеющейся информации. В процессе генерирования осуществляются целенаправленное извлечение и анализ сведений о каком-либо объекте или процессе. Например, проанализировав зачетную книжку студента, можно получить информацию о том, на каком факультете и по какой специальности он обучается, год поступления, как успевает по общим и специальным предметам, были ли срывы в учебе (пересдачи), сколько лет обучается, какие практики проходил, и др. Таким образом, *информация — это совокупность некоторых сведений о какой-либо материальной системе (предмете, объекте, процессе, событии), предназначенных для передачи, преобразования, хранения или непосредственного использования.*

В описанном примере *источником* информации является зачетная книжка студента. Информация, полученная в результате ее анализа, по запросу может быть представлена в другое учебное заведение или организацию, занимающуюся отбором нужных кадров или интересующую личность студента в иных целях.

Информация может быть:

- ♦ *личной* (касается тех или иных событий в личной жизни человека или группы людей);

- ♦ *массовой* (распространяется через СМИ: радио, телевидение, газеты, журналы и т.д.);
- ♦ *специальной* (научно-техническая, экономическая, медицинская и т.д.). Такая информация используется для осуществления процессов управления различных уровней.

Все три вида информации относятся к социальной информации, связанной с идеологией, культурой, наукой, образованием, управлением, удовлетворением личных нужд и т.д.

Однако информация — это не только сведения из книг и статей, средств массовой информации, но и сведения, хранящиеся в рельефе ключа, в структуре сложной биологической молекулы, в радиосигналах. Информация в рельефе ключа позволяет открыть с его помощью определенный («свой») замок. Информация, заключенная в структуре биологической молекулы, позволяет живой клетке синтезировать определенные белки для новых тканей или для уничтожения попавших в организм микробов. Информация в радиосигналах с Земли включает двигатель на орбитальном спутнике и переводит его на другую орбиту.

Сегодня нельзя назвать ни одной отрасли народного хозяйства, не охваченной информацией. Широкое развитие средств вычислительной техники и связи позволяет собирать, хранить, перерабатывать и передавать информацию в таких объемах и с такой скоростью, которые раньше были немыслимы. За информацией «охотятся», ее скрывают, защищают, получают и передают. Говорят, кто владеет информацией, тот владеет миром.

Производство информационных продуктов и услуг с использованием новых информационных технологий выделилось в отдельную отрасль. Формируется новая структура общества — *информационная*. Таким образом, информация служит важным средством существования общества, жизни каждого человека отдельно и является третьей, не менее важной, сущностью нашего мира после вещества и энергии.

Вместе с тем строгого научного определения информации не существует. В случаях, когда наука не может дать четкого определения какому-то явлению или предмету, пользуются *понятиями* о них. Последние отличаются от определений тем, что разные люди при несходных обстоятельствах могут вкладывать в понятия разный смысл. Так, *в быту* под информацией понимают те сведения, которые могут быть получены в результате общения с кем-либо, из книг либо из средств массовой информации и т.д. *Математик* будет рассматривать

это понятие шире, включая в него и те сведения, которые человек не получал извне, а создавал сам с помощью умозаключений. *Биолог* отнесет к информации не только данные, которые человек получил с помощью органов чувств или создал в уме, но и данные, которые хранит в себе с момента рождения до смерти, — генетический код, благодаря которому дети так похожи на своих родителей.

2.2. ВИДЫ ИНФОРМАЦИИ И СООТВЕТСТВУЮЩИЕ ИМ МЕРЫ

Различают три вида информации:

- ♦ синтаксическую;
- ♦ семантическую;
- ♦ прагматическую.

Синтаксическая информация формально отображает структурные характеристики информации и не затрагивает их смыслового содержания, т.е. способствует восприятию чисто внешних характеристик информации. Такую информацию называют *данными*, так как при ее использовании не имеет значения смысловая сторона. Термин «данные» подразумевает конкретную реализацию информации. Это могут быть факты, признаки, наблюдения, записанные человеком или прибором и фиксирующие результат либо ход процесса. Они могут быть представлены в числовом, буквенном, графическом изображениях и др. Этот вид информации характеризует ее как *продукт*, подлежащий *переработке* в целях информационного обеспечения решаемых задач.

Для обработки данных применяют различные средства, основными из которых являются *средства хранения, передачи и переработки*.

С точки зрения использования этих средств к базовым характеристикам информации относятся *способ представления, объем, скорость передачи и обработки* и др.

Семантическая информация характеризует смысловое содержание сообщения, а *прагматическая* — отражает отношение потребителя к информации, соответствие информации цели управления, которая на ее основе реализуется. Прагматический аспект связан с рассмотрением ценности информации, ее полноты, адекватности и других характеристик, т.е. с *прагматическим* использованием информации.

В соответствии с видами информации различают три меры (рис. 2.1).

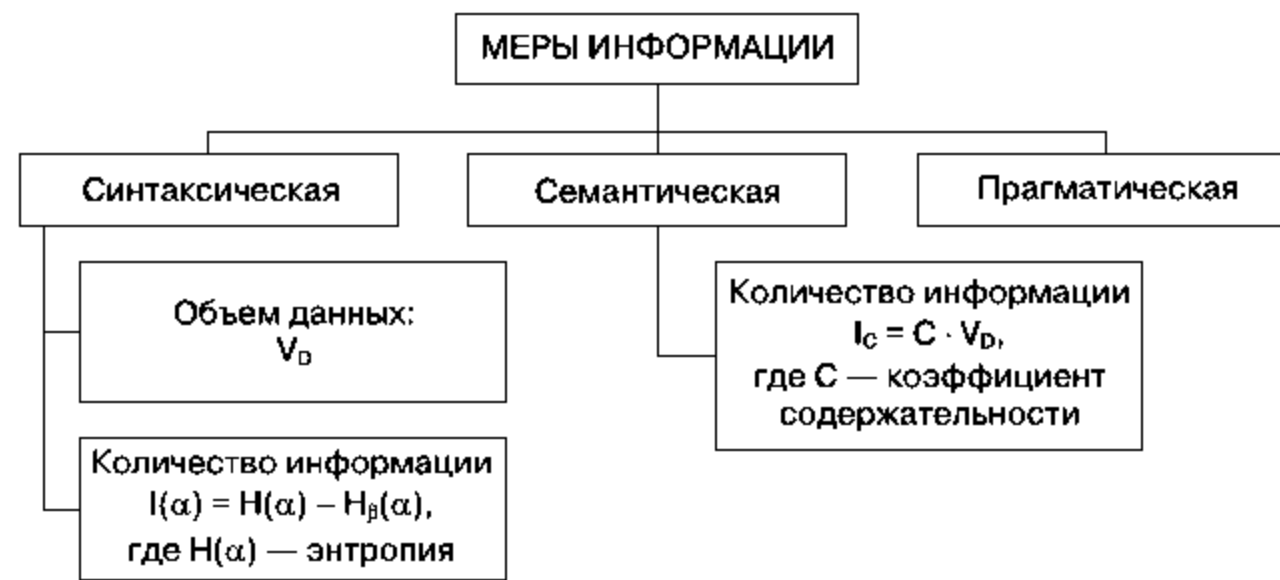


Рис. 2.1. Меры информации

Под количеством информации I понимают объем тех сведений, которые используются в процессе решения задачи, а под V_D — объем данных в сообщении, который измеряется числом символов (разрядов) в нем.

Количество синтаксической информации I — это степень уменьшения неопределенности после выбора (получения) сообщения в виде некоторого сигнала, которая определяется как разность неопределенности до получения сообщения и после:

$$I = H(\alpha) - H_\beta(\alpha),$$

где $H(\alpha)$ — мера неосведомленности потребителя информации о системе α , т.е. априорные (предварительные) сведения;

$H_\beta(\alpha)$ — мера неопределенности после получения сообщения β о системе α ;

$$H(\alpha) = - \sum_{i=1}^n P_i \log P_i,$$

где P_i — вероятность того, что система находится в i -м состоянии.

Для оценки семантической информации I_c используют тезаурусную меру, которая связывает семантические свойства информации со способностью пользователя принимать поступившее сообщение.

Тезаурус — это совокупность сведений, которыми располагает пользователь, или система в конкретной предметной области.

Количество семантической информации I_c , воспринимаемой потребителем, зависит от его тезауруса S_p , т.е. $I_c = f(S_p)$ и описывается кривой, представленной на рис. 2.2.

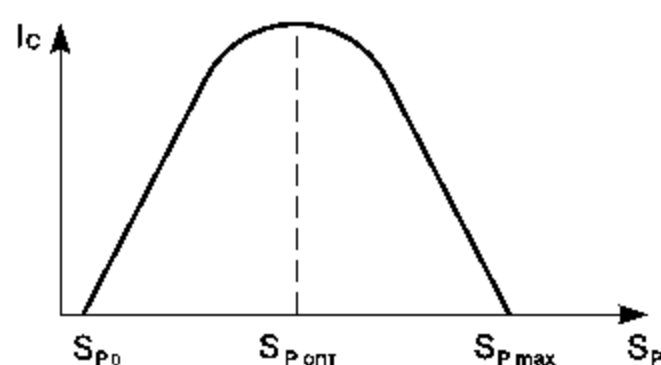


Рис. 2.2. Зависимость количества семантической информации, воспринимаемой потребителем, от его тезауруса $I_c = f(S_p)$

Рассмотрим два предельных случая, когда количество семантической информации $I_c = 0$:

- 1) $S_p = 0$ — пользователь не воспринимает (не понимает) поступающую информацию;
- 2) $S_p \rightarrow S_{p \max}$, т.е. пользователь «все знает» о предметной области, и поступающее сообщение ему не несет никакой информации.

Максимальное количество семантической информации $I_{c \max}$ потребитель приобретает, когда ее смысловое содержание согласуется с его тезаурусом ($S_p = S_{p \text{ опт}}$), а поступающая информация понятна пользователю и содержит ранее не известные сведения, отсутствующие в его тезаурусе.

Следовательно, количество семантической информации в сообщении, т.е. количество новых знаний, получаемых пользователем, является величиной относительной.

Относительной мерой количества семантической информации может служить коэффициент содержательности C :

$$C = \frac{I_c}{V_D},$$

где V_D — объем данных в сообщении (число символов в сообщении).

2.3. ИНФОРМАЦИЯ И ДАННЫЕ

Интуитивно понятия *информация* и *данные* часто отождествляются, хотя на самом деле они существенно различаются.

Данные — это конкретная реализация информации и могут быть представлены, как уже говорилось выше, в числовом, графическом или символьном виде. Так, данные о работе человеческого сердца могут быть представлены в графической форме в виде ЭКГ, снятой специ-

альным прибором — кардиографом. Данные о физических характеристиках человека (рост, вес, объем легких, давление, пульс и т.д.) снимаются с помощью различных приборов, отображаются в медицинской карте в виде отдельных записей и являются объективной характеристикой индивидуума — *данными* о нем и состоянии его здоровья. Их превращение в информацию всецело зависит от специфики проблемы, которую необходимо решить. Так, последовательность импульсов, зарегистрированных на ЭКГ, становится информацией лишь в ходе экспертного заключения кардиолога о работе сердца — здоровое оно или больное, какие отклонения имеют место и насколько они серьезны, какие меры должны быть приняты для лечения или какие физические ограничения могут быть рекомендованы пациенту. Например, призывника могут направить в авиацию, военно-морской флот или другой род войск в зависимости от требований, предъявляемых в этих подразделениях к претендентам (его данные должны соответствовать заданным требованиям).

Автоматизированная система, управляя каким-либо технологическим процессом на основе полученных данных, принимает нужное решение: добавить новые компоненты, поменять пропорции, выдать сигнал тревоги или включить средства пожаротушения.

Таким образом, *данные становятся информацией только при решении конкретной проблемы, т.е. в ходе их потребления*. Иначе, информацией будут лишь те данные, которые устраняют неопределенность в отношении того или иного вопроса и позволяют принять решение, например:

- ♦ разрешить полет экипажу авиалайнера;
- ♦ выйти в рейс водителю автобуса;
- ♦ начать операцию хирургу и др.

Данные, полученные из ЭКГ, отвечают на вопрос, является ли сердце больным или здоровым; конкретные физические данные претендента на некую должность (зрение, слух, кровяное давление и др.) говорят о его профессиональной пригодности.

Превращение данных в информацию осуществляется потребителем (пользователем) на основе собственной информационной модели.

Оценки о текущей успеваемости студентов будут оставаться данными, пока не поступит запрос об успеваемости конкретного студента в конкретном семестре. При ответе на вопрос данные становятся информацией.

Информационной моделью объекта или набора объектов назовем совокупность атрибутов (характеристик) данного объекта (объектов) вместе с числовыми или иными значениями этих атрибутов.

Например, можно составить информационную модель РГРТУ (сайт в Интернете), атрибутами которой будут: дата основания института, дата переименования в академию, число и название факультетов и кафедр, специальностей, число преподавателей и студентов, профессоров и доцентов, число абитуриентов дневного, вечернего и заочного отделений, время обучения разным специальностям и т.д.

Степень информативности данных определяется тем, насколько полно и эффективно с точки зрения затрат времени, усилий и средств они позволяют получить ответ на интересующий вопрос. *Информативность* зависит от содержания и формы представления данных. *Содержание* показывает, в какой мере необходимые для решения задачи сведения или факты отражены в данных. *Форма представления данных* определяет время и усилия, которые необходимо затратить пользователю на получение информации, что влияет на потребительскую ценность и рыночную стоимость информации.

2.4. ОБЩАЯ ХАРАКТЕРИСТИКА ПРОЦЕССА СБОРА, ПЕРЕДАЧИ, ОБРАБОТКИ И НАКОПЛЕНИЯ ИНФОРМАЦИИ

Важная составляющая часть информации — *данные*, представляющие собой зарегистрированные сигналы.

Этими сигналами могут быть изменения формы или параметров качества, изменения электрических, магнитных, оптических характеристик, изменения состояния электронной схемы и др.

В соответствии с методами регистрации данные могут храниться и переноситься на носителях различного вида: бумага, магнитные ленты и диски, оптические диски, CD-ROM и др.

На бумаге данные регистрируются путем изменения оптических характеристик ее поверхности. Изменение оптических свойств используется в CD-ROM-устройствах, осуществляющих запись лазерным лучом на пластмассовых носителях с отражающим покрытием. В магнитных лентах и дисках используют изменение магнитных свойств материала. Со свойствами носителей тесно связаны свойства информации. Любой носитель характеризуется *разрешающей способностью* (количество данных, записанных в принятых для носителя единицах измерения), *динамическим диапазоном* (логарифм отношения максимальной и минимальной амплитуд сигнала):

$$DD = \lg \frac{A_{\max}}{A_{\min}}.$$

Эти характеристики носителя нередко определяют такие свойства информации, как *полнота, достоверность, доступность*. Так, полноту информации проще обеспечить базой данных (БД), размещенных на CD-ROM-дисках, чем на гибких магнитных дисках (ГМД), поскольку в первом случае плотность записи данных на единицу длины дорожки намного выше. Доступность информации в книге для обычного потребителя выше, чем на том же CD-ROM-диске, поскольку не все потребители обладают знаниями и оборудованием для съема информации в электронном виде.

Смена типа носителя связана с задачей *преобразования* данных, относящейся к одной из важнейших задач обработки информации. Действительно, в структуре стоимости вычислительной системы (ВС) стоимость устройства ввода-вывода (УВВ) данных, работающего с носителем информации, составляет половину стоимости аппаратных средств.

2.4.1. Операции с данными

Обработка данных включает в себя множество операций, таких как: *сбор, формализация, фильтрация, сортировка, архивация, преобразование, защита, транспортировка*.

Полная совокупность всех этих действий над информацией в соответствии с принятой информационной технологией называется *информационным процессом*. Его составные части назовем *информационными процедурами*.

Информационные процедуры

- ◆ *Сбор данных* — накопление информации в целях обеспечения достаточной полноты для принятия решения.
- ◆ *Формализация* — приведение данных, поступающих от различных источников, к одной форме, чтобы сделать их сопоставимыми между собой и тем самым повысить уровень их доступности.
- ◆ *Сортировка* — упорядочение данных по заданному признаку для удобства использования, что также повышает уровень доступности данных.
- ◆ *Архивация* — организация хранения данных в удобной и легкодоступной форме с целью снижения экономических затрат на хранение данных (позволяет повысить общую надежность информационного процесса в целом).
- ◆ *Преобразование данных* — перевод данных из одной формы в другую. Преобразование может быть связано с изменением типа носителя. Так, например, книга может храниться как

в обычной бумажной форме, так и в электронном виде в памяти ЭВМ, на дискете, на магнитной ленте и др. Необходимость в многократном преобразовании информации возникает также и при транспортировке данных. Например, при передаче цифровых потоков данных по телефонным каналам связи требуется цифровые потоки преобразовать в аналоговые, поскольку телефонная сеть изначально ориентирована на передачу аналоговых сигналов в узком диапазоне частот. Для цифроаналоговых преобразований (ЦАП) используются специальные устройства — *модемы*.

Отметим также, что обработкой и преобразованием информации занимаются миллионы людей и на каждом рабочем месте выполняются еще свои специфические операции, необходимые для управления социальными, экономическими, научными и другими процессами.

- ♦ *Защита данных* — комплекс мер, направленных на предотвращение утраты, несанкционированного воспроизведения и модификации данных.
- ♦ *Транспортировка* — процесс передачи информации от мест ее генерации к местам хранения, обработки и использования. Варианты передачи могут быть разными: устный (по телефону), почтовый (с использованием бумаги), электронная почта, телекоммуникационные сети (Интернет), автоматизированные каналы связи.

Общая схема передачи информации по автоматизированным каналам связи приведена на рис. 2.3.

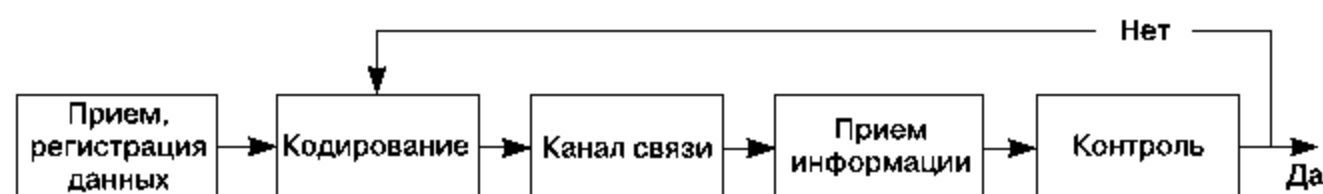


Рис. 2.3. Общая схема передачи информации по автоматизированным каналам связи

2.4.2. Задачи и методы обработки информации

Все разноплановые действия над информацией определим понятием *задачи обработки информации*.

Классификация задач обработки информации может быть выполнена по двум критериям:

- ♦ относительно самого содержания обработки, например поиск данных в массивах и др.;
- ♦ относительно глубины и сложности процедур.

Классификацию по первому критерию можно выполнить в соответствии с уровнем воздействия на информацию в процессе решения задачи:

- ♦ *синтаксический уровень* — обрабатываемая информация претерпевает лишь синтаксические изменения (прием, регистрация, пересылка, сортировка);
- ♦ *семантический уровень* — в результате преобразования информации возникают новые ее семантические элементы, которые носят, однако, безотносительный характер обеспечиваемой деятельности (всевозможные преобразования по аналитическим и логическим зависимостям);
- ♦ *прагматический уровень* — результаты преобразования содержат новые семантические элементы, но уже ориентированные на определенную область деятельности. Преобразование на этом уровне сводится к выработке информации, обеспечивающей принятие оптимального решения.

Оптимальным называют решение многовариантной задачи, которое из множества допустимых решений наилучшим образом удовлетворяет условиям данной задачи.

В теории управления доказано, что процесс принятия любого решения делится на две фазы:

- ♦ анализ ситуации с целью поиска оптимальных вариантов;
- ♦ непосредственное принятие решения.

Выделенные три уровня преобразования информации (синтаксический, семантический, прагматический) могут быть квалифицированы как три класса задач обработки информации:

- ♦ *информационно-поисковые* (преобразование на синтаксическом уровне);
- ♦ *логико-аналитические* (преобразование на семантическом уровне). Это сугубо вычислительные процессы;
- ♦ *поисково-оптимизационные* (преобразование на прагматическом уровне). Позволяют находить оптимальные решения в различных ситуациях.

2.5. ИНФОРМАЦИОННЫЕ СИСТЕМЫ

Важным является то обстоятельство, что независимо от природы и способа получения информация обладает *четырьмя общими свойствами*: ее можно создавать, хранить, обрабатывать и передавать.

Процессы, связанные с этими фазами (этапами) обращения информации, называют *информационными процессами*, а системы, реализующие их, — *информационными системами*.

Информационная система — это организационно упорядоченная совокупность документов, информационных технологий, в том числе с использованием средств вычислительной техники и связи, реализующих информационные процессы.

Информация, будучи аккумулированной и обработанной, дает новые сведения, приводит к новому знанию. Однако наиболее часто она используется для выработки управляющих воздействий в автоматических (чисто технических) и автоматизированных (человекомашинных) системах.

Информационные системы могут иметь различную природу — *техническую, биологическую, социальную* и т.д. В любой из них в той или иной мере происходят сбор и обработка информации. Примером простой *технической системы* является автомат для продажи газированной воды, который, получив монету, обрабатывает информацию о ней, анализирует ее и принимает решение:

- ◆ налить стакан воды без сиропа;
- ◆ налить стакан воды с одной порцией сиропа;
- ◆ налить стакан воды с двумя порциями сиропа;
- ◆ вернуть монету.

Более сложным является автомат по продаже билетов на пригородные электропоезда в железнодорожных кассах, анализирующий *совокупность* опущенных монет и принимающий решение о выдаче билета до той или иной железнодорожной зоны в зависимости от набора монет.

Примером информационной системы *биологической природы* являются система терморегуляции нашего организма, зрительная система и др. Кожа человека содержит около 300 000 клеток-датчиков, собирающих информацию о температуре тела. Собранная информация попадает в определенные участки головного мозга, центры теплоотдачи которого управляют охлаждением тела, а теплопродукции — нагреванием. В них информация обрабатывается, и при необходимости нагрева организма теплая кровь начинает интенсивнее циркулировать по сосудам, а для охлаждения усиливается процесс потоотделения. Если центры теплопродукции дезинформировать, например, раздражая их электрическим током, то они могут разогреть тело до смертельной температуры.

В автоматических системах задачи сбора, хранения, обработки и преобразования информации решает ЭВМ, и на основе анализа резуль-

татов сама вырабатывает соответствующие решения. Примерами таких систем являются станки с числовым программным управлением, гибкие автоматизированные производства, станки-роботы и др.

Если первичная информация извлекается из какого-либо объекта (процесса), а выходная применяется для целенаправленного изменения состояния того же объекта (процесса), причем информацию для выбора основных управляющих воздействий (принятия решения) использует человек, то такую автоматизированную информационную систему называют *автоматизированной системой управления* (АСУ). В этом случае ЭВМ анализирует огромные объемы информации и предлагает возможные варианты, ознакомившись с которыми, человек либо выбирает лучший с его точки зрения, либо ставит перед машиной новое условие и ждет следующего совета.

АСУ нашли широкое применение во всех сферах современного общества, в первую очередь как системы управления технологическими процессами (АСУТП) и коллективами людей. АСУТП служат для автоматизации различных функций на производстве: широко используются при организации поточных линий, изготовлении микросхем, для поддержания технологического цикла в машиностроении и т.п.

Большинство автоматизированных информационных систем являются локальными системами и функционируют на уровне предприятий и учреждений. В настоящее время происходит интенсивный процесс интеграции таких систем в корпоративные и далее — в региональные и глобальные системы.

Системы более высокого уровня становятся территориально рассредоточенными, иерархическими как по функциональному принципу, так и по технической реализации. Для обеспечения взаимодействия территориально рассредоточенных систем требуются протяженные высокоскоростные и надежные каналы связи, а для увеличения объема обрабатываемой информации — ЭВМ высокой производительности. Это приводит к необходимости коллективного использования дорогостоящих средств автоматизации (ЭВМ и линии связи) и обрабатываемой информации (баз данных).

С точки зрения возможности организации конкретных информационных процессов различают:

- ◆ информационно-справочные системы;
- ◆ информационно-поисковые системы;
- ◆ системы обработки и передачи данных;
- ◆ системы связи;
- ◆ системы организационного управления.

Последние предназначены для автоматизации функций управленческого персонала (информационные системы управления банками, гостиницами, торговыми фирмами и др.).

2.6. НОВЫЕ ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ

Информатика призвана создавать новые информационные технологии и системы для решения задач информатизации.

В информационной технологии в качестве исходного материала выступает *информация*, но уже качественно новая — о состоянии объекта, процесса или явления. Основными компонентами информационных технологий служат: *сбор данных* (первичной информации), *обработка данных*, получение *результата* — новой информации и передача ее потребителю.

Выделяют несколько поколений информационных технологий:

- ♦ бумажную, связанную с бумажным документооборотом;
- ♦ безбумажную, или электронную, относящуюся к появлению ЭВМ (середина XX в.);
- ♦ новую информационную технологию, связанную с внедрением персональных компьютеров и телекоммуникационных средств (с середины 1980-х гг.).

Новая информационная технология (компьютерная технология) — технология, основанная на использовании персональных компьютеров и телекоммуникационных средств.

2.7. ОЦЕНКА ИНФОРМАЦИИ

2.7.1. Общая постановка задачи

Объективная оценка того, сколько и какой информации необходимо для оптимального информационного обеспечения конкретного объекта (предприятия, учреждения, заведения, банка и т.д.), чтобы последний мог эффективно функционировать, является одной из важнейших задач. Задача измерения и оценки информации относится к числу неординарных задач, поскольку понятие информации является сложным.

Все представляющие интерес свойства информации невозможно оценивать каким-либо одним показателем. Для сколько-нибудь полной оценки информации необходима некоторая *совокупность* взаимосвязанных *показателей*, т.е. *система*. Имеющиеся на сегодня методы позволяют определить количество информации лишь на синтаксическом

уровне, в то время как при решении вопросов информационного обеспечения деятельности объекта прежде всего необходимы оценки на семантическом и прагматическом уровнях.

2.7.2. Количественная оценка синтаксической информации

Такая оценка информации необходима, чтобы сравнить друг с другом массивы хранящейся или передаваемой информации, а также оценить размеры носителей, хранящих информацию (МД, МЛ, CD и др.).

Выше говорилось, что информация передается в виде *сообщений* (это информация, представленная в определенной форме: текст, графика и др.), которые могут быть *непрерывными*, например передача речи по телефону, или *дискретными* (последовательность отдельных элементов). Так, при передаче телеграммы сообщением является текст, а элементами сообщения — буквы. Принимая сообщение о каком-либо событии, мы меняем свои знания о нем, однако определить, сколько информации содержится в полученном сообщении, в общем случае довольно трудно, т.е. одно и то же сообщение может давать одному потребителю много информации, другому — мало.

Для количественной оценки информации, выраженной в любой из форм (текст, видео, графика и др.), ее принято представлять в *стандартном виде* — в виде *двоичной последовательности*.

На самом деле простейшими характеристиками информации являются ее значения (истина, ложь), которые могут быть описаны элементами множеств:

{истина, ложь}
{true, false}
{Да, Нет}

Если истинный ответ «Да» представлять цифрой 1, а ложный «Нет» — цифрой 0, то запись всех ответов, образующих многозначную последовательность 0 и 1, называют *двоичной последовательностью*.

Такой процесс получения информации об объектах исследования называют *кодированием*.

2.7.3. Особенности представления различных видов информации

Прежде чем кодировать информацию, необходимо договориться о следующем:

- ◆ какие коды использовать;
- ◆ в каком порядке их записывать, хранить и передавать.

То есть должен быть определен язык *представления информации*. Основными характеристиками синтаксической информации являются *способ представления и объем*.

За единицу информации принят *бит* — количество информации в сообщении, в котором случайная величина с равной вероятностью принимает лишь два разных значения — «0» и «1» (такая единица информации предложена Р. Хартли).

В информационных документах широко используются русские и латинские буквы, цифры, математические и другие знаки — всего $\approx 200 \dots 250$. Для кодирования каждого такого знака используется 8-разрядная двоичная последовательность — байт. Например:

А — 11000001_2 ;

Л — 11001110_2 .

Объем информации, хранящейся на машинных носителях (*емкость памяти*), измеряется в байтах и его производных (КБ, МБ, ГБ, ТБ).

1 КБ = 1024 байт = 2^{10} байт = 1024·8 бит;

1 МБ = 1024 КБ = 2^{20} байт;

1 ГБ = 1024 МБ = 2^{30} байт;

1 ТБ = 1024 ГБ = 2^{40} байт.

Например, для хранения одной машинописной страницы текста требуется ≈ 2 КБ, а книги в 500 страниц — ≈ 1 МБ.

Для представления *графической информации* в двоичной форме используется так называемый *поточечный способ*. На первом этапе рассматриваемое изображение делят вертикальными и горизонтальными линиями на мелкие квадраты («точки») — *пиксели*. Чем больше точек, тем точнее информация об изображении. Далее записывается информация о каждом пикселе, которая содержит по крайней мере его номер и цвет. Совокупность данных о каждом квадрате представляет собой массив данных, описывающий с заданной точностью то или иное изображение.

Для примера рассмотрим виртуальную картину из 50 000 «точек». Тогда для представления номера квадрата потребуется 16 двоичных разрядов, так как 15 разрядами можно представить лишь 32 000 «точек» ($2^{15} \approx 32\,000$). Любой *цвет* может быть представлен в виде суммы *трех основных цветов* различной яркости: *зеленого, синего и красного*. Если яркость каждого цвета разбить на 16 градаций, то для их отображения потребуется 4 двоичных разряда ($2^4 = 16$). Таким образом, информация об одном квадрате изображения может представлять последовательность из 28 бит (рис. 2.4).

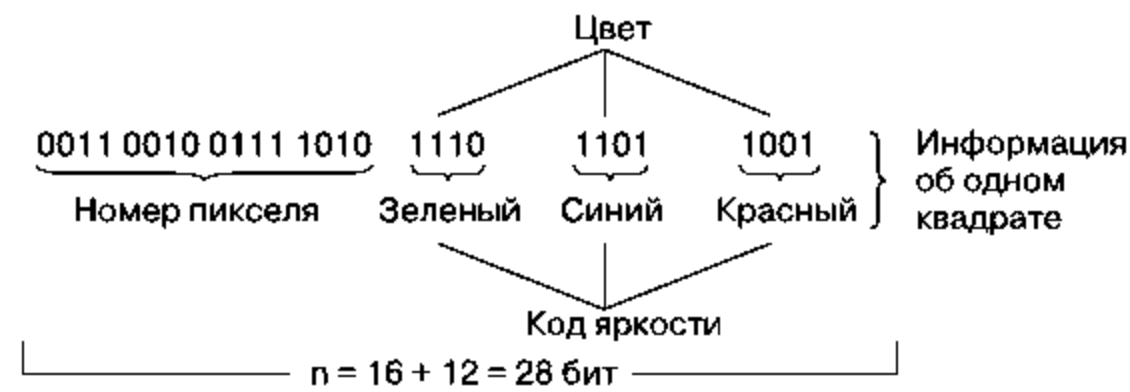


Рис. 2.4. Двоичное описание одного пикселя

Тогда объем памяти, необходимый для хранения всего массива данных (50 000 «точек»), описывающих картину:

$$V_D = 28 \text{ бит} \cdot 50\,000 = 1\,400\,000 \text{ бит} = 175\,000 \text{ байт} \approx 171 \text{ КБ.}$$

Естественно, что такой массив ничего не говорит о значимости картины.

2.7.4. Вероятностная оценка количества информации

Практически сообщение передается от источника к приемнику в материально-энергетической форме, например в форме электрического, звукового или иного сигнала.

Количеством информации называют числовую характеристику сигнала, которая не зависит от его формы и содержания и характеризует степень неопределенности, которая исчезает после выбора (получения) сообщения в виде данного сигнала.

В реальной жизни часто приходится иметь дело с явлениями или событиями, точный результат осуществления которых непредсказуем — погода на завтра, время ожидания автобуса на остановке, цвет шара, вынимаемого из урны, сторона монеты, упавшей после подбрасывания, и т.д. Число возможных вариантов результата для каждой из ситуаций может быть велико и точно не определено. Идеализированную модель такого рода ситуации называют *опытом*, а возможный вариант результата опыта — *исходом*.

Поскольку число возможных исходов опыта в общем случае велико, то определить точно результат опыта заранее невозможно, т.е. с проведением опыта связана некоторая *неопределенность*. Действительно, в опыте с двумя шарами разного цвета (например, черного и белого), находящимися в урне, цвет вынутого шара заранее предсказать нельзя. Ни один из вероятных исходов не имеет преимуществ перед другим. Тогда говорят, что исходы равновозможны или *равновероятны*.

Определим *вероятность* как число, характеризующее меру потенциальной возможности осуществления.

В опыте с двумя шарами вероятность каждого исхода $p_4 = p_5 = 1/2$. Если число шаров разного цвета в урне сделать равным n , то вероятность выбора шара конкретного цвета $p_i = 1/n$.

Впервые количественную оценку неопределенности ввел в 1928 г. американский ученый Р. Хартли для опыта X с n различными исходами:

$$H(X) = k \log_a n, \quad (2.1)$$

где X — информативный параметр сигнала;
 k — коэффициент пропорциональности;
 n — число возможных выборов (исходов);
 a — основание логарифма.

Для получения количественной оценки энтропии обычно используют основание логарифма a , равное двум. Полученная при этом единица измерения количества информации называется *битом* (bit — binary digit).

Если принять $k = 1$ и опустить a , то формула (2.1) примет вид:

$$H(X) = \log n. \quad (2.2)$$

Однако в оценке Р. Хартли не учтены вероятности различных исходов. К. Шеннон ограничил рамки применимости оценки Р. Хартли случаем, когда все n исходов в опыте X равновероятны ($p = 1/n$), а затем применил формулу к равновероятным исходам, усреднив полученные неопределенности по всем исходам.

Для опыта $X = \{x_1, x_2, \dots, x_n\}$, где $x_1, x_2, \dots, x_n$ — возможные исходы с вероятностями $p_1, p_2, \dots, p_n$, неопределенность каждого исхода равна $-\log p_1, -\log p_2, \dots, -\log p_n$, а математическое ожидание дает количественную оценку неопределенности — *энтропию*:

$$H = - \sum_{i=1}^n P_i \log P_i. \quad (2.3)$$

Интуитивно ощущается связь между заключенной в опыте *неопределенностью* и *информацией* об исходе — чем больше изначальная неопределенность исхода, тем больше информации о его фактическом наступлении, т.е. понятие энтропии тесно связано с понятием количества информации, под которым понимается мера устранения неопределенности в процессе получения сигнала адресатом. Например, если априорно ситуацию описать энтропией H_1 , то после получения сигнала энтропия должна уменьшиться до H_2 . Количество информации, полу-

ченное адресатом, равно $I = H_1 - H_2$. Если неопределенность снята полностью ($H_2 = 0$), то $I = H_1$.

Рассмотрим *пример* расчета количества информации в опыте, вероятности исходов которого различны. Так, если в урне из десяти шаров один черный, а остальные — белые, то вероятность выбора черного шара $p_ч = 1/10$ и количество информации о том, что выбран именно он, равно

$$H_ч = \log_2 \frac{1}{1/10} = \log_2 10 \approx 3,219 \text{ бит},$$

в то время как для белого шара

$$H_б = \log_2 \frac{1}{9/10} = \log_2 \frac{10}{9} \approx 1,048 \text{ бит}.$$

Количество информации тем больше, чем больше свободы в выборе сообщения, т.е. чем более неопределенно или менее вероятно передаваемое сообщение.

Энтропия заранее известного сигнала (значение его информативного параметра априорно известно) равна нулю. Формула для энтропии в этом случае будет состоять из слагаемых только двух видов: либо $1 \cdot \log 1$ для заранее известного сигнала, либо $0 \cdot \log 0$, так как вероятность появления всех других равна нулю.

Так как $1 \cdot \log 1 = 0$ и $\lim_{x \rightarrow 0} (x \log x) = 0$, то энтропия заранее известного сигнала равна нулю.

2.7.5. Связь «энтропия — информация»

Энтропия есть мера недостатка информации. Ее величина тем больше, чем более вероятно рассматриваемое состояние. В то же время информации тем больше, чем менее вероятно передаваемое сообщение.

Энтропия H достигает максимального значения, когда вероятности появления возможных значений информативного параметра сигнала одинаковы, т.е. при равенстве всех p_i ($P_i = 1/n$) соотношение (2.3) обращается в формулу Хартли (2.2):

$$H_{\max} = -\sum_{i=1}^n \frac{1}{n} \log \frac{1}{n} = \log n. \quad (2.4)$$

Пример 1. Определить количество информации в опыте с 10 шарами (1 черный и 9 белых).

Вероятность исхода с черным шаром $p_ч = \frac{1}{10}$, а с белым — $p_б = \frac{9}{10}$.

$$\text{Тогда } H = \frac{1}{10} \log_2 \frac{1}{1/10} + \frac{9}{10} \log_2 \frac{1}{9/10} = 0,469 \text{ бит.}$$

Опыт показал, что количество информации оказалось меньше, чем в опыте с двумя равновероятными исходами.

Пример 2. Если в тексте на русском языке, алфавит которого содержит 33 буквы и пробел, появление всех символов считать равновероятным, то вероятность появления каждого из них равна $1/34$ и количество информации, связанное с появлением каждого символа, составит

$$H = \log_2 34 \approx 5 \text{ бит.}$$

При разных вероятностях количество информации, приходящейся на одну букву, равно

$$H = \sum_{i=1}^{34} P_i \log \frac{1}{P_i}.$$

При двоичном кодировании количество информации на один символ $H = \log_2 2 = 1$ бит. Таким образом, количество информации в битах, заключенной в двоичном слове, равно длине самого слова.

Количественная оценка информации используется:

- ♦ при передаче информации на расстояние для определения *пропускной способности канала связи*;
- ♦ для определения количества информации, которую человек способен воспринимать, удерживать в памяти и перерабатывать в течение определенного времени, откуда вытекает важность дозировки учебной информации;
- ♦ для расчета объема оперативной памяти ЭВМ, чтобы оценить возможности машины при решении сложных задач.

2.8. ПОДХОДЫ К ОПРЕДЕЛЕНИЮ ПРАГМАТИЧЕСКИХ ПОКАЗАТЕЛЕЙ ИНФОРМАЦИИ

2.8.1. Оценка качества информации

Результат решения той или иной информационной задачи определяется не только количеством информации, т.е. объемом тех сведений, которые используются в процессе ее решения, но и качеством содержания, в частности, такими показателями, как:

- ♦ *полнота (достаточность)* сведений для информационного обеспечения решаемой задачи;
- ♦ *адекватность информации*, т.е. соответствие текущему состоянию объектов или процессов;

- ♦ *релевантность*, или чистота, информации, т.е. доля полезной информации в общем ее объеме (отсутствие среди необходимых данных ненужных, или, как принято говорить, шумов);
- ♦ *важность*, оцениваемая с точки зрения как важности самих задач, для которых она предназначена, так и ценности информации, используемой при их решении;
- ♦ *толерантность*, т.е. форма представления информации с точки зрения удобства восприятия и использования в процессе решаемой задачи.

2.8.2. Оценка важности информации

Для оценки *важности* информации используется показатель $K_{в.и}$, характеризующий значимость информации как с точки зрения задач, для решения которых она используется, так и с точки зрения организации ее обработки. Этим двум аспектам соответствуют две группы критериев. В *первой* группе критериев, оценивающей информацию по назначению, можно выделить два критерия:

- 1) $K_{в.з}$ — важность *самых задач*, для решения которых используется информация;
- 2) $K_{и.з}$ — важность *информации* для эффективного решения соответствующей задачи.

Во *второй* группе критериев, оценивающих информацию по условиям ее обработки, также выделим два критерия:

- 1) $K_{п.и}$, характеризующий *потери информации* в случае ее изменения в процессе обработки под воздействием помех;
- 2) $K_{с.в}$, характеризующий *уровень затрат* на восстановление нарушенной информации (важность информации с точки зрения стоимости ее восстановления).

Тогда очевидно, что

$$K_{в.и} = f(K_{в.з}, K_{и.з}, K_{п.и}, K_{с.в}), \quad (2.5)$$

т.е. для оценки важности информации необходимо уметь определять значения перечисленных коэффициентов и знать вид зависимости (2.5). К сожалению, на сегодняшний день не известно ни то, ни другое.

В качестве выхода из создавшегося положения используем подход, основанный на неформально-эвристических методах. Значения критериев, входящих в зависимость (2.5), будем выражать лингвистическими переменными (рис. 2.5).

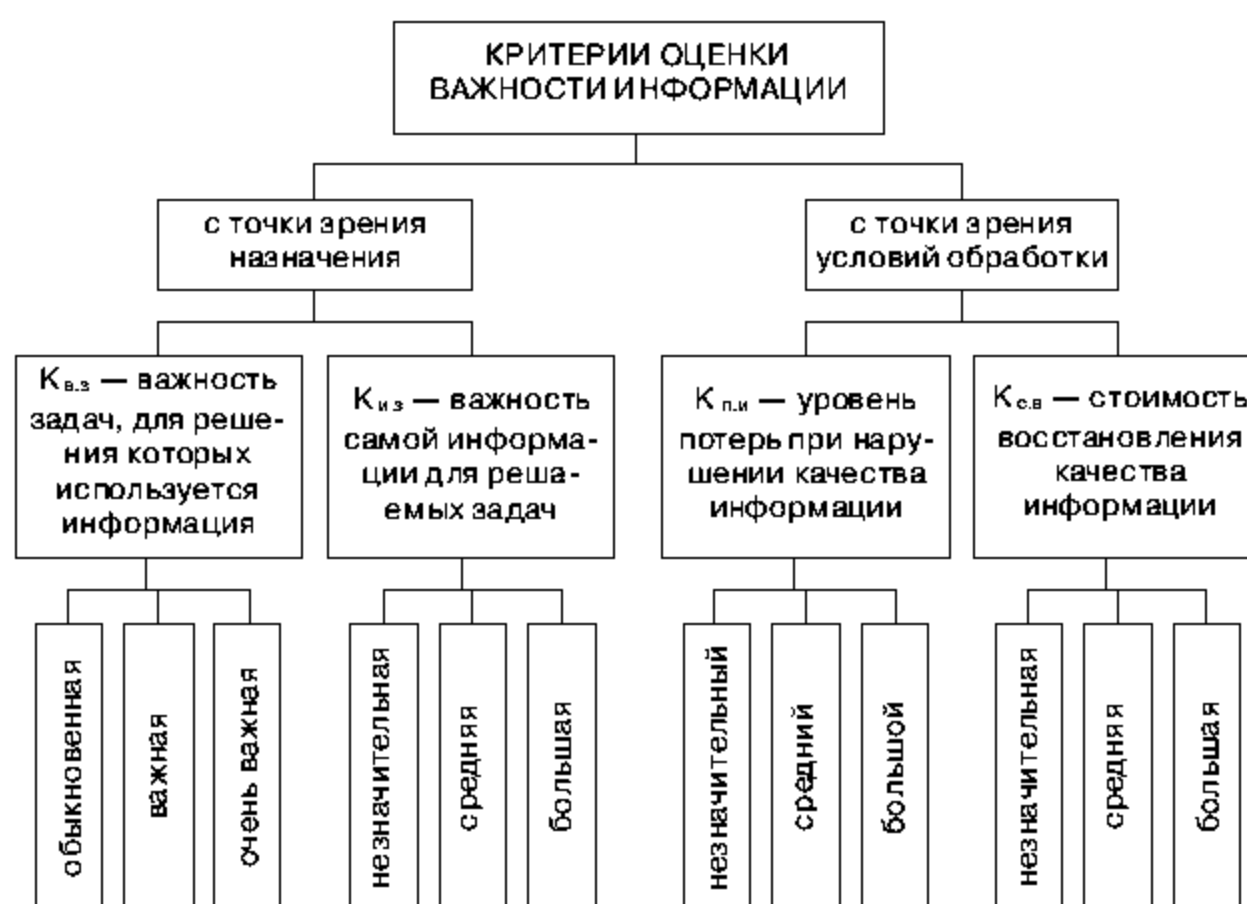


Рис. 2.5. Структура и значения критериев оценки важности информации

Затем сформируем возможные комбинации критериев в пределах каждой группы (табл. 2.1, 2.2).

Таблица 2.1

Первичная классификация важности информации по назначению (н)

Важность задач	Важность информации для задачи		
	незначительная	средняя	большая
Обыкновенная	1н	2н	3н
Важная	4н	5н	6н
Очень важная	7н	8н	9н

Таблица 2.2

Первичная классификация важности информации по параметрам обработки (о)

Уровень потерь	Стоимость восстановления		
	незначительная	средняя	большая
Незначительный	1о	2о	3о
Средний	4о	5о	6о
Большой	7о	8о	9о

Итоговая классификационная таблица (табл. 2.3) содержит 17 элементов, эквивалентных 17 классам важности. Косые линии определяют границы принадлежности класса соответствующим категориям важности — А, Б и т.д.

Таблица 2.3

Итоговая классификация важности информации

		А			Б			В		Г	
Важность информации		По параметрам обработки (о)									
		1о	2о	3о	4о	5о	6о	7о	8о	9о	
По назначению (н)	1н	1	2	3	4	5	6	7	8	9	Д
	2н	2	3	4	5	6	7	8	9	10	
	3н	3	4	5	6	7	8	9	10	11	
	4н	4	5	6	7	8	9	10	11	12	Е
	5н	5	6	7	8	9	10	11	12	13	
	6н	6	7	8	9	10	11	12	13	14	
	7н	7	8	9	10	11	12	13	14	15	Ж
	8н	8	9	10	11	12	13	14	15	16	
	9н	9	10	11	12	13	14	15	16	17	

Поскольку номер класса сам по себе не характеризует важность информации, да и оперировать с семнадцатью классами затруднительно, поделим классы на семь категорий важности, присвоив им содержательные наименования (табл. 2.4).

Таблица 2.4

Категории важности

Категория	Важность
А	МлВ (маловажная)
Б	ОбВ (обыкновенной важности)
В	ПсВ (полусредней важности)
Г	СрВ (средней важности)
Д	ПвВ (повышенной важности)
Е	БлВ (большой важности)
Ж	ЧвВ (чрезвычайной важности)

Оперировать с категориями проще, однако для аналитических расчетов необходимы количественные значения показателей важности информации, получить которые можно, построив *логистическую* кривую (рис. 2.6) зависимости значения важности информации от ее категории и класса.

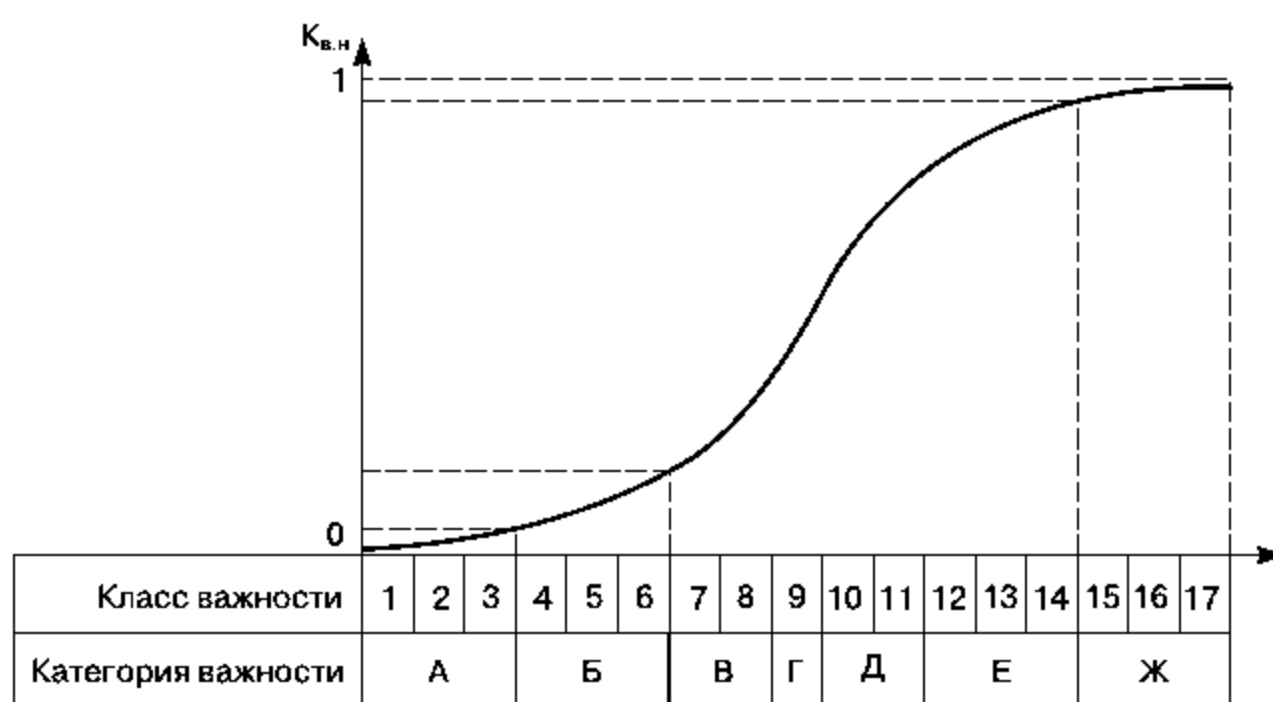


Рис. 2.6. График изменения важности информации

Естественно предположить, что важность информации категории А убывает от класса 3 к классу 1, приближаясь к нулю, а категории Ж — возрастает от класса 15 к классу 17, приближаясь к единице. Естественно также предположить, что возрастание важности информации от класса 1 к классу 17 происходит неравномерно. Наиболее адекватной, видимо, будет зависимость в виде кривой (см. рис. 2.6), позволяющей получить количественные показатели важности информации, значение которой меняется от 0 до 1.

Таким образом, алгоритм определения показателя важности информации следующий:

- ♦ экспертное определение значений критериев $K_{в.з}$, $K_{и.з}$, $K_{п.и}$, $K_{с.в}$ в лингвистических переменных;
- ♦ градация критериев по табл. 2.1 и 2.2;
- ♦ построение итоговой таблицы важности и логистической кривой на ее основе;
- ♦ вычисление значений $K_{в.н}$ по кривой.

Пример важной задачи, стоявшей в 2002 г. перед мировой общественностью, — уничтожение очага терроризма в Афганистане.

Составляющие:

- ♦ поиск центра очага (использование данных разведок различных стран);
- ♦ выбор средств доставки контингента от США до Афганистана;
- ♦ выбор оптимальных мест для строительства баз с учетом:
 - согласия выбранной страны;
 - стоимости строительства и затраченного времени;

времени полета и возврата самолетов и вертолетов на базу;
исследования возможностей использования готовых аэродромов для посадки тяжелых самолетов и вертолетов;
влияния на окружающую среду сливаемого горючего в случае необходимости (самолет с полной нагрузкой посадку совершить не может и керосин сливается).

Решение перечисленных вопросов говорит как о важности решаемой задачи, так и о важности информации, используемой для ее решения, и, естественно, результат оценки важности информации будет стремиться к единице.

2.8.3. Оценка адекватности информации

Адекватность информации — это определенный уровень соответствия созданного с помощью полученной информации образа реальному объекту, процессу или явлению.

Адекватность информации определяется двумя характеристиками:

- а) *объективностью съема информации* о предмете, процессе или явлении;
- б) *продолжительностью интервала времени t между началом съема информации и текущим моментом, т.е. до момента оценки ее адекватности.*

Объективность полученной информации, т.е. насколько точно отражает информация исследуемый процесс или предмет, в свою очередь, зависит от *способа* ее получения и *качества* его реализации.

Например, напряжение U можно измерить либо аналоговым способом (точность низкая), либо цифровым способом с высокой точностью.

Классификация характеристик по возможным способам получения (генерирования) их значений приведена на рис. 2.7.



Рис. 2.7. Классификация характеристик по способам получения их значений

Если качество определения значения характеристики оценивать тремя логистическими переменными — высокое, среднее, низкое, то все возможные значения адекватности информации по объективности ее определения можно структурировать в соответствии с табл. 2.5 и получить 10 классов.

Таблица 2.5

**Структуризация значений адекватности информации
по объективности ее определения**

Тип характеристики			Качество определения значения характеристики		
			высокое	среднее	низкое
Измеряемая	непосредственно	количественно	1	2	3
		качественно	2	3	4
	косвенно	аналитически	3	4	5
		логически	4	5	6
Неизмеряемая	имеющая аналоги	в данной среде	5	6	7
		в сходной среде	6	7	8
	не имеющая аналогов	конкретных	7	8	9
		даже отдаленных	8	9	10

Кривая, описывающая изменение адекватности от объективности съема (определения) информации о процессе или явлении, приведена на рис. 2.8. По аналогии с методикой оценки важности информации естественно предположить, что при *высоком* качестве определения значения *измеряемой* характеристики *непосредственно* и притом *количественно* адекватность соответствующей информации будет близка к единице, а при *низком* качестве определения значения *неизмеряемой* характеристики, не имеющей даже *отдаленного аналога*, адекватность информации близка к нулю. Естественно также предположить, что внутри данного интервала изменение адекватности происходит по логистической кривой.

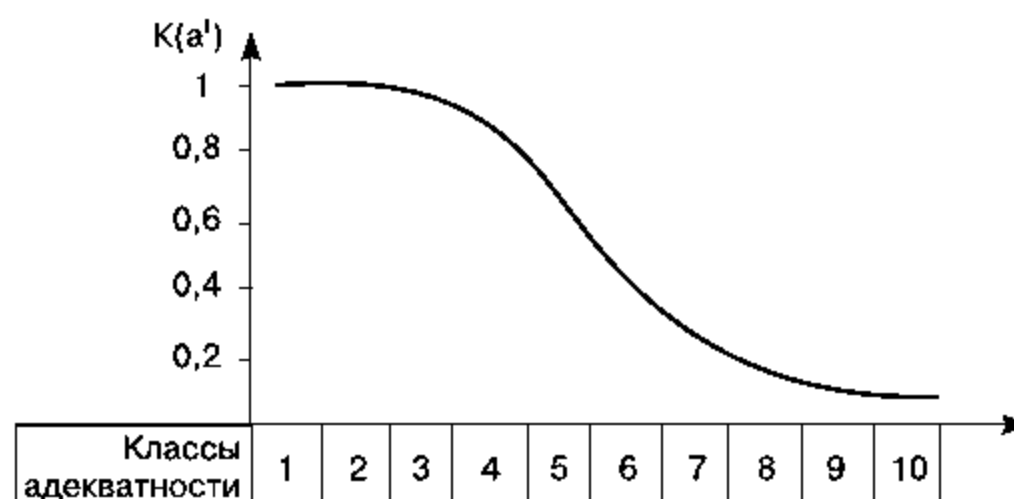


Рис. 2.8. Кривая изменения адекватности от объективности съема информации

Для оценки адекватности информации по *длительности* интервала времени Δt между моментом начала съема информации и текущим моментом пользуются известным в теории информации *законом старения информации* (рис. 2.9).

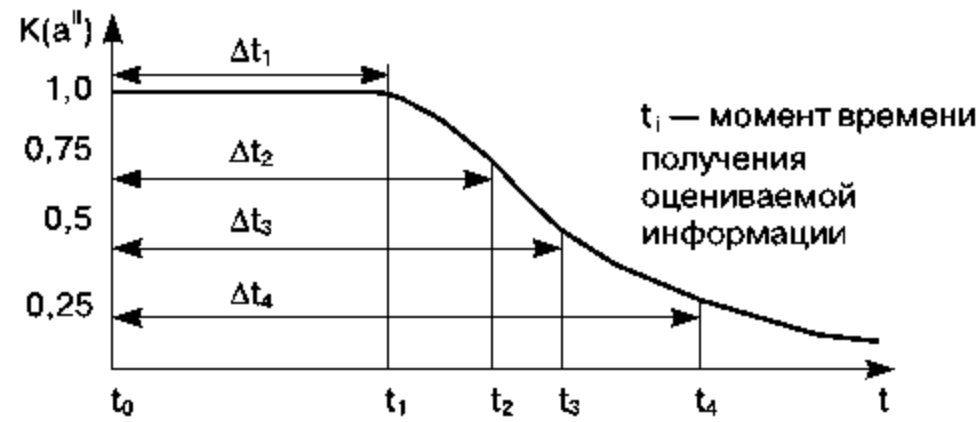


Рис. 2.9. Общий вид закона старения информации

Из рисунка 2.9 видно, что закон характеризуется четырьмя интервалами.

На интервале $\Delta t_1 = t_1 - t_0$ оцениваемая информация полностью сохраняет свою адекватность, на интервале $\Delta t_2 = t_2 - t_0$ адекватность уменьшается, но не более чем на 0,25, а на интервалах Δt_3 и Δt_4 — на половину и на 0,75 соответственно.

Обе составляющие адекватности $K(a')$ и $K(a'')$ зависят от большого числа факторов, многие из которых носят случайный характер. Поэтому будем считать, что в основном это случайные величины, и они могут интерпретироваться как вероятности того, что информация по соответствующему параметру является адекватной.

Поскольку в большинстве практических приложений важно, чтобы информация была адекватной одновременно по обоим параметрам, то в соответствии с теоремой умножения вероятностей общий показатель адекватности информации

$$K(a) = K(a') K(a''),$$

где a' и a'' — независимые величины.

2.8.4. Оценка полноты информации

Полнота (достаточность) информации — относительный показатель, поскольку информация оценивается применительно к определенной задаче или группе задач. Необходимо заранее составить перечень сведений, требуемых для решения конкретной задачи. Это должна быть полная и хорошо структурированная совокупность данных, не-

обходимых для информационного обеспечения объекта в соответствии с его целевым назначением.

Для представления таких сведений удобно воспользоваться так называемыми объектно-характеристическими таблицами (ОХТ), каждая из которых представляет собой двухмерную матрицу. В строках ее приводится перечень наименований объектов, процессов или явлений, предметов, входящих в круг интересов решаемой задачи, а в столбцах — наименование характеристик тех предметов, явлений или процессов, значения которых необходимы для решения задачи. Сами значения характеристик будут располагаться на пересечении соответствующих строк и столбцов. Совокупность всех ОХТ, необходимых для обеспечения решения всех задач объекта (предприятия, учреждения, заведения), может быть названа *информационным кадастром* объекта. Таким образом, неперенным условием оценки полноты информации является наличие информационного кадастра, под которым будем понимать полную и хорошо структурированную совокупность данных, необходимых и достаточных для эффективного функционирования объекта в соответствии с его целевым назначением. Представлять его удобно в виде упорядоченной совокупности так называемых ОХТ.

Для оценки *полноты информации* обозначим через $d_{\mu\nu}$ элемент, находящийся в μ -й строке и ν -м столбце рассматриваемого компонента соответствующей ОХТ, причем

$$d_{\mu\nu} = \begin{cases} 1, & \text{если по данному элементу информация имеется;} \\ 0, & \text{в противном случае.} \end{cases}$$

Тогда в качестве меры коэффициента полноты информации в данной ОХТ можно принять величину

$$K_n = \frac{\sum_{\nu} \sum_{\mu} d_{\mu\nu}}{mn},$$

где m — число строк, а n — число столбцов в рассматриваемой ОХТ.

Если учесть значимость (важность) каждого из элементов (пусть $K_{\mu\nu}^{(a)}$ — коэффициент важности элемента μ -й строки и ν -го столбца), то, очевидно, в качестве меры взвешенной полноты информации в рассматриваемой ОХТ можно принять величину

$$K_n^{(a)} = \frac{\sum_{\nu} \sum_{\mu} d_{\mu\nu} K_{\mu\nu}^{(a)}}{mn \sum_{\nu} \sum_{\mu} K_{\mu\nu}^{(a)}}.$$

2.8.5. Оценка релевантности информации

Релевантность есть показатель, характеризующий долю полезной информации в ее общем объеме V .

Коэффициент релевантности

$$K^{(p)} = \frac{N_p}{N_o},$$

где N_p — количество полезной информации;
 N_o — общий объем информации.

Количественная оценка такого показателя затруднительна. В сфере научно-технической информации под N_o понимается общее количество документов, выданных на запрос, а под N_p — количество документов, оказавшихся полезными. Однако такая оценка релевантности не применима к фактографической информации. Различают два вида информации — *документальную* и *фактографическую*. К первому виду относят источники, где могут находиться необходимые данные, документы о фактах, а ко второму — собственно факты, т.е. элементы информации, пригодные для непосредственного использования. В примере с ЭКГ (см. подраздел 2.3) выписка о болезни сердца, где описываются различные стороны заболевания, относится к первому виду, а сама кардиограмма — ко второму.

2.8.6. Оценка толерантности информации

Толерантность — способ представления информации с точки зрения ее использования потребителем, характеризуется временем и средствами, затраченными на извлечение информации, средствами отображения, возможностью восприятия информации и др. Подходы к оценке толерантности достаточно сложны [4] и в данной работе не рассматриваются.

3. СРЕДСТВА И СПОСОБЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

3.1. СООБЩЕНИЯ И ИХ ПРЕДСТАВЛЕНИЕ. ИСТОЧНИКИ СООБЩЕНИЙ

3.1.1. Общие сведения о сообщениях

Сообщение — это форма представления информации (текст, звук, видеоизображение, показания приборов и т.д.). С формальной точки зрения это совокупность знаков или первичных сигналов, содержащих информацию.

Конкретная форма сообщения (внешняя форма, изображение) называется *представлением*. Одно и то же сообщение может быть представлено различными способами. Например, сообщение о том, что в академии не курят, можно представить либо в текстовой форме: «Не курить!», «У нас не курят», «Курить запрещается», либо в графической форме (рис. 3.1).



Рис. 3.1. Знак, запрещающий курение

Формы представления допускают различные толкования:

- ♦ коса (девичья или инструмент для срезания травы);
- ♦ пол (мужской, женский или настил);
- ♦ ток (электрический или площадка для зерновых).

Важно установить *способ для выявления значения представления*. Переход от представления к абстрактной информации, т.е. к значению представления, называется *интерпретацией*.

Представление *интерпретируется* (истолковывается) для получения информации. Одно и то же представление (сообщение), по-разному интерпретируемое, может содержать различную информацию, например возможно разное толкование одних и тех же юридических

законов; единица информации *байт* может интерпретироваться устройством управления ЭВМ как:

- ♦ целое число без знака;
- ♦ целое число со знаком;
- ♦ команда и т.д.

Извлечение информации связано с изменением во времени значений какой-либо величины, характеризующей состояние x_i *источника информации* (ИИ). Так, измеряя периодически температуру тела больного (ИИ), врач получает информацию о его состоянии здоровья в процессе лечения. В этом смысле информационное сообщение можно представить функцией $X(t)$, которая может быть *аналоговой* или *цифровой* (дискретной). *Аналоговая* функция представляется в виде непрерывного ряда вещественных значений какой-либо физической величины в диапазоне изменения аргумента t (например, температура тела непрерывно изменяется в интервале времени измерения от значения $t_{\text{нач}}^0$ начального до $t_{\text{к}}^0$ конечного). При этом функция $X(t)$ передает характер изменения температуры во времени. *Дискретная* функция $X(t)$ на любом конечном отрезке (интервале) принимает конечное число различных значений. Соответственно и сообщения могут быть *непрерывными* (аналоговыми) или *дискретными*.

Непрерывные сообщения (см. подраздел 2.7.2) описываются непрерывной функцией времени. *Дискретное сообщение* представляет собой последовательность отдельных элементов, физическая природа которых может быть любой. Отдельные первичные сигналы с выхода источника дискретных сообщений называют *элементарными сообщениями*. Каждому элементарному сообщению соответствует определенное состояние x_i источника информации. Так, при передаче телеграммы сообщением является текст, а элементом сообщения — буква.

3.1.2. Источники сообщений

В зависимости от формы создаваемых сообщений различают источники дискретных и непрерывных (аналоговых) сообщений.

Дискретный источник формирует сообщение в виде случайных последовательных символов (статистически не связанных элементов сообщения) из фиксированного алфавита $x = \{x_1, x_2, \dots, x_n\}$ — *алфавита источника*. Сообщение приобретает *смысл* — несет в себе определенную информацию только тогда, когда состояние наблюдаемого объекта заранее не известно потребителю, а следовательно, получаемое им сообщение *случайно*. Совокупность всех возможных состояний и вероятностей их появлений образует *ансамбль сообщений*.

Таким образом, дискретный источник информации характеризуется определенным ансамблем состояний:

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ P(x_1) & P(x_2) & \dots & P(x_n) \end{pmatrix}$$

или

$$X = \begin{pmatrix} x_1 & x_2 & \dots & x_n \\ P_1 & P_2 & \dots & P_n \end{pmatrix},$$

причем $\sum_{i=1}^n P(x_i) = 1$ или $\sum_{i=1}^n P_i = 1$.

Принимая сообщение о каком-либо событии, мы получаем дополнительную информацию и меняем свои знания о нем. Определить, сколько информации содержится в сообщении, довольно сложно, поскольку одно и то же сообщение несет разную информацию потребителям — для одного ее будет много, для другого — мало.

Количество информации в сообщении определяется системой, не связанной с его конкретным содержанием, а отображающей лишь степень «неожиданности» (неопределенности), т.е. его вероятности. Маловероятное сообщение является неожиданным и поэтому содержит много информации, и наоборот.

Количество информации в отдельно взятом сообщении определяется величиной, обратной вероятности сообщения, вычисленной в логарифмических единицах:

$$H(x) = \log_a \frac{1}{P(x)} = -\log_a P(x),$$

где $P(x)$ — вероятность сообщения x ;
 a — основание логарифма.

При $a = 2$ количество информации, содержащейся в сообщении, выражается в двоичных единицах:

$$H(x) = -\log_2 P(x).$$

При $P(x) = 1$ количество информации $H(x)$ равно 0, т.е. сообщение об известном событии информацию не несет. Количество информации, содержащейся в небольших независимых сообщениях, равно сумме количества информации в каждом из них. Это соответствует представлениям об увеличении количества информации при получении дополнительных сообщений.

Если ансамбль состоит из двух сообщений x_1 и x_2 вида *да* или *нет* или 0 или 1, которые являются независимыми и равновероятными,

т.е. $P(x_1) = P(x_2) = 0,5$, то каждое сообщение несет двоичную единицу информации, которую принято называть *битом*:

$$H = -\log_2 P(x_1) = -\log_2 P(x_2) = -\log_2 0,5 = 1 \text{ бит.}$$

Для оценки информационных свойств сообщения в целом вводится среднее количество информации, приходящееся на одно сообщение, — *энтропия источника сообщений*. Энтропия источника определяется как математическое ожидание количества информации

$$H(x) = \sum_{i=1}^n P(x_i) \log_2 P(x_i). \quad (3.1)$$

Выражение (3.1) справедливо для источника с независимыми сообщениями.

Таким образом, энтропия источника определяется не только числом его возможных состояний, но и их вероятностями. Энтропия $H(x)$ должна быть непрерывной функцией вероятностей состояний $P(x_1)$, $P(x_2)$, ..., $P(x_n)$ (функционалом распределения вероятностей). При этом не важно, какие именно значения имеют x_1 , x_2 , ..., x_n , важны только количество этих значений и их вероятности, поскольку энтропия связывается только с фактом выбора, а не с множеством конкретных значений наблюдаемых явлений. Если же все P_i равны $P_i = \frac{1}{n}$, то $H(x)$ долж-

на быть монотонно возрастающей функцией от n .

Непрерывный источник характеризуется одномерной плотностью распределения случайной величины x — $P(x)$.

3.2. ВИДЫ СИГНАЛОВ И ИХ КЛАССИФИКАЦИЯ

3.2.1. Основные понятия временного представления сигнала

В самом общем смысле под *сигналом* понимается физическое явление или процесс, несущий информацию (сообщение) о каком-либо событии или состоянии объекта наблюдения, либо передача команды управления или оповещения.

В радиоэлектронике сигналы, как правило, представлены в виде электрического колебания, характеристики которого (ток или напряжение, длительность, частота, фаза и т.д.) однозначно связаны с каким-либо параметром конкретного физического процесса (сообщения). Поэтому наряду с термином «сигнал» используют термин «электрическое колебание».

Будем рассматривать электромагнитные сигналы как средство передачи данных. *Данные* — объекты, передающие смысл или информацию. *Сигналы* — это электрическое или электромагнитное представление данных. Сам сигнал является *функцией времени*, но для передачи данных его можно рассматривать и с точки зрения *частотного* представления. Ниже рассматриваются оба способа.

Если рассматривать сигнал как функцию времени, то он может быть непрерывным (аналоговым) или дискретным. *Непрерывным* называется сигнал, изменение интенсивности которого во времени описывается гладкой функцией, т.е. в сигнале нет пауз и разрывов (рис. 3.2). С математической точки зрения сигнал $U(t)$ является непрерывным, если для всех a $\lim_{t \rightarrow a} U(t) = U(a)$.

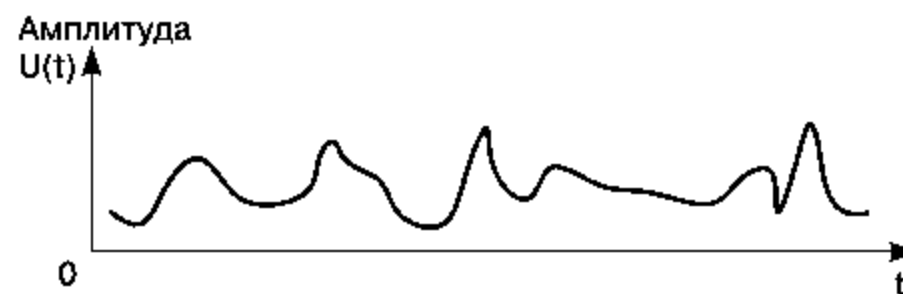


Рис. 3.2. Непрерывный сигнал

Примерами непрерывного сигнала являются речь, видеозображение. Многие данные, собираемые с помощью датчиков, например температура и давление, также принимают непрерывные значения.

Интенсивность *дискретного* сигнала в течение некоторого периода является постоянной, а затем также изменяется на постоянную величину (рис. 3.3). Примерами дискретного сигнала являются набор двоичных нулей и единиц, текст.

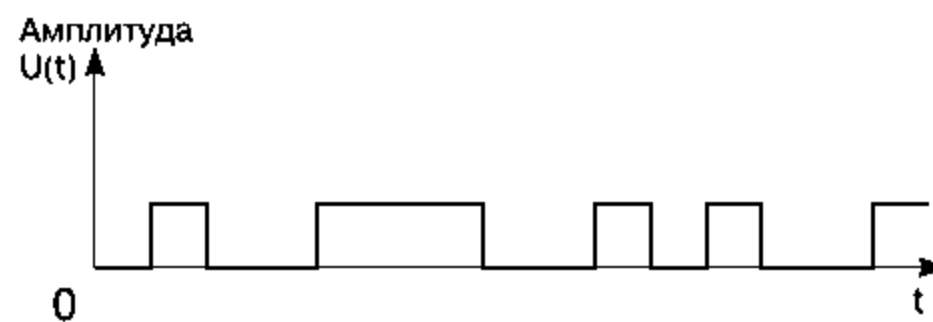


Рис. 3.3. Дискретный сигнал

Простейшим типом сигнала является периодический сигнал, в котором некоторая структура периодически повторяется во времени. На рисунке 3.4 приведен пример периодического непрерывного сигнала (синусоида) и периодического дискретного сигнала (прямоугольный импульс).

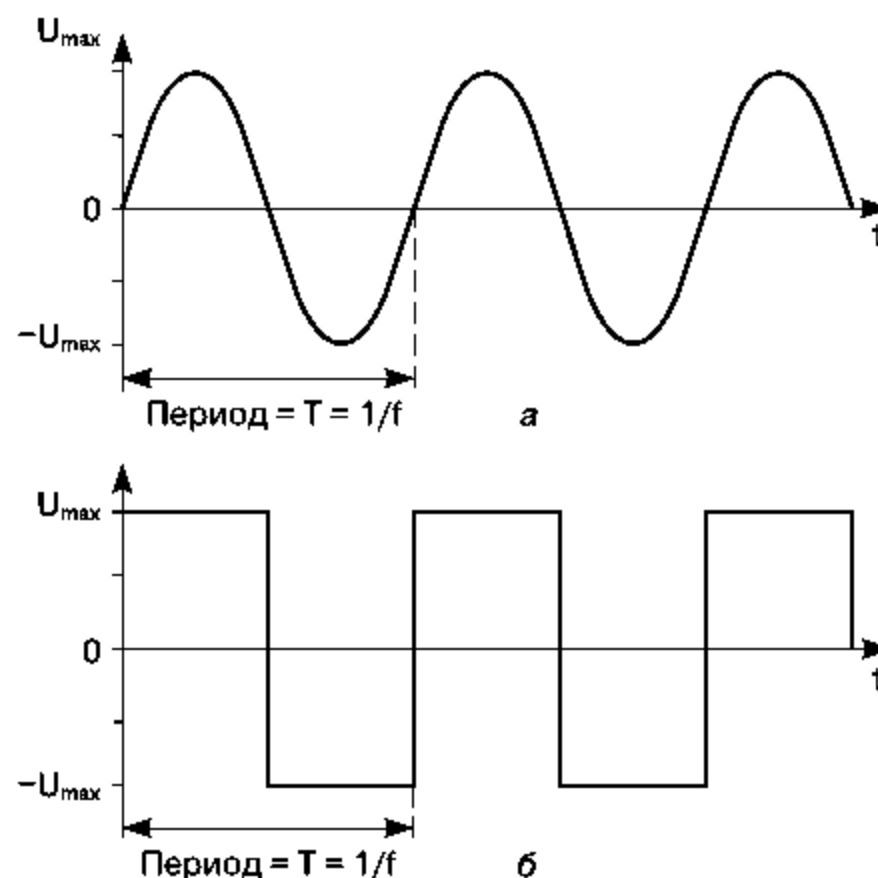


Рис. 3.4. Периодические сигналы:
а — синусоидальный; б — прямоугольный

В общем случае синусоидальный сигнал можно представить в виде

$$U(t) = U_{\max} \sin(\omega t + \varphi_0) = U_{\max} \sin(2\pi f t + \varphi_0) \quad (3.2)$$

и задать тремя параметрами: максимальной амплитудой $U_{\max}$, частотой f и фазой φ_0 . *Максимальной амплитудой* называется максимальное значение или интенсивность сигнала во времени. *Частотой* называется темп повторения сигналов (измеряется в периодах за секунду или герцах). Эквивалентным параметром частоты f является период сигнала T , представляющий собой время, за которое происходит повторение сигнала (см. рис. 3.4), следовательно, $T = 1/f$. *Фаза* является мерой относительного сдвига по времени в пределах отдельного периода сигнала.

Для синусоидальных сигналов существует два простых соотношения, одно из которых связано со временем, а другое — с пространством. Определим *длину волны* сигнала λ как расстояние, занимаемое одним периодом. Предположим, что сигнал распространяется со скоростью v . Тогда длина волны связана с периодом соотношением $\lambda = vT$ или, иначе, $\lambda f = v$.

По характеру все сигналы можно разделить на 4 группы.

1. *Непрерывная функция непрерывного аргумента* $x(t)$ (рис. 3.5). Такие сигналы называются просто непрерывными или аналоговыми. Они могут изменяться в произвольные моменты вре-

мени, принимая любые значения из непрерывного множества возможных значений $x(t) - \infty \leq t \leq \infty$.

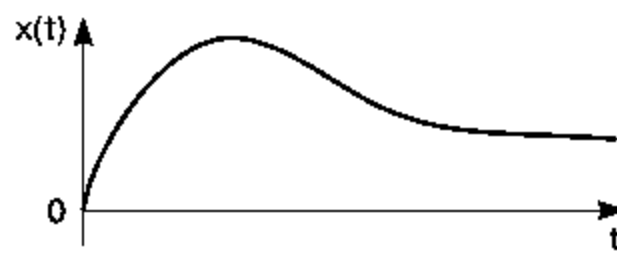


Рис. 3.5. Непрерывная функция непрерывного аргумента

2. *Непрерывная функция дискретного аргумента $x(t_j)$* (рис. 3.6). Функция может принимать произвольные значения, но изменяется только в определенные (дискретные) моменты времени $j = 0, 1, 2, \dots$ (квантование по времени). Квантование непрерывных сигналов по времени называют *дискретизацией*. Дискретизация позволяет любой непрерывный сигнал представить отдельными мгновенными значениями, соответствующими, например, амплитуде импульса в определенный момент времени t_j .

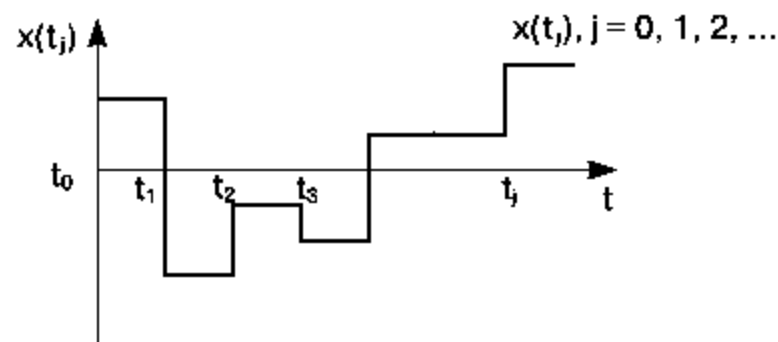


Рис. 3.6. Непрерывная функция дискретного аргумента

3. *Дискретная функция непрерывного аргумента $x_j(t)$* (рис. 3.7).

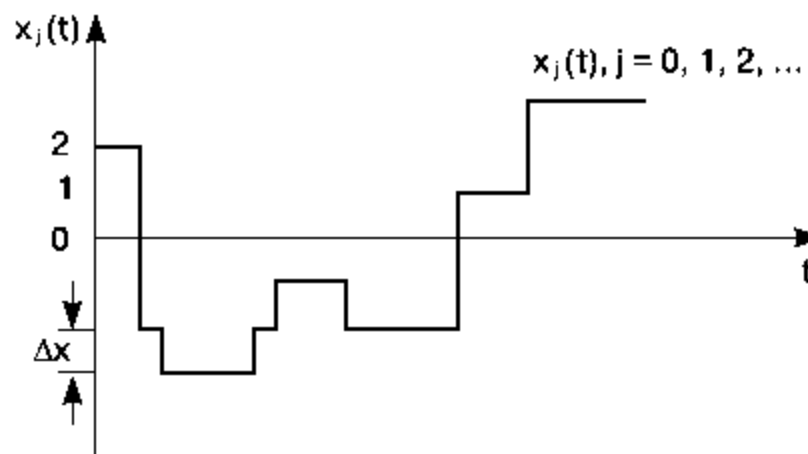


Рис. 3.7. Дискретная функция непрерывного аргумента

Дискретный сигнал непрерывного времени может изменяться в произвольные моменты времени, но величина сигнала принимает только конкретные разрешенные дискретные значения (уровни). Процесс замены непрерывной функции $x(t)$ функцией $i\Delta x(t)$, т.е. ограниченным числом дискретных значений по дискретной шкале уровней, называется *квантованием по уровню*. Непрерывное значение сигнала в момент времени t_k в этом случае можно представить ближайшим значением по дискретной шкале уровней (рис. 3.8), отнеся его либо к x_1 , либо к x_2 . Такое квантование приводит к погрешности, называемой *шумом квантования*

$$x_{\text{ш.кв}} \leq \pm \frac{\Delta x}{2}.$$

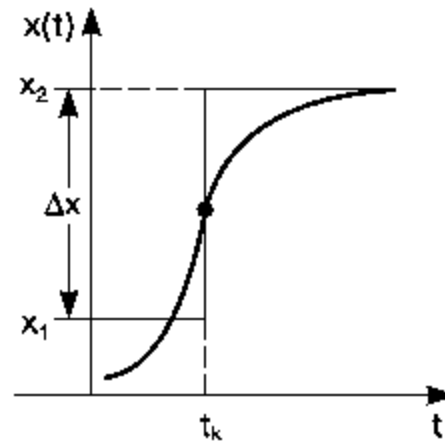


Рис. 3.8. Погрешность квантования

Обычно принимается, что шум квантования является случайным и подчиняется нормальному закону распределения (рис. 3.9).

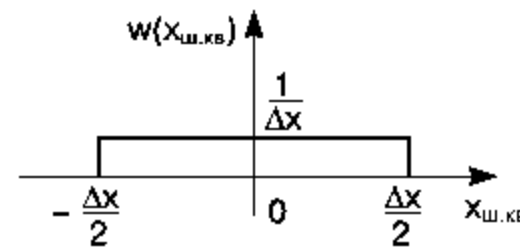


Рис. 3.9. Закон описания шума квантования

4. Дискретная функция дискретного аргумента (рис. 3.10).

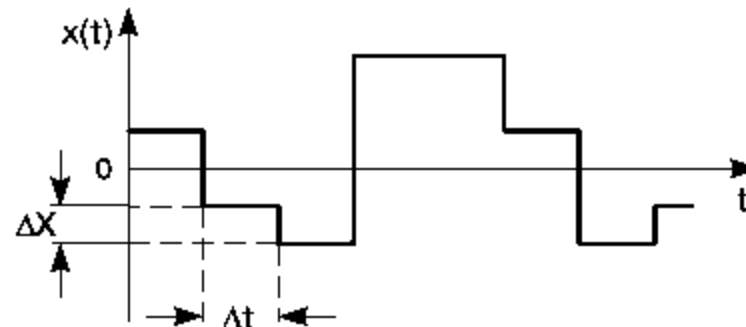


Рис. 3.10. Дискретная функция дискретного аргумента

Сигнал $x(t)$ в дискретные моменты времени принимает только конкретные дискретные значения, т.е. $x(t)$ заменяется сигналом $x_j(t_j)$, или, иначе, принимает ближайшее значение по шкале квантования по уровню $i\Delta x$ в моменты времени $j\Delta t$. Это квантование *по времени и по уровню*.

Иногда в отдельный класс выделяют *импульсные сигналы*, которые отличны от нуля лишь в течение конечного интервала времени. Импульсные сигналы на интервале своего существования могут быть непрерывными (например, импульсы колоколообразной формы) или дискретными (прямоугольные).

Все сигналы, как непрерывные, так и дискретные, могут быть *периодическими* (значение сигнала повторяется через определенные промежутки времени, называемые периодами) и *непериодическими*.

Система связи, осуществляющая перенос сигнала X_T из одной точки пространства в другую, — это сложный комплекс взаимодействующих технических средств (передатчик, канал связи (КС), приемник и др.) (см. рис. 3.52).

Преобразование сообщения в передаваемый сигнал X_T — некоторый материальный носитель сообщения — выполняет передатчик по строго определенному правилу (модуляция, кодирование, манипуляция), выбор которого зависит от типа сообщения и сигнала.

Доставленный в пункт приема сигнал X_T^1 должен быть снова преобразован в сообщение и затем передан получателю. Однако и сигнал X_T^1 , и принятое сообщение могут отличаться от сигнала X_T на входе системы связи и передаваемого сообщения a .

Основная причина искажения передаваемого сигнала X_T — воздействие помех в линии связи, непременной составляющей КС, и отчасти шумы в аппаратуре передачи данных (АПД), входящей в КС.

Восстановление переданного сообщения по принятому сигналу X_T^1 осуществляет *приемник*. Причем эта операция возможна, если известно правило преобразования сообщения в сигнал. На основе этого правила вырабатывается правило обратного преобразования сигнала в сообщение (демодуляция, декодирование). Оно и позволяет приемнику выбрать сообщение из известного множества возможных сообщений, в идеальном случае полностью совпадающее с переданным сообщением a_T . Однако восстановить переданное сообщение иногда и не удастся. Вследствие искажения принятого сигнала возможна ошибка при восстановлении сообщения.

Степень соответствия между переданным X_T и принятым сигналом X_T^1 определяет *достоверность передачи данных*, которая характеризуется вероятностью искажения бита и для КС без дополнительных средств защиты от помех составляет 10^{-5} — 10^{-6} .

3.2.2. Основные понятия частотного представления сигнала (спектральное представление сигналов)

Частотную природу сигнала определить несложно, если принять во внимание, что реальный электромагнитный сигнал составляется из многих частот. Так, любой периодический сигнал (рис. 3.11, *д*) может быть представлен в виде суммы синусоид, называемой *рядом Фурье*:

$$U(t) = \frac{U_0}{2} + \sum_{n=1}^{\infty} (a_n \cos n\omega_1 t + b_n \sin n\omega_1 t). \quad (3.3)$$

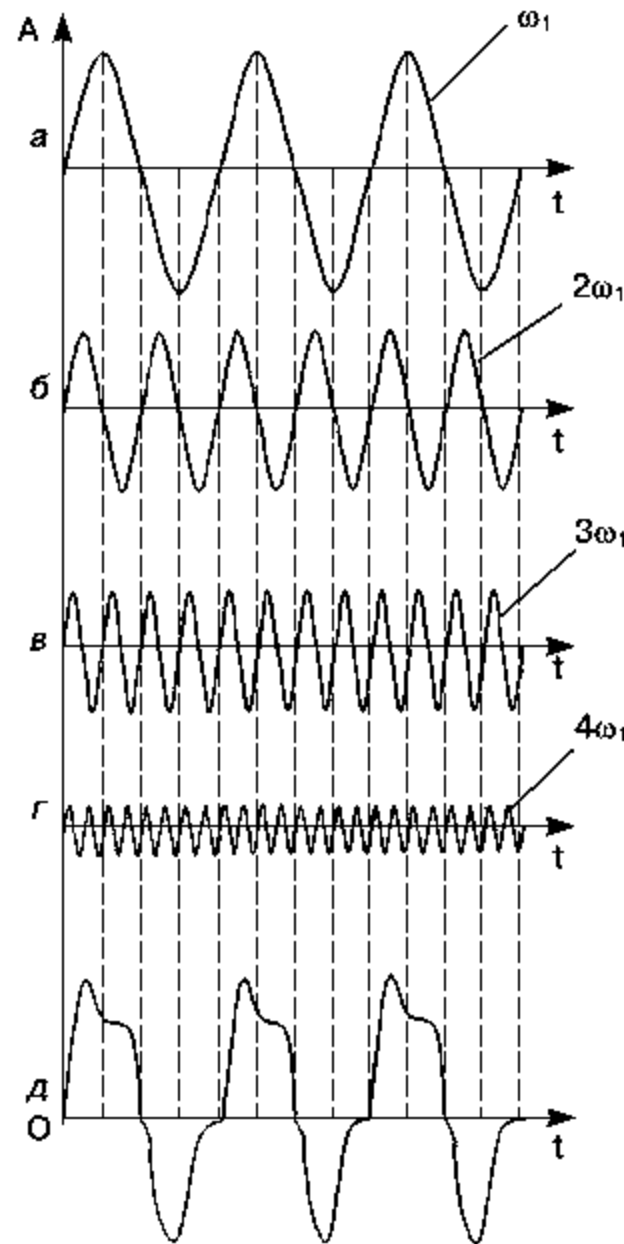


Рис. 3.11. Представление периодического сигнала суммой синусоид:

а—г — гармоники; *д* — сумма гармоник

Каждая составляющая синусоида (рис. 3.11, *а—г*) называется *гармоникой*. Частота $\omega_1 = 2\pi f_1$ называется *собственной частотой* или *собственной гармоникой*, если все остальные кратны ей. Периодический сиг-

нал с периодом T состоит из собственной частоты $f_1 = 1/T$ и частот, кратных ей. Период суммарного сигнала равен периоду собственной частоты. Если член $U_0 \neq 0$, то сигнал $U(t)$ содержит *постоянную составляющую* $\left(\frac{U_0}{2}\right)$. Значения коэффициентов, входящих в выражение

(3.3), вычисляются следующим образом:

$$\begin{aligned} U_0 &= \frac{2}{T} \int_0^T U(t) dt, \\ a_n &= \frac{2}{T} \int_0^T U(t) \cos n\omega_1 t dt, \\ b_n &= \frac{2}{T} \int_0^T U(t) \sin n\omega_1 t dt, \end{aligned}$$

где a_n, b_n — амплитуды косинусных и синусных членов разложения $U(t)$.

Такое представление, называемое *синусно-косинусным*, — наиболее простое для вычисления, но неудобно тем, что на каждой частоте присутствуют две компоненты. Существует другое представление, называемое *амплитудно-фазовым*, которое имеет вид

$$U(t) = \frac{U_0}{2} + \sum_{n=1}^{\infty} U_n \cos(n\omega_1 t + \varphi_n), \quad (3.4)$$

где $U_n = \sqrt{a_n^2 + b_n^2}$ — амплитуда (модуль), $\varphi_n = \arctg \frac{b_n}{a_n}$ — фаза (аргумент n -й гармоники).

Ряд Фурье можно представить и в комплексной форме

$$U(t) = \frac{1}{2} \sum_{n=-\infty}^{+\infty} U_n e^{jn\omega_1 t},$$

где $\omega_1 = 2\pi/T = 2\pi f_1$, $U_n = \frac{2}{T} \int_0^T U(t) e^{-jn\omega_1 t} dt$.

Так, для периодического сигнала $U(t)$, повторяющегося с частотой $\omega = 2\pi/T$ и периодом T , заданного в интервале $t_1 < t < t_2$ (рис. 3.12), постоянная составляющая и модуль определяются из выражений

$$\frac{U_0}{2} = \frac{1}{T} \int_{t_1}^{t_2} U(t) dt, \quad (3.5)$$

$$U_n = \frac{2}{T} \int_{t_1}^{t_2} U(t) e^{-jn\omega_1 t} dt. \quad (3.6)$$

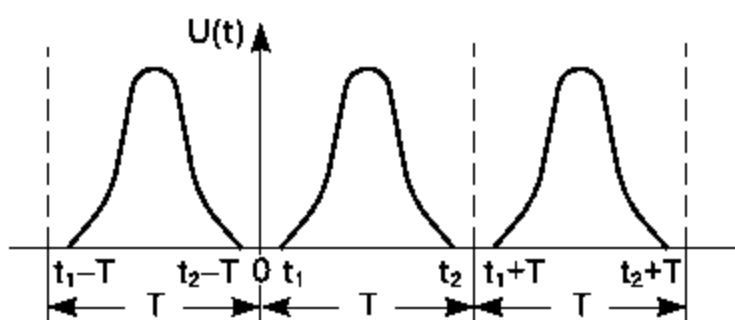


Рис. 3.12. Периодический сигнал

Для периодической последовательности прямоугольных импульсов амплитуды U и длительности τ (рис. 3.13) среднее значение (постоянная составляющая) вычисляется по формуле (3.7):

$$\frac{U_0}{2} = \frac{1}{T} \int_0^T U dt = \frac{\tau}{T} U. \quad (3.7)$$

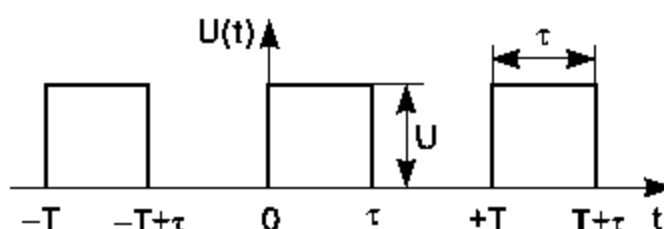


Рис. 3.13. Последовательность прямоугольных импульсов

Амплитуда и фаза n -й гармоники рассчитываются по соответствующим формулам [3]:

$$U_n = \frac{2U}{\pi n} \left| \sin \frac{n\omega_1 \tau}{2} \right|, \quad (3.8)$$

$$\varphi_n = \frac{n\omega_1 \tau}{2}. \quad (3.9)$$

После подстановки выражений (3.8), (3.9) в формулу (3.4) получим

$$U(t) = U \left[\frac{\tau}{T} + \frac{2}{\pi} \sum_{n=1}^{\infty} \frac{\sin \frac{n\omega_1 \tau}{2}}{n} \cos(n\omega_1 t - \varphi) \right]. \quad (3.10)$$

Диаграммы распределения по частоте амплитуд и фаз гармонических составляющих называются *спектральными диаграммами сигнала*, а линии, соответствующие амплитудам и фазам гармоник, — *спектральными линиями*.

Структура частотного спектра периодического сигнала полностью определяется двумя характеристиками — *амплитудой* и *фазой*, т.е. модулем U_n и аргументом комплексной амплитуды φ_n .

Если интересуют не значения амплитуд и начальных фаз гармоник, а только частоты, на которых они присутствуют, то говорят о *спектре*

частот сигнала. Представление сигнала в частотной области называется *спектральным*. Спектр периодического сигнала состоит из отдельных линий, соответствующих дискретным частотам: $0, \omega_1, \omega_2 = 2\omega_1, \omega_3 = 3\omega_1$ и т.д., и называется *линейчатым* или *дискретным*. Наглядное представление о ширине спектра и относительной величине его отдельных гармоник дает графическое изображение спектра (рис. 3.14). По оси ординат отложены модули амплитуд, по оси абсцисс — частоты соответствующих гармонических составляющих. Совокупность амплитуд гармонических составляющих называется *амплитудным спектром колебания*.

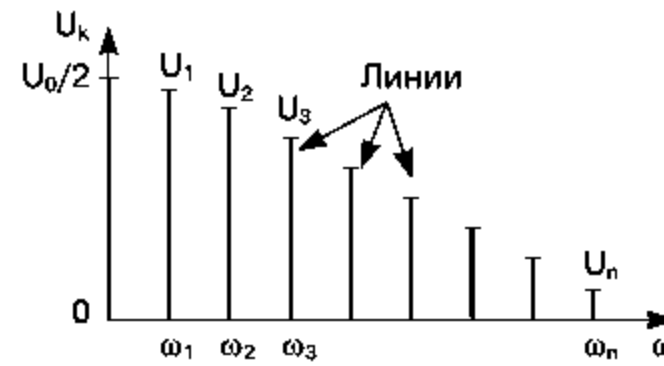


Рис. 3.14. Амплитудный спектр

Для исчерпывающей характеристики спектра подобное изображение должно быть дополнено заданием фаз отдельных гармоник.

Так, элементарное гармоническое колебание (рис. 3.15)

$$U(t) = U_k \cos\left(\omega_1 t - \frac{\pi}{2}\right) \quad (3.11)$$

можно представить в виде двух составляющих — *амплитудной*, отображенной единственной линией высотой U (рис. 3.16, а), и *фазовой* (рис. 3.16, б), изображенной линией, равной значению фазы $\varphi = -\pi/2$.

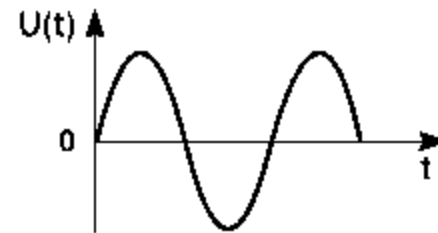


Рис. 3.15. Гармоническое колебание

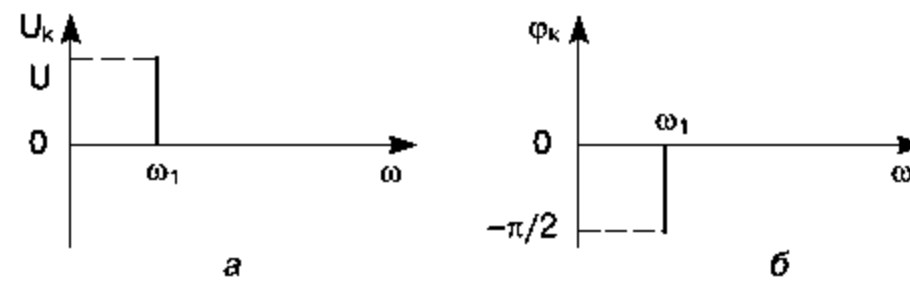


Рис. 3.16. Спектр гармонического колебания:

а — амплитудная составляющая; б — фазовая составляющая

Если сигнал $U(t)$ состоит из суммы двух гармоник, например первой и третьей (рис. 3.17)

$$U(t) = U \cos\left(\omega_1 t - \frac{\pi}{2}\right) + \frac{U}{3} \cos\left(3\omega_1 t - \frac{\pi}{2}\right), \quad (3.12)$$

то его представление в виде двух составляющих спектра имеет вид, приведенный на рис. 3.18.

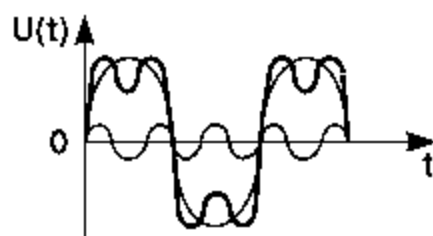


Рис. 3.17. Сложение гармоник

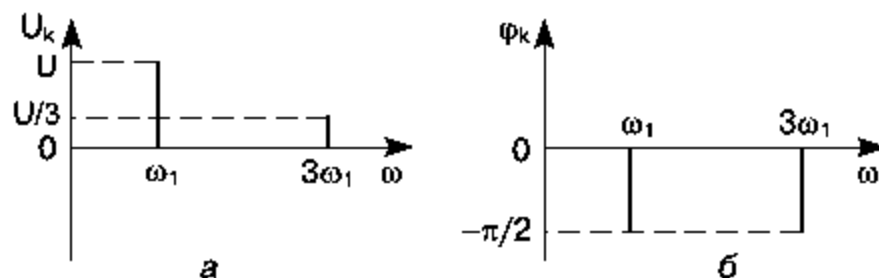


Рис. 3.18. Спектральное представление сигнала:
а — амплитудные составляющие; б — фазовые составляющие

Спектром сигнала называется диапазон частот, составляющих данный сигнал. Для сигнала (см. рис. 3.18) спектр лежит в области частот от ω до 3ω или от f до $3f$. Абсолютной шириной полосы сигнала называется ширина его спектра. В рассматриваемом случае (см. рис. 3.18) ширина полосы равна 2ω ($2f$).

Введем еще один термин — *постоянная составляющая*. Если в сигнале имеется гармоника нулевой частоты, то она называется *постоянной составляющей*. Результат добавления такой составляющей к сигналу (3.12) показан на рис. 3.19:

$$U(t) = 1 + U \cos\left(\omega_1 t - \frac{\pi}{2}\right) + \frac{U}{3} \cos\left(3\omega_1 t - \frac{\pi}{2}\right).$$

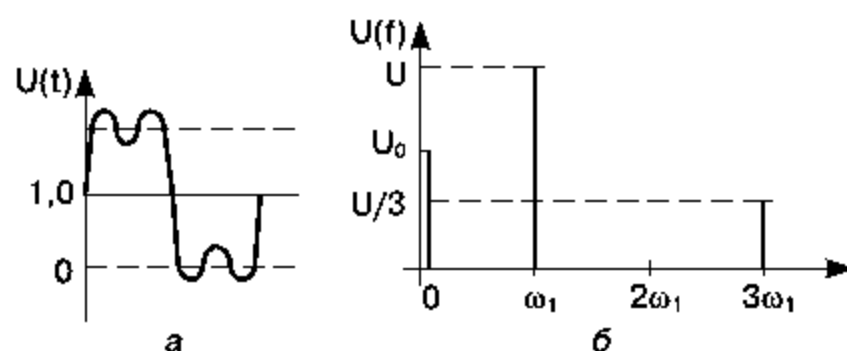


Рис. 3.19. Сигнал с постоянной составляющей (а) и его спектральное разложение (б)

При наличии этой составляющей сигнал имеет частотный член при $f = 0$ (см. рис. 3.19, б) и ненулевую среднюю амплитуду. Если постоянной составляющей нет, средняя амплитуда сигнала равна нулю.

Суммируя бесконечное число нечетных гармоник

$$U(t) = \sum_{k=1,3,5,\dots} U_k \cos(k\omega_1 + \varphi_k), \quad (3.13)$$

можно получить сигнал в форме последовательности прямоугольных импульсов (рис. 3.20). Его амплитудный и частотный спектры представлены на рис. 3.21.

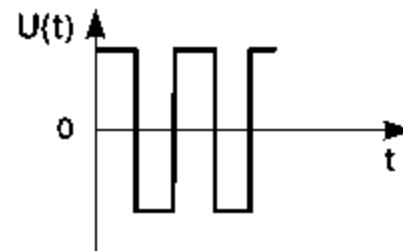


Рис. 3.20. Последовательность прямоугольных импульсов

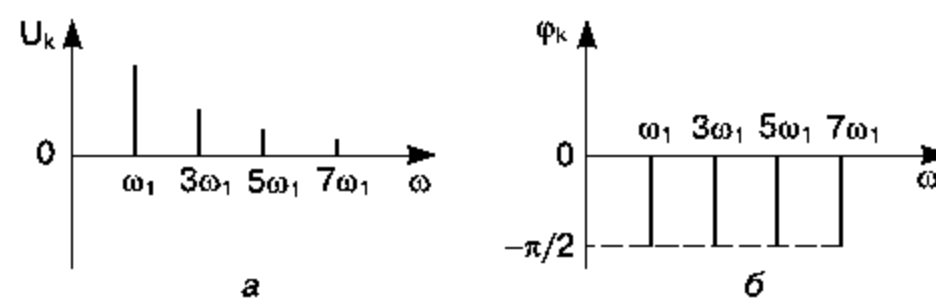


Рис. 3.21. Спектры:

а — амплитудный; б — частотный

Таким образом, периодический сигнал любой формы может быть представлен в виде суммы (в общем случае бесконечной) гармонических колебаний.

3.2.3. Спектральные характеристики наиболее распространенных детерминированных сигналов

Из всего многообразия сигналов простейшими и достаточно часто встречающимися на практике являются амплитудно-модулированный (АМ) и частотно-модулированный (ЧМ) сигналы.

Амплитудно-модулированным называется сигнал, амплитуда которого изменяется во времени по какому-либо закону, например по закону косинуса (рис. 3.22, а):

$$X(t) = \cos \Omega t = \cos 2\pi Ft. \quad (3.14)$$

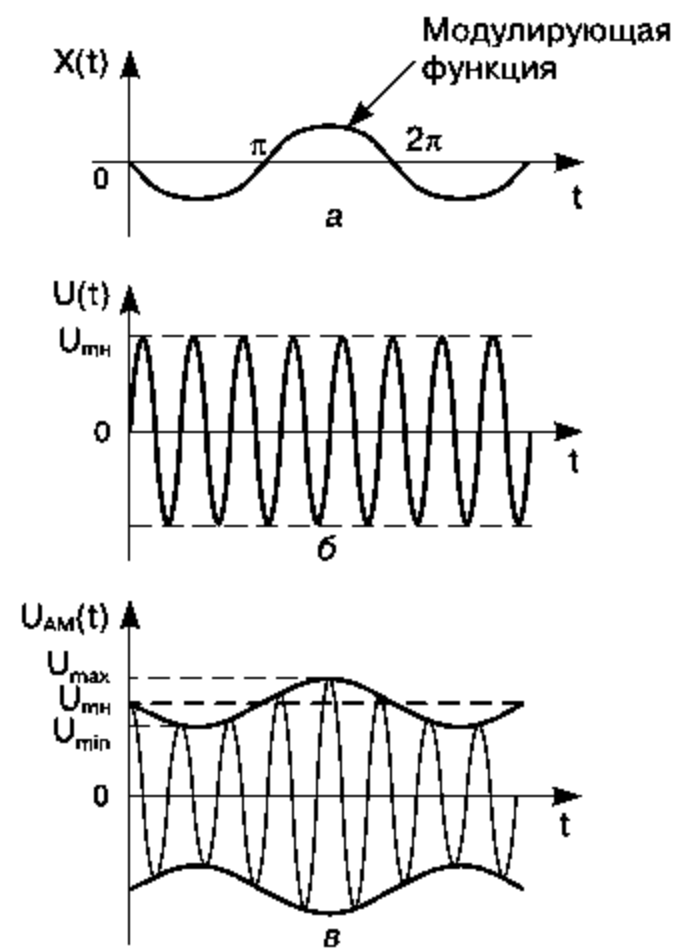


Рис. 3.22. Амплитудно-модулированный сигнал (в);
а — закон изменения амплитуды; б — несущая частота

Функция $X(t)$, определяющая закон изменения амплитуды колебания во времени, называется *модулирующей*, а ее частота F — частотой модулирования. Высокочастотное колебание (рис. 3.22, б) называется *несущим*:

$$U(t) = U_{mn} \sin(\omega_n t + \varphi_0),$$

где U_{mn} — амплитуда несущего колебания (в отсутствие модуляции);
 φ_0 — начальная фаза колебания (при $t = 0$);
 ω_n — частота несущего колебания.

Амплитудно-модулированный сигнал описывается выражением

$$U_{AM}(t) = U_{mn} [1 + mX(t)] \sin(\omega_n t + \varphi_0), \quad (3.15)$$

где $X(t)$ — функция, определяющая закон изменения амплитуды колебания $U_{mn} \sin \omega_n t$ во времени;
 m — коэффициент глубины амплитудной модуляции или просто коэффициент модуляции.

С физической точки зрения $X(t)$ характеризует степень воздействия модулирующей функции на амплитуду несущего колебания. В этом случае амплитуда несущего колебания будет изменяться от U_{max} до U_{min}

(рис. 3.22, в). Коэффициент m вычисляется как отношение разности амплитуд к их сумме

$$m = \frac{U_{\max} - U_{\min}}{U_{\max} + U_{\min}}$$

и может меняться от 0 до 1.

Амплитудный спектр АМ-сигнала состоит из трех линий (рис. 3.23), где f_n — частота несущего колебания, частоты $f_n + F$ и $f_n - F$ называются *боковыми частотами* (соответственно верхней и нижней). Разность $(f_n + F) - (f_n - F) = 2F$ называется *шириной спектра* АМ-сигнала.

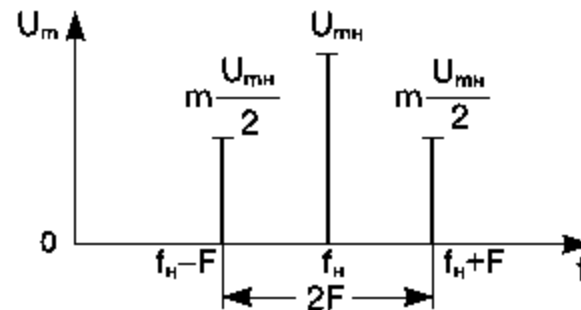


Рис. 3.23. Амплитудный спектр АМ-сигнала

Вся информация о модулирующей функции содержится в любой из боковых частот (нижняя боковая — это зеркальное отображение верхней боковой по отношению к несущей частоте f_n). Несущая не содержит никакой полезной информации, обе боковых несут одну и ту же информацию. F — частота изменения информативного параметра синусоиды — совпадает со скоростью передачи данных при использовании двух уровней амплитуды. Частота F определяет *пропускную способность* линии при данном способе кодирования. Если модулирующая функция содержит ряд спектральных составляющих, то ширина спектра оказывается равной удвоенной верхней частоте спектра (F_v) модулирующего колебания, т.е. если спектр модулирующей функции ограничен, то ограниченной оказывается ширина спектра АМ-колебания (рис. 3.24).

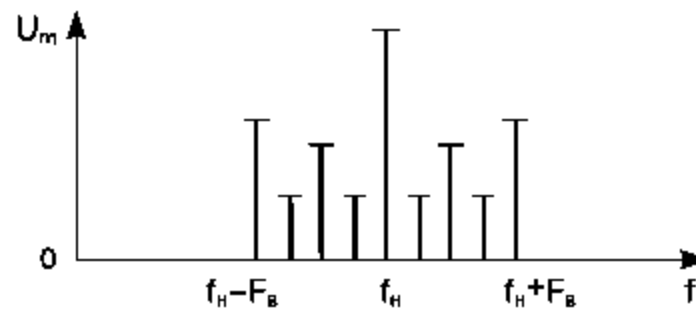


Рис. 3.24. Спектр АМ-колебания с ограниченной шириной

3.2.4. Периодическая последовательность радиоимпульсов

Частным случаем АМ-колебания служит периодическая последовательность радиоимпульсов. Радиоимпульсы получаются в случае, если модулирующая функция является периодической последовательностью прямоугольных импульсов (видеоимпульсов). Выражение, описывающее единичный видеоимпульс, можно записать в следующем виде:

$$X(t) = \begin{cases} U_0 & \text{при } -0,5\tau_u \leq t \leq 0,5\tau_u; \\ 0 & \text{при остальных значениях } t, \end{cases} \quad (3.16)$$

где τ_u — длительность видеоимпульса.

Используя разложение в ряд Фурье, можно показать, что амплитудный спектр последовательности видеоимпульсов с периодом T бесконечно широк, отдельные спектральные составляющие отстоят друг от друга на T^{-1} , а огибающая амплитудного спектра описывается функцией $\sin x/x$. Нули огибающей соответствуют частотам, вычисляемым по отношению $n\tau_u^{-1}$, где $n = 1, 2, \dots$. Графики периодической последовательности видеоимпульсов и амплитудного спектра этой последовательности показаны соответственно на рис. 3.25, а последовательности радиоимпульсов и ее амплитудного спектра — на рис. 3.26.

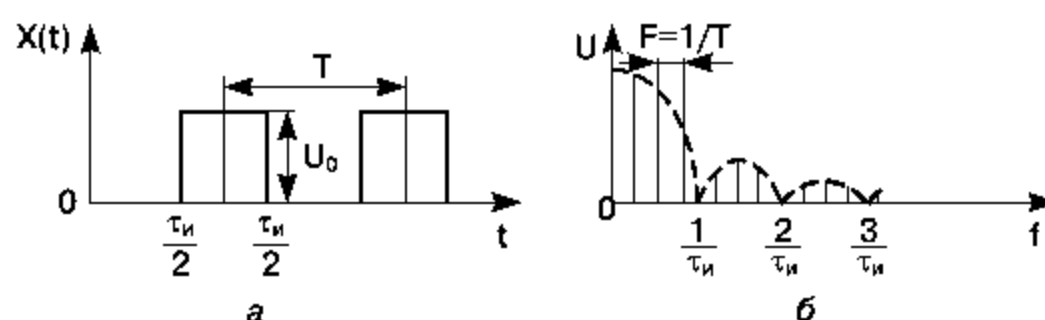


Рис. 3.25. Видеоимпульсы:
а — последовательность; б — амплитудный спектр последовательности

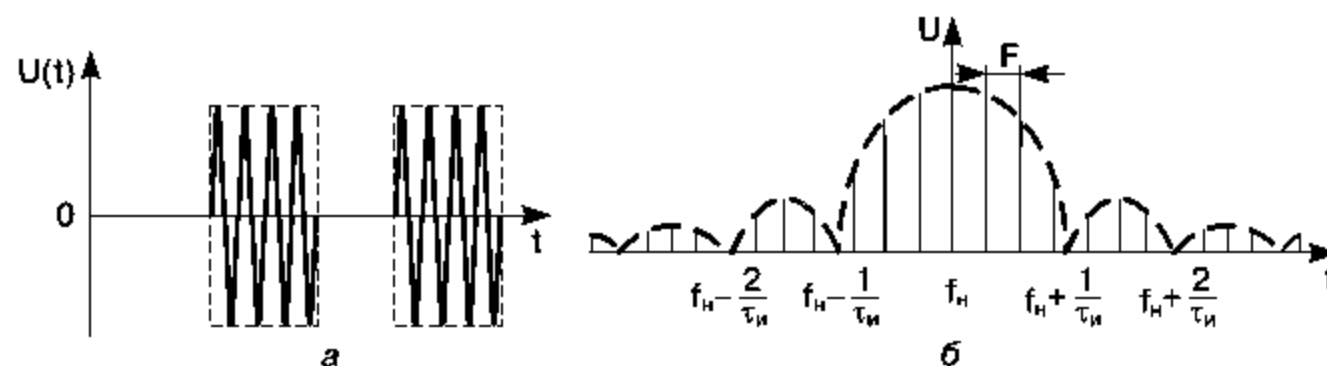


Рис. 3.26. Радиоимпульсы:
а — последовательность; б — амплитудный спектр последовательности

Амплитудный спектр последовательности радиоимпульсов, как и последовательности видеоимпульсов, теоретически бесконечно широк. Для колебаний, амплитудный спектр которых теоретически бесконечен, вводят понятие реальной ширины спектра. Под *реальной шириной спектра* понимается область частот, в которой сосредоточена большая часть энергии сигнала (около 99%). Для импульсных сигналов реальная ширина спектра рассчитывается следующим образом:

$$\Delta f \approx \frac{1}{\tau_k}. \quad (3.17)$$

3.2.5. Последовательность радиолокационных импульсов

Этот случай (рис. 3.27) характеризуется очень малым отношением длительности импульса к периоду повторения, т.е. $\frac{\tau}{T} \ll 1$. Величина, обратная этому отношению $N = \frac{T}{\tau} \gg 1$, называется *скважностью* импульсов. Большая по сравнению с длительностью импульса величина периода повторения T приводит к необходимости учитывать очень большое число гармоник.

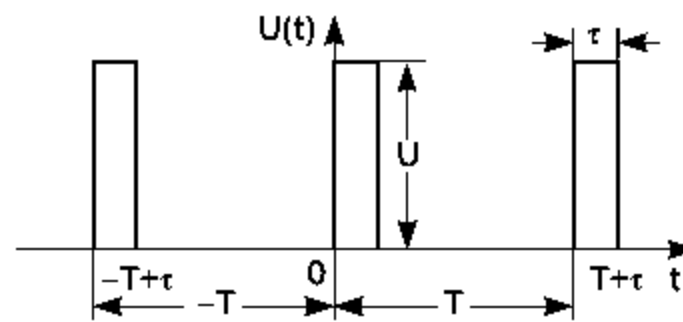


Рис. 3.27. Последовательность видеоимпульсов

Спектр в этом случае имеет вид, приведенный на рис. 3.28. Расстояние между спектральными линиями очень мало $\left(\omega_1 = \frac{2\pi}{T}\right)$, а амплитуды соседних гармоник близки по величине, что видно из формулы

$$A_k = \frac{2U}{\pi k} \sin \left| \frac{k\omega_1 \tau}{2} \right|,$$

которую в данном случае удобно записать в несколько видоизмененном виде:

$$A_k = \frac{2U}{\pi k} \sin \left(k\pi \frac{\tau}{T} \right). \quad (3.18)$$

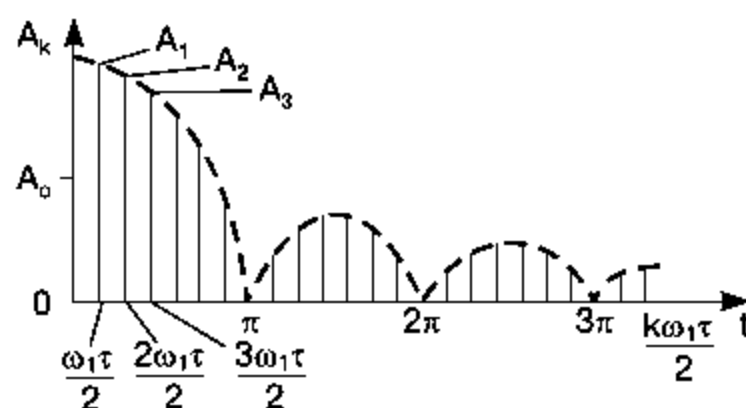


Рис. 3.28. Спектр видеоимпульсов

Из-за ничтожно малой величины отношения $\frac{\tau}{T}$ аргумент синуса с ростом k изменяется медленно, а при малых значениях k приближенно можно считать

$$\sin\left(k\pi\frac{\tau}{T}\right) \cong k\pi\frac{\tau}{T},$$

а амплитуды гармоник равными

$$A_k \cong \frac{2U}{\pi k} \frac{k\pi\tau}{T} = U \frac{2\tau}{T}. \quad (3.19)$$

Заметим, что при $\frac{\tau}{T} \ll 1$ постоянная составляющая, равная $A_0 = U \frac{\tau}{T}$, вдвое меньше, чем A_1 . Важно также, что постоянная составляющая во много раз меньше амплитуды импульса U .

3.2.6. Спектр непериодического колебания

Сигналы электросвязи, используемые для передачи информации, являются непериодическими. Непериодическое колебание в виде импульса (рис. 3.29, а) можно рассматривать как сумму бесконечно большого числа бесконечно малых по амплитуде гармонических колебаний, частоты которых располагаются очень близко друг к другу и зажимают в общем случае всю полосу частот. Таким образом, спектр непериодического колебания является непрерывным или сплошным (рис. 3.29, б). При этом принято говорить о *спектральной плотности амплитуд*.

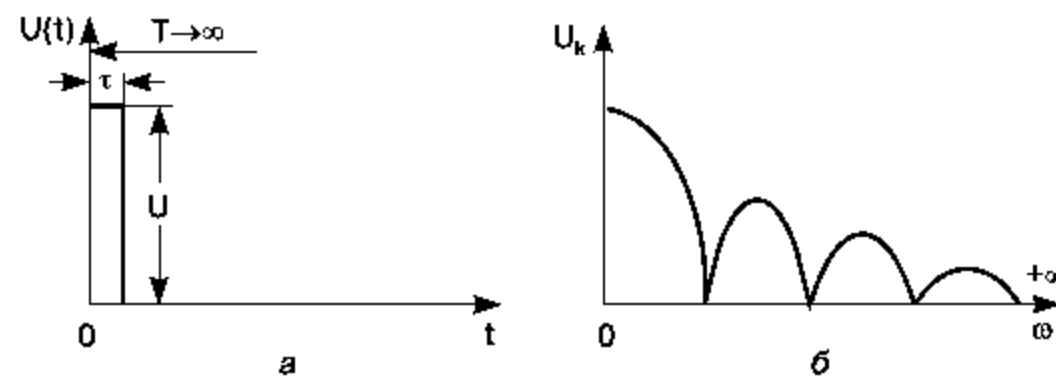


Рис. 3.29. Непериодическое колебание:
а — идеальный импульс; б — спектральное разложение

3.2.7. О значении частотного представления сигнала

Результатом рассмотренного частотного способа представления сигналов является вывод о том, что и аналоговые, и цифровые сигналы состоят из набора гармоник, принадлежащих некоторому диапазону электромагнитных частот. *Спектром* сигнала называется диапазон частот, составляющих данный сигнал. Для сигнала, приведенного на рис. 3.18, спектр лежит в области частот от f до $3f$. *Абсолютной шириной полосы* сигнала называется ширина его спектра. В рассматриваемом случае (см. рис. 3.18) ширина полосы равна $2f$. Ширина полосы многих сигналов, таких, например, как сигнал, показанный на рис. 3.29, а, бесконечна. Впрочем, большая часть энергии таких сигналов приходится на относительно узкую полосу частот, называемую *эффективной полосой* или просто *полосой*. От того, какие частоты используются в сигнале, зависят его передающие свойства и степень искажения сигнала.

Как аналоговые, так и цифровые сигналы могут распространяться в соответствующих передающих средах (радиолинии, кабельные линии, волоконно-оптические линии, спутниковая связь и др.). Искажение передающим каналом синусоиды какой-либо частоты приводит в конечном счете к искажению передаваемого сигнала любой формы, особенно если синусоиды различных частот искажаются неодинаково. В аналоговых сигналах искажения могут снизить качество сигнала, а в цифровых могут появиться ошибочные биты: двоичная единица, превратившаяся в нуль, и наоборот. Так, если аналоговый сигнал передает речь, то за счет искажения обертонов (боковых частот) изменяется тембр голоса.

К факторам, воздействующим на сигнал в процессе передачи, относятся:

- а) *затухание* — относительное уменьшение амплитуды или мощности сигнала при передаче по каналу сигнала определенной частоты. Затухание A (дБ) вычисляют по формуле

$$A = 10 \lg \frac{P_{\text{вых}}}{P_{\text{вх}}},$$

где $P_{\text{вых}}$, $P_{\text{вх}}$ — мощность сигнала на выходе и входе канала соответственно.

Поскольку при эксплуатации канала часто заранее известна основная частота передаваемого сигнала, т.е. частота, гармоника которой имеет наибольшие амплитуду и мощность, то достаточно знать затухание на этой частоте, чтобы приблизительно оценить искажение передаваемых по каналу сигналов;

- б) *ограниченность полосы пропускания* определяет диапазон частот синусоидального сигнала, при которых этот сигнал передается по каналу связи без значительных искажений;
- в) *задержка*: при передаче цифрового сигнала различные компоненты, из которых он образован, достигают приемника с различными задержками, поскольку скорость распространения синусоидального сигнала по линии связи изменяется с изменением частоты;
- г) *шумы*, которые постоянно присутствуют в реальном канале. В отсутствие передаваемого сигнала в идеальной линии связи должен быть нулевой уровень электрического сигнала. Однако на практике в линии имеют место случайные всплески даже тогда, когда сигнал не передается. Эти всплески называют *уровнем шумов* в линии, и в пределе по мере затухания передаваемого сигнала его уровень становится сравнимым с уровнем шума. Низкое отношение *сигнал/шум* свидетельствует о сигнале низкого качества;
- д) *пропускная способность линии* характеризует максимально возможную скорость передачи данных по линии связи (измеряется в битах в секунду (бит/с) и производных единицах Кбит/с, Мбит/с и т.д.). Пропускная способность линии зависит не только от ее характеристик, таких как амплитудно-частотная (показывает, как затухает амплитуда синусоиды или мощность на выходе канала связи по сравнению с амплитудой или мощностью на его входе для всех возможных частот передаваемого сигнала (рис. 3.30)), но и от того, какие сигналы передаются — аналоговые или цифровые.

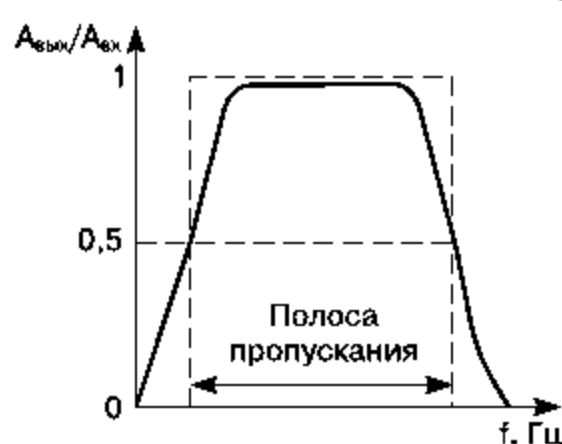


Рис. 3.30. Амплитудно-частотная характеристика

Если значимые гармоники сигнала, т.е. те гармоники, амплитуды которых вносят основной вклад в результирующий сигнал, попадают в полосу пропускания линии, то такой сигнал будет хорошо передаваться данной линией связи и приемник сможет правильно распознать информацию, отправленную по линии передатчиком. Если же значимые гармоники выходят за границы пропускания линии связи, то сигнал будет сильно искажаться, приемник будет ошибаться при распознавании информации. Воздействия различных факторов на сигнал в процессе передачи его по каналу связи показаны на рис. 3.31.

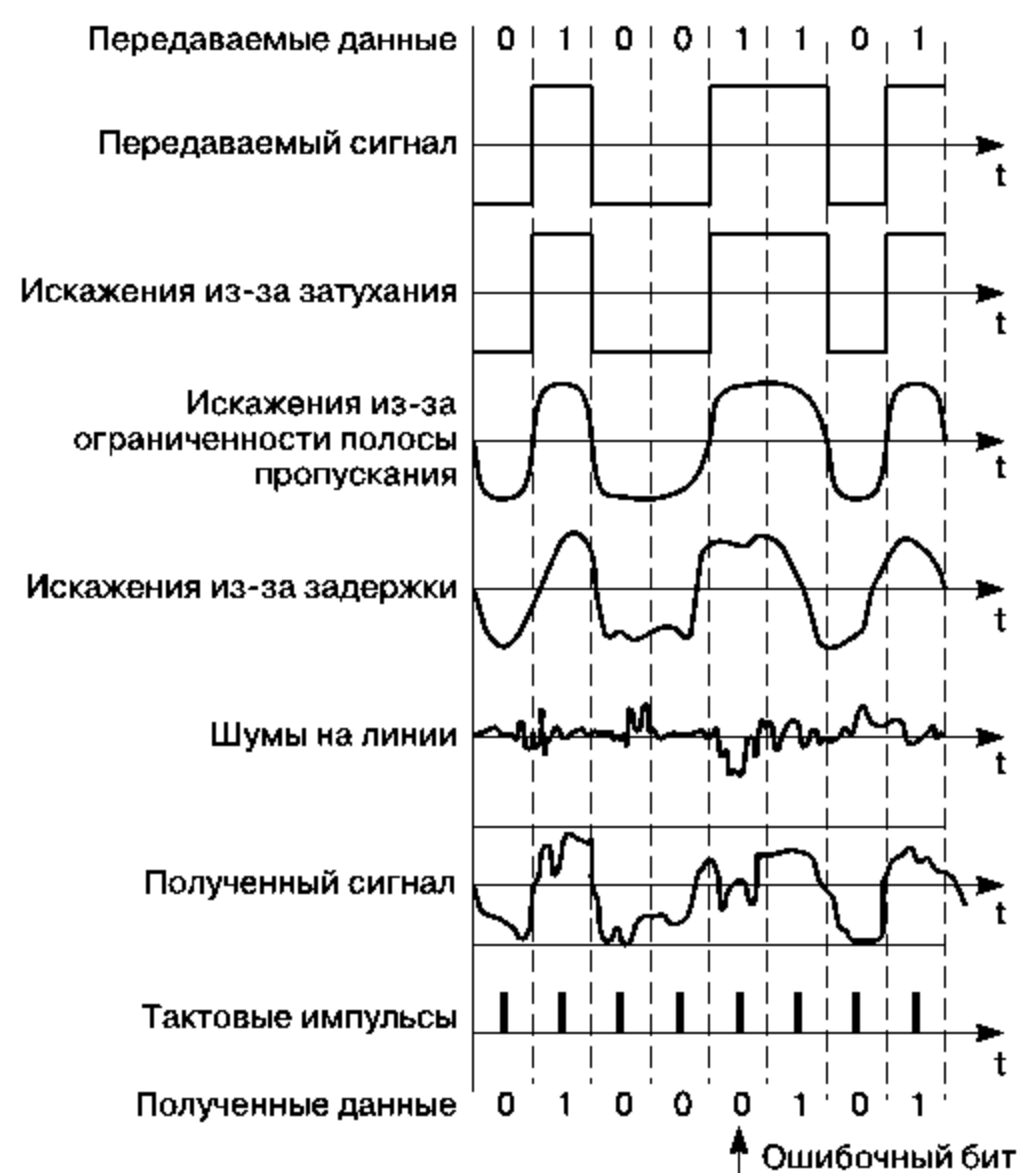


Рис. 3.31. Источники затухания и искажения сигнала

Сигналы, используемые для передачи информации, как правило, являются непериодическими и практически все имеют бесконечно широкий спектр частот. Чтобы форма сигнала в пункте приема точно совпадала с формой сигнала в пункте передачи, необходимо передавать этот бесконечно широкий спектр. Сделать это в реальных условиях не-

возможно, да в этом и нет практической необходимости. Поскольку основная часть спектра сосредоточена в конечном интервале частот, то на практике спектр сигнала ограничивают до такой степени, при которой еще возможно восстановление исходного сообщения. Например, при телефонной связи требуется выполнить два условия — разборчивость и узнаваемость; при передаче телевизионного сигнала нужно сохранить необходимую четкость изображения и т.д.

Некоторые особенности спектра сигнала имеют принципиальное значение. Так, отсутствие высокочастотных компонент означает, что для передачи сигнала требуется более узкая полоса. Или конкретный сигнал может содержать значительный диапазон частот, но любая передающая система на практике сможет вместить только ограниченную область частот, что, в свою очередь, ограничивает скорость передачи данных.

Таким образом, для понимания процесса передачи данных рассматривается ширина спектра различных сигналов, т.е. та часть спектра, которую необходимо передать по системе связи для уверенного восстановления сигнала при приеме. И анализ сигнала с точки зрения его частотного представления является более важным, чем с точки зрения временного представления. Техника определения спектра любого исходного сигнала известна. Для сигналов, которые хорошо описываются аналитически, например для последовательности прямоугольных импульсов одинаковой длительности и амплитуды (см. рис. 3.25, а), спектр легко вычисляется по формулам Фурье. Для сигналов произвольной формы, встречающихся на практике, спектр можно найти с помощью специальных приборов — *спектральных анализаторов*, которые измеряют спектр реального сигнала и отображают амплитуды составляющих гармоник на экране или распечатывают их на принтере.

3.3. СИСТЕМЫ ПЕРЕДАЧИ ДАННЫХ

3.3.1. Простейшая модель системы передачи данных.

Основные задачи передачи информации

Передача информации предполагает ее доставку от источника к получателю, которые разделены в пространстве или во времени (длительное хранение), по различным линиям связи либо обмен данными между двумя сторонами. Например, доставка курьером результатов тестиро-

вания в Москву, передача данных между рабочей станцией и сервером через общедоступную телефонную сеть или обмен речевыми сигналами между двумя телефонами через ту же самую сеть.

С практической точки зрения информация представляется в виде сообщения и передается от источника к получателю в материально-энергетической форме: либо с помощью таких носителей, как бумага, магнитная лента, магнитный или лазерный диски, либо с помощью некоторого физического процесса (ток, звуковые или электромагнитные волны), который в этом случае называют *сигналом*.

Сигнал, под которым далее будем понимать электрический сигнал, является, по сути, формой представления сообщения для передачи по системам связи. Таким образом, *сигнал* — это материальный носитель сообщения, представляющий собой физический процесс, отражающий данное сообщение путем изменения какого-либо параметра (амплитуды, фазы, частоты и др.).

Передача сообщений может быть реализована различными способами:

- ♦ непосредственно (личный контакт);
- ♦ путем доставки информационных документов курьером или по почте;
- ♦ с использованием систем автоматизированной передачи информации.

Достоинствами ручных способов передачи информации являются:

- ♦ минимальные капитальные затраты;
- ♦ высокая достоверность;
- ♦ конфиденциальность и др.

Основной их недостаток — низкая скорость передачи.

Сегодня невозможно представить обмен информацией между абонентами без использования современной *индустрии компьютерной передачи данных*, без интегрированных систем, выполняющих передачу и обработку всех типов данных и информации, что позволяет легко и единообразно получать доступ практически ко всем источникам данных и информации во всем мире. Под *данными* здесь будем понимать информацию, представленную в цифровой форме.

Совокупность технических средств, служащих для передачи информации на расстояние, называют системой передачи информации или *системой передачи данных*. Простейшая модель такой системы представлена блок-схемой на рис. 3.32.

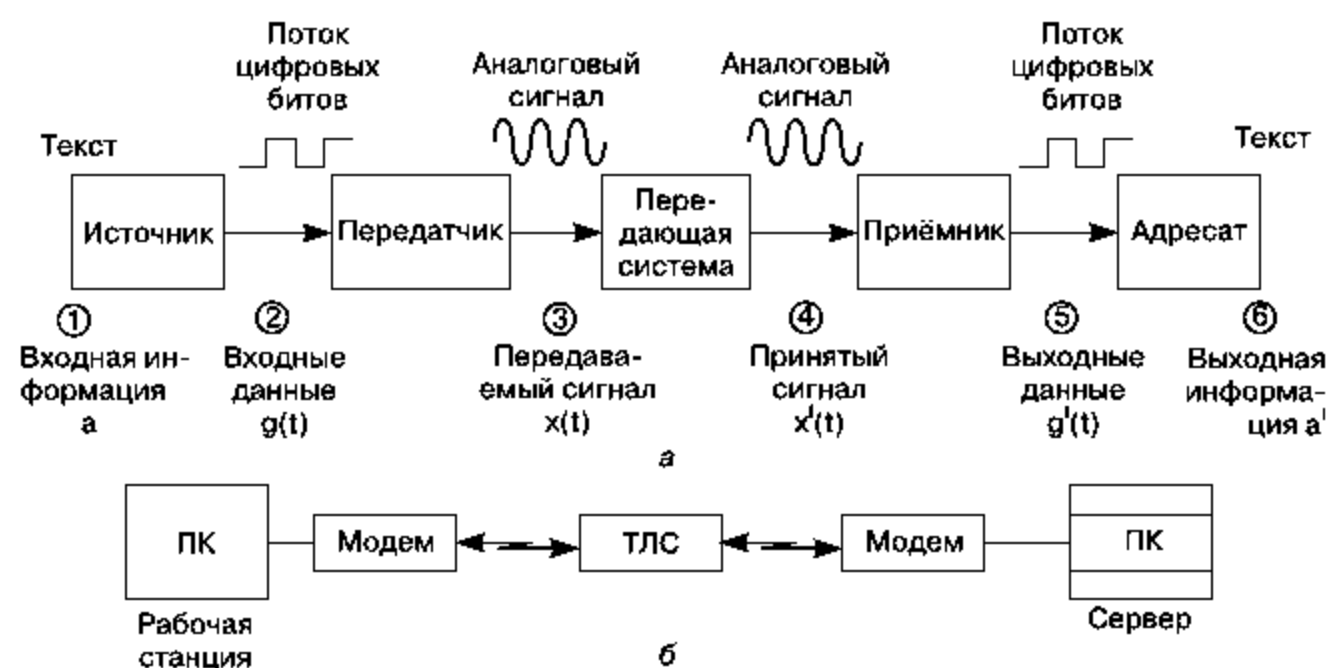


Рис. 3.32. Простейшая модель системы передачи данных:

а — общая блок-схема; б — пример

Рассмотрим основные элементы модели.

Источник генерирует данные, требующие передачи. Примером источника являются телефон, персональный компьютер.

С источником информации связано определенное множество сообщений $A = \{a_1, a_2, \dots, a_n\}$. Генерация некоторого сообщения a_i заключается в случайном выборе его из множества возможных. Какое это будет сообщение конкретно, получателю заранее неизвестно. Известно лишь, что оно принадлежит определенному множеству, т.е. $a_i \in A$, $i = \overline{1, n}$, где A может быть:

- ♦ конечным множеством символов;
- ♦ конечным набором детерминированных функций времени;
- ♦ бесконечным множеством значений некоторой непрерывной физической величины (температуры, давления и др.).

Сообщение, принадлежащее конечному множеству возможных значений, называется *дискретным*, а выбираемое из бесконечного множества — *непрерывным*.

Передатчик. Как правило, сообщения (данные) генерируются источником и не передаются непосредственно в той форме, в которой они были созданы. Для передачи сообщения его преобразуют в электромагнитный сигнал, который можно передать через конкретную передающую систему. Такое преобразование предполагает три этапа:

- ♦ преобразование формы;
- ♦ модуляцию;
- ♦ кодирование.

Это преобразование выполняет передатчик. Например, модем принимает от персонального компьютера цифровой поток двоичных сигналов (рис. 3.32, б) и преобразует его в аналоговый сигнал, который может обрабатываться телефонной сетью (ТЛС).

Передающая система — простая линия связи или сложная сеть (см. рис. 3.59), соединяющая источник информации с адресатом.

Приемник — устройство, получающее сигнал с выхода передающей системы и преобразующее его в форму, пригодную для обработки адресатом. Например, модем получает из ТЛС аналоговый сигнал и преобразует его в цифровой (рис. 3.32, б).

Адресат получает от приемника поступающие данные.

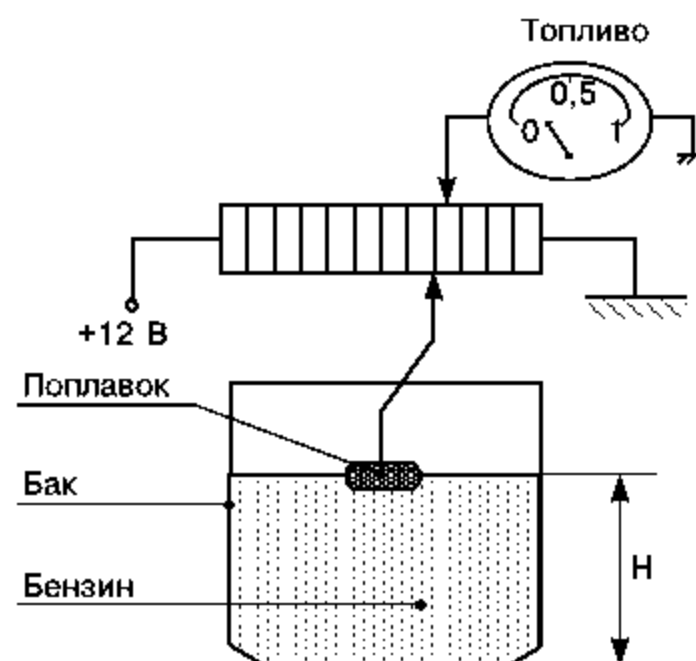


Рис. 3.33. Пример реализации простой модели системы передачи данных

В качестве простого примера *системы передачи данных* рассмотрим систему измерения уровня бензина в баке автомобиля (рис. 3.33). Источником информации является бензобак с поплавковым датчиком — первичным преобразователем (ПП). Входная информация — сообщение а об уровне H бензина в баке является аналоговой незлектрической величиной. Для передачи сообщения адресату (водителю) перемещение поплавка преобразуется во входной сигнал $g(t)$ — напряжение, пропорциональное величине уровня H, который без изменения, т.е. сигналы $g(t)$ и $x(t)$ идентичны, передается по проводам на приемник — прибор, проградуированный в относительных единицах и отображающий выходную информацию a' в виде

$$U_H = kH,$$

где H — уровень бензина в баке;
k — коэффициент пропорциональности.

Другим примером такой модели (см. рис. 3.32, *a*), позволяющим лучше понять процессы, связанные с передачей данных от одного ПК к другому, является электронная почта. Пользователь ПК для передачи текстового сообщения $a(t)$ другому пользователю запускает на компьютере программу электронной почты и вводит с клавиатуры сообщение. Строка символов попадает в буфер оперативной памяти, где представляется в виде последовательности битов g . ПК соединяется с передающей средой — телефонной линией связи (ТЛС) через модем, на который входные данные приходят по какой-либо шине или кабелю в виде последовательности перепадов напряжения $g(t)$, представляющих биты.

Изначально телефонная сеть была разработана для получения, коммутации и передачи аналоговых сигналов речевого диапазона частот ($\approx 30\text{—}3400$ Гц) и неприменима для оперирования с цифровыми сигналами, поэтому цифровые устройства присоединяются к сети с помощью модема (модулятора-демодулятора), преобразующего цифровые данные в аналоговые сигналы, и наоборот. Модулятор преобразует входящий поток $g(t)$ в сигнал $x(t)$, пригодный для передачи по ТЛС. В процессе передачи сигнала $x(t)$ через передающую среду происходят некоторые его искажения в результате действия различных помех. Поэтому полученный на выходе ТЛС сигнал $x'(t)$ может отличаться от сигнала $x(t)$. На основе сигнала $x'(t)$ и знаний о передающей среде приемник пытается восстановить исходный сигнал $x(t)$, в результате чего получается последовательность бит $g^1(t)$. Затем эти данные предоставляются адресату через устройство вывода (принтер, экран) в виде сообщения $a^1(t)$, являющегося точной копией сообщения a .

В примере с телефонным разговором (см. рис. 3.32) входными данными является сообщение a в форме звуковых волн, которые преобразуются телефоном в электрические сигналы некоторой частоты, а затем без изменения передаются по телефонной линии. Следовательно, входной $g(t)$ и передаваемый $x(t)$ сигналы идентичны. В результате действия помех при передаче через ТЛС полученный на ее выходе сигнал $x'(t)$ будет отличаться от $x(t)$. Однако, несмотря на это, сигнал $x'(t)$ преобразуется в форму звуковых волн без какой-либо коррекции и улучшения качества. Таким образом, выходное сообщение a^1 не является точной копией входного сообщения a , но для абонента, принимающего это звуковое сообщение, оно, как правило, понятно.

Рассмотренная модель и простые примеры, конечно, не позволяют представить всей сложности технической реализации системы передачи данных и круг задач, решаемых ею, а именно:

- ♦ *эффективное совместное использование возможностей средств связи, как правило, несколькими сообщающимися уст-*

ройствами (распределение передачи между разными пользователями, управление нагрузкой каналов и др.);

- ♦ *генерирование сигнала*, форма и интенсивность которого должны обеспечивать его прохождение через передающую среду и давать возможность приемнику интерпретировать сигнал как данные;
- ♦ *синхронизация* между передатчиком и приемником. Приемник должен иметь возможность определять начало и конец передачи сигнала, к тому же ему должна быть известна длительность каждого элемента сигнала;
- ♦ *управление информационным обменом*. При обмене данными между двумя сторонами необходимо оговаривать: режим обмена (работают устройства одновременно или по очереди), объем данных, который можно передавать за один раз, формат данных, перечень действий при возникновении ошибки, обеспечение гарантии, что источник не будет поставлять адресату данные быстрее, чем последний сможет их принять и обработать;
- ♦ *адресация и маршрутизация*. В случае если средство связи используется больше чем двумя устройствами, то источник должен указать на конкретно планируемого адресата. Передающая система должна гарантировать, что передаваемые данные получит адресат и только он. Кроме того, передающая система сама может быть сетью, в которой существуют различные пути передачи данных. Следовательно, необходимо выбрать определенный маршрут через эту сеть.

Из этого неполного перечня задач можно понять, что между простой идеей передачи данных от источника к адресату и ее реализацией лежит большой круг технических проблем, основными из которых являются:

- ♦ поведение информационных сигналов, распространяемых в передающей среде (затухание, амплитудные искажения, искажения от задержки и др.);
- ♦ различия между цифровыми и аналоговыми сигналами, а также между цифровой и аналоговой передачами;
- ♦ передающие среды и их характерные особенности;
- ♦ способы представления аналоговых и цифровых данных;
- ♦ основные характеристики систем передачи данных (ширина полосы, скорость передачи данных, пропускная способность и др.).

3.3.2. Данные и сигналы

Различают два типа данных — *аналоговые* и *цифровые*. Источниками аналоговых данных являются телефоны, видеокамеры, звуко- и видеовоспроизводящая аппаратура, а источниками цифровых — компьютеры и вычислительные устройства.

В общем случае *аналоговые данные* являются функцией времени, принадлежат ограниченному спектру частот, и, следовательно, их можно представить электромагнитным сигналом, имеющим аналогичный спектр. *Цифровые* же данные можно представить цифровыми сигналами, в которых используется два уровня постоянного напряжения: один уровень, например высокий, представляет двоичную единицу, а второй — двоичный ноль.

Однако существующие возможности этим не исчерпываются: *цифровые данные* можно представить аналоговыми сигналами, а *аналоговые данные* — цифровыми сигналами. Преобразование данных в сигналы может быть выполнено различными способами. Цифровые данные преобразуются в аналоговый сигнал с помощью модема, который преобразует серию двоичных импульсов напряжения в аналоговый сигнал, занимающий определенный спектр частот с центром на несущей частоте и способный распространяться в среде, подходящей для этой несущей частоты. Для большинства модемов этот спектр частот совпадает со спектром речи, что позволяет передавать эти данные по обычным голосовым телефонным линиям.

Для преобразования аналоговых данных в цифровые сигналы используют так называемую *дискретную модуляцию*, основанную на дискретизации непрерывных процессов как по амплитуде, так и по времени. Затем дискретные значения оцифровываются. Простейшим методом в этом случае является импульсно-кодовая модуляция (ИКМ) (рис. 3.34), которая широко применяется в цифровой телефонии. ИКМ опирается на теорему о дискретном представлении, которая формулируется следующим образом:

Если из сигнала $f(t)$ через постоянные промежутки времени производится выборка, причем скорость выборки больше, чем удвоенная максимальная частота сигнала, то такие выборки содержат всю информацию об исходном сигнале.

Отметим, что эти выборки являются аналоговыми и для их преобразования в цифровые каждой из них должен быть присвоен двоичный код. Операцию преобразования в этом случае выполняет устройство, называемое аналого-цифровым преобразователем (АЦП). Данные по каналам связи передаются в виде последовательности нулей и единиц, т.е. в цифровой форме. Таким образом, данные могут кодироваться в сигналы разными способами (рис. 3.35).

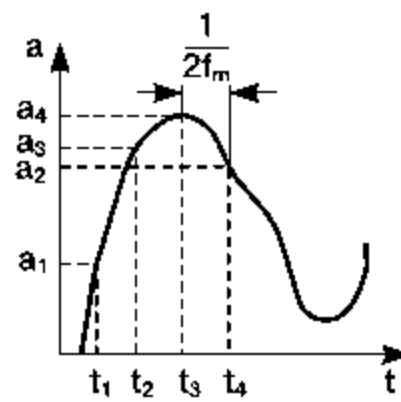


Рис. 3.34. Диаграмма ИКМ:

$a_1, \dots, a_4$ — амплитуды оцифрованного сигнала; $t_1, \dots, t_4$ — моменты квантования ($t_2 - t_1 = t_4 - t_3 = T = \frac{1}{2f_m}$ — шаг квантования)

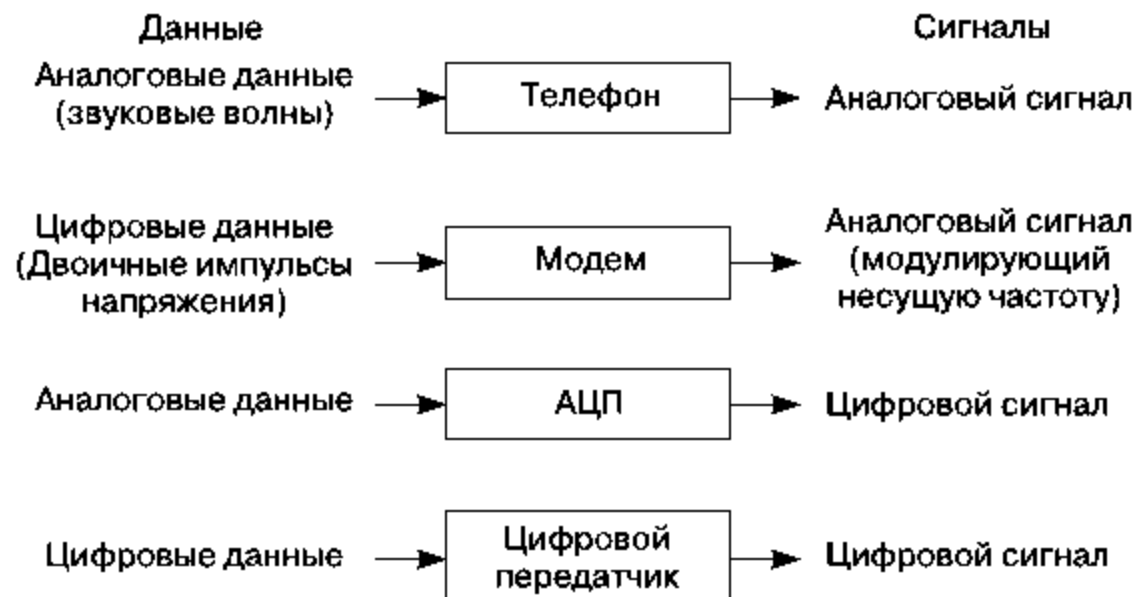


Рис. 3.35. Преобразование данных в сигнал

Поскольку в аналоговых средствах связи используются определенные диапазоны частот, то с целью генерации сигнала $x(t)$ нужного диапазона аналоговые данные $g(t)$ модулируются некоторой несущей частотой f_n (модуляция — процесс кодирования исходных данных несущим сигналом с частотой f_n).

Преобразование выполняется с помощью модема, после чего эти данные могут передаваться по аналоговой линии связи. Основными технологиями кодирования в этом случае являются амплитудная, частотная и фазовая модуляции. Суть любой из них заключается в видоизменении одной или нескольких характеристик несущей частоты с целью представления двоичных данных.

В основе преобразования аналоговых данных (речь, видеоизображение и др.) в цифровой сигнал лежит дискретизация — квантование непрерывных сигналов по времени и по уровню (см. рис. 3.10). Прин-

ципы квантования непрерывных сигналов рассмотрим на примере ИКМ. Амплитуда исходной непрерывной функции измеряется с заданным периодом (см. рис. 3.34), что обеспечивает *дискретизацию по времени*. Затем каждый замер представляется в виде двоичного числа определенной разрядности, что означает *дискретизацию по уровню* — непрерывное множество возможных значений амплитуды заменяется дискретным множеством ее значений. Подобную функцию и выполняет АЦП (кодирует аналоговые данные потоком битов).

Простейшей формой цифрового кодирования цифровых данных является присвоение одного уровня напряжения двоичному нулю, а другого — двоичной единице. Для увеличения производительности используются и другие, более сложные схемы, заключающиеся в изменении спектра сигнала и предоставлении возможности синхронизации.

Таким образом, любую информацию, как аналоговую (речь, изображение, видео и др.), так и цифровую (цифровые данные), можно кодировать аналоговыми или цифровыми сигналами. Выбор конкретной системы кодирования зависит от требований, которым она должна отвечать, и имеющихся сред и средств связи.

3.4. ПЕРЕДАЧА ДАННЫХ НА РАССТОЯНИЕ

3.4.1. Проблемы передачи.

Прямая и косвенная связь объектов

Передача данных от передатчика к приемнику происходит через некоторую передающую среду. *Передающие среды* подразделяются на *проводные* и *беспроводные*. В обоих случаях передача данных происходит путем распространения электромагнитных волн. Только в проводных средах (витая пара, коаксиальный кабель, оптоволокно) они *направляются* по физическому каналу связи, а в беспроводных (воздух, вакуум, морская вода) они *не направляются*. Путь передачи данных между двумя устройствами, по которому сигналы распространяются непосредственно от передатчика к приемнику без промежуточных устройств, за исключением усилителей и ретрансляторов, используемых для увеличения интенсивности сигнала, называется *прямым каналом связи*. Этот термин можно использовать как в проводных, так и в беспроводных средах.

Если направляемая передающая среда обеспечивает *прямой канал связи* между двумя системами и вся среда используется только ими, то такая среда называется *двухточечной*, а связь — *прямой* (рис. 3.36, а), на-

пример КС между двумя коммутирующими узлами. В этом случае две системы совместно используют двухточечный канал связи, их объекты могут общаться напрямую, т.е. данные и управляющая информация передаются непосредственно от одного объекта к другому без промежуточных активных агентов. При *многоточечной* проводной связи эта же среда может совместно использоваться более чем двумя устройствами, например, канал связи, соединяющий рабочие станции локальной сети (рис. 3.36, б). Если же системы соединяются с помощью коммутируемой сети связи (рис. 3.36, в), то два конечных объекта при передаче данных будут зависеть от работы других объектов и связь между ними является косвенной. В предельном случае два объекта могут даже не принадлежать к одной коммутируемой сети, а быть связанными косвенным образом с помощью двух или большего числа сетей. Такой набор взаимодействующих сетей называется *объединенной сетью* или *Интернетом* (рис. 3.36, г).

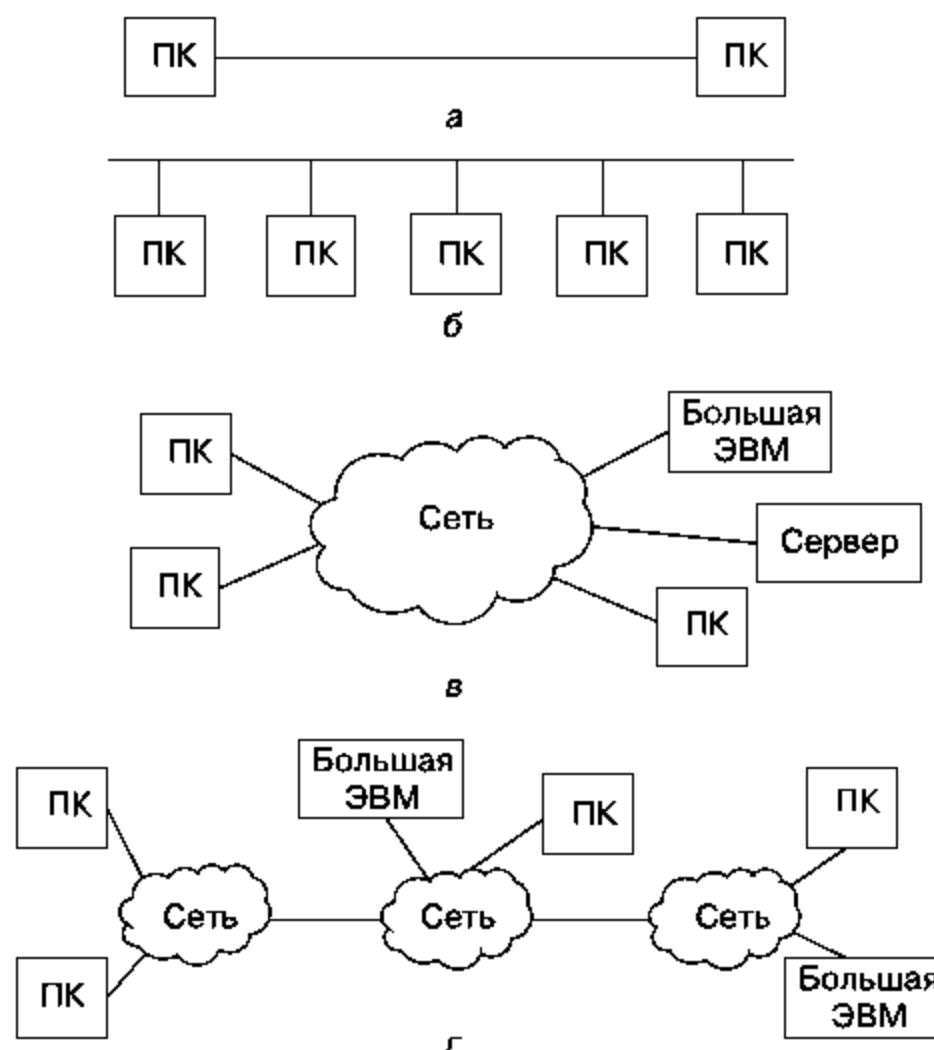


Рис. 3.36. Типы каналов связи:
а — двухточечное соединение; б — многоточечная
широковещательная сеть (ЛС); в — коммутируемая сеть;
г — объединенная сеть (Интернет)

Заметим, что термины «*объект*» и «*система*» используются в обобщенном виде. Примерами *объектов* могут служить пользовательские прикладные программы, пакеты передачи файлов, средства электронной почты, терминалы. Примеры *систем* — это компьютеры, терминалы и удаленные датчики. В некоторых случаях объекты и системы, их содержащие, совпадают (пример — терминалы). Будем считать *объектом* то, что способно передавать или принимать информацию, а *системой* — физически идентифицируемую сущность, содержащую один или несколько объектов. Чтобы два объекта могли успешно общаться, необходимо точно оговорить, какие данные, когда и каким образом передаются между ними. Все должно быть взаимно согласовано участвующими объектами.

Главная проблема при передаче данных — *искажение сигнала*. Для *аналоговых* сигналов оно обуславливает снижение качества получаемой информации и может повлиять на ее восприятие, для *цифровых* — доставку приемнику ошибочных битов, что приводит к ошибке при восстановлении сообщения.

И аналоговые, и цифровые данные можно передавать с помощью аналоговых или цифровых сигналов. Успешная передача данных зависит от двух факторов — качества передаваемого сигнала и характеристик передающей среды:

- ♦ *ширины полосы*: ширина полосы передаваемого сигнала, ограничиваемая передатчиком и природой передающей среды. Выражается в периодах в секунду или в герцах;
- ♦ *пропускной способности*: максимально возможная скорость, при которой информация может передаваться по данной линии связи или каналу связи;
- ♦ *скорости передачи данных*: скорость в битах в секунду (бит/с), с которой могут передаваться данные;
- ♦ *шума*: средний уровень шума в канале связи;
- ♦ *уровня ошибок*: скорость появления ошибок. Ошибкой считается прием единицы при переданном нуле, и наоборот.

Поскольку любой аналоговый или цифровой сигнал является суммой некоторого числа частотных компонент, то ключевым параметром, характеризующим сигнал, является *ширина полосы*, представляющая собой ширину диапазона частот, составляющих сигнал.

Передача данных может быть *симплексной* (сигналы передаются только в одном направлении: одна станция является приемником, а другая — передатчиком) или *дуплексной* (обе станции могут передавать сигналы одновременно).

3.4.2. Способы передачи данных

При передаче данных от источника к адресату необходимо знать:

- ♦ природу данных;
- ♦ реальные физические средства, используемые для передачи данных;
- ♦ необходимую в процессе передачи обработку или корректировку, способную обеспечить разборчивость получаемых данных.

Основным моментом при этом является то, с какими объектами приходится работать — с аналоговыми или цифровыми. Эти два термина часто используются в трех ситуациях, а именно при рассмотрении:

- ♦ данных;
- ♦ передачи сигналов;
- ♦ передачи данных.

Определим *данные* как объекты, передающие смысл или информацию. *Сигналы* — это электрическое или электромагнитное представление данных. *Передача сигнала* — это процесс физического распространения сигнала по соответствующей среде. *Передача данных* означает собственно передачу данных путем передачи и обработки сигналов.

Аналоговые данные принимают непрерывные значения в некотором диапазоне (звук, видеосигналы, температура, давление, частота), *цифровые данные* принимают только дискретные значения (текст, целые числа).

В общем случае аналоговые данные являются функцией времени и имеют ограниченный спектр частот; следовательно, такие данные можно представить электромагнитным сигналом, имеющим аналогичный спектр. Цифровые же данные можно представить цифровыми сигналами, имеющими разные уровни напряжения для каждого из двух двоичных чисел.

Различают два типа передач — *цифровую* и *аналоговую*. *Аналоговая передача* является средством передачи аналоговых сигналов $x(t)$. Аналоговый сигнал с постоянной частотой, называемой *несущим сигналом*, составляет основу такой передачи. Его частота выбирается совместимой с используемой передающей средой. Причем передаваемые данные $g(t)$, представляемые этим сигналом, могут быть как аналоговыми, например речь, так и цифровыми, например двоичные данные, проходящие через модем. В любом случае исходные данные преобразуются передатчиком в непрерывный по структурному параметру сигнал с такими характеристиками, которые обеспечивают его прохождение по данному каналу связи. Передача данных с помощью *несущего сигнала* обеспечивается мо-

дуляцией — процессом кодирования исходных данных несущим сигналом с частотой $f_{\text{н}}$. Канал, обеспечивающий передачу непрерывных (аналоговых) сигналов, называется *непрерывным каналом*.

Основными характеристиками непрерывных каналов связи являются:

- ◆ амплитудно-частотная характеристика;
- ◆ полоса пропускания;
- ◆ затухание;
- ◆ помехоустойчивость;
- ◆ шумы;
- ◆ пропускная способность;
- ◆ достоверность передачи данных.

Наибольший интерес из них представляют пропускная способность и достоверность передачи данных, поскольку они прямо влияют на производительность и надежность системы передачи и являются характеристиками как канала связи, так и способа передачи данных. В любом случае аналоговый сигнал после прохождения некоторого расстояния становится слабее (затухает).

Цифровая передача является средством передачи цифровых сигналов $x(t)$. При цифровой передаче исходные данные $g(t)$, которые также могут быть как аналоговыми, так и цифровыми, кодируются цифровым сигналом $x(t)$. Конкретная форма этого передаваемого сигнала зависит от метода кодирования и выбирается с точки зрения оптимального использования передающей среды. Канал, по которому передаются цифровые (дискретные) сигналы, называется *дискретным* или *цифровым*. Это комплекс технических средств, обеспечивающих передачу дискретного сигнала. В *цифровых каналах* на выходе передатчика $x(t)$ и входе приемника $x'(t)$ действует дискретный по структурному параметру сигнал. Способ представления дискретной информации в виде сигналов, подаваемых в линию связи, называется физическим или линейным кодированием. От способа кодирования зависит спектр сигналов и соответственно пропускная способность линии. Для передачи цифровых данных по дискретному каналу используют самосинхронизирующиеся коды, а для передачи аналоговых сигналов — ИКМ. Для качественной передачи оцифрованных непрерывных сигналов (голоса, изображения) сегодня применяют специальные цифровые сети, такие как ISDN, ATM и сети цифрового телевидения.

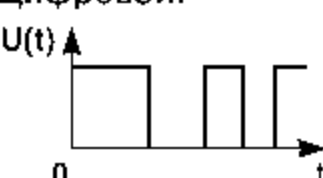
Таким образом, возможны четыре варианта передачи данных (табл. 3.1).

Таблица 3.1

Аналоговая и цифровая передача данных

Данные $g(t)$	Сигнал $x(t)$	Канал	Передача
1. Аналоговые	Аналоговый:  а) имеет тот же спектр, что и аналоговые данные; б) аналоговые данные кодируются с целью получения сигнала в другой области спектра	Аналоговый: передача сигнала через усилители	Аналоговая
2. Цифровые: кодируются на несущей частоте с использованием модема, преобразующего серию двоичных импульсов напряжения в аналоговый сигнал, занимающий определенный спектр частот с центром на несущей частоте	Аналоговый: переносчик цифровой информации — поток нулей и единиц, закодированных аналоговыми данными — амплитудой, фазой, частотой несущего сигнала	Аналоговый: передача сигнала через ретрансляторы, которые восстанавливают цифровую информацию из аналогового сигнала и создают новый, чистый аналоговый сигнал, препятствуя таким образом накоплению помех	Аналоговая: на другом конце линии другой модем демодулирует сигнал $x'(t)$ и восстанавливает исходные цифровые данные $g(t)$
3. Аналоговые: кодируются с использованием кодека (кодер-декодер), аппроксимирующего данные потоком битов	Цифровой: имеет разные уровни напряжения для представления двоичного нуля и двоичной единицы	Цифровой	Цифровая: приемник использует поток битов для воспроизведения аналоговых данных

Окончание

Данные $g(t)$	Сигнал $x(t)$	Канал	Передача
4. Цифровые	Цифровой:  использует два уровня постоянного напряжения, один из которых представляет 1, а другой 0	Цифровой	Цифровая: сигнал передается через ретрансляторы, на каждом из которых поток нулей и единиц восстанавливается из поступающего сигнала и используется для создания нового отправляемого цифрового сигнала

3.4.3. О выборе способа передачи данных

Информация передается от одной точки к другой с помощью электрических сигналов, которые могут распространяться в передающих средах в зависимости от спектра. *Аналоговый* сигнал может распространяться как в проводных (витая пара, коаксиальный кабель, оптоволокно и др.), так и в беспроводных (атмосфера, космическое пространство) передающих средах, и в общем случае имеет ограниченный спектр частот, *цифровой* же (последовательность импульсов напряжения) — только в проводной передающей среде и имеет широкий спектр частот.

При передаче данных по каналу связи применяют два основных способа физического кодирования:

- ♦ на основе синусоидального несущего сигнала — это *модуляция* или *аналоговая модуляция* (кодирование осуществляется за счет изменения параметров аналогового сигнала); устройство, осуществляющее модуляцию несущей синусоиды на передающей стороне и демодуляцию на приемной, называется *модемом* (модулятор-демодулятор);
- ♦ на основе последовательности прямоугольных импульсов — *цифровое кодирование*.

Эти способы отличаются шириной спектра результирующего сигнала. При использовании прямоугольных импульсов спектр получается весьма широким (достаточно вспомнить, что спектр идеального импульса имеет бесконечную ширину), применение синусоиды приводит к спектру меньшей ширины при той же скорости передачи данных.

Однако для реализации синусоидальной модуляции необходима более сложная и дорогая аппаратура, чем для реализации цифрового ко-

дирования. Как правило, *аналоговую модуляцию* применяют для передачи дискретных данных по каналам с узкой полосой частот, типичным представителем которых является канал тональной частоты, предоставляемый в распоряжение пользователям общественных телефонных сетей. Этот канал передает частоты в диапазоне от 300 до 3400 Гц. Таким образом, его полоса пропускания (ПП) равна 3100 Гц ($ПП = 3400 - 300 = 3100$ Гц).

Аналоговая передача является средством передачи аналоговых сигналов, при этом данные могут быть и аналоговыми, и цифровыми. В любом случае аналоговый сигнал после прохождения некоторого расстояния становится слабее (поглощается). Чтобы сигнал можно было передавать на большие расстояния, в аналоговые передающие системы вводятся усилители, повышающие энергию сигнала. К сожалению, усилители также усиливают и шумовые составляющие сигнала. С каждым усилителем, последовательно расположенным на пути следования сигнала, передаваемый сигнал становится все более искаженным. Для аналоговых данных, таких как речь, небольшое искажение несущественно, и данные остаются понятными, чего нельзя сказать о цифровых данных, в которые последовательные усилители вносят ошибки.

Цифровая передача данных зависит от содержания сигнала. Ширина полосы такого сигнала в каждом отдельном случае будет зависеть от конкретной формы сигнала и от порядка следования нулей и единиц. Из рисунка 3.37 можно видеть, что чем шире полоса сигнала, тем точнее она аппроксимирует поток цифровых импульсов. Цифровой сигнал можно передать только на ограниченные расстояния, прежде чем затухание, помехи и прочие факторы, вызывающие искажение сигнала, подвергнут опасности сохранность данных. Чтобы передать цифровые данные на большие расстояния, требуется использовать ретрансляторы, которые принимают цифровой сигнал, восстанавливают закодированную комбинацию нулей и единиц и повторно передают новый сигнал. Таким путем происходит компенсация затухания.

Очевидно, что передача сигнала, способного переносить исходную информацию на большие расстояния без искажений, с большой скоростью и с малыми материальными затратами, представляет собой сложную техническую задачу. Поскольку реальные устройства являются источниками как аналоговых по природе сигналов (телефоны, видеокамеры, звуко- и видеовоспроизводящая аппаратура), так и дискретных (компьютеры и вычислительные устройства), возникает вопрос, какой из методов передачи является предпочтительным.

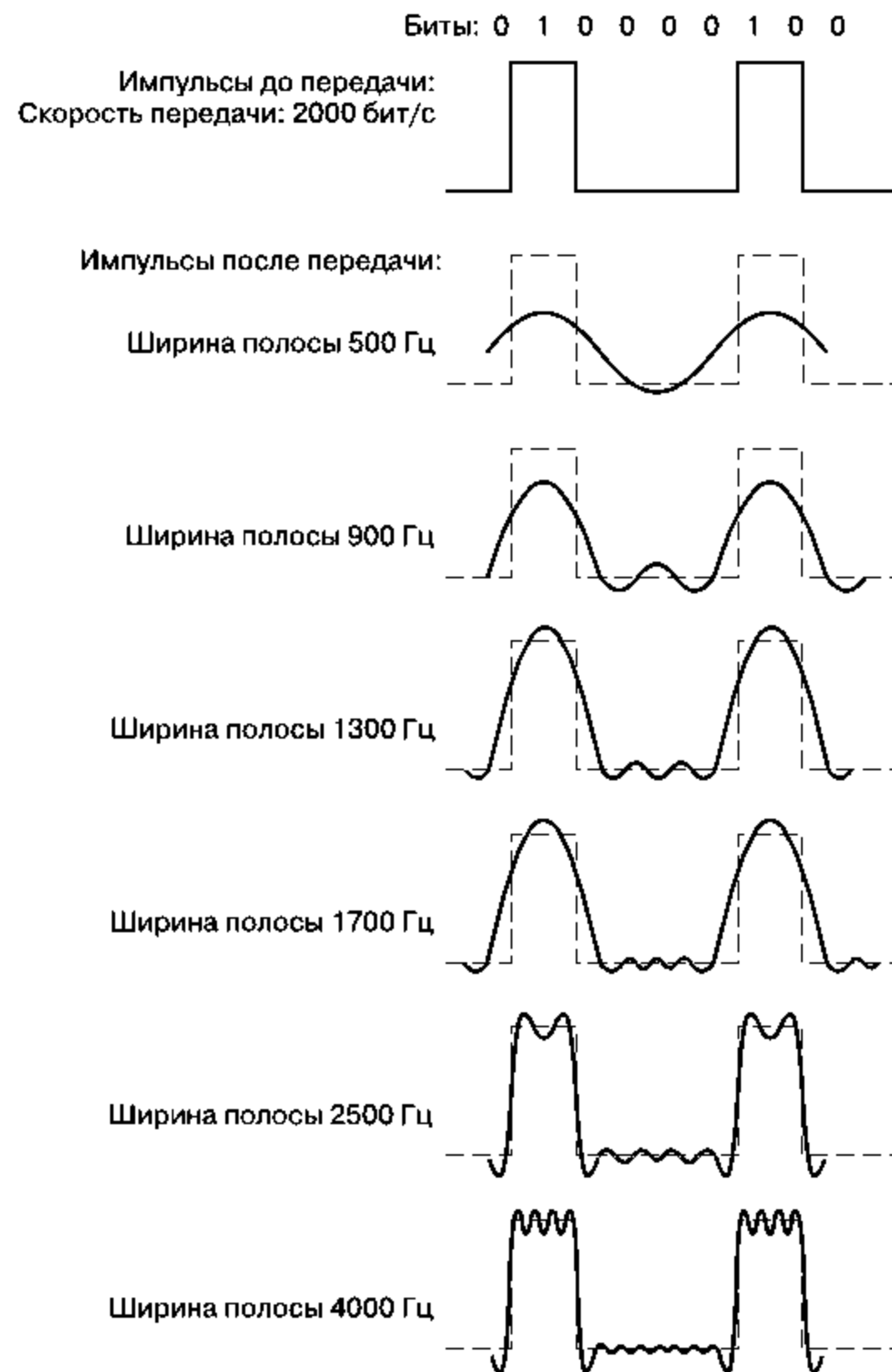


Рис. 3.37. Влияние ширины полосы на цифровой сигнал

На ранних этапах решения этой проблемы все типы данных, в том числе и компьютерные, передавались сигналами в аналоговой форме. Однако позже выяснилось, что передача данных в аналоговой форме не позволяет улучшить их качество после приема на другом конце линии, если они существенно исказились при передаче. Улучшение же качества линий требует огромных усилий и капиталовложений. Поэтому

службы, занимающиеся вопросами транспортировки данных, переходят на цифровую передачу и по возможности используют способы цифровой передачи сигналов и цифровые технологии. Это обусловлено следующими факторами:

- ◆ появление технологий больших и сверхбольших интегральных схем привело к продолжающемуся падению стоимости и уменьшению размера цифровых цепей. В оборудовании же, предназначенном для аналоговой передачи, такого явления не наблюдается;
- ◆ при использовании не усилителей, а ретрансляторов эффект от воздействия помех и других факторов, ухудшающих качество сигнала, не является кумулятивным. Поэтому становится возможным передавать данные на большие расстояния и по менее качественным линиям с помощью средств цифровой передачи, сохраняя при этом целостность данных;
- ◆ экономичнее становится создавать каналы передачи данных с очень широкими полосами, в том числе спутниковые каналы и оптоволоконные. Для эффективного использования пропускной способности таких каналов требуется высокая степень уплотнения, а этого легче и дешевле достичь с помощью цифровых (с разделением времени), а не аналоговых (с разделением частот) методов;
- ◆ к цифровым данным и оцифрованным аналоговым данным легко применяются методы кодирования данных, что повышает безопасность и конфиденциальность;
- ◆ при цифровой обработке аналоговых и цифровых данных все сигналы имеют одинаковую форму и могут обрабатываться сходным образом, что позволяет экономично объединить речевую, видео- и цифровую информацию.

3.4.4. Задачи транспортировки сигнала по линии связи

Переносчиком сообщений в системе (см. рис. 3.32, *а*) является сигнал — объект, транспортируемый через передающую систему — линию связи (ЛС) между передатчиком и приемником. *Линия связи* — физическая среда (атмосфера, космическое пространство, набор проводов и др.), по которой передаются сигналы. *Канал связи* (КС) — средства односторонней передачи данных, например полоса частот, выделенная одному передатчику при радиосвязи. Следует иметь в виду, что поня-

тия линия связи и канал связи в общем случае различаются. В одной ЛС может быть организовано несколько КС путем временного, частотного, кодового и других видов разделения. В этом случае говорят о логических (виртуальных) КС. Если канал полностью монополизирован ЛС, то он может называться *физическим каналом* и совпадать в этом случае с линией связи.

Преобразование сообщения в передаваемый сигнал $x(t)$ — некоторый материальный носитель сообщения — выполняет передатчик по строго определенному правилу (модуляция, кодирование, манипуляция), выбор которого зависит от типа сообщения и сигнала.

Доставленный в пункт приема через ЛС сигнал $x'(t)$ должен быть снова преобразован в сообщение и затем передан адресату. Однако и сигнал $x'(t)$, и принятое сообщение $a'(t)$ могут отличаться от сигнала $x(t)$ на входе ЛС и передаваемого сообщения a . Основная причина искажений передаваемого сигнала $x(t)$ — воздействие помех в линии связи, ограниченная полоса пропускания и отчасти шумы в аппаратуре передачи данных (АПД), входящей в КС.

Восстановление переданного сообщения по принятому сигналу $x'(t)$ осуществляет *приемник*. Причем эта операция возможна, если известно правило преобразования сообщения в сигнал. На основе этого правила вырабатывается правило обратного преобразования сигнала в сообщение — *демодуляция, декодирование*. Оно и позволяет приемнику выбрать сообщение из известного множества возможных сообщений. В идеальном случае выбранное сообщение a' полностью совпадает с переданным a . Однако восстановить переданное сообщение иногда и не удастся. Вследствие искажения принятого сигнала $x'(t)$ возможна ошибка при восстановлении сообщения.

Степень соответствия между переданным $x(t)$ и принятым $x'(t)$ сигналами определяет *достоверность передачи данных*.

Помехоустойчивость линии определяет ее способность уменьшать уровень помех, создаваемых внешней средой, на внутренних проводниках. Помехоустойчивость линии зависит от типа используемой физической среды, а также от экранирующих и подавляющих помехи средств самой линии. Наименее помехоустойчивыми являются радиолинии, хорошей устойчивостью обладают кабельные линии и отличной — волоконно-оптические линии, малочувствительные к внешнему электромагнитному излучению. Обычно для уменьшения помех, появляющихся из-за внешних электромагнитных полей, проводники экранируют и (или) скручивают.

Перекрестные наводки определяют помехоустойчивость кабеля к внутренним источникам помех, когда электромагнитное поле сигнала, передаваемого с выхода передатчика по одной паре проводников, наводит на другую пару проводников сигнал помехи. Если ко второй паре будет подключен приемник, то он может принять внутреннюю помеху за полезный сигнал.

Задача транспортировки сигнала, имеющего определенные характеристики, требует согласования их с характеристиками канала связи. Основными характеристиками сигнала являются:

- ♦ *длительность* T_c ;
- ♦ *максимальная частота спектра* F_c ; например, максимальная частота звукового сигнала для человека $F_{c \max} = 20$ кГц. Чем выше частота несущего периодического сигнала, тем больше информации в единицу времени передается по линии и тем выше пропускная способность линии при фиксированном способе физического кодирования. Однако при повышении частоты периодического несущего сигнала увеличивается и ширина спектра этого сигнала, т.е. разность между максимальной и минимальной частотами набора синусоид, которые в сумме дадут выбранную для физического кодирования последовательность сигналов. Линия связи передает этот спектр синусоид с теми искажениями, которые определяются ее полосой пропускания. Чем больше несоответствие между полосой пропускания линии и шириной спектра передаваемых информационных сигналов, тем сильнее сигналы искажаются и тем вероятнее ошибки в распознавании информации принимающей стороной, а следовательно, скорость передачи информации на самом деле оказывается меньше, чем можно было предположить;
- ♦ *мощность* S_c , под которой понимают допустимое превышение средней мощности передаваемого сигнала P_c над средней мощностью помехи P_n :

$$S_c = \lg \frac{P_c}{P_n}.$$

Произведение этих трех характеристик сигнала принято называть *объемом сигнала*:

$$V_c = T_c F_c S_c.$$

Аналогичные характеристики вводятся и для канала связи (КС):

- ♦ $T_{КС}$ — время, в течение которого канал предоставляется для передачи сигнала;
- ♦ $F_{КС}$ — полоса пропускания КС (определяет диапазон частот синусоидального сигнала);
- ♦ $S_{КС}$ — превышение мощности полученного сигнала P_s над мощностью уровня шумов $P_{ш}$ в канале связи, при которых этот сигнал передается по КС без значительных искажений: отношение амплитуды выходного сигнала к амплитуде входного превышает некоторый заранее заданный предел, обычно 0,5:

$$S_{КС} = 10 \lg \frac{P_s}{P_{ш}}.$$

Объем канала связи

$$V_{КС} = T_{КС} F_{КС} S_{КС}.$$

Для передачи сигнала без искажения по заданному КС необходимо выполнить условие $V_c < V_{КС}$, т.е. объем передаваемого сигнала должен быть меньше или равен объему КС.

При этом необходимо, чтобы:

- а) $T_c \leq T_{КС}$;
- б) $F_c \leq F_{КС}$, т.е. полоса пропускания канала должна быть больше, чем ширина спектра передаваемого сигнала;
- в) $S_c \leq S_{КС}$.

Если объемы V_c и $V_{КС}$ представить в виде кубов, то их геометрическое соотношение будет иметь вид, показанный на рис. 3.38.

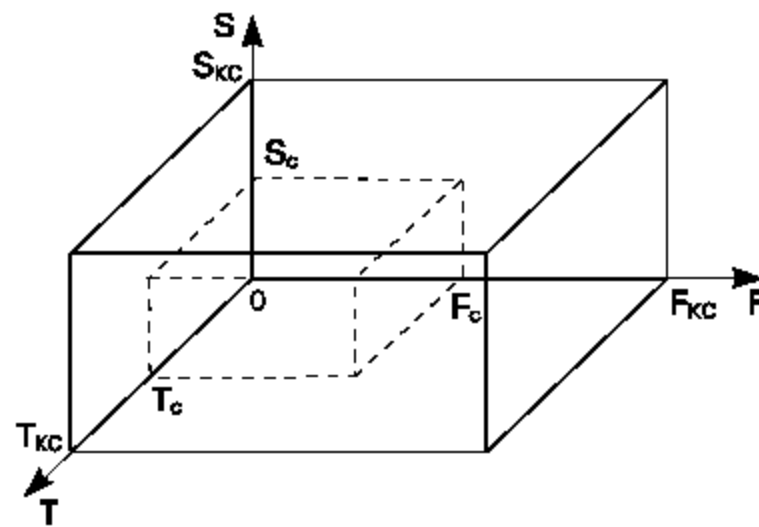


Рис. 3.38. Соотношение объемов сигнала и канала связи

Характеристики сигнала можно менять, сохраняя объем сигнала. Такая операция называется *деформацией объема сигнала*.

3.4.5. Типы линий связи

Передающие среды и линии связи, используемые для передачи сигнала, требуют отдельного рассмотрения, ибо являются важным фактором, определяющим скорость передачи данных и расстояние, на которое эти данные передаются. Кроме того, от ЛС зависит способ кодирования данных.

Линия связи (ЛС) — физическая среда (атмосфера, космическое пространство, набор проводов и др.), по которой передаются сигналы.

Канал связи (КС) — средства односторонней передачи данных, например полоса частот, выделенная одному передатчику при радиосвязи.

Одна и та же ЛС может служить одновременно для реализации одного или нескольких КС (многоканальная связь), т.е. в одной ЛС может быть организовано несколько КС путем временного, частотного, кодового и других видов разделения, по каждому из которых передается своя информация.

При временном мультиплексировании или разделении по времени каждому каналу выделяется некоторый квант времени, при частотном — некоторая полоса частот.

По каждому из каналов передается своя информация, и тогда говорят о логических (виртуальных) каналах связи. Если канал полностью монополизирован ЛС, то он называется физическим каналом и в этом случае совпадает с ЛС.

В системе передачи данных передающая среда играет роль физического пути между передатчиком и приемником. В подразделе 3.4.1 говорилось, что передающие среды делятся на *проводные*, представляющие физический канал связи, по которому и распространяется сигнал (витая пара, коаксиальный кабель, оптоволокно), и *беспроводные*, в которых передача производится через атмосферу, открытый космос или воду.

В зависимости от среды передачи данных различают линии связи:

- ◆ проводные (воздушные);
- ◆ кабельные (медные и волоконно-оптические);
- ◆ радиоканалы наземной и спутниковой связи;
- ◆ инфракрасные лучи.

Проводные ЛС представляют собой провода без каких-либо изолирующих или экранирующих оплеток, подвешенные на столбах и закрепленные с помощью изоляторов и специальной арматуры. По таким ЛС традиционно передают телефонные и телеграфные сигналы, однако их можно использовать и для передачи данных между отстоящими на расстоянии друг от друга компьютерами. Скорость передачи и помехоза-

щищенность таких ЛС являются низкими, поэтому они быстро вытесняются кабельными ЛС.

Кабельные линии связи представляют собой достаточно сложную конструкцию. *Кабелем* называют изделие, содержащее проводники с электрической, электромагнитной, механической и, возможно, климатической изоляцией. В компьютерных сетях применяют три основных типа кабеля:

- а) экранированные и неэкранированные скрученные пары медных проводов (*витая пара*). Скручивание проводов снижает влияние внешних помех на полезные сигналы, передаваемые по кабелю. Витая пара характеризуется отличным соотношением качества к стоимости и простотой монтажа;
- б) *коаксиальный кабель*: в проводник, имеющий форму полого цилиндра (внешний) вложен другой, изолированный от него проводник (внутренний);
- в) *волоконно-оптический кабель* состоит из волокон диаметром в несколько микрон, окруженных твердым покрытием и помещенных в защитную оболочку. Для передачи информации по оптоволоконному кабелю используется свет. Первые оптоволоконные кабели изготавливались из стекла, в настоящее время уже разработаны кабели на основе пластиковых волокон. Источником распространяемого по оптическим кабелям света являются светодиод или инжекционный лазерный диод, а кодирование информации осуществляется путем изменения интенсивности света. На принимающем конце кабеля детектор преобразует световые сигналы в электрические. Светодиод дешевле, функционирует в большом температурном диапазоне и служит дольше. Инжекционный диод, работающий по принципу лазера, более эффективен и может поддерживать более высокие скорости передачи данных. Оптоволоконный кабель является наиболее помехозащищенным, позволяет передавать информацию на большие расстояния и с большой скоростью (до 10 Гбит/с и выше), однако этот кабель дорогой, сложный в установке и обслуживании.

3.4.6. Беспроводная связь

Радиоканалы наземной и спутниковой связи образуются с помощью передатчика и приемника радиоволн. Существует много типов радиоканалов, различающихся как используемым частотным диапазоном, так и дальностью канала.

В *радиолинии* физической средой распространения высокочастотных электромагнитных колебаний является атмосфера. Применяется отражение сигнала ионосферой Земли.

В *радиорелейных линиях* физической средой также выступает атмосфера, но выполняются условия прямой видимости между передатчиком и приемником. Сигнал передается от одной ретрансляционной станции к другой, расположенной на определенном расстоянии. Эти станции восстанавливают характеристики сигнала, противодействуя потере информации.

В *спутниковых системах связи*, предназначенных для связи, телевидения, звукового вещания, передачи газет, данных и т.д., в качестве ретранслятора используются космические аппараты (спутники), которые располагаются на разных орбитах и имеют большую площадь обзора. Доступ к спутниковому ретранслятору обеспечивается методами временного и частотного разделения каналов. Радиотелефонная спутниковая связь использует цифровую передачу сообщений и не имеет ограничения по зоне обслуживания.

Инфракрасные беспроводные сети используют для передачи данных инфракрасные лучи.

Существуют, по сути, два типа конфигурации беспроводной передачи: *направленная* и *ненаправленная*. Прием и передача сигнала производятся с помощью антенны. При передаче антенна излучает электромагнитную энергию в среду (как правило, в воздух), а при приеме — улавливает электромагнитные волны из окружающей среды. В *направленной конфигурации* передающая антенна излучает сфокусированный электромагнитный луч, и, следовательно, передающая и принимающая антенны должны быть тщательно нацелены. В *ненаправленной* конфигурации передаваемый сигнал распространяется во всех направлениях и может приниматься большим числом антенн. В общем случае чем выше частота сигнала, тем легче его сфокусировать в направленный луч.

При рассмотрении беспроводной передачи данных интерес представляют три диапазона частот (направленная конфигурация):

- 1) 30 МГц — 1 ГГц — часть диапазона ультравысоких частот (УВЧ) (используется в некоторых приложениях сетей передачи данных и включает FM-радио, а также телевидение метрового и УВЧ-диапазонов);
- 2) 4—40 ГГц — диапазон *сверхвысоких частот* (СВЧ). В диапазоне СВЧ (свыше 4 ГГц) сигналы уже не отражаются ионосферой Земли, и для устойчивой связи требуется прямая видимость между передатчиком и приемником. Поэтому та-

кие частоты используют либо спутниковые (рис. 3.39), либо радиорелейные каналы, где это условие выполняется. Чем выше используемая частота, тем шире возможная полоса и тем выше потенциальная скорость передачи данных (табл. 3.2).

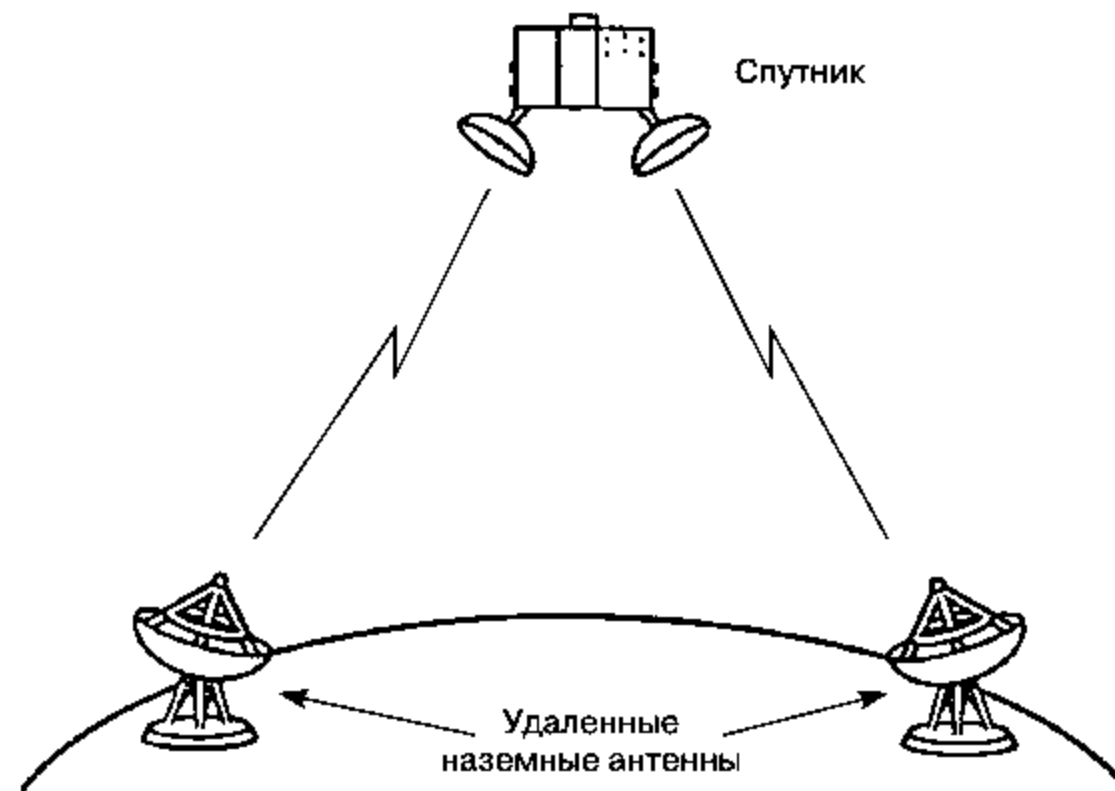


Рис. 3.39. Двухточечный канал, созданный с помощью спутниковой СВЧ-связи

Таблица 3.2

Производительность типичных цифровых СВЧ-систем

Диапазон, ГГц	Ширина полосы, МГц	Скорость передачи данных, Мбит/с
2	7	12
6	30	90
11	40	135
18	220	274

При передаче на одинаковые расстояния средства СВЧ-связи требуют значительно меньшего, чем коаксиальный кабель, числа усилителей или ретрансляторов. Они обычно используются для передачи как речи, так и телевизионного изображения.

Другой областью, в которой широко применяется СВЧ-связь, является создание коротких двухточечных каналов связи между зданиями. Как правило, с этой целью используется полоса 22 ГГц;

- 3) $3 \cdot 10^{11}—2 \cdot 10^{14}$ Гц) — *инфракрасный диапазон* (ИКД). В инфракрасных беспроводных сетях для передачи данных применяются инфракрасные лучи. Генерируемый источником сигнал в таких системах должен быть очень мощным, так как в противном случае значительное влияние будут оказывать другие источники. ИКД может быть использован в локальных двухточечных и многоточечных приложениях в ограниченных пространствах, например в отдельной комнате.

Основным источником потерь в *системах СВЧ-связи*, как и в любых других передающих системах, является затухание. В отличие от *витой пары* и *коаксиального кабеля* потери зависят от расстояния логарифмически. Следовательно, при использовании систем СВЧ-связи ретрансляторы и усилители могут размещаться реже, среднее расстояние между ними обычно равно 10—100 км.

Передача через спутник также применима для создания двухточечных магистралей между телефонными станциями и общественными телефонными сетями. Эта среда соперничает с наземными системами во многих протяженных каналах связи.

Для *спутниковой передачи* оптимален диапазон частот от 1 до 10 ГГц. На частотах меньше 1 ГГц возникают сильные помехи от естественных источников, в том числе помехи космического происхождения, солнечные, атмосферные, а также помехи от устройств, созданных человеком. На частотах выше 10 ГГц сигнал сильно поглощается атмосферой и атмосферными осадками.

Следует отметить, что поскольку спутниковая связь применяется на больших расстояниях, то при передаче от одной наземной станции к другой существует задержка распространения, равная приблизительно четверти секунды, которая заметна даже в обычных телефонных разговорах, а также порождает проблемы в области защиты от ошибок и управления потоком данных.

Одним из важных различий связи в инфракрасном и СВЧ-диапазонах является то, что лучи, используемые в инфракрасном диапазоне, не проникают сквозь стены, следовательно, проблемы конфиденциальности, возникающие в СВЧ-системах, в данном случае отсутствуют.

3.4.7. Проводная связь

Из проводных сред наиболее широко для передачи данных используются *витая пара*, *коаксиальный кабель* и *оптоволокно*.

Витая пара — изолированные медные проводники, попарно свитые между собой минимально необходимое число раз на определенном

отрезке длины. Витая пара является самой распространенной средой и может использоваться как для аналоговой, так и для цифровой передачи. Широко применяется в телефонной сети и незаменима при организации связи в пределах зданий, а также в локальных сетях, поддерживающих персональные компьютеры. Скорости передачи данных в таких случаях составляют порядка 10 Мбит/с. При аналоговой передаче на каждые 5—6 км линии требуется усилитель, при цифровой же передаче (аналоговых и цифровых сигналов) через каждые 2—3 км линии должен устанавливаться ретранслятор.

При двухточечной *аналоговой* передаче сигналов возможная ширина полосы доходит до 1 МГц. В такую полосу помешается несколько речевых каналов. При дальней двухточечной *цифровой* передаче сигналов скорость передачи данных может достигать нескольких Мбит/с, при передаче на небольшие расстояния скорость доходит до 100 Мбит/с и даже 1 Гбит/с.

По сравнению с другими проводными передающими средами (коаксиальный кабель, оптоволокно) *витая пара* ограничена в расстоянии, полосе и скорости передачи данных. Однако она дешевле названных сред и является наиболее простой в использовании.

В приложениях дальней связи витые пары могут применяться со скоростями передачи данных 4 Мбит/с и выше.

Коаксиальный кабель вследствие экранирования и концентрической структуры менее чем витая пара подвержен воздействию интерференции и перекрестных помех и позволяет использовать более широкий диапазон частот. Применяется для передачи как аналоговых, так и цифровых сигналов. Поскольку его частотные характеристики лучше характеристик витой пары, эта передающая среда может эффективнее использоваться на более высоких частотах и при больших скоростях передачи данных.

Коаксиальный кабель может применяться для организации связи на больших расстояниях и поддерживать большее число станций в одном канале связи, чем витая пара.

Для передачи *аналоговых сигналов* на значительные расстояния на каждые несколько километров требуются усилители, причем при использовании высоких частот они должны размещаться чаще. Спектр, используемый при аналоговой передаче данных, простирается до 500 МГц. При *цифровой* передаче сигналов на каждые несколько километров требуются ретрансляторы, которые при использовании высоких скоростей передачи данных должны располагаться чаще.

Коаксиальный кабель традиционно является важной частью междугородных телефонных сетей, также широко применяется для органи-

зации ближних соединений между устройствами. Цифровая передача сигналов позволяет использовать коаксиальный кабель для создания в вычислительных системах высокоскоростных каналов ввода-вывода. В наши дни эта система все сильнее конкурирует с наземной передачей по СВЧ и спутниковой связью.

Волоконно-оптический кабель — наиболее качественный тип кабеля. Он обеспечивает передачу данных с очень высокой скоростью (до 10 Гбит/с и выше) и, кроме того, лучше других типов передающей среды защищает данные от внешних помех. Основными отличиями оптоволоконна от витой пары и коаксиального кабеля являются:

- ♦ *большая пропускная способность.* Потенциальная ширина полосы оптоволоконна, а следовательно, и скорость передачи данных, огромна. При передаче данных на *десятки километров* скорость составляет порядка сотен гигабит в секунду, что *несравнимо* с максимумом в сотни мегабит в секунду при передаче по коаксиальному кабелю длиной 1 км и всего лишь несколькими мегабитами в секунду при передаче по витой паре длиной 1 км. При передаче по витой паре длиной *несколько десятков метров* скорость передачи составляет от нескольких сотен Мбит/с до 1 Гбит/с;
- ♦ *малый размер* (на порядок тоньше коаксиального кабеля и витой пары) и *небольшая масса*;
- ♦ *более низкое*, чем в коаксиальном кабеле или витой паре, *затухание*;
- ♦ *электромагнитная изоляция.* Система неуязвима для интерференции, импульсных или перекрестных помех, так как внешние электромагнитные поля на оптоволоконные системы не воздействуют;
- ♦ *высокая степень защиты от прослушивания.* Кроме того, к волокнам довольно трудно подсоединиться;
- ♦ *большее расстояние между ретрансляторами.* Расстояние между ретрансляторами в десятки километров является обычным, имеются также линии с расстояниями в сотни километров между ретрансляторами, в то время как в системах, использующих коаксиальные кабели и витые пары, ретрансляторы, как правило, расположены через каждые несколько километров. Меньшее число ретрансляторов означает меньшее число источников ошибок.

Повышенным спросом начинают пользоваться подводные оптоволоконные кабели.

Оптоволокно находит широкое применение в локальных сетях с пропускной способностью от 100 Мбит/с до 1 Гбит/с, которые поддерживают сотни или даже тысячи станций в больших административных зданиях и в комплексах зданий.

3.5. МОДУЛЯЦИЯ. ОБЩАЯ МОДЕЛЬ ПРОЦЕССА

В рассмотренных выше способах цифровой и аналоговой передачи данных по физическим каналам связи используют как аналоговую обработку цифровых данных — *модуляцию*, так и цифровую обработку аналоговых данных — *цифровое кодирование*. Эти способы обработки отличаются шириной спектра результирующего сигнала и сложностью аппаратуры, необходимой для их реализации. При цифровом кодировании спектр результирующего сигнала гораздо шире, чем при использовании модуляции при той же скорости передачи информации, однако для реализации синусоидальной модуляции необходима более сложная и дорогая аппаратура, чем для реализации цифрового кодирования прямоугольными импульсами.

При наличии только средств аналоговой передачи *модуляция* необходима для преобразования цифровых данных в аналоговую форму и причина этого очевидна. Этого нельзя сказать об аналоговых данных, ибо все речевые сигналы передаются по телефонным линиям с сохранением исходного спектра (*немодулированная передача*). Причинами модуляции аналоговых сигналов могут быть следующие:

- ♦ для эффективной передачи могут потребоваться более высокие частоты. При беспроводной передаче передавать немодулированные сигналы практически невозможно (потребовались бы километровые антенны);
- ♦ модуляция допускает частотное уплотнение, обеспечивающее эффективное использование высокоскоростных линий связи. (Несколько аналоговых сигналов, каждому из которых выделяется определенная полоса частот, одновременно распространяются через одну и ту же среду. Для смещения сигнала в требуемую полосу частот необходимы средства модуляции, а для объединения полученных сигналов разной частоты нужны средства уплотнения.)

При передаче данных по аналоговому КС используют определенный физический процесс, называемый *сигналом-переносчиком* или *несущим сигналом*. Это периодический сигнал, параметры которого изме-

няются по закону передаваемого сообщения. Если в качестве такого сигнала применяется синусоида, то он называется *несущей частотой*.

Таким образом, *модуляцией* называют процесс изменения параметров переносчика (несущей) в соответствии с передаваемым сообщением.

Этот процесс можно определить как объединение входного сигнала $x(t)$ и *несущего* сигнала с частотой f_n с целью создания сигнала, полоса которого (как правило) центрирована на частоте f_n (рис. 3.40).

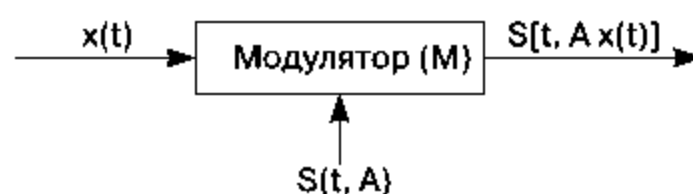


Рис. 3.40. Представление процесса модуляции:

$x(t)$ — передаваемое сообщение; $S(t, A)$ — несущий сигнал;
 A — информативный (изменяемый) параметр;
 $S[t, A x(t)]$ — модулированный сигнал

В зависимости от изменяемого параметра A различают три основных вида модуляции:

- 1) амплитудную;
- 2) частотную;
- 3) фазовую.

Обратное преобразование электромагнитных колебаний в первоначальный сигнал, осуществляемое на приемной стороне, называется *демодуляцией* (соответственно амплитудной, частотной и фазовой).

3.5.1. Амплитудная модуляция

Амплитудная модуляция (АМ) представляет собой простейшую форму модуляции (см. рис. 3.22). Модулирующая функция $x(t)$ изменяет амплитуду высокочастотной гармонической функции сигнала-переносчика $U(t)$. Математически данный процесс описывается выражением (3.15) или

$$U_{AM}(t) = [1 + mx(t)] \sin 2\pi f_n t,$$

где $\sin 2\pi f_n t$ — несущая;

$x(t)$ — входной сигнал (содержащий информацию), оба приведены к единичной амплитуде, $\varphi_0 = 0$;

m — коэффициент модуляции, представляющий собой отношение амплитуд входного и несущего сигналов.

Если амплитуда несущей модулируется сигналом $x(t) = \cos 2\pi Ft$, выражение результирующего амплитудно-модулированного сигнала примет вид

$$U_{AM}(t) = [1 + m \cos 2\pi Ft] \sin 2\pi f_n t. \quad (3.20)$$

Используя тригонометрические формулы (произведение $\sin \alpha \times \cos \beta$), получаем

$$U_{AM}(t) = \sin 2\pi f_n t + \frac{m}{2} \sin 2\pi(f_n - F)t + \frac{m}{2} \sin 2\pi(f_n + F)t. \quad (3.21)$$

Из выражения (3.21) следует, что результирующий сигнал состоит из гармоник исходной несущей частоты и двух гармоник, смещенных относительно несущей на F Гц. Из формулы (3.21) и рис. 3.22 видно, что при амплитудной модуляции происходит умножение входного сигнала на несущий сигнал. Огибающая результирующего сигнала равна $[1 + mx(t)]$ и, поскольку $m < 1$, огибающая точно воспроизводит исходный сигнал. При $m > 1$ огибающая пересечет временную ось, и информация будет потеряна.

Спектр амплитудно-модулированного сигнала (см. рис. 3.23) состоит из спектра несущего сигнала f_n и спектра входного сигнала $x(t)$, сдвинутого вверх и вниз относительно f_n , где $(f_n - F)$ — *нижняя*, а $(f_n + F)$ — *верхняя боковые частоты*. Из выражения (3.21) видно, что вся информация о модулирующей функции полностью содержится в любой из боковых частот.

Система с АМ, которая передает обе боковые и несущую частоту, известна как *двухполосная система* (DSB — double sideband) с передачей несущей. Несущая не содержит никакой полезной информации и может быть удалена, но с несущей или без нее полоса сигнала DSB вдвое больше полосы входного сигнала. Для сужения рабочей полосы частот канала связи возможно вытеснение не только несущей, но и одной из боковых, так как боковые частоты несут одну и ту же информацию. Этот факт используется в *однopolосных системах* связи, когда передается только одна из боковых полос — *однopolосная АМ с подавленной несущей*. Ширина спектра в этом случае равна F , а не $2F$, как в случае передачи несущей с двойной боковой полосой. Кроме того, для передачи сигнала требуется меньшая мощность, поскольку она уже не расходуется на передачу несущей и одной из боковых полос. Этот вид модуляции создает новый сигнал, идентичный оригиналу, но сдвинутый вверх по частоте. Частоту несущей выбирают в соответствии с условиями среды передачи.

3.5.2. Частотная модуляция

В системах частотной модуляции (ЧМ) информативным параметром A является частота f_n несущего сигнала:

$$U(t) = U_n \sin(\omega_n t + \varphi_0) = U_n \sin(2\pi f_n t + \varphi_0).$$

Если модулирующая функция $x(t) = \cos \Omega t = \cos 2\pi F t$, то

$$\omega = \omega_n [1 + m_f x(t)] = \omega_n [1 + (\Delta\omega / \omega_n) \cos \Omega t],$$

где $m_f = \Delta\omega / \omega_n$ — коэффициент частотной модуляции (относительное изменение частоты);

$\Delta\omega$ — девиация частоты.

Тогда сигнал-переносчик

$$\begin{aligned} U_{\text{ЧМ}} &= U_n \left(\sin \int_0^t \omega dt \right) = \\ &= U_n (\sin \omega_n t \cos \beta \sin \Omega t) + \cos \omega_n t \sin(\beta \sin \Omega t), \end{aligned} \quad (3.22)$$

где $\beta = \frac{\Delta\omega}{\Omega}$ — индекс модуляции.

Частотная модуляция *помехоустойчива*, поскольку искажению при помехах подвергается в основном амплитуда сигнала, а не частота. Необходимая для этого вида модуляции ширина спектра сигнала может быть значительно меньше всей полосы пропускания канала.

3.5.3. Фазовая модуляция

При фазовой модуляции (ФМ) информативным параметром A сигнала-переносчика служит фаза Φ несущей частоты ω_n :

$$U_{\text{ФМ}}(t) = U_n \sin [\omega_n t + \Phi + \Delta\varphi x(t)].$$

Если модулирующая функция $x(t) = \cos \Omega t$, то с учетом, что $\sin(\alpha + \beta) = \sin \alpha \cos \beta + \cos \alpha \sin \beta$, ФМ-сигнал будет описываться выражением

$$\begin{aligned} U_{\text{ФМ}}(t) &= U_n [\sin (\omega_n t + \Phi) \cos (\Delta\varphi \cos \Omega t) + \\ &+ \cos (\omega_n t + \Phi) \sin (\Delta\varphi \cos \Omega t)]. \end{aligned} \quad (3.23)$$

Сравнивая выражения (3.22) и (3.23), можно видеть, что ЧМ- и ФМ-сигналы похожи по форме. Различие заключается в том, что коэффициент модуляции для ФМ-сигнала постоянен, а индекс модуляции β для ЧМ-сигнала зависит от частоты модулирующего сигнала Ω .

Так же как и АМ, частотная и фазовая модуляции порождают сигнал с полосой, центрированной на частоте f_n , однако размеры этих полос существенно различаются. Частотная и фазовая модуляции требуют большей полосы, чем амплитудная.

3.5.4. Цифровое кодирование (передача цифровых данных с помощью аналоговых сигналов)

Типовым примером такой передачи является передача цифровых данных через телефонную сеть общего пользования. Цифровые устройства присоединяются к сети с помощью *модема* (модулятора-демодулятора), преобразующего цифровые данные в аналоговые сигналы, и наоборот.

Для преобразования цифровых данных в аналоговый сигнал используются три технологии кодирования или модуляции (применительно к цифровым сигналам — *манипуляции*) (рис. 3.41):

- 1) амплитудная манипуляция;
- 2) частотная манипуляция;
- 3) фазовая манипуляция.

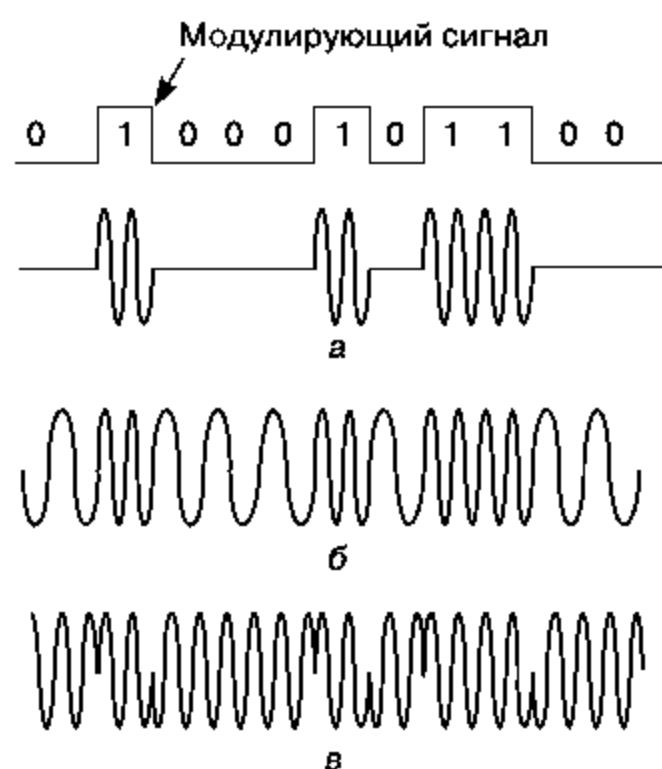


Рис. 3.41. Модуляция цифровых данных аналоговыми сигналами:

а — амплитудная манипуляция; б — частотная манипуляция;
в — фазовая манипуляция

При *амплитудной манипуляции* два двоичных значения представляются сигналами несущей частоты с двумя различными амплитуда-

ми. Одна из амплитуд, как правило, выбирается равной нулю; т.е. одно двоичное число представляется присутствием несущей частоты при постоянной амплитуде, а другое — ее отсутствием. Результирующий сигнал равен

$$U(t) = \begin{cases} \sin(2\pi f_n t) & \text{— двоичная 1,} \\ 0 & \text{— двоичный 0,} \end{cases}$$

где $\sin(2\pi f_n t)$ — несущий сигнал.

Метод амплитудной манипуляции чувствителен к внезапным скачкам и неэффективен. В телефонных линиях он обычно используется только при скоростях до 1200 бит/с.

Метод амплитудной манипуляции применяется для передачи цифровых данных по оптоволокну. Для передатчиков на светодиодах справедливо приведенное выше равенство. Иными словами, одна сигнальная посылка представляется световым импульсом, а другая — отсутствием света. Амплитудная манипуляция используется также в высокочастотных (от 3 до 30 МГц) радиопередачах, а в локальных сетях, применяющих коаксиальный кабель, ее можно использовать даже на более высоких частотах.

При *частотной манипуляции* два двоичных числа представляются сигналами двух различных частот, расположенных по обе стороны от несущей. Результирующий сигнал равен

$$U(t) = \begin{cases} U_n \cos(2\pi f_1 t) & \text{— двоичная 1,} \\ U_n \cos(2\pi f_2 t) & \text{— двоичный 0,} \end{cases}$$

где f_1 и f_2 — частоты, смещенные от несущей частоты f_n на величины, равные по модулю, но противоположные по знаку.

Частотная манипуляция менее восприимчива к ошибкам, чем амплитудная манипуляция. В телефонных линиях она также используется при скоростях до 1200 бит/с.

При *фазовой манипуляции* для представления данных выполняется смещение несущего сигнала. На рисунке 3.41, в приведен пример двухфазной системы. В данной системе двоичный 0 представляется сигнальным пакетом, фаза которого совпадает с фазой предыдущего посланного пакета, а двоичная 1 представляется сигнальным пакетом с фазой, противоположной фазе предыдущего пакета. Такая схема называется *диф-*

ференциальной фазовой манипуляцией, поскольку сдвиг фаз выполняется относительно предыдущего переданного бита, а не относительно какого-то эталонного сигнала. Результирующий сигнал равен

$$U(t) = \begin{cases} U_n \sin(2\pi f_n t + \pi) & \text{— двоичная 1,} \\ A \sin(2\pi f_n t) & \text{— двоичный 0,} \end{cases}$$

где фаза отсчитывается от предыдущего интервала передачи бита.

3.6. КОДИРОВАНИЕ ИНФОРМАЦИИ

Под *кодированием* информации будем понимать преобразование формы представления информации с целью обеспечения удобства ее передачи по каналам связи или хранения. Простым примером является передача сообщения в виде телеграммы, когда все символы кодируются с помощью телеграфного кода.

Декодирование информации — операция восстановления принятого сообщения.

Для передачи сообщения по КС необходимо иметь устройства для кодирования (кодеры) и для декодирования (декодеры) сообщений.

В широком смысле под *кодированием* понимают представление символов одного алфавита символами другого. Например, десятичное число 92 (символы 9 и 2) можно представить в символах двоичного алфавита:

$$92 = 0101\ 1100_2$$

или двоично-десятичного (BCD-кода), в котором каждая десятичная цифра представляется двоичной тетрадой:

$$92 = \underbrace{1001}_9 \underbrace{0010}_2.$$

Будем рассматривать *кодирование* как представление дискретных сообщений в виде некоторых комбинаций, составленных по определенным правилам из определенного числа символов — *элементов кода*. Число различных элементов, из которых слагаются комбинации, назовем *основанием кода* q . Так, элементами двоичного кода являются символы 0 и 1 ($q = 2$). В этом случае дискретные сообщения представляются в виде последовательности двоичных символов 0 и 1, например 101011_2 . При двоичном кодировании число кодовых комбинаций

$m = 2^n$, где n — число символов $x_i \in \{0, 1\}$ в кодовой комбинации (КК).

Так, при $n = 3$ $m = 2^3 = 8$:

$$\left. \begin{array}{ccc} x_1 & x_2 & x_3 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ \cdot & & \\ \cdot & & \\ 1 & 1 & 1 \end{array} \right\} m = 8.$$

В системах передачи информации в основном используют бинарные коды, т.е. коды с основанием $q = 2$. Учитывая это, главное внимание в дальнейшем будем уделять бинарным кодам.

Основные требования к кодированию:

- ♦ *среднее число символов, приходящихся на единицу сообщения, должно быть минимальным.* При отсутствии помех это требование приводит к выигрышу во времени при передаче сообщения или к сокращению объема памяти запоминающего устройства при хранении. Такое кодирование называется *безызбыточным* или *оптимальным*;
- ♦ *кодирование должно обеспечивать заданную достоверность* при передаче или хранении информации. Однако для этого требуется больше символов для представления сообщения, поэтому такое кодирование называется *избыточным* или *помехоустойчивым*. Помехоустойчивость кодов достигается введением избыточности в кодовые комбинации (вводят дополнительные разряды). Если, например, для кодирования всех символов внешнего алфавита достаточно иметь m -разрядный первичный код, то для обеспечения помехоустойчивости к его m разрядам добавляют k избыточных разрядов. При этом длина результирующей кодовой комбинации (КК) становится равной $n = m + k$. Коды, не обладающие избыточностью, не способны обнаруживать и тем более исправлять ошибки.

Таким образом, от способа кодирования сообщения зависят и время его передачи, и достоверность. Одновременное выполнение перечисленных требований невозможно, так как они являются противоречивыми. Для разрешения противоречия в системах передачи дискретных сообщений используют два алфавита: один алфавит достаточно боль-

щого объема применяется для представления сообщения на языке источника и получателя информации и называется *внешним алфавитом*; второй используют непосредственно для передачи информации по каналу, он содержит небольшое число символов и называется *внутренним алфавитом*. Чем меньше символов содержит внутренний алфавит, тем легче их различать в условиях помех. Проблемы помехоустойчивого кодирования решаются специальными методами.

3.6.1. Типы кодов

Все современные коды (рис. 3.42) делятся на *безызбыточные* (первичные) и *избыточные* (корректирующие).

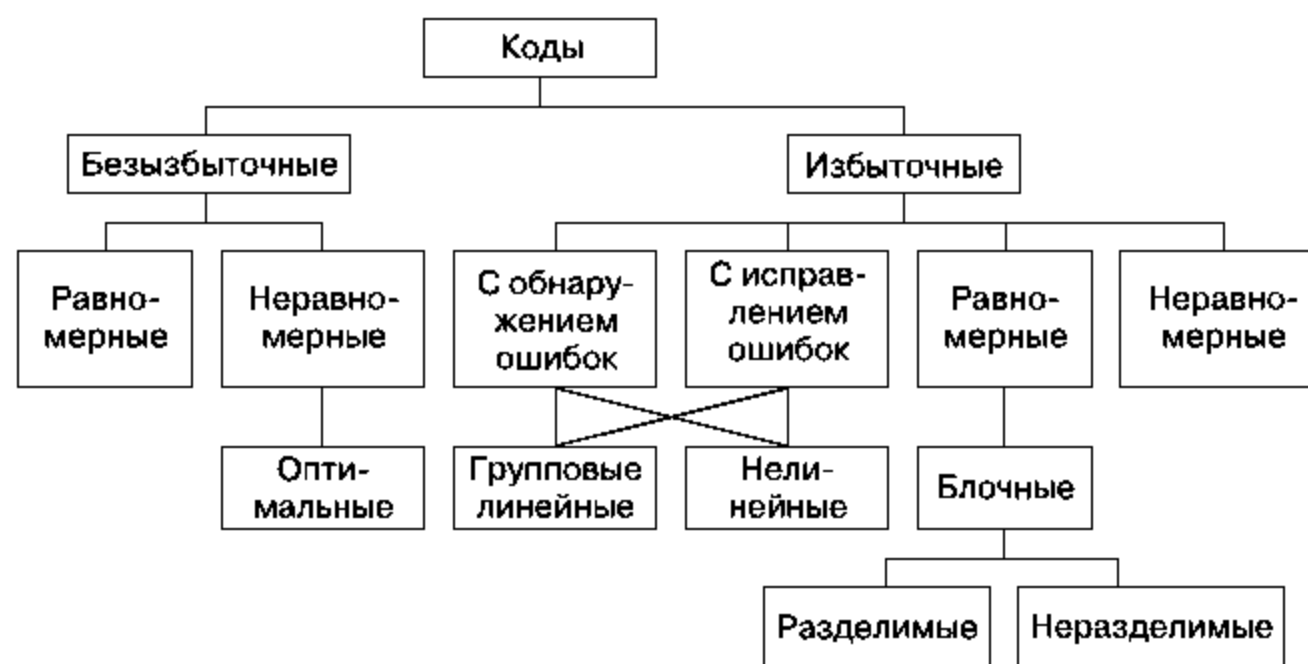


Рис. 3.42. Классификация кодов

Коды, формирующие кодовые комбинации различной длины, называются *неравномерными*, а коды, которым соответствуют кодовые комбинации равной длины, — *равномерными*. В *блочных* кодах кодовые комбинации кодируются и декодируются независимо друг от друга. Избыточные блочные коды могут быть *разделимыми* и *неразделимыми*. В *разделимых* кодах роль разрядов кодовых комбинаций разграничена: часть разрядов, часто совпадающая с разрядами исходного первичного кода, является информационной, а остальные играют роль проверочных (контрольных) разрядов. В *неразделимых* кодах все разряды равноправные, и в кодовой комбинации нельзя отделить информационные разряды от проверочных. Примером блочного разделимого кода служит код с проверкой на четность. Его кодовая комбинация имеет вид $x_1 x_2 \dots x_k y$. Первые k разрядов являются информационными и, как правило, сов-

падают с разрядами исходного первичного кода. Последний разряд является избыточным и определяется по формуле $y = x_1 \oplus x_2 \oplus \dots \oplus x_k$. Таким образом, его значение зависит от того, четное или нечетное число единиц в кодовой комбинации. Если четное, то $y = 0$, в противном случае $y = 1$.

3.6.2. Кодирование как средство контроля передаваемой информации

При передаче кодированных сообщений между устройствами одной ЭВМ транспортировка кодов осуществляется по короткому пути. При передаче кодированных сообщений от одной ЭВМ к другой этот путь может оказаться длинным. При прохождении сообщения по каналу связи на него воздействуют помехи (рис. 3.43).

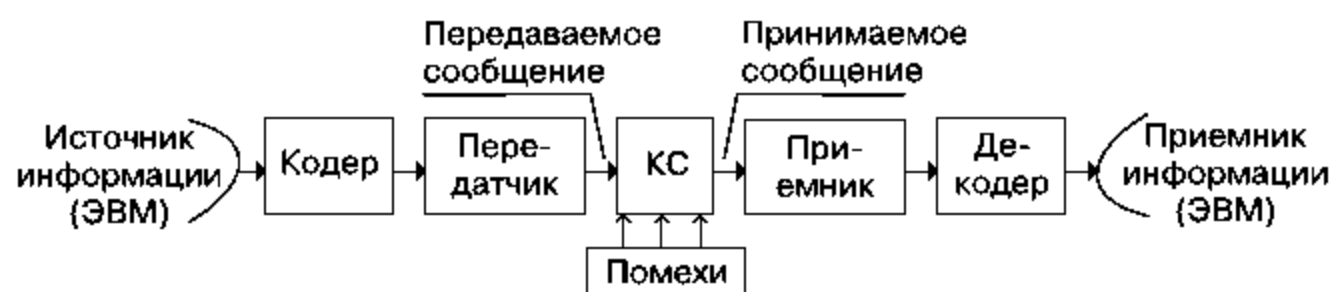


Рис. 3.43. Общая схема передачи сообщений между ЭВМ

При кодировании сообщений безизбыточными равномерными двоичными кодами длиной n используются все 2^n возможных кодовых комбинаций, т.е. с любой из 2^n кодовых комбинаций сопоставляется какой-либо символ внешнего алфавита. Изменение полярности одного или нескольких символов в какой-либо кодовой комбинации приводит к ошибочному ее декодированию. Наиболее опасными для кодированных сообщений являются импульсные помехи, под действием которых 0 может быть принят за 1, и наоборот (рис. 3.44).

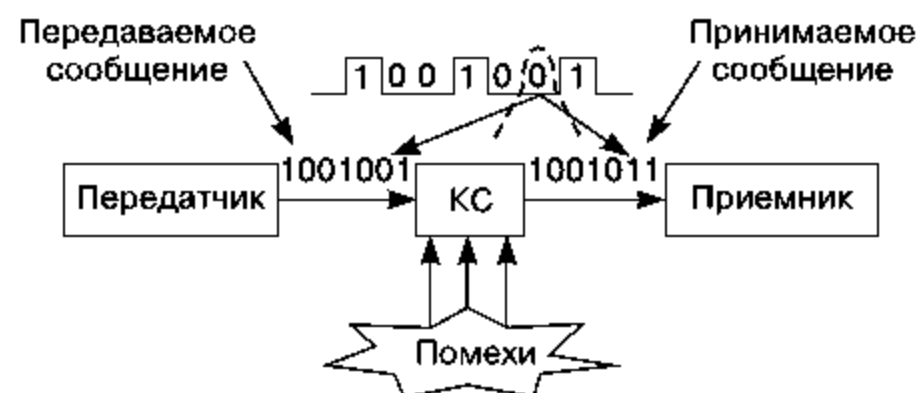


Рис. 3.44. Изменение кода сообщения под действием помех

Пример. Передать по КС с помехами сообщения А, Б, В, Г, используя *безызбыточное* кодирование.

Для кодирования четырех сообщений выберем 2-разрядные двоичные коды, позволяющие образовать четыре различные КК, и присвоим сообщениям следующие коды: А — 00, Б — 01, В — 10, Г — 11.

Если под действием помехи один из элементов кодовой комбинации, например 00, соответствующий А, изменится, то она превращается (трансформируется) в другую разрешенную комбинацию, например 01, которую декодер интерпретирует как сообщение Б (рис. 3.45), и ошибка не будет обнаружена. И так может быть с каждой из четырех кодовых комбинаций.

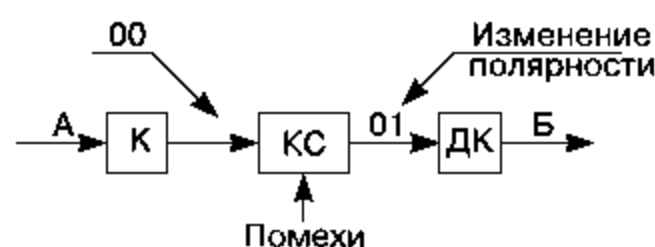


Рис. 3.45. Трансформация сообщения

Из примера следует, что *безызбыточные коды* не обладают помехозащищенностью и не позволяют обнаруживать ошибки. Для обнаружения и исправления ошибок в принимаемых комбинациях необходимо вводить *избыточность*.

Помехоустойчивость избыточного кода обеспечивается благодаря тому, что для передачи информации используются не все 2^n кодовых комбинаций равномерного кода, а лишь часть из них, получившая название *разрешенных КК*. Следовательно, число разрядов в КК должно быть таким, чтобы общее число комбинаций $m = 2^n$ превышало число разрешенных m_p , используемых для передачи информации, т.е. $m > m_p$.

Оставшаяся часть комбинаций $m - m_p$, составляющая *запрещенные КК*, при передаче информации не применяется. В разрешенных комбинациях полярность элементов должна отличаться друг от друга, по крайней мере, в двух разрядах. Если в результате помех разрешенная комбинация превратится в запрещенную, то ошибка может быть обнаружена или исправлена. В подтверждение сказанному те же сообщения А, Б, В, Г закодируем трехразрядными двоичными комбинациями ($n = 3$), для чего из общего числа возможных КК $m = 2^3 = 8$ в качестве *разрешенных m_p* выберем четыре с максимально отличающимися друг

от друга разрядами и каждой из них поставим в соответствие одно из сообщений, например:

$$\left. \begin{array}{l} A — 000 \\ B — 011 \\ B — 101 \\ \Gamma — 110 \end{array} \right\} KK \in m_p.$$

Остальные четыре комбинации (001, 010, 100, 111) относятся к запрещенным m_z .

При тех же условиях передачи сообщений, когда меняется полярность только одного элемента комбинации (рис. 3.46), т.е. действует оди-
ночная помеха, полученная на выходе КС (001), комбинация *не может* быть принята за сообщение, так как не является идентичной ни одной из разрешенных, т.е. $001 \notin m_p$, что и указывает на *ошибку* в передаваемом сообщении. Таким образом, появление запрещенной КК на приемной стороне может служить индикатором того, что произошла ошибка.

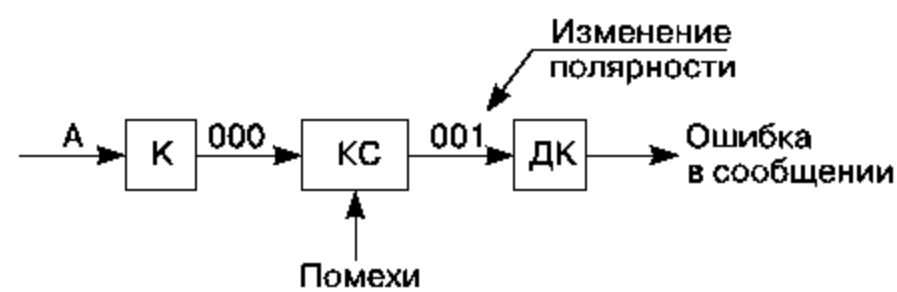


Рис. 3.46. Трансформация КК из разрешенной в запрещенную

Следовательно, 3-разрядный двоичный код является *помехоустойчивым* и позволяет обнаруживать *однократные* ошибки в сообщениях А, Б, В, Г. *Кратностью* ошибки называют число искаженных символов в кодовой комбинации.

Для защиты сообщений А, Б, В, Г от двукратных ошибок избыточность КК необходимо увеличить, т.е. кодировать их более длинными кодовыми комбинациями, например 4-разрядными, выбрав при этом из общего числа возможных комбинаций ($m = 2^4$) четыре разрешенных.

Конечно, и при избыточном кодировании ошибка может быть не обнаружена, если одна разрешенная комбинация трансформируется в другую разрешенную. Однако вероятность такого события гораздо меньше, чем вероятность превращения разрешенной комбинации в запрещенную, т.е. $P_{ip \rightarrow p} < P_{ip \rightarrow z}$.

Существуют и другие способы обнаружения ошибок в кодовых комбинациях, например *контроль на четность*. В этом случае каждому из сообщений А, Б, В, Г ставится в соответствие кодовая комбинация, в которой сумма элементов по mod 2 с учетом избыточного контрольного разряда y равна 0, т.е. $\oplus x_i \oplus y = 0$, где $x_i, y \in \{0, 1\}$.

Тогда любая кодовая комбинация, у которой эта сумма не равна 0, будет считаться *ошибочной*. Значения функции сложения по mod 2 для двух переменных приведены в табл. 3.3.

Таблица 3.3

Сложение по mod 2

x_1	x_2	$x_1 \oplus x_2$
0	0	0
0	1	1
1	0	1
1	1	0

Пример контроля на четность:

	x_1	x_2	x_3	y	
А —	0	0	0	0	$\oplus x_i = 0 \oplus 0 \oplus 0 \oplus 0 = 0$
Б —	0	0	1	1	$\oplus x_i = 0 \oplus 0 \oplus 1 \oplus 1 = 0$
В —	0	1	0	1	$\oplus x_i = 0 \oplus 1 \oplus 0 \oplus 1 = 0$
Г —	0	1	1	0	$\oplus x_i = 0 \oplus 1 \oplus 1 \oplus 0 = 0$

Таким образом, можно сделать вывод, что все четыре сообщения приняты без ошибок.

3.6.3. Систематические коды

Систематический код — это код, который кроме информационных разрядов содержит также контрольные разряды. Последние получаются в результате некоторых линейных операций над информационными разрядами. Поэтому можно говорить, что систематический код обладает избыточностью. *Избыточность* указывает степень удлинения КК для достижения определенной *корректирующей способности* кода, с которой связывают обычно возможность обнаружения и исправления ошибки. При этом *абсолютная* избыточность будет равна числу контрольных разрядов k , а *относительная* — отношению k/n , где $n = m + k$ — общее число разрядов в КК (рис. 3.47). Число информационных разрядов m определяется числом передаваемых дискретных сообщений.

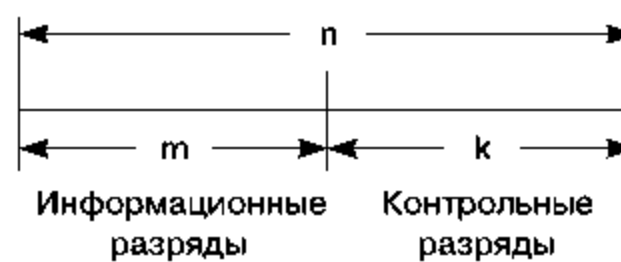


Рис. 3.47. Систематический код

Информационные и контрольные разряды во всех кодовых комбинациях занимают всегда одни и те же позиции.

3.6.4. Оценка корректирующей способности кода

Помехоустойчивые систематические коды обладают различными корректирующими способностями. Одни из них могут только обнаруживать ошибки, а другие не только обнаруживать, но и исправлять их. Остановимся на двух подходах к оценке корректирующих способностей кода — *вероятностном* и *минимального кодового расстояния*.

Вероятностная оценка корректирующей способности кода

Выше говорилось, что импульсные помехи носят случайный характер. Полной характеристикой *случайной* величины с вероятностной точки зрения является закон ее распределения, т.е. заданная в той или иной форме связь между ее возможными значениями и вероятностями их появления. Если искажение отдельных символов кодовой последовательности считать событиями независимыми и принять, что искажения нулевых и единичных символов одинаковы, то в качестве модели помехи используют помеху с *биномиальным* законом распределения.

Количественно корректирующая способность кода определяется *вероятностью* обнаружения или исправления ошибки. Вероятность ошибки кратности i в КК из n -символов применительно к помехе с биномиальным законом вычисляется следующим образом:

$$P_i^n = C_n^i P_z^i (1 - P_z)^{n-i}, \quad (3.24)$$

где C_n^i — число кодовых комбинаций, каждая из которых содержит i искаженных элементов (это число сочетаний из n по i):

$$C_n^i = \frac{n!}{i!(n-i)!};$$

P_z — вероятность искажения одного символа в КК;

$P_z^i (1 - P_z)^{n-i}$ — вероятность того, что искажены i символов КК, а остальные $(n - i)$ не искажены (по теореме умножения вероятностей).

Тогда полная вероятность искажения в КК:

$$P_{\Sigma} = \sum_{i=1}^j P_i^n = \sum_{i=1}^j C_n^i P_3^i (1 - P_3)^{n-i}. \quad (3.25)$$

Пример. Если принять $n = 6$, $P_3 = 0,01$ и вычислить P_i^n по формуле (3.24) для $i = 1, 2, 3$ ($P_1^6 = 0,057$, $P_2^6 = 0,014$, $P_3^6 = 0,000019$), можно видеть, что с повышением кратности ошибки вероятность ее резко убывает. Наибольший вес в сумме вероятностей (3.25) имеет вероятность искажения одного символа, и, следовательно, основное внимание необходимо обратить на обнаружение и исправление *одиночной ошибки*.

Кодовое расстояние

Для оценки корректирующих свойств кода используют еще понятие *минимального кодового расстояния* $d_{\min}$.

Кодовое расстояние $d(A, B)$ для кодовых комбинаций A и B определяется числом символов, которыми они отличаются друг от друга.

Для определения $d(A, B)$, которое называют еще *расстоянием по Хэммингу*, достаточно вычислить вес третьей кодовой комбинации, которая получается в результате сложения по mod2 исходных комбинаций A и B . *Весом* $V(S)$ двоичной кодовой комбинации будем называть число единиц в ней.

Пример 1. Найти $d(A, B)$ для кодовых комбинаций A и B :

$$A = 011011100_2; B = 100111001_2.$$

Для этого сложим их по mod2 и найдем вес суммарной двоичной комбинации

$$\begin{array}{r} A = 011011100_2 \\ \oplus B = 100111001_2 \\ \hline S = 111100101_2 \end{array}$$

$$V(S) = \sum_{i=1}^9 s_i = 6, (s_i = 1, i = \overline{1, 9}),$$

которая и определяет кодовое расстояние заданных комбинаций $d(A, B)$.

Если коды рассматривать как некоторые геометрические (пространственные) фигуры (например, триаду можно представить в виде единичного куба, имеющего координаты вершин, которые отвечают двоичным символам (рис. 3.48)), то в этом случае *кодое расстояние* воспринимается как *сумма длин ребер* между соответствующими вершинами куба (принято, что длина одного ребра равна единице). Тогда, например, $d(D, C) = 3$.

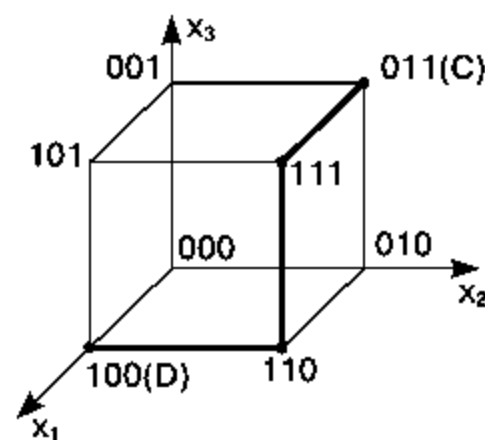


Рис. 3.48. Геометрическое представление кодов

Минимальное кодовое расстояние $d_{\min}$ — это минимальное расстояние, взятое по всем параметрам разрешенных кодовых комбинаций.

В теории кодирования показано, что систематический код способен обнаруживать ошибки только тогда, когда $d_{\min} \geq 2t$, где t — кратность обнаруживаемых ошибок (для одиночных ошибок $t = 1$).

Действительно, при $d_{\min} \geq 2t$ никакой вариант t -кратной ошибки не может перевести одну разрешенную кодовую комбинацию в другую разрешенную.

Так, если $d = 2$, то при $t = 1$ ни одна из разрешенных 3-разрядных кодовых комбинаций не переходит в другую разрешенную. Множество m_p может быть образовано по принципу четности в них числа 1. Так, КК $000, 011, 101, 110 \in \{m_p\}$; тогда $001, 010, 111, 100 \in \{m_s\}$. При таком кодировании обнаружатся все одиночные ошибки ($t = 1$), а также другие ошибки нечетной кратности (тройные).

Если же $d_{\min} \leq 2t$, то существует хотя бы одна пара кодовых комбинаций, отстоящих друг от друга на расстоянии, меньшем или равном $2t$, и найдется такой вариант t -кратной ошибки, который трансформирует одну комбинацию в другую.

Пример 2. Для кодовых комбинаций $A = 101100_2$ и $B = 100000_2$ кодовое расстояние $d(A, B) = 2$, так как

$$\begin{array}{r} A = 101100_2 \\ \oplus B = 100000_2 \\ \hline S = 001100_2 \end{array}$$

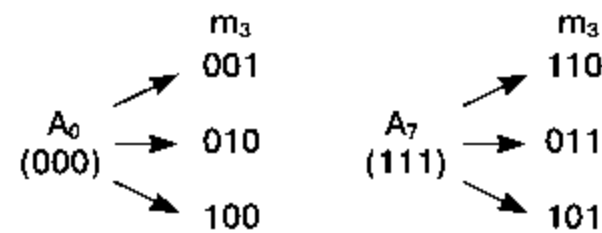
Следовательно, 2-кратная ошибка вида $S = 001100_2$ превратит кодовую комбинацию A в комбинацию B :

$$\begin{array}{r} A = 101100_2 \\ \oplus S = 001100_2 \\ \hline B = 100000_2 \end{array}$$

В случаях, когда необходимо не только обнаружить ошибку, но и исправить ее, минимальное кодовое расстояние должно удовлетворять условию $d_{\min} \geq 2t + 1$.

Исправление ошибок. Общая идея исправления ошибок кратности не более t заключается в следующем. Число возможных кодовых комбинаций помехоустойчивого кода разбивается на классы по числу разрешенных кодовых комбинаций. Разбиение осуществляется таким образом, чтобы в каждый класс входили *одна разрешенная* кодовая комбинация и ближайшие к ней запрещенные. При *декодировании* определяется, какому классу принадлежит принятая кодовая комбинация. Если она принята с ошибкой, т.е. является запрещенной, то исправляется на разрешенную кодовую комбинацию, принадлежащую тому же классу.

Например, для исправления одиночной ошибки ($t = 1$) каждой разрешенной кодовой комбинации сопоставим подмножество запрещенных. Чтобы эти подмножества не пересекались, d между разрешенными КК должно быть не менее трех, т.е. $d_{\min} \geq 3$. Тогда за разрешенные кодовые комбинации можно принять $A_0 = 000_2$ и $A_7 = 111_2$ и приписать каждой из них подмножество тех запрещенных комбинаций, к которым может привести одиночная ошибка.



3.6.5. Декодирование данных

После приема сообщения декодирование может производиться таким образом, что принятая кодовая комбинация отождествляется с той разрешенной, которая находится от нее на минимальном кодовом расстоянии $d_{\min}$. Такое декодирование называется декодированием по *методу максимального правдоподобия*.

В общем случае для обеспечения возможности исправления всех ошибок кратности до t включительно при декодировании по *методу максимального правдоподобия* каждая из ошибок должна приводить к запрещенной комбинации, относящейся к подмножеству исходной разрешенной.

3.6.6. Методы эффективного кодирования

Одной из основных характеристик кода является *избыточность*. Это полезное свойство, так как оно повышает помехоустойчивость кода. Однако для передачи избыточных кодов по КС требуется больше времени. Кроме того, они занимают больший объем памяти при хранении информации. Большая избыточность не всегда оправдана требованиями помехоустойчивости при передаче и хранении информации. Поэтому возникает задача устранения избыточности, получившая название *эффективного кодирования*.

Устранять избыточность можно различными способами, например путем *статистического* (вероятностного) кодирования, в котором используется тот факт, что не все символы в передаваемом сообщении встречаются с одинаковой частотой. При этом *часто* встречающимся буквам ставят в соответствие *короткие КК*, а *редко* встречающимся — *длинные*, что позволяет минимизировать среднее число двоичных символов, используемых для представления одной буквы сообщения, и представить сообщение с меньшим числом знаков. При отсутствии помех это позволяет уменьшить время передачи сообщения.

Алгоритмы, позволяющие автоматически строить неравномерные коды на основании известных частот появления символов, были впервые рассмотрены К. Шенноном и Н. Фано. В их методах нет существенных различий, поэтому соответствующий эффективный код получил название *кода Шеннона—Фано*.

Рассмотрим подробнее сам алгоритм кодирования, предварительно определив критерий оценки эффективности кодирования.

Эффективность кода определяют через коэффициент избыточности [2]

$$K_k = \frac{n_{\text{ср}} - n_{\text{min}}}{n_{\text{ср}}}, \quad (3.26)$$

где $n_{\text{ср}}$, n_{min} — средняя и минимальная длина кодовой комбинации соответственно.

Значение K_k изменяется от 0 (отсутствие избыточности в случае $n_{\text{ср}} = n_{\text{min}}$) до 1 (избыточность неограниченно велика) и характеризует качество помехоустойчивого кода: чем меньше избыточность кода при прочих равных условиях, тем код лучше.

Метод кодирования по Шеннону—Фано

1. Буквы алфавита сообщения выписываются в столбец в порядке убывания их вероятностей (табл. 3.4).

2. Далее они разделяются на две группы так, чтобы суммы вероятностей в каждой из них были по возможности одинаковы. Разбиение производится, пока в каждой из групп не останется по одному сообщению.
3. При соответствующем разбиении принадлежность к верхней половине обозначается символом 0, а к нижней — 1, что и предопределяет символы кода Шеннона—Фано.

Пример. Сообщения X_1, X_2, X_3, X_4 с вероятностями появления символов $P(X_1) = 0,5, P(X_2) = 0,25, P(X_3) = 0,125, P(X_4) = 0,125$ закодировать по Шеннону—Фано.

В соответствии с описанным методом кодирования построим табл. 3.4.

Таблица 3.4

Кодирование сообщений по Шеннону—Фано

X_i	$P(X_i)$	N деления			Символ кода			$n(X_i)$
		1	2	3	1	2	3	
X_1	0,5	I			0	—	—	1
X_2	0,25		I		1	0	—	2
X_3	0,125			I	1	1	0	3
X_4	0,125			II	1	1	1	3

С учетом статистических свойств источника сообщений длина КК для сообщений X_1, X_2, X_3, X_4 будет различной. Число символов в них $n(X_i)$ будет равно соответственно одному, двум и трем: $X_1 = 0_2, X_2 = 10_2, X_3 = 110_2, X_4 = 111_2$ (см. табл. 3.4). Среднее число символов, т.е. средняя длина КК

$$n_{cp} = \sum_{i=1}^4 P(X_i) n(X_i) =$$

$$= 0,5 \cdot 1 + 0,25 \cdot 2 + 0,125 \cdot 3 + 0,125 \cdot 3 = 1,75. \quad (3.27)$$

Если при равномерном кодировании требуется два двоичных разряда ($n = 2$), чтобы различить 4 сообщения, то в данном примере $n_{cp} = 1,75$, т.е. меньше.

Энтропия сообщения, составленного из символов X_1, X_2, X_3, X_4 , равна

$$H(x) = - \sum_{i=1}^4 P(X_i) \log_2 P(X_i) = 1,75.$$

Минимальная длина КК выходного алфавита ($m = 2$), необходимая для передачи сообщения без потери информации, составленная из символов входного алфавита ($X_i \in X$), равна

$$n_{min} = \frac{H(x)}{\log_2 m} = \frac{1,75}{\log_2 2} = 1,75.$$

Коэффициент избыточности кода $K_{\text{и}}$, построенного по алгоритму Шеннона—Фано, в соответствии с выражением (3.26) равен нулю, т.е. построенный код не имеет избыточности.

Длительность передаваемых сообщений при *равномерном* кодировании будет равна $\tau_c = 2\tau$. Поскольку при эффективном кодировании длительность сообщений τ_i будет различной, находят *усредненное* значение:

$$\tau_{\text{с.ср}} = \sum_{i=1}^4 P(X_i) \tau_{\text{с}i} = 0,5 \cdot 1\tau + 0,25 \cdot 2\tau + (0,125 \cdot 3\tau) \cdot 2 = 1,75\tau.$$

Очевидно, что среднее время передачи сообщения при эффективном кодировании сокращается ($\tau_{\text{с.ср}} < \tau_c$).

Характеристики кода Шеннона—Фано:

- ♦ код является *неравномерным* и *неизбыточным*;
- ♦ каждый символ кода кодируется целым числом битов;
- ♦ код является *префиксным*, т.е. допускает однозначное декодирование (любая короткая кодовая комбинация не является началом более длинной).

Пример. Декодировать заданную последовательность символов в кодах рассмотренного выше примера (см. табл. 3.4).

$$\begin{array}{ccccccc} & \text{Код} & & \text{Сообщение} & & & \\ 0 & 111 & 10 & 110 & 0_2 & \rightarrow & X_1 X_4 X_2 X_3 X_1. \\ \hline & \underbrace{}_{X_1} & \underbrace{}_{X_4} & \underbrace{}_{X_2} & \underbrace{}_{X_3} & \underbrace{}_{X_1} & \end{array}$$

Сокращение времени передачи сообщения (при отсутствии помех) за счет использования оптимальных последовательностей кодов является несомненным достоинством кодирования по Шеннону—Фано. Эффект достигается благодаря присвоению *высоковоероятным* сообщениям более *коротких* КК, и наоборот.

Недостатки кода:

- 1) поскольку код не обладает избыточностью, каждая ошибка ведет к искажению информации;
- 2) так как код *неравномерный*, некоторые символы кодируются длинными последовательностями, что приводит к длительным временным задержкам;
- 3) метод Шеннона—Фано не всегда приводит к однозначному построению кода, так как при разбиении на подгруппы (см. табл. 3.4) большей по вероятности можно сделать как верхнюю, так и нижнюю подгруппы.

От последнего недостатка свободен код Хаффмена.

Метод кодирования по Хаффмену

1. Все символы сообщения, как и в методе Шеннона—Фано, выписывают в основной столбец в порядке убывания их вероятностей.
2. Два нижних, т.е. наименее вероятных, символа объединяются в один вспомогательный с суммарной вероятностью. Верхнему символу этого объединения ставится в соответствие (в качестве цифры КК) цифра 0, нижнему — 1.
3. Вероятности символов, не участвовавших в объединении, и полученная суммарная вероятность снова располагаются в порядке убывания в дополнительном столбце (столбец 1 в табл. 3.5), и два последних, наименее вероятных символа, опять объединяются. Процесс продолжается до тех пор, пока не получим единственный вспомогательный символ с вероятностью 1.
4. Для составления кодовой комбинации, соответствующей заданной букве алфавита сообщений, необходимо проследить путь перехода к ней по строкам и столбцам таблицы. В последнем столбце, содержащем два знака, тому знаку, который имеет большую вероятность, присваиваем символ 1, а меньшую — 0. Двум объединенным знакам в предпоследнем столбце символы выставляются аналогично. Процесс продолжается до возвращения в основной столбец.

Пример. Выполнить кодирование по Хаффмену для ансамбля сообщений $(x_1 - x_8)$, приведенного в табл. 3.5.

Таблица 3.5

Кодирование сообщений по Хаффмену

Обозначение	Вероятность	Вспомогательные столбцы						
		1	2	3	4	5	6	7
x_1	0,22	0,22	0,22	→0,26	→0,32	→0,42	→0,58	1
x_2	0,20	0,20	0,20	0,22	0,26	0,32	0,42	→1 0
x_3	0,16	0,16	0,16	0,20	0,22	0,26	0	
x_4	0,16	0,16	0,16	0,16	0,20	0		
x_5	0,1	0,1	→0,16	0,16	0			
x_6	0,1	0,1	0,1	0				
x_7	0,04	→0,06	0					
x_8	0,02	0						

Более наглядно задачу можно решить, построив *кодвое дерево* — граф с взвешенными ребрами (рис. 3.49). Из точки, соответствующей вероятности 1, направляют две ветви. Ветви с большей вероятностью присваивают символ 1, а с меньшей — 0. Такое последовательное ветвление продолжается до тех пор, пока не доходят до вероятности каждой буквы. Двигаясь по кодвому дереву сверху вниз, можно записать для каждой буквы соответствующую ей кодовую комбинацию: x_1 — 01, x_2 — 00, x_3 — 111, x_4 — 110, x_5 — 100, x_6 — 1011, x_7 — 10101, x_8 — 10100.

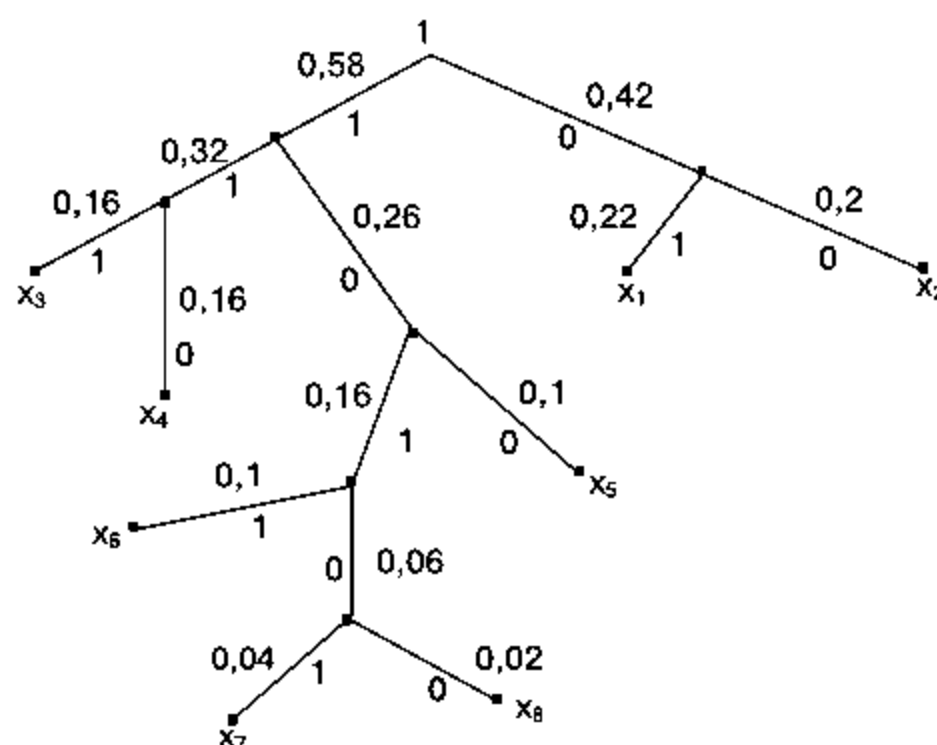


Рис. 3.49. Кодовое дерево

Код Хаффмена гарантирует однозначное кодирование каждой буквы сообщения с наименьшим для данного распределения вероятностей средним числом символов на букву и так же, как и код Шеннона—Фано, является *префиксным*.

3.7. СЖАТИЕ ДАННЫХ

Сжатие (компрессия) данных — это процесс сокращения числа битов, необходимых для передачи и хранения некоторого объема информации. Сжатие может быть *без потерь* — информация, восстановленная из сжатого сообщения, в точности соответствует исходной, и *с потерями* — восстановленная информация только частично соответствует исходной. *Первое* применяется для сжатия текстов и графики (алгоритм Лемпеля—Зива, RLE—Run Length Encoding, алгоритм Хаффмена и др.).

Алгоритм Лемпеля—Зива (Lempel—Ziv, LZ) лежит в основе архиваторов (pkzip, arj, lha) и программ динамического сжатия дисков

(Stacker, DoubleSpace). Основная идея этого алгоритма состоит в том, что второе и последующие вхождения некоторой строки символов в сообщение заменяются ссылкой на ее первое появление в сообщении.

Алгоритм RLE применяют для сжатия графики и видеоматериалов. Непрерывная последовательность одинаковых символов заменяется двумя байтами. В первом байте — символ, во втором — счетчик, т.е. число, которое показывает, сколько таких символов идет подряд.

Кодирование по Хаффмену состоит в замене информационных символов кодовыми последовательностями различной длины. Чем чаще используется символ, тем короче кодовая последовательность.

Сжатие с потерями [алгоритмы JPEG (Joint Photographic Expert Group), M-JPEG, MPEG (Motion Picture Expert Group)] ориентировано на сжатие неподвижных изображений.

Алгоритм M-JPEG использует для компрессии видео, в котором каждый отдельный кадр сжимается по методу JPEG.

Алгоритм MPEG ориентирован на обработку видео. При формировании потока данных исходят из предположения о том, что два соседних кадра в видеопоследовательности мало отличаются. Опорные кадры сжимаются по методу JPEG и передаются относительно редко. В основном передаются изменения между соседними кадрами.

Из приведенного краткого обзора алгоритмов сжатия очевидны два вывода:

- ◆ нет алгоритма, одинаково эффективного для данных разной природы;
- ◆ приведенные алгоритмы рассчитаны на сжатие данных, в которых есть последовательности одинаковых символов или одни символы встречаются чаще других.

Так как на сжатие данных передающая сторона тратит дополнительное время, к которому нужно еще прибавить аналогичные затраты времени на разворачивание этих данных принимающей стороной, то выгоды от сокращения времени на передачу сжатых данных обычно бывают заметны только для низкоскоростных каналов (около 64 Кбит/с). Многие программные и аппаратные средства сети способны выполнять *динамическую* компрессию, совмещенную с передачей данных. *Статическая* компрессия обеспечивает предварительное сжатие данных (например, с помощью популярных архиваторов типа ARJ, RAR, WinZip), после чего они отсылаются в сеть.

На практике используют ряд алгоритмов сжатия, каждый из которых применим к определенному типу данных.

Так, информация, содержащаяся в файлах, во многих случаях избыточна, и для устранения избыточности используют специальные ме-

тоды сжатия данных, основанные на поиске этой избыточной информации и последующем ее кодировании с целью получения минимального объема. Для представления сообщений с меньшим расходом знаков (разрядов) часто применяют энтропийные методы кодирования, например по Шеннону—Фано или Хаффмену (см. подраздел 3.6.6).

Рассмотрим сказанное на примере сообщений, передаваемых с помощью букв русского алфавита. Так как *е* и *ё*, а также *ь* и *ъ* можно принимать за одну букву и с учетом пробела общее число букв $n = 32$. Таким образом, каждую букву русского алфавита можно закодировать комбинацией из 5 двоичных разрядов. Если принять, что в сообщении будет N букв, то общее число разрядов сообщения равно $5N$.

Поскольку буквы в сообщении появляются с разной вероятностью, например, *О* и *Е* — чаще, *Э*, *Ф* — реже, то среднее количество информации на один символ сообщения, т.е. энтропия, рассчитывается по формуле

$$H = \sum_{i=1}^{32} P_i \log \frac{1}{P_i}, \quad (3.28)$$

где P_i — вероятность появления i -й буквы.

То есть энтропия определяется статистическими свойствами источника сообщения. В случае равномерности и независимости букв энтропию можно вычислить как $H_0 = \log_2 32 = 5$ бит/букву. В случае разных вероятностей появления букв энтропия равна $H_0 = 4,1$ бит/букву.

Таким образом, можно видеть, что описание каждой буквы пятью двоичными разрядами является избыточным. Сообщение из N букв в случае *равновероятных* исходов потенциально может переносить $I_0 = 5N$ бит информации, в случае *разновероятных* — количество переносимой информации составит $I = 4,1N$ бит.

3.7.1. Пример сжатия текстовой информации

Попробуем сжать передаваемое сообщение: *_мама_мыла_раму_*, используя энтропийное кодирование. Символы сообщения закодируем двумя способами: *равномерными* и *неравномерными* (по Шеннону—Фано) кодовыми комбинациями.

Равномерное кодирование

С учетом пробелов алфавит сообщения содержит семь символов, для кодирования которых достаточно трех двоичных разрядов (табл. 3.6). Длина комбинации определяется как ближайшее большее целое (ББЦ)

$n_{kk} = \lceil \log_2 7 \rceil_{\text{ББЦ}} = 3$, а заданный текст сообщения может быть представлен в виде последовательности двоичных символов длиной 48 бит:

000 010 001 010 001 000 010 110 011 001 000 100 001 010 101 000
 _ м а м а _ м ы л а _ р а м у _

Таблица 3.6

Равномерное кодирование

Символ	КК
—	000
А	001
М	010
Л	011
Р	100
У	101
Ы	110

Поскольку общее число символов в сообщении равно 16, количество информации в сообщении

$$I_0 = H_0 \cdot 16 = 3 \frac{\text{бита}}{\text{символ}} \cdot 16 \text{ символов} = 48 \text{ бит.}$$

Однако заметим, что расходование 3 бит на букву не означает, что каждая двоичная комбинация переносит 3 бита информации.

Неравномерное кодирование

Вероятности появления символов P_i легко оценить, посчитав, сколько раз каждый из них встречается в передаваемом сообщении:

$$P(-) = P(A) = P(M) = 1/4; P(L) = P(P) = P(Y) = P(Ы) = 1/16.$$

Далее в соответствии с методом кодирования (см. подраздел 3.6.6) строим табл. 3.7.

Таблица 3.7

Кодирование символов по Шеннону—Фано

Символ сообщения, i	Вероятность, P_i	N деления				Символы кода			
		1	2	3	4	1	2	3	4
—	1/4	I	I			0	0	-	-
А	1/4					0	1	-	-
М	1/4					1	0	-	-
Л	1/16	II	II	I	I	1	1	0	0
Р	1/16					1	1	0	1
У	1/16					1	1	1	0
Ы	1/16					1	1	1	1

Результаты неравномерного (энтропийного) кодирования символов сообщения сведены в табл. 3.8.

Таблица 3.8

Неравномерное кодирование

Символ	КК
—	00
А	01
М	10
Л	1100
Р	1101
У	1110
Ы	1111

После подстановки полученных кодов вместо символов то же самое передаваемое сообщение будет представлено последовательностью двоичных символов длиной 40 бит:

00 10 01 10 01 00 10 1111 1100 01 00 1101 01 10 1110 00

_ м а м а _ м ы л а _ р а м у _

С учетом вероятности появления символов в сообщении энтропия H_1 (среднее количество битов на символ) рассчитывается по формуле

$$H_1 = P_1 \log \frac{1}{P_1} + P_2 \log \frac{1}{P_2} + \dots + P_7 \log \frac{1}{P_7} = 2,5 \text{ бит/символ}$$

Тогда количество информации в том же сообщении будет равно

$$I_1 = H_1 \cdot 16 = 2,5 \cdot 16 = 40 \text{ бит.}$$

Таким образом, в результате энтропийного кодирования длина передаваемого сообщения сократилась с 48 до 40 бит.

3.7.2. Сжатие изображений

Рассмотренные модели сжатия текстовой информации могут быть применены и для сжатия изображений.

Часто передаваемые данные содержат большое число повторяющихся байтов. Например, при передаче небольшой картинки в черно-белом изображении будет идти много байтов, отображающих белую поверхность, определенное число — серую и немного — черную. Передатчик сканирует последовательность передаваемых байтов и, если обнаруживает последовательность с определенным числом одинаковых байтов, заменяет ее байтом, в котором указывает его значение (может указывать число его повторений и др.).

В цифровом изображении главным объектом является *пиксель*. Пусть требуется передать массив данных, представляющий собой «копию экрана», где на белом фоне изображена картинка (рис. 3.50).

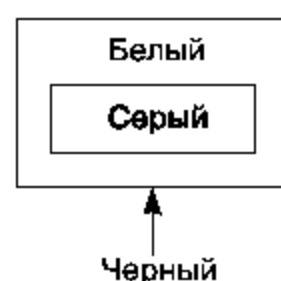


Рис. 3.50. Копия экрана

Простейшая схема сжатия изображений представлена на рис. 3.51, а кодирование байтов — в табл. 3.9.

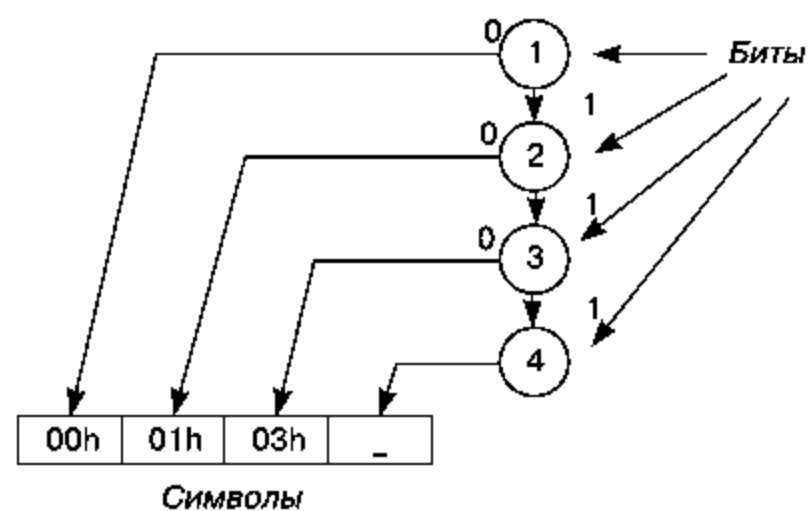


Рис. 3.51. Сжатие изображений

Таблица 3.9

Кодирование байтов

Исходные байты	Упакованные байты
0000 0000	0
0000 0001	10
0000 0011	110

Пусть картинка содержит 90 байт:

80 байт — 0000 0000 — белый;

7 байт — 0000 0001 — серый;

3 байта — 0000 0011 — черный.

Программа сжатия выполняет несколько этапов:

- 1) анализ повторения байтов. Строится гистограмма;
- 2) составляются новые кодирующие последовательности (см. табл. 3.9);
- 3) исходный массив сжимается. К нему пишется кодировочная таблица.

Рассмотрим пример расшифровки полученного сообщения.

Полученное сообщение:

$$\begin{array}{cc} 2 & C \\ \hline 0010 & 110010 \\ \hline 00h & 00h & 01h & 03h & 00h & 01h \end{array} \leftarrow \text{распаковка}$$

С учетом сжатия картинка (см. рис. 3.50) вместо 90 байт будет содержать 103 бита:

80 байт	→	80 бит
7 байт	→	7·2 бит
3 байт	→	3·3 бит

3.8. КАНАЛЫ ПЕРЕДАЧИ ИНФОРМАЦИИ

3.8.1. Общая характеристика

В подразделе 3.3.1 приводилась простейшая модель системы передачи данных (см. рис. 3.32) и описывались основные проблемы, связанные с передачей данных на расстояние. Рассмотренные ниже вопросы технической реализации идеи передачи данных от источника к адресату позволяют вернуться к рассмотренной модели уже на новом уровне (рис. 3.52) с учетом реальных технических решений, физических средств и способов, используемых при транспортировке данных, чтобы оценить основные технические характеристики автоматизированных систем передачи информации:

- ♦ скорость передачи информации;
- ♦ пропускную способность;
- ♦ ширину полосы;
- ♦ помехозащищенность, т.е. степень противодействия системы воздействию на нее помехам;
- ♦ достоверность передачи информации.

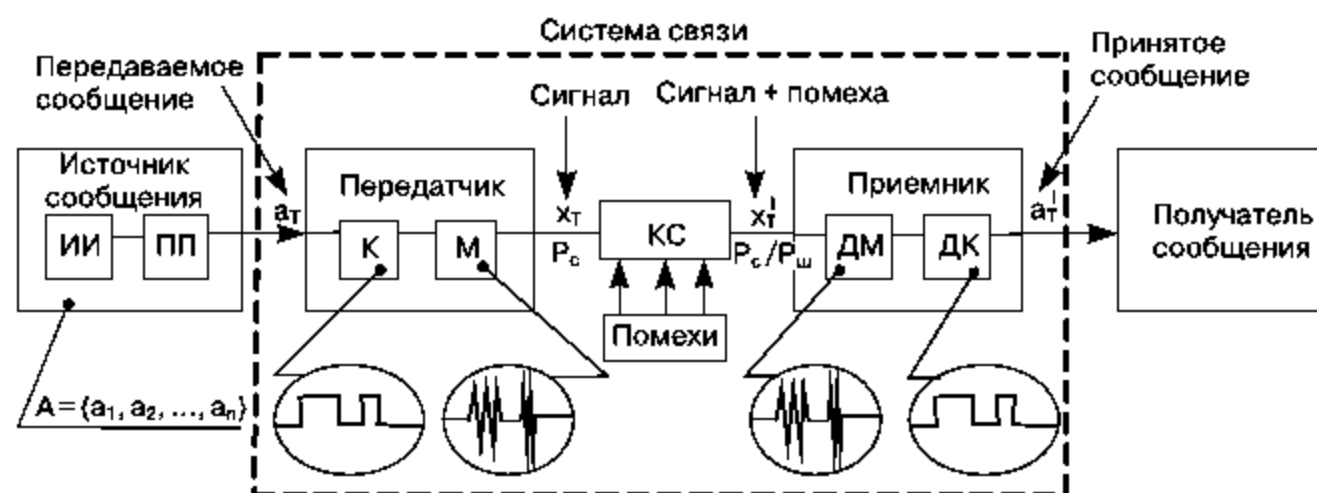


Рис. 3.52. Система передачи информации

Поскольку на КС могут действовать помехи, то следует говорить о перечисленных показателях в системах с помехами и с отсутствием помех в канале связи.

Скорость передачи

Скорость передачи информации $\bar{I}$ определяется количеством информации, передаваемой по КС в единицу времени, и в общем случае зависит от статистических свойств источника сообщений и параметров КС.

Если через канал связи *без помех* передается последовательность дискретных сообщений длительностью T , то скорость передачи информации по каналу определяется выражением:

$$\bar{I} = \lim_{T \rightarrow \infty} \frac{I}{T},$$

где I — количество информации, содержащейся в сообщении, бит/с.

Скорость передачи информации $\bar{I}$ зависит от:

- ♦ статистики передаваемых сообщений;
- ♦ способа передачи по каналу связи (цифровой или аналоговый);
- ♦ характеристики помех и их интенсивности.

Пропускная способность

Максимально возможное значение скорости передачи информации

$\bar{I}_{\max}$ принимают за *пропускную способность* канала без помех, т.е. $C = \bar{I}_{\max}$. Пропускная способность C измеряется в битах в секунду (бит/с), а также в производных единицах: Кбит/с, Мбит/с.

Заметим, что *пропускная способность* — это характеристика КС, которая не зависит от скорости передачи информации. Количественно пропускная способность КС оценивается максимальным числом двоичных единиц информации (бит), которое данный канал может передать за 1 с.

Для наиболее эффективного использования КС необходимо, чтобы скорость передачи информации $\bar{I}$ была как можно ближе к пропускной способности канала. Если скорость поступления информации на вход КС превышает C , то по КС будет передана не вся информация. Основное условие согласования источника информации и канала связи определяется соотношением:

$$\bar{I} \leq C. \quad (3.29)$$

Согласование осуществляется путем кодирования сообщения, выполняемого кодером K , после чего сообщение представляется в ви-

де последовательности прямоугольных импульсов. Затем сигналы модулируются (аналоговая передача) и поступают на КС. Согласование частоты передаваемых сигналов F_c с частотой канала связи F_k обеспечивается путем модуляции. Кодер и модулятор входят в состав передатчика сигнала.

При передаче сигналов по физическим КС различают два вида кодирования — *физическое* и *логическое*. При этом оба вида оказывают влияние на пропускную способность канала.

Способ представления дискретной информации в виде сигналов, подаваемых на линию связи, называется *физическим* кодированием. От выбранного способа кодирования зависит спектр сигналов и соответственно пропускная способность линии. Таким образом, для одного способа кодирования линия может иметь одну пропускную способность, а для другого — другую.

Если несущий сигнал изменится так, что равновероятно можно различить только два состояния его информативного параметра, то по оценке Р. Хартли любое изменение сигнала будет соответствовать наименьшей единице информации — *биту*. Если же сигнал может иметь *более двух* различных состояний, то любое его изменение будет нести *несколько битов* информации. Например, если информативными параметрами являются фаза и амплитуда синусоиды, причем различаются 4 состояния фазы: 0, 90, 180 и 270° и два значения амплитуды сигнала, то информационный сигнал может иметь 8 различных состояний и при одном изменении сигнала передается 3 бита информации.

Число изменений информативного параметра несущего периодического сигнала в секунду измеряется в *бодах* (baud). Период времени между соседними изменениями информативного параметра сигнала называется *тактом работы передатчика* или бодовым интервалом.

Пропускная способность линии в битах в секунду в общем случае не совпадает с числом бодов. Она может быть как выше, так и ниже числа бодов, что зависит от способа кодирования.

Если сигнал имеет более двух различных состояний, то пропускная способность в битах в секунду будет выше, чем число бодов. Так, модем, работающий со скоростью 2400 бод (с тактовой частотой 2400 Гц), при трех различных состояниях сигнала будет передавать информацию со скоростью 7200 бит/с.

При использовании сигналов с двумя различными состояниями может наблюдаться обратная картина. Это часто происходит потому, что для надежного распознавания приемником информации каждый бит в последовательности кодируется с помощью нескольких изменений

информативного параметра несущего сигнала. Например, при кодировании единичного значения бита импульсом положительной полярности, а нулевого значения бита — импульсом отрицательной полярности физический сигнал дважды изменяет свое состояние при передаче каждого бита. При таком кодировании пропускная способность линии в 2 раза ниже, чем число бодов, передаваемое по линии.

При *логическом* кодировании, которое выполняется до физического, бит исходной информации заменяется новой последовательностью битов, несущей ту же информацию, но обладающей дополнительными свойствами, например возможностью для приемной стороны обнаруживать ошибки в принятых данных. Сопровождение каждого байта исходной информации одним битом четности — это пример очень часто применяемого способа логического кодирования при передаче данных с помощью модемов. Другим примером логического кодирования может служить шифрование данных, обеспечивающее их конфиденциальность при передаче по общественным каналам связи. При логическом кодировании чаще всего исходная последовательность битов заменяется более длинной последовательностью, поэтому пропускная способность канала по отношению к полезной информации при этом уменьшается.

Таким образом, *пропускная способность системы передачи информации* зависит и от характеристик передаваемых сигналов, и от характеристик ЛС, в частности ее полосы пропускания. Существует прямая связь между шириной полосы и скоростью передачи данных — чем шире полоса передающей системы, тем более высокой скорости передачи данных можно достичь.

Из сказанного можно сделать *вывод*: если считать, что полоса сигнала сосредоточена около некоторой частоты, называемой *центральной частотой*, то чем больше эта центральная частота, тем шире возможная полоса и, следовательно, выше потенциальная скорость передачи данных. Если, например, сигнал сосредоточен на частоте 2 МГц, то максимальная ширина полосы составляет 4 МГц.

Ограниченность полосы пропускания приводит к частотным искажениям сигнала. Если значимые гармоники сигнала (т.е. те гармоники, амплитуды которых вносят основной вклад в результирующий сигнал) попадут в полосу пропускания линии, то такой сигнал будет хорошо проходить по данной линии связи и приемник сможет правильно распознать информацию, отправленную по линии передатчиком. Если же значимые гармоники выходят за границы полосы пропускания линии связи, то сигнал будет сильно искажаться, приемник будет ошибаться при распознавании информации, и следовательно, информация не сможет передаваться с заданной пропускной способностью.

Связь между полосой пропускания линии и ее максимально возможной пропускной способностью независимо от принятого способа физического кодирования установили Шеннон и Хартли. Эта формула называется законом Шеннона—Хартли:

$$C = B \log_2 \left(1 + \frac{P_c}{P_{\text{ш}}} \right),$$

где C — максимальная пропускная способность линии, бит/с;
 B — ширина полосы пропускания линии, Гц;
 $P_c/P_{\text{ш}}$ — отношение мощностей сигнала и шума, дБ.

Пропускную способность линии можно повысить за счет увеличения мощности передатчика или уменьшения мощности шума (помех) на линии связи. Обе эти составляющие поддаются изменению с большим трудом.

Найквист вывел формулу, определяющую зависимость максимальной скорости передачи информации C (бит/с) от ширины полосы пропускания B без учета шума в канале:

$$C = 2B \log_2 M,$$

где M — число различных состояний сигнала.

Если сигнал имеет два состояния, то пропускная способность равна удвоенному значению ширины полосы пропускания линии связи.

Если же передатчик использует более чем 2 устойчивых состояния сигнала для кодирования данных, то пропускная способность линии повышается, так как за один такт работы передатчик передает несколько битов исходных данных.

Так, если полоса пропускания телефонной сети равна 3100 Гц, а $M = 3$, то согласно формуле Найквиста максимальная скорость передачи данных будет равна

$$\bar{I}_{\text{max}} = C = 2B \log_2 M = 2 \cdot 3100 \cdot \log_2 8 = 18\,600 \text{ бит/с}.$$

Хотя формула Найквиста явно не учитывает наличие шума, косвенно его влияние отражается в выборе числа состояний информативного параметра сигнала. Для повышения пропускной способности канала хотелось бы существенно увеличить это число, но на практике этого сделать нельзя из-за шума на линии. Поэтому число возможных состояний сигнала фактически ограничивается соотношением мощности сигнала и шума, а формула Найквиста определяет предельную скорость передачи данных в том случае, когда число состояний уже выбрано с учетом возможностей устойчивого распознавания приемником.

Помехозащищенность

Прохождение сигнала по КС сопровождается не только его искажениями и воздействием помех. При каждой передаче данных получаемый сигнал x_r^i состоит из передаваемого сигнала x_t , искаженного передающей системой, а также дополнительных нежелательных сигналов, введенных где-либо между передатчиком и приемником. Нежелательные сигналы называются *помехами* или *шумами*. Помехи являются основным фактором, ограничивающим пропускную способность системы передачи информации. Их можно разделить на две категории — *предсказуемые* помехи, имеющие относительно постоянную величину, что дает возможность бороться с ними, и *случайные*, к которым относятся *импульсные помехи*, состоящие из нерегулярных импульсов или всплесков помех небольшой продолжительности и имеющие относительно высокую амплитуду. Причин возникновения таких помех несколько: это могут быть любые повреждения или дефекты систем связи либо же внешние электромагнитные возмущения, например молния.

В общем случае на аналоговые данные импульсные помехи не оказывают значительного воздействия. При передаче речи, например, может искажаться краткими шелчками или треском, оставаясь при этом разборчивой. Импульсные помехи являются основным источником ошибок при цифровой передаче данных. Ошибкой считается прием 1 при переданном 0, и наоборот. Например, узкий энергетический импульс продолжительностью 0,01 с не сможет разрушить звуковой сигнал, но сделает бесполезными 560 бит данных, передаваемых со скоростью 56 Кбит/с. Пример воздействия помехи на цифровой сигнал приведен на рис. 3.53, из которого видно, что случайного шума (случайные всплески импульсных помех) достаточно для изменения 1 на 0 или 0 на 1.

Помехоустойчивость линии определяет ее способность уменьшать уровень помех, создаваемых внешней средой, на внутренних проводниках.

Достоверность передачи данных характеризует вероятность искажения для каждого передаваемого бита данных.

Искажения передаваемых битов информации происходят как из-за помех на линии, так и в результате искажений формы сигнала, ограниченной полосой пропускания линии. Поэтому для повышения достоверности передаваемых данных нужно повышать степень помехозащищенности линии, снижать уровень перекрестных наводок в кабеле, а также использовать широкополосные линии связи.

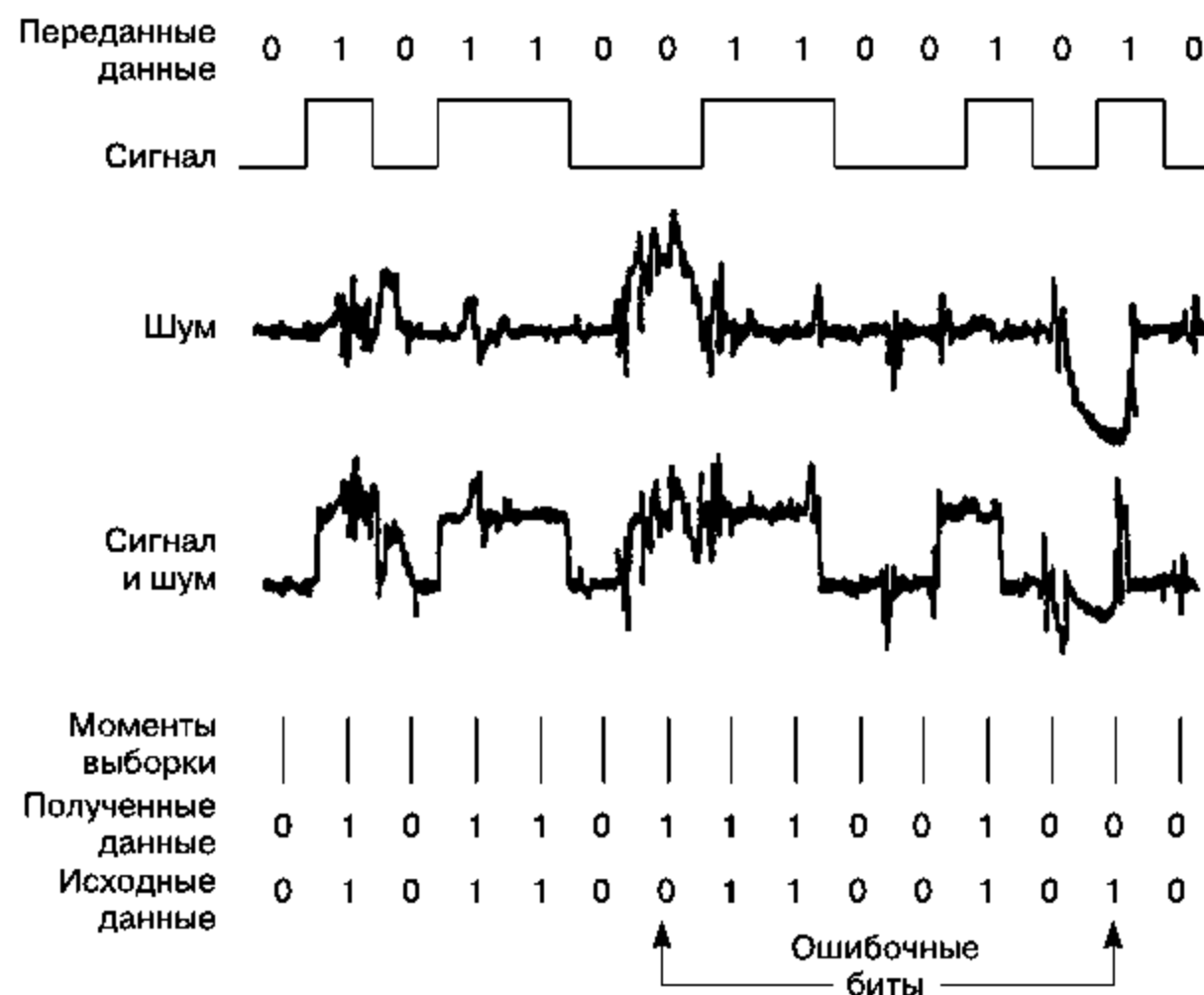


Рис. 3.53. Искажение цифрового сигнала под воздействием помехи

Перекрестные помехи — это нежелательное соединение каналов связи, когда электромагнитное поле сигнала, передаваемого с выхода передатчика по одной паре проводников, наводит на другую пару проводников сигнал помехи. Если ко второй паре будет подключен приемник, то он может принять наведенную внутреннюю помеху за полезный сигнал.

Достоверность передачи информации

Достоверность каналов связи без дополнительных средств защиты от ошибок, например самокорректирующихся кодов, составляет, как правило, 10^{-4} — 10^{-6} , а для оптоволоконных линий связи — 10^{-9} . Значение достоверности передаваемых данных, например 10^{-4} , говорит о том, что в среднем из 10 000 бит искажается значение одного бита.

На выходе каналов связи сигналы демодулируются и декодируются с целью перевода передаваемого сигнала в форму, приемлемую для получателя, т.е. в форму принятого сообщения.

Демодулятор и декодер входят в состав приемника. Поскольку передача сигнала по каналам связи сопровождается воздействиями иска-

жений и помех, основной функцией приемника является *распознавание принимаемого сигнала* в переданном колебании, после чего он преобразуется в сообщение. В современных цифровых системах связи основные функции передатчика и приемника выполняет *модем* — электронное устройство, которое преобразует цифровые сигналы в аналоговые, и наоборот.

Проблемы, связанные с *передачей данных*, заключаются в следующем. Средства связи стоят дорого, и в общем случае чем шире их полоса, тем они дороже. Более того, все каналы передачи, представляющие практический интерес, имеют ограниченную ширину полосы, что обусловлено либо физическими свойствами передающей среды, либо преднамеренным ограничением ширины полосы в самом передатчике. Эффективное использование имеющейся полосы при передаче цифровых сигналов означает обеспечение максимально возможной скорости передачи данных при существующем уровне ошибок. Главным ограничением при достижении такой эффективности являются помехи, ухудшающие качество сигнала, искажающие или повреждающие его.

В общем случае любой цифровой сигнал имеет бесконечную ширину полосы пропускания и при попытке передать этот сигнал передающая система наложит ограничения на ширину полосы, которую можно передать. Поэтому, с *одной стороны*, экономические и практические соображения подсказывают аппроксимировать цифровую информацию сигналом с ограниченной шириной полосы. С *другой стороны*, ограничение ширины полосы приводит к искажениям, которые затрудняют задачу интерпретации сигнала, поступившего в приемник. Существенными факторами, определяющими успешную интерпретацию сигнала, являются:

- ◆ отношение сигнал/шум;
- ◆ скорость передачи данных;
- ◆ ширина полосы.

При фиксированных основных параметрах справедливы следующие утверждения:

- ◆ увеличение скорости передачи данных приводит к увеличению скорости появления ошибочных битов или частоты битовых ошибок (определяется как вероятность приема ошибочного бита);
- ◆ увеличение отношения сигнал/шум уменьшает скорость появления ошибочных битов;
- ◆ увеличение ширины полосы позволяет увеличить скорость передачи данных.

3.8.2. Оценка основных характеристик дискретных каналов без шума

В дискретных каналах (рис. 3.54) на выходе передатчика и входе приемника действует дискретный по информативному параметру сигнал. С помощью этого сигнала можно закодировать конечное множество возможных сообщений. Таким образом, чем больше число возможных сообщений и, следовательно, чем больше возможных значений сигнала, тем больше априорная неопределенность и тем большее количество информации получает адресат, когда эта неопределенность снимается.

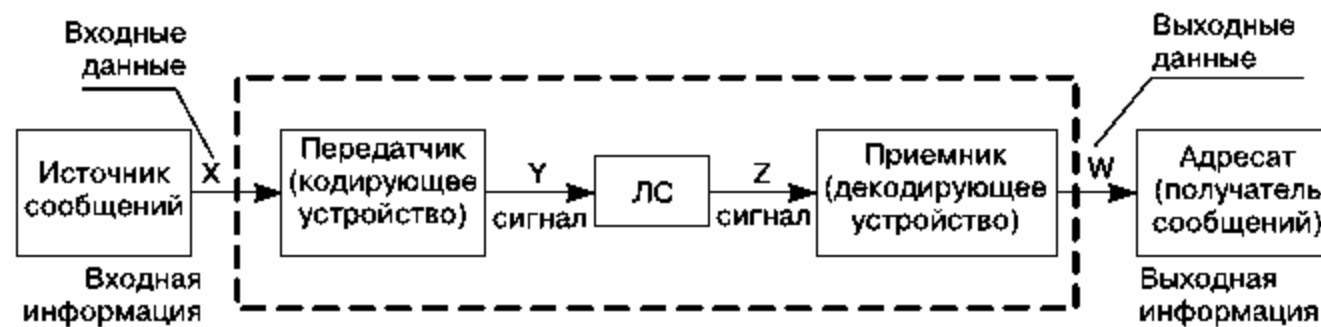


Рис. 3.54. Модель дискретного канала без шума

Если $X = \{x_1, x_2, \dots, x_m\}$, под которым можно понимать информативный параметр сигнала, а $x_1, x_2, \dots, x_m$ — возможные исходы с вероятностями $P_1, P_2, \dots, P_m$, неопределенность каждого исхода будет равна $-\log P_1, -\log P_2, \dots, -\log P_m$, а математическое ожидание дает количественную оценку неопределенности — *энтропию*, которая по оценке К. Шеннона, усреднившего неопределенности по всем m исходам, равна

$$H(X) = -\sum_{i=1}^m P_i \log P_i. \quad (3.30)$$

Понятие энтропии тесно связано с понятием *количества информации*, под которым понимается мера снятия неопределенности в процессе получения сигнала.

Пусть в модели, показанной на рис. 3.54, X — последовательность сигналов передаваемых сообщений, а W — последовательность сигналов принимаемых сообщений, в которых будет содержаться информация о передаваемых сообщениях X , количество которой обозначим через $I(W, X)$.

Тогда количество информации на выходе ЛС можно обозначить через $I(Z, Y)$, приняв за Y и Z последовательность сигналов на ее входе и выходе соответственно.

При отсутствии помех в ЛС связь между входом и выходом будет однозначной, и сигналы на выходе будут полностью соответствовать сигналам на входе ($Z = Y$).

Если кодер и декодер не вносят искажений, то сигналы W на выходе канала соответствуют сигналам на его входе, т.е. $W = X$, и тогда количество информации на приемном конце канала $I(W, X) = I(X, X) = H(X)$. По аналогии на выходе линии связи количество информации $I(Z, Y) = I(Y, Y) = H(Y)$.

Скорость передачи (бит/с) информации в канале без шумов и в ЛС будет определяться соответственно выражениями

$$\bar{I}(W, X) = \lim_{T \rightarrow \infty} \frac{H(X)}{T},$$

$$\bar{I}(Z, Y) = \lim_{T \rightarrow \infty} \frac{H(Y)}{T},$$

а пропускная способность (бит/с) — выражениями:

$$C_k = \bar{I}(W, X)_{\max} = \lim_{T \rightarrow \infty} \frac{H(X)_{\max}}{T},$$

$$C_{\text{ЛС}} = \bar{I}(Z, Y)_{\max} = \lim_{T \rightarrow \infty} \frac{H(Y)_{\max}}{T}.$$

При наличии помех степень неопределенности на приемном конце не будет снята полностью, так как необходимое для этого количество информации будет ограничено помехами, вследствие чего количество информации, полученной адресатом, будет равно

$$I'(W, X) = H(X) - H\left(\frac{Y}{Z}\right),$$

где $H\left(\frac{Y}{Z}\right)$ — величина, характеризующая интенсивность действия помех.

Энтропия $H(X)$ характеризует меру неопределенности информации до приема, а $I'(W, X)$ — меру снятия неопределенности после приема информации. Если неопределенность снята полностью, т.е. $H\left(\frac{Y}{Z}\right) = 0$, то $I'(W, X) = H(X)$.

3.8.3. Примеры вычислений скорости передачи и пропускной способности канала без шума

Предположим, что по каналу передаются M статистически независимых сообщений, длительность каждого из которых равна τ_c .

Тогда

$$C_k = \lim_{T \rightarrow \infty} \frac{H(X)_{\max}}{T} = \frac{M H(X)_{\max}}{M \tau_c} = \frac{H(X)_{\max}}{\tau_c}, \quad (3.31)$$

где $H(X)$ — энтропия одного сообщения.

Допустим, что $M = 4$, каждое из сообщений кодируется 2-разрядным двоичным кодом: $x_1 = 00$, $x_2 = 01$, $x_3 = 10$, $x_4 = 11$, вероятность сообщений одинакова: $P(x_1) = P(x_2) = P(x_3) = P(x_4) = 0,25$; а длительность импульса сообщения $\tau = 1,25$ мс (рис. 3.55).

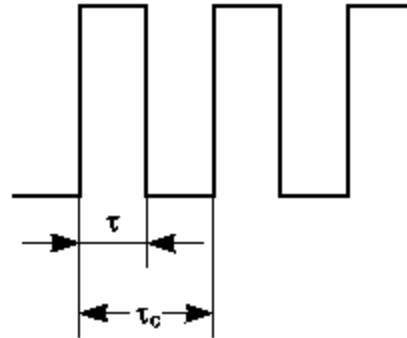


Рис. 3.55. Временные параметры сообщения

Тогда энтропия

$$H(x) = \sum_{i=1}^4 P(x_i) \log \frac{1}{P(x_i)} = \left(0,25 \log \frac{1}{0,25} \right) \cdot 4 = 2 \text{ бит/сообщение},$$

а скорость передачи сообщений

$$\bar{I} = \frac{H(x)}{\tau_c} = \frac{2}{2\tau} = \frac{1}{\tau} = \frac{1}{1,25 \cdot 10^{-3}} = 800 \text{ бит/с}.$$

Так как при выбранном способе кодирования вероятности символов 0 и 1 одинаковы, т.е. $P(0) = P(1)$, пропускная способность канала

$$C_k = (\bar{I})_{\max} = 800 \text{ бит/с}.$$

Для тех же четырех сообщений, но с *разными* вероятностями: $P(x_1) = 0,5$, $P(x_2) = 0,25$, $P(x_3) = P(x_4) = 0,125$, энтропия равна

$$\begin{aligned} H(x) &= \sum_{i=1}^4 P(x_i) \log \frac{1}{P(x_i)} = \\ &= 0,5 + 0,25 \cdot 2 + (0,125 \cdot 3) \cdot 2 = 1,75 \text{ бит/сообщение}, \end{aligned}$$

а скорость передачи сообщений

$$\bar{I} = \frac{H(x)}{\tau_c} = \frac{1,75}{2\tau} = \frac{1}{\tau} \cdot 0,875 = 700 \text{ бит/с}.$$

То есть при неравных вероятностях $\bar{I} < C_k$ ($700 < 800$) и возможности канала *недоиспользуются*. Источник сообщений оказывается статистически несогласованным с каналом связи ($P(0) \neq P(1)$, а точнее, $P(0) > P(1)$).

3.9. СОВРЕМЕННЫЕ ТЕХНИЧЕСКИЕ СРЕДСТВА ОБМЕНА ДАННЫХ И КАНАЛООБРАЗУЮЩЕЙ АППАРАТУРЫ

3.9.1. Интерфейсные устройства

В процессе передачи информации от источника к приемнику постоянно происходит обмен информацией между различными устройствами системы с помощью так называемого *интерфейса*, под которым будем понимать совокупность аппаратных, программных и конструкторских средств, обеспечивающих обмен информацией между устройствами системы. Это определение включает три составляющие: аппаратную, программную и конструкторскую. К *аппаратной* относят отдельные узлы системы, через которые осуществляется обмен, к *программной* — строгий порядок, алгоритм взаимодействия устройств, реализованный в виде специальных программ. Порядок следования информационных и служебных сигналов в процессе обмена информацией называют *протоколом* обмена. *Конструкторскую* составляющую образуют линии связи, объединенные в шины, по которым осуществляется передача информации.

В зависимости от удаленности источника и приемника информации используют либо *параллельный* способ передачи информации (*параллельный интерфейс*), либо *последовательный* (*последовательный интерфейс*).

При *параллельной* передаче информация передается по шинам данных (магистралям), состоящим из n параллельных проводников, причем все разряды слова данных передаются одновременно. Такой способ передачи используется, если длина линии связи не превышает полутора-пяти метров.

При *последовательной* передаче информация передается по одной линии связи бит за битом. Для этого необходимо преобразование данных из параллельного кода в последовательный (при передаче) и обратное преобразование из последовательного кода в параллельный (при приеме). Расстояние, на которое может передаваться информация последовательным способом, не ограничивается и определяется лишь мощ-

ностью передатчика. В частности, во всех информационных сетях передача осуществляется в последовательном коде.

Существует два основных способа обмена информацией: синхронный и асинхронный. При *синхронном обмене* темп выдачи информации определяет источник, который сопровождает выдачу импульсами синхронизации. Приемник информации при этом должен принимать данные в темпе, задаваемом источником. Если приемник не успел принять информацию, она будет потеряна, так как между источником и приемником нет обратной связи. Поэтому темп передачи данных должен учитывать быстродействие приемника и рассчитывается на наихудший случай.

При *асинхронном способе* обмена между источником и приемником существует обратная связь. В основе этого способа лежит *метод квитирования*, сущность которого состоит в том, что источник при каждой посылке данных должен получать от приемника подтверждение о том, что данные приняты, т.е. своеобразную «квитанцию». Сигналы, обеспечивающие такой диалог между источником и приемником, называются *сигналами квитирования*. При посылке данных источник сообщает об этом приемнику сигналом «Данные выданы». Приемник, получив этот сигнал, считывает слово данных и выдает источнику сигнал «Данные приняты». Только после получения этого сигнала источник приступает к посылке следующего слова данных. При таком обмене интервал времени приема-передачи данных будет переменным в зависимости от быстродействия источника и приемника и длины линии связи.

3.9.2. Сопряжение персональных компьютеров

Большинство цифровых устройств обработки данных (терминал, компьютер) генерируют простой цифровой сигнал в виде периодической последовательности импульсов двух различных уровней напряжения для представления двоичного нуля и двоичной единицы. Например, отсутствие напряжения — для представления 0, а постоянное положительное напряжение — для представления двоичной 1. Передающие возможности таких устройств ограничены, данные могут передаваться только на малые расстояния, поэтому такие устройства не могут напрямую связываться с ЛС или работать в сети. Они соединяются с передающей системой через специальные средства передачи данных, примером которых может служить *модем*.

Типовая схема сопряжения цифрового устройства и ее типичная конфигурация приведены на рис. 3.56.

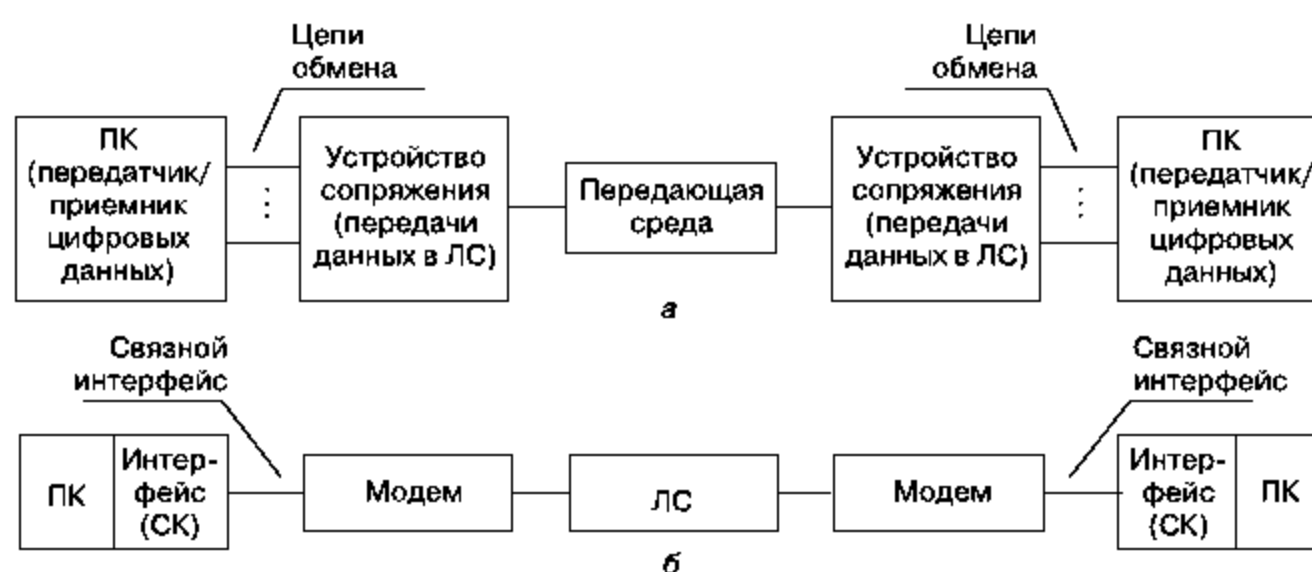


Рис. 3.56. Сопряжение ПК:

а — типовая схема сопряжения с передающей средой;
 б — типичная конфигурация

С одной стороны, устройство сопряжения отвечает за побитовый прием/передачу в передающей среде, с другой — обеспечивает взаимодействие с ПК, что в общем случае требует как обмена данными, так и управления информацией. Такой обмен осуществляется по набору проводников, называемых *цепями обмена*.

Передатчик и приемник, обменивающиеся информацией через ЛС, должны хорошо «понимать» друг друга, т.е. в приемниках должны использоваться те же коды и скорости передачи, что и в передатчиках. Чтобы облегчить работу пользователей и производителей устройств обработки данных, разработаны *стандарты*, задающие конкретный интерфейс между устройством обработки данных (ПК) и устройством передачи данных (модем) и имеющие четыре группы характеристик:

- ◆ механические;
- ◆ электрические;
- ◆ функциональные;
- ◆ процедурные.

Механические характеристики описывают реальные физические соединения этих устройств (кабели, разъемы), *электрические* — обмен сигналами (двоичными, управляющими), уровнями их напряжений, длительностью, скоростью передачи. *Функциональные характеристики* определяют функции цепей обмена — цепей данных, управления, синхронизации, «земли», а также дополнительных цепей (при работе устройств в дуплексном или полудуплексном режимах) и др. *Процедурная спецификация* определяет последовательность использования различных цепей в конкретных системах. Простейшая из таких систем — соединение двух

ПК, находящихся на небольшом расстоянии в пределах одного здания. Устройство, обеспечивающее подобное соединение, называется *модемом для линий ограниченной протяженности* (LDM — limited distance modem). Данный модем получает цифровые сигналы от ПК, преобразует их в аналоговые, после чего передает по небольшому отрезку передающей среды, например витой пары. На другом конце линии находится другой модем, который принимает аналоговые сигналы, преобразует их в цифровые и передает другому ПК.

Для передачи данных по телефонной сети потребуются дополнительные цепи.

3.9.3. Использование модема в качестве устройства передачи данных в ЛС

Для соединения ПК и модемов телефонной сети широко применяется интерфейс V.24/V.28 RS-232C — рекомендованный стандарт (RS — Recommended Standard), определяющий последовательный обмен двоичными данными со скоростью передачи до 20 000 бит/с между ПК и модемом. При скоростях передачи от 48 до 168 Кбит/с необходимы широкополосные модемы, работающие с интерфейсом V.35.

Заметим, однако, что при передаче данных между компьютерами круг проблем значительно шире, чем описан выше. Кроме наличия канала передачи данных между ними (прямого или через сеть связи) требуется выполнить еще ряд задач, таких как:

- ♦ активизацию источником прямого канала передачи данных (вызов через сеть устройства-адресата);
- ♦ проверку источником готовности адресата принять данные и др.

Таким образом, чтобы два объекта могли успешно общаться, между ними все должно быть взаимно согласовано: какие данные передаются, каким образом и когда. Источник должен быть уверен, что все данные получены адресатом и пришли в том порядке, в каком были отправлены. Принимаемые при этом соглашения называются *протоколами*, представляющими собой набор правил, контролирующих обмен данными между двумя объектами, в частности между двумя ПК.

Совокупность двух или нескольких компьютеров, взаимодействующих через сеть связи, называется *сетью компьютеров*. При рассмотрении компьютерных сетей понятие протокола является ключевым.

В современных системах связи основные функции передатчика и приемника выполняет устройство, называемое *модемом*.

Модемы разделяются на *аналоговые* и *цифровые*. Сегодня чаще применяются аналоговые, используемые для передачи данных между

удаленными ПК по телефонной ЛС. Модем имеет два интерфейса. Один интерфейс для аналоговой линии связи, второй — многопроводной цифровой интерфейс V.24/V.28 (RS-232C) для связи с ПК.

Цифровые данные с выхода ПК представляют собой серию двоичных импульсов. Аппаратные средства ПК, называемые связным контроллером (СК), преобразуют данные из параллельной формы представления в последовательную, и наоборот, поскольку ЛС работает в последовательном режиме. Затем последовательность двоичных импульсов преобразуется в аналоговые сигналы с помощью *модема*, кодирующего цифровые данные на несущей частоте. Получившийся в результате сигнал занимает определенный спектр частот с центром на несущей частоте и может распространяться в среде, подходящей для этой несущей частоты.

Наиболее распространенные модемы представляют цифровые данные в спектре частот, совпадающем со спектром речи, позволяя, таким образом, передавать эти данные по обычным голосовым телефонным линиям. На другом конце линии другой модем демодулирует сигнал и восстанавливает исходные данные (рис. 3.57).

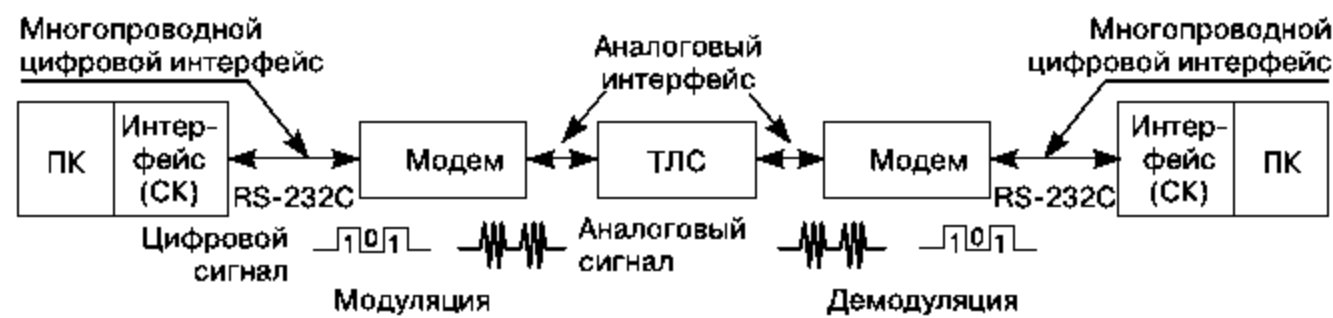


Рис. 3.57. Двухточечный канал связи ПК

Таким образом, *при передаче* данных от ПК в линию связи модем преобразует цифровой код (широкополосные импульсы) в аналоговый сигнал (узкополосные). *При приеме* из ЛС в ПК выполняется обратное преобразование узкополосного аналогового сигнала в цифровой.

Преобразование, выполняемое при передаче данных, связано с *модуляцией*. В современных модемах чаще всего используется 3 вида модуляции:

- ♦ частотная (FSK — Frequency Shift Keying);
- ♦ фазовая (PSK — Phase Shift Keying);
- ♦ квадратурная амплитудная (QAM — Quadrature Amplitude Modulation).

Поскольку при чисто амплитудной модуляции сигнала (рис. 3.58) его защищенность от помех крайне низкая (при передаче искажается

обычно лишь амплитуда сигнала), применяют более помехоустойчивую, но и более сложную *квадратурную амплитудную модуляцию*, при которой в такт передаваемым данным изменяются одновременно и фаза, и амплитуда сигнала.

Помехоустойчивость ЧМ- и ФМ-модулированных сигналов достаточно высокая. При частотной модуляции в соответствии с текущими значениями модулирующего сигнала (передаваемых данных) изменяется частота физического сигнала (обычно синусоидального) при неизменной амплитуде (см. рис. 3.58). В простейшем случае значениям 1 и 0 данных соответствуют два значения частот, например 980 и 1180 Гц, как было принято в одном из первых протоколов передачи данных (см. рис. 3.58).

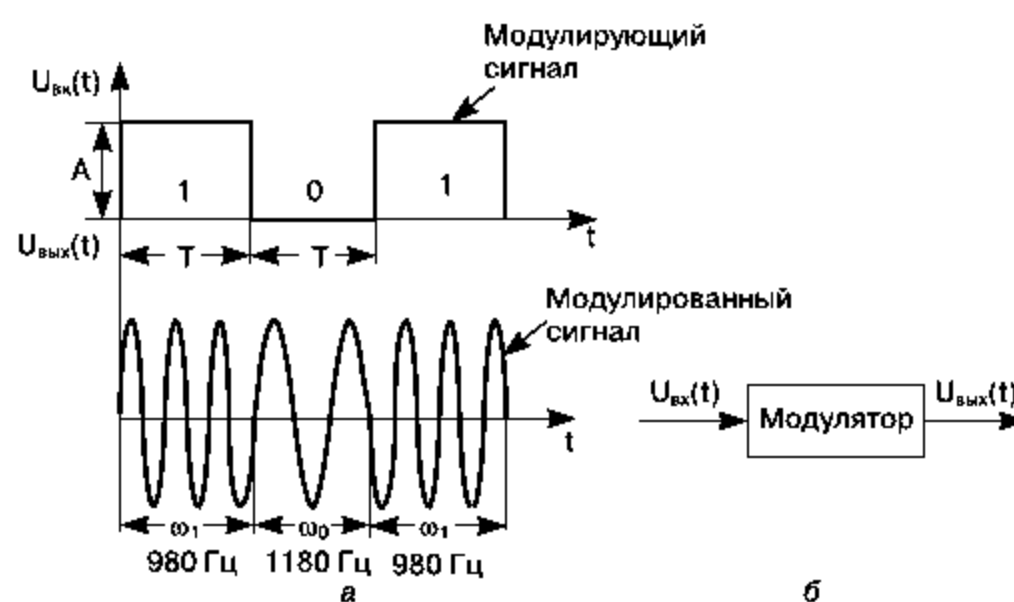


Рис. 3.58. Схема модулирования:

а — частотная модуляция; б — представление модулятора

Модулятор подключается к ПК через последовательный интерфейс RS-232C, через который осуществляется ввод-вывод данных в модем в последовательном формате (1 бит за 1 такт). Данные передаются в ЛС и принимаются модемом последовательно бит за битом по двум телефонным проводам, изображенным на логическом уровне одной линией. Физической средой передачи информации в низкоскоростных и среднескоростных ЛС является пара либо параллельных проводов, либо скрученных (витая пара). *Витая пара* уменьшает перекрестные наводки между проводниками, характеризуется меньшим затуханием сигнала при передаче на высоких частотах и меньшей чувствительностью к электромагнитным наводкам, чем параллельная пара проводов.

Передача данных и их преобразование выполняются в соответствии с протоколами, регламентирующими формат данных и процедуры

их передачи в КС, в частности подробно указывающих, как представить данные, какой способ модуляции выбрать, чтобы увеличить скорость передачи и их защищенность, как выполнить соединение с каналом, как обеспечить достоверность и др.

Современные протоколы основаны на использовании корректирующих кодов с обнаружением и исправлением ошибок, предусматривают внедрение в модемы функции *сжатия* данных, основанные на использовании алгоритма Хаффмена.

Современные модемы могут автоматически перестраиваться на работу с менее скоростными модемами, при передаче информации по ТЛС обеспечивают автоматическую коррекцию ошибок и компрессию данных. Операции коррекции, компрессии и декомпрессии позволяют резко повысить качество связи и скорость передачи информации.

3.10. СЕТЕВЫЕ ТЕХНОЛОГИИ ПЕРЕДАЧИ ДАННЫХ

3.10.1. Определение и примеры

Сетевая технология — это согласованный набор стандартных протоколов и реализующих их программно-аппаратных средств (например, сетевых адаптеров, драйверов, кабелей, разъемов), достаточный для построения вычислительной сети. Примерами базовых сетевых технологий (стандартных технологий объединения компьютеров в сеть) являются известные технологии локальных сетей Ethernet, Token Ring, FDDI или же технологии территориальных сетей X.25, Frame relay.

Для получения работоспособной сети в этом случае достаточно приобрести программные и аппаратные средства, относящиеся к одной базовой технологии — сетевые адаптеры с драйверами, концентраторы, коммутаторы, кабельную систему и т.п., — и соединить их в соответствии с требованиями стандарта на данную технологию.

Сети Ethernet, имеющие фиксированную топологию «общая шина», являются популярными благодаря своей экономичности. Для построения сети достаточно иметь по одному сетевому адаптеру для каждого компьютера, плюс один физический сегмент коаксиального кабеля нужной длины. В других технологиях, например в Token Ring, для создания даже небольшой сети требуется дополнительное устройство — концентратор. Достоинством сетей Ethernet является, кроме того, хорошая расширяемость, в них реализованы достаточно простые алгоритмы доступа к среде, адресации и передачи данных. Сетевые адаптеры и их драйверы просты, дешевы и поэтому обладают высокой надежностью.

3.10.2. Об использовании сетевых технологий

Передача данных между устройствами, которые соединены непосредственно, как правило, двухточечным каналом, часто оказывается непрактичной либо из-за их большой удаленности друг от друга, либо из-за большого числа соединяемых устройств. Проложить *прямой канал* между двумя устройствами, разделенными тысячами километров, было бы чрезвычайно *дорого*. Если же есть ряд компьютеров, каждому из которых в разное время требуются каналы связи со многими другими устройствами, то предоставлять каждой паре устройств отдельный канал связи непрактично. Решить возникшую проблему можно, если в качестве передающей системы в модели, показанной на рис. 3.32, использовать *локальную* или *глобальную* сеть (рис. 3.59).

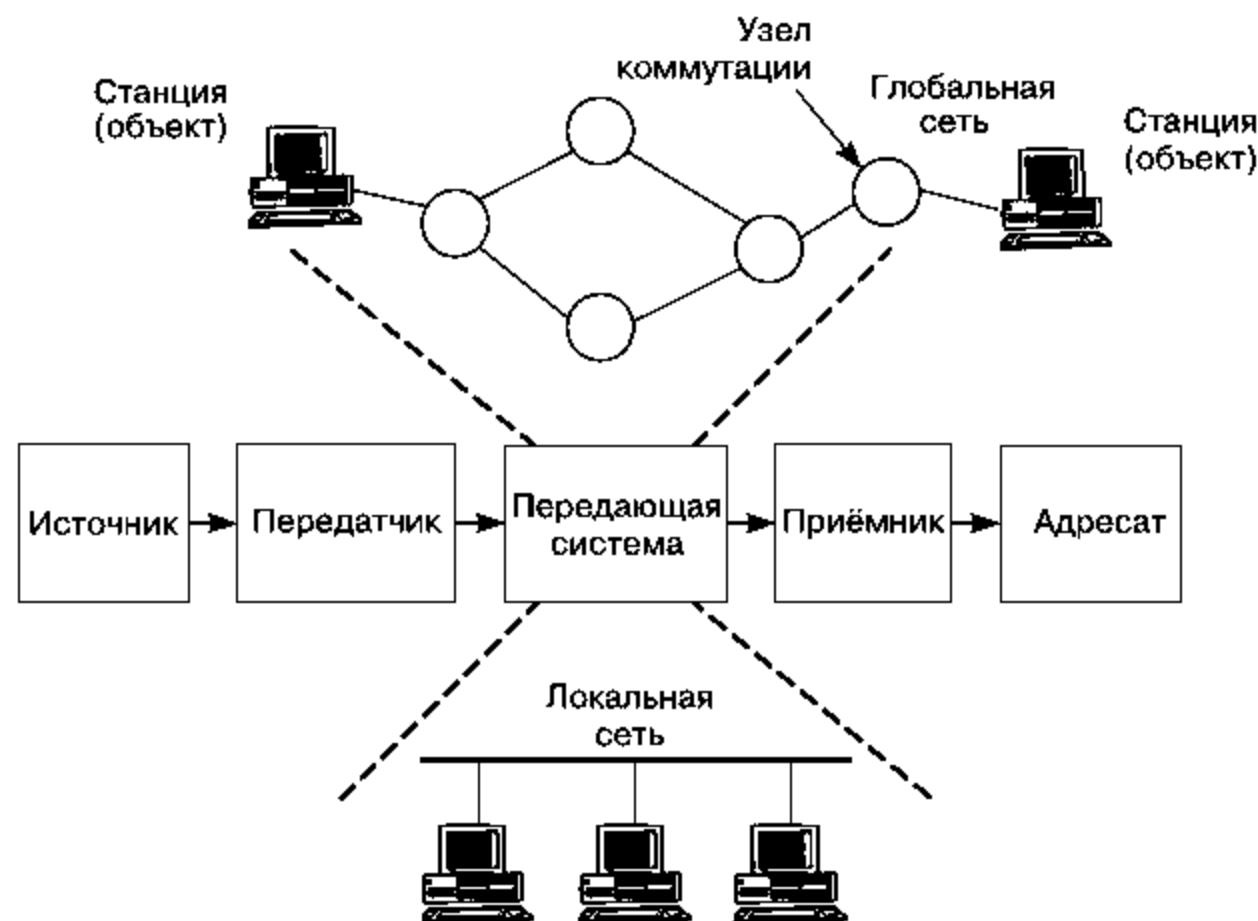


Рис. 3.59. Простейшие модели сетей

Конечные устройства (объекты), которые должны быть соединены, назовем *станциями* (компьютеры, терминалы, телефоны и др.). Коммутирующие устройства, служащие для обеспечения связи, будем называть *узлами*, а совокупность узлов — *сетью связи*. Узлы связываются между собой с помощью *каналов передачи*.

И *локальная*, и *глобальная* сети представляют собой сеть связи, которая соединяет ряд устройств и предоставляет средства для обмена информацией между ними.

3.10.3. Глобальные сети

Глобальные сети (Wide Area Networks (WAN)), как правило, охватывают большие географические области. Обычно глобальная сеть состоит из некоторого числа взаимосвязанных коммутирующих узлов. Сообщение, передаваемое любым устройством, определенным путем проходит через внутренние узлы к заданному адресату. Эти узлы (в том числе и граничные) не затрагивают содержание передаваемых данных, их назначение — просто перемещать данные от узла к узлу, пока не будет достигнут адресат.

Традиционно глобальные сети создавались на базе одной из двух технологий — *коммутации каналов* или *коммутации пакетов*, — отличающихся способом, которым узлы передают информацию от одного канала к другому на пути от источника к адресату. В более современных сетях основная роль отводится технологиям Frame Relay (ретрансляция кадров) и АТМ (Asynchronous Transfer Mode), называемой еще технологией ретрансляции ячеек (cell relay). В случае АТМ поток информации в каждом логическом соединении разделяется на *пакеты* (сообщения сетевого уровня) фиксированной длины, называемые *ячейками*, что упрощает обработку на каждом узле АТМ, а также облегчает использование АТМ при высоких скоростях передачи.

Метод *коммутации каналов* был разработан для обслуживания телефонного обмена, но теперь он используется и при обмене данными. Связь с коммутацией каналов подразумевает, что имеется заранее определенный тракт связи между двумя станциями (связанная последовательность каналов между узлами сети), т.е. тракт соединения устанавливается до того, как начинается передача данных. В каждом физическом канале для данного соединения выделяется логический канал. Данные, генерируемые исходным устройством, передаются по выделенному маршруту с максимально возможной скоростью. В каждом узле входящие данные немедленно перенаправляются или коммутируются в подходящий выходной канал. Связь с коммутацией каналов включает три фазы:

- 1) *установка соединения* (прежде чем передать какие-либо сигналы, необходимо установить сквозное соединение от станции к станции);
- 2) *передача данных* по сети от станции к станции. В зависимости от природы сети данные могут быть аналоговыми и цифровыми. Основным методом и для речи, и для данных становится цифровая двоичная передача;
- 3) *разрыв соединения*. После некоторого периода передачи данных соединение завершается обычно действием одной из двух стан-

ций. Соответствующим узлам должны быть переданы сигналы на освобождение выделенных ресурсов.

В сети с *коммутацией пакетов* данные передаются в виде последовательности небольших порций, или *пакетов*. Каждый пакет, длина которого обычно ограничена 1000 байтами, содержит часть данных пользователя или все данные, если сообщение невелико, и некоторую управляющую информацию, включающую как минимум сведения, необходимые для передачи пакета по сети к нужному получателю. Каждый пакет проходит сеть от узла к узлу по некоторому маршруту, идущему от источника к адресату. Каждый узел принимает пакет в полном объеме, временно фиксирует его в памяти и передает следующему узлу. Сети с *коммутацией пакетов* обычно используются при передаче данных между двумя компьютерами или между терминалом и компьютером. Первоначальные сети были рассчитаны на скорость передачи данных конечному пользователю примерно 64 Кбит/с.

Технология Frame Relay обеспечивает более высокие скорости передачи данных (до 2 Мбит/с).

Использование пакетов фиксированной длины (ячеек), т.е. технологии АТМ, позволило увеличить скорость передачи данных даже по сравнению с методом *Frame Relay* до десятков — сотен мегабит и даже нескольких гигабит в секунду.

3.10.4. Локальные сети

Локальная сеть (Local Area Network (LAN)) — это коммуникационная система, поддерживающая в пределах здания или некоторой другой ограниченной территории один или несколько высокоскоростных каналов передачи цифровой информации, предоставляемой подключенным устройствам для кратковременного монопольного использования.

Известны два основных класса локальных сетей (ЛС):

- 1) локальные вычислительные сети (ЛВС);
- 2) локальные информационно-вычислительные сети.

Первые решают задачи рационального использования ресурсов ЭВМ (диск, лазерный принтер, общекорпоративные файлы данных, например относящиеся к бухгалтеру, и другие файлы, к которым должны иметь доступ многие пользователи), распределенных на небольшой территории, в автоматизированных системах научных исследований, САПР и других системах, требующих доступа в реальном масштабе к локально-распределенным вычислительным ресурсам.

Вторые в основном решают задачи информационного обслуживания пользователей, реализуя функции доступа к удаленным файлам, электронной почты, передачи и обработки текстов и изображений.

Технология, на основе которой созданы все локальные сети, определяется *выбором топологии* (конфигурации физических связей компьютеров), *среды передачи* (витая пара, коаксиальный кабель, оптоволокно и др.) и *метода управления доступом к среде передачи данных*.



Рис. 3.60. Топологии локальных сетей:

а — звездообразная структура; б — общая шина; в — кольцевая ЛС
(HUB — центральное устройство; WS — рабочая станция)

Топология сети определяет многие ее характеристики — надежность, простоту присоединения новых узлов, т.е. расширяемость сети, пропускную способность и др. По топологическим признакам локальные сети можно представить тремя базовыми топологиями: *звезда*, *общая шина*, *кольцо* (рис. 3.60). В *звездообразной структуре* (рис. 3.60, а) каждый компьютер подключается отдельным кабелем к общему центральному устройству (HUB), называемому концентратором. Функция концентратора — направлять передаваемую информацию одному или всем узлам сети. Компьютеры, подключенные к сети, часто называют рабочими станциями (WS — work station) или узлами сети. *Основным преимуществом* такой топологии является высокая надежность. Любые неприятности с кабелем касаются лишь того компьютера, к которому этот кабель присоединен и только неисправность HUB может вывести из строя всю сеть. *Недостатком* топологии «звезда» является довольно высокая стоимость сетевого оборудования из-за необходимости приобретения концентратора. Кроме того, возможности наращивания узлов в сети ограничиваются числом портов концентратора.

В топологии «общая шина» (рис. 3.60, б) в качестве центрального элемента выступает пассивный кабель, к которому подключаются несколько компьютеров. Передаваемая по кабелю информация доступна сразу всем подсоединенным к нему абонентам. Производительность схемы невысока, а самый серьезный *недостаток* — низкая надежность. *Преимущества* схемы — дешевизна и простота наращивания.

Кольцевая ЛС (рис. 3.60, в) состоит из совокупности замкнутых в контур узлов. Данные циркулируют по кольцу от одного компьютера

к другому. Если компьютер распознает данные как «свои», то он копирует их себе во внутренний буфер. *Достоинство* кольца — возможность организации обратной связи — данные, сделав полный оборот, возвращаются к узлу-источнику. Поэтому этот узел может контролировать процесс доставки данных адресату. В то же время в сетях с кольцевой топологией необходимо принимать специальные меры, чтобы в случае выхода из строя или отключения какой-либо станции не прерывался канал связи между остальными станциями кольца.

Все сети независимо от топологии и реализуемого метода доступа к среде передачи данных имеют некоторые общие компоненты, функции и характеристики. В их числе:

- ◆ серверы (server) — компьютеры, предоставляющие свои ресурсы сетевым пользователям;
- ◆ среда (media) — способ соединения компьютеров;
- ◆ совместно используемые данные — файлы, предоставляемые серверами по сети;
- ◆ совместно используемые периферийные устройства (принтеры, библиотеки CD-ROM и т.д.) — ресурсы, предоставляемые серверами;
- ◆ ресурсы — файлы, принтеры и другие элементы, используемые в сети.

По принципу предоставления ресурсов в совместное использование ЛС разделяются:

- ◆ на одноранговые;
- ◆ на основе сервера;
- ◆ на комбинированные.

Различия между ними имеют принципиальное значение, поскольку определяют разные возможности этих типов сетей.

Одноранговые сети относительно просты. Все компьютеры равноправны, иначе говоря, нет отдельного компьютера, ответственного за администрирование всей сети.

Одноранговая сеть характеризуется рядом стандартных решений:

- ◆ компьютеры расположены на рабочих столах пользователей;
- ◆ пользователи сами выступают в роли администраторов и обеспечивают защиту информации;
- ◆ для объединения компьютеров в сеть применяется простая кабельная система.

Одноранговую сеть удобно использовать в коллективах до 10 человек, где пользователи расположены компактно, вопросы защиты данных не критичны и в обозримом будущем не ожидается значительного расширения сети.

Вместе с тем следует учитывать, что, поскольку в такой сети каждый компьютер функционирует и как клиент, и как сервер, пользователи должны обладать достаточным уровнем знаний, чтобы работать и как пользователи, и как администраторы своего компьютера.

Сети на основе сервера. Технология «клиент-сервер» является основной формой взаимодействия компьютеров в вычислительных сетях. Один из компьютеров располагает информационно-вычислительными ресурсами (процессы, файловая система, почтовая служба, служба печати, базы данных), а другие компьютеры пользуются ими. Компьютер, управляющий тем или иным ресурсом, принято называть *сервером* этого ресурса, а компьютер, желающий им воспользоваться, — *клиентом*. Поскольку круг задач, которые должны выполнять серверы, многообразен и сложен, серверы в больших сетях стали *специализированными*. Если ресурс — база данных, то говорят о *сервере баз данных*, назначение которого обслуживать запросы клиентов, связанные с обработкой данных. Если ресурс — файловая система, то говорят о *файловом сервере*. *Файл-серверы* и *принт-серверы* управляют доступом пользователей к файлам и принтерам соответственно, *почтовые* — передачей электронных сообщений между пользователями сети, *коммуникационные* — потоком данных и почтовых сообщений между вашей сетью и другими сетями или удаленными пользователями через модем и телефонную линию и др.

Комбинированные сети обладают лучшими качествами одноранговых сетей и сетей на основе сервера.

3.10.5. Области применения локальных сетей

Локальные сети находят множество применений, важнейшими из которых являются:

- ♦ *сети персональных компьютеров;*
- ♦ *сети баз данных и сети-хранилища*, используемые для связи больших систем, таких как мэйнфреймы, суперкомпьютеры и запоминающие устройства большой емкости;
- ♦ *высокоскоростные офисные сети*, способные поддерживать большое число и более широкое распределение офисных систем по сравнению с системами машинных залов;
- ♦ *магистральные ЛС* — ряд дешевых ЛС с низкой пропускной способностью в отдельных зданиях или отделах, связанные локальной сетью с высокой пропускной способностью.

3.10.6. Многоуровневая организация сетей

Вычислительная сеть представляет собой сложную иерархическую систему из большого числа разнообразных элементов (разнотип-

ных ЭВМ, узлов связи и другой реальной физической аппаратуры), взаимодействующих с помощью каналов связи. *Каналом связи* называют физическую среду (витая пара, коаксиальный кабель, оптоволоконная линия, эфир) и аппаратные средства, осуществляющие передачу информации от одного узла коммутации к другому либо к абоненту связи.

Одной из наиболее острых проблем организации сети является проблема совместности элементов. Выпускаемое производителями оборудование должно отвечать соответствующим *стандартам*. В компьютерных сетях идеологической основой стандартизации является *многоуровневый* подход к разработке средств сетевого взаимодействия. На его основе была создана стандартная многоуровневая архитектура, использующая универсальный модульный принцип организации управления процессами в сети. Сложная задача организации взаимодействия между устройствами в сети разбивается на более простые задачи-модули. Процедура декомпозиции включает в себя четкое определение функций каждого модуля, решающего отдельную задачу, и интерфейсов между ними. Все множество модулей разбивают на уровни, образующие соответствующую иерархию. При этом модули каждого из уровней решают соответствующие задачи. На нижнем, например, уровне осуществляется надежная передача электрических сигналов между соседними узлами, на более высоком — транспортровка сообщений в пределах всей сети с использованием при этом средств нижнего уровня и т.п.

В вычислительной сети связь в действительности устанавливается не между отдельными машинами, а между прикладными программами, или, более точно, между *прикладными вычислительными процессами*, протекающими в машинах. При этом под *прикладным процессом* следует понимать прикладную программу вместе с ее наборами данных и выделенными ей ресурсами машины.

Процесс порождается программой или пользователем и связан с данными. Ввод и вывод данных производится в форме *сообщений* (последовательности данных, имеющих законченное смысловое значение) через входные и выходные *порты* — программно-организованные точки (рис. 3.61).

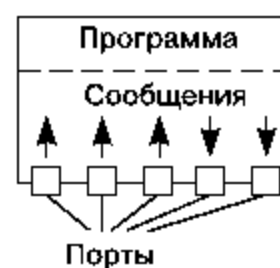


Рис. 3.61. Модель процесса

Как объект процесс представляется совокупностью портов, через которые он взаимодействует с другими процессами, т.е. обменивается сообщениями, которые передаются по каналам, создаваемым средствами сети (рис. 3.62).

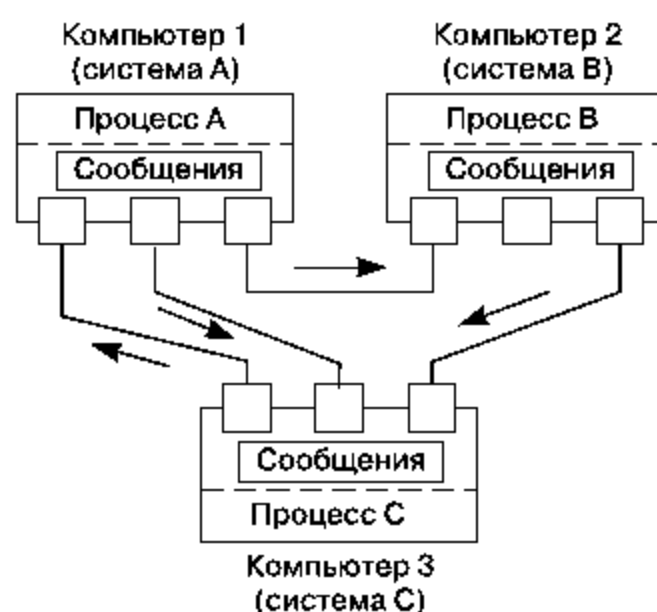


Рис. 3.62. Взаимодействие процессов

Этот обмен реализуется с помощью целого ряда специальных процессов и обслуживающих их аппаратных и программных средств: процессов установления маршрута передачи сообщения, управления передачей, установления соединения, интерпретации принятого сообщения и др. Промежуток времени, в течение которого взаимодействуют процессы, называют *сеансом*.

С учетом сказанного будем рассматривать *вычислительную сеть* как совокупность систем (ЭВМ, узлов связи и др.), в каждой из которых существует некоторая совокупность процессов, взаимодействующих через передающую среду путем обмена сообщениями. Организация вычислительных сетей должна *удовлетворять* следующим основным требованиям:

- ♦ *открытости* — возможности взаимодействия систем с различными техническими характеристиками и логической структурой, возможности включения дополнительных ЭВМ, терминалов и др.;
- ♦ *гибкости* — сохранению работоспособности при изменении структуры в результате выхода из строя ЭВМ, линий и узлов связи;
- ♦ *эффективности* — обеспечению требуемого качества обслуживания пользователей при минимальных затратах.

Для обеспечения открытости, гибкости и эффективности управление процессами организуется на основе *модели взаимодействия открытых систем* — модели OSI (Open System Interconnection), *стандартизирующей* их взаимодействие. Модель определяет различные уровни взаимодействия, приписывает им стандартные имена и указывает, какие функции должен выполнять каждый уровень.

В каждой из систем программные и аппаратные модули, реализующие определенные функции обработки и передачи данных в соответствии с моделью OSI, распределены по уровням 1—7. Уровень 1 является нижним, а уровень 7 — верхним (рис. 3.63). Уровню могут быть поставлены в соответствие некоторые процессы, аппаратные и программные средства (объекты управления). Каждый уровень обслуживает соседний старший уровень.

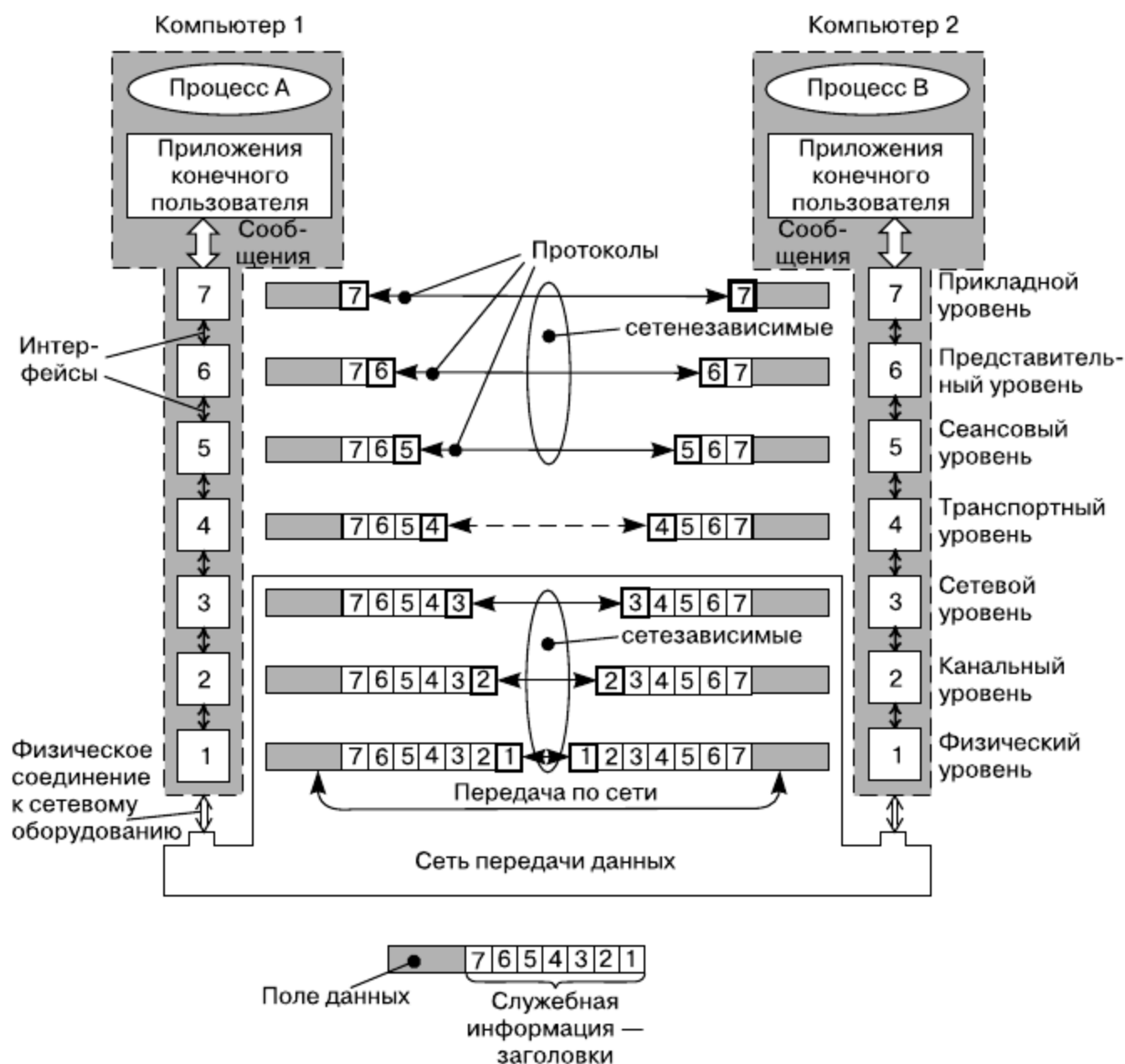


Рис. 3.63. Модель OSI

Связь между объектами соседних уровней одной системы реализуется *межуровневым интерфейсом*. Организация взаимодействия между одинаковыми уровнями различных систем (процессов) определяется соответствующим *протоколом* — формализованными правилами, определяющими тип и структуру сообщений, их последовательность и процедуры обработки.

Функции всех уровней модели OSI можно отнести к одной из двух групп:

- ♦ зависящие от конкретной технической реализации сети (три нижних уровня — *физический, канальный и сетевой*);
- ♦ ориентированные на работу с приложениями (три верхних уровня — *прикладной, представительный, сеансовый*) и мало зависящие от технических особенностей построения сети; на протоколы этих уровней не влияют какие бы то ни было изменения в топологии сети, замена оборудования или переход на другую сетевую технологию.

Транспортный уровень является промежуточным, он скрывает все детали функционирования нижних уровней от верхних, что позволяет разрабатывать приложения, не зависящие от технических средств непосредственной транспортировки сообщений.

Уровень 1 — *физический* — обеспечивает передачу дискретной информации через передающую среду (телефонная линия, коаксиальный кабель, спутниковая связь и др.). На этом уровне определяются характеристики электрических сигналов, передающих дискретную информацию (крутизна фронтов импульсов, тип кодирования, скорость передачи сигналов и др.). Здесь также стандартизуются типы разъемов и назначение каждого контакта.

Функции физического уровня реализуются во всех устройствах, подключенных к сети. Со стороны компьютера функции физического уровня выполняются сетевым адаптером или последовательным портом.

Уровень 2 — *канальный* — обеспечивает надежную передачу данных на физическом уровне. Для обеспечения надежности используются средства контроля принимаемых данных, позволяющие выявлять ошибки в поступающих данных. При обнаружении ошибки производится повторный запрос данных. Уровень управления каналом обеспечивает передачу данных с достоверностью, необходимой для нормальной работы системы, по недостаточно надежному физическому каналу.

Уровень 3 — *сетевой* — обеспечивает передачу данных через базовую систему передачи данных (маршрутизация, коммутация и адресация информации, управление потоками данных). Сетевые протоколы

реализуют продвижение пакетов через сеть. Одной из главных задач сетевого уровня является *маршрутизация* — проблема выбора наилучшего пути.

Уровень 4 — *транспортный* — реализует процедуры сопряжения объектов сети (ЭВМ и др.) с базовой сетью передачи данных (СПД). Назначение транспортной службы — организация единого для всей вычислительной сети интерфейса между процессами. Это система, состоящая из транспортных модулей, связывающих ЭВМ с СПД.

Уровень 5 — *сеансовый* — организует сеансы связи на период взаимодействия процессов. Управляет ведением диалога между модулями своего уровня путем синхронизации сообщений. На этом уровне по запросам процессов создаются порты для приема и передачи сообщений и организуются соединения — логические каналы.

Уровень 6 — *представительный* — имеет дело с формой представления передаваемой по сети информации, не меняя при этом ее содержания. На этом уровне выполняются преобразование и сжатие передаваемой информации, управление форматами и ряд других операций.

Уровень 7 — *прикладной* — это в действительности просто набор разнообразных протоколов, с помощью которых пользователи сети получают доступ к разделяемым ресурсам, таким как файлы, принтеры или гипертекстовые Web-страницы, а также организуют свою совместную работу, например, с помощью протокола электронной почты. Единица данных, которой оперирует прикладной уровень, — *сообщение*.

Передача информации от компьютера 1 к компьютеру 2 — адресату реализуется на основе модели OSI следующим образом. Пусть *приложение* обращается с запросом к прикладному уровню, например к файловой службе. На основании запроса программное обеспечение прикладного уровня формирует сообщение стандартного формата, состоящее из *заголовка* и *поля данных* (см. рис. 3.63). *Заголовок* содержит служебную информацию, которую необходимо передать через сеть прикладному уровню машины-адресата (компьютер 2), чтобы сообщить ему, какую работу надо выполнить. В рассматриваемом случае заголовок, очевидно, должен содержать информацию о месте нахождения файла и типе операции, которую необходимо над ним выполнить. *Поле данных* может быть пустым или содержать, например, данные, которые необходимо записать в удаленный файл.

Прикладной процесс А в системе (уровень 7) формирует сообщение *прикладному процессу В*, сообщаясь только с линией взаимодействия этих двух процессов, но не с организацией сети. Фактически сообщения, формируемые процессом А, проходят последовательно через

уровни 7—1, подвергаясь процедурам последовательного обрамления (каждый уровень добавляет к сообщению собственную служебную информацию — заголовок своего уровня, в котором содержатся указания для протокола этого же уровня машины-адресата). Затем сообщения передаются по каналу связи и через уровни 1—6 через систему интерфейсов достигают прикладного уровня и поступают к процессу В. При этом каждый из уровней 1—7 анализирует и обрабатывает свой заголовок, выполняет соответствующие ему функции, а затем удаляет этот заголовок и передает сообщение вышележащему уровню.

Важнейшим направлением стандартизации в области вычислительных сетей является *стандартизация коммуникационных протоколов*.

В настоящее время в сетях используется большое количество наборов протоколов для разнородной вычислительной среды — *стеков* коммуникационных протоколов: TCP/IP, SMB и др.

На нижних уровнях — *физическом и канальном* — применяются одни и те же хорошо стандартизованные протоколы Ethernet, Token Ring, FDDI и другие, которые позволяют использовать во всех сетях одну и ту же аппаратуру. На верхних уровнях все сети работают по своим собственным протоколам. Эти протоколы часто не соответствуют рекомендуемой модели OSI разбиению на уровни. В частности, функции сеансового и представительного уровня, как правило, объединены с прикладным уровнем.

Следует четко различать модель OSI и стек OSI. Если модель OSI является концептуальной схемой взаимодействия открытых систем, то стек OSI — это набор вполне конкретных спецификаций протоколов.

Стек OSI, в отличие от других стеков протоколов, полностью соответствует модели OSI и включает в себя спецификации протоколов для всех семи уровней взаимодействия, определенных этой моделью. Протоколы стека OSI достаточно сложны, что требует больших затрат вычислительной мощности центральных процессоров, поэтому они больше подходят для мощных машин, а не для сетей ПК. В связи с этим в модели OSI используются различные стеки, поддерживающие популярные стандарты различных уровней — физического, канального и др.

Каждому из семи уровней модели OSI соответствуют свои протоколы.

Функции представительного и прикладного уровней выполняет протокол SMB (Server Message Block). На его основе реализуется файловая служба, а также службы печати и передачи сообщений между приложениями.

Соответствие некоторых наиболее распространенных протоколов уровням модели OSI приведено в табл. 3.10.

Таблица 3.10

Распространенные протоколы и уровни модели OSI

Уровень модели OSI	Стек			
	IBM/ Micro- soft	TCP/IP	Novell	Стек X.25
Прикладной	SMB	Telnet, FTP, SNMP, SMTP, WWW	NCP, SAP	X.339
Представительный				X.226
Сеансовый	Net-BIOS	TCP		X.225
Транспортный				X.224
Сетевой		IP, RIP, OSPF		X.25
Канальный	802.3 (Ethernet), 802.5 (Token Ring), FDDI, PPP, Fast Ethernet, SLIP, 100VG-AnyLAN, ATM, LAP-B, LAP-D			HDLC
Физический	Экранированная и неэкранированная витая пара, коаксиальный кабель, оптоволокно, радиоволны			X.21

4. СРЕДСТВА И МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ

4.1. ОБЩАЯ ХАРАКТЕРИСТИКА ВЫЧИСЛИТЕЛЬНЫХ СРЕДСТВ ОБРАБОТКИ ИНФОРМАЦИИ

Широкое использование в науке, технике и других сферах человеческой деятельности самой разнообразной информации потребовало создания удобных средств для ее хранения, обработки и передачи. В результате были созданы различные типы вычислительных машин (ВМ).

Термином вычислительная машина (ВМ) будем обозначать комплекс технических и программных средств, предназначенный для автоматизации, подготовки и решения задач пользователей.

По форме представления информацию разделяют на *аналоговую* и *дискретную*. Обе формы имеют особое значение при создании, хранении, передаче и обработке информации с помощью средств вычислительной техники.

В вычислительных машинах информация представляется с помощью электрических сигналов. При этом возможны две формы представления числового значения какой-либо переменной, допустим X :

- ♦ в виде *одного* сигнала — например, электрического напряжения, которое сравнимо с величиной X (аналогично ей). Так, при $X = 320$ единицам на вход вычислительного устройства можно подать напряжение 3,2 В (масштаб представления 0,01 В/ед.);
- ♦ в виде *нескольких* сигналов — нескольких импульсов напряжений, которые сравнимы с числом единиц, десятков, сотен в X и т.д.

Первая форма представления информации (с помощью аналога) называется *аналоговой* или *непрерывной*. Величины, представленные в такой форме, могут принимать принципиально любые значения в определенном диапазоне. Число значений такой величины бесконечно велико, а отсюда и названия — *непрерывная* информация и *непрерывная* величина, т.е. промежутки между значениями, которые она может принимать, отсутствуют.

Вторая форма представления информации называется *дискретной* (с помощью набора напряжений, каждое из которых соответствует одной из цифр представляемой величины). Такие величины, принимающие не все возможные, а лишь определенные значения, называются

дискретными. В отличие от непрерывной величины число значений дискретной величины всегда будет конечным.

Для создания вычислительной машины, использующей непрерывную форму представления информации, требуется меньшее число устройств, чем для ВМ, использующей дискретную форму, поскольку в первой каждая величина представляется одним, а не несколькими сигналами. Однако эти устройства будут сложнее, так как должны различать значительно большее число состояний сигнала.

В зависимости от формы представления данных различают *аналоговые*, *цифровые* (электронные) и *гибридные* вычислительные машины (рис. 4.1).

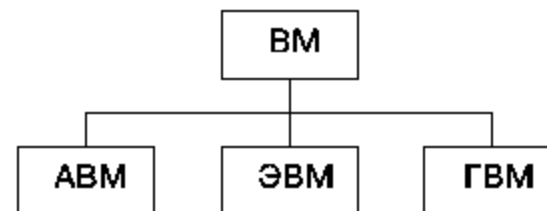


Рис. 4.1. Классификация ВМ по принципу действия

В *аналоговых* ВМ используется непрерывная форма представления сигналов. Рабочие характеристики электронных схем в АВМ моделируют процесс вычислений. Например, для моделирования операции сложения двух чисел $z = x + y$ можно использовать схему, выходное напряжение U_z которой представляет собой сумму двух входных напряжений. То есть, соединив последовательно два источника U_x и U_y , получим $U_z = U_x + U_y$ (рис. 4.2), где $U_z = kz$, если $U_x = kx$, $U_y = ky$, $k = \text{const}$.

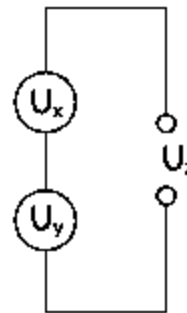


Рис. 4.2. Схема моделирования операции сложения

Для изучения различных процессов и явлений в науке и технике широко применяют *метод аналогий* (рис. 4.3), когда вместо оригинала А изучают его модель В. Такое замещение возможно при наличии *соответствий (аналогий)* между А и В. Различают аналогии двух видов — *физические* и *математические*.

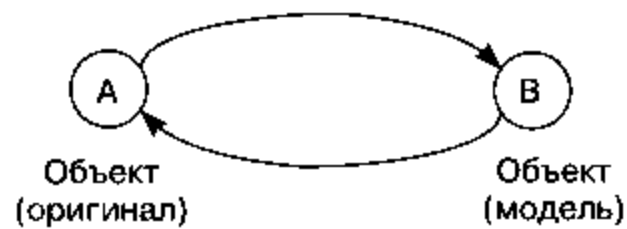


Рис. 4.3. Метод аналогий: оригинал и модель

В первом случае объект В является уменьшенной (реже увеличенной) копией оригинала, и при изменении оригинала требуется менять и модель, что является недостатком такого метода.

Во втором случае оригинал и модель имеют сходство математических описаний, хотя их физическая природа может быть совершенно разной.

Для исследования математических моделей используют АВМ, представляющие собой совокупность соединенных определенным образом так называемых *решающих блоков*, которые могут интегрировать сигнал, дифференцировать, выполнять любое его функциональное преобразование и т.п. (рис. 4.4).



Рис. 4.4. Решающие блоки

Подготовка задачи для решения на АВМ, т.е. *программирование*, включает два этапа:

- 1) соединение (коммутация) решающих блоков в соответствии с задачей;
- 2) расчет масштабов физических величин и времени.

Все математические величины в АВМ (МН-7, МН-18, АВК-31) представляются напряжением постоянного тока в диапазоне от -10 до $+10$ В.

Эти напряжения изменяются *непрерывно* во времени (могут принять любое значение в указанном диапазоне), *моделируя* изменение математических величин.

Достоинствами АВМ являются:

- ♦ *высокое быстродействие*, обусловленное тем, что вычислительный процесс *разворачивается в пространстве*, т.е. все решающие блоки работают параллельно, одновременно выполняя преобразования, на которые разбит процесс вычислений. Процесс можно развернуть и во времени, возлагая на одни и те

же устройства различные преобразования (последовательно во времени), сократив число преобразующих устройств. Однако при этом потребуются запоминать промежуточные результаты. Запоминание большого числа физических величин без искажения их количественных характеристик связано с весьма серьезными техническими трудностями;

- ♦ *простота программирования*, которое сводится к соединению решающих блоков в соответствии с алгоритмом.

К недостаткам АВМ относятся:

- ♦ *ручная коммутация* решающих блоков (программирование), связанная с реализацией алгоритма. Правда, сейчас появляются работы по автоматизации программирования на АВМ (машина «Экстрэма»);
- ♦ *низкая точность* решения задач, обусловленная тем, что количественное соответствие между числом и моделирующей его физической величиной удается выдержать с ограниченной точностью. Поскольку установить или измерить напряжение точно нельзя, исходные данные будут содержать ошибки, которые будут присутствовать во всех промежуточных и окончательных результатах. Кроме того, каждый решающий блок вносит свою погрешность, т.е. точность решения задачи на АВМ прямо связана с точностью решающих блоков, которая зависит, в свою очередь, от точности элементов, из которых блоки построены. Точность современных АВМ составляет сотые доли процента.

Эти машины предназначены в основном для решения задач, описываемых системами дифференциальных уравнений: исследование поведения подвижных объектов, моделирование процессов и систем, оптимальное управление.

Однако из-за сложности технической реализации устройств выполнения логических операций с непрерывными сигналами, длительного хранения таких сигналов и их точного измерения АВМ не могут эффективно решать задачи, связанные с хранением и обработкой больших объемов информации, которые легко решаются цифровыми электронными вычислительными машинами (ЭВМ).

ЭВМ, компьютер, вычислительная система — это слова-синонимы, они обозначают одно и то же.

Современные ЭВМ представляют собой сложный комплекс технических и программных средств, предназначенных для решения задач научного и технического характера в целях выполнения экономических и стратегических расчетов, решения задач управления различными технологическими процессами и объектами. Разнообразие областей при-

менения существенно влияет на архитектуру ЭВМ, состав и назначение ее устройств, организацию процессов вычислений.

Термин «электронная ВМ» не совсем точен, ибо он отражает применяемую для построения ЭВМ техническую базу, а не принцип устройства. Термин вычислительная машина (ВМ) тоже устарел, поскольку современные ЭВМ решают не только вычислительные задачи, но и многие другие, в том числе:

- ♦ информационно-поисковые;
- ♦ издательской деятельности;
- ♦ перевода с одного языка на другой;
- ♦ игры в шахматы и др.

Поэтому сегодня наряду с термином ЭВМ чаще используют термин «компьютер» (computer — вычислитель). Решение любой задачи на компьютере сводится к обработке информации.

ЭВМ — система по обработке информации, которая состоит:

- ♦ из аппаратных средств (электронных и электромеханических устройств);
- ♦ программных средств (программное обеспечение);
- ♦ документации.

Таким образом, структура вычислительной системы допускает деление ее на уровни, самыми низкими из которых являются *аппаратные* или *технические средства* — интегральные логические микросхемы и электронные приборы.

Следующий уровень — *архитектура*, т.е. соединение элементов и характеристик ЭВМ, к которым имеет непосредственное отношение пользователь (этот уровень также можно считать аппаратным). На этом уровне иерархии элементами структуры ЭВМ являются память, процессор, устройство ввода-вывода и др.

Самый высокий и, возможно, самый существенный уровень — *программное обеспечение* или *программы*, управляющие работой ЭВМ. На этом уровне сама ЭВМ может считаться элементом (рис. 4.5).

Исходные *данные* задачи, являющиеся конкретной реализацией информации, обрабатываются на ЭВМ и преобразуются в *результат* на основе заданного алгоритма — *программы* (см. рис. 4.5).



Рис. 4.5. Представление ЭВМ на самом высоком уровне

ЭВМ выполняет заданную ей программу автоматически, т.е. без вмешательства человека. К достоинствам ЭВМ относятся *универсальность и неограниченная точность*. С помощью ЭВМ может быть решена любая задача, алгоритм которой известен, и в этом смысле ЭВМ является *универсальным* средством обработки информации.

В ЭВМ любые математические величины (данные) представляются в числовой форме, т.е. все исходные, промежуточные и окончательные результаты изображаются в виде совокупности цифр в *двоичной* системе счисления. При этом каждый разряд X_i может принимать значение 0 или 1, т.е. является двоичной переменной, $X_i \in \{0, 1\}$. Физические значения переменной X_i соответствуют двум уровням напряжения, например:

$$X_i = 0 \rightarrow U = 0 \div 0,8 \text{ В};$$

$$X_i = 1 \rightarrow U = 2,4 \div 5 \text{ В}.$$

Точность данных в ЭВМ определяется числом разрядов и их представлением, а не точностью элементов в каждом разряде. Таким образом, увеличивая разрядность машины ($n = 32, 64, 128$), можно увеличивать точность представляемых данных неограниченно.

Функциональные возможности современных ЭВМ (функция характеризует алгоритм работы системы, т.е. определяет правила получения результата), как основные, так и дополнительные, являются достаточно широкими и могут быть реализованы аппаратным, программным или аппаратно-программным способами.

Каждый из этих способов имеет свои плюсы и минусы и влияет на производительность ЭВМ, которая в ЭВМ общего назначения измеряется десятками и сотнями миллионов операций в секунду.

В результате совершенствования технологии изготовления ЭВМ увеличиваются их мощность и возможности и уменьшаются габариты. Так, современные микроЭВМ обладают практически теми же возможностями, которыми обладали большие ЭВМ 60-х гг. или мини-ЭВМ 70-х гг. XX в.

В ЭВМ, ориентированных на решение узкого класса задач, но с различными данными (*специализированные ЭВМ*), программу логично записать в ПЗУ, хранящее информацию при отключении питания, либо реализовать аппаратно, т.е. зашить в электронные схемы машины. В последнем случае повышаются надежность и быстродействие ЭВМ. Такой подход используют во многих видах бортовой ВТ, работающей в жестком режиме реального времени.

Гибридные вычислительные машины (ГВМ) сочетают достоинства аналоговых и цифровых вычислительных машин.

В настоящее время преобладает тенденция объединения разных вычислительных систем в *вычислительные сети* различного масштаба (рис. 4.6), что позволяет объединять информационно-вычислительные ресурсы для более эффективной реализации информационных процессов.

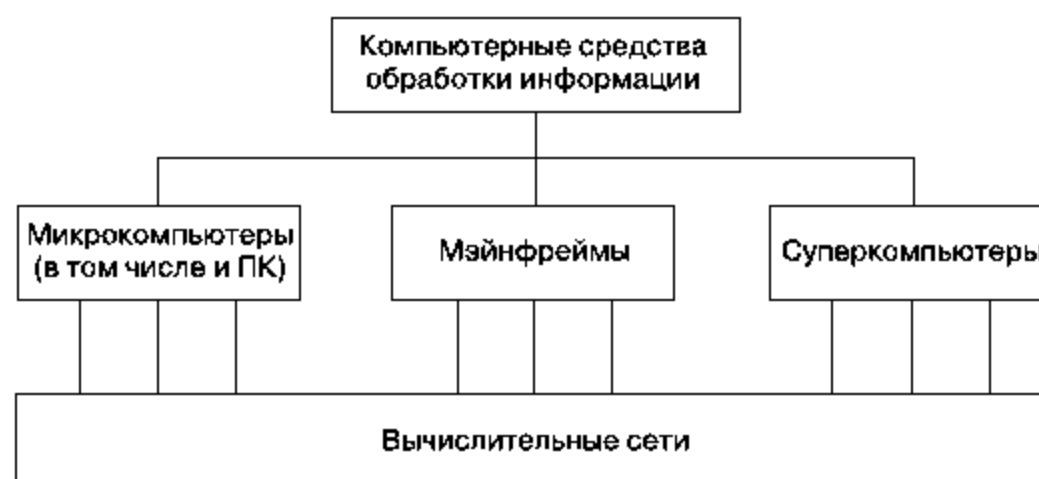


Рис. 4.6. Классификация компьютерных средств обработки информации

Вычислительная (компьютерная) сеть — комплекс территориально рассредоточенных ЭВМ различной производительности и терминальных устройств, соединенных между собой каналами передачи данных [5].

Чтобы определить производительность ЭВМ, часто указывают тактовую частоту микропроцессора, зная которую можно достаточно точно определить время выполнения любой машинной операции и оценить быстродействие ЭВМ, которое измеряется в следующих единицах:

- ♦ МИПС (MIPS) — миллион операций в секунду над числами с фиксированной точкой;
- ♦ МФЛОПС (MFLOPS) — миллион операций в секунду над числами с плавающей точкой;
- ♦ ГФЛОПС (GFLOPS) — миллиард операций в секунду над числами с плавающей точкой.

Однако *производительность* компьютера определяется не только характеристиками применяемого микропроцессора, но и емкостью оперативной памяти, типами периферийных устройств, качеством конструктивных решений и др.

Микрокомпьютеры — это компактные по сравнению с мэйнфреймами ЭВМ, имеющие производительность до сотен МИПС. *Персональные компьютеры* — это микрокомпьютеры универсального назначения,

рассчитанные на одного пользователя. Разновидностью микрокомпьютера является *микроконтроллер* — специализированное устройство, основанное на микропроцессоре и встраиваемое в систему управления или технологическую линию.

Мэйнфреймы — это более сложные и дорогие машины высокой производительности, максимальный объем оперативной памяти которых достигает 342 Тбайт.

Суперкомпьютеры (суперЭВМ) — высокопроизводительные, сверхбыстродействующие ЭВМ (более 100 МФЛОПС), создаются в виде высокопараллельных многопроцессорных вычислительных систем.

4.2. ЭТАПЫ РЕШЕНИЯ ЗАДАЧИ НА ЭВМ

Процесс решения задачи на ЭВМ разделяется на ряд последовательных этапов (рис. 4.7).

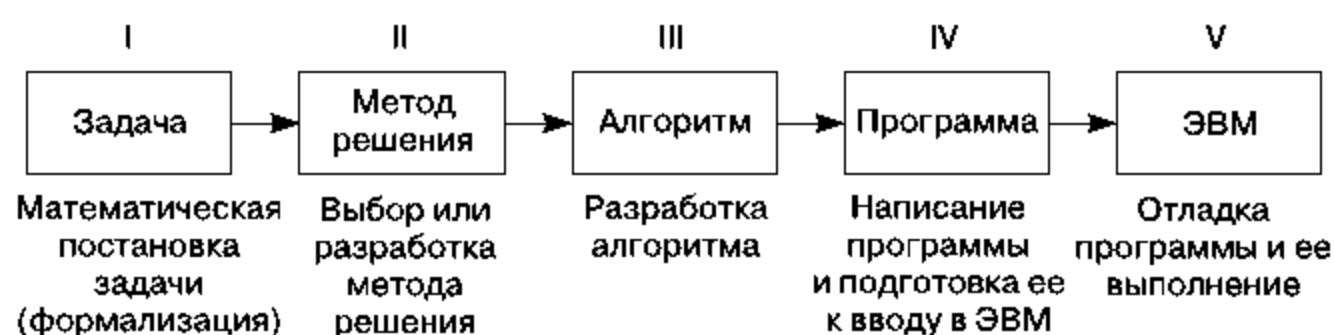


Рис. 4.7. Этапы решения задачи на ЭВМ

Решение задачи начинается с ее *постановки*, изложенной на языке строго определяемых математических понятий. Точное описание исходных данных, условий задачи и цели ее решения называется *математической постановкой задачи*. Этап математической постановки задачи называют еще этапом *формализации*, так как на этом этапе многие из условий задачи, заданные в форме различных словесных описаний, необходимо выразить на точном (формальном) языке математики. На этом этапе определяют состав и характеристики исходных данных, различные условия и ограничения записывают в математической форме. Разработка математической модели исследуемого объекта позволяет свести решение реальной задачи к математической. В случае если задача является математической, этап формализации может и не потребоваться.

Поскольку на ЭВМ решаются сложные задачи и не только вычислительные, этап формализации может потребовать значительных усилий и времени.

Специалисты-постановщики считают, что выполнение этапа формализации равнозначно выполнению половины всей работы по программированию.

Задача составления, например, расписания в вузе относится к сложным задачам, так как связана с распределением большого числа групп и потоков разной численности по различным аудиториям с ограничением времени занятий в каждой аудитории и с разделением занятий по степеням сложности в течение дня. Она сводится к известной в комбинаторной геометрии задаче нахождения покрытия некоторой поверхности заданным набором прямоугольников, причем искомое покрытие должно удовлетворять ряду ограничений, записанных в виде математических уравнений, и оцениваться значением критерия качества. Построение математической модели может быть использовано для решения и других задач — раскроя, компоновки и т.д.

Разработка метода решения. Цель этапа — свести задачу к математической модели, для которой известен метод решения. Если известно несколько методов, то предстоит выбрать из них оптимальный. Если ни один из них не подходит, то не исключается возврат к этапу формализации с целью упрощения решаемой задачи путем ввода дополнительных ограничений или, наоборот, снятия некоторых из них.

Разработка алгоритма. Алгоритм устанавливает последовательность точно определенных действий, приводящих к решению задачи. Часто не удается получить решение в явном виде, т.е. в виде формулы, связывающей исходные данные и результаты. В таких случаях решение ищется в виде алгоритма.

Рассмотренные этапы носят творческий характер, однако этап построения алгоритма помимо творческих поисков включает в себя и чисто технологические вопросы. Используя определенную дисциплину при конструировании алгоритма, можно получить алгоритм с явно выраженной структурой, что облегчает дальнейшую работу с ним. Аналогичная технология может быть использована и на последующих этапах — при разработке программы и при работе с этой программой.

Написание программы (разработка программы). Алгоритм, предназначенный для исполнения на ЭВМ, должен быть записан на языке программирования (см. подраздел 4.11.2).

Программа — это набор машинных команд (МК), которые следует выполнять компьютеру для реализации того или иного алгоритма. Иными словами, *программа* — это форма представления алгоритма для исполнения его машиной. Программист пишет программу на одном из языков программирования (исходный текст). Исходные тексты по-

мещают в текстовые файлы. Затем с исходными текстами работают программы-посредники — трансляторы (компиляторы и интерпретаторы). В результате трансляции получается программа, готовая к исполнению на ЭВМ.

Однако в процессе разработки программ могли быть допущены ошибки (содержательные), которые сохранились и в результирующей программе. Они не могут быть устранены на стадии трансляции, так как для их обнаружения необходима информация о сути решаемой задачи. Их может устранить только сам пользователь. Поэтому любая программа до ее практического использования должна пройти *этап отладки*, чтобы устранить ошибки, допущенные на предыдущих этапах.

В процессе отладки пользователь подготавливает систему тестов, с помощью которых проверяется работа программы в различных режимах работы.

Для облегчения процесса отладки в состав ОС включают специальные сервисные программы — *отладчики*. Пользователь формулирует задачу отладчику, который выполняет задание и выдает необходимую информацию о поведении программы. Программа, *прошедшая отладку*, готова для практического применения.

Рассмотренный процесс решения задачи на ЭВМ имеет в основе триаду:

модель — алгоритм — программа.

Системный подход к разработке программного обеспечения (ПО) рассматривается в технологии программирования или в методах разработки ПО.

4.3. АЛГОРИТМЫ. ОСНОВНЫЕ ТРЕБОВАНИЯ К АЛГОРИТМАМ

Алгоритм — это однозначно заданная последовательность элементарных действий (шагов) над исходной информацией, приводящая за конечное число их применений к результату (решение задачи, стратегия в игре, сборка узла автомобиля и др.).

Однако алгоритмом можно назвать лишь ту последовательность, которая удовлетворяет определенным требованиям. *Основные требования к алгоритму*:

- ◆ детерминированность (однозначность);
- ◆ понятность;
- ◆ направленность;

- ♦ результативность;
- ♦ массовость;
- ♦ эффективность.

Детерминированность. Все шаги алгоритма делят на условные и безусловные. Каждый из них должен быть четко и однозначно определен и не должен допускать произвольной трактовки исполнителем. Действия, выполненные на каждом шаге, должны быть однозначны независимо от исходных данных алгоритма. Такая особенность алгоритма называется *детерминированностью* или *определенностью*. Это замечательное свойство алгоритма и позволяет использовать в качестве его исполнителя *электронную вычислительную машину*. Типичным примером множества элементарных действий является *система команд ЭВМ*. Современные универсальные ЭВМ выполняют сотни команд по выполнению операций приема данных, их обработки, запоминания, управления вычислительным процессом, выводу данных и др.

Понятность. Алгоритмы должны быть ориентированы на конкретного исполнителя и сформулированы с учетом его возможностей и способностей. Например, словесно описанный алгоритм подсчета числа гласных в русском алфавите может быть понятен только человеку, знающему русский язык. Если же исполнитель не знает его, то алгоритм надо переписать на язык, *понятный* исполнителю, и сообщить дополнительные сведения, т.е. перечислить гласные русского алфавита. Для ЭВМ алгоритм должен быть написан на одном из алгоритмических языков в виде *программы*.

Направленность (правильность) означает, что алгоритм применяется к исходным данным и приводит к получению результата. Так, для алгоритма сложения исходными данными являются пары слагаемых (чисел), а результатом — сумма (число). Для величин, не попадающих в область исходных данных, для которых составлен алгоритм, он не может дать адекватного результата, и в этом случае должно быть оговорено число, являющееся результатом. Это свойство определяет *правильность* алгоритма, т.е. при правильных входных данных он должен гарантировать результат, а при других — выдавать соответствующее сообщение.

Результативность (сходимость) — свойство, которое означает, что после конечного числа шагов, зависящих от данных, алгоритм должен остановиться с указанием того, что считать результатом. В этом случае говорят, что алгоритм сходится.

Массовость — применимость алгоритма к разным наборам входных данных, т.е. один и тот же алгоритм можно применять для решения целого класса однотипных задач.

Эффективность. Отдельные указания исполнителю, содержащиеся в каждом шаге алгоритма, называются командами, а совокупность команд, которые могут быть выполнены исполнителем, называется *системой команд* (СК) исполнителя. Исполнители отличаются друг от друга своими возможностями, т.е. наборами команд, которые они понимают и могут выполнить. Следовательно, алгоритм, рассчитанный на конкретного исполнителя, должен содержать простые шаги, чтобы их можно было выполнить точно и за конечное время, и содержать те команды, которые входят в его СК.

Реализация алгоритма связана с возможностью выполнения каждой команды за конечное время. *Эффективный* алгоритм позволяет решать задачу за минимальное время.

4.4. КОМПЬЮТЕР КАК СРЕДСТВО АВТОМАТИЗИРОВАННОЙ ОБРАБОТКИ ИНФОРМАЦИИ

4.4.1. Основные понятия и определения

Электронная вычислительная машина (ЭВМ), или компьютер, — это физическая система, служащая для автоматизации процесса вычислений и алгоритмической обработки информации путем выполнения программы, хранимой в памяти машины.

Основополагающими понятиями компьютера являются понятия его *архитектуры* и *структуры*.

Архитектура определяется как совокупность свойств компьютера, существенных для пользователя. Говоря об *архитектуре* ВМ, подразумевают ее логическое построение, т.е. ее представление с точки зрения программиста, а это — перечень средств, которыми программист может пользоваться, составляя программу на уровне машинных команд: перечень и формат команд, формы представления данных, механизмы ввода-вывода, способы адресации памяти и др. При этом не затрагиваются вопросы физического построения машины: состав устройств, число регистров процессора, емкость памяти, интерфейс, тактовая частота ЦП и т.д. Этот круг вопросов будем определять понятием *структурная организация*.

Структура компьютера — некоторая *модель*, описывающая его логическую организацию на уровне аппаратуры, а именно: состав устройств, связь между ними (*интерфейс*), характер взаимодействия аппаратных модулей в процессе выполнения различных задач. Несмотря на большое разнообразие современных ЭВМ (сверхбольшие, большие,

мини-ЭВМ, микроЭВМ), каждый класс которых продолжает пополняться новыми представителями, в их логической структуре можно выделить устройства, входящие в состав любой ЭВМ: *центральный процессор (ЦП), память и устройства ввода-вывода (УВВ)*.

Архитектура и структурная организация — это две стороны описания ВМ и вычислительных систем. *Вычислительную систему (ВС)* определим как совокупность взаимосвязанных и взаимодействующих процессоров или вычислительных машин, периферийного оборудования и программного обеспечения (ПО), предназначенную для подготовки и решения задач пользователей. Применительно к ВС термин «архитектура» распространяется также на распределение функций между составляющими ВС и взаимодействие этих составляющих.

Функциональные возможности машины делят на *основные* (хранение, обработка, обмен информацией с внешними устройствами) и *дополнительные*, обеспечивающие:

- ♦ эффективные режимы работы;
- ♦ диалог с пользователем;
- ♦ высокую надежность и др.

4.4.2. Принципы организации ЭВМ

Большинство ЭВМ построено на основе некоторых общих принципов, заложенных американским ученым Джоном фон Нейманом (фоннеймановские ЭВМ или компьютеры фоннеймановской классической архитектуры), основными из которых являются:

- ♦ *двоичное кодирование* информации и ее деление на слова *фиксированной длины*;
- ♦ *принцип произвольного доступа* к основной памяти (ОП).

Основная память имеет *линейно-адресную* организацию, т.е. состоит из последовательно расположенных друг за другом ячеек (байтов), которые нумеруются от 0 до $N - 1$ (рис. 4.8). Такое представление памяти получило название *адресного пространства*. Это максимальное число ячеек основной памяти, которое может быть адресовано непосредственно процессором. Общее число ячеек памяти N называют *объемом памяти*. Номер ячейки называется *адресом*. Каждый байт имеет свой уникальный адрес. Адреса записываются в шестнадцатеричной системе счисления, например, 3A7Bh, где h — *идентификатор* шестнадцатеричной системы счисления. Такая память имеет *адресный принцип* построения и *побайтовую* организацию.



Рис. 4.8. Адресное пространство

Процессору в произвольный момент времени доступна *любая* ячейка памяти, причем время доступа t_d (время чтения и записи информации) одинаково для всех ячеек. Такая организация памяти отличается ее от других запоминающих устройств, например магнитных лент (память с *последовательным* доступом). Магнитную ленту (МЛ) можно тоже разбить на слова, но доступ к слову можно получить *только* после просмотра предыдущих участков МЛ:

- ◆ *представление алгоритма программой*, состоящей из команд;
- ◆ *команды и данные хранятся в одной памяти*. Отличие заключается в способе считывания слова. При этом адресное пространство памяти делится на две области (см. рис. 4.8):
область программ,
область данных;
- ◆ *вычислительный процесс определяется как последовательное выполнение команд* в порядке, определяемом программой;
- ◆ *жесткость структуры*, т.е. ее неизменность в процессе работы ЭВМ;
- ◆ *принцип программного управления*: решение любой задачи выполняется автоматически по программе, хранимой в памяти. Соседние команды программы располагаются в ячейках памяти со следующими по порядку адресами и выполняются преимущественно в той же очередности, в какой они размещены в ОП. Для решения новой задачи достаточно составить новую программу и ввести ее в ОП;
- ◆ *принцип хранимой программы*: программа хранится в памяти наравне с данными, а следовательно, над командами можно выполнять всевозможные операции как над числами, что ведет к повышению быстродействия. Для решения любой задачи на ЭВМ в нее надо ввести *данные и программу* (см. рис. 4.5).

4.4.3. Представление информации в ЭВМ

Данные — информация, представленная в формализованном виде, удобном для восприятия ЭВМ. В ЭВМ данными являются числа, команды, адреса, логические переменные, текст и т.п. Данные, как правило, кодируются. Под *кодированием* понимается представление символов одного алфавита символами другого. В ЭВМ используется *двоичное кодирование*, буквы которого представляются *первичным* алфавитом данных (0 и 1). Наименьшим элементом информации в ЭВМ является *бит* — двоичная переменная, например X , принимающая значение 0 либо 1, т.е. $X = \{0, 1\}$.

Для удобства работы из них создаются более сложные конструкции: *байты*, *слова*, *двойные слова* (слова двойной длины), которые составляют *вторичный* алфавит.

Слово (двоичное) — упорядоченная последовательность из n бит $X_{n-1} \dots X_1 \dots X_2 X_1 X_0$, имеющая определенный смысл: элемент цифровой информации, который передается, хранится и обрабатывается как одно целое.

Слово характеризуется *длиной*, которая совпадает с разрядностью центрального процессора (или с разрядной сеткой) машины, осуществляющего его обработку. Наименьшей обрабатываемой единицей в ЭВМ является *байт* — 8-разрядное слово (рис. 4.9). Из последовательности байтов могут быть составлены машинные элементы информации большей длины. Например, в 16-разрядных МП длина слова составляет 16 бит (2 байта), а в 32-разрядных — 32 бита (4 байта). Более сложной структурой являются *массивы данных*.

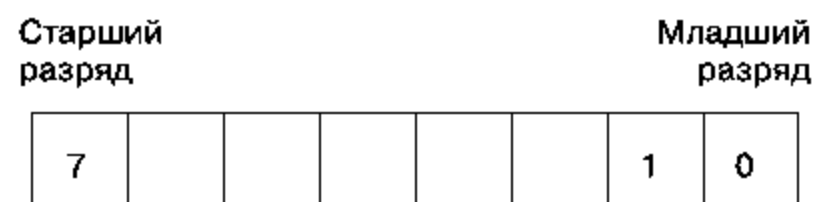


Рис. 4.9. Байт

Байт может отображать:

- ◆ целое число без знака;
- ◆ целое число со знаком;
- ◆ символ (код символа);
- ◆ часть более сложной конструкции (слова, двойного слова и т.д.).

В общем случае все данные в ЭВМ отображаются в виде *двоичных (машинных) слов* — последовательности нулей и единиц, которыми ЭВМ оперирует одновременно.

Для их хранения ЭВМ содержит большое число ячеек памяти и *регистров*.

Регистр — устройство для хранения одного машинного слова. Условное изображение 16-разрядного регистра с именем AX приведено на рис. 4.10, где AH и AL — старший и младший байты регистра соответственно.

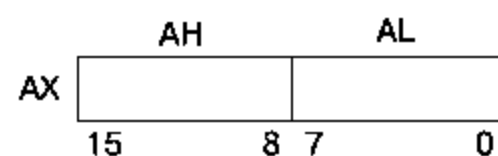


Рис. 4.10. Условное изображение регистра

Характеризуя регистр, говорят о его разрядности, т.е. длине слова, которое он может хранить.

Каждый разряд регистра представляет собой ячейку памяти объемом 1 бит — *триггер*, который может находиться в одном из двух устойчивых состояний — 0 и 1. Условное изображение на схемах одного из типов триггеров — синхронизируемого RS-триггера приведено на рис. 4.11. Он имеет два установочных входа: S — set (установка в 1), R — reset (установка в 0), вход синхронизации C и два выхода — *прямой* Q и *инверсный* $\bar{Q}$.

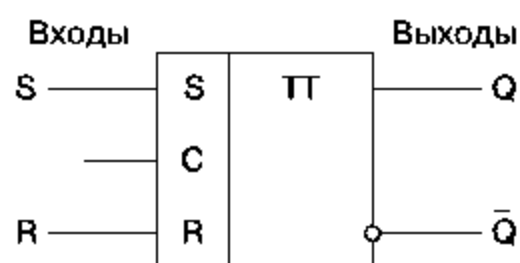


Рис. 4.11. RS-триггер

Выходной сигнал триггера (одного разряда регистра) при изменении его состояний под воздействием комбинаций входных сигналов показан на рис. 4.12. Переход от 0 к 1 и от 1 к 0 происходит не мгновенно, однако в определенные моменты времени этот сигнал достигает значений, которые воспринимаются элементами ЭВМ как 0 или 1.

Поскольку память ЭВМ состоит из *конечной* последовательности ячеек, а ячейки из *конечной* последовательности битов, *объем* представляемой в ЭВМ информации *ограничен* емкостью памяти. Поэтому числовая информация может быть представлена только с определенной точностью, зависящей от архитектуры памяти данной ЭВМ.

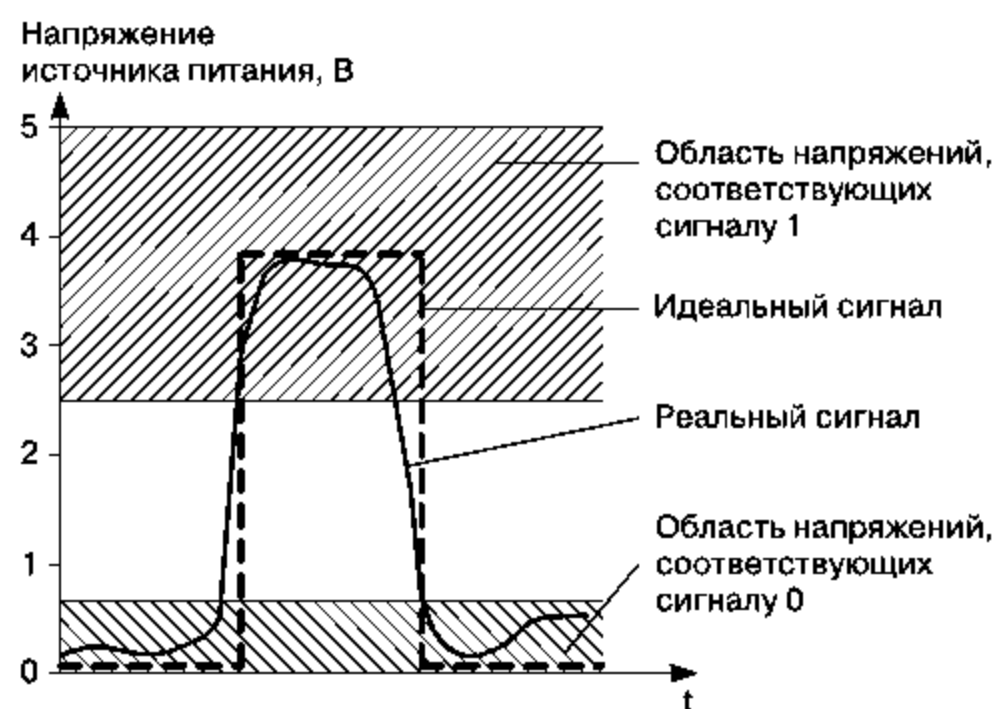


Рис. 4.12. Графическое изображение двоичного сигнала

СИСТЕМЫ СЧИСЛЕНИЯ

Понятие системы счисления. Система счисления (СС) — это совокупность цифр (символов) и правил для обозначения чисел.

Системы счисления делятся на *позиционные* и *непозиционные*.

Непозиционная система счисления — система, в которой символы, используемые для представления числа, не меняют своего значения в зависимости от местоположения (позиции) в записи числа. К непозиционным системам относится, например, *римская система счисления* (табл. 4.1).

Таблица 4.1

Римская система счисления

Римские цифры	I	V	X	L	C	D	M
Значение (обозначаемое число)	1	5	10	50	100	500	1000

Запись чисел в этой системе счисления осуществляется по следующим правилам:

- ♦ если цифра *слева* меньше, чем цифра *справа*, то левая вычитается из правой цифры, например 4 запишется как IV;
- ♦ если цифра *справа* меньше или равна цифре *слева*, то эти цифры складываются, например 6 запишется как VI, 8 — как VIII, 20 — как XX.

Так, число 1865 в римской системе счисления имеет вид MCCMLXV.

В общем случае непозиционные системы счисления характеризуются сложными способами записи чисел и правилами выполнения арифметических операций.

В настоящее время наиболее широко распространены *позиционные системы счисления*, в которых значение цифры определяется ее местоположением (*позицией*) в представлении числа. Наиболее привычной и распространенной из них является *десятичная система счисления* (рис. 4.13):

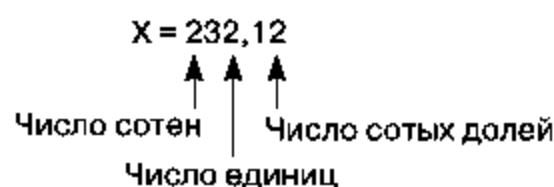


Рис. 4.13. Десятичная система счисления

При записи числа цифра 2, занимающая различные позиции, имеет различные значения.

Упорядоченный набор символов (цифр) $a_0, a_1, \dots, a_n$, используемых для представления любых чисел в заданной позиционной системе счисления, называют ее *алфавитом*, число символов (цифр алфавита) — ее *основанием* (q), а саму систему счисления называют *q-ричной*.

Основание системы счисления q показывает, во сколько раз увеличивается значение цифры в числе при смещении ее на одну позицию влево.

Таким образом, десятичное число $X = 232,12$ может быть разложено по степеням числа 10:

$$X = 232,12 = 2 \cdot 10^2 + 3 \cdot 10^1 + 2 \cdot 10^0 + 1 \cdot 10^{-1} + 2 \cdot 10^{-2},$$

$q = 10$ — основание десятичной СС.

Последовательность коэффициентов такого полинома и представляет собой десятичную запись числа X .

В общем случае любое q -ричное число можно представить в виде последовательности коэффициентов $a_{n-1} a_{n-2} \dots a_1 a_0$:

$$X_q = a_{n-1} a_{n-2} \dots a_1 a_0 a_{-1} \dots a_{-m}.$$

Такая запись числа в системе счисления с основанием q означает сокращенную запись выражения:

$$X_q = a_{n-1} \cdot q^{n-1} + a_{n-2} \cdot q^{n-2} + \dots + a_1 \cdot q^1 + a_0 \cdot q^0 +$$

$$+ a_{-1} \cdot q^{-1} + \dots + a_{-m} \cdot q^{-m} = \sum_{i=-m}^{n-1} a_i q^i, \quad (4.1)$$

где $a_i \in \{0, 1, 2, \dots, q-1\}$ — цифры q -ричной СС;

n и m — число разрядов целой и дробной частей соответственно;

q^i — вес i -го разряда.

В вычислительной технике кроме десятичной системы широкое применение находят двоичная ($q = 2$), восьмеричная ($q = 8$) и шестнадцатеричная ($q = 16$) системы счисления.

Анализ систем счисления и построенных на их основе кодов с позиций применения в системах передачи, хранения и преобразования информации показывает, что чем больше основание системы q , тем меньше разрядов требуется для представления данного числа, а следовательно, меньше времени для его передачи.

Двоичная система счисления. Основание $q = 2$, $a_i \in \{0, 1\}$, например:

$$X = 13_{10} = 1101_2 = 1 \cdot 2^3 + 1 \cdot 2^2 + 0 \cdot 2^1 + 1 \cdot 2^0.$$

Достоинствами двоичной системы счисления являются:

- ♦ простота выполнения операций сложения, вычитания и умножения:

$$0 + 0 = 0; 1 + 0 = 1; 0 + 1 = 1; 1 + 1 = 10;$$

$$0 \cdot 0 = 0; 1 \cdot 0 = 0; 0 \cdot 1 = 0; 1 \cdot 1 = 1;$$

- ♦ простота технической реализации элементов и устройств для хранения двоичных слов и выполнения над ними арифметических операций;
- ♦ тесная взаимосвязь с символикой булевой алгебры.

Недостаток системы — длинная запись двоичных чисел.

Восьмеричная система счисления. Основание $q = 8$, $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7\}$.

Пример: $X = 37_8 = 3 \cdot 8^1 + 7 \cdot 8^0 = 24 + 7 = 31_{10}$.

Шестнадцатеричная система счисления. Основание $q = 16$, $a_i \in \{0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F\}$.

Используется при программировании для сокращения записи двоичного числа путем эквивалентной замены 4-разрядных двоичных чисел — *тетрад* 16-ричными цифрами в соответствии с табл. 4.2.

Таблица 4.2

Системы счисления

Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3

Окончание

Десятичная	Двоичная	Восьмеричная	Шестнадцатеричная
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Перевод чисел из одной системы счисления в другую

В компьютере информация представляется в виде чисел, записанных в той или иной системе счисления (СС), кратной степени двойки (двоичной, 16-ричной и др.). Такие операции, как ввод, вывод и обработка чисел на компьютере, связаны с их преобразованием из одной СС в другую.

Различают перевод *целых, дробных и смешанных* чисел.

Перевод целых чисел

Перевод из десятичной в q -ричную систему счисления. Перевод можно выполнить различными способами — либо делением на основание системы счисления, либо методом *поразрядного взвешивания*, либо методом *взвешенного кодирования* (используется для перевода из десятичной системы счисления в двоичную).

При переводе *по первому способу* пользуются следующим правилом: *исходное число X и последовательно получающиеся частные делят на основание новой системы счисления q до получения частного, меньшего q . Остатки, полученные при делении, являются разрядами числа в новой q -ричной системе счисления. Последний остаток, за который принимается последнее частное, является **старшим** разрядом числа, т.е. для записи числа X_q остатки записываются в порядке, обратном их получению.*

В качестве примера переведем десятичное число $X = 39$ в двоичную и 16-ричную СС.

Процесс перевода поясняется табл. 4.3 и 4.4.

Таблица 4.3

Перевод $X_{10} \rightarrow X_2$

Исходное число и частные		q		Частное		Остаток	Разряд
39	:	2	=	19	→	1	Млад- ший
19	:	2	=	9	→	1	↑
9	:	2	=	4	→	1	
4	:	2	=	2	→	0	
2	:	2	=	1	→	0	
				1	→	1	Стар- ший

Таблица 4.4

Перевод $X_{10} \rightarrow X_{16}$

Исходное чис- ло		q		Частное		Остаток	Разряд
39	:	16	=	2	→	7	Младший
				2	→	2	Старший

Таким образом, $X = 39 = 100111_2 = 27_{16}$.

Двоичную запись числа можно получить, имея его 16-ричное представление, если каждой 16-ричной цифре ($a_i \in \{0, 1, \dots, 9, A, B, C, D, E, F\}$) поставить в соответствие *тетраду* — обычную 4-разрядную двоичную последовательность $a_3a_2a_1a_0$, где $a_i \in \{0, 1\}$.

Например:

$$X = 27_{16} = \underbrace{0010}_2 \underbrace{0111}_7_{2-16} = 100111_2.$$

Для *обратного* перехода, от двоичного числа к 16-ричному, заданная двоичная последовательность разделяется на тетрады со стороны младших разрядов (недостающие дополняются нулями) и каждой из них ставится в соответствие 16-ричная цифра.

Например:

$$X = 10110011101_2 = \underbrace{0101}_5 \underbrace{1001}_9 \underbrace{1001}_D_{2-16} = \underbrace{5}_{16^2} \underbrace{9}_{16^1} \underbrace{D}_{16^0} h \leftarrow \text{вес разряда}$$

Десятичный эквивалент шестнадцатеричного числа можно найти по формуле (4.1), т.е.

$$X_{10} = 5 \cdot 16^2 + 9 \cdot 16^1 + 13 \cdot 16^0 = 1437.$$

Для перевода по *второму* способу необходимо определить число двоичных разрядов, требуемых для размещения заданного десятичного числа. Из неравенства $2^5 < 39 < 2^6$ видно, что пяти двоичных разрядов не хватает для размещения числа 39 (требуется шесть разрядов).

Тогда X можно представить в виде последовательности коэффициентов a_i , $i = 0, 5$, каждый из которых имеет свой вес, т.е. $X = a_5 a_4 a_3 a_2 a_1 a_0$. Затем на основании выражения (4.1) исключают ненужный вес путем выбора нулевых значений для соответствующих коэффициентов a_i (в рассматриваемом примере $a_4 = a_3 = 0$).

Тогда

$$X = 39 = \underset{a_5}{1 \cdot 2^5} + \underset{a_4}{0 \cdot 2^4} + \underset{a_3}{0 \cdot 2^3} + \underset{a_2}{1 \cdot 2^2} + \underset{a_1}{1 \cdot 2^1} + \underset{a_0}{1 \cdot 2^0} = 100111_2.$$

Перевод дробных чисел

Перевод из десятичной в q -ричную систему счисления. При переводе дробных чисел говорят о переводе с заданной точностью и используют метод последовательного умножения на основание новой системы счисления q .

Правило. Исходное число X и получающиеся дробные части последовательно умножают на q до получения дробной части, равной нулю (при точном переводе), или до получения нужного числа цифр q -ричной записи (при переводе с заданной точностью). Число X в q -ричной системе образуется как последовательность целых частей произведений.

В качестве примера найдем двоичный эквивалент правильной десятичной дроби $X = 0.375$.

$$\begin{array}{r} \times 0.375 \\ \hline 2 \\ \hline 0.750 \end{array} \quad \begin{array}{r} \times 0.750 \\ \hline 2 \\ \hline 1.500 \end{array} \quad \begin{array}{r} \times 0.500 \\ \hline 2 \\ \hline 1.000 \end{array}$$

$X = 0.011_2$

При переводе дробных чисел, содержащих знаменатель, кратный степени двойки (2^n), числитель переводится как целое число по соответствующему правилу, а затем точка переносится влево на n разрядов.

Например, $X = 6/8 = 6/2^3 = 0.110_2$; $X = 6/16 = 6/2^4 = 0.0110_2$; $X = 13/32 = 13/2^5 = 0.01101_2$.

Перевод смешанных чисел

При переводе смешанного числа целая и дробная части его переводятся отдельно по соответствующим правилам, а затем результаты их перевода соединяются через точку.

Например, $X = 39.375 = 100111.011_2$.

Перевод из q -ричной системы счисления в десятичную. Перевод основан на вычислении значений числа по формуле (4.1). Так, например, десятичный эквивалент двоичного числа $X = 1011.101_2$ вычисляется как сумма $X_{10} = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 + 1 \cdot 2^{-1} + 0 \cdot 2^{-2} + 1 \cdot 2^{-3} = 11 \frac{5}{8}$.

Десятичный эквивалент шестнадцатеричного числа $X = 3C_{16}$ вычисляется аналогично:

$$X_{10} = 3 \cdot 16^1 + 12 \cdot 16^0 = 48 + 12 = 60.$$

Перевод чисел из СС с основанием q_1 в СС с основанием q_2 . Перевод числа X из системы счисления с произвольным основанием q_1 в систему счисления с основанием q_2 осуществляется через десятичную систему счисления (кроме переводов $2 \leftrightarrow 8$, $2 \leftrightarrow 16$), т.е. по схеме $q_1 \rightarrow 10 \rightarrow q_2$.

Двоично-кодированная десятичная система счисления. Используется для представления в ЭВМ десятичных чисел путем замены каждой десятичной цифры соответствующей двоичной тетрадой или BCD-кодом.

Для кодирования цифр 0—9 чаще всего используется BCD-код с весами 8-4-2-1 (табл. 4.5).

Таблица 4.5

Кодирование цифр в коде с весами 8-4-2-1

Десятичная цифра	0	1	2	3	4	5	6	7	8	9
BCD-код	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001

Десятичное число $X = 4295$, представленное в BCD-коде, имеет вид:

$$X = 4295 = 0100\ 0010\ 1001\ 0101_{2-10}.$$

В такой форме представляются десятичные числа в ЭВМ при выполнении арифметических операций в десятичной СС.

Обратный перевод: $X_{2-10} = 0011\ 0111\ 1000 = 378$.

Представление числовых данных в ЭВМ

В вычислительных машинах применяются две формы представления двоичных чисел:

- ♦ *естественная*, или форма с фиксированной точкой (ФФТ);
- ♦ *полулогарифмическая* (нормальная), или форма с плавающей точкой (ФПТ).

Представление чисел в форме с ФТ. В форме с фиксированной точкой все числа представляются в виде последовательности цифр с *постоянным* для всех чисел положением точки: *после младшего разряда* для *целых чисел* и *перед старшим разрядом* для *дробных чисел*. Их разрядные сетки представлены соответственно на рис. 4.14.



Рис. 4.14. Разрядные сетки чисел с фиксированной точкой
а — после младшего разряда; б — перед старшим разрядом

Целое число X , например $X=100111_2$, может интерпретироваться как:

- ♦ целое без знака;
- ♦ целое со знаком.

В *целом без знака* все двоичные разряды числа являются значащими, т.е. имеют соответствующий вес.

Например, в шестиразрядном числе $X=100111_2$ все шесть двоичных разрядов являются значащими:

$$\begin{array}{cccccc}
 \text{Старший} & & & & & \text{Младший} \\
 \downarrow & & & & & \downarrow \\
 5 & 4 & 3 & 2 & 1 & 0 \quad \text{— разряд} \\
 x = 1 & 0 & 0 & 1 & 1 & 1_2 \\
 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \quad \text{— вес двоичного разряда,}
 \end{array}$$

и тогда десятичный эквивалент числа (X_{10}) можно определить по формуле

$$X_{10} = \pm \sum_{i=0}^{n-1} a_i q^i.$$

Поскольку $q = 2$, а коэффициенты двоичного ряда $a_i \in \{0, 1\}$, то:

$$X_{10} = \underset{a_5}{1} * 2^5 + \underset{a_4}{0} * 2^4 + \underset{a_3}{0} * 2^3 + \underset{a_2}{1} * 2^2 + \underset{a_1}{1} * 2^1 + \underset{a_0}{1} * 2^0 = 32 + 4 + 2 + 1 = 39.$$

В *целом со знаком* (форма представления чисел наиболее распространенная) *старший* $n - 1$ разряд используется для представления *знака* числа (рис. 4.15): *единица* в знаковом разряде — *признак отрицательного числа*, *0* — *положительного*.

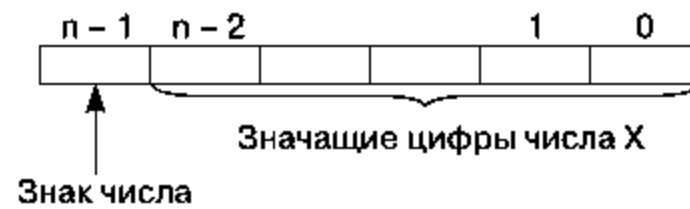


Рис. 4.15. Разрядная сетка целого числа со знаком

Форматы целых чисел в ЭВМ. *Форматом числа* называют его представление в конкретной *разрядной сетке* ЭВМ, под которой понимают набор двоичных разрядов для представления *машинного слова* в конкретной ЭВМ.

Для представления *целых чисел* в памяти компьютера используют три машинных формата: byte, word, long. Форматы целых чисел со знаком (рис. 4.16, а) отличаются от форматов беззнаковых чисел (рис. 4.16, б) тем, что старший бит разрядной сетки отводится под знак числа, который кодируется следующим образом:

$$\text{Sign(Sg)}_{(\text{знак})} = \begin{cases} 0, & \text{если } X \geq 0; \\ 1, & \text{если } X < 0. \end{cases}$$

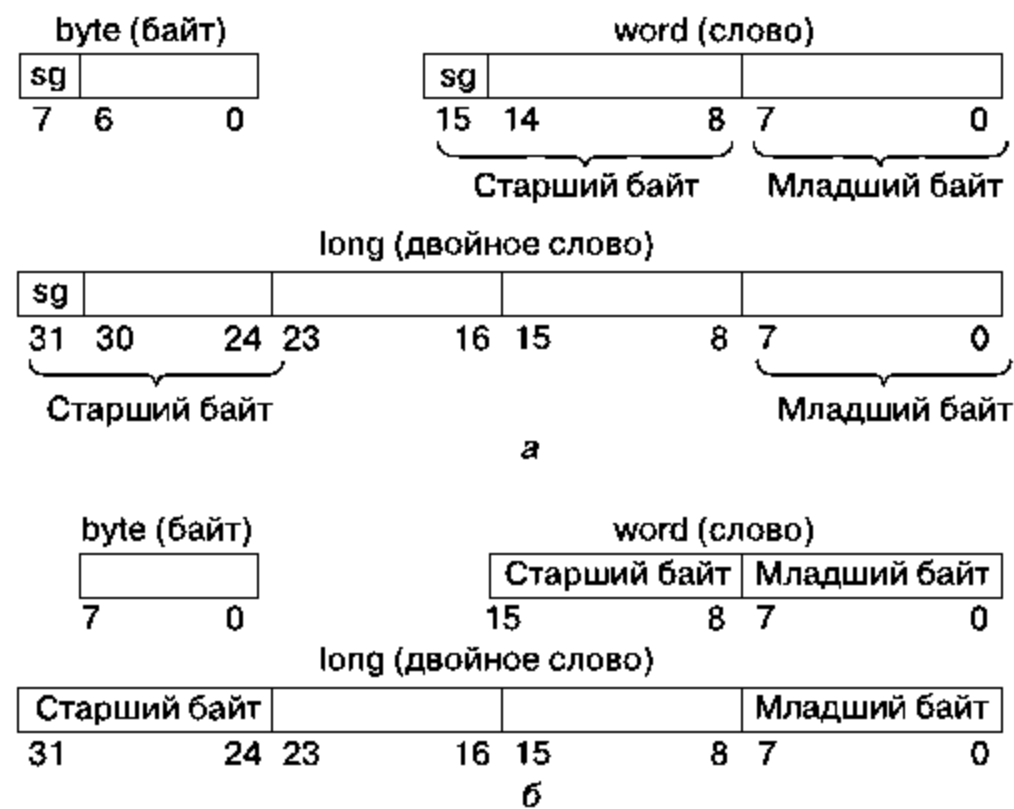


Рис. 4.16. Форматы чисел

а — со знаком; б — без знака

В современных ЭВМ естественная форма представления используется как вспомогательная и *только для целых чисел*.

Диапазон представления чисел с ФТ. Этот диапазон определяется неравенством:

$$-X_{\max} \leq X \leq X_{\max},$$

где $X_{\max}$ — наибольшее значение числа X .

На числовой оси ему соответствуют две области: *положительных* и *отрицательных* чисел (рис. 4.17).

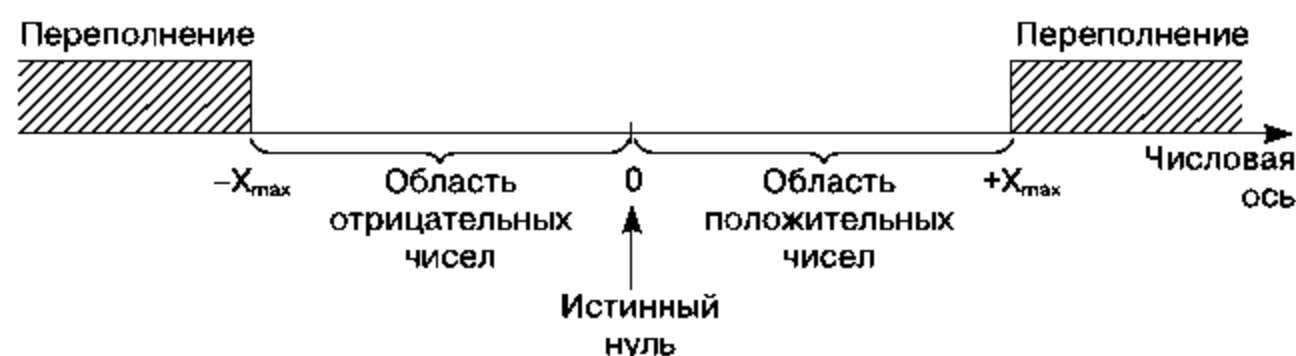


Рис. 4.17. Расположение чисел на числовой оси

Если в результате операции получится число, выходящее за допустимый диапазон, происходит *переполнение* разрядной сетки, что нарушает нормальное функционирование машины.

При n -разрядной сетке $X_{\max} = 2^n - 1$ и целое *беззнаковое* число

$$X = \underbrace{X_{n-1}, X_{n-2}, \dots, X_0}_{\text{Значащие разряды}}$$

находится в диапазоне

$$0 \leq X \leq 2^n - 1.$$

При $n = 8$, например, X находится в диапазоне $0 \leq X \leq 255$.

Для *целых чисел со знаком* (рис. 4.18)

$$X = \underbrace{X_{n-1}}_{\text{sg}} \underbrace{X_{n-1} X_{n-2} \dots X_1 X_0}_{\text{Значащие разряды}}$$

максимальное положительное число $+X_{\max} = \underbrace{0}_{\text{sg}} \underbrace{111\dots1}_{n-1 \text{ разрядов}} = 2^{n-1} - 1,$

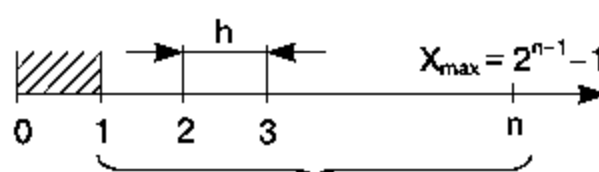


Рис. 4.18. Область положительных чисел со знаком

а максимальное отрицательное $-X_{\max} = \underbrace{1}_{\text{sg}} \underbrace{00\dots0}_{n-1 \text{ разрядов}} = -2^{n-1}$, диапа-

зон X определяется неравенством

$$-2^{n-1} \leq X \leq 2^{n-1} - 1.$$

Поскольку старший бит числа занят знаком, диапазон положительных чисел сокращается в 2 раза.

Например, для $n = 8$ диапазон чисел со знаком будет

$$-128 \leq X \leq 127.$$

Для *правильных дробей* диапазон определяется неравенством

$$2^{-n} \leq |X| \leq 1 - 2^{-n}.$$

Правильные дроби используются для представления *мантиссы* числа с плавающей точкой.

Точность представления чисел с ФТ. Абсолютная погрешность. Два любых соседних числа, представленных на числовой оси (рис. 4.19, а) отличаются друг от друга на величину $h = 1$, называемую *шагом дискретизации*.

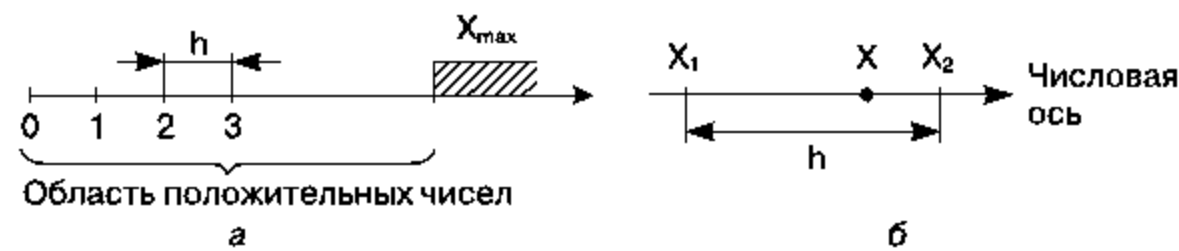


Рис. 4.19. Распределение положительных чисел с ФТ:

а — область целых положительных чисел; б — представление числа X в области дискретных значений

Величина h (рис. 4.19, б) определяет максимальную *абсолютную погрешность* $\Delta_{\max} = h = 1$ представления числа X (без учета округления).

Тогда *относительная погрешность* δ целого числа без знака составит:

$$\delta = \frac{\Delta}{X};$$

при этом $\delta_{\min} = \frac{\Delta}{X_{\max}} = \frac{1}{2^n - 1} \approx 2^{-n}$, например, при $n = 8$ $\delta_{\min} =$

$$= \frac{1}{2^8 - 1} = \frac{1}{255}; \quad \delta_{\max} = \frac{\Delta}{X_{\min}} = \frac{1}{1} \cdot 100\% = 100\%.$$

Из этого следует, что для чисел с ФТ необходимо избегать работы в области малых чисел.

Относительная погрешность δ целого числа со знаком равна:

$$\delta_{\min} = \frac{\Delta}{X_{\max}} = \frac{1}{2^{n-1} - 1} \approx 2^{-(n-1)}.$$

Так, при $n = 8$ $\delta_{\min} = \frac{1}{2^{8-1} - 1} = \frac{1}{127}$, т.е. δ увеличивается в 2 раза

по сравнению с погрешностью для числа без знака.

Поскольку разрядность n ограничена, то *диапазон и точность* чисел, представленных в форме с ФТ, являются *ограниченными*, что считается недостатком такой формы.

Представление чисел в форме с ПТ. *Нормальная* форма представления чисел с плавающей точкой (ПТ) является основной в современных ЭВМ.

Каждое число с плавающей точкой представляется в виде *двух* групп цифр. Первая группа называется *мантиссой*, а вторая — *порядком*. Абсолютная величина мантиссы должна быть меньше 1, а порядок — целым числом. В общем случае число в форме с ПТ может быть представлено в виде:

$$X = \pm M_x q^{\pm P_x}, \quad (4.2)$$

где M_x — *мантисса* ($|M_x| < 1$);

P_x — *порядок* (целое число), который определяет истинное место точки в записи мантиссы. Значение порядка определяется числом разрядов, на которое необходимо сдвинуть точку;

q — *основание системы счисления*.

Мантисса (значащая часть числа) обычно представляется *правильной дробью*:

$$M_x = \pm 0.d_1 d_2 \dots d_n.$$

Условие $d_1 \neq 0$ является характерным признаком *нормализованного* числа в форме с ПТ. При этом обеспечивается запись максимального количества значащих цифр числа, т.е. максимальная точность его представления. *Нормализованное* представление используется для однозначной записи вещественных чисел. *Нормализованная мантисса* двоичного числа представляется десятичным числом, лежащим в диапазоне $2^{-1} \leq |M_x| < 1$.

Пример. Целое число $X = -21 = -10101_2$ представить в форме с ПТ.
 Место точки

Место точки
 $\downarrow$
 а) $X = -0.10101 \cdot 2^{+5} = \underbrace{-0.10101}_{M_x} \cdot \underbrace{10^{+101}}_q$;

$X = -0.10101 \cdot 2^{+5} = -0.10101 \cdot 10^{+101}$ или $X = -0.10101e5$;

Место точки
 $\downarrow$
 б) $X = -1.0101 \cdot 2^{+4} = -1.0101 \cdot 10^{+100}$ или $X = -1.0101e4$, где e — степень двойки (знак перед e не ставится, если e — положительное число). Знак числа определяется знаком мантиссы.

Из примера видно, что в форме с ПТ точка переместилась («уплыла») на пять разрядов влево, чтобы оказаться перед старшей значащей единицей (а) либо сразу после нее (б), уменьшив тем самым число в 2^5 раз (а) и в 2^4 раз (б) соответственно.

Мантисса числа X записывается соответственно как:

а) $M_x = -0.10101_2$ или в общем виде $M_x = \overset{1 \ 0 \ 1 \ 0 \ 1}{\underset{\uparrow \text{Место точки}}{d_0 d_1 d_2 d_3 d_4}}$, где

$d_i \in \{0, 1\}$;

б) $M_x = -1.0101_2$ или $M_x = \underset{\uparrow \text{Место точки}}{d_0} \cdot \underbrace{d_1 d_2 d_3 d_4}_{\text{Часть мантиссы, хранящейся в памяти}}$

Разряд d_0 в нормализованных числах всегда равен 1 и в памяти компьютера не хранится («скрытый бит»), а подразумевается. Таким образом, мантисса, хранящаяся в памяти компьютера, — всегда число дробное (рис. 4.20).

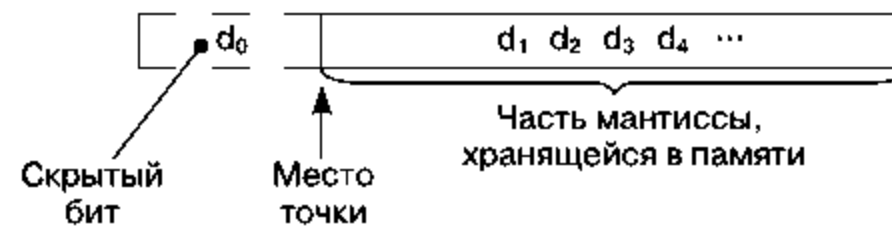


Рис. 4.20. Представление n -разрядной мантиссы

Форма представления чисел с ПТ позволяет значительно *увеличить* диапазон и точность представления чисел при *той же* длине разрядной сетки, что и для чисел с ФТ.

Представление чисел с ПТ в памяти компьютера. Вещественные числа в отличие от целых представляются и хранятся в памяти в *прямом коде*, их представление стандартизовано. Как правило, такие числа представляются в компьютерах в трех машинных форматах:

- ♦ одинарной точности (4 байта);
- ♦ двойной точности (8 байт);
- ♦ расширенной точности (10 байт), имеющих одинаковую структуру (рис. 4.21).

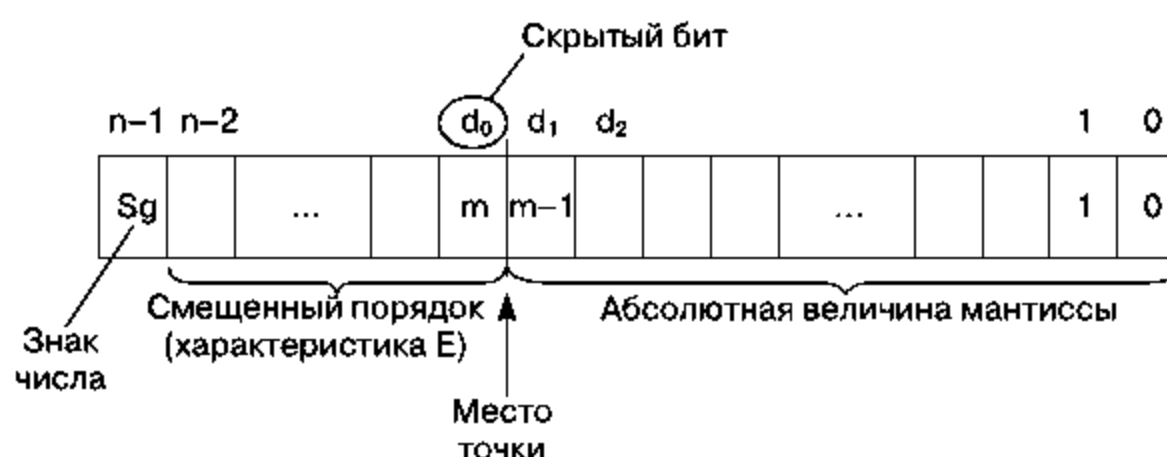


Рис. 4.21. Структура представления вещественных чисел

При этом порядок P_x n -разрядного нормализованного числа задается *смещенным кодом*, который используется для *упрощения операций* над порядками чисел с ПТ (сравнение, сложение, вычитание) и позволяет выполнять их как над беззнаковыми числами, что в итоге упрощает операцию сравнения самих нормализованных чисел. Увеличение скорости сравнения чисел с ПТ важно в алгоритмах с большим числом сравнений, например в алгоритмах сортировки.

Разрядность *порядка* определяет *диапазон* представимых чисел при заданном формате, а разрядность *мантиссы* — *точность* числа: чем *больше* разрядов отводится под нее, тем *выше* точность представления числа.

Некоторые фирмы-разработчики ПО используют нестандартные форматы, в частности в компиляторе языка Pascal фирмы Borland используется формат длиной 6 байтов.

В различных алгоритмических языках принятые стандарты описываются разными ключевыми словами. В языке Turbo Pascal определены следующие типы вещественных чисел: single, real, double, extended, comp. Данные о первых трех из них приведены в табл. 4.6.

Таблица 4.6

Типы вещественных чисел в языке Turbo Pascal

Тип	Длина, байт	Мантисса (десятичных значащих цифр)	Диапазон десятичного порядка
Single	4	7—8	–45 ... +38
Real	6	11—12	–39 ... +38
Double	8	15—16	–324 ... +308

Во всех типах мантисса хранится в прямом коде (ПК), порядок задается в так называемой *смещенной форме*. Смещенный порядок Е (характеристика) равен истинному порядку П, увеличенному на значение смещения D:

$$E = П + \underbrace{D}_{\text{Смещение}}$$

Величина смещения D выбирается равной 127 для single, 129 — для real и 1023 — для double так, чтобы характеристика была целым положительным числом.

Смещенный порядок Е можно считать целым положительным или беззнаковым числом:

$$E_{\min} = 0, E_{\max} = 255.$$

Разрядность характеристики Е определяет диапазон чисел, представленных в этом формате.

Размеры полей и их размещение в каждом конкретном формате приведены соответственно на рис. 4.22—4.24.



Рис. 4.22. Формат single

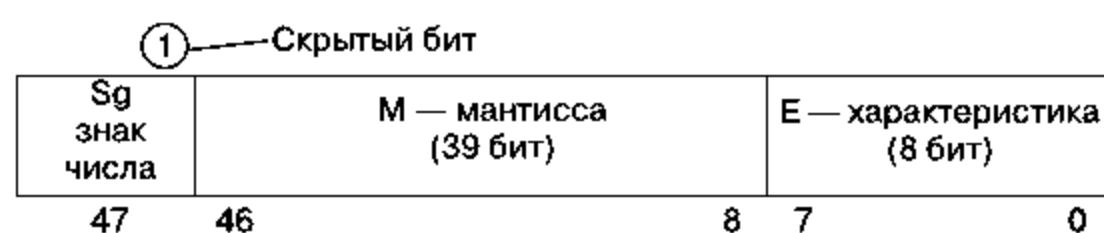


Рис. 4.23. Формат real



Рис. 4.24. Формат double

Нормализованное число *одинарной* точности в формате single представляет собой 32-разрядное число со знаком, 8-разрядным смещенным порядком и мантиссой, занимающей 23 бита (с 22 по 0). При этом, хотя для мантиссы отведено 23 разряда, в операциях участвуют 24 разряда, так как старший разряд мантиссы нормализованного числа не хранится (он всегда равен 1; это так называемый *скрытый бит*, который при аппаратном выполнении операции автоматически восстанавливается и учитывается).

Особенности формата single:

- ♦ невозможно представить числа, которые ненормализованы (за исключением нулевого порядка);
- ♦ скрытый бит *не позволяет* в этом формате представить нуль, и он должен кодироваться как специальное значение (нуль кодируется с нулевым смещенным порядком и мантиссой).

Диапазон чисел, представленных в формате single, составляет $\pm 10^{\pm 38}$, а точность — 7—8 десятичных разрядов (см. табл. 4.6).

Пример 1. Представить число $X = -237.625$ в формате single.

Двоичный код числа:

$$X = -11101101.101_2.$$

В форме с ПТ число X запишется в виде

$$X = -\underbrace{1.1101101101}_{M_x} \cdot 2^{+7} \quad \leftarrow \begin{matrix} \uparrow \\ \Pi_x \end{matrix}$$

Смещенный порядок $E = \Pi_x + 127 = 7 + 127 = 134$. В формате single X представляется в виде

$$\begin{array}{c} \textcircled{1} \leftarrow \text{скрытый бит} \\ \underbrace{1}_{sg} \quad \underbrace{1000\ 0110}_E \quad \underbrace{1101\ 1011010\dots 0}_{M_x} \end{array}$$

Разбив двоичную последовательность на тетрады, получим 16-ричное представление числа X : C3 6D A0 00h.

Пример 2. Представить число $X = +0.125$ в формате single.

Двоичный код числа:

$$X = +0.001_2.$$

В форме с ПТ число X запишется в виде

$$X = \underbrace{1.0}_{\text{нормализованная } M_x} \cdot 2^{-3 \leftarrow P_x}$$

Нормализованная M_x

Смещенный порядок $E = P_x + 127 = -3 + 127 = 124$. В формате single X представляется в виде:

$$\underbrace{0}_{sg} \underbrace{0111\ 1100}_{E} \underbrace{00000000000\dots 0}_{M_x}$$

Разбив двоичную последовательность на тетрады, получим 16-ричное представление числа X : 3E 00 00 00h.

В формате *real* (см. рис. 4.23) смещенный порядок также занимает 8 бит (разряды с 7 по 0). Размер поля для хранения мантииссы занимает 39 бит (с 46 по 8), старший бит мантииссы не хранится.

В формате *double* (двойной точности) (см. рис. 4.24) смещенный порядок занимает 11 бит (разряды с 62 по 52), мантиисса — 52 бита (с 51 по 0) и старший разряд ее также не хранится.

Расширенный формат (РТ) позволяет хранить ненормализованные числа в виде 80-разрядного числа со знаком: 15-разрядным смещенным порядком, 64-разрядной мантииссой и величиной смещения $D = 16\ 383$. Вещественные числа в формате РТ (рис. 4.25) имеют явный бит $d_0 = 1$, занимающий 63-й разряд мантииссы.



Рис. 4.25. Расширенный формат

Значение числа X с ПТ равно

$$X = (-1)^S \cdot M \cdot 2^{(E-D)},$$

где разрядность мантииссы $m = 23, 39, 52, 64$.

Следует отметить, что вещественный формат (см. рис. 4.21) с m -разрядной мантииссой дает возможность абсолютно точно представлять m -разрядные целые числа, т.е. любое двоичное целое число, содержащее не более m разрядов, может быть без искажений преобразовано в вещественный формат.

О представлении числа 0 в форматах с ПТ. Число 0 в форматах с ПТ может быть представлено как:

- а) *ненормализованный нуль* — нули во всех разрядах М и П, т.е.
 $0.000\dots 0 \quad 0.000\dots 0.$

Истинный нуль — нули во всех разрядах, включая скрытый бит;

- б) *нормализованный нуль* — нули во всех разрядах М, а $P = -|P_{\max}|$, т.е.

$$0.000\dots 0 \quad 1.111\dots 1;$$

$$-P_{\max} = 1.000\dots 0 \text{ или } 1.111\dots 1.$$

Диапазон представления чисел с ПТ. Ограничимся рассмотрением диапазонов нормализованных *положительных* чисел.

В старых ЭВМ использовался и стал каноническим классический двоичный формат для представления чисел с ПТ, содержащий четыре поля (рис. 4.26).

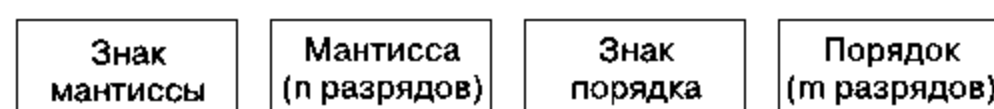


Рис. 4.26. Классический двоичный формат

Из общей формы записи числа с ПТ (выражение (4.2)) следует, что *минимальное* значение числа X будет получено при минимальном значении мантииссы $M_x = 0.100\dots 0_2$ и максимальном (по абсолютному значению) отрицательном порядке $|P_x|_{\max} = 1.111\dots 1_2$, т.е. $X_{\min}$ в классическом двоичном формате представляется в виде:

$$X_{\min} : \underbrace{0}_{\text{Знак } M_x} \underbrace{100\dots 00}_{M_{x_{\min}}} \underbrace{1}_{\text{Знак } P_x} \underbrace{11\dots 11}_{|P_{x_{\max}}| = -(2^m - 1)}$$

и вычисляется следующим образом:

$$X_{\min} = M_{\min} \cdot 2^{-P_{\max}} = \frac{M_{\min}}{2^{+P_{\max}}} = 2^{-1} \cdot 2^{-(2^m - 1)} = 2^{-2^m}.$$

Максимальное значение числа $X_{\max}$ будет получено, если значения мантииссы и положительного порядка будут максимальными, т.е. в формате рис. 4.26 $X_{\max}$ будет представлено в виде

$$X_{\max} : \underbrace{0}_{\text{Знак } M_x} \underbrace{11\dots 11}_{M_{x_{\max}} = (1 - 2^{-n})} \underbrace{0}_{\text{Знак } P_x} \underbrace{11\dots 11}_{+P_{x_{\max}} = 2^m - 1}$$

$\uparrow$
 ≈ 1

и вычисляется следующим образом:

$$X_{\max} \approx \underbrace{1}_{M_{x_{\max}}} \cdot 2^{\underbrace{(2^n - 1)}_{P_{x_{\max}}}} \approx 2^{2^n - 1}.$$

Таким образом, *диапазон представимых чисел с ПТ* определяется числом *разрядов n* , отведенных *под порядок*, а в современных стандартных форматах — числом разрядов, отведенных под смещенный порядок E .

Точность представления чисел. Любому числу X в формате с ПТ $X = M_x \cdot 2^{P_x}$ соответствует определенная точка на числовой оси. Очевидно, что ближайшие к нему представимые положительное и отрицательное числа $X_{\text{бл}}^-$ и $X_{\text{бл}}^+$ отличаются на единицу младшего разряда мантиссы (рис. 4.27):

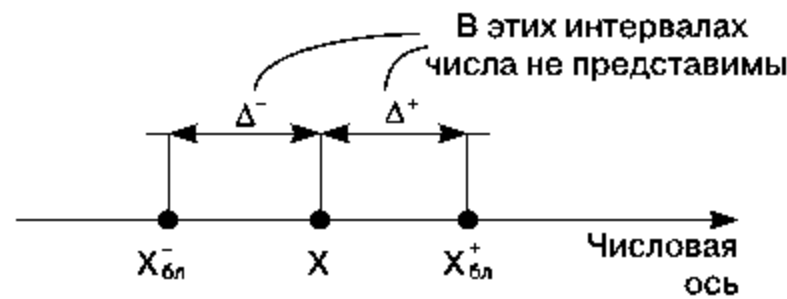


Рис. 4.27. Соседние числа с ПТ на числовой оси

$$X_{\text{бл}}^- = (M - 2^{-n}) \cdot 2^P;$$

$$X_{\text{бл}}^+ = (M + 2^{-n}) \cdot 2^P.$$

Значение разности между соседними представимыми числами составляет *абсолютную погрешность представления числа* $\Delta = h = X_{\text{бл}}^+ - X = +2^{-n} \cdot 2^P$, которая зависит только от величины порядка.

Однако при вычислениях более важной является *относительная погрешность*:

$$\delta = \frac{\Delta}{X} = \frac{2^{-n} \cdot 2^P}{M \cdot 2^P} = \frac{2^{-n}}{M}. \quad (4.3)$$

Подставляя в выражение (4.3) минимальное и максимальное значения мантиссы, получаем *минимальное и максимальное значения относительной погрешности* соответственно:

$$\delta_{\min} = \frac{2^{-n}}{M_{\max}}; \quad \delta_{\max} = \frac{2^{-n}}{M_{\min}}. \quad (4.4)$$

Из выражения (4.4) видно, что *относительная погрешность* представления чисел с ПТ почти не зависит от величины числа и определяется *только разрядностью мантиссы*.

Ниже приводятся примеры расчета относительной погрешности для формата одинарной точности (32 бита) в современном и старом стандартах соответственно:

$$\text{В формате single: } \delta_{\min} = \frac{2^{-24}}{1 - 2^{-24}} \approx 2^{-24}; \quad \delta_{\max} = \frac{2^{-24}}{1 \cdot 2^{-4}} \approx 2^{-20}.$$

$$\text{В классическом формате: } \delta_{\min} = \frac{2^{-24}}{1 - 2^{-24}} \approx 2^{-24}; \quad \delta_{\max} = \frac{2^{-24}}{2^{-1}} \approx 2^{-23}.$$

Достоинствами чисел с ПТ являются:

- ♦ большой диапазон представимых чисел;
- ♦ высокая относительная точность, практически не зависящая от величины числа. Платой за это является более *сложная структура числа*, представленного в виде двух частей M_x и P_x , что усложняет методику выполнения операций (отдельные действия над M_x и P_x). Увеличение числа операций ведет к снижению скорости выполнения арифметических операций над числами с ПТ.

При выполнении операций над командами, когда нет опасности переполнения, но требуется выполнять операции *быстро*, используют формат с ФТ. При обработке очень больших или очень малых чисел используют ФПТ, т.е. программист выбирает форму представления чисел, исходя из особенностей конкретной задачи.

Представление символьной информации в ЭВМ. Символьная (алфавитно-цифровая) информация хранится и обрабатывается в ЭВМ в форме цифрового кода, т.е. каждому символу ставится в соответствие отдельное двоичное слово — код. При выборе метода кодирования руководствуются объемом и способами обработки символьной информации. Так как многие типы информации содержат в значительном объеме цифровую информацию, применяются две системы кодирования: символьной информации и десятичных чисел.

Среди наборов символов наибольшее распространение получили знаки кода ASCII (American Standard Code for Information Interchange — стандартный американский код для обмена информацией).

ASCII (или КОИ-7) — семиразрядный специальный код, обеспечивающий 128 различных битовых комбинаций, находит широкое применение в мини- и микроЭВМ, в том числе в персональных компьютерах.

Пример кодирования некоторых символов приведен в табл. 4.7.

Таблица 4.7

Пример кодирования символов

Символ	ASCII-код
0	0110000 или 30_{16}
1	0110001 или 31_{16}
2	0110010 или 32_{16}
.....	
9	0111001 или 39_{16}
A	1000001 или 41_{16}
B	1000010 или 42_{16}
C	1000011 или 43_{16}

Для контроля на четность (или нечетность) к семи битам ASCII-кода часто добавляют старший восьмой бит. Поэтому символы передаются байтами.

В настоящее время широкое применение находят и другие виды кодировок, например система кодирования Windows-1251, используемая на большинстве ПК, работающих на платформе Windows, КОИ-8 — широко распространенная система кодирования в компьютерных сетях и др.

Машинные коды

Для представления (кодирования) чисел со знаком используются специальные машинные коды. Причем важен такой способ кодирования чисел, при котором их знак и значение двоичные цифры при сложении (и вычитании) кодов чисел обрабатывались бы одинаково. Этим условиям удовлетворяет *дополнительный код*, который используется для хранения чисел и выполнения арифметических операций над ними. Следует помнить, что ЭВМ работает не с числами, а с их кодами. Чтобы отличить запись числа от записи кода, в последнем случае используют квадратные скобки. Например, дополнительный код числа X обозначают в виде $[X]_{\text{дк}}$. Еще одной особенностью, отличающей запись кода от записи числа, является *точка*, отделяющая знаковый бит от значащих разрядов. Точка не является частью кода, в машинной сетке не ставится и используется лишь при ручной записи кодов.

Прямой код (ПК). Прямой код n -разрядного двоичного числа $X = \pm 0 X_{n-2} X_{n-3} \dots X_1 X_0$ определяется следующим образом

$$[X]_{\text{пк}} = \begin{cases} X, & \text{если } X \geq 0, \\ 2^{n-1} + |X|, & \text{если } X < 0, \end{cases}$$

т.е. прямой код положительного числа есть само число, а отрицательного — единица в знаковом разряде и модуль числа в значащих разрядах.

Пример. Найти ПК числа $X = \pm 17$ и представить его в формате байта.

$$X = +17 = +0001\ 0001_2, [X]_{\text{ПК}} = 0.0010001_2 = 11\text{h};$$

$$X = -17 = -0001\ 0001_2, [X]_{\text{ПК}} = 1.0010001_2 = 91\text{h}.$$

ПК используется крайне редко (для представления мантиисы числа, операндов при делении и умножении).

Обратный код (ОК). Обратный код n -разрядного двоичного числа $X = 0\ X_{n-2}\ X_{n-3} \dots X_1\ X_0$ определяется следующим образом:

$$[X]_{\text{ОК}} = \begin{cases} X, & \text{если } X \geq 0, \\ |\bar{X}|, & \text{если } X < 0, \end{cases}$$

где $|\bar{X}| = \bar{0}\ \bar{X}_{n-2}\ \bar{X}_{n-3} \dots \bar{X}_1\ \bar{X}_0$ — инверсия модуля числа X ;

$\bar{X} = (q-1) - X_i$ — инверсия числа X_i . Для $q = 2$ инверсии: $0 = 1, 1 = 0$.

Так, обратный код отрицательного числа $X = -0011\ 0110_2$

$$[X]_{\text{ОК}} = 1.100\ 1001_2 = \text{C9h}.$$

Выполнение арифметических операций над обратными кодами сопряжено с рядом трудностей, поэтому ОК в современных ЭВМ практически не используется.

Дополнительный код (ДК). Дополнительный код n -разрядного двоичного числа $X = \pm 0\ X_{n-2}\ X_{n-3} \dots X_1\ X_0$ определяется следующим образом:

правило 1

$$[X]_{\text{ДК}} = \begin{cases} X, & \text{если } X \geq 0, \\ |\bar{X}| + 1, & \text{если } X < 0, \end{cases}$$

либо, если $X < 0$, то

$$[X]_{\text{ДК}} = 0 - |\bar{X}|$$

(правило 2 применяется реже).

Для отрицательных двоичных чисел есть еще одно правило образования ДК: исходное число X просматривается справа налево со стороны младших разрядов. Встретившиеся до первой единицы нули и сама единица остаются без изменений, остальные разряды инвертируются.

Так, если число $X = \pm 01011_2$, то ДК положительного и отрицательного X соответственно равны:

$$[+X]_{\text{ДК}} = 0.1011_2; [-X]_{\text{ДК}} = 10100_2 \leftarrow |\bar{X}|, \text{ т.е. } [-X]_{\text{ДК}} = 1.\underbrace{0101_2}_{\text{«хвост»}} + \frac{1}{1.0101_2}$$

Длина числа в ДК может быть увеличена до любого количества разрядов путем *копирования* (тиражирования) его *знакового разряда слева*. При этом заданное число X оказывается «хвостом» двоичной последовательности требуемой длины в 8, 16, 32, ... бит. Например, приведенное число X в формате байта (8 бит) и слова (16 бит) имеет соответственно вид

$$[X]_{\text{ДК}} = 1.111 \underbrace{0101_2}_{\text{«хвост»}} = \text{F5h} = 1.111 \ 1111 \ 1111 \ \underbrace{0101_2}_{\text{«хвост»}} = \text{FFF5h},$$

где h — идентификатор 16-ричной системы счисления.

Первые два правила образования ДК могут быть использованы применительно к любой СС. Однако следует иметь в виду, что в системах счисления, отличных от двоичной, понятие *инверсия* заменяется на *дополнение до $q - 1$* .

Так, для $X = -0019_{16}$ $[X]_{\text{ДК}} = |\bar{X}| + 1 = \text{FFE6} + 1 = \text{FFE7}_{16}$.

В общем случае дополнение q -ричной цифры до $q - 1$ определяется как:

$$\bar{a}_q = (q - 1) - a_q.$$

Так, дополнения 16-ричных цифр до $q - 1$ будут следующими:

$$\begin{aligned} \bar{0} = \text{F}, \bar{1} = \text{E}, \bar{2} = \text{D}, \bar{3} = \text{C}, \bar{4} = \text{B}, \bar{5} = \text{A}, \bar{6} = 9, \bar{7} = 8, \\ \bar{9} = 6, \bar{\text{A}} = 5, \bar{\text{B}} = 4, \bar{\text{C}} = 3, \bar{\text{D}} = 2, \bar{\text{E}} = 1, \bar{\text{F}} = 0. \end{aligned}$$

Обратный переход (от кода к числу): $[X]_{\text{ДК}} \rightarrow X$. Переход от ДК к числу выполняется по тому же правилу 1, что и от числа к коду. Для отрицательных чисел $X = -([X]_{\text{ДК}} + 1)$.

Пример. Найти десятичный эквивалент числа X , представленного в ДК в формате байта (а) и слова (б):

$$\text{а) } [X]_{\text{ДК}} = \text{A3h} = 1.0100011_2; X = -01011101_2 = -5\text{Dh} = -(5 \cdot 16^1 + 13) = -93;$$

$$\text{б) } [X]_{\text{ДК}} = \text{FFA3h} = 1.1111111 \ 10100011_2 = -0000 \ 0000 \ 0101 \ 1101_2 = -005\text{Dh} = -93.$$

Из анализа рассмотренных машинных кодов видно, что *положительные* числа в прямом, обратном и дополнительном кодах представляются одинаково — двоичными кодами с цифрой 0 в знаковом разряде.

Отрицательные десятичные числа при вводе в машину обычно автоматически преобразуются в дополнительный или обратный двоичный код и в таком виде *хранятся, перемещаются и участвуют в операциях*. При выводе из машины происходит обратное преобразование кодов в отрицательные десятичные числа.

4.4.4. Команды в ЭВМ

Алгоритм решаемой на ЭВМ задачи представляется в виде *программы* — конструкции, подробно предписывающей процедуру решения задачи на машине.

Команда — управляющая информация, которая определяет, какую операцию необходимо выполнить и над какими *операндами* (числами, участвующими в операции). Каждая команда в закодированном виде содержит ответ на четыре вопроса:

- ♦ какая операция должна выполняться на данном шаге алгоритма;
- ♦ над какими операндами выполняется операция;
- ♦ куда надо поместить результат операции;
- ♦ куда (к какой команде) необходимо перейти после выполнения очередной операции.

Формат команды представлен на рис. 4.28.



Рис. 4.28. Формат команды

Команда ЭВМ содержит *операционную (ОЧ)* и *адресную части (АЧ)*. В операционной части команды (КОП) в двоичном коде определенным образом записывается тип выполняемой операции. По этому коду процессор ЭВМ определяет, какую операцию необходимо выполнить: пересылку, сложение, вычитание и др. В *адресной части* в общем случае прямо или косвенно содержится информация об *адресах* данных и результата, хранимых в памяти, т.е. об адресах ячеек памяти или регистров, где расположены операнды и результат. Так, например, двоичный код команды пересылки MOV AX,[0500H], инициирующей выполнение в МП 8086 операции пересылки операнда из ячейки памяти

с адресом 0500H в регистр AX ($AX \leftarrow [M]_{500}$), занимает три байта: один байт — код операции ($A1_{16}$) и два байта — адресная часть (0005_{16}):

$$\underbrace{10100001}_{\text{КОП}} \underbrace{00000000000000101}_2 = \underbrace{A1_2}_{\text{ОЧ}} \underbrace{0005_{16}}_{\text{АЧ}}.$$

При записи *кода* адреса в памяти *вначале* указывается *старшая часть* адреса (00_{16}), а затем — младшая (05_{16}). Для сокращения двоичной записи код команды представляют в 16-ричной системе счисления.

Код команды сложения ADD AX,AX занимает два байта ($03\ C0_{16}$).

Различают команды: *одноадресные*, *двухадресные* и *трехадресные* (рис. 4.29). Существуют также *безадресные* команды.

В *одноадресных* командах предполагается использование в ЦП регистра-аккумулятора, адрес которого подразумевается; в *двухадресных* — адрес результата совпадает с адресом одного из операндов, на место которого и записывается результат операции.

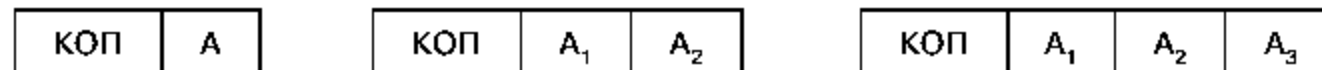


Рис. 4.29. Типы команд

4.5. ЛОГИЧЕСКИЕ ОСНОВЫ ПОСТРОЕНИЯ ЭВМ

4.5.1. Алгебра логики. Основные понятия

Для структурно-функционального описания *логических схем*, составляющих основу любого дискретного вычислительного устройства, ЭВМ или ВС в целом, используется аппарат *булевой алгебры*, или *алгебры логики*, созданной английским математиком Дж. Булем. На практике булеву алгебру применил К. Шеннон. Он разработал метод описания любой сети, состоящей из совокупности переключателей и реле, *математическими выражениями* и принцип их преобразования на основе правил *булевой алгебры*. Поскольку между релейными и современными электронными схемами существуют аналогии, аппарат булевой алгебры находит широкое применение для анализа, описания и проектирования последних.

Булева алгебра оперирует двумя понятиями — *истинность* и *ложность* высказывания. *Высказывание* — некоторое предложение, о котором можно однозначно сказать, *истинно* оно или *ложно*. Например, «сейчас идет снег». Это утверждение может быть истинным или лож-

ным. Истинное значение высказывания можно принимать за 1, а ложное — за 0, что позволяет рассматривать *высказывание* как двоичную переменную $x \in \{0, 1\}$. Такое сходство с двоичной системой счисления удобно при использовании алгебры логики для анализа и синтеза схем цифровых устройств. Во всех случаях в тех или иных точках логических схем сигналы двух различных уровней могут представляться двоичными символами $\{0, 1\}$. Поэтому множество элементов булевой алгебры называется бинарным, а сама алгебра — бинарной или переключательной. В алгебре логики все *высказывания* обозначаются буквами латинского алфавита — x, y, z . В дальнейшем этими буквами будем обозначать булевы переменные. Набор переменных x, y, z может рассматриваться как n -разрядный двоичный код, разрядами которого являются эти переменные.

Булева алгебра позволяет удобно оперировать с булевыми выражениями, описывающими те или иные электронные узлы, а также на формальном уровне *упрощать* их путем эквивалентных преобразований, благодаря чему появляются возможности создавать технически более совершенные электронные устройства.

Из простых высказываний с помощью определенных *операций* можно строить более *сложные высказывания*, т.е. *функции* алгебры логики. Операции булевой алгебры часто встречаются в программном обеспечении вычислительных устройств, где они используются для замены аппаратной логики на программную. Основными или *базовыми* операциями булевой алгебры являются операции «И» (AND), «ИЛИ» (OR) и «НЕ» (NOT).

Операция «И» называется логическим умножением или *конъюнкцией* и обозначается одним из знаков « $\cdot$ », « $\wedge$ » или « $\&$ (амперсанд)». Операция «ИЛИ» называется логическим сложением или *дизъюнкцией* и обозначается знаком « $+$ » или « $\vee$ ». Операция «НЕ» называется логическим отрицанием или *инверсией* и обозначается знаком « $'$ » или « $\neg$ ». При выполнении операций применяются отношение эквивалентности « $=$ » и скобки « $()$ », которые определяют порядок выполнения операций. Если скобок нет, то операции выполняются в следующей последовательности: логическое отрицание, логическое умножение и логическое сложение. Правила выполнения операций в алгебре логики определяются рядом аксиом, теорем и следствий. Основные соотношения (*теоремы*) булевой алгебры можно сформулировать в виде законов, представленных в табл. 4.8.

Таблица 4.8

Основные законы булевой алгебры

Формульное представление	Закон булевой алгебры
$\{0, 1\}' = \{1 0\};$ $x \vee 0 = x \cdot 1 = x; x \vee 1 = 1; x \cdot 0 = 0$	Операции с константами
$x \vee x = x;$ $x \cdot x = x$	Идемпотентность
$(x')' = x$	Двойное отрицание
$x \vee y = y \vee x;$ $x \cdot y = y \cdot x$	Коммутативность
$x \vee (x \cdot y) = x;$ $x \cdot (x \vee y) = x;$	Поглощение
$(x \vee y)' = x' \cdot y';$ $(x \cdot y)' = x' \vee y'$	Закон Моргана
$(x \vee y) \vee z = x \vee (y \vee z) = x \vee y \vee z;$ $(x \cdot y) \cdot z = x \cdot (y \cdot z) = x \cdot y \cdot z$	Ассоциативность
$x \vee y \cdot z = (x \vee y) \cdot (x \vee z);$ $x \cdot (y \vee z) = x \cdot y \vee x \cdot z$	Дистрибутивность

Истинность перечисленных законов легко доказывается методом *совершенной индукции*, т.е. перебором всех возможных случаев на основе таблицы истинности. Основные законы (см. табл. 4.8) дают правила преобразования булевых выражений, позволяющие получать эквивалентные выражения.

4.5.2. Представление функций алгебры логики

Функция в алгебре логики — алгебраическое выражение, которое содержит элементы алгебры логики x, y, z , связанные между собой операциями, определенными в этой алгебре.

Функция $f(x_1, x_2, \dots, x_n)$, которая, как и ее аргументы, может принимать только два значения, называется *булевой* или *переключательной* функцией (ПФ). *Задание* булевой функции означает, что каждому из возможных сочетаний аргументов поставлено в соответствие определенное значение f . При n аргументах общее число сочетаний $m = 2^n$. Так как каждому сочетанию аргументов соответствуют два значения функции (0, 1), общее число функций $F = 2^{2^n}$. *Переключательная функция* может быть задана *таблицей истинности* или *с помощью булевых выражений* в одной из нормальных форм — *совершенной дизъюнктивной* (Сов. ДНФ) или *совершенной конъюнктивной* (Сов. КНФ). *Нормальной формой* считают представление этих функций путем *суперпозиций* вспомогательных функций — *конституэнт единицы* K_j^n или *конституэнт нуля* L_j^n .

$$\text{Сов. ДНФ } f^n = \vee K_j^n, \quad j \in T_1;$$

$$\text{Сов. КНФ } f^n = \wedge L_j^n, \quad j \in T_0,$$

где T_1, T_0 — множество наборов, на которых функция принимает значение 1 и 0 соответственно.

Эти соотношения используются для синтеза логических (*переключательных*) функций и вычислительных схем.

4.5.3. Комбинационные схемы

Так как целью булевой алгебры является структурно-функциональное описание логических схем, то необходимо располагать соответствующей *методикой*. Представив логическую схему, или *комбинационную схему*, в виде черного ящика (рис. 4.30, а), имеющего n входов и один выход, ее поведение можно определять некоторой *переключательной функцией* $f(x_1, x_2, \dots, x_n)$ — функцией n логических переменных, например трех:

$$f^3 = \overline{x_1} \overline{x_2} \overline{x_3} \vee \overline{x_1} x_2 x_3 \vee x_1 \overline{x_2} x_3. \quad (4.5)$$

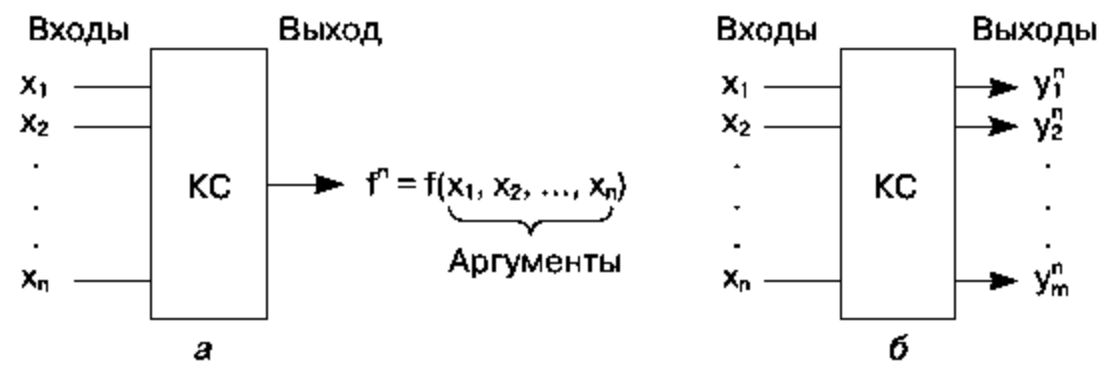


Рис. 4.30. Представление комбинационной схемы:

а — комбинационная схема с одним выходом;
б — комбинационная схема со многими выходами

Комбинационная схема может иметь и несколько выходов (рис. 4.30, б). Примером такой комбинационной схемы служит преобразователь кода "8421 → 2421" (рис. 4.31).

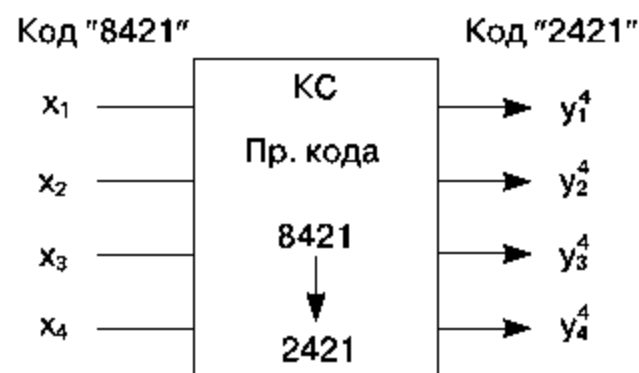


Рис. 4.31. Комбинационная схема преобразователя кода

Элементарными переключательными функциями являются функции одного и двух аргументов. Существует 16 функций двух аргументов. С помощью набора переключательных функций двух аргументов можно описать *любую* цифровую систему. На практике используют

не все функции, а лишь те из них, которые методом *суперпозиции* (замена одной функции другими) обеспечивают представление любой другой функции. Набор таких функций называют *функционально-полным* или *базисом*. Основными наборами являются: *булевый базис* $B_1 = \{\vee, \wedge, \neg\}$, включающий операции И, ИЛИ, НЕ и два универсальных базиса — $B_2 = \{\downarrow\}$, $B_3 = \{/ \}$, каждый из которых включает только одну функцию — *Пирса* (ИЛИ-НЕ) или *Шеффера* (И-НЕ) соответственно. С помощью этих функций можно описать поведение любой цифровой системы.

Для описания поведения комбинационной схемы (см. рис. 4.30, а) необходимо установить соответствие между входными и выходными сигналами, т.е. выразить функцию f^n как функцию входных переменных. Зависимость *вход* $\Rightarrow$ *выход* в комбинационной схеме в общем случае может быть описана различными способами — *временными диаграммами*, *таблично*, *номером* переключательной функции и др., в том числе и *аналитически*, например в Сов. ДНФ (выражение (4.5)). Задание условий работы любой комбинационной схемы в виде формул упрощает процесс построения самой комбинационной схемы, так как существует ряд эквивалентных преобразований, в результате которых логические формулы упрощаются. Переключательные функции оказались особенно полезными для описания работы логических элементов ЭВМ.

Комбинационная схема может быть построена на элементах различной степени сложности — на *дискретных* логических элементах (ЛЭ) или на *интегральных* схемах (ИС) с различным уровнем интегрирования — малым (МИС), средним (СИС), большим (БИС) и сверхбольшим (СБИС). В традиционном понимании ЛЭ — это схема, реализующая одну из элементарных переключательных функций, например И, ИЛИ, НЕ, И-НЕ (рис. 4.32) либо другие функции.

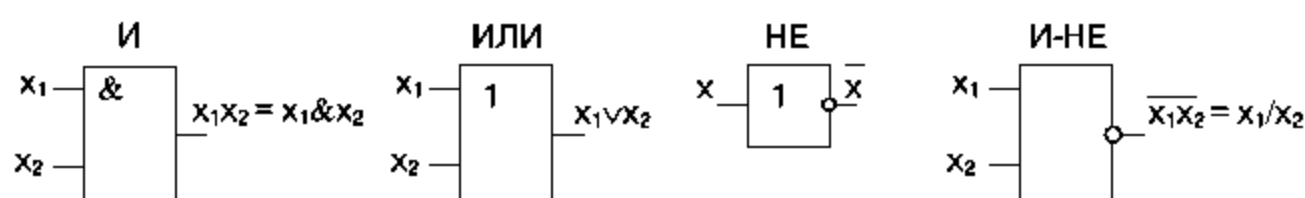


Рис. 4.32. Логические элементы

Комбинационная схема (рис. 4.33) представляет собой совокупность логических элементов И, ИЛИ, НЕ, организованных определенным образом для реализации зависимости (4.5). Сложность таких схем оценивается *по Квайну* (подсчитывается общее число входов — C^b схем И и ИЛИ). Для комбинационной схемы, показанной на рис. 4.33, цена $C^b = 12$.

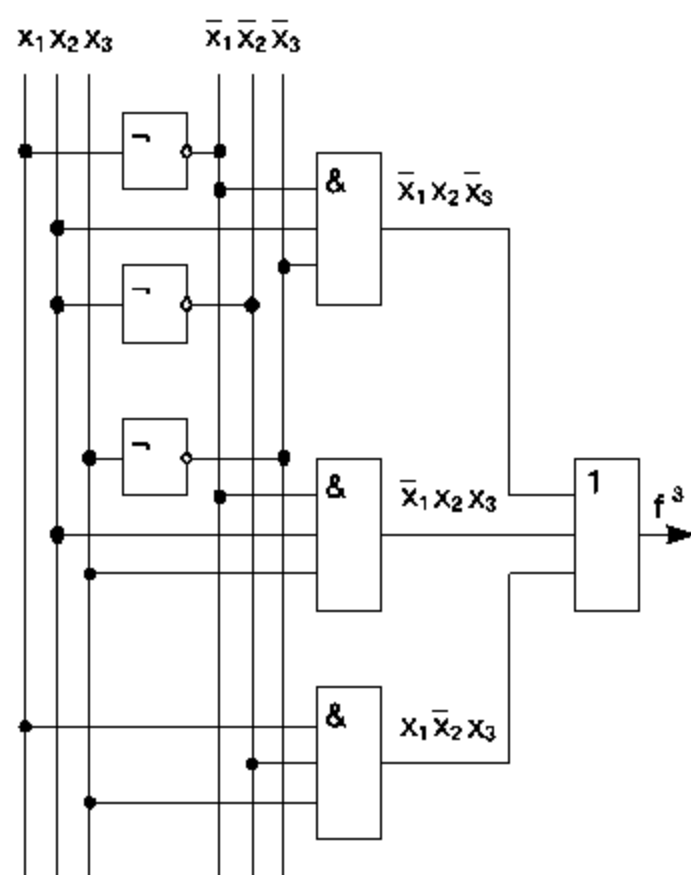


Рис. 4.33. Комбинационная схема в примере

В процессе проектирования комбинационной схемы на основе стандартных МИС и СИС используются как эвристические подходы, так и формализованные методы. Традиционный и хорошо изученный *метод* реализации комбинационной схемы с применением МИС включает следующие этапы:

- ♦ описание алгоритма функционирования комбинационной схемы;
- ♦ минимизация переключательных функций (отдельных и системы в целом), описывающих поведение комбинационной схемы;
- ♦ переход к базису И-НЕ либо ИЛИ-НЕ;
- ♦ реализация системы переключательных функций на стандартных ИС заданной серии

и представляет собой *многошаговый итерационный процесс* с возвратом назад и пересмотром ранее принятых решений.

Критерием аппаратной сложности комбинационных схем, реализованных на ИС, является число корпусов ИС либо число их выводов для устройств, реализуемых на печатной плате. Такой метод построения комбинационной схемы может быть использован для реализации сравнительно простых устройств на ИС средней степени интеграции наиболее перспективных в настоящее время серий 1533, 1531 (ТТЛШ) и 1554 (КМОП).

Сложные цифровые устройства проектируются сегодня на базе *программируемых логических интегральных схем* (ПЛИС), представляющих собой новую элементную базу, обладающую гибкостью заказных БИС и доступностью традиционной «жесткой» логики. Главным отличительным свойством ПЛИС является возможность их настройки самим пользователем путем *программирования*, под которым понимается изменение внутренней структуры ИС таким образом, чтобы она обеспечивала реализацию заданных переключательных функций на аппаратном уровне. Простейшими представителями ПЛИС являются программируемые постоянные запоминающие устройства (ППЗУ), программируемые логические матрицы (ПЛМ), программируемые матрицы логики (ПМЛ) и другие стандартные БИС, ориентированные на реализацию комбинационной логики.

Метод построения цифровых схем на *новой* элементной базе с использованием *новых технологий* проектирования изложен в работе [7].

4.5.4. Пример практического применения алгебры логики

Пусть требуется разработать схему устройства, позволяющего жильцам трехэтажного дома независимо включать и отключать лампочку освещения подъезда с любого этажа. Процесс построения схемы устройства можно упростить, если алгоритм его функционирования задать в одной из стандартных форм (таблично, алгоритмически и др.). При описании зависимости *вход* $\Rightarrow$ *выход* проектируемой схемы важно определить, что считать аргументами, а что — функцией, т.е. что подавать на вход комбинационной схемы, а что считать ее выходом.

Примем за аргументы переменные x_1, x_2, x_3 , которыми обозначим выключатели на соответствующих этажах. Тогда выходом устройства будет переключательная функция 3 аргументов $f(x_1, x_2, x_3)$. Переменная x_i , как и функция $f \in \{0, 1\}$. Будем считать, что $x_i = 1$, если выполнено одно из действий над выключателем, причем неважно какое — включить или выключить его. При этом неважно также, что конкретно означает $f = 1$ — горящую или выключенную лампочку. Важным является другое. Если на одном из этажей произведено *действие* над выключателем ($x_i = 1$), то лампочка должна *адекватно* на него отреагировать. Если на одном из этажей $x_i = 1$ означает включить лампочку ($f = 1$), то $x_i = 1$ на любом другом этаже может означать только ее отключение ($f = 0$), т.е. действию равно противодействие. Если третий аргумент окажется равным единице ($x_i = 1$), то функция вновь примет единичное

значение. Таким образом, равенство 1 двух аргументов обратит функцию f в 0, а трех аргументов — в 1. Описанный алгоритм можно задать таблично (табл. 4.9) или описать аналитически, например в Сов. ДНФ:

$$f^3 = \bar{x}_1 \bar{x}_2 x_3 \vee \bar{x}_1 x_2 \bar{x}_3 \vee x_1 \bar{x}_2 \bar{x}_3 \vee x_1 x_2 x_3. \quad (4.6)$$

Таблица 4.9

Табличное представление алгоритма

x_1	x_2	x_3	f^3
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	1

Построение комбинационной схемы, реализующей эту зависимость (4.6), аналогично построению комбинационной схемы (см. рис. 4.33), описываемой выражением (4.5).

4.5.5. Цифровые автоматы

Преобразование информации в компьютере реализуется электронными устройствами двух классов — *комбинационными схемами* и *цифровыми автоматами*.

Понятие *цифрового автомата* (ЦА) как средства для представления и обработки любых видов информации является одним из основных понятий информатики. Сегодня создаются и используются различные системы и устройства для преобразования дискретной информации, широко применяемые в качестве различного рода технических автоматов, вычислительных устройств и их функциональных блоков, устройств управления роботами, управляющих объектами по заданному алгоритму и т.д. Большой класс таких преобразователей объединяется под общим названием *автомат*. Некоторые авторы определяют автомат как математическую модель реальных *преобразователей дискретной информации*. Его функционирование состоит в том, что последовательность $z_1, z_2, \dots$ символов конечного или в общем случае бесконечного алфавита Z , поступающая на вход, вызывает на его выходе определенную последовательность $\omega_1, \omega_2, \dots$ символов того же или другого алфавита.

Таким образом, самая общая математическая модель преобразователя дискретной информации (ПДИ) описывается функцией $\varphi: Z^* \rightarrow W^*$, которая отображает множество Z^* всех последовательностей символов алфавита Z в другое множество W^* последовательностей символов алфавита W . Такая интерпретация позволяет схематично представить преобразователь как устройство, реализующее отображение одного множества на другое (рис. 4.34).

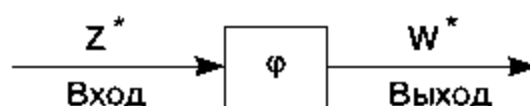


Рис. 4.34. Формальная модель ПДИ

Результат преобразования $\text{вход} \Rightarrow \text{выход}$ (см. рис. 4.34) зачастую зависит не только от того, какая информация в данный момент появилась на входе, но и оттого, что происходило раньше, от *предыстории* преобразования. Чтобы как-то запоминать предыстории и отличать одну от другой, преобразователь должен иметь память. Для этого в модель (см. рис. 4.34) вводят алфавит состояний $Q = \{q_1, q_2, \dots, q_m\}$ и такой преобразователь (рис. 4.35) называют *конечным автоматом (КА)*, если множество входных сигналов Z и множество возможных состояний Q конечны.

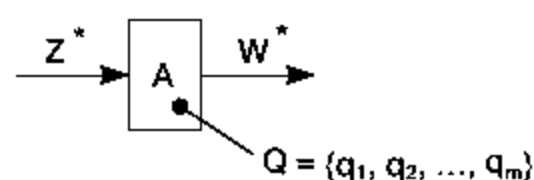


Рис. 4.35. Конечный автомат

Таким образом, *конечные автоматы* включают набор состояний и переходов между ними, зависящих от входных данных. Модель КА (см. рис. 4.35), имеющую один вход и один выход, называют еще *абстрактным автоматом (АА)*, поскольку в ней абстрагируются от реальных физических входных и выходных сигналов, рассматривая их просто как буквы некоторого алфавита и в связи с идеализированным дискретным временем, в котором работает АА.

Преобразователи дискретной информации, математической моделью которых является АА, называют *цифровыми автоматами (ЦА)*. Одним из наиболее распространенных в настоящее время примеров ЦА является ЭВМ. Автоматы с числом внутренних состояний более одного ($|Q| > 1$) составляют класс *автоматов с памятью*. Частным случаем

таких автоматов являются комбинационные схемы, выход которых не зависит от состояния и определяется текущим входным символом z_t . Математической моделью комбинационной схемы является функция $\omega(t) = \lambda(z(t))$.

Конечный автомат представляет собой хотя и абстрактную, но с функциональной точки зрения довольно точную модель дискретного процесса либо дискретного вычислительного или управляющего устройства. Эта модель является удобным средством описания многих систем, взаимодействующих с окружающим миром и реагирующих на поток внешних событий, так называемых реактивных систем, составляющих многие компоненты аппаратного и программного обеспечения. Такая модель обладает наглядностью и выразительностью, ясностью семантики (трактовки элементов модели) и в то же время достаточно строга и формальна. Она лежит в основе большого числа разнообразных приложений — от языковых процессоров до систем управления реальным временем и протоколов связи.

Любой ЦА с памятью, независимо от алгоритма его работы, на структурном уровне может быть представлен в виде обобщенной схемы (рис. 4.36), содержащей память и комбинационную схему, реализующую специфическую для конкретного автомата логику.

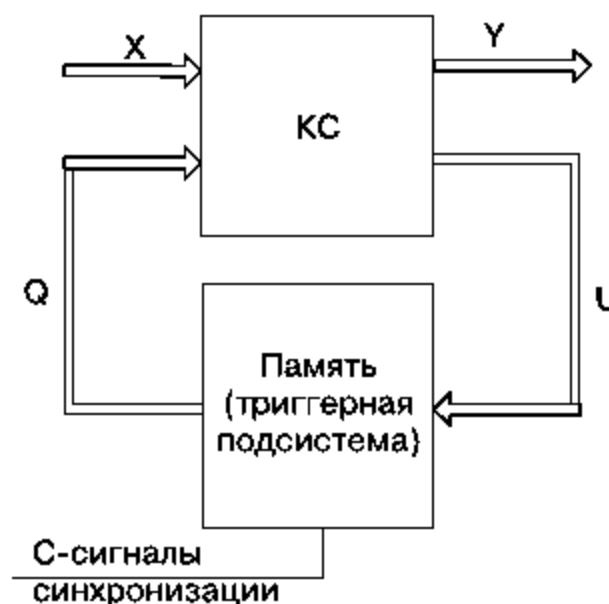


Рис. 4.36. Обобщенная схема ЦА

Комбинационная схема служит для преобразования входных сигналов X и информации о состоянии устройства Q в выходные сигналы Y и сигналы возбуждения элементов памяти U , определяющие смену состояний автомата.

Память строится из предварительно выбранных элементарных автоматов — *триггеров*.

Функционирование такого автомата в общем виде описывается системой функций:

$$\left. \begin{aligned} Y &= \lambda(X, Q) \\ Q(t+1) &= \delta(X, Q) \\ U &= \psi(X, Q) \end{aligned} \right\}, \quad (4.7)$$

где λ — функция выходов;

δ — функция переходов;

ψ — функция возбуждения триггеров;

X, Y, U, Q — двоичные слова:

$X = (x_1 \dots x_l \dots x_L)$ — входное слово;

$Y = (y_1 \dots y_n \dots y_N)$ — выходное слово;

$U = (u_1 \dots u_r \dots u_R)$ — слово функций возбуждения (ФВ);

$Q = (q_1 \dots q_t \dots q_R)$ — слово состояния автомата.

Такое представление автомата позволяет свести *задачу синтеза структурного автомата* при заданных элементарных автоматах к задаче синтеза комбинационной схемы. При этом комбинационная схема реализует систему переключательных функций, состоящую из K функций выходов:

$$\left. \begin{aligned} Y_1 &= \lambda_1(x_1, \dots, x_L, Q_1, \dots, Q_R) \\ Y_2 &= \lambda_2(x_1, \dots, x_L, Q_1, \dots, Q_R) \\ &\dots\dots\dots \\ Y_k &= \lambda_k(x_1, \dots, x_L, Q_1, \dots, Q_R) \end{aligned} \right\} \quad (4.8)$$

и R функций возбуждения:

$$\left. \begin{aligned} U_1 &= \psi_1(x_1, \dots, x_L, Q_1, \dots, Q_R) \\ U_2 &= \psi_2(x_1, \dots, x_L, Q_1, \dots, Q_R) \\ &\dots\dots\dots \\ U_R &= \psi_R(x_1, \dots, x_L, Q_1, \dots, Q_R) \end{aligned} \right\}. \quad (4.9)$$

Такие комбинационные схемы, в отличие от выпускаемых промышленностью стандартных функциональных блоков комбинационного типа (шифраторов, дешифраторов, мультиплексоров, компараторов, умножителей и др.) и выполняющих типовые для цифровых устройств микрооперации, называются комбинационными схемами с *произвольной логикой*.

4.6. СТРУКТУРНАЯ ОРГАНИЗАЦИЯ ЭВМ

4.6.1. Структурная схема фон неймановской ЭВМ

Наиболее удобной формой представления модели ЭВМ является ее *структурная схема*, состоящая из отдельных блоков компьютера и связей между ними. На рисунке 4.37 представлена структурная схема ЭВМ Джона фон Неймана.

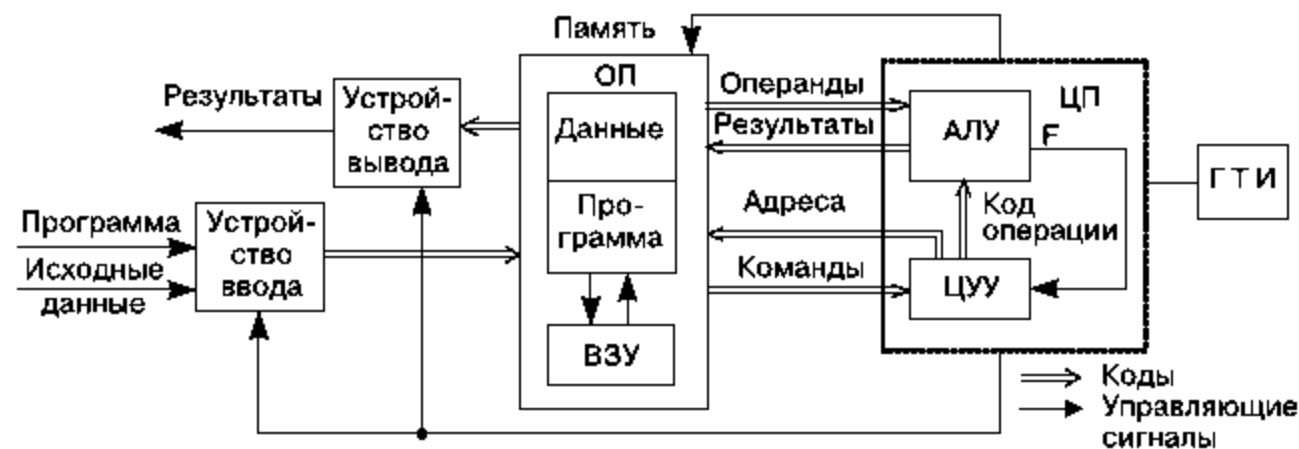


Рис. 4.37. Обобщенная структурная схема ЭВМ

Любая задача на компьютере решается автоматически (*принцип программного управления*) путем исполнения определенной *программы* (последовательность машинных команд), хранящейся в *памяти* компьютера.

Основными блоками ЭВМ являются:

- ◆ центральный процессор (ЦП);
- ◆ память (основная — ОП и внешняя — ВЗУ);
- ◆ устройства ввода-вывода (УВВ).

Центральный процессор

Центральный процессор (ЦП) — основной блок компьютера. Он осуществляет управление взаимодействием всех основных компонентов ЭВМ и непосредственно осуществляет обработку информации в соответствии с заданным алгоритмом, выполняя следующие операции:

- ◆ организация обращения к ОП: выборка команды, ее дешифрирование и производство вычислений (выполнение команд);
- ◆ инициирование работы внешних устройств;
- ◆ организация системы прерываний (прием и обработка запросов на прерывание).

Выбрав команду из памяти, ЦП выполняет указанную в ней операцию, например сложение, вычитание и т.д. Если операнд размещен

в памяти, то центральный процессор формирует его *адрес* и осуществляет выборку операнда из ОП. После выборки текущей команды процессор формирует адрес следующей команды.

Исполнение текущей программы может быть прервано при поступлении специальных сигналов — *запросов на прерывания*, которые вырабатываются либо в самой ЭВМ, либо в связанной с ней внешней среде. Функция на запросы прерываний состоит в том, что процессор прекращает выполнение текущей программы и переходит на выполнение другой — *программы обработки прерывания* (ПОП). Так как прерывание и его обработка могут изменить внутреннее состояние ЦП, то он сохраняет данные в ОП перед началом работы ПОП. Для этого в ОП пересылаются содержимое регистра команд, блока управления регистрами (УР) и управляющая информация.

После завершения работы ПОП состояние ЦП восстанавливается, позволяя продолжить выполнение прерванной команды.

Память ЭВМ

Под *памятью* ЭВМ понимают устройства, служащие для запоминания и представления информации, а именно входных и промежуточных значений и окончательных результатов, а также программ.

Память разделяют на *основную* (ОП) и *внешнюю* (ВЗУ). ОП предназначена для хранения программ и данных, имеет ограниченную емкость и содержит ЗУ двух типов: *оперативное запоминающее устройство* (ОЗУ) или запоминающее устройство с произвольной выборкой (ЗУПВ) и постоянное запоминающее устройство (ПЗУ).

В *ОЗУ* хранятся программы и данные, участвующие в вычислительном процессе на текущем этапе функционирования компьютера, которые могут быть изменены в произвольный момент времени. Исполняемый в текущий момент фрагмент программы *обязательно* находится в ОП. ОЗУ работает в трех режимах: *запись, чтение, хранение* и является *энергозависимым* типом памяти, т.е. информация теряется после отключения питания.

ПЗУ служит для хранения неизменной информации (загрузочные программы ОС, программы тестирования устройств компьютера, подпрограммы, микропрограммы, константы и т.п.). Изменить содержимое ПЗУ пользователь не может. ПЗУ работает в двух режимах: *чтение* и *хранение* и является *энергонезависимым* типом памяти, т.е. хранит информацию после отключения питания.

Внешняя память (ВП) или внешнее запоминающее устройство (ВЗУ) отличается практически неограниченной емкостью, так как чис-

ло блоков ВП можно наращивать по мере необходимости. ВЗУ реализуется на основе устройств с подвижным носителем информации (магнитные и оптические диски, магнитные ленты (МЛ) и др.).

Носитель — материальный объект, способный хранить информацию. По типу носителя все ВЗУ можно разделить на накопители на магнитной ленте (МЛ) и дисковые. Дисковый носитель может иметь жесткую (НЖМД) или гибкую основу (НГМД) — дискеты (флоппи-диски), компакт-диски (compact-disk) или просто CD — диски с оптическим способом записи информации, имеющие достаточно большую емкость (650—800 МБ) и DVD-диски (Digital Versatile Disk, цифровой диск общего назначения) емкостью до 17 ГБ данных. Все современные компьютеры имеют привод CD-ROM.

Накопитель можно рассматривать как совокупность носителя и соответствующего привода, представляющего собой сочетание механизма чтения-записи с соответствующими электронными схемами управления.

Устройства ввода-вывода

Устройства ввода-вывода обеспечивают взаимодействие машины с пользователем, объектами управления и другими компьютерами и содержат:

- 1) диалоговое средство пользователя:
 - ◆ видеомонитор (дисплей) — устройство для отображения вводимой-выводимой информации;
 - ◆ устройство речевого ввода (звуковые мыши со сложным ПО, позволяющие распознавать воспроизводимые человеком буквы и слова);
- 2) *устройства ввода*:
 - ◆ клавиатура — устройство для ручного ввода числовой, текстовой и управляющей информации;
 - ◆ сканеры — устройства ввода в компьютер информации непосредственно с бумажного носителя (тексты, схемы, рисунки, графики, фото); сканер подобно копировальному аппарату создает копию изображения бумажного документа, но не на бумаге, а в электронном виде, т.е. создается электронная копия изображения;
 - ◆ указатели (графические манипуляторы (мышь, планшет, световое перо и т.п.));
- 3) *устройства вывода*:
 - ◆ дисплеи;
 - ◆ принтеры — печатающие устройства для вывода информации из компьютера (матричные, струйные, лазерные);

4) *устройства связи и телекоммуникации:*

- ◆ адаптеры (контроллеры) — специальные электронные схемы, обеспечивающие подключение периферийных устройств (дисплей, жесткий диск и др.) к системной шине (магистралах) компьютера. Устройство для подключения компьютера к локально-вычислительной сети (ЛВС) называется сетевым адаптером;
- ◆ аналого-цифровые (АЦП) и цифроаналоговые (ЦАП) преобразователи;
- ◆ сетевые интерфейсные платы и карты;
- ◆ мультиплексоры передачи данных;
- ◆ модемы.

Перечисленные устройства используются для подключения компьютера к каналам связи, другим компьютерам и вычислительным системам (ВС), другим средствам автоматизации;

5) *средства мультимедиа:*

мультимедиа — это комплекс аппаратных и программных средств, позволяющих пользователю общаться с компьютером, используя различные естественные среды — звук, видеоизображение и т.д.

4.6.2. Функционально-регистрационная организация ЭВМ

Назначение и взаимодействие основных блоков модели и составляющих их элементов

Следующим, более низким после структурной модели (см. рис. 4.37) уровнем в иерархии представления фоннеймановской ЭВМ является ее функционально-регистрационная модель (рис. 4.38), раскрывающая принцип взаимодействия основных блоков (центрального процессора и памяти) на уровне регистров.

Центральный процессор (ЦП)

В ЦП можно выделить две основные части: *управляющую* (центральное устройство управления — ЦУУ) и *операционную* (ОЧ). Операционная часть содержит *средства*, в частности АЛУ и различные регистры, необходимые для преобразования информации. ЦУУ организует автоматическое выполнение программ и функционирование ЭВМ как единой системы.

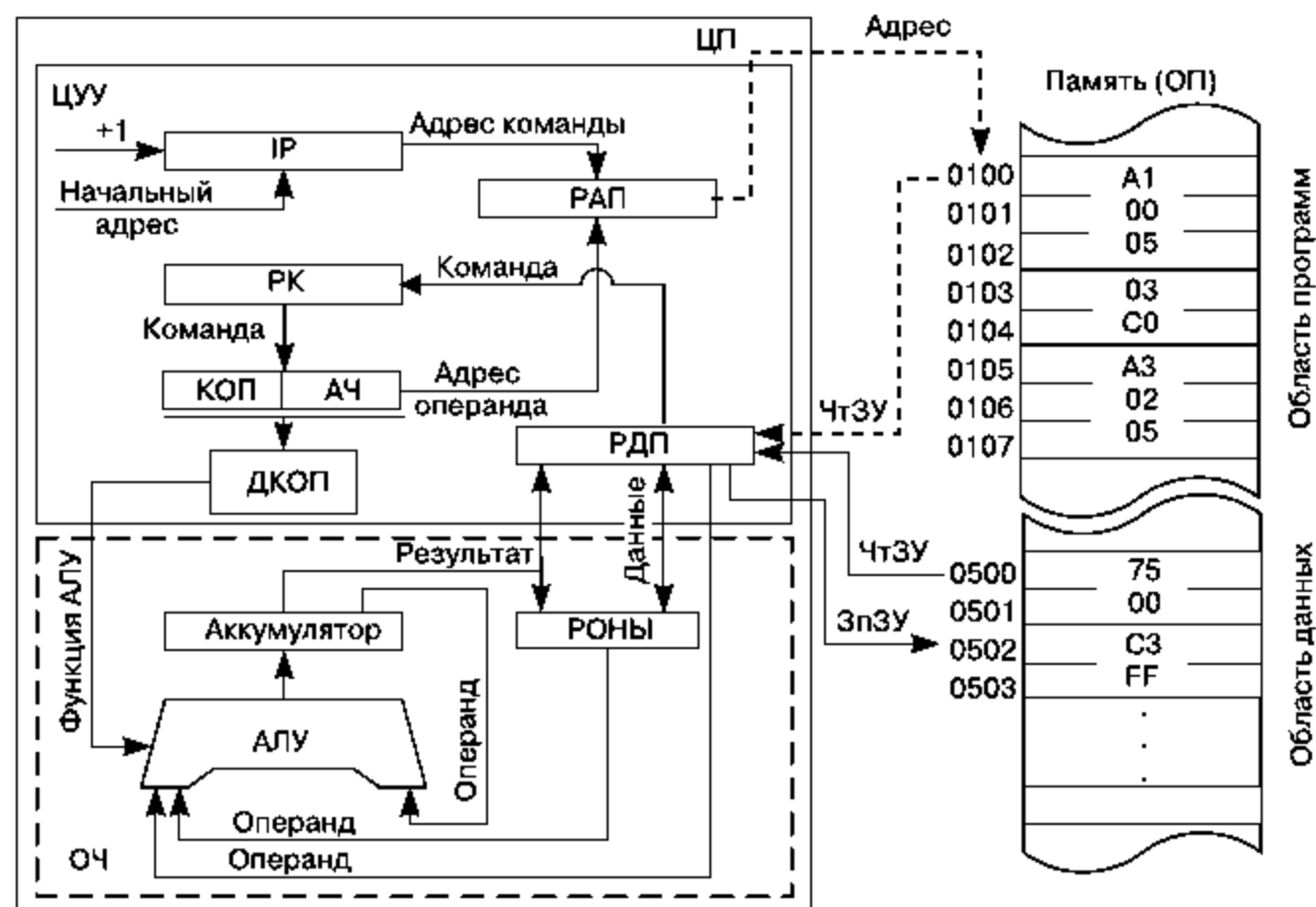


Рис. 4.38. Функционально-регистрационная ЭВМ
Дж. фон Неймана

ЦП работает *циклически*, т.е. выполняя каждую команду программы, он повторяет одни и те же действия:

- ♦ выборка команды из ОП;
- ♦ выборка данных из ОП (формирует адрес операндов и осуществляет их чтение из ОП);
- ♦ выполнение операции, указанной в выбранной команде, например сложение, вычитание, пересылка и др.;
- ♦ запись результата по указанному в команде адресу;
- ♦ формирование адреса следующей команды.

Выполнение команды процессора разделяется на более мелкие операции микропроцессора — *микрооперации* (МкОп).

МкОп — это определенные элементарные действия над машинными данными. Совокупность МкОп называется *микрокомандой* (МкК). Последовательность МкК, обеспечивающая выполнение данной операции, например умножение двух чисел с фиксированной точкой (ФТ), называется *микропрограммой* (МкП).

Таким образом, каждая команда ЭВМ реализуется соответствующей МкП, хранящейся в ПЗУ.

Процесс функционирования ЦП во времени представляет собой последовательность тактов (интервалов). Тактовые импульсы задает одна из микросхем, входящая в микропроцессорный комплект (чипсет) и расположенная на материнской плате. Частота F_T — один из показателей, определяющих быстродействие ЦП и в значительной степени быстродействие ЭВМ в целом. Первые процессоры x86 могли работать с частотой не выше 1,77 МГц, а сегодня рабочие частоты процессоров уже превышают миллиард тактов в секунду (1 ГГц).

Для выполнения одной команды ЭВМ требуется, как правило, несколько тактов. В течение каждого из них выполняется одна МкК микропрограммы, реализующей данную команду ЭВМ. Чем выше частота тактов, поступающих на процессор, тем больше команд он может исполнить в единицу времени, тем выше его производительность.

Так, для многих ПК частота $F_T = \sim 433$ МГц при работе с ОП, а без памяти — более 2 ГГц.

Последовательность сигналов, инициирующих выполнение соответствующей последовательности МкК, находящихся в ПЗУ, вырабатывает *центральное устройство управления* (ЦУУ). Кроме того, ЦУУ координирует работу всех устройств ЭВМ путем посылки управляющих сигналов, а именно: обмен информацией между ЦП и ОП, хранение и обработка информации, интерфейс и диагностика и др.

В ходе *вычислительного процесса* (последовательность действий, составляющих задачу обработки информации) ЦП работает в режиме непосредственного обмена с основной памятью (рис. 4.39).

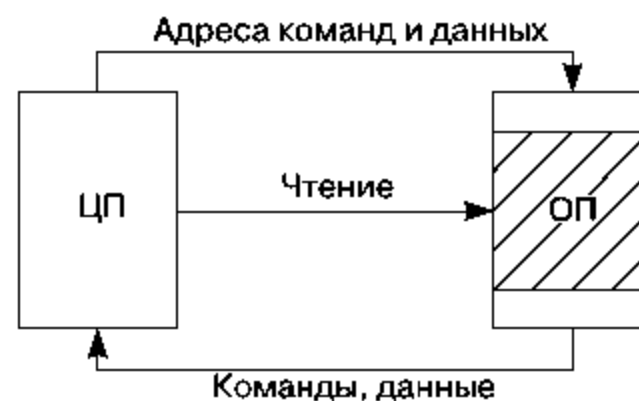


Рис. 4.39. Взаимодействие ЦП с ОП

По быстродействию ОП и ЦП должны быть согласованы, но поскольку обмен данными внутри ЦП происходит в несколько раз быстрее, чем обмен с другими устройствами, например с ОП; для уменьше-

ния числа обращений к ней внутри ЦП создают буферную область — так называемую *кэш-память* (рис. 4.40).

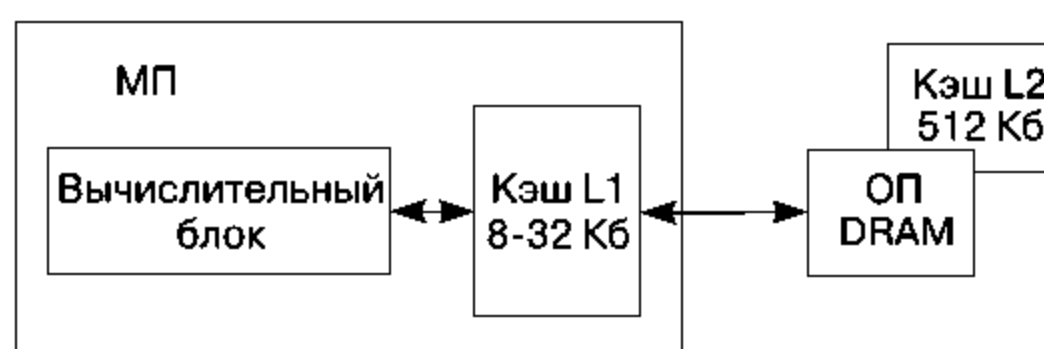


Рис. 4.40. Обмен данными с ОП через буфер

В кэш-памяти хранятся данные, которые микропроцессор (МП) будет использовать в ближайшие такты своей работы. Кэш-память работает на частоте, согласованной с частотой ядра процессора. Быстрый доступ к кэш-данным позволяет сократить время выполнения очередных команд и повысить производительность ЭВМ.

Основным элементом *операционной части* ЦП является *арифметико-логическое устройство* (АЛУ), условное обозначение которого на схемах приведено на рис. 4.41. АЛУ служит для выполнения арифметических и логических операций (сложение, вычитание, конъюнкция и др.), предусмотренных системой команд ЭВМ. Тип операции, выполняемой АЛУ, задается с помощью сигнала «Функция АЛУ», выдаваемого ДКОП (задается той частью командного слова, которая называется КОП).

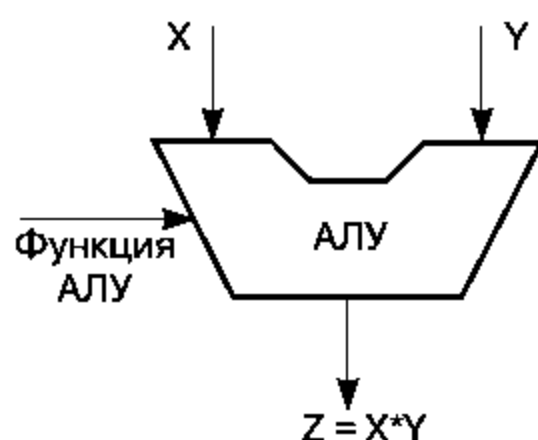


Рис. 4.41. Обозначение АЛУ на схемах

Z — результат операции, реализуемой АЛУ; X , Y , Z — двоичные либо двоично-десятичные слова; $*$ — любая из арифметических либо логических операций, входящих в СК ЭВМ

Структура АЛУ приведена на рис. 4.42.

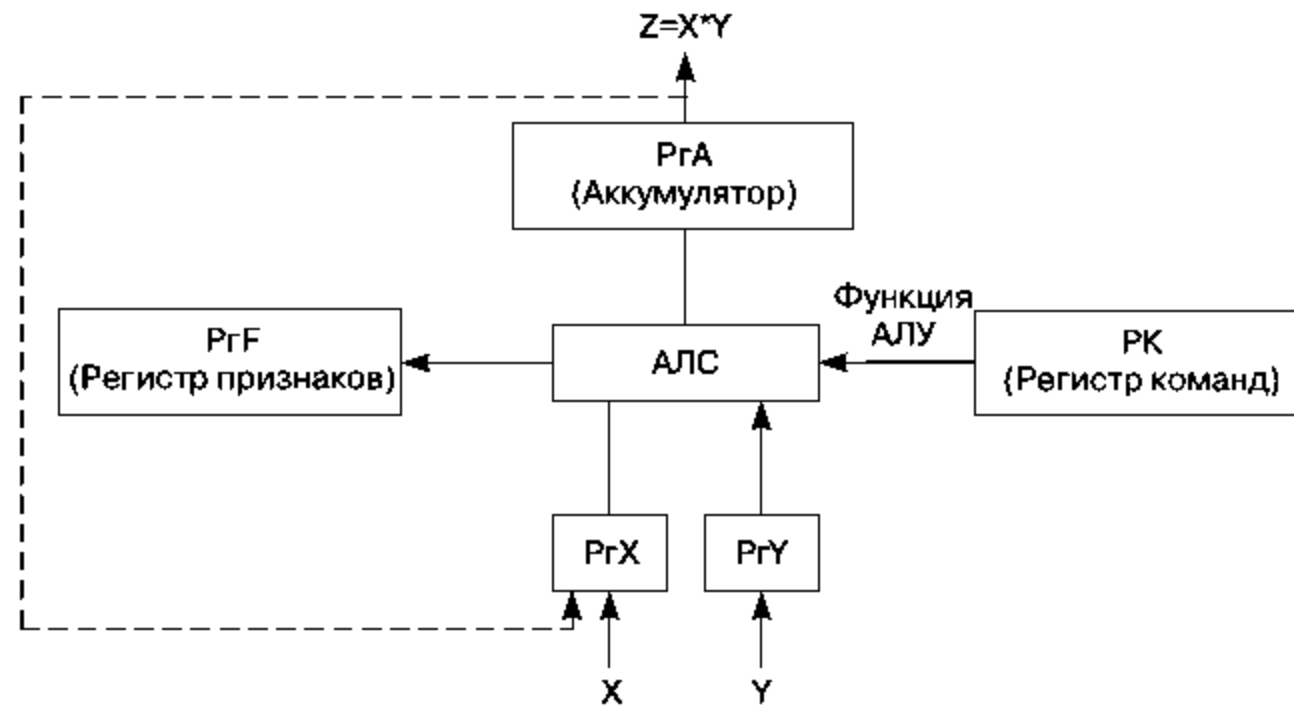


Рис. 4.42. Структура АЛУ

В состав АЛУ входят:

- ♦ *арифметико-логическая схема (АЛС)* — сложная комбинационная схема, настраиваемая на выполнение арифметических, логических и других операций в соответствии с кодом операции команды, хранящимся в регистре команд (РК). Это могут быть операции над двоичными числами с ФТ и ПТ, над десятичными числами, представленными в двоичной форме, либо операции по обработке символьной информации над словами фиксированной и переменной длины и др. Логические операции выполняются над отдельными битами, группами битов, байтами и их последовательностями. АЛУ характеризуется набором операций, которые оно выполняет, и средним быстродействием, т.е. количеством операций, выполняемых в единицу времени;
- ♦ *регистр-аккумулятор (РгА)*, в котором фиксируется результат реализуемой операции (также может хранить один из операндов двоичных операций);
- ♦ *регистры РгХ и РгУ* для хранения исходных операндов;
- ♦ *РгF-регистр признаков (флажков) результата:*
 - признак нуля (Z — zero);
 - признак знака (S — sign);
 - признак переноса (C — carry);
 - признак переполнения (O — overflow);
 - признак четности (P — parity) и др.

Каждый из признаков в регистре PnF занимает *один* бит, хранится в виде *нуля* либо *единицы* и формируются следующим образом:

$$Z = \begin{cases} 1, & \text{если нули во всех разрядах PrA,} \\ 0 & \text{в противном случае. Например, если } \begin{array}{c} \text{PrA} \\ \boxed{00001000} \\ \begin{array}{cc} 7 & 0 \end{array} \end{array}, \text{ то } Z = 0; \end{cases}$$

$$S = \begin{cases} 1, & \text{в случае отрицательного результата (содержимое PrA} < 0), \\ 0 & \text{в случае положительного результата (содержимое PrA} \geq 0). \end{cases}$$

Флажок S дублирует старший бит аккумулятора.

$$O = \begin{cases} 1, \text{ если результат операции превышает диапазон допустимых} \\ \text{чисел и не размещается в разрядной сетке } (Z < Z_{\max}), \\ 0 \text{ в противном случае;} \end{cases}$$

$$P = \begin{cases} 1, & \text{если число единиц в } P_{\Gamma A} \text{ четно,} \\ 0 & \text{в противном случае;} \end{cases}$$

$$C = \begin{cases} 1, & \text{если есть перенос из старшего разряда } P_{iA}, \\ 0 & \text{в противном случае.} \end{cases}$$

Принципы выполнения арифметико-логических операций в АЛУ являются общими для ЭВМ различных классов. Различия касаются схемотехнического решения организации АЛУ и принципов реализации операций, обеспечивающих ускорение их выполнения.

Конструктивно АЛУ может быть выполнено на большой интегральной схеме (БИС) или СБИС (одна или несколько). При этом ЦП может содержать одно АЛУ специального типа или несколько специализированных для каждой операции. В последнем случае повышается структурная сложность ЦП, но возрастает быстродействие и упрощается вычисление отдельных операций. Такой подход широко используется в современных ЭВМ общего назначения и суперЭВМ.

Роль *сверхбыстродействующей* памяти ЭВМ (более высокого быстродействия, чем ОП) выполняют *регистры общего назначения* (РОНЫ), входящие в операционную часть ЦП. Это память небольшого объема, повышающая его быстродействие и логические возможности. РОНЫ служат для временного хранения операндов, результатов операций и используются в качестве базовых и индексных регистров, указателей стека и др.

Неотъемлемым элементом *управляющей части* ЦП любой ЭВМ, построенной в соответствии с принципом *программного управления*, является *указатель команд* IP (Instruction Pointer) или счетчик команд. Это двоичный счетчик, в котором хранится и модифицируется адрес *очередной* команды программы. Получить его можно, увеличив адрес ячейки, из которой была считана текущая команда, на длину выполняемой 1, представленную числом занимаемых ею ячеек. Реализацию такого режима обеспечивает указатель команд IP, т.е.

$$\begin{array}{ccccc} \text{IP} & \leftarrow & \text{IP} & + & 1. \\ \text{Новое} & & \text{Старое} & & \text{Число байтов} \\ \text{значение} & & \text{значение} & & \text{выполняемой команды} \end{array}$$

Регистр команд (РК) служит для временного хранения кодов команд программы, считываемых из памяти и подлежащих выполнению. Команда становится «видимой» для процессора только с момента загрузки команды в РК. Процесс загрузки команды в РК называется *выборкой команды*.

Регистр адреса памяти (РАП) хранит адрес ячейки основной памяти, с которым производится обмен данными вплоть до завершения операции с этой ячейкой (считывание или запись).

Регистр данных памяти (РДП) служит буфером при работе с памятью: при чтении в него заносится содержимое ячейки ОП, при записи помещается информация, подлежащая сохранению в ячейке ОП. Момент считывания и записи в ячейку определяется соответственно сигналами ЧтЗУ и ЗпЗУ.

Дешифратор кода операции (ДКОП) на основе кода команды генерирует сигналы, определяющие функцию АЛУ, и инициирует процессы считывания необходимых операндов.

Основными параметрами процессора являются:

- ♦ *разрядность* (определяется разрядностью внутренних регистров) показывает, сколько бит данных процессор может принять и обработать в своих регистрах за один такт. Первые процессоры x86 были 16-разрядными. Современные процессоры семейства Intel Pentium являются 32-разрядными, хотя и работают с 64-разрядной шиной данных;
- ♦ *быстродействие*; исполнение каждой команды в процессоре занимает определенное число тактов, и чем выше частота тактов (от 4,77 МГц и больше), тем больше команд в единицу времени он может выполнить. Тактовые сигналы процессор

получает от *материнской платы*, которая в отличие от него, изготовленного на кристалле кремния, представляет собой *набор проводников и микросхем*. По чисто физическим причинам предел тактовой частоты лежит в диапазоне от 100 до 133 МГц. Для получения более высокой частоты происходит внутреннее умножение исходной частоты на коэффициенты $k = 3; 3,5; 4 \dots$;

- ♦ *система и структура команд*; различают команды: одноадресные, двухадресные, трехадресные (см. рис. 4.29);
- ♦ *способ представления данных* (с ФТ или с ПТ либо в VCD-форматах);
- ♦ *способы адресации данных*;
- ♦ *объем регистровой памяти* (число РОНов);
- ♦ *способ организации машинного цикла*.

Общее описание работы ЭВМ

Система команд ЭВМ. Каждый ПК имеет определенную *систему команд*, характеризующуюся списком команд и их структурой (форматом). Список команд отражает тот набор операций, выполнение которых предусмотрено на данной ЭВМ (в современных больше 100). В любой *системе команд* можно выделить следующие операции:

- ♦ *арифметические* над двоичными числами с ФТ и ПТ;
- ♦ *логические*;
- ♦ *операции пересылки*, например $AX \leftarrow [M]_{500}$;
- ♦ *операции обращения* к периферийным устройствам (специальные команды обращения к *портам* — регистрам данных периферийных устройств, которые могут быть выделены в отдельное пространство либо адресованы так же, как ячейки ОП, используя операции пересылки);
- ♦ *операции передачи управления* (для организации ветвлений, обращений к подпрограммам и выхода из них).

Порядок выполнения команд. Команды программы, расположенные в ОП компьютера, выполняются автоматически либо в *естественном*, либо в *принудительном* порядке.

В *первом* случае состояние указателя команд изменяется в соответствии с выражением $IP \leftarrow IP + 1$, где 1 — число байтов текущей команды, что обеспечивает переход к выполнению следующей по порядку команды программы. Для изменения естественного порядка выполне-

ния команд в системе команд ЭВМ предусмотрены специальные команды *условного* и *безусловного перехода*.

При выполнении команды *безусловного* перехода в IP заносится адрес следующей команды из адресной части команды безусловного перехода. Таким образом, обеспечивается выполнение той команды, на которую указывает адрес безусловного перехода (*принудительный порядок*).

Команда *условного* перехода (УП) в зависимости от условия может либо изменить, либо оставить неизменным значение IP. Проверка условия выполняется другой командой, выполняемой *перед* командой УП. Результат проверки заносится в регистр признаков результата (P_{RF}).

Таким образом, команда УП реализуется исполнением естественной или принудительной последовательности команд в зависимости от условия.

Выборка команд из ОП прекращается после выборки специальной команды — *останов*, предписывающей прекращение исполнения данной программы.

Исполнение программы типичной ЭВМ общего назначения. Процесс решения задачи определяется алгоритмом и заключается в исполнении ЦП последовательности команд в соответствии с программой, хранящейся в ОП наравне с данными.

Программа и данные загружаются в ОП через устройство ввода и хранятся в ней в виде последовательности двоичных слов.

Выполнение программы начинается с загрузки в IP адреса первой исполняемой команды (см. рис. 4.38). Содержимое IP пересылается в РАП, и в ОП посылается сигнал считывания, по которому первая команда (с начальным адресом 0100h) считывается из памяти и загружается в РДП, содержимое которого затем пересылается в РК. На этой стадии команда готова для ее декодирования центральным устройством управления (ЦУУ), в частности дешифратором кода операции (ДКОП), и выполнения.

Для команды, содержащей операцию, которая должна быть выполнена в АЛУ, необходимо получить требуемые операнды. Если операнд находится в ОП (а он может быть и в РОН), его необходимо выбрать из памяти. Для этого в РАП пересылается адрес операнда, содержащийся в адресной части команды, и начинается цикл *чтения* (Чт). Операнд, выбранный из памяти в РДП, может быть передан в АЛУ.

Выбрав таким образом один или несколько операндов, АЛУ может выполнить требуемую операцию, сохранив ее результат в одном из РОНов.

Если результат операции необходимо запомнить в ОП, он должен быть послан в РДП. Адрес ячейки, в которую необходимо поместить результат, пересылается в РАП, и начинается цикл *записи* (Зп).

Содержимое IP увеличивается, указывая следующую команду. Помимо передачи данных, между ЦП и ОП необходимо обеспечить обмен данными с внешними устройствами (ВУ), что делают машинные команды управления вводом-выводом.

Естественный порядок выполнения может быть нарушен при поступлении *сигнала прерывания* (см. подраздел 4.6.1).

4.7. ХРАНЕНИЕ ИНФОРМАЦИИ

4.7.1. Общая характеристика устройств хранения информации

Хранение является одной из основных операций, осуществляемых над информацией. Под хранением информации понимают ее запись в запоминающее устройство (ЗУ) для последующего использования.

Запоминающее устройство (память) — устройство, способное принимать данные и сохранять их для последующего считывания.

В компьютерных системах обработки информации выделяют следующие основные типы памяти: *регистровую, основную, кэш-память и внешнюю* (рис. 4.43). Кроме того, в ЭВМ могут присутствовать различные специализированные виды памяти, характерные для тех или иных устройств вычислительной системы, например *видеопамять*.

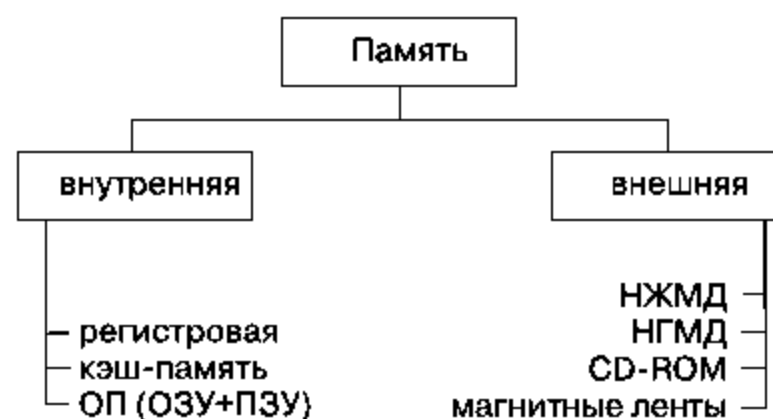


Рис. 4.43. Типы памяти

Регистровая память в составе процессора или других устройств ЭВМ предназначена для кратковременного хранения небольшого объема информации, непосредственно участвующей в вычислениях или операциях обмена (ввода-вывода). Работает в режиме непосредственного обмена с ЦП и по возможности согласована с ним по быстродействию.

Использование регистровой памяти позволяет сократить число обращений к другим типам памяти, требующим большего времени для операций обмена информацией.

Основная память (ОП) предназначена для оперативного хранения программ и данных и обмена данными, непосредственно участвующими в процессе обработки. Отличается малым временем доступа при достаточно больших объемах. Физически ОП реализуется на основе больших полупроводниковых интегральных схем и подразделяется на два вида:

- ♦ постоянное запоминающее устройство (ПЗУ);
- ♦ оперативное запоминающее устройство (ОЗУ) или *ЗУПВ* — *запоминающее устройство с произвольной выборкой* (см. подраздел 4.6.1. Память ЭВМ).

На рисунке 4.44 приведена логическая организация основной памяти ЭВМ в виде последовательности ячеек памяти размером в один байт (значения адресов, кодов команд и данных заданы в 16-ричной СС). Например, команды программы хранятся, начиная с адреса 100h, а данные — с адреса 500h. Байты *одной команды* размещаются в соседних ячейках памяти. Длина команд, например в МП семейства Intel 80X86, применяемых в персональных компьютерах IBM PC, варьируется от 1 до 6 байт. *Первый байт команды* располагается по *младшему* адресу. Младшая часть слова данных также хранится по младшему адресу.

Адрес	Память
0100	1A
0101	02
0102	04
0103	2B
0104	D0
0105	B2
0106	...
...	...
0500	60
0501	00
...	...

Рис. 4.44. Логическая организация памяти

Кэш-память служит для хранения копий информации. Это очень быстрое ЗУ небольшого объема, которое служит *буфером* между устройствами с различным быстродействием. Обычно используется при обмене данными между микропроцессором и основной памятью для компенсации разницы в скорости обработки информации процессором и несколько менее быстродействующей оперативной памятью (см. рис. 4.40). Кэш-памятью управляет специальное устройство — контроллер, кото-

рый, анализируя выполняемую программу, пытается предвидеть, какие данные и команды вероятнее всего понадобятся в ближайшее время процессору и подкачивает их в кэш-память. При этом возможны как «попадания», так и «промахи». В случае *попадания*, т.е. если в кэш подкачаны нужные данные, их извлечение из памяти происходит без задержки. Если же требуемая информация в кэш-памяти отсутствует, то процессор считывает ее непосредственно из оперативной памяти. Соотношение числа попаданий и промахов определяет *эффективность* кэширования.

Внешняя память (ВП) используется для *долговременного* хранения больших объемов информации. В качестве устройств ВП наиболее часто применяются:

- ◆ накопители на жестких магнитных дисках (НЖМД);
- ◆ накопители на гибких магнитных дисках (НГМД);
- ◆ накопители на оптических компакт-дисках (CD-ROM и DVD-ROM);
- ◆ магнитооптические накопители (МО), реагирующие как на оптическое, так и на магнитное воздействие;
- ◆ ленточные накопители (стримеры).

Время доступа к информации, расположенной на внешних ЗУ, составляет миллисекунды (10^{-3} с) в отличие от элементов оперативной памяти со временем доступа в пределах наносекунд (10^{-9} с).

4.7.2. Основные характеристики памяти

Конструктивно память состоит из огромного числа запоминающих элементов (ЗЭ), каждый из которых может находиться в одном из двух состояний, кодируемых двоичной цифрой: 0 или 1. ЗЭ группируются в запоминающие ячейки, предназначенные для хранения слов, т.е. таких порций информации, которые могут *пересылаться* между памятью и процессором и ими *обрабатываться*. Поиск нужного слова памяти выполняется, как правило, по его адресу.

Основные характеристики запоминающих устройств:

- ◆ *информационная емкость* V_n — максимально возможный объем хранимой информации. Измеряется в битах, байтах и других структурных единицах, состоящих из определенного числа байтов (КБ, МБ, ГБ, ТБ);
- ◆ *быстродействие* — *время доступа* и *время обращения* к памяти. *Временем доступа* t_d называют временной интервал между началом обращения ЦП за данными (ЦП выставил на адрес-

ной шине адрес требуемой ячейки памяти и послал по шине управления сигнал на чтение или запись данных) и моментом появления данных из адресуемой ячейки на шине данных.

Временем обращения $t_{обр}$ называют интервал между началом и окончанием ввода и вывода информации в память. Полный цикл обращения к памяти занимает больше времени, чем время доступа, так как включает время на поиск места в памяти, время на запись и время на считывание информации из памяти;

- ♦ *время записи* — интервал времени, необходимый для перезаписи содержимого шины данных в связанную с ней ячейку памяти. Минимальная величина этого интервала определяется физическими свойствами элементов памяти.

Цикл считывания и цикл записи определяются как время с момента выдачи процессором адреса требуемой ячейки памяти и сигнала на считывание или запись (они могут выдаваться позднее адреса) до того момента, когда заканчиваются все действия, связанные с выполняемой операцией, и память будет готова выполнить следующую операцию;

- ♦ *ширина доступа* — объем считанной (записанной) информации за одно обращение к памяти. В зависимости от типов памяти этот объем — 1 байт, слово, группа байтов. Именно время доступа и ширина доступа определяют производительность операций с памятью ЭВМ;
- ♦ *стоимость одного бита* определяется отношением стоимости памяти к ее информационной емкости;
- ♦ *потребляемая мощность*.

Заметим, что названные основные характеристики ЗУ максимально противоречивы, например большая информационная емкость не сочетается с высоким быстродействием, а быстродействие — с низкой стоимостью.

Для оптимального сочетания объема, быстродействия и стоимости память содержит несколько типов ЗУ, отличающихся организацией, назначением и способами хранения (см. рис. 4.43).

4.7.3. Классификация основной памяти

Для классификации ЗУ важнейшим признаком служит *способ доступа к данным*. По этому признаку все ЗУ делятся на ЗУ с прямым (*адресные*), последовательным (*последовательные*) и ассоциативным (*ассоциативные*) доступами (рис. 4.45).

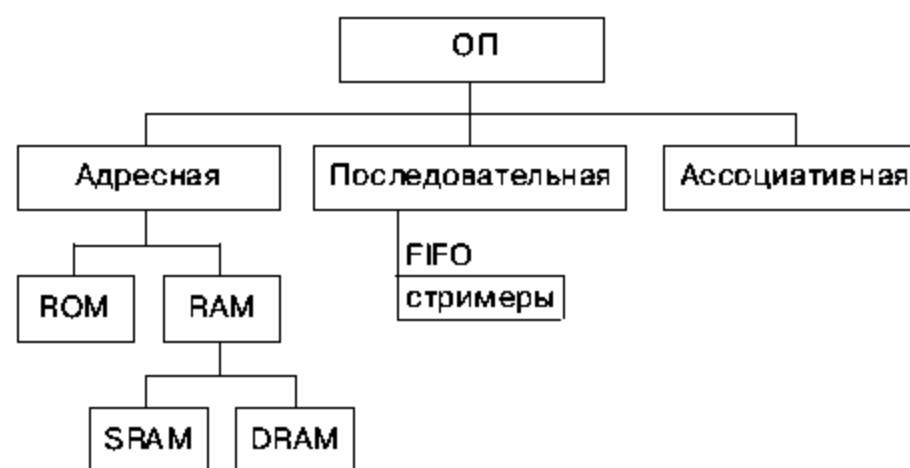


Рис. 4.45. Классификация ЗУ

В ЗУ с *прямым доступом* размещение и поиск информации основаны на *адресном принципе* хранения слова. *Адресом* слова является номер содержащей его ячейки. При такой организации памяти команда должна прямо или косвенно указывать адрес ячейки через адресные регистры. Все ячейки такой памяти в момент обращения равнодоступны.

К запоминающим устройствам с *прямым доступом* относятся полупроводниковые ОЗУ или RAM (Random Access Memory) и ПЗУ или ROM (Read Only Memory), а также дисковые ЗУ.

По способу хранения информации ОЗУ (RAM) делятся на статические (SRAM — Static RAM) и динамические (DRAM — Dynamic RAM). В *статических ОЗУ* запоминающими элементами являются триггеры, сохраняющие свое состояние, пока схема находится под напряжением и не выполняется новая запись данных. В *динамических ОЗУ* данные хранятся в виде зарядов конденсаторов, образуемых элементами МОП-структур. Саморазряд конденсаторов ведет к разрушению данных, поэтому они должны периодически (каждые несколько миллисекунд) регенерироваться. Вместе с тем следует заметить, что плотность упаковки динамических элементов памяти в несколько раз больше плотности упаковки SRAM. Однако *быстродействие* SRAM в 4—5 раз *больше*, чем DRAM. SRAM используется в кэш-памяти. В качестве ОЗУ в современных микропроцессорных системах чаще всего используются динамические ОЗУ с запоминающим конденсатором с асинхронным или *синхронным* доступом (SDRAM).

Постоянные запоминающие устройства (ПЗУ) можно разделить по *способу их программирования* на следующие категории:

- ♦ *масочные ПЗУ*. Программируются однократно при изготовлении и не допускают последующего изменения информации;

- ♦ *программируемые ПЗУ (ППЗУ)*. Позволяют однократно изменить состояние запоминающей матрицы электрическим путем по заданной программе;
- ♦ *репрограммируемые (перепрограммируемые) ПЗУ, или РПЗУ*. Допускают возможность многократного электрического перепрограммирования. Стирание хранящейся в РПЗУ старой информации перед записью новой осуществляется по-разному: *электрическое* в EPROM (electrically PROM) либо *ультрафиолетовое* стирание с помощью ультрафиолетового излучения (РПЗУ с УФ-стиранием). К памяти типа EPROM (ПЗУ с электрическим программированием и электрическим стиранием после программирования) относится также *Flash-память* — энергозависимая память, допускающая многократную перезапись своего содержимого. Она подобна перепрограммируемому ПЗУ по запоминающему элементу, но имеет структурные и технологические особенности, позволяющие выделить ее в отдельный вид.

В ЗУ с *последовательным доступом* записываемые данные образуют некоторую *очередь*. Считывание из очереди происходит слово за словом либо в порядке записи (FIFO), либо в обратном порядке (LIFO) («последним пришел, первым вышел»). Организация с последовательным доступом *присуща* видеосистемам и магнитным лентам.

В произвольный момент времени с магнитной ленты считывается лишь то слово, которое находится непосредственно под головкой чтения-записи. Для чтения другого слова его надо переместить под головку, предварительно просмотрев предыдущие участки.

Стековая адресация, реализующая безадресное задание операндов, широко используется в мини-, микро- и некоторых суперЭВМ.

Безадресные команды на основе стековой адресации предельно сокращают формат команд, экономя память и повышая производительность ЭВМ. Эта адресация используется при организации переходов к подпрограммам и возврату из них, а также в аппаратно-программных системах прерываний.

Ассоциативная память: обеспечивает поиск информации не по ее расположению в памяти (по *адресу* — прямой доступ) или месту в очереди — последовательный доступ, а по некоторому признаку, например на совпадение определенных полей слов — *тегов* (от англ. *tag*). На выход выдаются слова, удовлетворяющие признаку. Основная область применения — кэширование данных.

4.7.4. Логическая организация основной памяти

Основная память делится на две логические области:

- ♦ непосредственно адресуемая память (НАП);
- ♦ расширенная память.

Доступ к ячейкам *расширенной* (дополнительной) памяти возможен при использовании специальных программ (драйверов).

Драйвер — программа, управляющая работой памяти или внешних устройств ЭВМ и организующая обмен информацией между ЦП, ОП и внешними устройствами.

Драйвер, управляющий работой памяти, называется *диспетчером памяти*.

НАП представляет собой линейную последовательность n -разрядных ячеек, в каждой из которых хранится определенное количество информации, кратное байту. С точки зрения ЦП эта область памяти есть последовательность байтов, каждый из которых имеет свой уникальный адрес (см. рис. 4.44). При адресации *многобайтовых* единиц (слово, двойное слово) их адресом является адрес *младшего байта*, т.е. *младший байт* расположен по *младшему адресу*.

На рисунке 4.46 приведен пример размещения трех единиц данных X_1 , X_2 , X_3 различных форматов в памяти по адресу $A13Eh$.

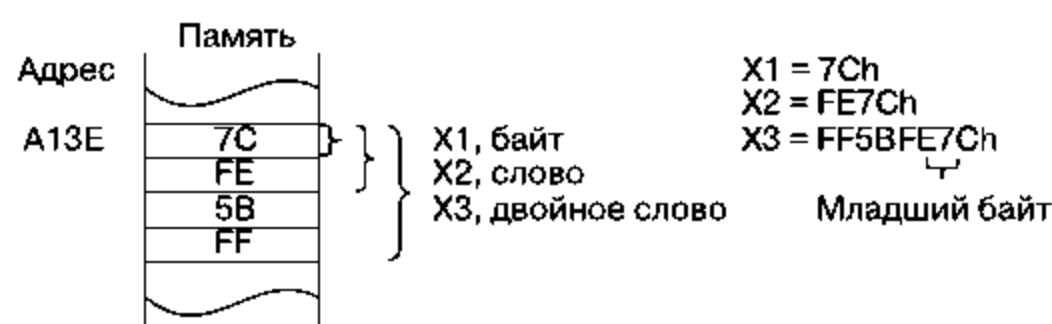


Рис. 4.46. Адресация многобайтовых данных

Для каждого ЦП или МП существует своя непосредственно адресуемая память, другими словами, свое максимально адресуемое пространство памяти, объем V которой определяется разрядностью m *шины адреса* МП и составляет $V = 2^m$ ячеек памяти. Вызываемый микропроцессором физический адрес есть адрес байта. Например, регистры МП $i8086$ и $i80286$ являются 16-разрядными, и максимальное пространство, которое каждый из них может «видеть», составляет 64 КБ. Для обращения к адресному пространству в 1 или 16 МБ требуется особый механизм — *механизм сегментации*, при котором адресное пространство разделяют на участки, называемые *сегментами* (рис. 4.47).

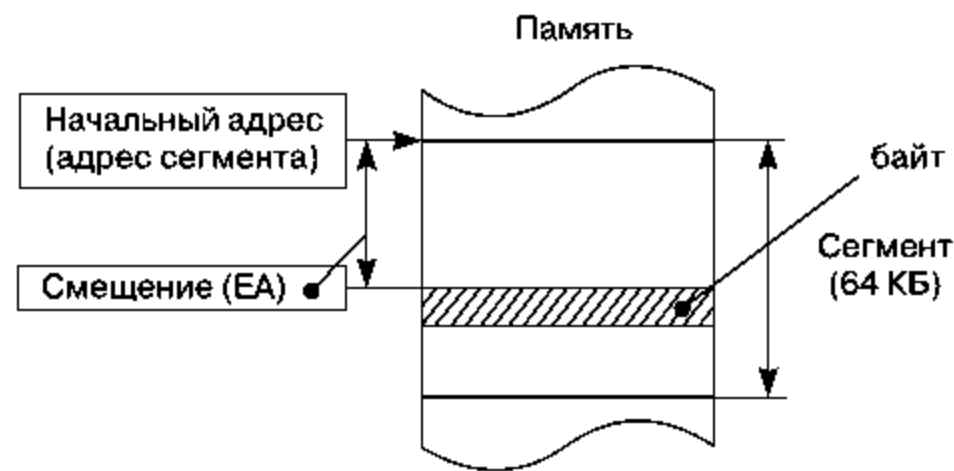


Рис. 4.47. Сегмент памяти

По существу внутри МП вместо 20-разрядного физического адреса формируется так называемый *логический (виртуальный) адрес*, состоящий из двух частей — *сегмента и смещения*, каждая из которых хранится в отдельном 16-разрядном регистре (рис. 4.48).

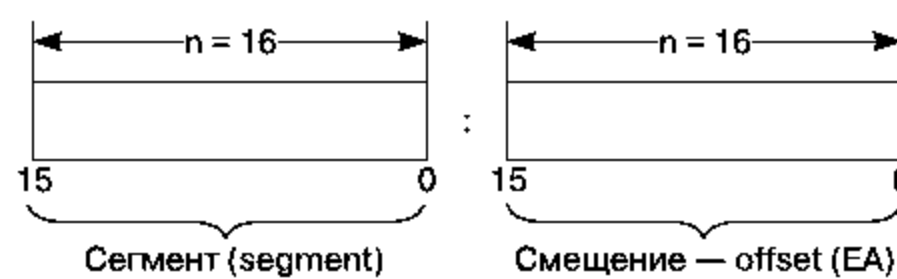


Рис. 4.48. Логический (виртуальный) адрес

В *регистре сегмента* хранится начальный (*базовый*) адрес сегмента, а в *регистре смещения* — *эффективный (исполнительный) адрес (EA)* (из регистра IP), используемый для обращения к байтам внутри сегмента. Он указывает на величину смещения байта относительно базы: для первого байта оно равно 0000h, для последнего — FFFFh.

Таким образом, для обращения к байту адресного пространства в 1 Мб надо задать адрес сегмента и адрес смещения внутри этого сегмента, например ds:43BCh.

Для формирования *физического* адреса байта (см. рис. 4.47) из двух частей логического адреса надо выполнить два действия:

- ◆ значение сегмента сдвинуть влево на четыре двоичных разряда;
- ◆ к сдвинутой последовательности прибавить величину смещения (с помощью 20-разрядного сумматора (СМ)) (рис. 4.49).



Рис. 4.49. Вычисление физического адреса

Такое вычисление адреса байта выполняется при каждом обращении к памяти. Одновременно можно работать с четырьмя сегментами. Их базовые адреса находятся в четырех регистрах сегментов:

- CS — регистр сегмента программ;
- DS — регистр сегмента данных;
- SS — регистр сегмента стека;
- ES — регистр сегмента дополнительных данных.

Если надо работать с большим числом сегментов, например с пятью, содержимое одного из регистров сегмента переносится в обычный регистр, а в освободившийся регистр сегмента записывается начальный адрес пятого сегмента, и далее он используется для доступа к ячейкам памяти пятого сегмента.

4.7.5. Память — основной ресурс ЭВМ

Память ЭВМ относится к основным ресурсам машины. Ее характеристики во многом определяют характеристики всей ЭВМ и ее возможности. Рассмотрим влияние быстродействия памяти на быстродействие ЭВМ в целом на простом примере. Допустим, что быстродействие машины составляет 200 тыс. оп./с, т.е. время выполнения одной операции, например сложения, составляет 5 мкс ($t_{\text{сл}} = 5 \text{ мкс}$). За это время необходимо выполнить саму операцию сложения и трижды обратиться к ОП — считать два операнда и записать результат в память.

Для выполнения операции сложения с заданной скоростью ($t_{\text{сл}} = 5 \text{ мкс}$) время обращения к ОП должно составлять порядка 1 мкс ($t_{\text{обр}} = 1 \text{ мкс} = 10^{-6} \text{ с}$), т.е. в 5 раз меньше, чем время выполнения операции сложения. Таким образом, быстродействие памяти должно превышать быстродействие ЭВМ в 5 раз. В современных ОЗУ время доступа (считывания данных и запись в ОЗУ) составляет

$$t_{\text{дост}} \approx 10 \text{ нс} = 10 \cdot 10^{-9} \text{ с}.$$

Правило определения адреса операнда называется *способом (или режимом) адресации*. Он определяет порядок выполнения операций над

адресной частью команды и содержимым одного или нескольких регистров процессора для вычисления *эффективного* ЕА или исполнительного адреса $A_{исп}$, по которому хранится операнд в основной памяти или в блоке РОН микропроцессора. Значение адреса $A_{исп}$ — всегда целое двоичное число без знака:

$$A_{исп} = A_{n-1} A_{n-2} \dots A_2 A_1 A_0, \text{ где } A \in \{0, 1\}.$$

Число бит адреса m определяет максимальное адресуемое процессором число ячеек памяти, равное 2^m , которое называют *адресным пространством памяти* (см. подраздел 4.7.4).

4.8. ОРГАНИЗАЦИЯ ВВОДА-ВЫВОДА ИНФОРМАЦИИ В ЭВМ

Под *вводом-выводом* понимают любой обмен словами и данными между *центральной частью* (ЦП и ОП) — ядром ЭВМ и УВВ — внешними (ВУ) или периферийными устройствами (ПУ). Этот обмен осуществляется с помощью *интерфейса ввода-вывода* — совокупности *аппаратных* (регистры, шинные формирователи и т.п.) и *программных средств* (протоколы обмена, драйверы ввода-вывода и др.).

Несмотря на широкое развитие системных интерфейсов ЭВМ, в общем случае можно выделить *два основных способа* использования системного интерфейса для организации обмена информацией с ВУ:

- ♦ с применением специальных команд ввода-вывода;
- ♦ по аналогии с обращениями к памяти.

Интерфейсная модель связи ядра ЭВМ с УВВ представлена на рис. 4.50.

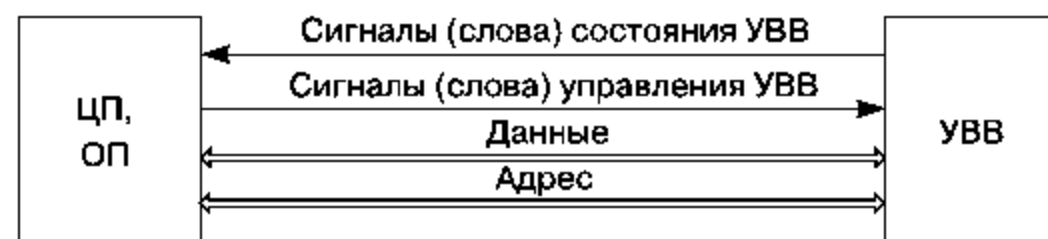


Рис. 4.50. Взаимодействие ядра ЭВМ с УВВ

Сигналы состояния УВВ передают в центральную часть ЭВМ информацию о состоянии УВВ, например УВВ готово к обмену данными.

Сигналы управления УВВ (сброс, подтверждение приема, запись, чтение и др.) служат для управления устройствами ввода-вывода.

При использовании для обмена с ВУ команд ввода-вывода *адрес (номер)* ВУ передается по шине адреса, однако по этой же шине пере-

даются и адреса ячеек памяти. Информация на шине адреса имеет смысл адреса (номера) ВУ только при наличии специальных управляющих сигналов, например таких, как «Ввод из ВУ», «Вывод из ВУ», инициируемых соответствующими командами ввода-вывода ЭВМ.

Устройства ввода-вывода составляют 80—85% состава компьютера и его стоимости. Взаимодействие центральной части ЭВМ с внешними устройствами осуществляется через *контроллеры (адаптеры)* ввода-вывода. В структуре ЭВМ с *единой системной шиной* для физической организации ввода-вывода каждое внешнее устройство подключается к системной шине (СШ) через свой контроллер (рис. 4.51). С точки зрения программиста, контроллер УВВ представляет собой набор n -разрядных регистров ввода-вывода, часто называемых *портами* ввода-вывода, через которые УВВ взаимодействует с ЦП и ОП. Каждый порт имеет свой номер или адрес; таким образом, *обращение к УВВ — это обращение к порту*.

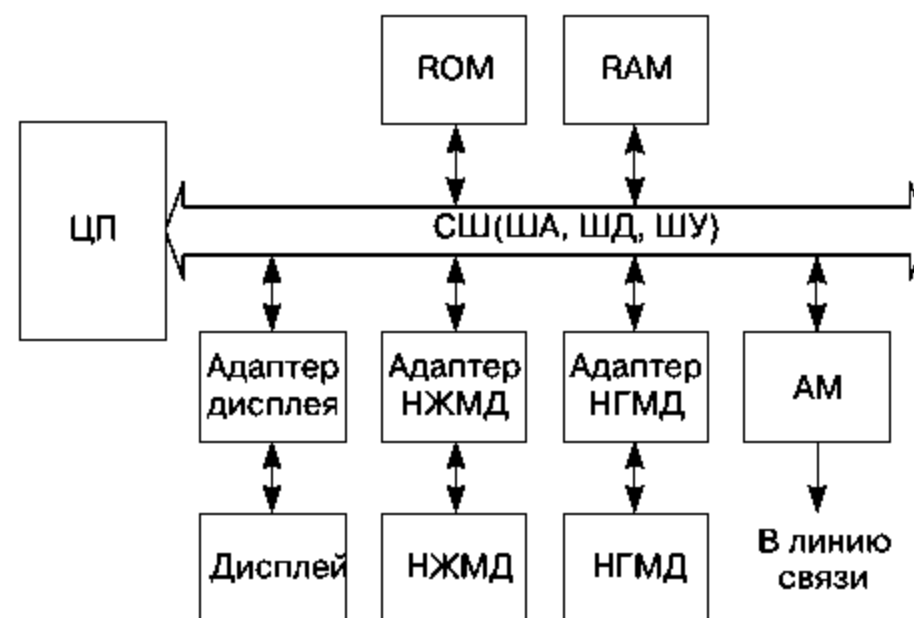


Рис. 4.51. Подключение ВУ к ЭВМ

Однако такое положение дел, когда для каждого типа ВУ требуется уникальный контроллер, конечно, не устраивало ни разработчиков средств вычислительной техники, ни ее пользователей. Наиболее перспективным оказался путь *стандартизации* набора информационных и управляющих сигналов, которыми обмениваются контроллер и ВУ. Так, для *разных типов* ЭВМ были разработаны контроллеры, обеспечивающие связь с ВУ по стандартному *параллельному* и по стандартному *последовательному* каналам передачи данных. Основу типичного контроллера ВУ, применяемого в ЭВМ с различными системными интерфейсами и обеспечивающего программно-управляемый обмен ин-

формацией с ВУ (операции *ввод* и *вывод*), составляют несколько регистров (*порты ввода-вывода*), которые служат для временного хранения передаваемой информации. Каждый регистр имеет свой адрес. *Регистры входных и выходных данных* (рис. 4.52) работают соответственно только в режиме чтения и только в режиме записи.

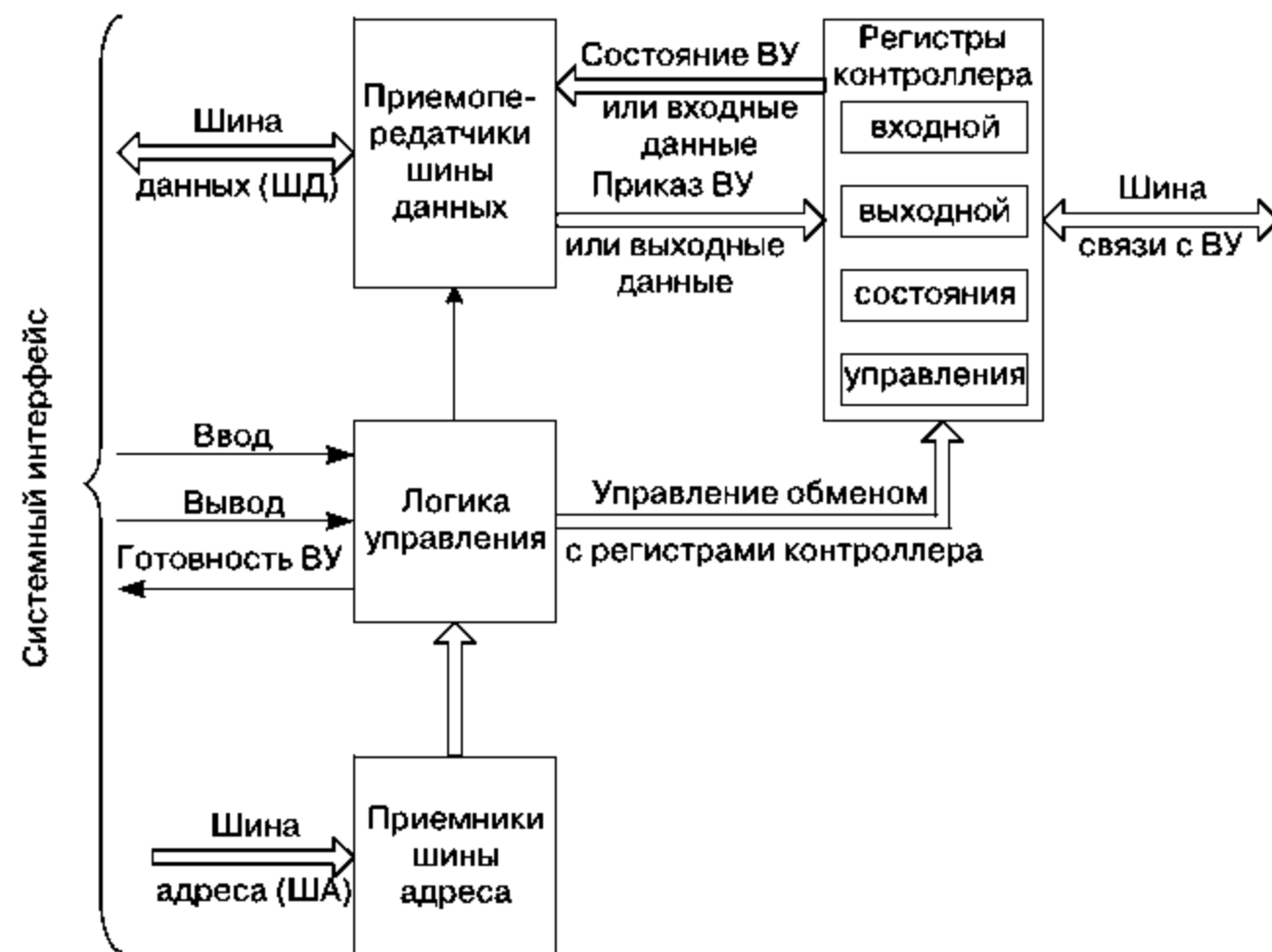


Рис. 4.52. Структура контроллера ВУ для системного интерфейса с разделенными ША и ШД

Регистр *состояния* работает только в режиме *чтения* и содержит информацию о текущем состоянии ВУ (включено-выключено, готово к обмену данными и т.п.). Регистр *управления* работает только в режиме *записи* и служит для приема из ЭВМ приказов для ВУ. В контроллерах, используемых для подключения достаточно простых ВУ (клавиатура, мышь и т.п.), удастся совместить регистры состояния и управления, что позволяет сократить число используемых в контроллере портов ввода-вывода, а следовательно, и адресов, выделенных для данного ВУ.

Приемопередатчики шин адреса и данных служат для физического подключения электронных схем контроллера к соответствующим шинам системного интерфейса.

В настоящее время используются в основном программируемые контроллеры, режимы работы которых устанавливаются специальными

командами ЭВМ и определяются программами обмена с ВУ. Программируемые контроллеры настраивают на синхронный или асинхронный режимы обмена, используя сигналы прерывания или без них, на заданную скорость обмена и т.д.

4.9. ОСНОВНЫЕ ХАРАКТЕРИСТИКИ ЭВМ

Основными характеристиками ЭВМ являются:

- ♦ *операционные ресурсы* — перечень того, что может делать ЭВМ в ее базовой конфигурации. Операционные ресурсы характеризуются системой команд, способом представления данных (ФФТ или ФПТ) и операций над ними;
- ♦ *емкость ОП* — объем информации, который можно оперативно, в темпе работы процессора обрабатывать на компьютере;
- ♦ *емкость накопителя на ЖМД* (от 10 до 300 ГБ);
- ♦ *разрядность ЭВМ и кодовых шин интерфейса* — максимальное число двоичных разрядов, над которыми одновременно могут выполняться машинные операции (чем больше разрядность, тем выше производительность);
- ♦ *быстродействие ЭВМ* — число элементарных операций, реализуемых в единицу времени, или время выполнения одной операции;
- ♦ *производительность ЭВМ* — число обобщенных операций (команд), выполняемых за единицу времени. Определяется структурой ЦП, разрядностью слова, формой представления числовых данных, частотой повторения различных операций. Рассчитывается аналитически или определяется экспериментальным путем. Результат зависит от метода и сложности оцениваемых задач. Производительность измеряется в МИПСах (MIPS) — миллион операций в секунду над числами с ФТ и МФЛОПСах (MFLOPS) — миллион операций в секунду над числами с ПТ. Для характеристики производительности ЭВМ часто указывают тактовую частоту, которая более объективно определяет быстродействие, так как выполнение любой операции требует определенного числа тактов (чем выше частота, тем меньше длительность такта, следовательно, возрастает скорость выполняемых операций);
- ♦ *виды и емкость кэш-памяти*;
- ♦ *надежность ЭВМ* — способность системы выполнить полностью и правильно все заданные ей функции. Определяет-

ся рядом показателей, имеющих вероятностный характер, а именно:

$P(t)$ — *вероятность безотказной работы*: вероятность того, что ЭВМ проработает безотказно в течение заданного интервала времени;

$T_{\text{ср}}$ — *наработка на отказ* — средняя продолжительность работы ЭВМ до отказа;

$\lambda(t)$ — *интенсивность отказов* — вероятность отказа за единицу времени после данного момента времени при условии, что до этого момента отказов не было;

- ♦ *пропускная способность (ПС)* подсистемы ввода-вывода. Число характеризуется числом каналов ввода-вывода. ПС позволяет определить возможности ЭВМ при обмене информацией с различными ВУ или другими ЭВМ. Измеряется количеством переданных в секунду байтов, килобайтов, мегабайтов.

4.10. СТРУКТУРНАЯ ОРГАНИЗАЦИЯ ПК (IBM PC)

Персональный компьютер (ПК), или *персональная ЭВМ (ПЭВМ)* — электронно-вычислительная машина, с которой может работать пользователь, не являющийся профессиональным программистом. ПК характеризуется развитым («дружественным») человекомашинным интерфейсом, малыми габаритами, относительно невысокой стоимостью и универсальностью применения. Основное влияние на архитектуру ЭВМ оказывает способ передачи данных, адресов и сигналов управления между ЦП и другими компонентами ЭВМ. Особенностью ПЭВМ является построение ее не единым неразъемным устройством, а с использованием принципов *модульности* и *магистральности*, позволяющих легко менять конфигурацию ПК и модифицировать (upgrade) устаревшие блоки и узлы. Магистральность означает наличие в системе *магистралей (системной шины)*, связывающей все входящие в систему модули (рис. 4.53).

Модуль — конструктивно законченный функциональный элемент ЭВМ, который можно подключить непосредственно к магистральной.

Модули имеют одинаковое конструктивное исполнение и одинаково включаются в систему. Они объединяются в единую ЭВМ на основе стандартной системы связи — *стандартного интерфейса* (сопряжения). Интерфейс объединения модулей в ЭВМ называется *внутримашинным интерфейсом*. Обмен информацией по магистральной осуществляется под управлением ЦП, однако часть функций обмена может быть реализована и в самих модулях с помощью *специальных электронных схем*, обеспечивающих подключение модулей к магистральной.

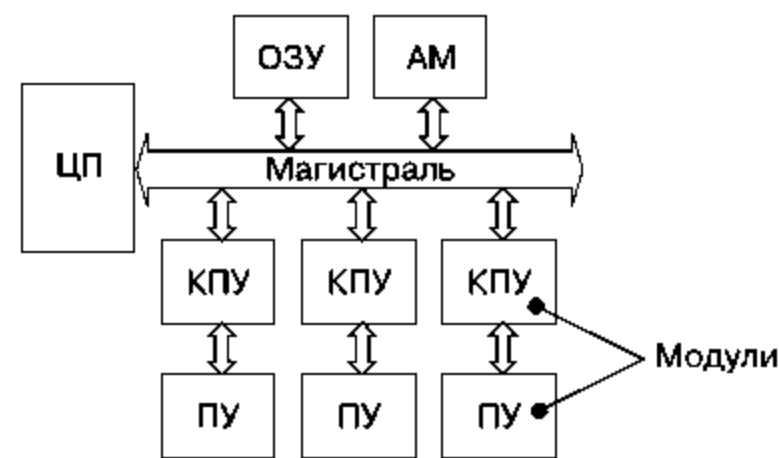


Рис. 4.53. Структура магистрально-модульной ЭВМ:

АМ — адаптер магистральной (сетевой адаптер);
 ПУ — периферийное устройство; КПУ — контроллер ПУ

В зависимости от сложности реализуемых функций и их состава эти электронные схемы называются *контроллерами* либо *контроллерами периферийных устройств* (КПУ), либо *интерфейсными блоками*, либо *адаптерами*. ПУ подключаются к магистральной через КПУ (см. рис. 4.53), например адаптер дисплея, адаптер жесткого диска и др. (см. рис. 4.51). Связь такой ЭВМ с другими машинами осуществляется через адаптер магистральной, или *сетевой адаптер*, — устройство для подключения ЭВМ (ПК) в локальную сеть, за счет чего пользователь получает доступ к данным, находящимся в других ПК, и к устройствам, отсутствующим в составе данной конфигурации ПК. Изменяя число и состав модулей, т.е. конфигурацию машины, можно изменять ее возможности.

Для подключения новых модулей и их замены требуется унификация средств обмена информацией.

Правила подключения модулей к магистральной определяются *системным внутренним интерфейсом* (*внутримашинный интерфейс*). Системный интерфейс существенно влияет на все характеристики ЭВМ, особенно на *производительность*, так как она может ограничиваться не быстродействием микропроцессора, а возможностями магистральной по скорости передачи информации.

Различают также *внешние интерфейсы* или интерфейсы периферийного оборудования, задающие правила сопряжения с устройствами ввода-вывода.

Под *интерфейсом* понимают совокупность *информационно-логических, электрических и конструктивных требований*, выполнение которых обеспечивает сопряжение различных модулей (элементов, узлов, устройств) системы и их работоспособность.

Информационно-логические требования определяют структуру и состав линий и сигналов, способы кодирования и форматы данных,

адресов, команд, протоколы обмена для различных режимов работы, которые непосредственно влияют на основные характеристики интерфейса — пропускную способность, надежность обмена, аппаратные затраты.

Электрические требования задают необходимые параметры сигналов на линиях интерфейса: уровни, длительности фронтов и самих сигналов, нагрузочные способности и т.п.

Конструктивные требования указывают на тип соединительного элемента, распределения линий по его контактам, геометрические размеры платы, каркаса и других конструктивных элементов.

При *магистральной организации* связи обмен информацией между устройствами ЭВМ происходит по трем и более *шинам*, которые заменяют все многообразие связей в ЭВМ с традиционной архитектурой (см. рис. 4.37).

Шиной (bus) называют совокупность линий, передающих сигналы одного функционального назначения.

Различают следующие основные шины *системной* магистрали:

- ♦ *шина данных* (ШД) для передачи кодов данных между модулями, которая характеризуется своей разрядностью, по сути, числом линий в шине;
- ♦ *шина адреса* (ША), передающая код адреса модуля и элементов в модуле, например ячейки памяти или регистра контроллера ввода-вывода, из которых необходимо получить информацию. В этом случае по ША передаются адрес ячейки памяти и адрес порта устройства ввода-вывода. *Адресная шина* также характеризуется разрядностью — числом линий;
- ♦ *шина управления* (ШУ), включающая линии, по которым передаются сигналы управления обменом, запросы прерывания, сигналы синхронизации и т.п.;
- ♦ *шина питания*, которая подводит питающее напряжение ко всем потребителям, подключенным к магистрали.

Упрощенная блок-схема ПК приведена на рис. 4.54.

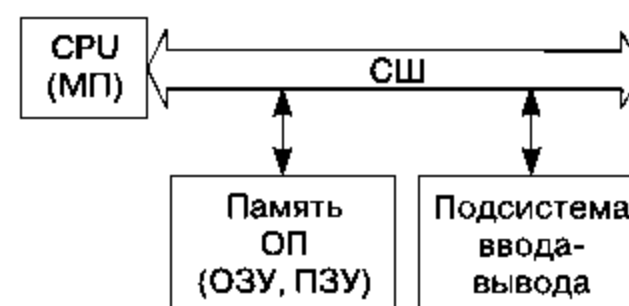


Рис. 4.54. Типовая структура ПК

ПК содержит:

- ♦ *микропроцессор* (МП), или *центральный процессор* (ЦП). Конструктивно современный МП представляет собой сверхбольшую интегральную схему (СБИС), реализованную на одном полупроводниковом кристалле — тонкой пластинке кристаллического кремния прямоугольной формы площадью всего несколько квадратных миллиметров. На этой пластинке размещены схемы, реализующие все функции процессора;
- ♦ *память* (внутреннюю, включающую ОЗУ и ПЗУ, и внешнюю);
- ♦ *подсистему ввода-вывода*;
- ♦ *контроллеры (адаптеры)* для подключения периферийных устройств к шинам МП с целью обеспечения совместимости их интерфейсов. Они осуществляют непосредственное управление ПУ по запросам микропроцессора и реализуются, как правило, на отдельных печатных платах. Важнейшую роль в работе ПК имеет *контроллер прерываний*, принимающий запрос на прерывание от ПУ. Он определяет уровень приоритета этого запроса и выдает сигнал прерывания процессору, получив который, последний приостанавливает выполнение текущей программы и переходит к выполнению программы обслуживания того прерывания, которое запросило ПУ. После завершения программы обслуживания восстанавливается выполнение прерванной программы;
- ♦ *системную шину* (СШ), обеспечивающую сопряжение и связь всех устройств между собой и включающую шину адреса ША, шину данных ШД, шину управления ШУ и шину питания.

Адресная шина большинства современных процессоров состоит из 32 параллельных линий (32-разрядная). В зависимости от того, есть напряжение на какой-то из линий или нет, она определяет, что на этой линии выставлено — единица или ноль. Комбинация из 32 нулей и единиц образует 32-разрядный адрес, указывающий на одну из ячеек оперативной памяти. К ней и подключается процессор для считывания данных из ячейки в один из своих регистров.

По *шине данных* происходит копирование данных из ОП в регистры процессора и обратно. В компьютерах, собранных на базе процессоров Intel Pentium, шина данных состоит из 64 линий (64-разрядная), по которым за один такт на обработку поступают сразу 8 байтов.

Шина управления (шина команд) в большинстве современных процессоров 32-разрядная, хотя существуют 64-разрядные процессоры и даже 128-разрядные.

Системная шина обеспечивает три направления передачи информации:

- ◆ между МП и ОП;
- ◆ между МП и портами ввода-вывода ВУ;
- ◆ между ОП и портами ввода-вывода ВУ (в режиме прямого доступа к памяти).

В состав основной (*базовой*) конфигурации ПК входит *системный блок*, включающий в себя следующие основные узлы ПК:

- ◆ *материнскую (системную) плату*, т.е. печатную плату, на которой размещены *МП, ОП, центральная магистраль (шина), контроллер шины* и несколько *слотов (разъемов)*;
- ◆ *дисплей*;
- ◆ *НГМД*;
- ◆ *CD-ROM*;
- ◆ *звуковую карту*;
- ◆ *клавиатуру*;
- ◆ *мышь*.

Все блоки, а точнее, их *порты ввода-вывода*, через соответствующие унифицированные разъемы (стыки) подключаются к шине единообразно (непосредственно или через контроллер). Порт ввода-вывода — аппаратура сопряжения, позволяющая подключить к МП другое устройство ПК. Периферийные устройства (внешние по отношению к процессору) взаимодействуют с системной шиной ПК через *порты ввода-вывода*, входящие в состав *интерфейсных БИС* системной платы и *адаптеров ВУ*. *Порт* — это 8- или 16-битный регистр, через который осуществляется взаимодействие с ВУ. Через эти порты передаются данные, слова-состояния ВУ и слова-управления от МП. Доступ к регистрам обеспечивается командами ввода-вывода IN и OUT.

Управление системной шиной (СШ) осуществляется МП либо непосредственно, либо через *контроллер шины*, который формирует основные сигналы управления. Обмен информацией между ВУ и СШ выполняется с использованием ASCII-кодов.

В *магистральной ЭВМ* в каждый момент времени связываются только те устройства, которые должны осуществлять обмен информацией. Это так называемое *временное распределение связей вместо пространственного*, используемого в ЭВМ традиционной архитектуры. Подключение требуемых элементов структуры ЭВМ осуществляется под управлением ЦП с помощью схем с *тремя состояниями* (рис. 4.55): два состояния — логические: 0 и 1, третье — высокое выходное сопротивление, что означает, по сути, обрыв выхода схем или отключе-

ние схемы от шины. Сигнал управления третьим состоянием ОЕ (активизация) и является сигналом, разрешающим прохождение информации на разряды шин (табл. 4.10).

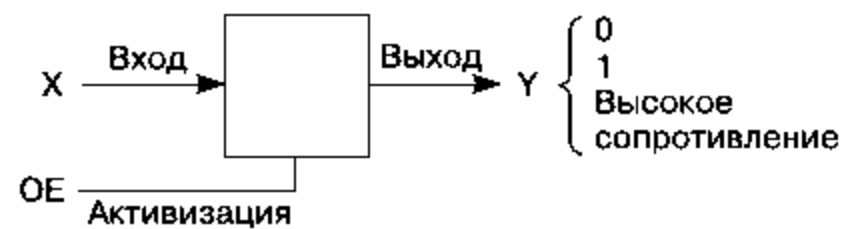


Рис. 4.55. Схема с тремя состояниями

Таблица 4.10

Табличное описание работы схемы с тремя состояниями

X	OE	Y
0	1	0
1	1	1
0	0	Высокое сопротивление
1	0	Высокое сопротивление

Все ВУ (внешние устройства) сопрягаются с помощью *контроллеров (адаптеров)*, которые вставляются в разъемы расширения (*слоты*) ПК. Такое решение обусловлено тем, что форма представления данных и управляющие сигналы в разных ВУ *существенно различаются*, так как существенно различны функции устройств, физические принципы их работы и форма взаимодействия с человеком. Данные, считываемые с дисков, представляются последовательностью импульсов, каждый из которых несет значение одного бита. Те же данные в СШ изображаются комбинацией 32 одновременно передаваемых импульсов. Для поддержки взаимодействия устройств необходимо выполнять преобразование форм информации. Эту задачу и решают специальные устройства — *адаптеры*:

- ♦ *видеоадаптеры* (они же видеоплаты, видеокарты) — это устройства, преобразующие набор данных, подлежащих отображению на экране, видеосигналы, посылаемые монитору по кабелю;
- ♦ *адаптеры портов ввода-вывода*;
- ♦ *сетевые адаптеры* (сетевые карты);
- ♦ *звуковые платы* (аудиокарты).

Основными характеристиками СШ являются:

- ♦ количество обслуживаемых устройств;
- ♦ пропускная способность (максимальная скорость передачи информации между блоками), зависящая от разрядности СШ.

В качестве системной магистрали в современных ПЭВМ используются:

- ♦ *шины расширений* — шины общего назначения, позволяющие подключать большое число разнообразных ВУ:

ISA (Industrial Standard Architecture) — 16-разрядная шина с 24-адресными линиями (адресное пространство шины — 16 МБ) работает асинхронно на частоте 8 МГц;

EISA (Extended ISA) — 32-разрядная шина данных и адреса (ее адресное пространство — 4 ГБ, производительность — до 32 МБ/с) допускает подключение 10 ВУ. Как и ISA, данный стандарт считается устаревшим;

PCI (Peripheral Component Interconnect) — синхронная 32- или 64-разрядная шина, работающая на частоте 33 или 66 МГц. При частоте до 66 МГц обеспечивает производительность 264 МБ/с для 32-разрядных данных и 528 МБ/с — для 64-разрядных данных;

FSB (Front Side Bus) — специальная шина, используемая для связи памяти и процессора, начиная с Intel Pentium Pro. Работает на частоте от 100 МГц и выше, ее пропускная способность при частоте 100 МГц составляет порядка 800 МБ/с;

AGP (Advanced Graphic Port) — усовершенствованный графический порт. Шина разработана для подключения видеоадаптера — устройства, требующего высокой скорости передачи данных;

локальные шины используются для подключения небольшого числа внешних устройств определенного класса. Среди них VLB (VESA Local Bus — локальная шина стандарта VESA) с повышенной частотой служит для связи процессора и памяти в обход основной шины. Впоследствии в эту шину «врезали» интерфейс для подключения видеоадаптера, который позволил поднять тактовую частоту локальной шины и обеспечил пиковую пропускную способность до 130 МБ/с.

Другие локальные шины: IDE (Integrated Device Electronics), EIDE (Enhanced IDE), SCSI (Smart Computer System Interface) применяются чаще всего в качестве интерфейса только для ВЗУ.

4.11. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ

В общей архитектуре современных ЭВМ и их систем (ВС) *программное обеспечение* (ПО) является наиболее гибкой компонентой, которая обеспечивает решение задачи пользователя, функционирование аппаратных компонентов и развитого пользовательского интерфейса в процессе подготовки, отладки и решения задач.

Под *программным обеспечением* понимается совокупность программ, выполняемых вычислительной системой и необходимых для эксплуатации документов.

Совокупность архитектур *аппаратной* (Hardware) и *программной* (Software) сред образует *общую* архитектуру ЭВМ (рис. 4.56), определяющую сферу ее приложений, и интерфейс с проблемной (предметной) областью, которая задается совокупностью задач, нуждающихся в вычислительных средствах.

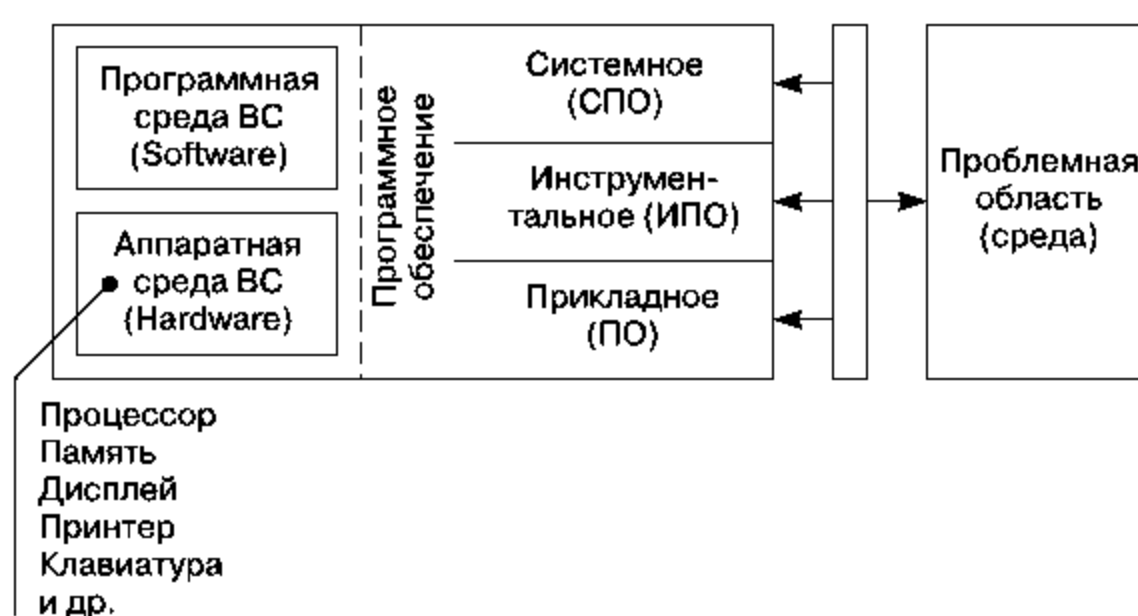


Рис. 4.56. Интерфейс общей архитектуры ВС с проблемной средой

К программному обеспечению относится также вся область деятельности по проектированию и разработке ПО:

- ◆ технология проектирования программ (например, нисходящее проектирование и объектно-ориентированное проектирование и др.);
- ◆ методы тестирования программ;
- ◆ анализ качества работы программ;
- ◆ документирование программ;
- ◆ разработка и использование программных средств, облегчающих процесс проектирования программного обеспечения, и многое другое.

Выделяют три класса программных продуктов:

- ♦ системное программное обеспечение;
- ♦ инструментарий технологии программирования;
- ♦ пакеты прикладных программ.

4.11.1. Системное программное обеспечение ЭВМ

Системное программное обеспечение (СПО) — это совокупность программ и программных комплексов, обеспечивающих работу компьютеров и сетей ЭВМ.

Системные программы служат для управления ресурсами компьютера — центральным процессором, памятью, устройствами ввода-вывода. Системное программное обеспечение разрабатывается так, чтобы компьютер мог эффективно выполнять прикладные программы. Структура СПО представлена на рис. 4.57.



Рис. 4.57. Классификация СПО

Системное ПО состоит из *базового* программного обеспечения, которое, как правило, поставляется вместе с компьютером, и *сервисного* программного обеспечения, которое может быть приобретено дополнительно.

Основной компонентом базового ПО является *операционная система* (ОС).

Решение любой задачи на компьютере требует, с одной стороны, программ, определяющих последовательность всех операций, подлежащих выполнению, а с другой — определенных аппаратных средств (ЦП, ОП и др.).

Многие процедуры управления аппаратными средствами являются более или менее *стандартными* и по существу не зависят от программных средств. Основная задача ОС заключается в *автоматизации* этих *стандартных*, часто достаточно тривиальных, процедур. Функциональные связи между ресурсами ВС можно представить в виде рис. 4.58.



Рис. 4.58. Связи между ресурсами ВС

Ресурсом является любой объект вычислительной системы, который может использоваться вычислительными процессами и соответственно разделяться между ними.

Операционная система играет роль *связующего звена* между аппаратурой компьютера, с одной стороны, и выполняемыми программами, а также пользователем — с другой, т.е. ОС можно рассматривать с *двух* точек зрения.

С *технической точки зрения* — это комплекс программного обеспечения, управляющего ресурсами компьютера, процессами обработки информации, использующими эти ресурсы и данные. *Управление ресурсами* сводится к процедурам упрощения доступа к ним и динамического распределения их между конкурирующими процессами.

С *точки зрения пользователя* ОС формирует удобный пользовательский интерфейс, программное окружение, на фоне которого выпол-

няется разработка и осуществляется выполнение *прикладной программы*, реализующей решение конкретной задачи пользователя.

Управление процессами обработки информации заключается в организации и реализации эффективных режимов функционирования компьютера, таких как:

- ◆ однопользовательский/многопользовательский;
- ◆ однозадачный/многозадачный;
- ◆ режим реального времени;
- ◆ работа в однопроцессорных, многопроцессорных, многомашинных, в том числе и сетевых, ВС и др.

Управление данными — это организация и хранение данных, обработанных на компьютере, в том числе создание библиотек и баз данных, эффективного доступа к данным и их выборки.

Операционная система обычно хранится во внешней памяти компьютера — на диске. При включении компьютера она считывается с дисковой памяти и размещается в оперативном ЗУ. Этот процесс называется *загрузкой операционной системы*. Наиболее простые и универсальные услуги ОС, связанные с осуществлением ввода-вывода, выполняет *базовая система ввода-вывода* (BIOS). В ее функции входят также автоматическое тестирование основных аппаратных компонентов (оперативной памяти и других) при включении машины и вызов блока начальной загрузки операционной системы.

В функции операционной системы входят:

- ◆ осуществление диалога с пользователем;
- ◆ ввод-вывод и управление данными;
- ◆ планирование и организация процесса обработки программ;
- ◆ распределение ресурсов (оперативной памяти, процессора, внешних устройств);
- ◆ запуск программ на выполнение;
- ◆ всевозможные вспомогательные операции обслуживания;
- ◆ передача информации между различными внутренними устройствами;
- ◆ программная поддержка работы периферийных устройств (дисплея, клавиатуры, дисковых накопителей, принтера и др.).

Характерная особенность *операционной системы ПК* — «дружественность», т.е. обеспечение необходимого *сервиса пользователю* в процессе выполнения прикладной программы и *комфортных условий программисту* для разработки и отладки программ, а также их хранения, преобразования и отображения информации.

Исходя из функциональных возможностей, ОС классифицируются:

- ◆ по числу одновременно работающих пользователей: *одно- и многопользовательские* (многопользовательские, в отличие от однопользовательских, поддерживают одновременную работу нескольких пользователей за различными терминалами);
- ◆ числу программ, одновременно работающих под управлением системы: *одно- и многозадачные*. Многозадачные поддерживают одновременное выполнение нескольких программ (задач), существующих в рамках одной вычислительной системы, и процессор распределяет ресурсы компьютера между задачами;
- ◆ типу интерфейса: *командные* (предполагают ввод пользователем команд с клавиатуры при выполнении действий по управлению ресурсами компьютера) и *объектно-ориентированные (графические)* — управление ресурсами компьютера путем осуществления операций над *объектами*, представляющими *файлы, каталоги (папки), дисководы, программы, документы* и т.д.;
- ◆ типу использования ресурсов: *локальные и сетевые*. Сетевые предназначены для управления ресурсами нескольких компьютеров, объединенных в сеть в целях совместного использования данных и других сетевых ресурсов.

Операционная система для персонального компьютера, ориентированного на профессиональное применение, должна содержать следующие основные программные компоненты:

- ◆ управление вводом-выводом;
- ◆ управление файловой системой;
- ◆ планирование процессов;
- ◆ анализ и выполнение команд, адресованных операционной системе.

Каждая операционная система имеет свой командный язык, который позволяет пользователю выполнять те или иные действия:

- ◆ обращаться к каталогу;
- ◆ выполнять разметку внешних носителей;
- ◆ запускать программы и др.

Анализ и исполнение команд пользователя, включая загрузку готовых программ из файлов в оперативную память и их запуск, осуществляет *командный процессор* операционной системы.

Наибольшее распространение для ПК получили операционные системы:

- ◆ MS-DOS (Microsoft Disk Operating System) для IBM PC;
- ◆ Windows;
- ◆ Windows NT;
- ◆ Unix и ей подобные.

Операционная система MS-DOS — самая распространенная на 16-разрядных персональных компьютерах. Включает следующие основные модули:

- ◆ блок начальной загрузки (Boot Record);
- ◆ модуль расширения BIOS (IO.SYS);
- ◆ модуль обработки прерываний (MSDOS.SYS);
- ◆ командный процессор (COMMAND.COM);
- ◆ утилиты MS-DOS.

Каждый из указанных блоков выполняет определенную часть функций, возложенных на ОС. Так, *блок начальной загрузки* (загрузчик) считывает с диска в оперативную память две другие части операционной системы — модуль расширения базовой системы ввода-вывода и модуль обработки прерываний.

Модуль расширения базовой системы ввода-вывода дает возможность использования *дополнительных драйверов*, обслуживающих новые внешние устройства, а также драйверов для нестандартного обслуживания внешних устройств.

Модуль обработки прерываний реализует основные высокоуровневые услуги DOS.

Командный процессор DOS обрабатывает команды, вводимые пользователем.

Утилиты DOS, выполняющие действия обслуживающего характера (разметка дискет, проверка дисков и др.), поставляются вместе с операционной системой в виде отдельных файлов.

Начиная с 1996 г., MS-DOS была заменена ОС Windows — 32-разрядной многозадачной операционной системой с графическим интерфейсом и расширенными сетевыми возможностями. Windows — это интегрированная среда, обеспечивающая эффективный обмен информацией между отдельными программами и предоставляющая пользователю широкие возможности обработки текстовой, графической, звуковой и видеоинформации. Интегрированность подразумевает также совместное использование ресурсов компьютера всеми программами. Эта операционная система обеспечивает работу пользователя в сети,

предоставляя встроенные средства поддержки для обмена файлами и меры по их защите, возможность совместного использования принтеров, факсов и других общих ресурсов.

В настоящее время широко применяются Windows 2000/XP, которые входят в линейку систем Windows NT, обеспечивающую многозадачность, надежность, многопроцессорную поддержку, защиту данных, удобство работы в рамках локальной сети и многое другое. Это операционные системы нового поколения для делового использования на самых разнообразных компьютерах — от портативных до серверов.

ОС Unix — многозадачная операционная система, способная обеспечить одновременную работу очень большого числа пользователей. Сейчас существуют десятки операционных систем, которые можно объединить под общим названием Unix. В основном это коммерческие версии, выпущенные производителями аппаратных платформ для компьютеров своего производства. Достоинством Unix является ее простота для понимания, изменения и переноса на другие платформы. Можно сказать, что она — одна из наиболее открытых систем. Несмотря на разнообразие версий Unix, основой всего семейства являются принципиально одинаковая архитектура и ряд стандартных интерфейсов. Переход на другую версию системы для пользователя может оказаться незаметным.

В настоящее время широко применяются достаточно мощные системы Linux. Свободное распространение и использование исходных текстов стали основой феномена Linux. При традиционной разработке программных продуктов основой служат проектирование и написание текстов, при разработке по методу Linux — макетирование, отладка и тестирование. Первые два этапа распараллелить сложно, а с отладкой и тестированием дело обстоит проще. Иными словами, разработка по методу Linux — это метод проб и ошибок, построенный на интенсивном тестировании.

Сервисные системы являются дополнительным расширением пользовательского интерфейса ОС и делятся условно:

- ◆ на интерфейсные системы;
- ◆ оболочки;
- ◆ утилиты и др.

Интерфейсные системы — мощные сервисные системы, чаще графического типа, совершенствуют не только пользовательский, но и программный интерфейс.

Операционные оболочки — это специальные программы, которые упрощают общение пользователя с командами операционной си-

стемы. Они имеют текстовый и графический варианты интерфейса конечного пользователя.

Наиболее популярными текстовыми и графическими оболочками являются:

- ◆ Norton Commander;
- ◆ Far Manager;
- ◆ Windows Commander;
- ◆ Norton Navigator и др.

Эти программы существенно упрощают задание управляющей информации для выполнения команд операционной системы и упрощают работу конечного пользователя.

Утилиты автоматизируют выполнение типовых часто используемых процедур, реализация которых потребовала бы от пользователя специальных программ.

Наиболее популярными являются:

- ◆ обслуживание МД: форматирование, восстановление удаленных файлов, а также содержимого в случае разрушения;
- ◆ обслуживание файлов и каталогов (создание, копирование, пересылка, удаление и др.);
- ◆ архивирование;
- ◆ защита от вирусов и др.

4.11.2. Инструментарий технологии программирования

Инструментарий технологии программирования — совокупность программ и программных комплексов, обеспечивающих технологию разработки, отладки и внедрения создаваемых программ.

Инструментарий включает в себя специализированные программные продукты — инструментальные средства разработчика, среди которых выделяют:

- ◆ средства для создания приложений;
- ◆ CASE-технологию (Computer-Aided System Engineering) — методы анализа, проектирования и создания программных систем, предназначенные для автоматизации процессов разработки и реализации информационных систем.

Остановимся более подробно на средствах для создания приложений, используемых в ходе разработки, корректировки и расширения программ. К ним относятся:

- ◆ языки и системы программирования;
- ◆ инструментальная среда пользователя.

Языки программирования делятся:

- ◆ *на машинные*, воспринимаемые аппаратной частью компьютера (машинные коды);
- ◆ *машинно-ориентированные* (МОЯ), которые отражают структуру конкретного типа компьютера (*ассемблеры*);
- ◆ *алгоритмические*, которые не зависят от архитектуры компьютера и используются для представления структуры алгоритма: Паскаль, Фортран и др.;
- ◆ *процедурно-ориентированные*, предназначенные для возможности описания программы как совокупности процедур (подпрограмм): C++, Паскаль и др.;
- ◆ *проблемно-ориентированные*, предназначенные для решения задач определенного класса: Лисп, Симула, Fox Base, Fox Pro и др.

Ассемблеры позволяют создавать эффективные программы, реализующие все возможности, предусмотренные системой команд и конструкцией ЭВМ. В операторах МОЯ непосредственно учитываются особенности конкретной ЭВМ.

Программы, составленные на ассемблерах, являются более короткими и требуют меньшего объема памяти. Однако разработка программ на ассемблерах предъявляет высокие требования к уровню подготовки программистов, которые должны хорошо знать конструктивные особенности ЭВМ.

При создании сложных прикладных систем с целью повышения быстродействия или сокращения требуемых объемов памяти значительное число прикладных процедур разрабатывается с помощью ассемблеров.

Помимо языка ассемблера, который дает возможность использовать все особенности машины и потому может быть применен для решения задач любого типа, существуют *языки* программирования *высокого уровня* (ЯВУ), ориентированные на различные классы задач. К ним относятся Фортран, Паскаль, Си и др. Один из самых первых языков высокого уровня — язык Фортран активно используется на персональных компьютерах и применяется главным образом при разработке прикладных систем, ориентированных на научные исследования, автоматизацию проектирования, где уже накоплены обширные стандартные библиотеки программ. Его наиболее популярной версией является Фортран-90, отдельные элементы которого реализованы в Microsoft Fortran, а в полном объеме — в Fortran PowerStation.

Классическим языком программирования является *Паскаль*, который первоначально был создан как учебный язык, а позже стал применяться для решения научных задач.

Язык *Си* применяется главным образом для создания системных и прикладных программ, в которых скорость работы и объем памяти являются критическими параметрами. Язык *Си* более точно, чем *Паскаль*, позволяет программисту учитывать аппаратные особенности ЭВМ. Основными *достоинствами* языков *Паскаль* и *Си*, необходимыми при построении больших программных систем, являются:

- ♦ возможность работы с данными сложной структуры;
- ♦ наличие развитых средств для выделения отдельных частей программ в процедуры;
- ♦ модульность, т.е. возможность независимой разработки отдельных частей программ и последующего их связывания в единую систему.

Программа, написанная на одном из алгоритмических языков высокого уровня, называется *исходной программой*. Исходная программа не может быть непосредственно выполнена на ЭВМ, так как машина понимает только язык машинных команд, т.е. *команд*, записанных в виде *двоичных кодов*. Программа, составленная из таких команд, называется *объектной программой*. Поскольку каждый процессор имеет свою систему команд, т.е. тот набор машинных команд, которые он может выполнить, исходная программа должна быть переведена (транслирована) на язык команд конкретной ЭВМ. Перевод текста программы с какого-либо языка программирования на язык команд конкретной ЭВМ называется *трансляцией*. Такой перевод осуществляется автоматически специальными программами — *трансляторами*, создаваемыми для каждого алгоритмического языка и для каждого типа компьютера (рис. 4.59).

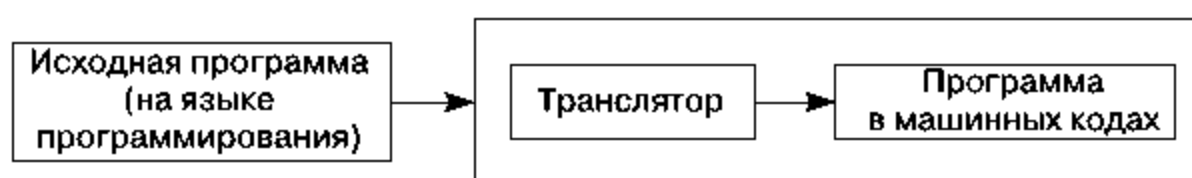


Рис. 4.59. Перевод текста программы

Выделяют три типа трансляторов:

- ♦ ассемблеры;
- ♦ интерпретаторы;
- ♦ компиляторы.

Ассемблер — самый простой вид транслятора, выполняющего перевод с машинно-ориентированного языка в машинные коды.

Трансляторы для программ, написанных на языках высокого уровня (ЯВУ), делятся на компиляторы и интерпретаторы.

Компилятор просматривает исходную программу с начала до конца, создает в машинных командах оттранслированную версию всей программы *в целом* и лишь затем передает ее на исполнение или дальнейшую обработку. *Достоинство способа* — трансляция выполняется один раз, а многократное выполнение результирующей программы может осуществляться с большой скоростью. *Недостатком* является то, что результирующая программа может занимать *большой объем ОП*, так как один оператор ЯВУ при трансляции заменяется сотнями и даже тысячами команд. *Отладка и видоизменение* транслируемой программы в этом случае весьма *затруднительны*.

Интерпретатор последовательно просматривает операторы входной программы в режиме «оператор за оператором», по ходу работы превращает каждый оператор ЯВУ в машинные команды и тут же обеспечивает его выполнение с имеющимися данными. *Интерпретирующая программа* занимает *небольшой объем памяти*, поскольку в ней в любой момент времени находятся лишь исходный текст программы и оттранслированная версия не более чем одного оператора, ее *легко исправлять и изменять*. Однако выполняется она *медленно*, поскольку при каждом исполнении заново осуществляется поочередная интерпретация всех операторов.

Инструментальные программные средства используются в ходе разработки, корректировки и расширения программ и включают:

- ♦ средства написания программ (редакторы);
- ♦ преобразования программ к виду, удобному для выполнения на ПК (ассемблеры, компиляторы, загрузчики и редакторы связей);
- ♦ средства отладки: консоль и отладка программ.

Полученный после трансляции объектный модуль выполняться не может, так как может содержать неразрешенные ссылки на другие модули или программы, а также перемещаемый код. Поэтому перед выполнением программы ее объектный модуль должен быть обработан специальной программой — *редактором связей*, разрешающим все внешние ссылки и создающим *загрузочный модуль*, и *загрузчиком*, который определяет для загрузочного модуля абсолютные адреса в ОП. После загрузчика исполняемая программа имеет расширение *exe* и ее можно

отлаживать. Отладку можно производить с помощью отладчика TD. После этого программа может выполняться. Общая схема преобразования исходной программы в выполняемый загрузочный модуль приведена на рис. 4.60.

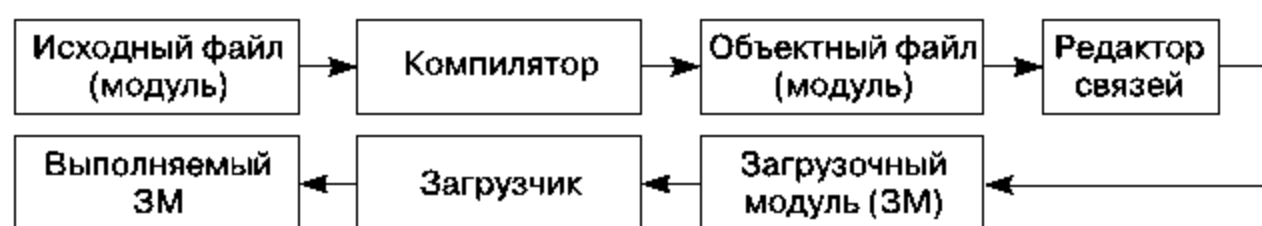


Рис. 4.60. Общая схема преобразования исходной программы

4.11.3. Прикладное программное обеспечение (пакеты прикладных программ)

Пакет прикладных программ (application program package) — комплекс взаимосвязанных программ для решения задач определенного класса конкретной предметной области.

Пакеты прикладных программ (ППП) являются самым многочисленным классом программных продуктов. В него входят программные продукты, выполняющие обработку информации различных предметных областей.

Наиболее распространенными прикладными программами являются текстовые и табличные процессоры, графические редакторы, системы управления базами данных, различные интегрированные пакеты и др.

Текстовый редактор — программа, используемая специально для ввода и редактирования текстовых данных.

Текстовые редакторы могут обеспечивать выполнение разнообразных функций, таких как:

- ◆ редактирование строк текста;
- ◆ возможность использования различных шрифтов символов;
- ◆ копирование и перенос части текста с одного места на другое или из одного документа в другой;
- ◆ контекстный поиск и замена частей текста;
- ◆ форматирование текста;
- ◆ автоматическая нумерация страниц;
- ◆ обработка и нумерация сносок;
- ◆ проверка правописания и подбор синонимов;
- ◆ распечатка подготовленного текста на принтере в нужном количестве экземпляров и т.д.

Графический редактор — программа, предназначенная для автоматизации процессов построения на экране дисплея графических изображений и предоставляющая пользователю возможности рисования линий, кривых, раскраски областей экрана, создания надписей различными шрифтами и т.д. Большинство редакторов позволяют выводить изображения в виде, готовом для включения в документ, подготовленный с помощью текстового редактора.

Табличный процессор — комплекс взаимосвязанных программ, предназначенных для обработки электронных таблиц.

Электронная таблица — компьютерный эквивалент обычной таблицы, состоящей из строк и граф, на пересечении которых располагаются ячейки, содержащие числовую информацию, формулы или текст.

Табличные процессоры представляют собой удобные средства для проведения бухгалтерских и статистических расчетов. В каждом пакете имеются сотни встроенных математических функций и алгоритмов статистической обработки данных. Кроме того, имеются мощные средства для связи таблиц между собой, создания и редактирования электронных баз данных.

База данных — один или несколько файлов данных, предназначенных для хранения, изменения и обработки больших объемов взаимосвязанной информации. В базе данных университета, например, может храниться:

- ♦ информация о факультетах и кафедрах, их составляющих, числе специальностей;
- ♦ информация о штатном расписании, преподавателях и инженерах, научных работниках и обслуживающем персонале;
- ♦ данные о ежегодно выпускаемых дипломированных специалистах;
- ♦ учебные планы;
- ♦ приказы и распоряжения администрации и т.п.

Даже небольшие изменения какой-либо информации могут приводить к значительным изменениям в других местах. Базы данных работают под управлением систем управления базами данных.

Система управления базами данных (СУБД) — система программного обеспечения, позволяющая обрабатывать обращения к базе данных, поступающие от прикладных программ конечных пользователей.

СУБД дают возможность объединять большие объемы информации и обрабатывать их, сортировать, делать выборки по определенным критериям и обеспечивают правильность, полноту и непротиворе-

чивость данных, а также удобный доступ к ним. Популярные СУБД — Oracle, Paradox и FoxPro.

Для менее сложных применений вместо СУБД используются информационно-поисковые системы (ИПС), выполняющие следующие функции:

- ◆ хранение большого объема информации;
- ◆ быстрый поиск требуемой информации;
- ◆ изменение хранимой информации, добавление и удаление;
- ◆ вывод ее в удобном для пользователя виде.

Интегрированные пакеты — набор нескольких программных продуктов, объединенных в единый удобный инструмент.

В наиболее развитые из них входят такие необходимые программы, как *текстовый редактор, электронная таблица, СУБД, программа презентаций, средство поддержки электронной почты*. При этом все части пакета составляют единое целое, что облегчает как их освоение, так и ежедневное использование.

4.12. ВЫПОЛНЕНИЕ АРИФМЕТИЧЕСКИХ И ЛОГИЧЕСКИХ ОПЕРАЦИЙ В ЭВМ

В классической фоннеймановской ЭВМ функция арифметической и логической обработки данных возлагается на арифметико-логическое устройство АЛУ (см. рис. 4.41). Минимальный вариант АЛУ должен содержать аппаратуру для реализации лишь основных логических операций, сдвигов, а также сложения и вычитания чисел в форме с ФТ. С этим набором можно программным способом обеспечить выполнение остальных арифметических и логических операций как для чисел с фиксированной точкой, так и для других форм представления информации.

Для большинства современных ЭВМ формат с фиксированной точкой является общепринятым. По этой причине соответствующие операционные устройства называются *целочисленными*. В форме с фиксированной точкой представляются как числа без знака, так и числа со знаком. Поэтому целочисленное операционное устройство должно обеспечивать выполнение арифметических операций сложения и вычитания над числами без знака и со знаком.

Одна из возможных структур операционного блока (ОБ) для сложения и вычитания чисел со знаком в форме с фиксированной точкой приведена на рис. 4.61.

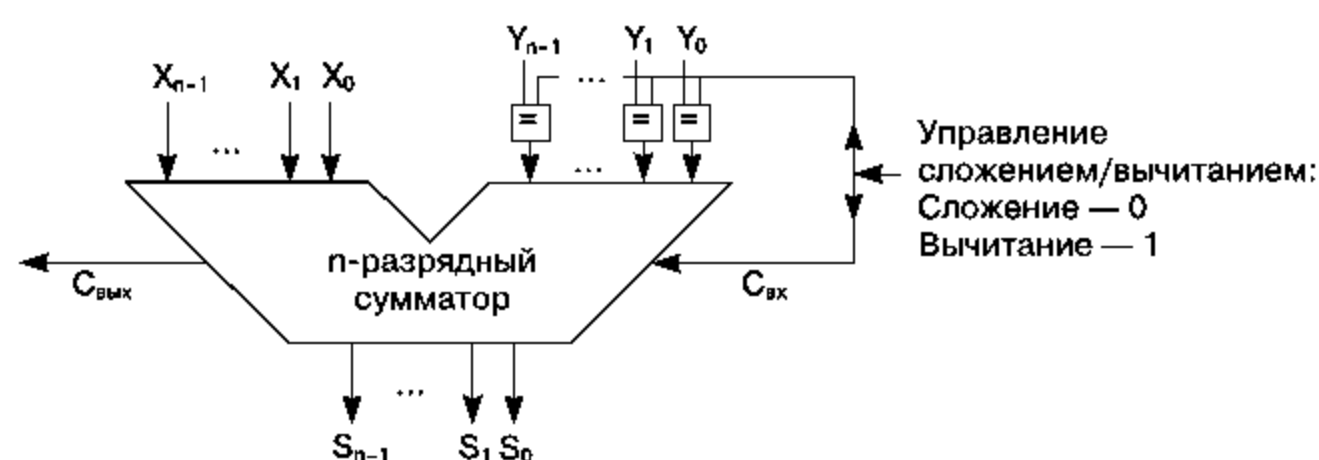


Рис. 4.61. Структура ОБ для сложения и вычитания

Центральным звеном устройства является n -разрядный двоичный сумматор. Операнд X поступает на вход сумматора без изменений. Операнд Y предварительно проходит через схемы сложения по модулю 2, поэтому вид кода Y , поступающего на другой вход сумматора, зависит от выполняемой операции. Операцию вычитания часто заменяют операцией сложения, для чего знак вычитаемого меняют на противоположный, т.е. $X - Y = X + (-Y)$. При выполнении сложения (управляющий сигнал 0) результат на выходе определяется выражением $S = X + Y$. При вычитании (управляющий сигнал 1) на вход сумматора подаются инверсные значения всех разрядов Y и, кроме того, на вход переноса $C_{вх}$ в младший разряд сумматора поступает 1. В итоге на выходе ОБ образуется сумма $S = X + \bar{Y} + 1$, что соответствует прибавлению к операнду X операнда Y с противоположным знаком, т.е. *вычитанию*.

4.12.1. Сложение целых двоичных чисел

Сложение беззнаковых чисел

При сложении чисел X и Y на основе n -разрядного АЛУ необходимо, чтобы отсутствовало переполнение, т.е. $(X + Y) < 2^n$.

Поразрядное суммирование выполняется в соответствии с выражением:

$$C_{i+1}S_i = X_i + Y_i + C_i, \quad (4.10)$$

где C_{i+1} и C_i — выходной и входной переносы соответственно;

X_i, Y_i — одноразрядные операнды;

S_i — одноразрядная сумма.

Пример. Сложить в формате байта беззнаковые числа X и Y .

$$\begin{array}{r} 1101 \leftarrow \text{переносы} \\ X = 0000\,0101_2 \\ + Y = 0000\,1101_2 \\ \hline S = 0001\,0010_2 \end{array}$$

Сложение чисел со знаком

Операция выполняется в дополнительном коде. При этом необходимо исходные числа предварительно представить в ДК. Результат также формируется в ДК.

Сложение в дополнительных кодах выполняется по правилу: сумма дополнительных кодов слагаемых равна дополнительному коду суммы:

$$[X]_{\text{ДК}} + [Y]_{\text{ДК}} = [X + Y]_{\text{ДК}},$$

если

- ♦ отсутствует переполнение, т.е. $|X + Y| \leq 2^n - 1$;
- ♦ знаковые разряды участвуют в операции наравне со значащими;
- ♦ единица переноса из знакового разряда суммы отбрасывается.

Пример 1.

$$\begin{array}{r} X = +17 = +0001\ 0001 \rightarrow [X]_{\text{ДК}} = 0.001\ 0001_2 \\ Y = -9 = -0000\ 1001 \rightarrow [Y]_{\text{ДК}} = 1.111\ 0111_2 \\ \hline [X + Y]_{\text{ДК}} = 0.000\ 1000_2 (+8). \end{array}$$

Пример 2.

$$\begin{array}{r} X = -17 = -0001\ 0001 \rightarrow [X]_{\text{ДК}} = 1.110\ 1111_2 \\ Y = +9 = +0000\ 1001 \rightarrow [Y]_{\text{ДК}} = 0.000\ 1001_2 \\ \hline [X + Y]_{\text{ДК}} = 1.111\ 1000_2. \end{array}$$

Для проверки результата необходимо перейти от дополнительного кода суммы к числовому эквиваленту: $[X + Y]_{\text{ДК}} \rightarrow (X + Y)$ на основе соответствующего Правила 1 (подраздел 4.4.3). В итоге получаем: $(X + Y) = -00001000_2 = -8$.

При этом формируются также признаки результата, а именно

$$Z = 0, S = 1, C = 0, O = 0, P = 0.$$

При сложении n -разрядных двоичных чисел (бит знака и $n - 1$ значащих цифр) возможен результат, который будет содержать n значащих цифр (ситуация, известная как *переполнение*). «Лишний» бит занимает позицию знака, что приводит к *некорректности* результата. Операционный блок должен обнаруживать факт переполнения и сигнализировать о нем. Правило обнаружения переполнения следующее: *если суммируются два положительных либо два отрицательных операнда, переполнение имеет место, когда знак результата противоположен знаку слагаемых* (рис. 4.63).

$$\begin{array}{rcl} X = +5 = +0101 \rightarrow [X]_{\text{ДК}} = 0.001_2 & X = -6 = -0110 \rightarrow [X]_{\text{ДК}} = 1.010_2 \\ Y = +4 = +0100 \rightarrow [Y]_{\text{ДК}} = 0.100_2 & Y = -5 = -0101 \rightarrow [Y]_{\text{ДК}} = 1.011_2 \\ \hline [X + Y]_{\text{ДК}} = 1.001_2 & [X + Y]_{\text{ДК}} = *0.101_2 \end{array}$$

Рис. 4.63. Сложение с переполнением

4.12.2. Вычитание целых двоичных чисел**Вычитание беззнаковых чисел**

Поразрядное вычитание с учетом заема выполняется в соответствии с выражением

$$C_{i+1}r_i = X_i - Y_i - C_i, \quad (4.11)$$

где C_{i+1} и C_i — выходной и входной заемы соответственно;

X_i, Y_i — одноразрядные операнды;

r_i — одноразрядная разность.

Правило вычитания двоичных цифр следующее:

$$\begin{array}{l} 0 - 0 = 0; \\ 1 - 1 = 0; \\ 1 - 0 = 1; \\ 0 - 1 = 11. \\ \quad \downarrow \\ \text{Заем} \end{array}$$

Пример 1.

$$\begin{array}{r} 1 \leftarrow \text{Заемы} \\ X = 17 = 0001 \ 0101 \\ - Y = 09 = 0000 \ 1001 \\ \hline (X - Y) = 0000 \ 1000 \ (8). \end{array}$$

Пример 2.

$$\begin{array}{r} 1111 \leftarrow \text{Заемы} \\ X = 09 = 0000 \ 1001 \\ - Y = 17 = 0001 \ 0001 \\ \hline (X - Y) = 1111 \ 1000 \ (-8). \end{array}$$

В случае если уменьшаемое X меньше вычитаемого Y , результат формируется в дополнительном коде, что и имеет место во втором примере:

$$[X - Y]_{\text{ДК}} = 1111 \ 1000_2; [X - Y]_{\text{ДК}} \rightarrow (X - Y) \rightarrow -0000 \ 1000_2 \ (-8).$$

Вычитание чисел со знаком

Операция выполняется в дополнительном коде. При этом необходимо исходные числа предварительно представить в ДК. Результат также формируется в ДК.

Вычитание в дополнительных кодах выполняется по правилу: *разность дополнительных кодов вычитаемого и уменьшаемого равна дополнительному коду разности*:

$$[X]_{\text{ДК}} - [Y]_{\text{ДК}} = [X - Y]_{\text{ДК}},$$

если

- ♦ $|X - Y| \leq 2^{n-1}$ (переполнение отсутствует);
- ♦ знаковые разряды участвуют в операции наравне со значащими;
- ♦ единица заема из знакового разряда разности отбрасывается.

Пример.

$$\begin{aligned}
 & \quad \quad \quad *1111 \quad \leftarrow \text{Заемы} \\
 X = 13 = +0000\ 1101 & \rightarrow [X]_{\text{дк}} = 0.000\ 1101_2 \\
 Y = -7 = -0000\ 0111 & \rightarrow [Y]_{\text{дк}} = 1.111\ 1001_2 \\
 [X - Y]_{\text{дк}} &= 0.001\ 0100_2 (+20)
 \end{aligned}$$

В большинстве операционных вычислительных устройств в целях упрощения их конструкции операция вычитания не используется. Вместо нее производится сложение дополнительных кодов уменьшаемого и вычитаемого, т.е. вычитание выполняется в соответствии с правилом: для вычитания одного числа (вычитаемого) из другого (уменьшаемого) необходимо взять дополнение вычитаемого и прибавить его к уменьшаемому. Под дополнением здесь понимается вычитаемое с противоположным знаком, представленное в дополнительном коде.

Пример.

$$\begin{aligned}
 & \quad \quad \quad *1\ 111 \quad \leftarrow \text{Переносы} \\
 X = 13 = +0000\ 1101 & \rightarrow [X]_{\text{дк}} = 0.000\ 1101_2 \\
 Y = -(-7) = +7 = +0000\ 0111 & \rightarrow [-Y]_{\text{дк}} = 0.000\ 0111_2 \\
 [X + (-Y)]_{\text{дк}} &= 0.001\ 0100_2 (+20)
 \end{aligned}$$

4.12.3. Умножение целых двоичных чисел

Умножение чисел без знака

Пусть необходимо вычислить произведение $Z = XY$, где множимое X и множитель Y — целые n -разрядные числа:

$$X = x_{n-1}, x_{n-2}, \dots, x_1, x_0; Y = y_{n-1}, y_{n-2}, \dots, y_1, y_0.$$

Представим Y в развернутом виде

$$Y = y_{n-1} \cdot 2^{n-1} + y_{n-2} \cdot 2^{n-2} + \dots + y_1 \cdot 2^1 + y_0 \cdot 2^0.$$

Тогда

$$Z = XY = (Xy_{n-1})2^{n-1} + (Xy_{n-2})2^{n-2} + \dots + Xy_1 2^1 + (Xy_0)2^0.$$

Произведение множимого на один разряд множителя $z_i = (Xy_i)2^i$ называется *частичным произведением*, а сумма k таких произведений — k -й суммой частичных произведений (СЧП). При $k = n$ СЧП превращается в произведение Z . Произведение двух n -разрядных сомножителей имеет длину, равную $2n$. Различают 4 схемы последовательного умножения: две с анализом младшего разряда множителя (со сдвигом мно-

жимого X либо СЧП) и две с анализом *старшего* разряда множителя (со сдвигом множимого X либо СЧП).

В любой из четырех схем цикл умножения на один разряд множителя y_i в общем случае включает 2 такта:

- ♦ такт суммирования τ_{Σ} ,
- ♦ такт сдвигов $\tau_{\text{сдв}}$.

В трех из них такт суммирования является первым. При умножении с анализом старшего разряда Y со сдвигом множимого X первым считается такт сдвига.

Структурная схема такого умножения приведена на рис. 4.64.

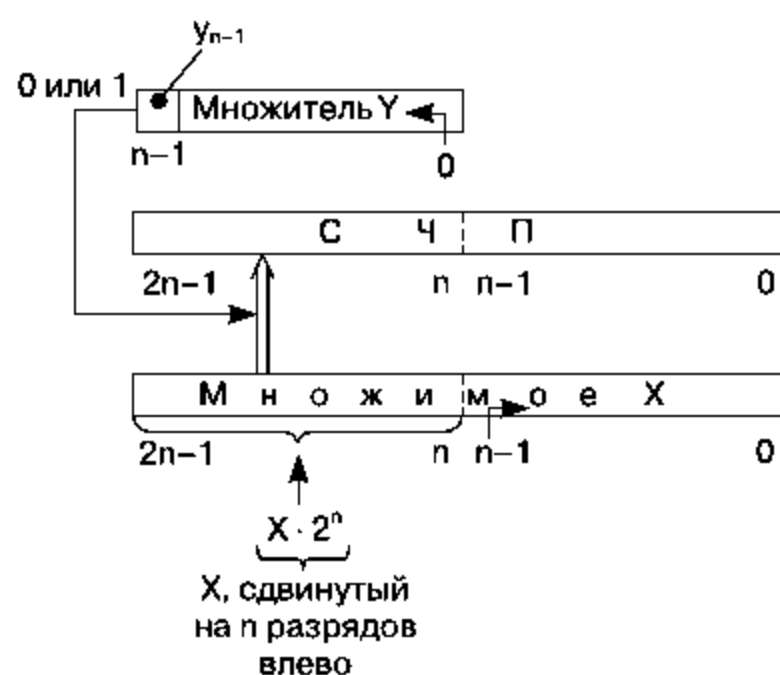


Рис. 4.64. Структурная схема

Аналитически умножение описывается выражением

$$Z = XY = 0 + \underbrace{(X \cdot 2^{n-1})}_{(X \cdot 2^n)2^{-1}} y_{n-1} + \underbrace{(X \cdot 2^{n-2})}_{(X \cdot 2^{n-1})2^{-1}} y_{n-2} + \dots + (X \cdot 2^1) y_1 + (X \cdot 2^0) y_0.$$

Множимое X , сдвинутое вправо на 1 разряд.

Схема алгоритма умножения приведена на рис. 4.65. Особенность его состоит в том, что анализируется старший разряд множителя Y , *выдвинутый* из регистра, который может храниться во флажке C — флажке переноса.

Пример умножения 4-разрядных сомножителей ($X = 13 = 1101_2$, $Y = 10 = 1010_2$) приведен в табл. 4.11.

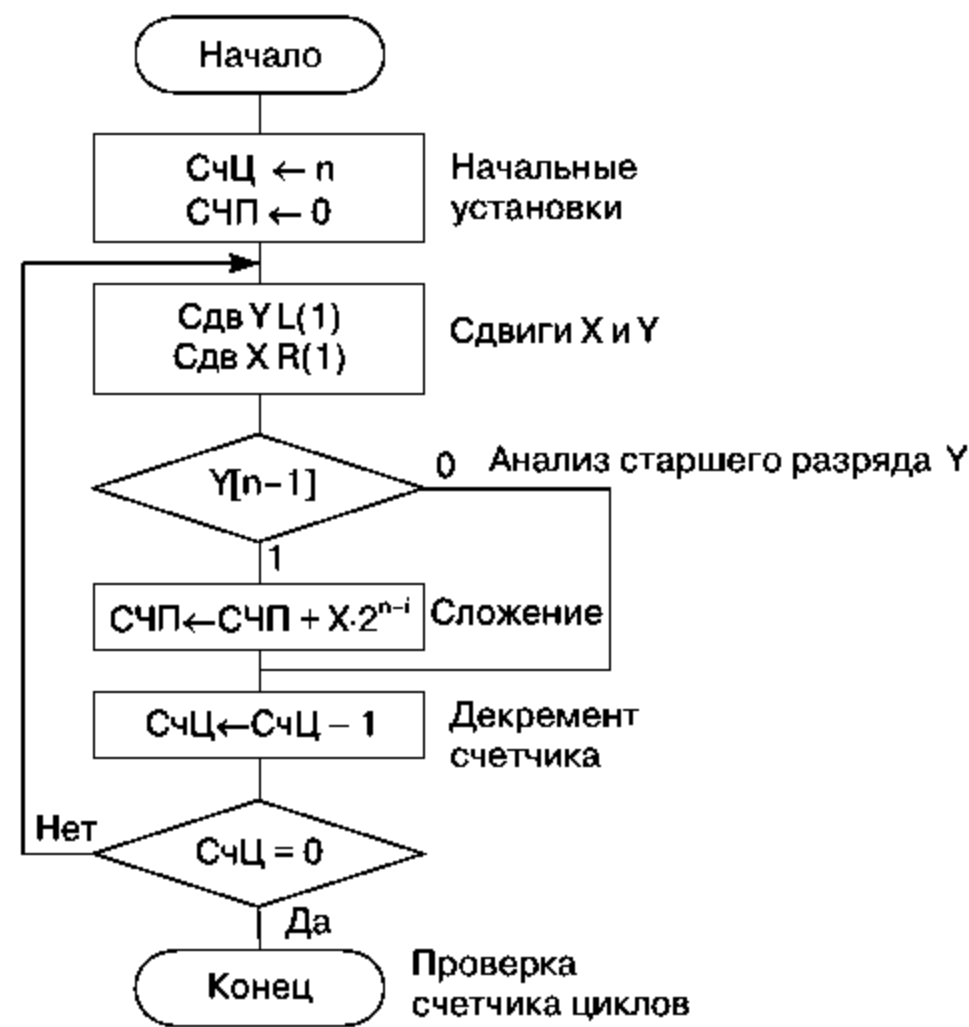


Рис. 4.65. Схема алгоритма

Таблица. 4.11

Трасса исполнения циклов

Цикл	Такт	Y	X	СЧП	СЧЦ
Начальное состояние		1010	1101 0000 ($X \cdot 2^4$)	0000 0000	СЧЦ ← 4
1	$\tau_{сдв}$	10100	0110 1000		
	τ_{Σ}			$\begin{array}{r} 0000\ 0000 \\ + 0110\ 1000 \\ \hline 0110\ 1000 \end{array}$	СЧЦ ← (4-1)
2	$\tau_{сдв}$	01000	0011 0100		СЧЦ ← (3-1)
3	$\tau_{сдв}$	10000	0001 1010		
	τ_{Σ}			$\begin{array}{r} 0110\ 1000 \\ + 0001\ 1010 \\ \hline 1000\ 0010 \end{array}$ $Z = X \cdot Y$	СЧЦ ← (2-1)
4	$\tau_{сдв}$	00000	0000 1101		СЧЦ ← (1-1)

Умножение чисел со знаком

Для отрицательных чисел, хранящихся, как правило, в памяти компьютера в дополнительном коде, удобнее использовать один из алгоритмов умножения в ДК, например умножение в ДК с преобразованием, при котором оба сомножителя преобразуются (инвертируются, а затем увеличиваются на единицу младшего разряда: $X_{пр} = [\bar{X}]_{ДК} + 1$, $Y_{пр} = [\bar{Y}]_{ДК} + 1$), если Y отрицательное число, либо остаются неизменными при $Y \geq 0$. Схема умножения может быть любой из 4 схем, названных выше.

Пример.

Найти $Z = XY$, используя схему умножения с анализом младшего разряда Y со сдвигом СЧП, если

$$X = +3 = +0011_2, Y = -5 = -0101_2; [X]_{ДК} = 0.011_2, [Y]_{ДК} = 1.011_2.$$

Поскольку $Y < 0$, требуется преобразование сомножителей:

$$X_{пр} = [\bar{X}]_{ДК} + 1 = \begin{array}{r} 1100_2 \\ + 1 \\ \hline 1101_2 \end{array}; Y_{пр} = [\bar{Y}]_{ДК} + 1 = \begin{array}{r} 0100_2 \\ + 1 \\ \hline 0101_2 \end{array}.$$

Трасса исполнения циклов приведена в табл. 4.12.

Таблица 4.12

Умножение в ДК с преобразованием

Цикл	Такт	Y	СЧП	СчЦ
Начальное состояние		0101	0000 0000	СчЦ ← 4
1	τ_{Σ}		$\begin{array}{r} + 1101 \leftarrow X \cdot 2^4 \\ \hline 1101 0000 \end{array}$	
	$\tau_{сдв}$	0010	1110 1000	СчЦ ← (4-1)
2	$\tau_{сдв}$	0001	1111 0100	СчЦ ← (3-1)
3	τ_{Σ}		$\begin{array}{r} + 1101 0000 \\ \hline 11100 0100 \end{array}$	
	$\tau_{сдв}$	0000	1110 0010	СчЦ ← (2-1)
4	$\tau_{сдв}$	0000	1111 0001	СчЦ ← (1-1)

$[Z]_{ДК}$

После перехода от кода к числу ($[Z]_{ДК} \rightarrow Z$) получим $Z = -00001111_2 = -15$.

Реализация умножения путем последовательного выполнения операций сложения и сдвига в течение n тактов занимает значительное время, что является недопустимым для ряда применений. Поэтому разработаны различные алгоритмы ускоренного умножения, основанные на одновременном анализе нескольких битов множителя, и быстродействующие *однотактные* умножители, например, матричные, которые выпускаются в виде отдельных микросхем либо входят в состав БИС в качестве операционных узлов.

4.12.4. Деление целых двоичных чисел

Если умножение сводится к последовательности сложений и сдвигов, то *деление* — к последовательности *вычитаний* и сдвигов. Операция реализуется циклическими действиями. Вначале делитель Y вычитается из делимого X , а затем из остатков A_i .

При делении целых чисел получают целую часть частного и остаток W , как дополнительный результат (округление не производится)

$$Z = \underbrace{\left] \frac{X}{Y} \right[}_{\text{Целая часть}} + \underbrace{\left] \frac{W}{Y} \right[}_{W - \text{остаток}}$$

Различают два алгоритма деления:

- ♦ с восстановлением остатка;
- ♦ без восстановления остатка.

Деление с восстановлением остатка

Получение цифры частного Z_i включает в себя три действия:

- 1) вычитание $X - Y$, а затем $A_i - Y$.

По знаку остатка A_i определяется цифра частного:

$$Z_i = \begin{cases} 1, & \text{если } A_i \geq 0; \\ 0, & \text{если } A_i < 0; \end{cases}$$

- 2) восстановление отрицательного остатка, т.е. $A_i + Y$;
- 3) сдвиг $X(A_i)$ влево на один разряд. В *последнем цикле сдвиг не производится!*

Возможны два варианта деления: со сдвигом делимого X (остатков A_i) либо со сдвигом делителя Y . На практике часто используют

первый вариант (рис. 4.66, а). Форматы операндов для него приведены на рис. 4.66, б.

Остаток W от деления размещается в старших разрядах регистра делимого.



Рис. 4.66. Деление со сдвигом делимого:

а — схема деления; б — форматы данных и результата

Особенности деления

1. Частное получается, начиная со старших разрядов. Установку цифры частного производят путем выполнения операции $\text{INC } Z$ ($Z \leftarrow Z + 1$). При формировании следующей цифры частного z_i частное Z сдвигается влево на один разряд.
2. Перед началом деления X и Y сравниваются на наличие переполнения, которое имеет место, если старшая половина X больше делителя Y . Допустим, $X = \underline{11} 00_2$ (12_{10}), $Y = 10_2$ (2_{10}). Поскольку $X_{\text{ст}} = 11_2 > 10_2$, то будет переполнение. Действительно, в соответствии с рис. 4.66, б под целую часть частного отводится 2 разряда и результат $Z = 110_2$ (6_{10}) не размещается в отведенной разрядной сетке. Поскольку частное Z формируется, начиная со старшего разряда, то старшая цифра его будет потеряна и результат искажен. Схема алгоритма приведена на рис. 4.67.

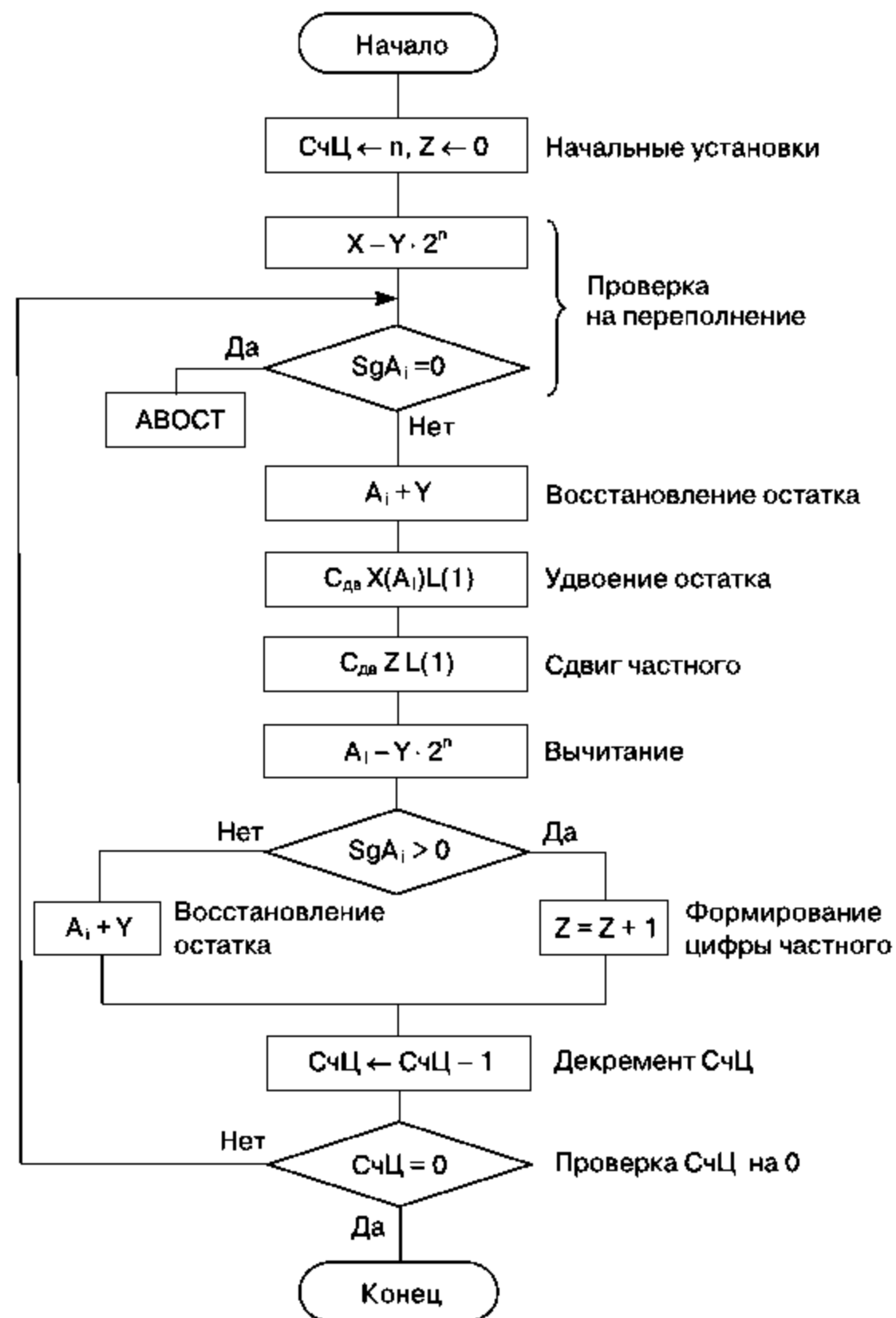


Рис. 4.67. Схема алгоритма

Пример. Найти $Z = X/Y$, если $X = 25 = 00011001_2$, $Y = 4 = 0100_2$.

	$X_{\text{ст}}$	$X_{\text{мл}}$			
0-й цикл	$\left[\begin{array}{r} 0001\ 1001 \\ -0100 \\ \hline 1101\ 1001 \\ +0100 \\ \hline 0001\ 1001 \end{array} \right]$	(X) $-(Y \cdot 2^4)$ $A_0 < 0, V = 0, Z_4 = 0$ $+Y$ (восстановление остатка)	$\left[\begin{array}{c} \text{Pr } Z \\ 0000. \\ \leftarrow \end{array} \right]$	3	↑ Место точки
1-й цикл	$\left[\begin{array}{r} 0011\ 0010 \\ -0100 \\ \hline 1111\ 0010 \\ +0100 \\ \hline 10011\ 0010 \end{array} \right]$	$(2A_0)$ Сдв. A_0 L(1) $-Y$ (вычитание) $A_1 < 0, Z_3 = 0$ $+Y$ (восстановление остатка)	$\left[\begin{array}{c} 0000. \\ \uparrow \end{array} \right]$		
2-й цикл	$\left[\begin{array}{r} 0110\ 0100 \\ -0100 \\ \hline 0010\ 0100 \end{array} \right]$	$(2A_1)$ Сдв. A_1 L(1) $-Y$ (вычитание) $A_2 > 0, Z_2 = 1$	$\left[\begin{array}{c} 0001. \\ \uparrow \end{array} \right]$	$Z \leftarrow Z+1$	
3-й цикл	$\left[\begin{array}{r} 0100\ 1000 \\ -0100 \\ \hline 0000\ 1000 \end{array} \right]$	$(2A_2)$ Сдв. A_2 L(1) $-Y$ (вычитание) $A_3 > 0, Z_1 = 1$	$\left[\begin{array}{c} 0010. \\ \leftarrow \\ 0011. \\ \uparrow \end{array} \right]$	Сдв. Z L(1) $Z \leftarrow Z+1$	
4-й цикл	$\left[\begin{array}{r} 0001\ 0000 \\ -0100 \\ \hline 1101\ 0000 \\ +0100 \\ \hline 0001\ 0000 \end{array} \right]$	$(2A_3)$ Сдв. A_3 L(1) $-Y$ (вычитание) $A_4 < 0, Z_0 = 0$ $+Y$ (восстановление остатка)	$\left[\begin{array}{c} 0110. \\ \leftarrow \\ \uparrow \end{array} \right]$	Сдв. Z L(1)	
	$\underbrace{0001\ 0000}_{W \text{ (остаток)}}$				

Примечание. Знак остатка A_i удобно определять по значению флажка C :

$$C = \begin{cases} 1, & \text{если } A_i = (X - Y) < 0; \\ 0, & \text{если } A_i = (X - Y) \geq 0. \end{cases}$$

Деление без восстановления остатка

Алгоритм такого деления сводится к следующим этапам:

1. Остаток A_i *всегда* удваивается.

Анализируется знак остатка ($\text{Sg } A_i$), по которому определяется цифра частного

$$Z_i = \begin{cases} 1, & \text{если } \text{Sg } A_i = 0; \\ 0, & \text{если } \text{Sg } A_i = 1, \end{cases}$$

и выполняется одна из двух операций:

- а) вычитание $(2A_i - Y)$, если остаток положительный;
- б) сложение $(-2A_i + Y)$, если остаток отрицательный.

Примечание. В последнем цикле вместо удвоения остатка его восстанавливают.

Пример. Выполнить деление тех же чисел ($X = 25$, $Y = 4$) без восстановления остатка со сдвигом делимого.

	$X_{ст}$	$X_{мл}$		
0-й цикл	0001 1001	(X)		Рг Z
	$\overline{0100}$	$-(Y \cdot 2^4)$		$\begin{array}{ c } \hline 0000 \\ \hline \end{array}$
	11101 0001	$A_0 < 0, V = 0, Z_4 = 0$		$\begin{array}{ c } \hline 3 \quad 0 \\ \hline \end{array}$
1-й цикл	1011 0010	$(-2A_0)$, Сдв. $A_0L(1)$		
	$\overline{0100}$	$+Y$ (сложение)		$\begin{array}{ c } \hline 0000 \\ \hline \end{array}$
	1111 0010	$A_1 < 0, Z_3 = 0$		
2-й цикл	1110 0100	$(-2A_1)$		
	$\overline{0100}$	$+Y$ (сложение)		$\begin{array}{ c } \hline 0001 \\ \hline \end{array}$
	10010 0100	$A_2 > 0, Z_2 = 1$		
3-й цикл	0100 1000	$(2A_2)$		$\begin{array}{ c } \hline 0010 \\ \hline \end{array}$
	$\overline{0100}$	$-Y$ (вычитание)		$\begin{array}{ c } \hline 0011 \\ \hline \end{array}$
	0000 1000	$A_3 > 0, Z_1 = 1$		
4-й цикл	0001 0000	$(2A_3)$		$\begin{array}{ c } \hline 0110 \\ \hline \end{array}$
	$\overline{0100}$	$-Y$ (вычитание)		
	1101 0000	$A_4 < 0, Z_0 = 0$		
	$\overline{0100}$	$+Y$ восстановление остатка		
	0001 0000			
	W (остаток)			
Результат деления (Z)				$\begin{cases} \begin{array}{ c } \hline 0110 \\ \hline \end{array}]Z[\\ \begin{array}{ c } \hline 0001 \\ \hline \end{array} W \end{cases}$

4.12.5. Выполнение в ЭВМ логических операций

Помимо арифметических операций, операционный блок любой вычислительной машины предполагает выполнение основных логических операций и сдвигов. Чаще всего такие операции реализуются дополнительными схемами, входящими в состав целочисленных операционных блоков. К базовым логическим операциям относятся: *логическое отрицание* (НЕ), или *инверсия*; *логическое сложение* (ИЛИ), или *дизъюнкция*; *логическое умножение* (И), или *конъюнкция*; *сложение по модулю два* (исключающее ИЛИ).

Логические операции выполняются в ЭВМ над *логическими переменными*:

$X = X_{n-1} X_{n-2} \dots X_1 X_0$ — логическая переменная X ,

$Y = Y_{n-1} Y_{n-2} \dots Y_1 Y_0$ — логическая переменная Y ,

где $X_i, Y_i \in \{0, 1\}$.

Логическая переменная, как и другой машинный элемент информации, представляется в виде двоичного слова.

Конъюнкция ($Z = X \& Y$)

Эта операция реализуется путем выполнения поразрядных операций конъюнкции $Z_i = X_i \& Y_i$ в соответствии с табл. 4.13.

Таблица 4.13

Конъюнкция

X_i	Y_i	Z_i
0	0	0
0	1	0
1	0	0
1	1	1

Пример.

$X =$ 0000 1111

$Y =$ 0110 1011

$Z =$ 0000 1011

Переносы между разрядами результата отсутствуют. Часто операцию конъюнкции называют операцией *маскирования*. При этом одну из переменных называют *маской*. С помощью операции маскирования можно выделять один или группу бит, определять модуль числа, знак числа и т.д.

Пример. Выделить модуль переменной X .

$X = X_7 X_6 X_5 X_4 X_3 X_2 X_1 X_0$

$Y = 0 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1 \ 1$ — маска определения $|X|$.

$Z = X \& Y = 0 \ X_6 X_5 X_4 X_3 X_2 X_1 X_0$

Частным случаем конъюнкции является операция *тестирования* $X \& Y$, при выполнении которой устанавливаются только признаки результата операции конъюнкции без формирования его значения.

Дизъюнкция ($Z = X \vee Y$)

Операция реализуется путем выполнения поразрядных операций дизъюнкции $Z_i = X_i \vee Y_i$ в соответствии с табл. 4.14 и может быть использована для образования какого-либо нового слова из частей слова.

Таблица 4.14

Дизъюнкция

X_i	Y_i	Z_i
0	0	0
0	1	1
1	0	1
1	1	1

Пример.

$$\begin{array}{r} X = \vee 0000 \quad 1001 \\ Y = \vee 1011 \quad 0000 \\ Z = 1011 \quad 1001 \end{array}$$

Сложение по модулю два

(исключающее или) — ($Z = X \oplus Y$)

Операция над переменными реализуется поразрядно в соответствии с табл. 4.15 и может быть использована для обнуления аккумулятора, определения знака произведения или частного путем сложения знаков операндов по модулю два.

Таблица 4.15

Сложение по модулю два

X_i	Y_i	Z_i
0	0	0
0	1	1
1	0	1
1	1	0

Пример.

$$\begin{array}{r} X = \oplus 1000 \quad 1110 \\ Y = \oplus 1000 \quad 1110 \\ Z = 0000 \quad 0000 \end{array}$$

Инверсия (отрицание) — ($X = \bar{X}$)**Пример.**

$$X = 00000110, \bar{X} = 11111001.$$

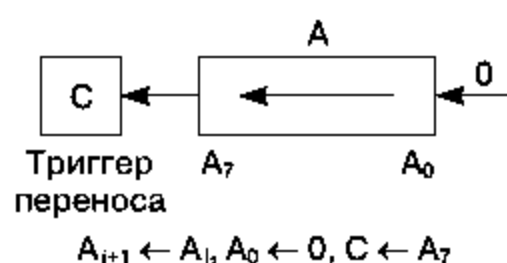
Сравнение ($X - Y$)

При сравнении формируются только признаки результата вычитания $X - Y$, а сам результат нигде не фиксируется.

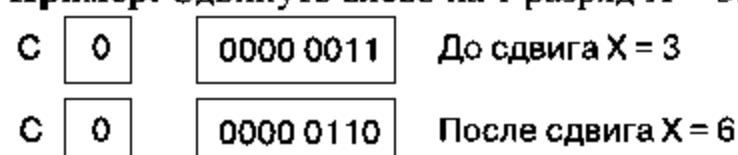
Сдвиги двоичных слов

Различают следующие типы сдвигов: логические, арифметические и циклические.

При логическом сдвиге слова влево (соответствует умножению на два: $A \leftarrow 2A$) содержимое всех его бит сдвигается влево на 1 разряд. Освобождающийся разряд заполняется нулем, а выдвинутый бит чаще всего размещается в триггере переноса C . Схема сдвига для 8-разрядного регистра-аккумулятора приведена на рис. 4.68.

**Рис. 4.68.** Логический сдвиг влево на один разряд**Рис. 4.69.** Логический сдвиг вправо на один разряд

Пример. Сдвинуть влево на 1 разряд $X = 3$.



При логическом сдвиге вправо (соответствует делению на два: $A \leftarrow A/2$) содержимое всех его бит сдвигается вправо на 1 разряд (рис. 4.69).

Арифметический сдвиг влево полностью совпадает с логическим сдвигом влево (при представлении числа в ДК). Если число задано в ПК, то старший (знаковый) разряд не изменяется:

до сдвига: $[X]_{\text{ПК}} = 1000100 \quad (-4);$
 после сдвига: $[X]_{\text{ПК}} = 1001000 \quad (-8).$

При арифметическом сдвиге *вправо* содержимое старшего (знакового) разряда не изменяется (рис. 4.70).

**Рис. 4.70.** Арифметический сдвиг вправо на один разряд

При *циклическом сдвиге* слова *вправо* выдвинутый бит размещают в триггере переноса C , а в освобождающийся разряд A_7 размещают бит из триггера переноса (рис. 4.71).

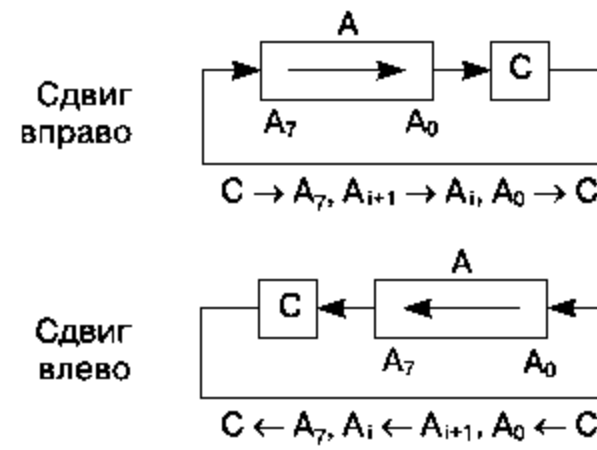


Рис. 4.71. Циклический сдвиг на один разряд

Если в системе команд нет операции логического или арифметического сдвигов, то любой из них можно реализовать с помощью циклического сдвига. Последний используется также для организации сдвига слов двойной длины.

ОСНОВНЫЕ ТРЕБОВАНИЯ К ВЫПОЛНЕНИЮ ЛАБОРАТОРНЫХ РАБОТ

Предлагаемые работы являются двухчасовыми, требуют предварительной (домашней) подготовки, четкой организации и творческого подхода во время выполнения в лаборатории. Процесс выполнения каждой работы состоит из трех этапов:

- ♦ подготовка к работе;
- ♦ непосредственное выполнение;
- ♦ отчет о выполнении работы.

Подготовка к работе предполагает ознакомление с описанием предстоящей работы как по методическим указаниям учебного пособия, которые содержат необходимый минимум теоретических сведений о работе, порядок и методику выполнения, примеры выполнения практических заданий, так и по рекомендуемым дополнительным источникам*. Этап включает также подготовку отчета (теоретической и практической частей).

Готовность к работе проверяется на этапе *входного контроля*, при котором каждый студент должен ответить на 1—2 тематических вопроса из серии «что и как делать?» и показать аккуратно подготовленный отчет.

**Студент, не имеющий отчета и не прошедший входной контроль,
к работе не допускается!**

Непосредственное выполнение работы делится на две части. Первая часть содержит общие для всех упражнения, поясняющие методику выполнения работы и ее особенности. Вторая часть включает в себя индивидуальные задания, во время выполнения которых студенты получают практические навыки работы в среде Windows (работа с элементами интерфейса ОС, с антивирусными программами, создание собст-

* Для выполнения лабораторных работ 11—16 необходимо загрузить программное обеспечение с сайта Рязанского государственного радиотехнического университета (электронный адрес: www.rsreu.ru/content/view/1729/1288).

венных каталогов и папок, текстовых и графических документов, выполнение простых арифметических операций и др.).

Подготовленный дома отчет на этом этапе может корректироваться и дополняться в случае, если не реализуется составленная программа, описанный порядок выполнения упражнений не приводит к нужному результату и т.д. Необходимые исправления и изменения вносятся в отчет в оперативном порядке.

Каждый студент *должен отчитаться* за практическую часть работы, выполнив индивидуальное задание по указанию преподавателя. Например, скопировать текстовый файл, вернуть файл из корзины, создать собственную папку, удалить указанный каталог, создать фрагмент рисунка и т.д. По результатам выполнения индивидуального задания выставляется отметка в журнале.

Отчет о выполнении работы. Работа считается *полностью выполненной* после *защиты отчета* и при наличии *отметки* о выполнении практической части. *Отчет* о лабораторной работе — индивидуальный творческий документ о ее выполнении, который должен содержать:

- ◆ краткое изложение основных теоретических положений по изучаемой теме;
- ◆ четко сформулированный вариант индивидуального задания;
- ◆ практические результаты в виде таблиц, графиков, расчетов и т.д. с описанием способов их получения.

Перечень вопросов, которые должны быть рассмотрены в теоретической и практической частях, приводится в описании на каждую работу отдельно.

Содержание отчета *не должно* быть механической копией методических указаний. Поскольку это разные документы, преследующие разные цели, то и подход к изложению одних и тех же вопросов в них должен быть различным. Если в *указаниях* описывается методика выполнения работы, то в *отчете* должны быть представлены результаты, полученные на основе этой методики, способы их достижения, используемые средства, что определяется уже индивидуальными способностями пользователя, его умением выделять главное, пользоваться дополнительной литературой и др.

Приведенная в отчете информация должна быть релевантной (полезной) и полной, т.е. достаточной, чтобы дать ответы на все поставленные вопросы и продемонстрировать, насколько глубоко пользователь разобрался в изучаемом вопросе. Отчет должен быть выполнен по ГОСТу,

т.е. все таблицы, рисунки, графики должны быть пронумерованы, иметь ссылки по тексту и т.д.

Пример типичной ошибки

Приведенные в отчетах таблицы пронумерованы так, как в методических указаниях, например 3 и 4, в то время как отчет — это новый документ и нумерация таблиц должна начинаться с единицы.

Для защиты отчета необходимо правильно и полно ответить на 2—3 вопроса преподавателя (из числа контрольных) либо решить задачу.

Лабораторная работа 1

ОСНОВЫ РАБОТЫ В WINDOWS 2000/XP: ИЗУЧЕНИЕ ИНТЕРФЕЙСНЫХ ЭЛЕМЕНТОВ И РАБОТА С НИМИ

1.1. Цель работы

1. Изучение модели процесса обработки информации на ПК и функций объектов, участвующих в этом процессе (ОС, ПП, пользователя).
2. Знакомство с ОС Windows 2000/XP.
3. Изучение интерфейсных элементов и получение практических навыков работы с ними (работа с мышью, окнами, программами).

1.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 7—9].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (понятие ОС, общее представление о Windows, отличительные особенности, функциональные возможности, особенности технологии работы на ПК под управлением Windows, основные интерфейсные элементы: меню, окна, мышь, кнопки, ярлыки и др.).
4. Практическая работа с элементами интерфейса (подраздел 1.8).
5. Отчет о проделанной работе:
 - ♦ индивидуальное выполнение на ПК операций по заданию преподавателя;
 - ♦ оформление отчета;
 - ♦ защита отчета.

1.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ основные сведения об операционных системах;
- ◆ основные сведения об ОС Windows, особенности;
- ◆ модель работы пользователя при работе на ПК под управлением ОС Windows 2000/XP (рис. Л-1.1);
- ◆ основные интерфейсные элементы, их функции и работа с ними.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ протоколы выполнения заданий 1, 2 и 3.

1.4. Компьютер и пользователь.

Место и роль операционной системы

Современный компьютер обладает мощными аппаратными (память, процессор, устройства ввода-вывода и др.) и программными (текстовые и графические редакторы, СУБД, ЭТ и т.д.) средствами обработки информации, представленной в виде звука (речь, музыка), видеоизображения, графики, текста, баз данных (каталоги, справочники), таблиц и т.д. и хранящейся в файлах в его памяти (гибкие диски, жесткие диски, CD-ROM и др.).

В процессе обработки информации над ней выполняется множество операций:

- ◆ поиск;
- ◆ чтение (ввод);
- ◆ запись (сохранение);
- ◆ редактирование (изменение, удаление, вставка, копирование);
- ◆ обработка (получение новых данных на основе имеющихся);
- ◆ преобразование формы;
- ◆ отображение (представление в форме, удобной для пользователя) и т.д.

Каждая операция реализуется прикладной программой.

По сути, работа компьютера заключается в исполнении прикладных программ (приложений Windows), средствами которых ведется обработка информации.

Модель обработки информации пользователем на компьютере, место ОС и прикладных программ представлена на рис. Л-1.1.

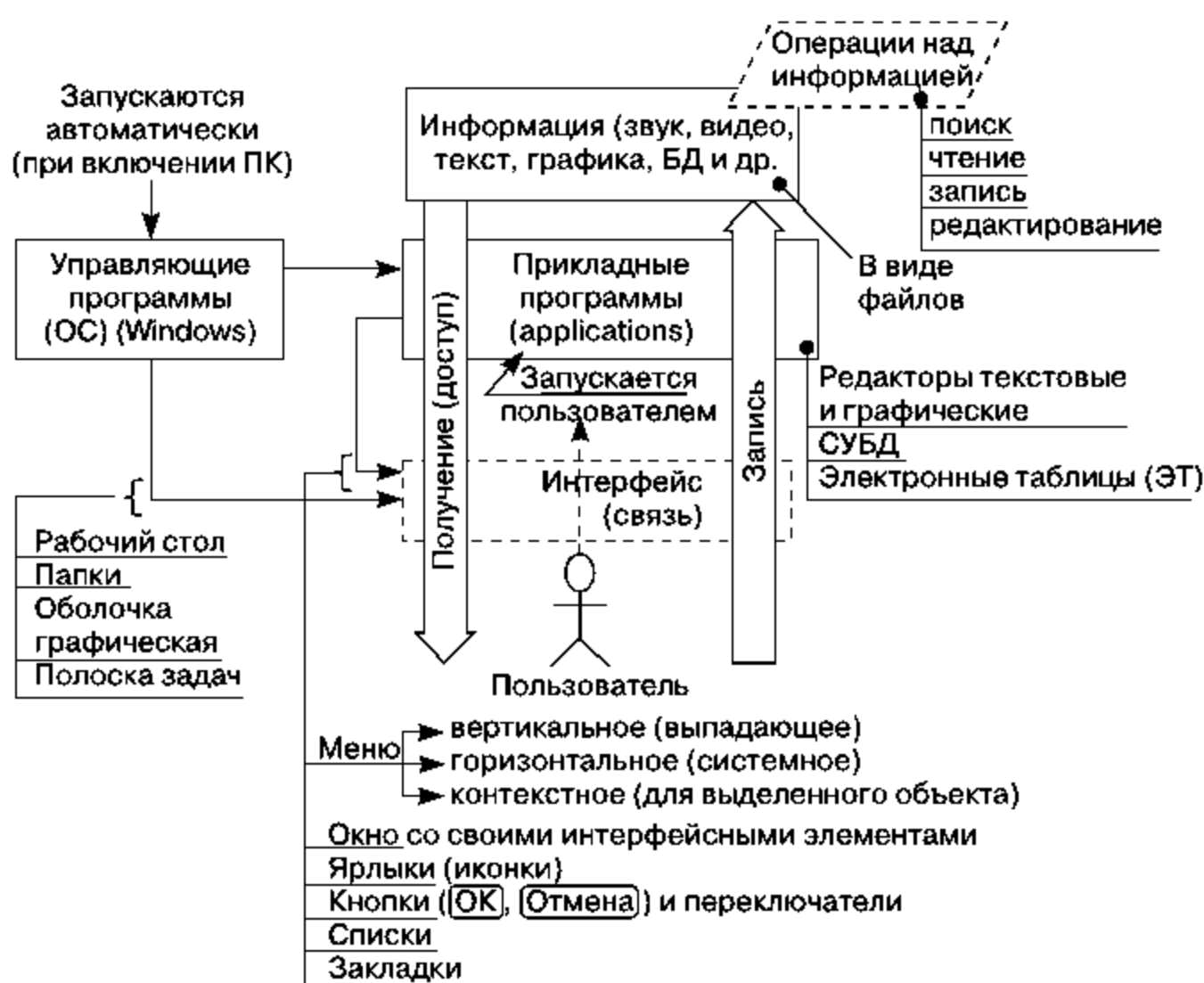


Рис. Л-1.1. Модель обработки информации на ПК

Из рисунка Л-1.1 видно, что пользователь не имеет непосредственного доступа к информации. Два блока разделяют их:

- ♦ прикладные программы (applications) — «исполнители» операций по обработке информации;
- ♦ интерфейс — средство связи пользователя с этими «исполнителями» и другими устройствами компьютера.

Реально пользователь может подержать в руках дискету или компакт-диск (CD) с имеющейся на них информацией, но не более. Найти необходимую информацию, например, какой-то текст, прочитать его, сделать «твердую» копию (распечатать на принтере) — все это может сделать только прикладная программа — *текстовый редактор*. Пользователь должен выбрать необходимую для соответствующих операций прикладную программу и, пользуясь элементами интерфейса, дать команду на ее запуск.

Работой прикладных программ управляет ОС: она загружает их в память, управляет исполнением и взаимодействием программ между

собой, связью с внешними устройствами, обеспечивает интерфейс с пользователем.

Таким образом, диалог с компьютером (доступ к информации, запись и хранение, запуск прикладных программ и др.) пользователь реализует через интерфейс, часть которого предоставляет ОС (рабочий стол, панели и др.), а часть — прикладные программы (меню, окна, ярлыки и др.).

1.5. Основные сведения об операционной системе

Каждый компьютер обязательно комплектуется операционной системой, для которой создается свой набор прикладных программ (приложений).

Операционная система (ОС) — это комплекс системных и служебных программ, осуществляющих управление ресурсами компьютера, запуск прикладных программ и их взаимодействие (интерфейс) с внешними устройствами и другими программами, а также обеспечивающих диалог пользователя с компьютером (интерфейс пользователя).

Ресурсом является любой компонент компьютера и предоставляемые им возможности — центральный процессор, оперативная и внешняя память, внешние устройства, программное обеспечение, которое на нем установлено, и др.

Исходя из выполняемых функций, ОС классифицируются:

- ♦ *по числу одновременно работающих пользователей: одно- и многопользовательские*; последние, в отличие от однопользовательских, поддерживают одновременную работу на ПК нескольких пользователей за различными терминалами;
- ♦ *числу программ, одновременно работающих под управлением системы: одно- и многозадачные*. Многозадачные поддерживают выполнение нескольких программ-задач, существующих в рамках одной вычислительной системы, в один момент времени и процессор распределяет ресурсы компьютера между задачами;
- ♦ *типу интерфейса: командные* (предполагают ввод пользователем команд с клавиатуры при выполнении действий по управлению ресурсами компьютера) и *объектно-ориентированные* (графические) — управление ресурсами вычислительной системы путем осуществления операций над объектами, пред-

ставляющими *файлы, каталоги (папки), дисководы, программы, документы* и т.д.;

- ♦ *типу использования ресурсов: локальные и сетевые.* Сетевые предназначены для управления ресурсами нескольких компьютеров, объединенных в сеть с целью совместного использования данных и других сетевых ресурсов.

Главными функциями ОС являются:

- ♦ создание файловой системы;
- ♦ обслуживание файловой системы;
- ♦ установка и запуск внешних программ;
- ♦ установка и поддержка оборудования;
- ♦ обеспечение интерфейса пользователя и др.

Наиболее распространенными ОС на сегодняшний день являются системы семейства *Windows, Unix* и *Linux*.

ОС *Windows 9x/ME/2000/XP* является многозадачной операционной системой с графическим интерфейсом и расширенными сетевыми возможностями.

ОС *Unix* — многозадачная операционная система, способная обеспечить одновременную работу очень большого числа пользователей. Это машинно-независимая ОС, что обеспечивает ее высокую мобильность и легкую переносимость прикладных программ на универсальные компьютеры, мини-ЭВМ и ПК различной архитектуры. Существуют десятки операционных систем, которые можно объединить под общим названием *Unix*.

Широко используют также достаточно мощные системы *Linux*. Свободное распространение и использование исходных текстов стали основой феномена *Linux*.

Операционная система управляет ресурсами ПК, выполнением программ, организует обмен информацией с внешними устройствами ПК, выполняет работы по хранению информации во внешней памяти и обслуживанию дисков.

Основными функциональными компонентами ОС являются:

- ♦ *файловая система:* совокупность средств ОС, обеспечивающих доступ к данным;
- ♦ *драйверы:* программы, управляющие работой внешних устройств;
- ♦ *командный процессор:* системная программа, обеспечивающая выполнение команд, вводимых пользователем с клавиатуры (или другим способом).

1.6. Операционная система Windows 2000/XP

ОБЩИЕ СВЕДЕНИЯ И ОСНОВНЫЕ ПОНЯТИЯ

Windows — это интегрированная среда, обеспечивающая эффективный обмен информацией между отдельными программами и предоставляющая пользователю широкие возможности обработки текстовой, графической, звуковой и видеoinформации. Интегрированность подразумевает также совместное использование ресурсов компьютера всеми программами. Эта операционная система обеспечивает работу пользователя в сети, предоставляя встроенные средства поддержки для обмена файлами и меры по их защите, возможность совместного использования принтеров, факсов и других общих ресурсов.

Большинство современных ПК в настоящее время работают под управлением ОС Windows 2000/XP, которые входят в линейку систем Windows NT и обеспечивают пользователям многозадачность, надежность, многопроцессорную поддержку, защиту данных, удобство работы в рамках локальной сети и многое другое. Это операционные системы нового поколения для делового использования на самых разнообразных компьютерах — от портативных до серверов.

Windows 2000/XP — 32-разрядная ОС с графическим пользовательским интерфейсом, поддерживающая мультимедийные средства и предназначенная для управления автономным или сетевым компьютером платформы IBM PC.

Многозадачность является функцией, реализующей параллельное выполнение нескольких задач. *Задача* (приложение) — это программа, находящаяся в состоянии выполнения.

Интерфейс Windows 2000/XP является объектно-ориентированным, ОС управляет ресурсами компьютера путем осуществления операций над *объектами*, которые условно можно разделить на три категории (рис. Л-1.2). Такая классификация является условной (например, главное меню и подчиненные ему являются папками), но в целом она отражает принципиальную разницу между поведением и обслуживанием трех категорий объектов ОС.

Графический интерфейс — это простое и удобное средство, обеспечивающее пользователю выполнение необходимых действий при работе с программами и данными путем манипулирования их наглядными графическими отображениями, значками, ярлыками и др. (рис. Л-1.3).



Рис. Л-1.2. Классификация объектов

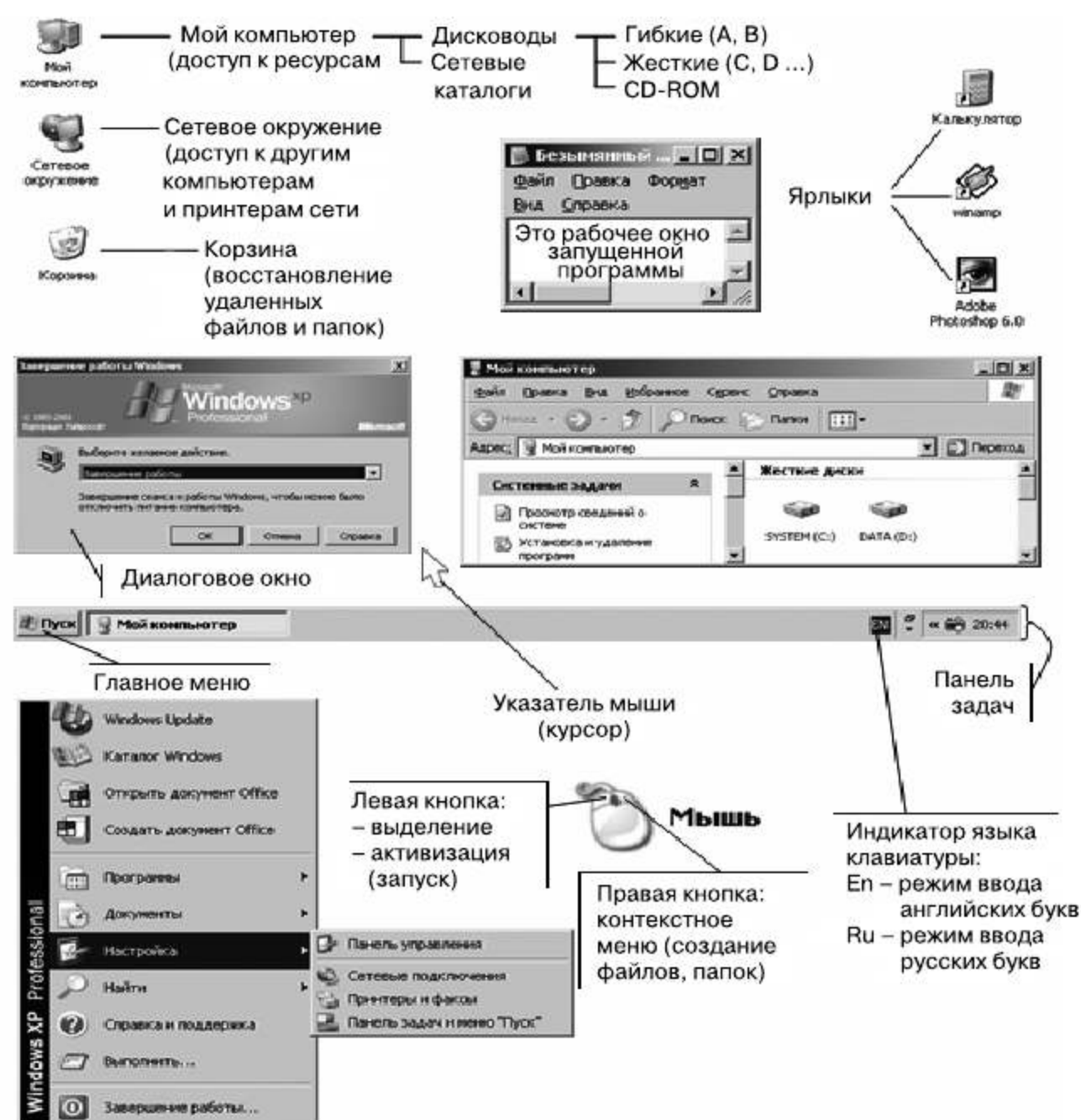


Рис. Л-1.3. Графические образы объектов

Основу интерфейса составляют *окна* — область экрана, отведенная для приложения, документа или диалога с пользователем. На рис. Л-1.4 приведено окно папки *Мой компьютер*. Для любого приложения и его документов предусматриваются отдельные окна — *окна программ* и *окна*

документов (их может быть несколько, если программа может обрабатывать сразу несколько объектов (как Word или Excel)). *Документ* — это любой объект обработки прикладной программы (фактически это может быть, например электронная таблица). Документ может быть загружен в память машины для обработки (фактически загружается соответствующий файл). Документ может быть сохранен на диске в виде соответствующего файла. *Окно программы* (приложения) содержит собственное *меню* и *панели инструментов*, настраиваемые пользователем.

После загрузки системы на экране монитора появляется изображение, похожее на то, что приведено на рис. Л-1.4.



Рис. Л-1.4. Рабочий стол с окном папки *Мой компьютер*

Изображение (рис. Л-1.4) состоит из двух частей: *рабочего стола* (своеобразный «фасад», «титulusный лист» системы) с ярлыками на нем (*Мой компьютер*, *Сетевое окружение*, *Корзина*) и *панели задач* с кнопкой *Пуск*, расположенной в нижней части экрана.

Рабочий стол — наивысший объект в иерархии объектов оболочки Windows 2000/XP (рис. Л-1.5). На рабочем столе могут размещаться различные значки, но два из них — *Мой компьютер* и *Корзина* — находятся всегда.



Рис. Л-1.5. Основная иерархия папок в Windows 2000/XP

Мой компьютер позволяет раскрыть папку, содержащую объекты, соответствующие всем устройствам компьютера.

Корзина предназначена для того, чтобы выбрасывать в нее ненужные файлы, удаленные средствами оболочки. Файлы из корзины можно вернуть.

Папку (и значок) *Мой компьютер* можно переименовать, но их нельзя удалить с рабочего стола, *Корзину* нельзя ни переименовать, ни удалить.

Каждый логический диск A:, C:, D: и т.д. является вершиной дерева обычных папок: диск (корневой каталог) содержит файлы и папки (каталоги) 1-го уровня, папки 1-го уровня содержат папки 2-го уровня и т.д. На уровне пользователя Windows 2000/XP поддерживают следующую терминологию:

- ♦ *программы* (приложения) — файлы с расширением .EXE или .COM.
- ♦ *документ* — файл, созданный с помощью соответствующей программы и логически *связанный* с ней (текстовый документ связан с текстовым редактором (MS Word), графический файл связан с графическим редактором (Paint), работающим с изображениями);
- ♦ *папка* (каталог) — некий контейнер для хранения других папок, программ, документов и ярлыков. Папка может быть и пустой. Содержимое «контейнера» отображается значками в *окне папки*. Папки бывают двух видов: *обычные* (просто каталог с файлами) и *специальные* (*Мой компьютер*, *Принтеры* и др.);
- ♦ *тип документа* — расширение файла (документа).

1.7. Интерфейсные элементы Windows:

меню, окна, ярлыки, курсор мыши, списки, переключатели

Меню

Меню — объект на экране, отображающий список возможных действий и позволяющий выбрать необходимое. Состоит из отдельных пунктов, перемещение по которым и выбор нужного осуществляются

с помощью мыши (рис. Л-1.6). Меню бывают *вложенными* (многоуровневыми), когда при выборе пункта появляется еще одно меню и т.д.



Рис. Л-1.6. Классификация меню

Контекстное меню вызывается для конкретного объекта (*Документ, Папка, Программа*), который был выделен с помощью мыши ранее, и предназначено для упрощения работы с данным объектом. Его пункты соответствуют основным действиям над объектом.

Главное меню — один из основных системных элементов управления Windows 2000/XP, которое открывается (всплывает) щелчком по кнопке *Пуск*. С помощью него можно: запустить все программы, установленные под управлением ОС или зарегистрированные в ней, открыть последние документы, с которыми выполнялась работа, получить доступ ко всем средствам настройки ОС, а также доступ к поисковой и справочной системам Windows 2000/XP. Рассмотрим назначение пунктов главного меню (см. рис. Л-1.3).

Программы — представляет собой иерархическое меню доступных прикладных и служебных программ.

Документы — вызывает список последних 15 документов, открывавшихся пользователем средствами оболочки Windows 2000/XP.

Настройка — содержит команды вызова средств настройки: *Панели управления, Принтеров и Панели задач*. Последняя команда позволяет изменять вид *Панели задач*, редактировать, удалять и добавлять пункты и команды меню *Программы*, а также добавлять команды в *Главное меню*.

Найти — вызывает программу поиска требуемых файлов и папок, как на данном компьютере, так и на других компьютерах локальной сети.

Справка — служит для вызова справочной системы, предоставляющей пользователю сведения о возможностях Windows 2000/XP, порядке и правилах работы с ней.

Выполнить — предназначена для загрузки приложений. Диалоговое окно загрузки содержит кнопку *Обзор*, обеспечивающую поиск файлов.

Завершение работы — вызывает диалоговое окно завершения работы с Windows.

При выборе позиции *Пуск* программы меню и контекстного меню необходимо выполнить действия:

- ◆ активизировать меню — щелкнуть левой кнопкой мыши (для контекстного меню — правой);
- ◆ курсор мыши навести на нужный пункт;
- ◆ щелкнуть левой кнопкой (нажать и отпустить);
- ◆ для выхода из главного меню надо нажать кнопку [Esc].

Окна

Различают три типа окон (рис. Л-1.7):

- ◆ окно программы (приложения);
- ◆ окно документа (окно объекта обработки программы);
- ◆ диалоговое окно.

} Рабочие окна

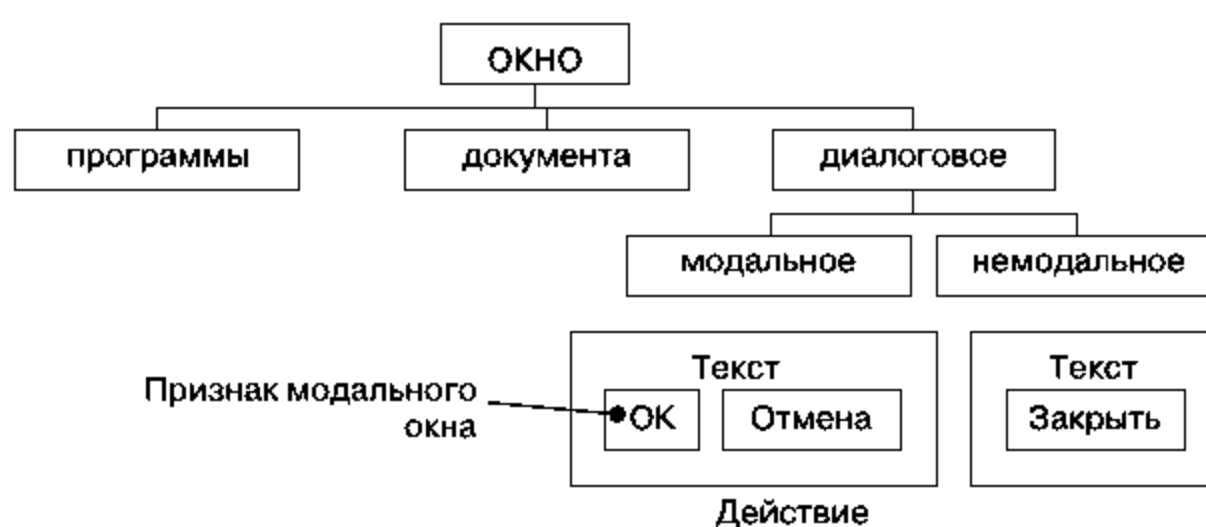


Рис. Л-1.7. Типы окон

Рабочее окно — это представитель программы. Именно в этом окне происходит обмен информацией между пользователем и программой.

Его стандартные элементы:

- ◆ значок системного меню;
- ◆ заголовок окна;
- ◆ строка меню;
- ◆ кнопки (свернуть, развернуть, восстановить);
- ◆ строка состояния и рабочее поле окна.

Двойной щелчок мыши, например, по значку *Мой компьютер*, открывает окно этой папки (рис. Л-1.8).

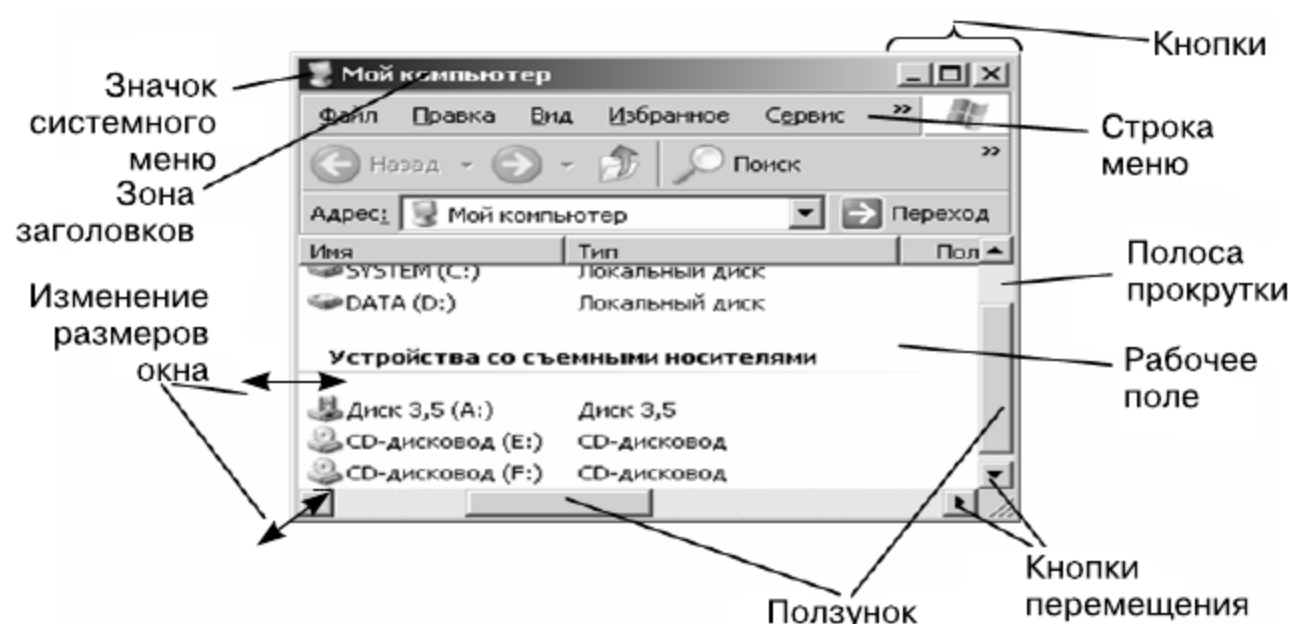


Рис. Л-1.8. Окно папки *Мой компьютер*

Окно приложения и окно документа имеют три варианта представления на экране:

- ♦ *полноэкранный* — окно развернуто и занимает весь экран (рабочий стол);
- ♦ *нормальный* — окно занимает часть экрана;
- ♦ *значок* (пиктограмма) — окно свернуто в пиктограмму.

Диалоговое окно не меняет своих размеров и имеет только *нормальный* вариант представления, т.е. занимает часть экрана.

Полосы прокрутки используются для просмотра информации, которая не поместилась в окне. Это вертикальные или горизонтальные полосы, располагающиеся справа и внизу окна соответственно. По ним передвигается так называемый ползунок, который показывает положение выводимой информации относительно всего текста. Полосы прокрутки имеют два элемента управления (см. рис. Л-1.8):

- ♦ *кнопку перемещения* (информацию можно плавно передвигать в окне);
- ♦ *ползунок* (позволяет быстро перемещать информацию в окне) — показывает нахождение текущего фрагмента по отношению ко всему объему информации.

Ярлыки

Ярлык представляет собой указатель на некоторый программный объект и содержит полный путь доступа к нему. Пользователь может создать ярлык для любого объекта Windows (файла, папки, диска и т.д.) и поместить его на Рабочем столе. Ярлыки существенно ускоряют и облегчают доступ к часто используемым программам и документам. Что-

бы обратиться к объекту, достаточно дважды щелкнуть левой кнопкой мыши по его ярлыку. Воздействуя мышью на ярлык, можно выполнить одно из трех действий с соответствующим ярлыку объектом (рис. Л-1.9).



Рис. Л-1.9. Способы воздействия мышью на ярлык и инициализируемые при этом операции

Мышь

Мышь является основным средством работы в среде Windows.

Пользователь может выполнять *щелчок* левой или правой кнопками либо *двойной щелчок* левой кнопкой, либо *указывать* на соответствующий объект путем установки на него курсора мыши. Большинство современных манипуляторов «мышь» имеют колесико, дублирующее функции вертикальной полосы прокрутки.

Щелчок — кратковременное нажатие на кнопку мыши. Служит для ввода команд, выбора пункта меню, выделения объекта. При щелчке *важно*, чтобы в момент нажатия на кнопку корпус мыши оставался неподвижным.

Двойной щелчок — быстрое нажатие 2 раза на одну и ту же кнопку мыши. Используется, например, для просмотра содержимого дисков, устройств, папок.

Указание — установка курсора мыши. При установке курсора мыши на элементы некоторых меню раскрывается меню следующего уровня, всплывающее меню, подсказки.

Перетаскивание — перемещение указателя мыши при нажатой кнопке. Приводит к переносу выбранного объекта в нужное место экрана.

1.8. Практическая работа на ПК

Практическая работа включает в себя три части:

- ◆ операции с мышью;
- ◆ запуск и закрытие программ;
- ◆ операции с окнами.

В конце выполнения практической части необходимо корректно завершить работу с ОС Windows.

ЧАСТЬ 1. ОПЕРАЦИИ С МЫШЬЮ

Выполняя щелчки левой либо правой кнопкой мыши, пользователь инициирует выполнение следующих операций.

1. *Выделение объекта.* При этом необходимо *указать* на объект и щелкнуть левой кнопкой мыши. Цвет, например, значка и его название изменятся по сравнению с другими значками. С помощью операции *выделения* пользователь указывает операционной системе объект, над которым он предполагает выполнить очередную операцию.
2. *Отмена выделения.* Курсор необходимо переместить за пределы выделенного объекта, например значка, на *свободное место Рабочего стола* и выполнить щелчок. В результате отменяется (снимается) операция выделения, и цвет значка восстанавливается.
3. *Перетаскивание* — перемещение указателя мыши при нажатой кнопке. Приводит к перемещению выбранного объекта в нужное место экрана. Операция может быть использована для изменения положения на рабочем столе окна, значка, ярлыка и других объектов.
4. *Управление кнопками.* Кнопки являются одним из самых распространенных интерфейсных элементов Windows и присутствуют во всех окнах, на панели задач и др. Основная операция, выполняемая с кнопкой, — щелчок по ней. Например, для *закрытия* окна *Мой компьютер* необходимо найти в правом верхнем углу окна значок закрытия окна (кнопка ) и щелкнуть по левой кнопке мыши, установив курсор на значке . В результате окно исчезает с экрана.
5. *Открытие контекстного меню.* При щелчке *правой* клавишей мыши по значку, например *Мой компьютер* либо *Корзина*, на экране отобразится контекстное меню *соответствующего* объекта. Щелчком *правой* кнопки мыши на свободную область рабочего стола открывается контекстное меню настроек *Рабочего стола*.
6. *Закрытие контекстного меню.* Закрыть контекстное меню можно, щелкнув левой кнопкой мыши за его пределами.
7. *Доступ к основным вычислительным ресурсам компьютера.* При двойном щелчке левой клавишей мыши по значку *Мой компьютер* на экране открывается окно с тем же названием, со-

держашее значки, расположенные во внутренней области окна: дисковых устройств и папок. *Значки* имеют различное графическое отображение и представляют собой центральные аппаратные и программные элементы ПК, т.е. *основные вычислительные ресурсы компьютера*.

Задание 1

1. *Выделить и отменить выделение значка Мой компьютер* и повторить задание еще для 2—3 значков рабочего стола.
2. Открыть доступ к основным вычислительным ресурсам ПК (см. выше п. 7).
3. *Управляя кнопками* окна *Мой компьютер* (в правом углу зоны заголовка), инициировать выполнение предписанных им операций.
4. Изменить положение значка *Мой компьютер* на экране, *перетаскив* его на свободное место, и повторить задание для двух других значков.
5. *Открыть и закрыть контекстные меню* для объектов: *Мой компьютер, Корзина, Рабочий стол*. Виды меню занести в отчет.
6. Создать два ярлыка для *Корзины*, для чего необходимо:
 - ◆ щелкнуть правой кнопкой мыши на значке *Корзина*;
 - ◆ в открывшемся контекстном меню выбрать пункт *Создать ярлык*или:
 - ◆ нажать правую кнопку мыши и, удерживая ее, переместить курсор мыши на свободную область *Рабочего стола*;
 - ◆ отпустить кнопку мыши;
 - ◆ в открывшемся контекстном меню выбрать пункт *Создать ярлык*.

ЧАСТЬ 2. ОПЕРАЦИИ С ПРОГРАММАМИ (запуск, работа, закрытие)

Для *запуска* программы (приложения) необходимо открыть окно, которое ее представляет и позволяет пользователю вести диалог с программой, т.е. обмениваться информацией.

1. *Запуск программы*:
 - а) левой кнопкой мыши щелкнуть кнопку *Пуск*, расположенную на панели задач рабочего стола;
 - б) в открывшемся *Главном меню* курсором мыши указать на строку с названием группы *Программы* и задержать курсор на 2—3 с, после чего всплывает подменю;

- в) в открывшемся подменю появятся названия подгрупп, входящих в состав группы *Программы*. Перемещая курсор мыши сначала *строго вдоль строки*, на которую он был установлен, а затем в вертикальном направлении вверх и вниз, указать на нужную строку, например группу *Стандартные*, и открыть ее, указав курсором и задержав его на 2—3 с;
 - г) перемещаясь в открывшемся меню по правилам, предписанным в пункте (в), найти нужную программу, например *Блокнот*, и запустить ее, щелкнув левой клавишей мыши. На экране появится окно программы редактора *Блокнот*, представляющее собой выбранную программу.
2. *Работа с программой.*
Окно содержит собственное меню и панели инструментов, настраиваемые пользователем и позволяющие работать с программой. Следует набрать 2—3 строки содержательного текста и отредактировать их.
3. *Закрытие программы.*
Выполнение программы можно завершить, закрыв ее окно. Так, для закрытия программы *Блокнот* необходимо щелкнуть левой кнопкой мыши на кнопке  в правом углу заголовка окна *Блокнот*.

Задание 2

Повторить действия по открытию и закрытию окна программы, запустив еще 2—3 программы, например, *Калькулятор*, *WordPad*, *Paint*.

ЧАСТЬ 3. ОПЕРАЦИИ С ОКНАМИ

(открытие, закрытие, свертывание, восстановление, изменение размеров, переключение на другое окно)

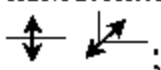
Рабочие окна (приложения и документы) имеют три варианта представления на экране (полноэкранный, нормальный, значок или пиктограмма).

Для изменения варианта представления окна применяются:

- а) кнопки в зоне заголовков окна  .
- б) команды системного меню, дублирующие функции кнопок зоны заголовков (значок системного меню находится в левой части зоны заголовка, см. рис. Л-1.8).

Действия с окном программы:

- ♦ *свернуть* в пиктограмму: кнопка .
- ♦ *развернуть* до полноэкранного: кнопка .
- ♦ *восстановить* нормальный размер: кнопка .
- ♦ *закрыть*: кнопка .

- ♦ *изменить границы* (если окно занимает не весь экран):
;
- ♦ *переключиться на другое окно* (с помощью *Панели задач* или выделения на *Рабочем столе* требуемого окна).

ОТКРЫТИЕ ОКНА

Поскольку *окно* является представителем программы, то для его открытия необходимо запустить соответствующую программу. Так, для открытия окна программы *Блокнот* необходимо запустить программу *Блокнот*. Для изменения варианта представления окна используют кнопки в правом углу его заголовка (см. часть 3, п. а).

ИЗМЕНЕНИЕ РАЗМЕРОВ ОКНА

Для изменения размеров окна *Блокнота* необходимо поместить указатель мыши на границу окна, чтобы он принял форму *раздвоенной* стрелки; далее, удерживая левую кнопку мыши, передвинуть границу окна в любую сторону и отпустить кнопку мыши. Для изменения размеров окна по диагонали необходимо «*потянуть*» курсором *угол* окна.

ЗАКРЫТИЕ ОКНА

Чтобы закрыть окно, используя меню приложения, имеющее значок в левом верхнем углу окна, необходимо щелкнуть по этому значку и в открывшемся меню выбрать команду *Закрыть*.

Задание 3

Повторить действия по открытию, изменению размеров и закрытию окна, запустив еще 2—3 программы, например *Калькулятор*, *WordPad*, *Paint*.

ЗАВЕРШЕНИЕ РАБОТЫ С СИСТЕМОЙ

ВНИМАНИЕ! Для правильной работы системы выполнение нижеперечисленных действий перед каждым выключением компьютера обязательно!

Выбрать в *Главном меню* пункт *Завершение работы*. В появившемся диалоговом окне выбрать пункт *Выключить компьютер* и нажать кнопку *Да*. После этого необходимо дождаться сообщения «*Теперь питание компьютера можно отключить*» и только тогда нажать кнопку выключения компьютера.

Контрольные вопросы

1. Что такое операционная система? Перечислите ее функции.
2. Классификация операционных систем.
3. Общая характеристика и особенности ОС Windows 2000/XP.

4. Модель обработки информации на компьютере. Место ОС.
5. Основные термины, используемые в Windows 2000/XP.
6. Понятие объекта. Классификация объектов.
7. Перечислите основные папки Windows 2000/XP.
8. Понятие документа. Как открыть документ? Поиск документов.
9. Окна в Windows 2000/XP. Элементы окна.
10. Операции с окнами.
11. Как открыть меню *Справка*? Правила использования справки.
12. Назначение и возможности папки *Мой компьютер*.
13. Назначение и возможности кнопки *Пуск*.
14. Просмотр «содержимого компьютера».
15. Порядок действий при запуске прикладных программ.
16. Мышь и основные элементы работы с ней.
17. Что такое ярлык? Отношение ярлык—объект.
18. Порядок действий при завершении работы с системой.

Лабораторная работа 2

РАБОТА В WINDOWS С ФАЙЛАМИ И КАТАЛОГАМИ В РЕЖИМЕ КОМАНДНОЙ СТРОКИ

2.1. Цель работы

1. Изучение основных операций и команд *режима командной строки*.
2. Отработка практических навыков использования команд для работы на ПК.

2.2. Порядок выполнения работы

1. Изучение методических указаний (дома). Дополнительная литература [1, 7—9].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (цель работы, технология работы в режиме командной строки на ПК под управлением ОС Windows, основные операции с каталогами и файлами, их смысл, порядок действий при выполнении операций).
4. Работа на компьютере:
 - ◆ регистрация;
 - ◆ выполнение операций с каталогами (см. подраздел 2.6);
 - ◆ создание дерева каталогов в соответствии с заданным вариантом (табл. Л-2.1, номер варианта соответствует номеру бригады);
 - ◆ выполнение операций с файлами (см. подраздел 2.6).

5. Отчет о проделанной работе:

- ◆ индивидуальное выполнение на ПК операций по заданию преподавателя (смена текущего каталога, просмотр каталога, переименование файла, удаление файла и др.);
- ◆ оформление отчета (дорабатывается по результатам практической работы на ПК);
- ◆ защита отчета.

2.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ понятия: файл, каталог, дерево каталогов и др.;
- ◆ основные команды *режима командной строки* ОС Windows.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ подробный протокол выполнения всех операций с каталогами и файлами. Особенности выполнения операций;
- ◆ создание дерева каталогов (описание последовательности выполняемых операций).

2.4. Организация данных. Основные понятия

Для хранения программ и данных в ПК используются дисковые накопители: на *гибких магнитных дисках* (НГМД — дискеты емкостью 1,44 МБ флоппи-диски), компакт-диски (compact-disk), или просто *CD-диски* (рис. Л-2.1) с оптическим способом записи информации, имеющие по сравнению с дискетами достаточно большую емкость (650—800 МБ), DVD-диски емкостью от 4,7 до 24 ГБ и так называемые *жесткие диски* (ЖД) с емкостью в сотни гигабайт. Все дисковое пространство (на гибких и жестких магнитных дисках) делится на логические диски, имена которых обозначаются первыми буквами латинского алфавита (C:, D: — для ссылок к накопителям на ЖД и A:, B: — к накопителям на гибком диске).



Рис. Л-2.1. Компакт-диск

Все современные компьютеры имеют привод CD-ROM или DVD-ROM. Самый распространенный привод такого рода использует выдвижной лоток, на который кладется сам диск. На передней панели такого привода расположены кнопка *Eject* для загрузки/выгрузки диска, индикатор обращения к приводу и гнездо для подключения наушников. Главной функцией стандартного привода CD(DVD)-ROM является считывание с помощью лазерного луча информации, записанной на диске. Новые модели приводов также позволяют пользователю с помощью специальных программ записывать информацию на оптические компакт-диски.

На сегодняшний день оптические (лазерные) диски — недорогой массово воспроизводимый, достаточно надежный и долговечный материальный носитель информации, записанный цифровым способом, что делает его незаменимым для сохранения и распространения программ (установочных пакетов), мультимедийных программ, а также для долговременного хранения больших объемов информации, например баз данных.

Наряду с оптическими дисками широко используются накопители информации с электронным способом записи (USB Flash drive), которые устанавливаются в USB-порт компьютера. Они имеют емкость до нескольких гигабайтов при крайне миниатюрных размерах и отсутствии движущихся деталей. Операционная система автоматически распознает его как дополнительный дисковый накопитель и показывает в папке *Мой компьютер* пиктограммой *Съемный диск*. В дальнейшем пользователь работает с этим диском как с жестким. При удалении флэш-диска пиктограмма *Съемный диск* исчезает.

ФАЙЛЫ

Файл является основным носителем информации в Windows и представляет собой определенную физическую область на диске, имеющую некоторые атрибуты (свойства) и помеченную именем. Содержанием файла может быть готовая программа, текстовый документ, рисунок или фотография, таблица базы данных, звукозапись, видеоклип и т.д., т.е. файл — это логически связанная совокупность данных или программ.

Название файла состоит из двух частей:

<имя>.<расширение>

Расширение (.exe, .txt и др.) характеризует тип файла (выполняемые программы, текст и др.), например F1.TXT — текстовый файл с именем F1. *Значки*, используемые для отображения файлов в окнах

приложений, определяются типом файла. Для удобства работы с файлами на диске создаются каталоги.

КАТАЛОГИ

Имена файлов регистрируются в каталогах (или директориях).

Каталог — это специальное место на диске, в котором хранятся имена файлов и другие сведения о них (размер, время последнего обновления, атрибуты и т.д.), т.е. *каталог* — это таблица оглавлений файлов.

Все каталоги (кроме корневого) фактически являются файлами специального вида. Каждый каталог, как и файл, имеет имя, но, как правило, без расширения, и он может быть зарегистрирован в другом каталоге. Например, если каталог X зарегистрирован в каталоге Y, то говорят, что X — *подкаталог* Y, а Y — *надкаталог* или *родительский каталог* для X.

Каталог, с которым в настоящий момент работает пользователь, называется *текущим*.

На любом диске всегда существует *главный*, или *корневой*, каталог, который обозначается как <имя_диска>:\ (например, C:\). В нем регистрируются файлы и подкаталоги (каталоги первого уровня). В каталогах первого уровня регистрируются файлы и подкаталоги второго уровня и т.д.

Файловая система Windows предоставляет возможность создавать на диске иерархическую структуру каталогов, называемую *деревом каталогов*, подобную перевернутому дереву, в котором главный каталог образует корень дерева, а остальные каталоги подобны ветвям. В этой системе (структуре) элементами каталога могут быть не только обычные файлы, но и другие подчиненные каталоги. Если используется файл не из текущего каталога, необходимо указать, в каком каталоге этот файл находится, указав *путь к файлу* — последовательность из имени диска и имен каталогов, разделенных символом «\». После имени диска ставится двоеточие.

Таким образом, для точной идентификации каталога (файла) в общем случае необходимо указать его полный адрес (имя) в файловой системе. Например, D:\USERS\840 — полный адрес каталога 840. То есть каталог 840 находится в каталоге USERS на диске D:, являясь его подкаталогом.

К функциям обслуживания файловой системы относятся такие операции с каталогами и файлами, как:

- ◆ *создание и именованье* (присвоение им имен);
- ◆ *переименование*;
- ◆ *удаление* и др.

2.5. Режим командной строки ОС Windows 2000/XP

ОБЩИЕ СВЕДЕНИЯ

Первые ПК работали под управлением дисковой операционной системы MS-DOS (DOS — Disk Operating System) — однопользовательской, однозадачной ОС, обеспечивающей взаимодействие пользователя с компьютером *с помощью команд, вводимых пользователем (интерфейс командной строки)*.

Следует заметить, что ОС Windows унаследовала от MS-DOS возможность работы в *режиме командной строки*, который используется при возникновении внештатных ситуаций, таких как:

- ♦ удален файл, необходимый для загрузки и работы ОС;
- ♦ система не загружается по непонятным причинам и др.

При работе в режиме командной строки вводится понятие *главного*, или *корневого*, каталога, который обозначается символом «\» (обратный слеш). В нем регистрируются файлы и подкаталоги (каталоги 1-го уровня). В каталогах 1-го уровня регистрируются файлы и подкаталоги 2-го уровня и т.д.

Если используется файл не из текущего каталога, необходимо показать, в каком каталоге этот файл находится, указав *путь к файлу*.

Для точной идентификации каталога (файла) в режиме командной строки в общем случае необходимо указать его полный адрес (имя) в файловой системе. Например, D:\USERS\847 — полный адрес каталога 847.

Из приведенного примера следует, что каталог 847 находится в каталоге USERS на диске D:, являясь его подкаталогом.

РАБОТА В РЕЖИМЕ КОМАНДНОЙ СТРОКИ

Работа в режиме командной строки под управлением ОС Windows выполняется следующим образом.

Пользователь вводит команду. Реакция системы на введенную команду отображается на экране либо в виде предложения о том, что делать дальше, либо в виде запроса о следующих действиях, либо в виде результата выполненной по команде операции и готовности системы воспринимать следующую команду.

ВВОД КОМАНДЫ

Для ввода команды необходимо набрать ее на клавиатуре и нажать клавишу [Enter]. Для редактирования вводимой команды можно пользоваться следующими клавишами:

- [Backspace] или [←] — стирание предыдущего символа;
- [Esc] — удаление всей команды;
- [F3] — повторение предыдущей команды.

2.6. Практическая работа на ПК

РЕГИСТРАЦИЯ

- ◆ Щелкнуть мышью по кнопке *Пуск*;
- ◆ в открывшемся меню выбрать пункт *Выполнить...*;
- ◆ в появившемся окне ввести `cmd` и нажать [Enter];
- ◆ в открывшемся окне ввести `d:` и нажать [Enter];
- ◆ ввести `cd users\номер группы`, например `840`, и нажать [Enter].

Реакцией ОС на выполнение действия является появление в окне командной строки приглашения к работе

`D:\USERS\840>`,

после которого с помощью клавиатуры можно набирать команды режима командной строки.

Примечание. Все предписанные операции выполняются *только* в своем каталоге, в данном случае в каталоге `840`.

ОЧИСТКА ЭКРАНА

Для очистки экрана монитора от имеющейся на нем информации используется команда `cls`. Наберите ее и нажмите клавишу [Enter]. После исполнения команды на экране остается только приглашение системы.

РАБОТА С КАТАЛОГАМИ

Основные команды

<code>md (mkdir)</code>	— создание каталога;
<code>dir</code>	— просмотр каталога;
<code>cd (chdir)</code>	— смена текущего каталога;
<code>rd (rmdir)</code>	— удаление каталога.

Создание каталога

Для создания нового каталога используется команда `mkdir` (сокращенно `md`). Ее формат:

`md <дисковод:\> <имя нового каталога> —`

создание нового каталога в корневом каталоге выбранного диска.

Создайте в вашем рабочем каталоге `D:\USERS\840` каталог `UNIVERSITY`, а затем вложите в него еще два каталога: `FACULTY\IVANOV`.

В дальнейшем вместо имени каталога `IVANOV` целесообразно использовать свою фамилию, записав ее латинскими буквами, и считать ее именем каталога. Допишите имеющуюся строку на экране монитора

командой `md` и в конце каждой строки нажмите [Enter]. На экране монитора это выглядит следующим образом:

```
D:\USERS\840> md UNIVERSITY
D:\USERS\840> md UNIVERSITY\FACULTY
D:\USERS\840> md UNIVERSITY\FACULTY\IVANOV
D:\USERS\840> _.
```

В результате выполнения команды будет создана структура каталогов, показанная на рис. Л-2.2.

```
D:\USERS\840\ — UNIVERSITY
                  |
                  |— FACULTY
                        |
                        |— IVANOV
```

Рис. Л-2.2. Структура каталогов

Просмотр каталога

Для вывода содержимого каталога предназначена команда `dir`, имеющая следующий формат: `dir <имя диска> <имя каталога>`.

Просмотрите содержимое каталогов `UNIVERSITY` и `UNIVERSITY\FACULTY`.

Просмотр каталога `UNIVERSITY` реализуется командой `dir UNIVERSITY`

```
D:\USERS\840>dir UNIVERSITY
```

В результате ее выполнения на экране отображается информация:

```
Volume in drive D is WORK
Volume Serial Number is 1B29-8EF1
Directory of D:\USERS\840\
.                <DIR>  9-10-1994 2:30p
..               <DIR>  9-10-1994 3:01p
FACULTY          <DIR>  9-10-1994 3:22p
                23 file(s)      0 bytes
                21837632 bytes free
D:\USERS\840> _.
```

Просмотр каталога `UNIVERSITY\FACULTY` реализуется с помощью команды `dir UNIVERSITY\FACULTY`

```
D:\USERS\840>dir UNIVERSITY\FACULTY
D:\USERS\840> _.
```

Смена текущего каталога

Для изменения текущего каталога используется команда `chdir` (сокращенно `cd`). Ее формат: `cd <имя диска> <имя каталога>`.

Перейдите в каталоги UNIVERSITY и UNIVERSITY\FACULTY.

Переход в каталог UNIVERSITY осуществляется командой cd UNIVERSITY

```
D:\USERS\840> cd UNIVERSITY
D:\USERS\840\UNIVERSITY>_
```

Переход в каталог FACULTY осуществляется командой cd FACULTY

```
D:\USERS\840\UNIVERSITY> cd FACULTY
D:\USERS\840\UNIVERSITY\FACULTY >_
```

В результате произведенных действий текущим будет каталог FACULTY.

Для перехода в надкаталог используется команда cd .., для перехода в корневой каталог — cd\. Перейдите последовательно в надкаталог и корневой каталог.

Удаление каталога

Удалить можно *только пустой* каталог, т.е. такой, в котором нет подкаталогов и файлов. Для удаления пустого каталога существует команда *rmdir* (сокращенно *rd*). Ее формат: *rd* <имя пустого подкаталога>.

Задание 1

Удалить каталоги \UNIVERSITY\FACULTY\IVANOV.

Удалим каталог IVANOV с помощью команды *rd* IVANOV

```
D:\USERS\840\UNIVERSITY\FACULTY>_rd IVANOV
D:\USERS\840\UNIVERSITY\FACULTY>_
```

Просмотрим результат выполнения команды, введя команду *dir*:

```
D:\USERS\840\UNIVERSITY\FACULTY>dir
Volume in drive D is WORK
Volume Serial Number is 1B29-8EF1
Directory of D:\USERS\840\UNIVERSITY\FACULTY
.                <DIR>  9-10-1994 2:30p
..               <DIR>  9-10-1994 3:01p
                22 file(s)      0 bytes
                21837632 bytes free
D:\USERS\840\UNIVERSITY\FACULTY>_
```

Удалим каталог FACULTY. Перед тем как вводить команду удаления *rd* FACULTY, необходимо перейти в надкаталог UNIVERSITY с помощью команды *cd ..*

```
D:\USERS\840\UNIVERSITY\FACULTY>cd ..
D:\USERS\840\UNIVERSITY>rd FACULTY
D:\USERS\840\UNIVERSITY>_
```

Удалим каталог UNIVERSITY. Перед удалением перейдем в корневой каталог, выполнив команду cd .. Затем воспользуемся командой удаления rd UNIVERSITY

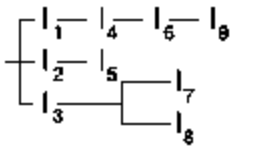
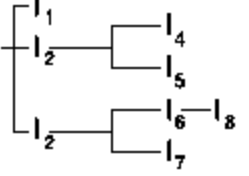
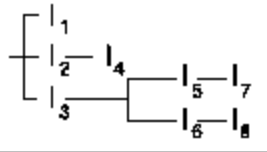
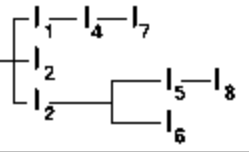
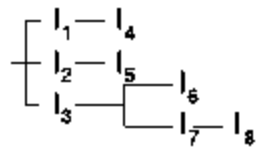
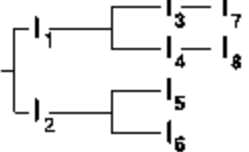
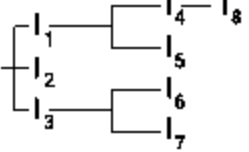
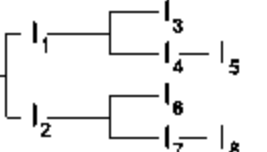
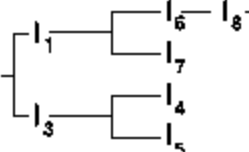
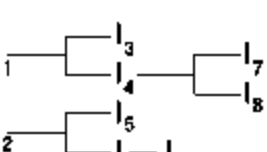
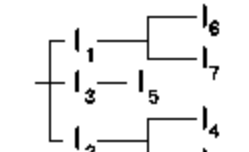
```
D:\USERS\840\UNIVERSITY>cd ..
D:\USERS\840>rd UNIVERSITY
D:\USERS\840>_
```

Создание дерева каталогов

Создать дерево каталогов в соответствии с заданным вариантом (см. табл. Л-2.1). Подробный протокол действий, выполняемых при создании дерева, занести в отчет.

Таблица Л-2.1

Дерево каталогов (варианты заданий)

Вариант	Задание	Вариант	Задание
1	D:\USERS* 	7	D:\USERS* 
2	D:\USERS* 	8	D:\USERS* 
3	D:\USERS* 	9	D:\USERS* 
4	D:\USERS* 	10	D:\USERS* 
5	D:\USERS* 	11	D:\USERS* 
6	D:\USERS* 	12	D:\USERS* 

* Поставить номер своей группы.

РАБОТА С ФАЙЛАМИ**Основные команды**

<code>copy</code>	— копирование файла (файлов);
<code>copy con <файл></code>	— создание текстового файла;
<code>copy <файл> prn</code>	— печать файла;
<code>type</code>	— просмотр файла;
<code>ren</code>	— переименование файла;
<code>del</code>	— удаление файла.

СОЗДАНИЕ ТЕКСТОВОГО ФАЙЛА (В ТЕКУЩЕМ КАТАЛОГЕ)

Небольшой текстовый файл достаточно просто ввести непосредственно с клавиатуры построчно. В конце каждой строки надо нажимать клавишу [Enter], а после ввода последней строки — нажать клавишу [F6] и затем [Enter].

Создайте текстовый файл `text.txt` командой `copy con text.txt`, содержащей следующий текст:

Мы, студенты гр. ..., поступили на ФВТ РГРТУ в ... году.

На экране монитора это выглядит следующим образом:

```
D:\USERS\840> copy con text.txt
Мы, студенты гр. ..., поступили на ФВТ РГРТУ в ... году.
^Z
1 file(s) copied
D:\USERS\840>_
```

Убедитесь в том, что файл создан в текущем каталоге, для чего воспользуйтесь командой `D:\USERS\840>dir`

На экране монитора появится сообщение:

```
Volume in drive D is WORK
Volume Serial Number is 1B29-8EF1
Directory of D:\USERS\840
.                <DIR>  9-10-1994 2:30p
..               <DIR>  9-10-1994 3:01p
TEXT.TXT        75    9-10-1994 1:24p
    __ file(s)    __ bytes
    _____ bytes free
D:\USERS\840>_
```

КОПИРОВАНИЕ ФАЙЛА

Для копирования файлов используется команда `copy`. Ее формат:

`copy <имя файла> <куда копировать>.`

Скопируйте файл text.txt с диска D:\USERS\840 в одну из папок, созданных при выполнении предыдущего задания (например, I1), с помощью команды `copy text.txt I1:`

```
D:\USERS\840>copy text.txt I1.
```

На экране монитора появляется сообщение:

1 file(s) copied D:\USERS\840>_

Убедитесь в том, что файл скопирован в нужный каталог, для чего воспользуйтесь командой `dir I1:`

```
D:\USERS\840>dir I1.
```

На экране монитора появится содержимое каталога I1. Найдите в нем файл с именем text.txt.

КОПИРОВАНИЕ ГРУППЫ ФАЙЛОВ

В названии файла можно употреблять символ *. Он заменяет собой любое число любых букв в имени или расширении файла:

*.txt — все файлы с расширением txt.

doc.* — файлы с именем doc и любым расширением.

. — любые файлы.

Например, команда `copy *.* a:\` означает копирование всех файлов из текущего каталога на дискету.

РАБОТА С ТЕКСТОВЫМИ ФАЙЛАМИ

Вывод содержимого файла на экран монитора

Формат команды: `type <имя файла>`.

Выведете на экран содержимое файла text.txt. В этом случае используется команда `D:\USERS\840>type text.txt`.

На экране появляется:

Мы, студенты гр. ..., поступили на ФВТ РГРТУ в ... году.

```
D:\USERS\840>_
```

Переименование файла

Переименование файлов осуществляется командой:

`ren <старое имя файла> <новое имя файла>`.

Задание 2

Переименовать текстовый файл text.txt в document.doc, для чего введите команду:

```
ren text.txt document.doc
```

```
D:\USERS\840>ren text.txt document.doc
```

```
D:\USERS\840>_
```

Убедитесь в том, что файл переименован. Просмотрите содержимое текущего каталога командой `D:\USERS\840>dir`.

На экране монитора появится сообщение:

```
Volume in drive D is WORK
Volume Serial Number is 1B29-8EF1
Directory of D:\USERS\840
.                <DIR>  9-10-1994 2:30p
..               <DIR>  9-10-1994 3:01p
FACULTY.DOC      75    9-10-1994 3:09p
    __ file(s)    __ bytes
    _____ bytes free
D:\USERS\840>_
```

Удаление файла

Для удаления файлов используется команда `del: del <имя файла>`

Задание 3

Удалите файл `document.doc` в каталоге `D:\USERS\840`, для чего введите команду `D:\USERS\840>del document.doc`.

Предупреждение! Убедитесь в том, что именно этот файл вы хотите удалить, проверив правильность написания имени файла.

После выполнения команды система выведет приглашение:

```
D:\USERS\840>_
```

Убедитесь в том, что файл удален, для чего просмотрите содержимое текущего каталога:

```
D:\USERS\840>dir
Volume in drive D is WORK
Volume Serial Number is 1B29-8EF1
Directory of D:\USERS\740
.                <DIR>  9-10-1994 2:30p
..               <DIR>  9-10-1994 3:01p
    __ file(s)    __ bytes
    _____ bytes free
D:\USERS\840>_
```

Можно сразу удалить группу файлов. Например, команда `del *.txt` означает удаление файлов с расширением `txt` из текущего каталога.

Контрольные вопросы

1. Основные носители информации и их краткая характеристика.
2. Назначение режима командной строки.
3. Что расположено на передней панели привода CD-ROM?

4. Понятие файла. Правила построения имен файлов.
5. Понятие каталога. Поясните принцип построения структуры каталогов на диске.
6. Поясните смысл понятий: «текущий дисковод», «текущий каталог», «путь к файлу», «полное имя файла».
7. Поясните на примерах использование символов «*» и «?» в именах файлов.
8. Формат команды режима командной строки и возможности ее редактирования.
9. В файловой структуре (см. табл. Л-2.1, вариант 10) текущим является каталог I_6 . Запишите команды для выполнения следующих действий:
 - а) создать в каталоге I_4 два текстовых файла с вашими именами. Скопировать содержимое каталога на дискету и убедиться, что операция реализована;
 - б) создать в каталоге I_4 два текстовых файла с вашими именами и переименовать их;
 - в) удалить каталоги I_4 и I_5 .
10. Запишите последовательность команд для выполнения следующих действий: установить диск Z в качестве текущего, перейти к его корневому каталогу, установить текущий каталог ACAD; вывести на экран содержимое каталога LEXICON (тремя способами: вывод по умолчанию, с явным указанием текущего каталога и его полному пути).
11. Запишите последовательность команд для выполнения следующих действий: образовать в каталоге FAM два файла с именами F1.TXT и F2.TXT. В первый файл ввести строчку «Здравствуйте, ...», во второй — строчку со своей фамилией и инициалами, например «Петров И.И.». Убедитесь, что файлы созданы и содержат введенную информацию.

Лабораторная работа 3

ОСНОВЫ РАБОТЫ В WINDOWS 2000/XP: ФАЙЛЫ И ПАПКИ

3.1. Цель работы

Изучение основных приемов работы с файлами и папками в ОС Windows 2000/XP.

3.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 7—9].
2. Подготовка отчета (дома).

3. Индивидуальный входной контроль (основные операции с файлами и папками: открытие, копирование, перемещение, создание, переименование, удаление, восстановление, особенности выполнения; использование интерфейсных элементов при работе с файлами и папками).
4. Работа на компьютере:
 - а) регистрация:
 - ◆ открыть окно *Мой компьютер*;
 - ◆ выбрать диск D:;
 - ◆ открыть папку USERS;
 - ◆ открыть папку с именем номера группы (для удобства работы остальные окна закрыть);
 - б) работа с файлами и папками (см. подраздел 3.5);
 - в) выполнение заданий (см. подраздел 3.6).
5. Отчет о проделанной работе:
 - а) индивидуальное выполнение на ПК заданной преподавателем операции (создание файла или папки, отправка их на диск, перемещение, изменение имени и т.д.);
 - б) оформление отчета;
 - в) защита отчета.

3.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ основные операции с файлами и папками, используемые интерфейсные средства (основные понятия, перечень операций и особенности их выполнения).

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ протокол выполнения практических заданий;
- ◆ дерево каталогов (раздел 3.6, задание 10).

3.4. Файлы и папки

ОСНОВНЫЕ ПОНЯТИЯ

Пользователю часто приходится работать с файлами и папками. Под *папкой* в общем случае понимается именованное место хранения программ, документов и других папок. Папке в *Windows* ставятся в соответствие такие параметры, как тип, имя, размер, дата создания.

Значок, используемый для отображения папки с файлами, приведен ниже:



Значки, используемые для отображения файлов в окнах приложений, определяются типом файла.

ПРАВИЛА ФОРМИРОВАНИЯ ИМЕН ФАЙЛОВ В WINDOWS

В Windows принята следующая конструкция имен папок и файлов:

- ♦ имена и расширения могут содержать до 255 символов;
- ♦ в имени файла запрещено использование определенных символов: * | \ ? / " :

Эти символы зарезервированы для использования операционной системой.

Windows отображает имена объектов теми буквами (прописными и строчными), которыми оно вводится, однако делает исключение для названий файлов или папок, набранных в одном регистре (верхнем или нижнем), длина которых не превышает восьми символов.

3.5. Работа с файлами и папками

ОТКРЫТИЕ ФАЙЛА ИЛИ ПАПКИ

Для *открытия* файла или папки следует выполнить следующую последовательность действий.

1. Дважды щелкнуть мышью по значку Мой компьютер.
2. Дважды щелкнуть мышью по диску, на котором содержится файл. Это может быть гибкий, жесткий или лазерный диск.
3. Дважды щелкнуть мышью по файлу или по папке, которую следует открыть. Перечисленная последовательность действий уже выполнялась в данной работе в ходе регистрации (см. подраздел 3.2, п. 4, а) и вы уже открыли папку с именем вашей группы. Если нужная папка или файл находятся не в корневом каталоге диска, то предварительно необходимо двойным щелчком мыши по папке перейти в директорию, где находится необходимый файл или папка.

Примечания: 1. Изменить способ отображения файлов в окне позволяют команды меню *Вид*.

2. Если файл находится на другом компьютере, вместо значка *Мой компьютер* следует дважды щелкнуть мышью по значку *Сетевое окружение*.

ОТКРЫТИЕ НЕДАВНО ИСПОЛЬЗОВАВШЕГОСЯ ФАЙЛА

Последовательность действий для *открытия недавно использовавшегося файла* следующая.

1. Щелкнуть левой клавишей мыши по кнопке *Пуск* и выбрать команду *Документы* (рис. Л-3.1).
2. Выбрать нужный документ, щелкнув по нему мышью.

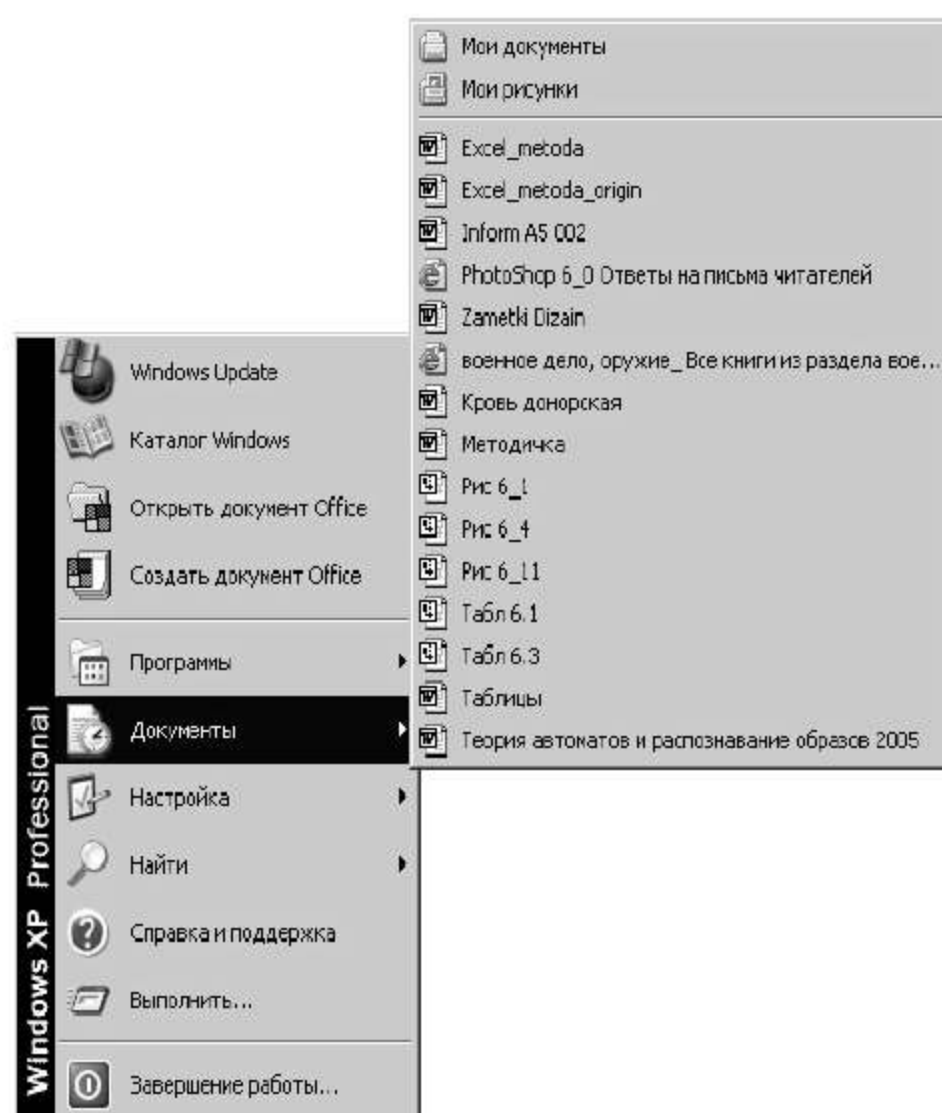
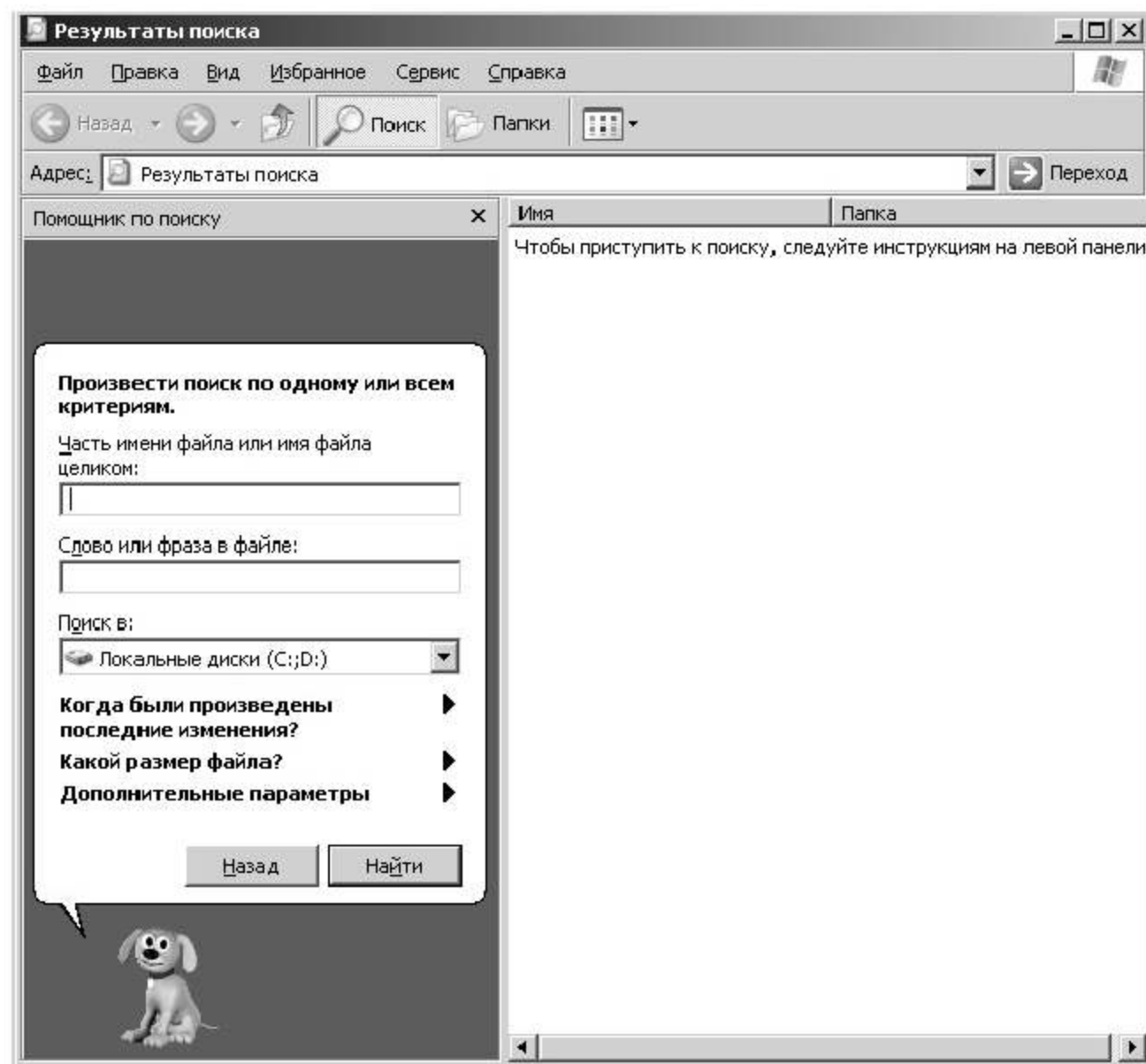


Рис. Л-3.1. Меню команды *Документы*

Примечание. Некоторые программы не помещают имена открытых файлов в меню *Документы*. Если нужный документ отсутствует в списке, нажмите кнопку *Пуск*, выберите команду *Найти | Файлы и папки*. В открывшемся окне в поле *Искать имена файлов или папок* (рис. Л-3.2) наберите имя нужного файла или папки, нажмите *Найти*. Затем в списке имен найденных папок и файлов в нижней части окна выберите нужное имя и дважды щелкните мышью.

Рис. Л-3.2. Окно команды *Найти***КОПИРОВАНИЕ ФАЙЛА ИЛИ ПАПКИ**

Для копирования файла или папки необходимо:

2. В окне *Мой компьютер** выбрать файл или папку, которую требуется скопировать.
2. Выполнить одно из действий:
 - а) в меню *Правка* выбрать команду *Копировать*;
 - б) нажать правую клавишу мыши и в появившемся контекстном меню выбрать пункт *Копировать* (рис. Л-3.3);
 - в) нажать комбинацию клавиш [Ctrl] + [C].
3. Открыть папку или диск, куда требуется вставить копию.

* При выполнении действий с папками и файлами вместо окна *Мой компьютер* следует работать с папкой с именем вашей группы.

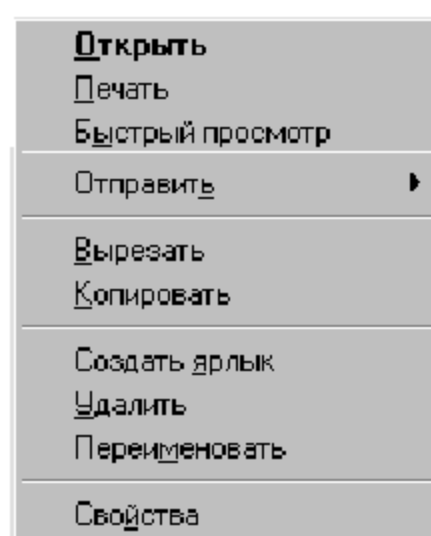


Рис. Л-3.3. Контекстное меню

4. Выполнить одно из действий:
- а) в меню *Правка* выбрать команду *Вставить*;
 - б) нажать правую клавишу мыши и выбрать пункт *Вставить*;
 - в) нажать комбинацию клавиш [Ctrl] + [V].

Примечание. Для выделения сразу нескольких файлов или папок выбирайте объекты, удерживая клавишу [Ctrl].

ПЕРЕМЕЩЕНИЕ ФАЙЛА ИЛИ ПАПКИ

Для *перемещения* файла или папки необходимо:

- 1) в окне *Мой компьютер* (* см. предыдущую сноску) выбрать файл или папку, которую требуется переместить;
- 2) выполнить одно из действий:
 - а) в меню *Правка* выбрать команду *Вырезать*;
 - б) нажать правую клавишу мыши (рис. Л-3.3) и выбрать *Вырезать*;
 - в) нажать комбинацию клавиш [Ctrl] + [X];
- 3) открыть папку или диск, куда требуется поместить файл или папку;
- 4) выполнить одно из действий:
 - а) в меню *Правка* выбрать команду *Вставить*;
 - б) нажать правую клавишу мыши и выбрать *Вставить*;
 - в) нажать комбинацию клавиш [Ctrl] + [V].

Примечание. Файл или папку можно просто перетащить в новое место.

СОЗДАНИЕ ЯРЛЫКА

Ярлыки позволяют быстро запустить программу или открыть файл или папку без необходимости искать расположение файла в окне

проводника. Особенно полезно создавать ярлыки для часто используемых программ, файлов или папок. Существуют два способа создания ярлыков.

Создание ярлыка в папке

Для этого необходимо:

- 1) в окне *Мой компьютер* (* см. предыдущую сноску) выбрать папку, в которой требуется создать ярлык;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Создать*, а затем выбрать команду *Ярлык*;
 - б) нажать правую клавишу мыши и выбрать *Создать | Ярлык*;
- 3) выполнить выведенные на экран инструкции.

Помещение ярлыка на рабочий стол

Последовательность действий при этом следующая:

- 1) в окне *Мой компьютер* (* см. предыдущую сноску) выбрать объект, например файл, программу, папку, принтер или компьютер, для которого требуется создать ярлык;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Создать ярлык*;
 - б) нажать правую клавишу мыши и выбрать *Создать | Ярлык*;
- 3) перетащить ярлык на *Рабочий стол*.

Примечания: 1. Чтобы изменить параметры ярлыка, например тип открываемого окна, щелкните на ярлыке правой кнопкой и выберите команду *Свойства*.
2. Чтобы удалить ярлык, перетащите его на значок *Корзина*. На сам объект удаление ярлыка не влияет.

СОЗДАНИЕ ФАЙЛА

Для создания файла необходимо:

- 1) перейти в папку, в которой необходимо создать файл;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Создать*, а затем выбрать тип создаваемого файла (рис. Л-3.4);
 - б) щелкнуть правой клавишей мыши, выбрать команду *Создать* и тип создаваемого файла *Текстовый документ* (рис. Л-3.4).

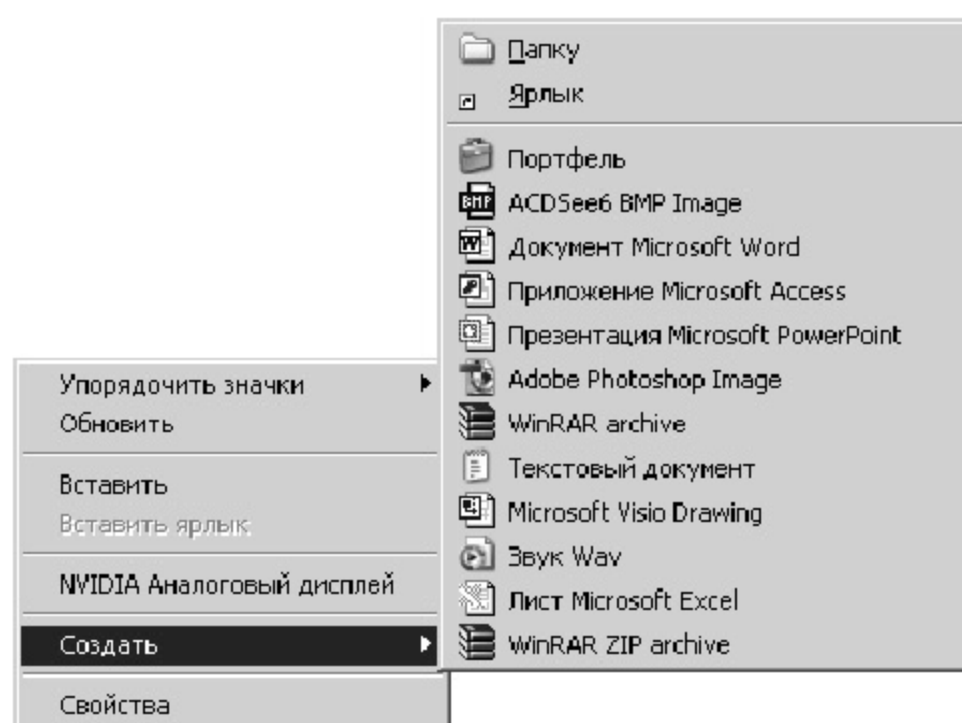


Рис. Л-3.4. Меню создания файла

В окне появляется новый файл с именем, которое присвоено ему по умолчанию;

- 3) ввести имя нового файла и нажать клавишу [Enter].

СОЗДАНИЕ ПАПКИ

Для создания папки необходимо:

- 1) выбрать диск, на котором требуется создать новую папку;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Создать*, а затем выбрать *Папка*;
 - б) нажать правую клавишу мыши и выбрать *Создать | Папка*.
 В окне появляется новая папка с именем, которое присвоено ей по умолчанию;
- 3) ввести имя новой папки (переход на русский язык [Ctrl] + [Shift]) и нажать клавишу [Enter].

Примечание. Если новую папку требуется создать внутри другой папки, выберите эту папку и выполните шаги 2 и 3.

ИЗМЕНЕНИЕ ИМЕНИ ФАЙЛА ИЛИ ПАПКИ

Для *изменения имени* файла или папки необходимо выполнить следующие действия:

- 1) выделить файл или папку, которую требуется переименовать;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Переименовать*;
 - б) щелкнуть левой клавишей мыши на имени файла;
- 3) ввести новое имя и нажать клавишу [Enter].

УДАЛЕНИЕ ФАЙЛА ИЛИ ПАПКИ

Порядок действий для удаления следующий:

- 1) выбрать файл или папку, которую требуется удалить;
- 2) выполнить одно из действий:
 - а) в меню *Файл* выбрать команду *Удалить*;
 - б) нажать правую клавишу мыши (см. рис. Л-3.3) и выбрать команду *Удалить*.

Примечание. Можно также напрямую перетащить значок файла или папки на значок *Корзина*. Если при перетаскивании удерживать клавишу [SHIFT], объект будет удален без помещения в папку *Корзина*.

ВОССТАНОВЛЕНИЕ УДАЛЕННЫХ ФАЙЛОВ И ЯРЛЫКОВ

Для восстановления удаленного файла необходимо:

- 1) на рабочем столе дважды щелкнуть мышью по значку *Корзина*;
- 2) выбрать файлы или ярлыки, которые требуется восстановить;
- 3) в меню *Файл* выбрать команду *Восстановить*.

Примечания: 1. При восстановлении файла, находившегося в удаленной папке, вначале будет восстановлена сама эта папка.
2. Файлы, удаленные в окне эмуляции командной строки, файлы, удаленные с сетевых дисков, а также файлы, удаленные со съемных носителей, в папку *Корзина* не помещаются. Такие файлы удаляются сразу без возможности восстановления.
3. Чтобы открыть файл, находящийся в папке *Корзина*, перетащите его значок на *Рабочий стол*, а затем дважды щелкните по значку.
4. Чтобы восстановить только часть файлов из папки *Корзина*, выбирайте их, удерживая клавишу [Ctrl], а затем выберите в меню *Файл* команду *Восстановить*.

СПРАВКА

Для получения справки при работе с файлами и папками необходимо выполнить одно из действий:

- ♦ нажать клавишу F1;
- ♦ щелкнуть левой клавишей мыши на кнопке *Пуск* и выбрать команду *Справка*.

3.6. Практическая работа на ПК**ЗАДАНИЯ**

1. Создать папку DIR_(N группы)_(N бригады), например DIR_840_5.

2. Внутри созданной папки создать еще две с именами Dir_1 и Dir_2.
3. Внутри папки Dir_1 создать текстовый документ с именем Text.
4. Внутри папки Dir_2 создать файл точечного рисунка с именем Image.
5. Скопировать файл Text в папку Dir_2, а в папке Dir_1 удалить его.
6. Переместить файл Image из папки Dir_2 в папку Dir_1.
7. Восстановить из корзины файл Text. Убедиться, что он появился в папке Dir_1.
8. В папке Dir_1 переименовать файл Text в Document.
9. В папке Dir_2 создать ярлык для файла Image.
10. Просмотреть и зарисовать в отчет получившееся дерево каталогов с расположенными в них файлами (рис. Л-3.5).

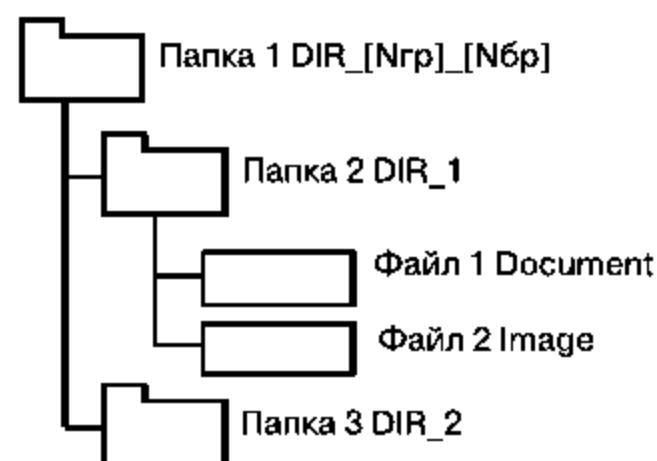


Рис. Л-3.5. Пример дерева каталогов

Контрольные вопросы

1. Понятие файла и папки в Windows, особенности.
2. Назовите основные операции, выполняемые над файлами и папками.
3. Какие файлы содержатся в меню команды *Документы*?
4. Для чего нужна папка *Корзина*?
5. Какие файлы не помещаются в папку *Корзина*?
6. Какие ограничения вводятся на имена файлов в Windows?
7. Как можно выделить несколько файлов или папок?

Лабораторная работа 4
ОСНОВЫ РАБОТЫ В WINDOWS 2000/XP:
СТАНДАРТНЫЕ ПРОГРАММЫ БЛОКНОТ И PAINT

4.1. Цель работы

Изучение основных приемов работы с текстовым и графическим редакторами. Отработка навыков создания текстовых и графических объектов.

4.2. Порядок выполнения работы

1. Изучение текстового (*Блокнот*) и графического (*Paint*) редакторов (дома). Дополнительная литература [методические указания, 7—9].
2. Подготовка отчета (дома).
3. Подготовка небольшого фрагмента текста для редактирования и рисунка. Текст должен содержать 6—7 печатных строк и может касаться деятельности РГРТУ, кафедры, факультета, описания программ или ОС Windows (можно использовать описания предыдущих лабораторных работ). Пример текста приведен на рис. Л-4.4. Рисунок должен быть содержательным и включать функционально необходимые элементы, связанные между собой, а не простой набор разнородных элементов. (Это может быть, например, вагон троллейбуса, схема алгоритма, рисунок дома, офиса и др.)
4. Индивидуальный входной контроль (назначение стандартных программ, их возможности, создание и сохранение текстовых документов, поиск определенных символов или слов, перенос, копирование текста, основные операции графического редактора, создание графического файла и т.д.).
5. Практическая работа на ПК:
 - а) регистрация:
 - ◆ открыть окно *Мой компьютер*;
 - ◆ выбрать диск D:;
 - ◆ открыть папку USERS;
 - ◆ открыть папку с именем номера группы (для удобства работы остальные окна закрыть);
 - б) работа с редакторами (запуск, выполнение различных операций);
 - в) выполнение практических заданий (см. подраздел 4.7).

6. Отчет о проделанной работе:

- ◆ индивидуальное выполнение заданных преподавателем операций по работе с текстом либо графическим редактором;
- ◆ оформление отчета с результатами выполненных заданий;
- ◆ защита отчета.

4.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ назначение программ *Блокнот* и *Paint*, основные возможности, операции, особенности.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ протокол выполнения практических заданий;
- ◆ результаты выполнения практических заданий (ч. 1 п. 10 и ч. 2 п. 5).

4.4. Стандартные программы Блокнот и Paint

ОБЩИЕ СВЕДЕНИЯ

Работа любого вуза, предприятия, организации немыслима без создания сотрудниками разнообразных текстовых и графических документов, обработки входящей и исходящей корреспонденции. Текстовые документы порождаются также практически любым приложением ОС Windows. Поэтому обработка текстовой (с буквами и цифрами) и графической (с рисунками и фотографиями) информации — одна из наиболее часто встречающихся задач офисной деятельности.

Для изучения основных приемов ввода и редактирования текстовых и графических документов можно использовать программы *Блокнот* и *Paint*.

Текстовый редактор *Блокнот* является одним из стандартных приложений Windows и распространяется вместе с операционной системой. Он предназначен для создания и редактирования несложных текстовых файлов, не требующих форматирования. *Блокнот* работает только с файлами в формате ASCII (простой текст).

Paint — это программа, поставляемая в комплекте с Windows и предназначенная для создания и редактирования на экране изображений (картинок). Эти изображения сохраняются в графических файлах растрового формата (.BMP или .PCX). Каждое изображение — это мозаика из точек (пикселей), окрашенных в тот или иной цвет. Создан-

ные в *Paint* рисунки можно вставлять в документы всех универсальных приложений Windows: Word, Excel, Power Point. Редактор *Paint* позволяет создавать довольно сложные и внешне привлекательные рисунки, схемы, чертежи (в цвете или черно-белые). Средства и инструменты пользователя: палитра цветов, кисть, распылитель, ластик для стирания, карандаши для рисования геометрических фигур (линий, прямоугольников, эллипсов, многоугольников). Редактор позволяет вводить тексты, а богатый набор шрифтов из комплекта Windows дает возможность вставлять в рисунок профессиональные надписи. Имеются и ножницы для вырезания фрагментов картинки — вырезанный элемент можно переместить, скопировать, уменьшить, увеличить, наклонить и т.д.

Paint — неплохое средство для обучения элементарным навыкам обработки графических объектов, которые помогут при необходимости освоить более мощные графические средства. Вместе с тем *Paint* не предназначен для серьезных графических работ, для технического проектирования (деталей, машин, домов), для редактирования фотоиллюстраций и т.д. Можно сказать, что в семье графических редакторов *Paint* занимает скромное место, однако для большинства пользователей возможностей *Paint* вполне достаточно.

4.5. Текстовый редактор Блокнот

ЗАПУСК ПРОГРАММЫ

Загрузка программы-редактора *Блокнот* производится по команде главного меню *Пуск* | *Программы* | *Стандартные* | *Блокнот*.

После запуска на экране монитора появляется рабочее окно редактора, похожее на окно, представленное на рис. Л-4.1.

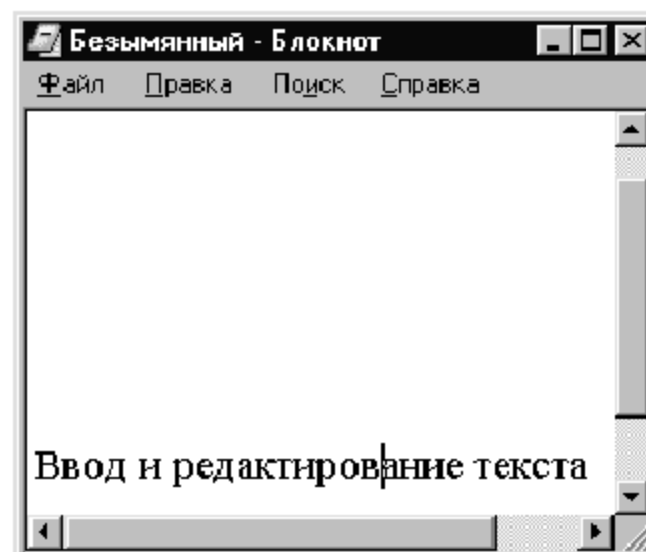


Рис. Л-4.1. Окно текстового редактора *Блокнот*

СОЗДАНИЕ И СОХРАНЕНИЕ ТЕКСТОВЫХ ДОКУМЕНТОВ

Для *создания* текстового документа необходимо выполнить следующие шаги.

1. Перейти в папку, где будет размещаться новый файл.
2. Щелкнуть правой клавишей мыши и выбрать *Создать | Текстовый документ* (ЛР 3, см. рис. Л-3.4).
3. Ввести имя нового файла и нажать [Enter].
4. Нажать [Enter], после чего откроется окно *Блокнот*.
5. Набрать текст файла.
6. Выбрать в *Блокноте* меню *Файл | Сохранить* для сохранения под тем же именем или выбрать *Файл | Сохранить как...* и ввести новое имя.

ПОИСК ОПРЕДЕЛЕННЫХ СИМВОЛОВ ИЛИ СЛОВ

Для *поиска* символов и слов в тексте необходимо выполнить следующие шаги.

1. В меню *Поиск* выбрать команду *Найти*.
2. В поле *Образец* (рис. Л-4.2) ввести фрагмент текста, который требуется найти.
3. Нажать кнопку *Найти далее*.

Примечание. Если надо найти последующее повторение строки поиска, то необходимо еще раз нажать мышью кнопку *Найти далее* в окне поиска или, если вы закрыли окно поиска, выбрать в меню *Поиск* пункт *Найти далее*.

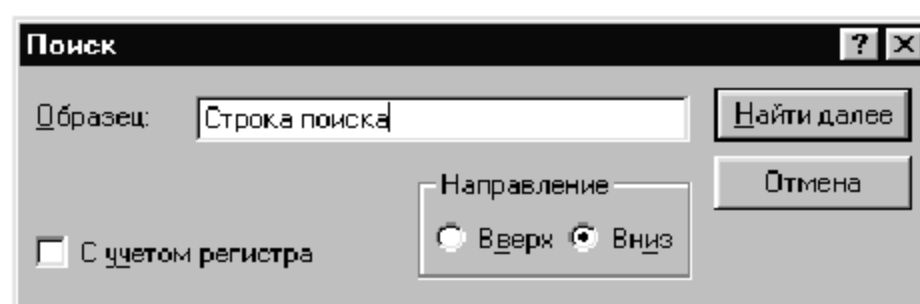


Рис. Л-4.2. Окно поиска строки

ПЕРЕНОС, КОПИРОВАНИЕ И УДАЛЕНИЕ ТЕКСТА

Для работы с фрагментом текста его необходимо сначала *выделить*, что можно сделать, выполнив следующие действия:

- ♦ установить курсор в начало или в конец выделяемого фрагмента;
- ♦ нажать кнопку мыши и, не отпуская ее, протащить указатель мыши до противоположной границы фрагмента, а затем, чтобы:
- ♦ *перенести* — в меню *Правка* выбрать пункт *Вырезать*, после чего переместить курсор в новое место и в меню *Правка* выбрать пункт *Вставить*;

- ♦ *скопировать* — в меню *Правка* выбрать пункт *Копировать*, переместить курсор в новое место и в меню *Правка* выбрать пункт *Вставить*;
- ♦ *удалить* — в меню *Правка* выбрать пункт *Удалить*.

Для отмены удаления фрагмента необходимо в меню *Правка* выбрать пункт *Отменить*.

СПРАВКА

Для получения справки при работе с программой *Блокнот* необходимо выполнить одно из действий:

- ♦ нажать клавишу [F1];
- ♦ в меню *Справка* выбрать пункт *Вызов справки*.

4.6. Графический редактор Paint

ОСНОВНЫЕ ОПЕРАЦИИ ГРАФИЧЕСКОГО РЕДАКТОРА:

- ♦ рисование указателем мыши произвольных изображений;
- ♦ ввод текстовых надписей тем или иным шрифтом;
- ♦ увеличение, уменьшение, изменение пропорций картинки или ее участка;
- ♦ операции с фрагментами, позволяющие вырезать участки картинки, перемещать их, копировать, составлять изображение из нескольких фрагментов и т.д.;
- ♦ дополнительные операции (стирание участков картинки, изменение цветов, печать и т.д.).

ЗАПУСК ПРОГРАММЫ

Запуск программы *Paint* осуществляется командой главного меню *Пуск | Программы | Стандартные | Paint*. После этого на экране отображается окно *Paint* (рис. Л-4.3), в котором выделяются следующие основные элементы:

- ♦ *панель инструментов* — позволяет выбирать средство или так называемый инструмент рисования;
- ♦ *палитра* — позволяет выбирать основной (цвет рисования) и фоновый цвет;
- ♦ *строка состояния* — показывает выбранный инструмент и координаты курсора.

СОЗДАНИЕ И СОХРАНЕНИЕ ГРАФИЧЕСКОГО ФАЙЛА

Для создания графического файла необходимо выполнить следующие действия.

1. Открыть папку, где будет размещаться новый файл.
2. Нажать правую клавишу мыши и выбрать *Создать | Точечный рисунок*.

3. Ввести имя нового файла и нажать [Enter].
4. Нажать [Enter], после чего откроется окно *Paint*.
5. Нарисовать рисунок (простой, в порядке пробы).
6. Выбрать в меню *Файл | Сохранить* для сохранения под тем же именем или выбрать *Файл | Сохранить как...* и ввести новое имя.

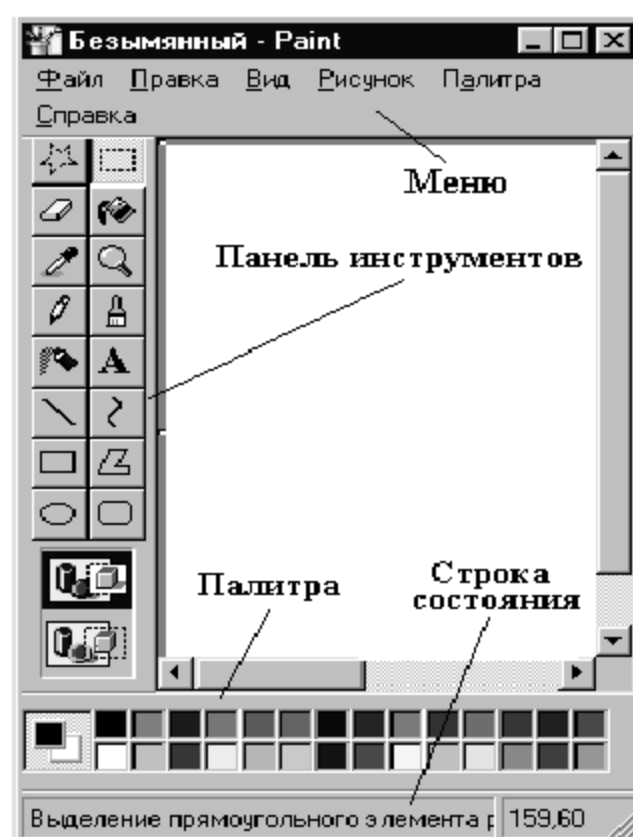


Рис. Л-4.3. Окно графического редактора *Paint*

СОСТАВ ПАНЕЛИ ИНСТРУМЕНТОВ

В состав панели инструментов входят следующие кнопки:

- *выделение произвольной области* — выделение области произвольной формы;
- *выделение* — выделение прямоугольной области;
- *ластик/цветной ластик* — стирание и замена одного цвета на другой;
- *заливка* — закраска ограниченных областей и фигур цветом;
- *выбор цветов* — выбор в качестве основного или цвета фона цвета, присутствующего на рисунке;
- *масштаб* — изменение текущего масштаба отображения рисунка;
- *карандаш* — рисование произвольной линии;

-  — *кисть* — рисование произвольных линий различной толщины;
-  — *распылитель* — рисование баллончиком с краской;
-  — *надпись* — ввод и форматирование текста;
-  — *линия* — рисование прямой линии различной толщины;
-  — *кривая* — рисование линии заданной кривизны;
-  — *прямоугольник* — рисование прямоугольника;
-  — *многоугольник* — рисование произвольного многоугольника;
-  — *эллипс* — рисование окружностей и эллипсов;
-  — *скругленный прямоугольник* — рисование прямоугольника со скругленными углами.

- Примечания:**
1. Текущие основной цвет и цвет фона показываются слева от палитры. Для рисования основным цветом необходимо использовать левую кнопку мыши, цветом фона — правую.
 2. Выбор основного цвета осуществляется щелчком левой кнопки по палитре, выбор цвета фона — правой.
 3. Существует возможность отменить три последних действия при редактировании изображения. Для этого надо использовать команду *Отменить* в меню *Правка*.

СПРАВКА

Для получения справки при работе с программой *Paint* необходимо выполнить одно из действий — нажать клавишу F1 или в меню *Справка* выбрать пункт *Вызов справки*.

4.7. Практическая работа на ПК

Практическая работа включает две части:

- ♦ работу с текстовым редактором;
- ♦ работу с графическим редактором.

ЧАСТЬ 1. РАБОТА С ТЕКСТОВЫМ РЕДАКТОРОМ БЛОКНОТ

Задания

1. Создать текстовый файл *Document* (щелкнуть правой кнопкой мыши и выбрать в контекстном меню *Создать | Текстовый документ*).
2. Открыть окно *Блокнот* (дважды щелкнуть по новому файлу левой кнопкой мыши).

3. Набрать приготовленный дома фрагмент текста (пример на рис. Л-4.4).

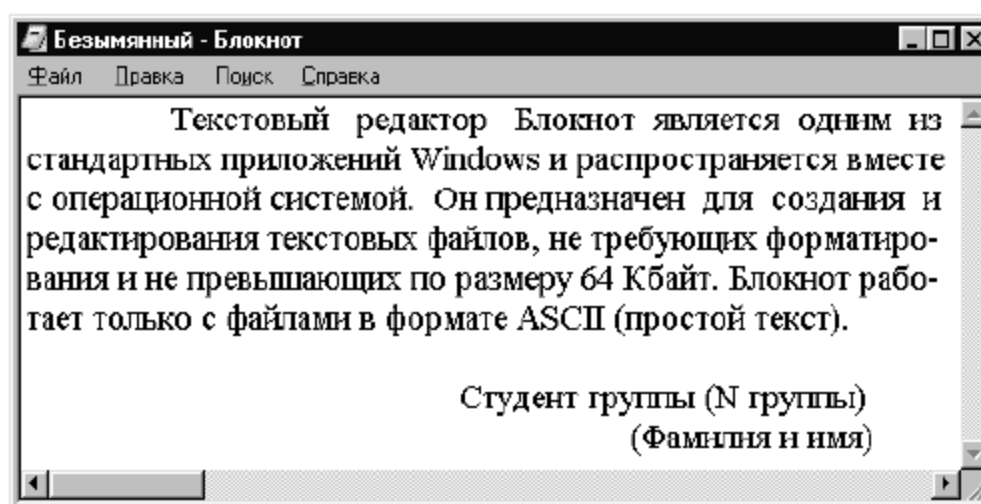


Рис. Л-4.4. Пример текста

4. Сохранить изменения в файле *Document*.
5. Заккрыть окно *Блокнот*.
6. Открыть файл *Document*, используя команду *Документ* в меню *Пуск*.
7. Отредактировать приготовленный дома фрагмент текста:
 - а) перенести третью строку в начало текста;
 - б) пятую строку текста скопировать так, чтобы две одинаковые строки следовали друг за другом;
 - в) найти конкретное слово текста, а затем слово, заведомо отсутствующее в тексте.
8. Сохранить изменения в файле *Document*.
9. Заккрыть окно *Блокнот*.
10. Записать в отчет последовательность операций и действий, выполненных в ходе редактирования текста.

ЧАСТЬ 2. РАБОТА С ГРАФИЧЕСКИМ РЕДАКТОРОМ PAINT

Задания

1. Создать графический файл *Image* (щелкнуть правой кнопкой мыши и выбрать в контекстном меню *Создать | Точечный рисунок*).
2. Открыть окно *Paint* (дважды щелкнуть по новому файлу левой кнопкой мыши).
3. Нарисовать заготовленный дома рисунок (например, рис. Л-4.5), используя при этом все(!) инструменты *Панели инструментов*.

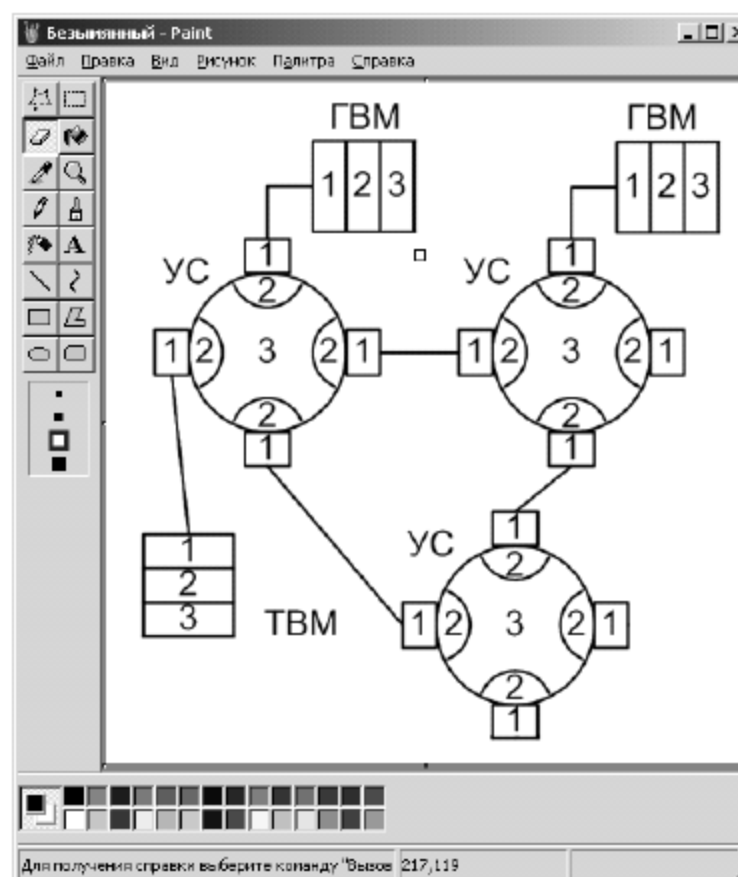


Рис. Л-4.5. Фрагмент сети

4. Сохранить изменения в файле *Image*.
5. Для созданного рисунка записать в отчет список графических объектов и использованных для их рисования инструментов.
6. Закрыть окно *Paint*.

Контрольные вопросы

1. Назначение программы *Блокнот* и ее основные особенности.
2. Назначение программы *Paint* и ее основные особенности.
3. Назовите основные операции работы с текстовыми документами.
4. Назовите основные операции работы с фрагментами текста.
5. Назовите основные операции графического редактора *Paint*.
6. Назовите основные инструменты рисования редактора *Paint*.
7. Каким образом выбираются основной и фоновый цвета в редакторе *Paint*?
8. Какие инструменты применяются для рисования различных линий?

Лабораторная работа 5
ОСНОВЫ РАБОТЫ НА ПК
ПОД УПРАВЛЕНИЕМ ОПЕРАЦИОННОЙ ОБОЛОЧКИ
FAR MANAGER

5.1. Цель работы

Изучение основных функций FAR Manager и их реализация при работе с каталогами и файлами. Управление дисковой и файловой системами. Нарботка практических навыков управления компьютером с помощью операционной оболочки FAR Manager.

5.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 7—9].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (цель работы, общее представление о FAR Manager, функциональные возможности FAR Manager, технология работы на ПК под управлением FAR Manager, комбинации клавиш для управления панелями, команды оболочки FAR Manager: понятия, способы задания, перечень команд для работы с каталогами и файлами).
4. Работа на компьютере:
 - ◆ запуск FAR Manager;
 - ◆ выполнение операций с каталогами и файлами под управлением FAR Manager.
5. Создание дерева каталогов в соответствии с заданным вариантом (см. табл. Л-2.1, ЛР 2).
6. Отчет о проделанной работе:
 - ◆ индивидуальное выполнение на ПК операций по заданию преподавателя (перейти в нужный каталог, удалить каталог, скопировать названный файл и др.);
 - ◆ оформление отчета (дорабатывается по результатам практической работы на ПК);
 - ◆ защита отчета.

5.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ цель работы, основные сведения о FAR Manager, особенности работы на ПК под управлением FAR Manager.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ♦ подробный протокол о выполнении практических заданий по работе с каталогами и файлами, работе в сети.

5.4. Основные сведения об операционной оболочке FAR Manager**ЧТО ТАКОЕ FAR MANAGER?**

FAR Manager — программа управления файлами и архивами в операционных системах семейства Windows (9x/Me/NT/2000/XP). FAR Manager работает в текстовом режиме и позволяет просто и наглядно выполнять большинство необходимых действий: просматривать файлы и каталоги, редактировать, копировать и переименовывать файлы, а также многое другое.

FAR Manager имеет многоязычный, легко настраиваемый интерфейс.

Функциональность FAR Manager существенно расширяется за счет внешних подключаемых DLL-модулей — *плагинов* (этому способствует набор специальных интерфейсов — Plugins API). Например, работа с архивами, FTP-клиент, временная панель и просмотр сети реализованы с помощью *плагинов*, включенных в стандартную поставку FAR.

Программа FAR Manager (версия 1.70) является одной из популярных программ-оболочек для работы с ОС Windows и обладает следующими функциональными возможностями:

- ♦ выполнение большинства команд режима командной строки;
- ♦ наглядное изображение структуры файлов и каталогов на магнитном диске;
- ♦ копирование, переименование, удаление файлов и каталогов;
- ♦ просмотр и поиск файлов различных типов;
- ♦ архивирование и разархивирование файлов;
- ♦ осуществление связи между компьютерами;
- ♦ редактирование текстовых файлов и т.д.

Таким образом, большинство действий, реализуемых с помощью ОС Windows, может быть выполнено с помощью FAR Manager. В FAR Manager для выполнения операций с файлами и каталогами необходимы лишь клавиши: управления курсором [↓] [↑] [←] [→], функциональные [F1]...[F10], [Enter] и только в случаях ввода текстовой информации — остальная часть клавиатуры. Кроме того, не требуется помнить структуру каталогов и файлов, потому что вся информация о ней выводится

на экран монитора в удобной форме. Если есть манипулятор «мышь», то можно все операции выполнять с его помощью, что еще более облегчает работу. Можно сказать, что FAR Manager выполняет тройную функцию, обеспечивая:

- 1) наглядное отображение файловой системы на экране и удобные средства для путешествий по этой системе;
- 2) простой и гибкий механизм диалога с Windows;
- 3) всевозможные служебные функции (манипуляция с файлами, информационный сервис и др.).

ЗАГРУЗКА FAR MANAGER. ЭКРАН FAR MANAGER

Файлы, необходимые для работы программы FAR Manager, обычно находятся на диске C: в каталоге Program Files\FAR.

Для запуска FAR Manager необходимо:

- ◆ щелкнуть мышью по кнопке *Пуск*;
- ◆ выбрать пункт *Выполнить...*;
- ◆ в появившемся окне ввести *far* и нажать [Enter].

В Windows ярлык *FAR Manager* часто помещают на *Рабочий стол*.

Для завершения работы с FAR Manager необходимо нажать клавишу [F10].



Рис. Л-5.1. Общий вид экрана FAR Manager

После запуска FAR Manager в верхней части появляются две синие панели (окна), каждая из которых содержит оглавление одного из каталогов файловой системы (рис. Л-5.1). Обе панели имеют одинаковое

назначение и функциональные возможности. Ниже располагается командная строка, где можно вводить команды операций системы. В последней строке экрана находится список клавиш [F1]—[F10] с кратким обозначением их функций. В частном случае на обеих панелях может демонстрироваться один и тот же каталог. Имя логического диска и имя каталога указаны в *заголовке* каждой панели.

Заголовок одной из панелей выделен серо-зеленым цветом, означающим, что именно этот диск и этот каталог являются текущими, т.е. рабочими.

5.5. Практическая работа на ПК

ВЫПОЛНЕНИЕ ОПЕРАЦИЙ

Смена языка интерфейса

Для смены языка интерфейса необходимо:

- ◆ нажать клавишу [F9];
- ◆ перейти в меню *Options* (Параметры) с помощью курсорных клавиш и выбрать *Languages* (Языки), нажать [Enter];
- ◆ в появившемся меню выбрать язык интерфейса, выделив нужную строку и нажав [Enter];
- ◆ в новом меню таким же образом выбрать язык справочной системы.

Задание 1

Убедитесь, что язык интерфейса FAR Manager является русским. Если это не так, то для смены языка интерфейса на русский необходимо:

- ◆ нажать [F9];
- ◆ перейти в меню *Options* и выбрать *Languages*, нажать [Enter];
- ◆ в появившемся меню *Main language* выбрать *Russian* (Русский) и нажать [Enter];
- ◆ в следующем меню также выбрать *Russian* (Русский) и нажать [Enter].

Смена диска

Для смены диска необходимо:

- ◆ одновременно нажать клавиши [Alt] и [F1] (левая панель) или [Alt] и [F2] (правая панель);
- ◆ в появившемся окне (рис. Л-5.2) с помощью клавиш управления курсором выбрать имя диска и нажать клавишу [Enter].

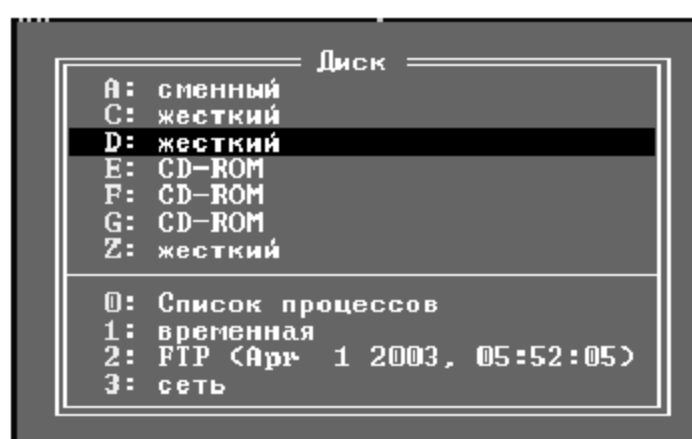


Рис. Л-5.2. Меню выбора

Задание 2

Перейдите в левой панели на диск D: и войдите в свой каталог для чего выполните следующие действия:

- ◆ нажмите комбинацию клавиш [Alt] + [F1];
- ◆ в меню (рис. Л-5.2) выберите диск D: и нажмите клавишу [Enter];
- ◆ выберите USERS и нажмите [Enter];
- ◆ выберите номер своей группы, например 740.

ОПЕРАЦИИ С КАТАЛОГАМИ

Для работы с каталогами в FAR Manager используют следующие клавиши:

[Ctrl] + [F1]	— убрать/показать левую панель;
[Ctrl] + [F2]	— убрать/показать правую панель;
[Ctrl] + [U]	— поменять панели местами;
[Ctrl] + [O]	— убрать/показать обе панели;
[Alt] + [F1]	— выбор диска на левой панели;
[Alt] + [F2]	— выбор диска на правой панели;
[Tab]	— смена активной панели;
[F7]	— создание каталога;
[Alt] + [F10]	— просмотр дерева каталогов;
[F8]	— удаление каталога.

Создание каталога (F7)

Для создания в текущем каталоге подкаталога необходимо выполнить следующие действия:

- ◆ нажать клавишу [F7];
- ◆ в появившемся на экране окне (*Создание каталога*) ввести имя создаваемого подкаталога;
- ◆ нажать клавишу [Enter].

Создайте каталог UNIVERSITY в своем каталоге и вложите в него каталоги FACULTY\CHAIR EVM.

Для этого:

- ◆ нажмите клавишу [F7];
- ◆ в появившемся окне наберите UNIVERSITY и нажмите клавишу [Enter].

Подкаталог UNIVERSITY появится на активной панели и будет выделен указателем:

- ◆ перейдите в каталог UNIVERSITY, нажав клавишу [Enter];
- ◆ нажмите клавишу [F7];
- ◆ в появившемся окне наберите FACULTY и нажмите клавишу [Enter];
- ◆ перейдите в каталог FACULTY, нажав клавишу [Enter];
- ◆ нажмите [F7];
- ◆ в появившемся окне наберите CHAIR EVM и нажмите клавишу [Enter].

В результате выполнения этих действий каталог FACULTY будет текущим.

Просмотр содержимого каталога

Содержимое текущего каталога высвечивается на панели. Один из файлов или подкаталогов на экране выделен указателем. Панель, содержащая указатель, называется *активной*. Клавишами перемещения курсора можно перемещать указатель внутри панели, выделяя другой файл или подкаталог. Клавишей [Tab] можно переместить указатель на другую панель.

Действия при нажатии клавиши [Enter]:

- ◆ если командная строка содержит команду, то выполняется эта команда;
- ◆ если командная строка пуста и выделено имя каталога или «. .», то нажатие клавиши [Enter] приводит к переходу в выделенный подкаталог или надкаталог (его содержимое высвечивается на панели);
- ◆ если командная строка пуста и выделено имя файла, то выполняемые действия зависят от его расширения.

Переход между каталогами

Для перехода в нужный подкаталог необходимо выделить указателем его имя на активной панели и нажать [Enter].

Для перехода в надкаталог требуется выделить указателем на активной панели строку «. .» и нажать [Enter].

Перейдите в каталог UNIVERSITY.

Так как каталог UNIVERSITY является надкаталогом текущего каталога FACULTY, то выполните следующие действия:

- ♦ с помощью стрелок выделите строку «. .»;
- ♦ нажмите клавишу [Enter].

В результате каталог UNIVERSITY становится текущим.

Удаление каталога

Как уже говорилось выше, в любом каталоге могут находиться файлы и каталоги более низкого уровня.

FAR Manager позволяет удалять как *полные* (содержащие файлы и подкаталоги), так и *пустые* каталоги.

Для удаления каталога необходимо:

- ♦ выделить его на панели;
- ♦ нажать клавишу [F8] (на экране появляется запрос на подтверждение операции);
- ♦ чтобы удалить каталог, необходимо нажать клавишу [Enter].
Нажатие клавиши [Esc] отменяет удаление.

Удаление каталога является ответственной операцией, поэтому FAR Manager выдает на экран монитора дополнительное предупреждение (рис. Л-5.3). Удаление каталога подтверждается нажатием клавиши [Enter] и отменяется нажатием клавиши [Esc].

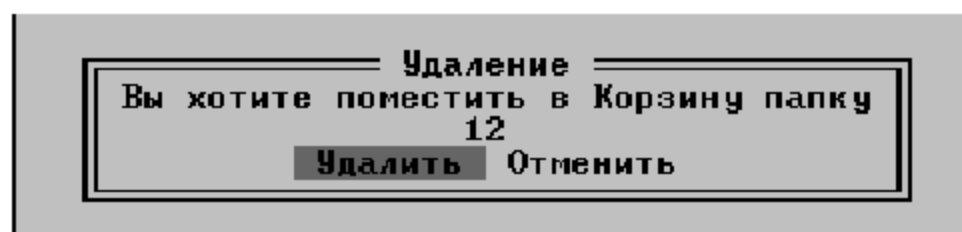


Рис. Л-5.3. Окно подтверждения операции удаления

Удалите ветвь каталогов \UNIVERSITY\FACULTY\CHAIR EVM. Для этого выполните действия:

- ♦ перейдите в подкаталог FACULTY, выделив на его панели строку «. .» и нажав клавишу [Enter];
- ♦ выделите каталог CHAIR EVM;
- ♦ нажмите клавишу [F8];
- ♦ подтвердите удаление каталога CHAIR EVM нажатием клавиши [Enter];
- ♦ перейдите в надкаталог FACULTY, выделив на панели строку «. .» и нажав клавишу [Enter];
- ♦ нажмите клавишу [F8];
- ♦ подтвердите удаление каталога FACULTY нажатием клавиши [Enter];

ОПЕРАЦИИ С ФАЙЛАМИ

Для работы с файлами в FAR Manager используют следующие клавиши:

[Shift] + [F4]	— создание нового файла;
[F4]	— редактирование файла;
[F3]	— просмотр файла;
[F5]	— копирование файла;
[F6]	— переименование файла;
[Alt] + [F7]	— поиск файла;
[F8]	— удаление файла;
[Insert]	— пометка файла.

В FAR Manager можно работать в двух режимах:

- ♦ редактирования;
- ♦ просмотра.

**Создание текстового файла
с помощью встроенного редактора**

Для создания *нового* текстового файла в текущем каталоге необходимо:

- ♦ нажать комбинацию клавиш [Shift] + [F4];
- ♦ в появившемся окне ввести имя создаваемого файла;
- ♦ нажать клавишу [Enter].

Задание 4

Создайте текстовый файл text.txt, для чего:

- ♦ нажмите комбинацию клавиш [Shift] + [F4];
- ♦ в появившемся окне введите имя файла text.txt;
- ♦ нажмите клавишу [Enter].

После выполнения этих действий начинает свою работу *текстовый редактор* FAR Manager (рис. Л-5.5). На экране монитора, вверху, расположена строка статуса, в которой отображаются:

- ♦ имя редактируемого файла;
- ♦ размер доступной памяти;
- ♦ код текущего символа;
- ♦ позиция курсора (мигающий символ «_» на экране);
- ♦ текущий режим набора символов — вставка (Ins) или замещение (Over) и т.д.



Рис. Л-5.5. Окно редактора FAR Manager

Изменение режима Ins/Over осуществляется нажатием клавиши [Insert]. Внизу располагается строка подсказки, напоминающая назначение функциональных клавиш, используемых в редакторе FAR Manager.

РЕДАКТОР FAR MANAGER

Редактор позволяет осуществлять ввод и редактирование текстовых файлов. Он обладает следующими возможностями:

- ◆ работа с фрагментами (блоками) текста;
- ◆ встроенный калькулятор;
- ◆ поиск и замена необходимого фрагмента текста и т.д.

Символы набираемого на клавиатуре текста появляются в той позиции, на которую показывает курсор. Для перехода на новую строку нажимается клавиша [Enter].

Клавиши переключения алфавита (русский/латинский) зависят от программного обеспечения:

- ◆ [Ctrl] (чаще всего);
- ◆ правая и левая клавиши [Shift];
- ◆ [Alt] и т.д.

Для удаления символов и строк можно использовать клавиши:

[Del]	— удаление символа, на который указывает курсор;
[←]	— удаление символа слева от курсора;
[Ctrl] + [Y]	— удаление строки;
[Ctrl] + [K]	— удаление текста от текущего положения курсора до конца строки.

Курсор перемещается с помощью клавиш перемещения курсора.

Кроме того, его можно перемещать по тексту с помощью клавиш быстрого перемещения:

[PgUp] и [PgDn]	— на страницу (экран) вверх и вниз;
[Home] и [End]	— на начало и конец редактируемой строки;
[Ctrl] + [Home]	— на начало редактируемого файла;
[Ctrl] + [End]	— на конец редактируемого файла.

При редактировании файла можно использовать следующие функциональные клавиши:

[F1]	— вывод на экран справки (выход из режима по нажатию клавиши [Esc]);
[F2]	— сохранить редактируемый файл под собственным именем;
[Shift] + [F2]	— сохранить отредактированный файл под другим именем;
[Shift] + [F10]	— сохранить отредактированный файл и выйти из редактора;
[F7]	— поиск текста в редактируемом файле.

Выход из редактора осуществляется нажатием клавиши [F10] или [Esc]. При этом если файл не был сохранен, то на экране появляется меню-предупреждение (рис. Л-5.6).

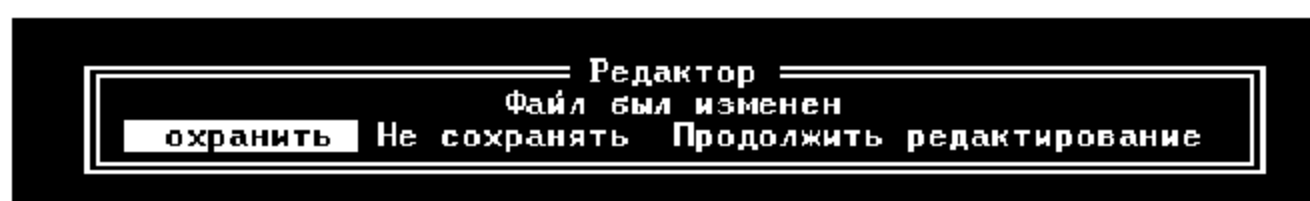


Рис. Л-5.6. Меню-предупреждение

Сохранить	— выход из редактора с сохранением сделанных изменений;
Не сохранять	— выход из редактора без сохранения сделанных изменений;
Продолжить редактирование	— отмена выхода.

С помощью клавиш управления курсором выберите нужный пункт и нажмите клавишу [Enter].

Наберите следующий текст:

Редактор FAR Manager позволяет осуществлять ввод и редактирование текстовых файлов. Он обладает следующими возможностями:

- ◆ работа с фрагментами;
- ◆ встроенный калькулятор;
- ◆ поиск и замена необходимого фрагмента текста.

После набора текста выполните следующие действия:

- ◆ сохраните набранный текст, нажав клавишу [F2];
- ◆ нажмите клавишу [F10] для *выхода из редактора*.

Редактирование существующего файла

Для редактирования *существующего* файла необходимо:

- ◆ выделить его на панели;
- ◆ нажать клавишу [F4] (для *входа* в режим редактирования).

После выполнения этих действий происходит переход в *режим редактирования*.

Отредактируйте файл text.txt, изменив последнюю строку на следующую:

Редактор FAR Manager позволяет . . .

.
. .
.

- ◆ поиск и замена подстроки в тексте.

Для сохранения сделанных изменений в файле text.txt выполните следующие действия:

- ◆ нажмите клавишу [Esc];
- ◆ в появившемся меню с помощью клавиш управления курсором выберите пункт *Сохранить*;
- ◆ нажмите [Enter].

При просмотре и редактировании файлов можно изменять кодировку отображения текста с помощью нажатия клавиши [F8]. При выполнении операции сохранения файл будет сохранен в последней выбранной кодировке.

Просмотр текстового файла

Для просмотра текстового файла необходимо:

- ◆ выделить файл на панели;
- ◆ нажать клавишу [F3].

После выполнения этих действий происходит переход в режим *просмотра файла*. Просмотр файла осуществляется с помощью клавиш управления курсором, а также клавишами быстрого перемещения:

[PgUp] и [PgDn] — на страницу (экран) вверх и вниз;
[Home] и [End] — перемещение к началу или концу файла.

При просмотре файла, как и в режиме редактирования, можно осуществлять поиск требуемого фрагмента текста. Для этого необходимо:

- ◆ нажать клавишу [F7];
- ◆ в появившемся окне набрать фрагмент текста для поиска;
- ◆ нажать клавишу [Enter].

Выход из режима просмотра происходит при нажатии клавиши [F10] или [Esc].

Просмотрите файл text.txt, выполнив действия:

- ♦ выделите файл text.txt на панели;
- ♦ нажмите клавишу [F3];
- ♦ выйдите из режима просмотра, нажав клавишу [Esc].

Копирование файлов

Для копирования файла необходимо:

- ♦ на активной панели (источник) установить исходный каталог и выделить копируемый файл;
- ♦ на другой панели (приемник) установить каталог, в который будет произведено копирование;
- ♦ нажать клавишу [F5] (на экране появляется запрос на подтверждение копирования);
- ♦ для осуществления копирования нажмите клавишу [Enter], для отмены — клавишу [Esc].

При копировании файла на экран выводится полоска-диаграмма, наглядно изображающая процесс копирования (рис. Л-5.7).

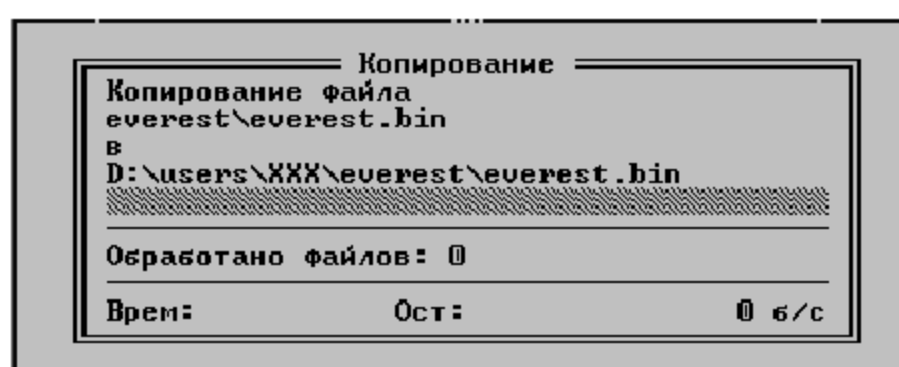


Рис. Л-5.7. Полоска-диаграмма

Внимание! На современных компьютерах при копировании файлов малого размера она появляется и исчезает слишком быстро.

Скопируйте файл text.txt в каталог FACULTY, выполнив действия:

- ♦ нажмите комбинацию клавиш [Alt] + [F2] (правая панель). В появившемся окне выберите диск D и нажмите [Enter];
- ♦ перейдите на правую панель, нажав клавишу [Tab];
- ♦ перейдите в каталог FACULTY, выделив строку «..» и нажав [Enter];
- ♦ нажмите [Tab] для перехода на левую панель;
- ♦ выделите файл text.txt и нажмите [F5];
- ♦ подтвердите копирование нажатием клавиши [Enter].

Скопируйте файл text.txt из каталога FACULTY в каталог UNIVERSITY, выполнив действия:

- ♦ в левой панели перейдите в каталог FACULTY, выделив строку «. .» и нажав клавишу [Enter];
- ♦ перейдите в каталог UNIVERSITY, выделив еще раз строку «. .» и нажав клавишу [Enter];
- ♦ перейдите на правую панель, нажав клавишу [Tab];
- ♦ выделите файл text.txt, находящийся в каталоге FACULTY, и нажмите клавишу [F5];
- ♦ подтвердите копирование нажатием клавиши [Enter].

Перенос файлов

Перенос файлов осуществляется аналогично копированию файлов, но вместо клавиши [F5] используется клавиша [F6]. При этом надо помнить о том, что исходный файл удаляется из текущего каталога (смысл операции переноса).

Поиск файлов на диске

Для быстрого поиска файла во всех каталогах на текущем диске надо выполнить следующие действия:

- ♦ нажать комбинацию клавиш [Alt] + [F7];
- ♦ в поле *Одна или несколько масок файлов* набрать название файла, который требуется найти;
- ♦ с помощью клавиш управления курсором выделить пункт *Искать*;
- ♦ нажать клавишу [Enter]; поиск файла начнется.

Если вы забыли имя или расширение файла, то при наборе названия файла можно использовать символ *, который заменяет собой любое число символов в имени или расширении файла, например:

*.txt — найти все файлы с расширением .txt;

doc.* — найти все файлы с именем doc и любым расширением.

Если вы забыли и имя, и расширение нужного файла, то можно искать по содержанию. Для этого необходимо:

- ♦ нажать комбинацию клавиш [Alt] + [F7];
- ♦ в поле *Одна или несколько масок файлов* набрать символы «*.*», означающие поиск любого файла;
- ♦ в поле *Содержащий текст* набрать любую уникальную информацию из нужного файла (например, какое-нибудь слово);
- ♦ с помощью клавиш управления курсором выделить пункт *Искать*;
- ♦ нажать клавишу [Enter]; поиск файла начнется.

По окончании поиска на экран выводится список найденных файлов. Клавишами перемещения курсора можно выделить нужный файл (если найдено более одного файла) и, нажав клавишу [Enter], перейти в каталог, где находится этот файл.

Если нужный файл не был найден, нажмите [Esc] для выхода из режима поиска файла.

Удаление файлов

Для удаления файла необходимо:

- ◆ выделить данный файл на панели;
- ◆ нажать клавишу [F8];
- ◆ в появившемся запросе (рис. Л-5.8) выделить пункт *Удалить* для удаления выбранного файла либо *Отменить* для отмены операции;
- ◆ нажать клавишу [Enter].

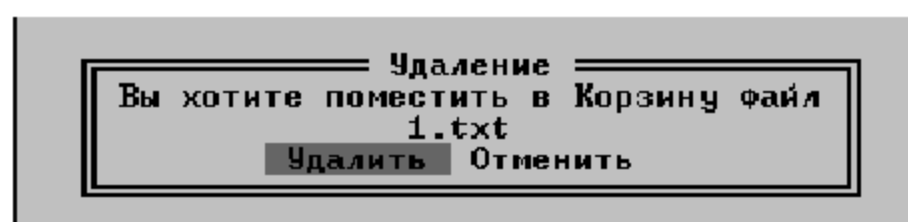


Рис. Л-5.8. Окно запроса на удаление

Удалите файл text.txt из каталога FACULTY, выполнив действия:

- ◆ убедитесь, что текущим является каталог FACULTY;
- ◆ выделите указателем файл text.txt и нажмите клавишу [F8];
- ◆ подтвердите удаление нажатием клавиши [Enter].

Удалите файл text.txt из каталога UNIVERSITY:

- ◆ перейдите в каталог UNIVERSITY, выделив на панели строку «. .» и нажав клавишу [Enter]. Убедитесь, что текущим стал каталог UNIVERSITY;
- ◆ выделите указателем файл text.txt и нажмите клавишу [F8];
- ◆ подтвердите удаление нажатием клавиши [Enter].

Операции над группой файлов

Для проведения операции копирования, перемещения и удаления одновременно нескольких файлов необходимо пометить каждый файл, выделив его на панели и нажав клавишу [Insert] (помеченные файлы выделяются цветом). Для снятия пометки необходимо выделить файл и повторно нажать клавишу [Insert]. Далее требуемая операция осуществляется, как описано выше.

РАБОТА В СЕТИ

FAR Manager поддерживает работу пользователя в сети двумя способами:

- 1) через *Сетевое окружение* — доступ к другим компьютерам в сети и их разделяемым сетевым ресурсам;
- 2) через доступ к ftp-серверам сети.

Доступ к сетевым ресурсам

Чтобы получить доступ к сетевым ресурсам, необходимо:

- ♦ одновременно нажать клавиши [Alt] и [F1] (левая панель) или [Alt] и [F2] (правая панель);
- ♦ в появившемся окне с помощью клавиш управления курсором выбрать пункт *Сеть* и нажать клавишу [Enter].

Работа с каталогами и файлами, открытыми для доступа в сети, выполняется аналогично работе с каталогами и файлами на дисках. Однако чтобы *не потерять данные*, рекомендуется копировать файлы, с которыми требуется произвести действия, в свой *локальный каталог*, т.е. работать с ними на диске.

Имена рабочих групп сети, а также имена компьютеров сети, отображаются как имена каталогов. Верхним уровнем дерева каталогов сети является (для данной лабораторной работы) Microsoft Windows Network. При попытке перейти на более высокий (..) уровень FAR Manager отобразит на панели последний каталог, открытый на панели пользователем на диске.

Скопируйте файл *informnet.txt*, находящийся на компьютере с именем WS208- (WS210-), в свой рабочий каталог, выполнив действия:

- ♦ перейдите в левой панели в *network*;
- ♦ найдите в дереве сетевую группу *EVM* и войдите в нее;
- ♦ найдите компьютер с именем *WS208-* и войдите в него;
- ♦ выберите сетевой каталог *Incoming* и войдите в него;
- ♦ откройте каталог *INFORM*;
- ♦ скопируйте из него в свой рабочий каталог файл *informnet.txt* и перепишите в отчет его содержимое.

Работа с FTP-ресурсами

Работа с ftp-ресурсами выполняется так же, как и с сетевым окружением за исключением того, что в меню выбирается вкладка *FTP*, которая открывает на панели список ftp-серверов.

FTP-сервер — сервер сети, предоставляющий пользователям возможность доступа к ресурсам сети через имена (адреса) в *internet-стандарте*.

Для ввода существующего адреса ftp и перехода к нему в FAR Manager необходимо:

- ◆ перейти в *FTP*;
- ◆ нажать комбинацию клавиш [Shift] + [F4];
- ◆ ввести в поле `ftp://user:password@host:port/dir` адрес `ftp.evm.rgrta.edu` (рис. Л-5.9) и нажать клавишу [Enter].

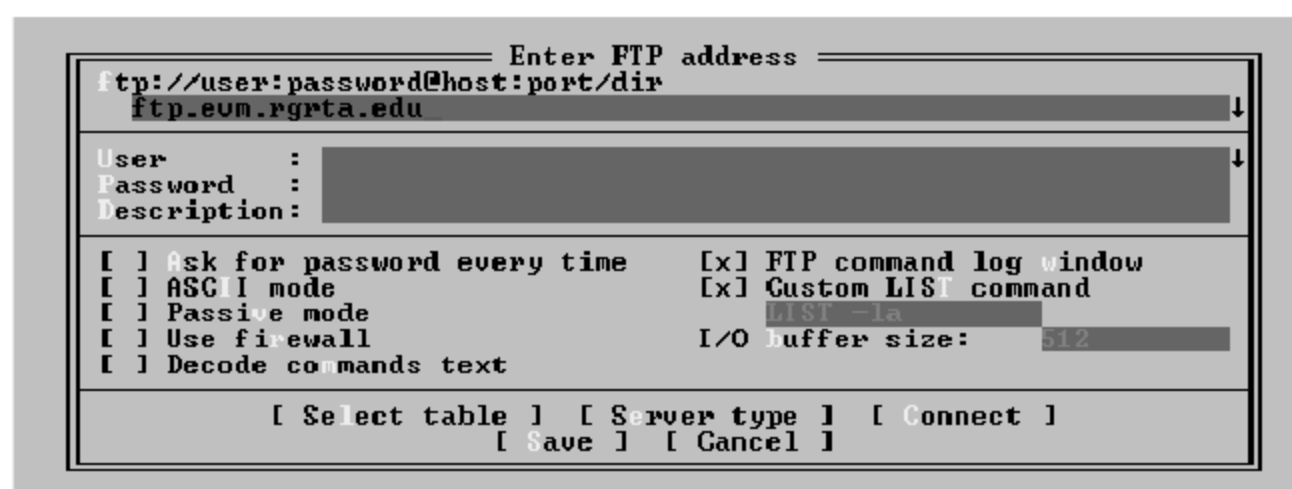


Рис. Л-5.9. Окно ввода существующего адреса ftp

Скопируйте файл `informftp.txt`, находящийся на ftp-сервере с адресом `ftp.evm.rgrta.edu`, в свой рабочий каталог, выполнив действия:

- ◆ перейдите в левой панели в *FTP*;
- ◆ просмотрите список доступных ftp-серверов и найдите в нем `ftp.evm.rgrta.edu`;
- ◆ если `ftp.evm.rgrta.edu` в списке отсутствует, то введите этот адрес;
- ◆ выберите адрес `ftp.evm.rgrta.edu` и войдите в него;
- ◆ выберите каталог `labrab` и войдите в него;
- ◆ откройте каталог `INFORM`;
- ◆ скопируйте из него в свой рабочий каталог файл `informftp.txt` и перепишите в отчет его содержимое.

Контрольные вопросы

1. Что такое операционная оболочка?
2. Перечислите основные функции FAR Manager.
3. Что такое рабочая панель FAR Manager?
4. Что такое команда в FAR Manager? Назовите способы ее задания.
5. Перечислите команды для работы с каталогами и файлами.
6. Перечислите комбинации клавиш для управления панелями.
7. Работа на ПК в FAR Manager.

8. Что происходит при перемещении файла? В каких случаях при перемещении файла надо проверять свободное место на диске?
9. Что происходит при удалении файла?
10. Для каких целей можно использовать выделение группы файлов?
11. Как с помощью клавиатуры вызвать в командную строку предыдущую команду, выданную вами?
12. Вы выделили маркером некий файл и нажали клавишу [F3]. На экране появилась «абракадабра». Что это означает?

Лабораторная работа 6

ЗАЩИТА ОТ КОМПЬЮТЕРНЫХ ВИРУСОВ

6.1. Цель работы

1. Изучение компьютерных вирусов и методов борьбы с ними.
2. Работа с антивирусной программой Kaspersky Anti-Virus Scanner.

6.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [5, 7, 9].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (цель работы, основные сведения о вирусах и методах борьбы с ними, порядок работы с программой Kaspersky Anti-Virus Scanner).
4. Работа на компьютере:
работа с антивирусной программой Kaspersky Anti-Virus Scanner: проверка папки D:\USERS* (номер группы) на наличие вирусов.
5. Оформление отчета (дорабатывается по результатам работы с программой Kaspersky Anti-Virus Scanner).
6. Защита отчета.

6.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ♦ понятие о компьютерном вирусе, признаки заражения компьютерным вирусом, действия при заражении компьютерным вирусом;
- ♦ классификация вирусов и их характеристика, методы борьбы с вирусами;

- ♦ виды антивирусных программ и их характеристика;
- ♦ работа с программой Kaspersky Anti-Virus Scanner. Назначение основных компонентов (главное меню, окно тестирования), команды и доступ к ним.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

проверка папки D:\USERS* на наличие вирусов:

- ♦ перечень выполняемых операций и их смысл;
- ♦ назначение устанавливаемых опций диалогового окна *Параметры*;
- ♦ отчет о ходе проверки папки на наличие вирусов. Назначение сообщений, выдаваемых программой Kaspersky Anti-Virus Scanner в ходе проверки.

6.4. Общие сведения о вирусах

ЧТО ТАКОЕ ВИРУСЫ И ПРИЗНАКИ ЗАРАЖЕНИЯ КОМПЬЮТЕРА

Компьютерный вирус — это специально написанная небольшая по размерам программа, которая может «приписывать» себя к другим программам, т.е. «заражать» их, а также изменять или удалять файлы, проигрывать мелодии, выводить на экран различные видеоэффекты. Программа, внутри которой находится вирус, называется «зараженной», или «инфицированной». При запуске такой программы управление получает вирус, который сначала выполняет заложенные в него действия, а затем запускает саму программу.

Существует множество разновидностей вирусов, и проявляют они себя по-разному — одни из них после определенного количества успешных заражений совершают деструктивные действия, начиная от повреждения различных файлов и заканчивая уничтожением информации на всем диске, другие — приурочивают свою деятельность к определенным датам (1 января, 8 марта, пятница 13-го и т.п.).

Факт заражения компьютера вирусом можно установить по наличию следующих признаков:

- ♦ воспроизведение компьютером различных звуковых и видеоэффектов;
- ♦ свечение индикатора дисковода, когда к нему нет обращения;
- ♦ увеличение размеров исполняемых файлов (exe, com, sys, dll, ovl, bin, drv и т.д.);
- ♦ наличие файлов с некорректным временем создания (например, 10 ч 25 мин 62 с);

- ♦ существенное уменьшение объема свободной оперативной памяти;
- ♦ неуместное появление системных сообщений;
- ♦ появление на диске зарегистрированных дефектных кластеров;
- ♦ постоянное увеличение количества файлов на диске;
- ♦ неожиданное зависание компьютера с невозможностью перезагрузки;
- ♦ разрушение файловой структуры.

**КЛАССИФИКАЦИЯ ВИРУСОВ.
ХАРАКТЕРИСТИКА КЛАССОВ**

Вирусы можно разделить на классы по трем признакам (рис. Л-6.1):

- ♦ среда обитания и способ распространения;
- ♦ способ заражения;
- ♦ деструктивные возможности.

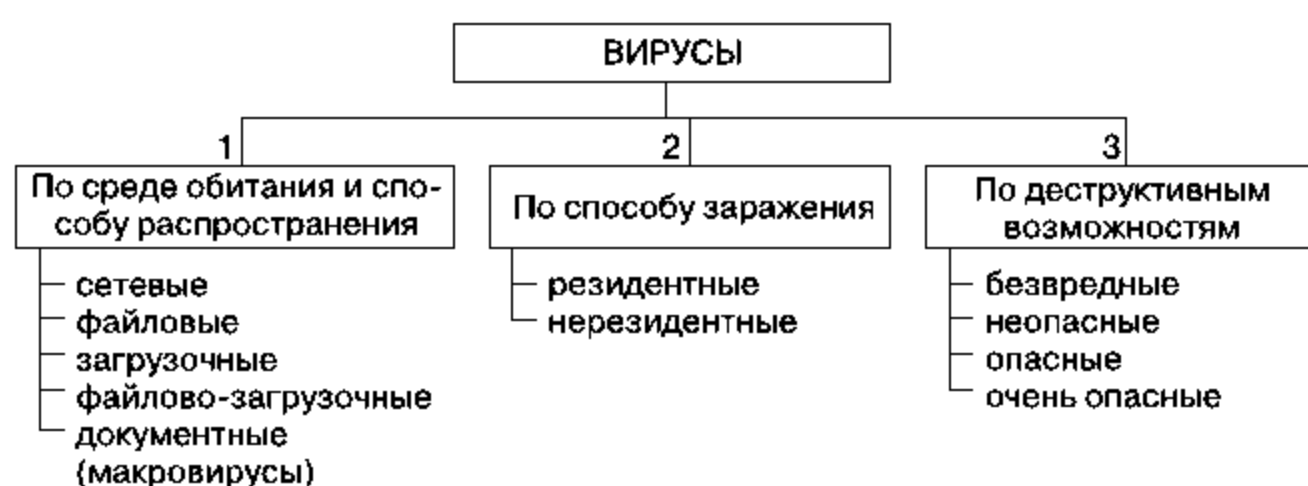


Рис. Л-6.1. Классификация вирусов

Вирусы 1-го класса:

- ♦ *сетевые* — распространяются по компьютерным сетям, файлы и диски не заражают;
- ♦ *файловые* — внедряются в выполняемые файлы;
- ♦ *загрузочные* — внедряются в область диска, используемую при загрузке операционной системы;
- ♦ *файлово-загрузочные* — комбинация предыдущих двух видов;
- ♦ *макровирусы* — наиболее многочисленная разновидность вирусов; внедряются в документы MS Word (например, рис. Л-6.2), MS Excel, MS PowerPoint благодаря наличию в этих программах встроенных языков VisualBasic и WordBasic, получают управление при открытии или сохранении зараженного документа, при запуске или выходе из соответствующего приложения.

Использование электронных таблиц позволяет осуществить обмен информацией между сотрудниками предприятия. Представление данных подразумевает подведение некоторых важных показателей. MS Excel позволяет быстро составлять сводные отчеты с помощью нескольких способов определения итогов и соединения данных.



Рис. Л-6.2. Результат действия вируса макрос W97.THUS на текстовый документ

Комбинированные сетевые макровирусы не только заражают создаваемые документы, но и рассылают копии этих документов по электронной почте (печально известный вирус «I love you»).

Вирусы 2-го класса:

- ♦ *резидентные* — постоянно находятся в оперативной памяти и заражают запускаемые в это время программы;
- ♦ *нерезидентные* — активны только в момент запуска зараженной программы.

Вирусы 3-го класса:

- ♦ *безвредные* — приводят лишь к уменьшению свободной памяти на диске;
- ♦ *неопасные* — ограничиваются графическими, звуковыми и прочими эффектами;
- ♦ *опасные* — могут привести к серьезным сбоям в работе компьютера;
- ♦ *очень опасные* — могут привести к потере программ, уничтожить данные, стереть необходимую для работы компьютера информацию.

Вирус, как и всякая программа, может иметь ошибки, вследствие чего невозможно предсказать поведение даже самого безвредного вируса при работе с Windows или другими системами. Соблюдая меры профилактики и имея под рукой необходимый набор антивирусных средств, можно вовремя обнаружить и обезвредить любой вирус.

МЕТОДЫ БОРЬБЫ С КОМПЬЮТЕРНЫМИ ВИРУСАМИ

Для предотвращения заражения вирусами используют такие профилактические меры, как:

- ♦ создание архивных копий всех важных материалов;
- ♦ проверка любых новых программ, компакт-дисков и дискет с помощью антивирусов;

- ♦ ограничение доступа по записи к внешним носителям;
- ♦ использование лицензионных программ.

Для более полной проверки компьютера желательно применять несколько различных антивирусных программ, которые подразделяются:

- ♦ на *программы-сканеры* — проверяют файлы на наличие в них уникальной последовательности байтов (*сигнатуры*), принадлежавшей одному из известных ранее вирусов, а также проводят так называемый *эвристический анализ*, т.е. поиск в теле программы действий, характерных для вирусов;
- ♦ *программы-доктора* — лечат зараженные файлы, вырезая из них тело вируса.

Часто антивирусные программы объединяют в себе функции детектора и доктора:

- ♦ *программы-мониторы* — работают непрерывно с момента включения компьютера, отслеживают нестандартные действия программ и сообщают об этом пользователю. Пользователь может разрешить или запретить выполнение соответствующей операции;
- ♦ *программы-блокировщики* — небольшие резидентные программы, предназначенные для предотвращения распространения одного конкретного вируса.

На сегодня самыми распространенными антивирусными программами являются Kaspersky Anti-Virus, Norton AntiVirus, AVP и DrWeb, которые выполняют функции детектора и доктора. Они могут находить и уничтожать десятки тысяч различных вирусов.

Всегда желательно иметь наиболее свежие версии антивирусных программ, так как они позволяют обнаруживать большее число вирусов.

6.5. Описание программы Kaspersky Anti-Virus Scanner

ДОСТОИНСТВА KASPERSKY ANTI-VIRUS SCANNER

Программа Kaspersky Anti-Virus Scanner, входящая в антивирусный пакет программ Kaspersky Anti-Virus, обрела широкую известность благодаря таким достоинствам, как:

- ♦ постоянное обновление, т.е. наличие новых версий, способных находить и уничтожать только что появившиеся вирусы;
- ♦ наличие эффективного эвристического анализатора, который позволяет выявлять вирусы, еще не известные авторам программы;
- ♦ возможность лечения почтовых баз;

- ♦ удобный пользовательский интерфейс;
- ♦ возможность работы как в режиме командной строки, так и в диалоговом режиме. В диалоговом режиме появляется возможность получения оперативной помощи, что значительно упрощает работу.

РАБОТА В ДИАЛОГОВОМ РЕЖИМЕ

Программа Kaspersky AV Scanner при запуске выводит на экран следующие элементы (рис. Л-6.3):

- ♦ *главное меню* — позволяет пользователю изменить настройки сканирования, запустить/остановить сканирование, просмотреть результаты и т.д.
- ♦ *панель инструментов* — содержит кнопки, нажатие которых позволяет пользователю инициировать те или иные действия;
- ♦ *рабочая область*;
- ♦ *строка состояния* — содержит отчет о ходе проверки памяти и файлов на наличие вирусов.

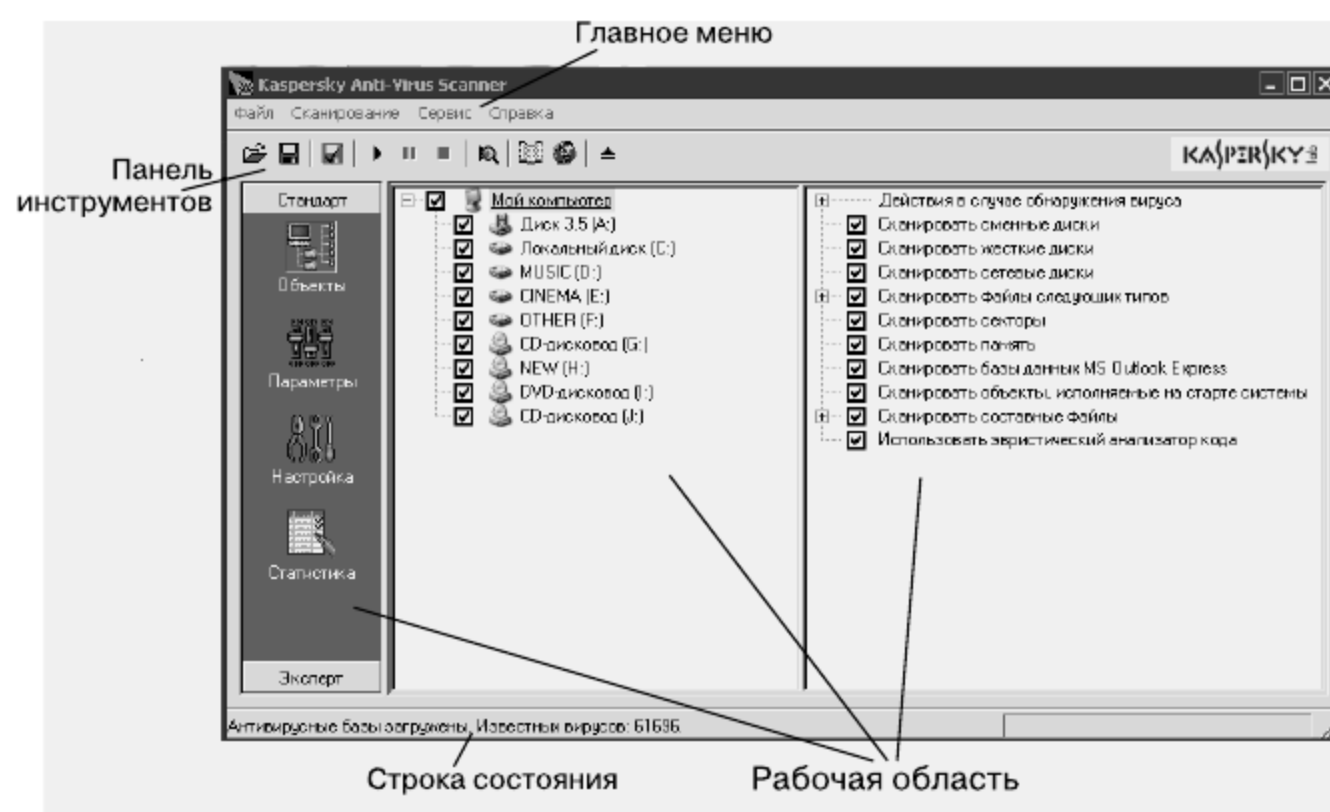


Рис. Л-6.3. Главное окно программы Kaspersky AV Scanner

Меню программы Kaspersky AV Scanner позволяет выполнять различные операции, а каждый пункт меню (*Файл*, *Сканирование*, *Сервис*, *Справка*) дает пользователю доступ к ряду команд, сгруппированных по функциональному назначению (табл. Л-6.1).

Таблица Л-6.1

Группы команд меню программы Kaspersky AV Scanner

Функция	Команда	Назначение
Файл	Загрузить профиль	Загрузить набор параметров из файла настройки
	Сохранить профиль	Сохранить набор параметров в файле настройки
	Сохранить профиль как	Сохранить набор параметров в файле настройки с другим именем
	Сохранить профиль по умолчанию	Назначить набор параметров, загружаемых по умолчанию
	Последние загруженные профили	Выбрать загружаемый файл настроек из перечня файлов, с которыми недавно работали
	Выгрузить Kaspersky Anti-Virus Scanner	Выгрузить программы Kaspersky AV Scanner из памяти
	Выход	Заккрыть главное окно программы
Сканирование	Начать сканирование/Остановить сканирование	Запустить/Остановить процесс сканирования
	Приостановить сканирование/Продолжить сканирование	Приостановить/Возобновить процесс сканирования
	Изменить приоритет процесса сканирования	Изменить приоритет процесса сканирования
	Просмотреть параметры сканирования	Показать параметры в виде последовательного текста
Сервис	Обновить антивирусные базы	Обновить антивирусные базы данных
	Показать отчет	Показать окно с отчетом
	Создать список вирусов	Сгенерировать список всех известных на настоящий момент вирусов
Справка	Содержание	Вызвать справочную систему
	Антивирус Касперского в Web	Открыть окно браузера с информацией о лаборатории Касперского
	О программе	Показать краткие сведения о программе

Для выбора одного из пунктов меню можно использовать мышь или:

- ◆ нажать клавишу [F10];
- ◆ с помощью курсорных клавиш подвести указатель к нужному пункту;
- ◆ нажать клавишу [Enter].

Рабочая область главного окна состоит из двух частей. В левой части находится список категорий и соответствующих им значков. В правой части отображается содержимое категорий. Существует четыре категории: *Объекты*, *Параметры*, *Настройка* и *Статистика*.

Категория *Объекты* позволяет задать область сканирования (перечень дисков и проверяемых папок), подлежащие сканированию объекты и правила обработки инфицированных файлов. Все эти настройки организованы в специализированный элемент управления — *дерево настроек иерархии объектов*.

Категория *Параметры* позволяет задать общие настройки, а категория *Настройка* — специальные настройки программы с помощью обычного дерева настроек.

Категория *Статистика* позволяет просматривать результаты работы программы в таблице.

Для выбора одной из категорий необходимо щелкнуть мышью по соответствующему значку.

НАСТРОЙКА ПАРАМЕТРОВ СКАНИРОВАНИЯ

Категория *Объекты* рабочей области служит для выбора области сканирования и объектов, подлежащих сканированию. Выбор области и объектов осуществляется с помощью дерева настроек, которое представляет собой две панели: левая — список дисков компьютера и правая — дерево настроек выбранного в левом списке объекта.

Область проверки задается в левой панели, которая содержит иерархию файловой системы компьютера вместе с индикаторами проверки (переключателями) каждой из областей файловой системы. Индикатор проверки может быть включен: ☒, что соответствует указанию проверить выбранную область, либо выключен: ☐, что соответствует пропуску выбранной области.

Чтобы программа сканировала область файловой системы, необходимо включить расположенный слева от ее названия индикатор проверки. Для изменения состояния индикатора следует щелкнуть по нему левой кнопкой мыши.

Чтобы выбрать для сканирования группу дисков, в панели с иерархией файловой системы следует щелкнуть мышью по строке *Мой компьютер* и включить в правой панели с деревом настроек нужный переключатель:

- ☒ Сканировать сменные диски.
- ☒ Сканировать жесткие диски.
- ☒ Сканировать сетевые диски.

Области разного уровня иерархии файловой системы компьютера имеют разный набор настроек. Область *Мой компьютер* имеет дерево настроек с максимальным количеством настроек. В список настроек включены параметры сканирования оперативной памяти, загрузочных секторов, сканирования групп дисков и т.д. *Параметры области диска* позволяют только сканировать загрузочные сектора этого диска, а также включать или отключать проверку файловой системы диска. *Параметры области папки* не позволяют отключить проверку файловой системы. Но в состав параметров папок, как и всех вышележащих областей, входит выбор действий с зараженными и подозрительными объектами, выбор типа сканируемых файлов и включение дополнительных режимов сканирования.

ПАРАМЕТРЫ СКАНИРОВАНИЯ

- ☒ Действия в случае обнаружения вируса (*Спросить пользователя, Только отчет, Лечить, Переименовывать объект, Удалять объект*).
- ☒ Сканировать файлы следующих типов (*Все, что могут быть заражены; Все файлы; Файлы по маске; Исключать по маске*).
- ☒ Сканировать сектора — тестирование загрузочного сектора диска для проверки на наличие загрузочных или файлово-загрузочных вирусов.
- ☒ Сканировать память — проверка оперативной памяти компьютера на наличие резидентных вирусов.
- ☒ Сканировать базы данных MS Outlook Express.
- ☒ Сканировать объекты, исполняемые на старте системы, — сканирование объектов, которые операционная система исполняет автоматически сразу после загрузки.
- ☒ Сканировать составные файлы (*Архивы; Самораспаковывающиеся архивы; Почтовые базы данных; Почтовые текстовые форматы; Вложенные объекты*).
- ☒ Использовать эвристический анализатор кода — проверка файлов и загрузочных областей дисков на наличие неизвестных компьютерных вирусов.

6.6. Практическая часть

ПРОВЕРКА ПАПКИ D:\USERS* НА НАЛИЧИЕ ВИРУСОВ (РАБОТА С ПРОГРАММОЙ KASPERSKY AV SCANNER)

Проверка папки на наличие вирусов предусматривает выполнение одних и тех же действий на каждой машине, но поскольку папки содержат файлы, которые могут быть заражены разными вирусами, реакция программы Kaspersky AV Scanner будет различной.

Для проверки папки необходимо выполнить следующие действия.

1. Запустить программу Kaspersky AV Scanner в диалоговом режиме, для чего необходимо щелкнуть левой кнопкой мыши по кнопке *Пуск*, выбрать подменю *Программы*, далее выбрать группу *Kaspersky Anti-Virus* и в появившемся меню щелкнуть мышью по строке *Kaspersky Anti-Virus Scanner*.

На экране появится главное окно программы, а также значок  на панели задач, щелкнув правой кнопкой мыши по которому, можно вызвать системное меню.

Далее программа Kaspersky AV Scanner производит загрузку в память известные ей антивирусные базы и проверяет себя на наличие вирусов, что подтверждается выводом соответствующего сообщения в строке состояния:

Антивирусные базы загружены. Известных вирусов: 61696

2. Выбрать категорию *Объекты*. В левой панели рабочей области выключить индикатор *Мой компьютер* (при этом отключатся индикаторы всех дисков). Перейти в режим работы *Эксперт*, выполнив щелчок по соответствующей надписи, расположенной в левой части рабочей области. После этого в правой части открыть содержимое диска D щелчком по символу «+», затем найти и таким же образом открыть папку *USERS*, в которой необходимо найти каталог с названием * (номер группы) и включить соответствующий ему индикатор. В правой панели рабочей области включить опции: *Спросить пользователя*, *Сканировать файлы следующих типов (Все файлы)*, *Сканировать составные файлы (Архивы, Самораспаковывающиеся архивы)*, *Использовать эвристический анализ кода*.

3. Щелкнуть мышью по кнопке , которая находится на панели инструментов программы, или в пункте *Сканирование* главного меню выбрать команду *Начать сканирование*.

После этого программа Kaspersky AV Scanner начинает проверку папки. В процессе сканирования файлов, находящихся в папке, при обнаружении вируса на экране появится диалоговое окно, оповещающее об этом (рис. Л-6.4).



Рис. Л-6.4. Диалоговое окно запроса действий при обнаружении зараженного объекта

Это означает, что программа Kaspersky AV Scanner нашла объект, инфицированный вирусом с условным названием *EICAR-Test-File*.

4. Выбрать действие, которое будет совершено над зараженным файлом — *Удалять объект*. Для этого необходимо включить соответствующий индикатор. Таким же образом включить опцию *Применить ко всем инфицированным объектам*. Далее щелкнуть мышью по кнопке *OK*.

После окончания процесса проверки папки в правой части рабочей области главного окна программы должна появиться статистика выполнения процесса тестирования (рис. Л-6.5).

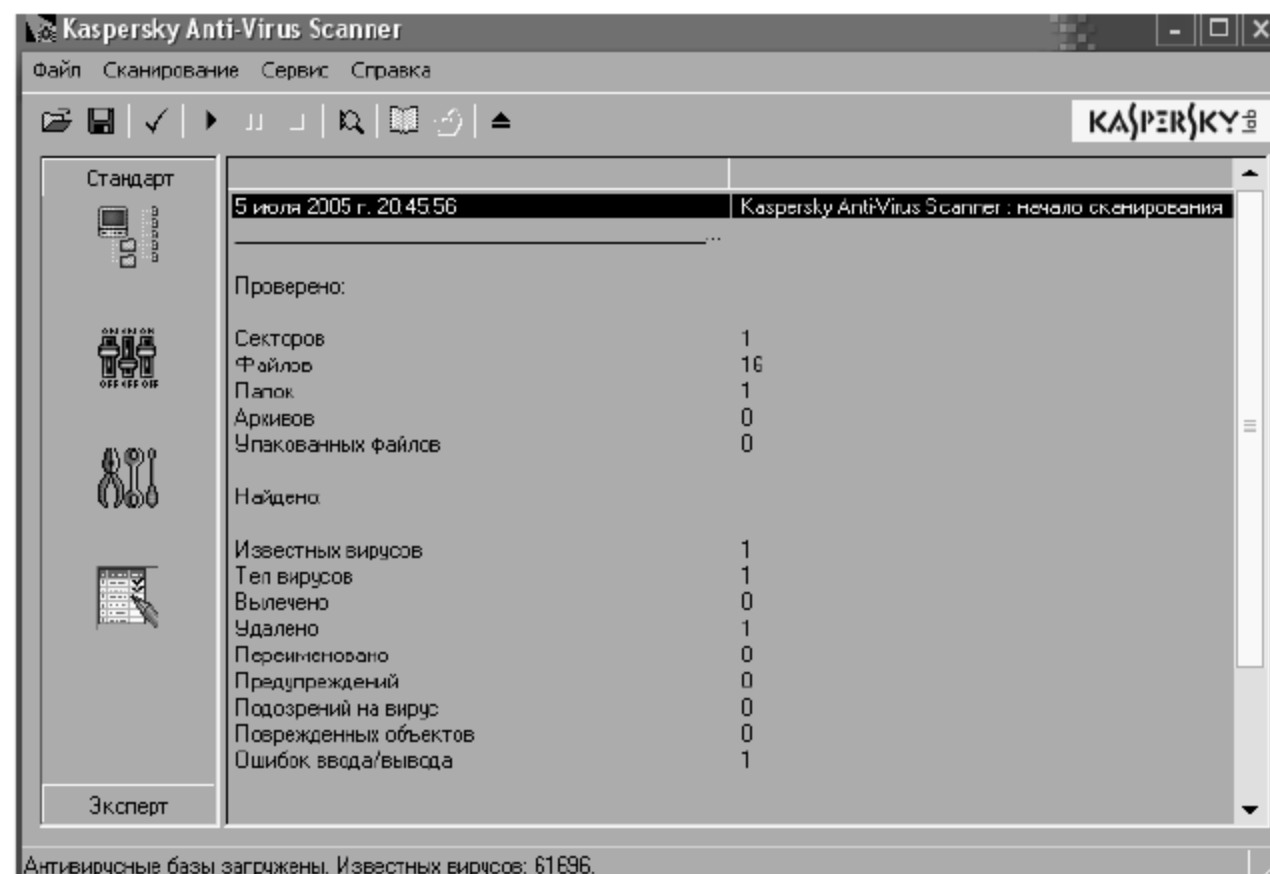


Рис. Л-6.5. Категория *Статистика*

Это означает, что программа Kaspersky AV Scanner нашла в файлах, находящихся в папке, одно тело вируса и удалила зараженный объект.

5. Для выхода из программы Kaspersky AV Scanner необходимо в подменю *Файл* выбрать команду *Выгрузить Kaspersky Anti-Virus Scanner*.

Контрольные вопросы

1. Что такое компьютерный вирус?
2. Классификация компьютерных вирусов.
3. Профилактические меры защиты от компьютерных вирусов.
4. Виды антивирусных программ.
5. Что такое поиск вируса по сигнатуре и эвристический анализ?
6. Назовите признаки заражения компьютера вирусами.
7. Для чего нужна программа Kaspersky AV Scanner?
8. Какие параметры сканирования можно изменять в категории *Объекты* программы Kaspersky AV Scanner?
9. Что может делать программа Kaspersky AV Scanner с файлами, в которых обнаружен вирус?

Лабораторная работа 7 ТЕКСТОВЫЙ РЕДАКТОР MICROSOFT WORD 2002 (ОСНОВЫ РАБОТЫ)

7.1. Цель работы

1. Изучение основных операций редактора Microsoft Word.
2. Нарботка практических навыков по созданию и оформлению документов.

7.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 7, 8].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (основные сведения о текстовом редакторе MS Word, основные операции редактора, возможности среды редактирования MS Word, основные элементы и их назначение).
4. Практическое освоение основных операций редактора MS Word (см. задания в подразделе 7.8. Основные операции).

5. Выполнение индивидуального задания в соответствии с номером варианта (см. табл. Л-7.2). Исходный текст составляется дома при подготовке отчета. Фрагмент текста должен содержать 3 абзаца по 5—7 печатных строк и может касаться деятельности РГРТУ, кафедры, факультета, описания работы ЭВМ, программ и др.
6. Отчет о практической работе: индивидуальное выполнение на ПК операций по заданию преподавателя:
 - ◆ оформление отчета (с представлением отредактированного текста);
 - ◆ защита отчета.

7.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ основные сведения о редакторе MS Word (перечень основных операций с документами и их краткая характеристика);
- ◆ назначение элементов интерфейса;
- ◆ группы команд основного меню, перечень их операций и действий.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

Подробный протокол выполнения индивидуального задания:

- ◆ исходный текст;
- ◆ *перечень* основных операций над текстом (раскрыть);
- ◆ порядок выполнения операций;
- ◆ действия, необходимые для выполнения каждой операции;
- ◆ отредактированный текст (распечатку) вложить в отчет (при наличии в лаборатории принтера).

7.4. Общие сведения о текстовом редакторе Microsoft Word

Текстовый редактор — это программа, обеспечивающая пользователя ПК средствами создания, обработки, печати и хранения документов различной природы и степени сложности, включающих текстовые данные, таблицы, графики, изображения, формулы и т.д.

Microsoft Word представляет собой самый распространенный и один из наиболее мощных текстовых редакторов. Основными операциями редактора MS Word по работе с документами являются:

- ◆ набор (ввод) текста;
- ◆ редактирование содержимого документа (внесение каких-либо изменений в текст документа: изменение взаимного располо-

жения отдельных частей документа, поиск и замена символов, слов и участков текста);

- ♦ форматирование содержимого документа (оформление документа таким образом, чтобы он стал более наглядным, удобочитаемым, эстетичным и соответствовал правилам оформления документов данного типа).

Кроме того, MS Word обладает такими возможностями, как:

- ♦ одновременная работа с несколькими документами;
- ♦ проверка орфографии и грамматики;
- ♦ автоматическое форматирование документа;
- ♦ включение в документы таблиц, рисунков и математических формул;
- ♦ коллективная работа над большими документами и т.д.

В настоящее время доступны несколько версий редактора: MS Word 2000, MS Word 2002, MS Word XP, которые отличаются удобством работы и числом дополнительных возможностей по обработке документов.

7.5. Запуск текстового редактора MS Word

Для запуска редактора MS Word необходимо открыть главное меню, щелкнув левой кнопкой мыши по кнопке *Пуск* на панели задач Windows, в разделе *Программы* открывшегося меню выбрать пункт *Microsoft Word* и щелкнуть по нему левой кнопкой мыши.

7.6. Среда редактирования MS Word

ЭЛЕМЕНТЫ СРЕДЫ И ИХ НАЗНАЧЕНИЕ

Среда редактирования MS Word (рис. Л-7.1) содержит следующие элементы:

- ♦ строку заголовка документа;
- ♦ основное меню;
- ♦ панели инструментов;
- ♦ окно редактирования;
- ♦ горизонтальную и вертикальную линейки;
- ♦ полосы прокрутки документа;
- ♦ строку состояния.

Управление системой редактирования и непосредственно обработка документа реализуются с помощью специальных команд. *Команда* — это инструкция, требующая от текстового редактора выполнения определенных действий.

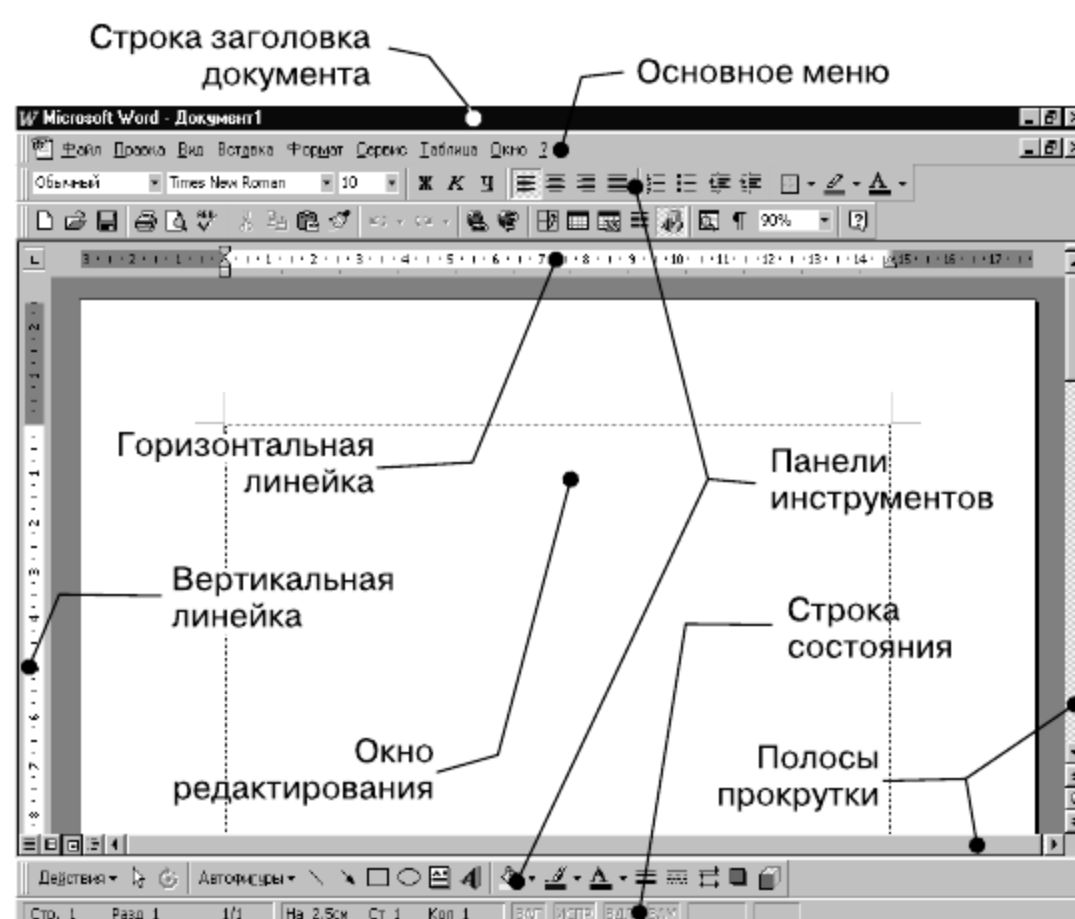


Рис. Л-7.1. Элементы среды редактирования MS Word

Действия могут быть объединены в группы, формируемые по типу выполняемых операций. Каждая такая группа имеет свое название (файл, правка, вид и т.д.), которое является именем отдельного пункта основного меню MS Word (табл. Л-7.1).

Таблица Л-7.1

Группы команд основного меню редактора MS Word

Команда	Действие
Файл	Операции над документом в целом: создание нового, открытие существующего, закрытие, просмотр, печать, установка параметров страницы
Правка	Операции редактирования: копирование, удаление, вставка текста или графики, команды поиска и замены фрагментов текста
Вид	Определение параметров изображения редактируемого документа: вид, масштаб, линейки, настройка панелей инструментов
Вставка	Использование дополнительных символов, специальных полей (номеров страниц, оглавления, названий, ссылок, указателей, сносок), графических, текстовых и специальных объектов
Формат	Выбор параметров шрифта, абзаца, стиля, колонок, установка оформления и заполнения, автоформатирование документа
Сервис	Проверка правописания, выбор языка, расстановка переносов, установка опций текстового редактора

Окончание

Команда	Действие
Таблица	Операции обработки таблиц: создание, преобразование, форматирование, использование математических формул
Окно	Управление окнами редактирования документов: открытие новых, упорядочивание, переключение между открытыми окнами
Справка	Вызов справочной информации по работе с MS Word

Система меню MS Word представляет собой интуитивно понятный пользовательский интерфейс. Меню команд выполнено в виде иерархической системы всплывающих подменю, т.е. подменю, активизирующихся при выборе некоторого пункта основного меню. Каждый пункт всплывающего меню соответствует определенной команде или названию меню следующего уровня (такие пункты отмечаются символом ▸). Активизация основного меню и выбор требуемого пункта могут быть реализованы с помощью мыши — указатель мыши нужно установить на строку меню, щелкнуть левой кнопкой и, перемещая указатель мыши, выбрать требуемый пункт. *Запуск команды на выполнение* осуществляется щелчком левой кнопкой мыши. Чтобы *закрыть* меню, не выбрав ни одной команды, следует щелкнуть левой кнопкой мыши в области редактирования документа. При этом в окне редактирования документа должен появиться мигающий курсор.

Для ускоренного вызова некоторых команд можно использовать специальные сочетания клавиш, которые указываются в меню справа от соответствующей команды. Например, команде открытия файла *Файл | Открыть...* соответствует сочетание клавиш [Ctrl] + [O]. Такие сочетания клавиш также называют «горячими клавишами».

Под основным меню расположены *панели инструментов*, на которых находятся кнопки для быстрого доступа к тем командам меню, которые используются наиболее часто.

Окно редактирования документа занимает основную площадь окна редактора. Эта область предназначена для ввода и редактирования текста документа. После запуска редактора MS Word в окне редактирования появляется пустой документ (чистый лист), готовый для набора текста.

Слева и вверху окна редактирования документа расположены *вертикальная и горизонтальная линейки* с делениями и маркерами. На горизонтальной линейке показаны значения правого и левого полей страницы и значения параметров абзаца, внутри которого в данный момент находится курсор. Вертикальная линейка видна только в режиме разметки страницы. На ней показаны значения верхнего и нижнего полей страницы.

Справа и внизу в окне редактирования документа расположены горизонтальная и вертикальная *полосы прокрутки*. Они используются для перемещения по документу наряду с курсорными клавишами и клавишами [PageUp] и [PageDown].

Строка состояния в самом низу окна редактирования документа содержит сведения о документе и подсказки о выполняемых редактором действиях.

7.7. Основные операции с документами

ВВОД И ИСПРАВЛЕНИЕ ТЕКСТА

Место, в котором будет размещаться набираемый текст, указывает *курсор* (мерцающая вертикальная черточка). Во время набора текста при достижении правого поля текст автоматически переносится на следующую строку. Для разделения текста на абзацы при завершении предыдущего абзаца нажимают клавишу [Enter]. Набранные символы можно удалять как справа, так и слева от курсора с помощью клавиш [Delete] и [←] соответственно.

Для перемещения курсора в нужное место текста следует либо щелкнуть мышью в этой позиции, либо воспользоваться курсорными клавишами. Если вместе с клавишей [Вправо] или [Влево] нажать клавишу [Ctrl], то курсор перепрыгнет на начало следующего или предыдущего слова соответственно.

В процессе набора текста введенные символы могут добавляться в документ или замещать ранее введенные символы. Соответствующие режимы ввода текста называются режимами *вставки* и *замены*. В режиме *вставки* вводимый символ помещается в позицию курсора, а текст, находящийся правее, сдвигается вправо. В режиме *замены* вводимый символ помещается в позицию курсора и замещает символ, находящийся справа от него.

Основным режимом ввода текста в редакторе MS Word является режим вставки. Для того чтобы изменить режим ввода текста, нужно дважды щелкнуть левой кнопкой мыши на надписи *ЗАМ* в строке статуса. Когда MS Word находится в режиме замены, надпись *ЗАМ* подсвечена.

РАБОТА С ФРАГМЕНТАМИ ТЕКСТА, ИСПОЛЬЗОВАНИЕ БУФЕРА ОБМЕНА

Для выполнения операции с целым фрагментом текста (например, замена шрифта, удаление целого предложения, выравнивание сразу нескольких абзацев и т.д.) нужный фрагмент необходимо сначала *выделить*, что можно сделать различными способами. Самый простой

из них — это щелкнуть левой клавишей мыши на начало фрагмента и переместить курсор мыши по тексту (выделяемый фрагмент при этом подсвечивается). Выделить нужный участок можно также с помощью курсорных клавиш, удерживая при этом клавишу [Shift].

Дополнительные способы выделения:

- Слово** — двойной щелчок мышью на нужном слове;
Строка — щелчок мышью по левому полю напротив строки;
Абзац — двойной щелчок мышью по левому полю напротив абзаца;
Весь текст — комбинация клавиш [Ctrl] + [A].

MS Word позволяет работать с *буфером обмена*, который служит для временного хранения различной информации (фрагментов текста, рисунков и т.д.). Выделенный текст можно:

- удалить из документа, нажав клавишу [Delete];
- скопировать в буфер обмена, нажав клавиши [Ctrl] + [Insert];
- вставить хранящийся в буфере обмена текст в текущую позицию курсора можно, нажав клавиши [Shift] + [Insert].

ФОРМАТИРОВАНИЕ АБЗАЦЕВ

Внешний вид документа зависит в большей степени от форматирования отдельных абзацев. Среди элементов форматирования абзацев основными являются:

- ♦ выбор способа выравнивания текста;
- ♦ изменение положения его левой и правой границ относительно страницы.

В редакторе MS Word *абзац* — это фрагмент текста, заключенный с двух сторон символами конца абзаца (¶), которые вставляются в текст при нажатии клавиши [Enter]. Обычно эти символы не отображаются, но их можно сделать видимыми, нажав кнопку с изображением символа конца абзаца на панели инструментов.

ВЫРАВНИВАНИЕ

При вводе текст по умолчанию сдвигается влево и выравнивается *по левому полю*. Однако он может быть также *отцентрирован*, *выровнен по его правому полю* или сразу *по обоим полям*.

Для задания выравнивания нужно *выделить абзацы*, которые требуется отцентрировать или выровнять, и нажать соответствующую кнопку выравнивания (рис. Л-7.2) на *панели инструментов*.



Рис. Л-7.2. Кнопки выравнивания абзацев

ОТСТУПЫ

Текст в абзацах располагается между правым и левым полями страницы. Однако можно изменить границы абзаца, установив для него отступы, т.е. расстояния от левой и правой границ абзаца до соответствующих полей страницы.

Для создания отступов предназначена горизонтальная линейка с маркерами. Каждый маркер позволяет настраивать отдельный вид отступа (рис. Л-7.3).

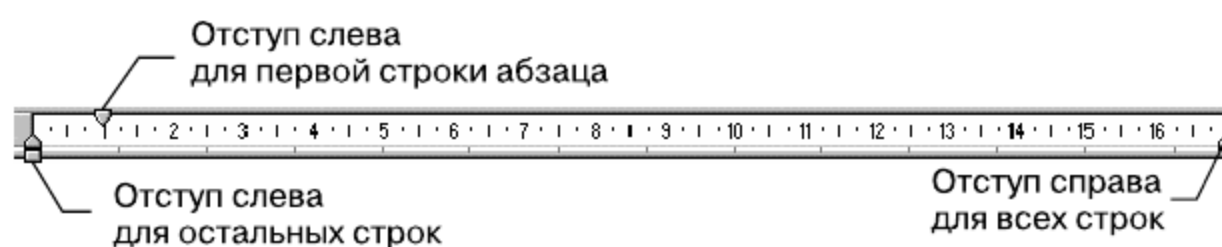


Рис. Л-7.3. Горизонтальная линейка

Для задания отступа необходимо выделить нужные абзацы, а затем на горизонтальной линейке с помощью мыши переместить маркеры отступов в желаемую позицию.

РАБОТА СО ШРИФТАМИ

Одним из основных элементов оформления документа является применение разных шрифтов. MS Word позволяет выбирать такие параметры шрифта, как:

- ◆ тип (Times New Roman, Arial, Courier и т.д.);
- ◆ способ начертания (**жирный**, *курсив*, подчеркнутый, ^{верхний индекс}, _{нижний индекс});
- ◆ размер;
- ◆ цвет.

Для изменения шрифта нужно сначала выделить фрагмент текста, а затем в подменю *Формат* выбрать пункт *Шрифт*. В открывшемся окне диалога можно изменить любые параметры шрифта. Для быстрого выбора типа шрифта, его размера и способа начертания удобно пользоваться соответствующими элементами на панели инструментов (рис. Л-7.4).



Рис. Л-7.4. Элементы настройки параметров шрифта

РАБОТА С ТАБЛИЦАМИ В ТЕКСТОВЫХ ДОКУМЕНТАХ**Создание таблицы**

Документы Word часто содержат данные, оформленные в виде таблицы, которая используется для более удобного расположения информации документа. Таблица состоит из *столбцов* и *строк*, на пересечении которых находятся *ячейки*. В ячейках размещается информация произвольного типа: текст, числа, графика, формулы. Для перемещения по ячейкам таблицы используются *мышь* или клавиши: <←>, <→>, <↑>, <↓>.

Таблицы удобны для представления различных списков, перечней, финансовой информации и т.п. Таблица может быть вставлена в любом месте документа, для чего необходимо установить курсор в то место документа, куда она вставляется, и выбрать пункт *Вставить-Таблица* подменю *Таблица* основного меню. В результате откроется окно диалога *Вставка таблицы* (рис. Л-7.5).

В этом окне можно задать число строк и столбцов таблицы и ширину каждого столбца. После нажатия кнопки *ОК* в месте расположения курсора появится таблица с заданным числом строк и столбцов.

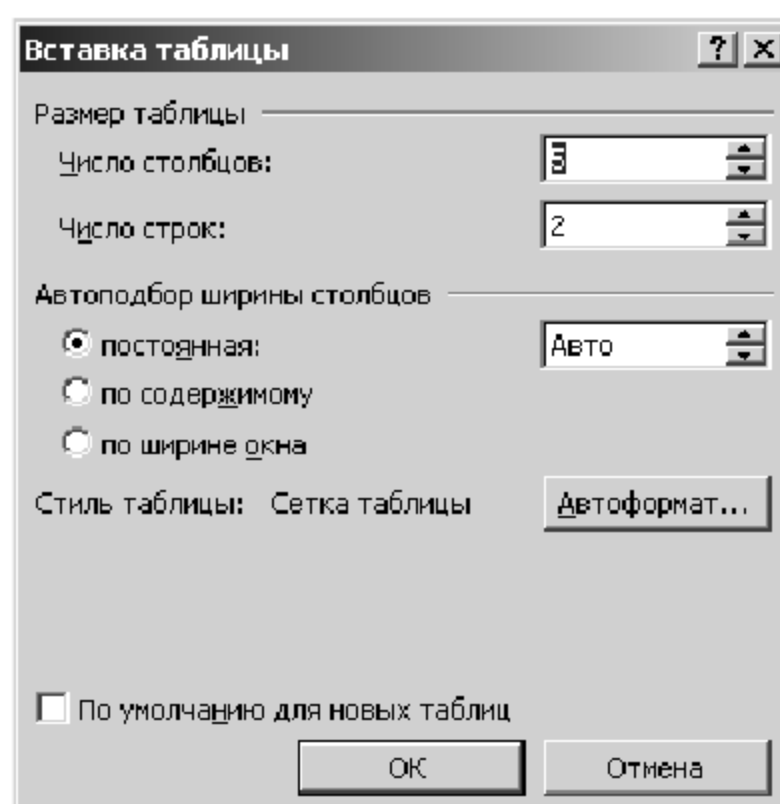


Рис. Л-7.5. Окно *Вставка таблицы*

Заполнение таблицы. Изменение структуры таблицы

Следующим шагом формирования таблицы является ее заполнение информацией. Для перемещения внутри таблицы используются те же способы, что и для перемещения по документу. При вводе инфор-

мации в ячейку таблицы, когда число символов текста превышает ширину ячейки, ее высота автоматически увеличивается на одну строку, и текст переносится на следующую строку внутри ячейки.

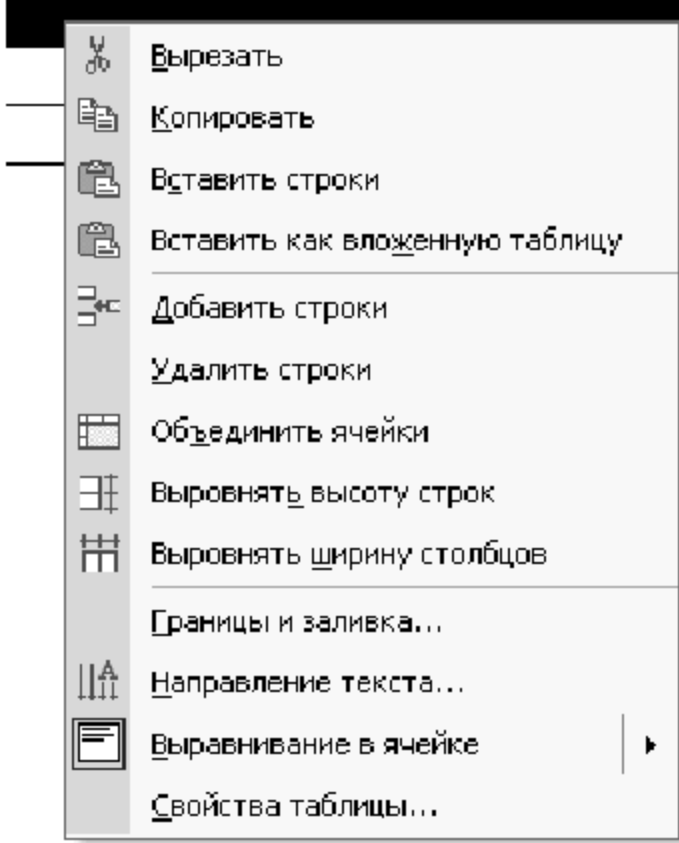
При работе с таблицей возникает необходимость изменения ее структуры — изменения числа строк и столбцов первоначально созданной таблицы. Перечень основных действий по реструктуризации таблицы представлен в табл. Л-7.2.

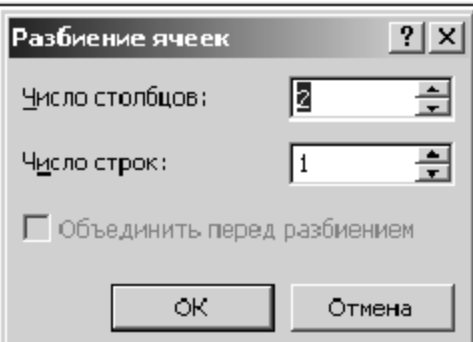
Таблица Л-7.2

Основные действия над таблицами

Действие	Содержание действия
Изменение ширины столбца (наиболее просто выполнить с помощью мыши)	Подведите указатель мыши к правой границе столбца и небольшими горизонтальными передвижениями мыши добейтесь, чтобы указатель принял вид двойной вертикальной линии со стрелками; нажмите левую клавишу мыши и, не отпуская ее, переместите границу столбца влево или вправо до нужной ширины, после чего отпустите клавишу мыши
Выделение одной ячейки таблицы	Установите указатель мыши между левой границей ячейки и текстом внутри нее (указатель мыши примет форму стрелки, направленной вправо); щелчком левой клавиши мыши выделите ячейку
Выделение нескольких ячеек таблицы	Установите указатель мыши в любое место первой ячейки из группы выделяемых ячеек; перемещая указатель мыши при нажатой левой клавише в последнюю ячейку группы, выделить нужное число ячеек по вертикали или по горизонтали
Выделение строк таблицы	Поместите указатель мыши на свободное место слева от этой строки вне таблицы; щелчком левой клавиши мыши выделите строку
Выделение столбцов таблицы	Подведите указатель мыши к верхней границе выделяемого столбца (указатель мыши примет вид стрелки, направленной вниз); щелчком левой клавиши мыши выделите столбец; для выделения нескольких столбцов, не отпуская клавишу мыши, переместите указатель на требуемое число столбцов влево или вправо
Снятие выделения	Поставьте курсор в произвольное место; щелкните левой клавишей мыши
Вставка строки	Выделите строку, перед которой необходимо вставить новую строку и щелкните правой кнопкой мыши (на экране появится контекстное меню, показанное ниже); выберите команду <i>Добавить строки</i>

Продолжение

Действие	Содержание действия
Удаление строки	Выделите удаляемую строку и щелкните правой кнопкой мыши; выберите из контекстного меню команду <i>Удалить строки</i>
Контекстное меню (графический объект)	
Вставка столбца	Выделите столбец, перед которым хотите вставить новый столбец, и щелкните правой кнопкой мыши; в вызванном контекстном меню выберите команду <i>Добавить столбцы</i>
Удаление столбца	Выделите удаляемый столбец и щелкните правой кнопкой мыши; в вызванном контекстном меню выберите команду <i>Удалить столбцы</i>
Объединение ячеек	Выделите расположенные рядом ячейки, которые хотите объединить; щелкните правой клавишей мыши; в открывшемся контекстном меню выберите команду <i>Объединить ячейки</i>
Разбиение ячеек	Поместите курсор в разбиваемую ячейку; щелкните правой кнопкой мыши; в открывшемся контекстном меню выберите команду <i>Разбить ячейки</i> (на экране появится окно диалога <i>Разбиение ячеек</i>); в полях <i>Число столбцов</i> и <i>Число строк</i> выберите нужное число столбцов и строк, на которое нужно разбить ячейку; нажмите кнопку <i>OK</i>

Действие	Содержание действия
Окно диалога <i>Разбиение ячеек</i> (графический объект)	

Суммирование содержимого строки или столбца

MS Word позволяет выполнить автоматическое суммирование содержимого строки или столбца таблицы. Для подсчета суммы надо поместить курсор в ту ячейку, где должен располагаться результат суммирования. Чтобы вычислить сумму значений строки таблицы, эта ячейка должна находиться в конце суммируемой строки. Для вычисления суммы значений столбца курсор должен располагаться в самой нижней ячейке суммируемого столбца. После нажатия кнопки *Автосумма-Σ* на панели инструментов *Таблицы и границы* в ячейке с курсором появится сумма значений ячеек таблицы, расположенных сверху или слева от этой ячейки.

СОХРАНЕНИЕ ДОКУМЕНТА

Во время работы над документом сам документ находится в оперативной памяти и может быть потерян в результате непредвиденных ситуаций. Поэтому необходимо систематически сохранять документ на жестком диске. Для сохранения документа надо дать имя файлу, содержащему документ, и решить, в какой папке и на каком диске его поместить. Для сохранения документа в виде файла можно воспользоваться кнопкой *Сохранить* на панели инструментов *Стандартная*.

Первое сохранение документа. Если документ ранее не сохранялся, и работа начата «с чистого листа» вызовом, например, MS Word через главное меню, то на экране появится окно диалога *Сохранение документа* (рис. Л-7.6).

В поле *Имя файла* введите имя файла, в котором будет сохранен документ, или согласитесь с именем, предложенным MS Word, например Doc.1.

В поле *Папка* окна *Сохранение документа* нажмите кнопку со стрелкой ▼. Раскроется список доступных дисков и папок на вашем ком-

пьютере. Сначала выберите диск и нажмите клавишу *Открыть*, его название появится в верхнем окне, а в основном окне появится список папок, содержащихся на этом диске.

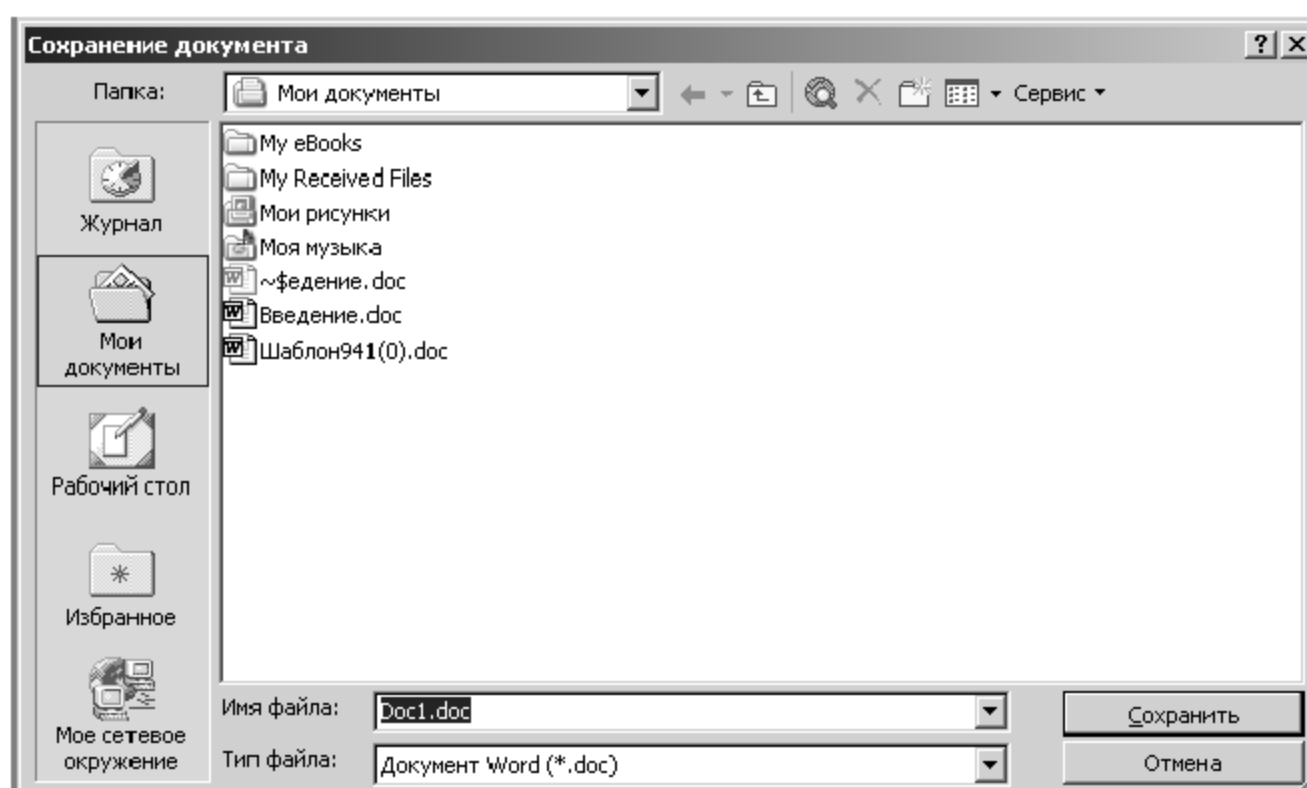


Рис. Л-7.6. Диалоговое окно сохранения документа

Из списка папок выберите ту, в которой хотите сохранить файл. Название выбранной папки появится в верхнем окне, а в основном окне — список папок, вложенных в нее.

Нажмите кнопку *Сохранить* окна диалога *Сохранение документа*.

Сохранение файла под прежним именем. После работы с файлом, которому уже было присвоено имя, нажмите кнопку *Сохранить* на панели инструментов *Стандартная*, и файл будет сохранен под прежним именем.

Сохранение файла под другим именем. Файл можно сохранить под другим именем или в другом месте. Для такого сохранения используется команда *Файл — Сохранить как*. При выполнении этой команды появляется описанное выше окно диалога, в котором можно задать новое имя и новое место хранения документа.

Создание новой папки перед сохранением документа. Можно создать новую папку перед сохранением документа. Для этого нужно нажать кнопку *Создать папку* на верхней панели диалогового окна, в котором выбрано место сохранения документа. В появившемся окне *Создание папки* введите имя папки и нажмите кнопку *ОК*. Затем сохраняйте документ во вновь созданной папке обычным образом.

7.8. Практическая работа на ПК

Практическая работа с использованием редактора MS Word состоит из двух частей:

- ♦ выполнение основных операций по работе с документом (см. п. 7.7. Основные операции с документами);
- ♦ выполнение индивидуального задания в соответствии с номером варианта (табл. Л-7.3).

ОСНОВНЫЕ ОПЕРАЦИИ

Ввод и исправление текста

Задание 1

Ввести два абзаца, разделенных пустой строкой (см. ниже), для чего необходимо выполнить следующие действия:

- а) набрать текст первого абзаца;
- б) два раза нажать клавишу [Enter];
- в) набрать текст второго абзаца;
- г) нажать клавишу [Enter].

Редактор Microsoft Word хранит запись последних трех мест, в которых вводился или редактировался текст. Чтобы перейти к месту предыдущего редактирования текста, следует нажать нужное число раз клавиши [SHIFT] + [F5].

Примечание. Это средство можно использовать также для возвращения к месту предыдущего редактирования текста после сохранения документа.

Задание 2

Слово «Редактор» заменить словосочетанием «Текстовый редактор», для чего необходимо:

- а) установить курсор перед словом «Редактор»;
- б) вставить в этом месте слово «Текстовый», нажать пробел;
- в) перейти в режим замены символов, для чего нажать клавишу [Insert];
- г) изменить первую букву слова «Редактор» на строчную, для этого просто ввести букву «р»;
- д) вернуться в режим вставки символов.

Задание 3

Фразу «следует нажать» заменить словом «нажмите». Для этого:

- а) выделите словосочетание «следует нажать»;
- б) удалите выделенный фрагмент, нажав клавишу [Delete];
- в) наберите слово «нажмите».

Задание 4

Во втором абзаце слово «также» поместить перед словом «использовать». Для выполнения задания использовать буфер обмена, для чего необходимо выполнить действия:

- а) выделить слово «также», для чего два раза щелкнуть по нему мышью;
- б) скопировать выделенное слово в буфер обмена — клавиши [Ctrl] + [Insert];
- в) удалить выделенный фрагмент из текста с помощью клавиши [Delete];
- г) установить курсор перед словом «использовать»;
- д) вставить в это место содержимое буфера обмена — клавиши [Shift] + [Insert].

Форматирование абзацев**Задание 1**

Выровнять абзацы по ширине между правым и левым полями. Для этого требуется:

- а) выделить оба абзаца, для чего следует установить курсор мыши с левой стороны напротив первой строки; нажать левую кнопку мыши и, удерживая ее, переместить курсор к последней строке, отпустить кнопку;
- б) нажать соответствующую кнопку выравнивания на панели инструментов.

Задание 2

Создать в первом абзаце красную строку, для чего:

- а) выделить первый абзац, щелкнув два раза мышью по левому полю напротив абзаца;
- б) с помощью мыши переместить на горизонтальной линейке маркер *Отступ слева для первой строки абзаца* в позицию 1 см.

Работа со шрифтами**Задание 1**

Установить для всего текста тип шрифта Arial и размер 14. Для этого:

- а) выделить весь текст с помощью клавиш [Ctrl] + [A];
- б) выбрать в подменю *Формат* команду *Шрифт...*;
- в) в открывшемся диалоговом окне выбрать тип шрифта *Arial* и размер **14**;
- г) нажать кнопку **ОК**.

Задание 2

Изменить прямое начертание слов «Microsoft Word» на курсивное, для чего:

- а) выделить слова «Microsoft Word»;
- б) нажать кнопку *K* на панели инструментов.

Задание 3

Выделить слово «Примечание» жирным шрифтом, выполнив действия:

- а) выделить слово «Примечание»;
- б) нажать на панели инструментов кнопку *Ж*.

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ**(ВВОД И РЕДАКТИРОВАНИЕ ПОДГОТОВЛЕННОГО ДОМА ТЕКСТА)**

Перечень заданий для каждого из вариантов приведен в табл. Л-7.3, их содержание — в табл. Л-7.4.

Таблица Л-7.3

Номера вариантов и заданий

Номер варианта	Перечень выполняемых заданий
1	1, 2, 4, 9, 13, 17, 25, 26, 27
2	1, 3, 5, 8, 12, 18, 23, 26, 27
3	1, 3, 5, 10, 14, 16, 24, 25, 27
4	1, 4, 6, 8, 12, 20, 21, 26, 27
5	1, 2, 5, 9, 13, 15, 18, 24, 27
6	1, 8, 10, 12, 13, 19, 22, 25, 27
7	1, 4, 9, 14, 17, 18, 25, 26, 27
8	1, 5, 8, 9, 11, 13, 15, 19, 23, 27
9	1, 2, 5, 10, 14, 17, 20, 25, 27
10	1, 3, 4, 6, 8, 12, 15, 19, 22, 27

Таблица Л-7.4

Содержание заданий

Номер задания	Задание
1	Набрать любой текст, состоящий не менее чем из 15 строк, разделенный на три абзаца
2	Перенести первый абзац в конец текста
3	Поменять местами первое и последнее предложения текста
4	Выровнять абзацы по центру
5	Выровнять абзацы по левому и правому полям и создать для них красные строки
6	Выровнять последний абзац по ширине и создать для него красную строку
7	Выровнять последний абзац текста по центру

Окончание

Номер задания	Задание
8	Добавить перед первым абзацем строку-заголовок
9	Добавить в конец текста строку с текущей датой
10	Добавить в конец текста строку со своими именем и фамилией
11	Выровнять строку-заголовок по правому полю
12	Выровнять строку-заголовок по центру
13	Выровнять последнюю строку по правому полю
14	Выровнять последнюю строку по центру
15	Установить для текста тип шрифта <i>Times New Roman</i>
16	Установить для текста тип шрифта <i>Arial</i>
17	Установить для текста тип шрифта <i>Courier</i>
18	Установить для текста размер шрифта 11
19	Установить для текста размер шрифта 12
20	Установить для текста размер шрифта 13
21	Установить для заголовка жирный шрифт размером 14
22	Установить для заголовка шрифт размером 16
23	Установить для заголовка шрифт размером 14 с подчеркиванием
24	Установить для последней строки жирный шрифт
25	Установить для последней строки подчеркивание
26	Сделать текст абзацев курсивным
27	Распечатать полученный документ

Примечание. Созданные и отредактированные текстовые файлы сохранять в папке * (номер группы), для чего:

- ◆ в основном меню выбрать пункт *Файл | Сохранить как*;
- ◆ в окне сохранения документа выбрать имя папки;
- ◆ ввести имя файла;
- ◆ нажать *Сохранить*.

РАБОТА С ТАБЛИЦАМИ В ТЕКСТОВЫХ ДОКУМЕНТАХ

Создайте таблицу из 5 столбцов и 1 строки и заполните ее данными об итогах вступительных экзаменов по прилагаемому образцу:

№ п/п	Ф.И.О.	Математика	Физика	Русский язык
1	Иванов И.И.	57	49	35

1. Добавьте строку перед первой и введите данные соседа, сидящего слева или справа.
2. Добавьте столбец «Дата» после каждого столбца с соответствующим названием предмета для указания даты сдачи и заполните столбцы.
3. Добавьте третью строку и запишите в ее вторую ячейку текст «Итого».

4. Добавьте столбец справа таблицы с заголовком «Общий балл».
5. Заполните столбец «Общий балл», начиная сверху, просуммировав числа по строкам. Заполните строку «Итого», просуммировав числа по столбцам.
6. Первые две строки с фамилиями поменяйте местами.
7. Объедините ячейки с названиями предметов и добавьте над ними строку с названием «Предметы». Отредактируйте их содержание так, чтобы таблица имела следующий вид:

Итоги вступительных экзаменов

№ п/п	Ф.И.О.	Предметы						Общий балл
		Математика	Дата	Физика	Дата	Русский язык	Дата	
1	Иванов И.И.	57	05.05.10	49	13.05.10	35	20.05.10	141
2	Сидоров В.В.	55						
	Итого	112						

8. Измените ширину столбцов так, чтобы таблица располагалась по всей ширине страницы и ячейки были равномерно заполнены.
9. Над таблицей поместите заголовок «Итоги вступительных экзаменов». Заголовок выполнить жирным шрифтом.
10. Сохраните созданную таблицу в папке с именем своей группы.

Контрольные вопросы

1. Что представляет собой текстовый редактор?
2. Отличительные особенности редактора MS Word.
3. Как запустить редактор MS Word?
4. Перечислите основные операции редактора MS Word по работе с документами.
5. Какие версии редактора MS Word вы знаете?
6. Перечислите дополнительные возможности редактора MS Word.
7. Назовите текстовые редакторы, которые вы знаете и их отличительные особенности.
8. Перечислите элементы среды редактирования MS Word и поясните их назначение.
9. Как выделить слово, строку, абзац текста?
10. Что такое буфер обмена?
11. Перечислите основные операции при работе с таблицами.
12. Перечислите перечень действий, которые необходимо выполнить, чтобы добавить строку (столбец), изменить содержание ячейки.

Лабораторная работа 8 СЖАТИЕ ИНФОРМАЦИИ В ПК

8.1. Цель работы

Изучение методов сжатия данных. Изучение программы-архиватора WinRAR и работа с ней. Нарботка практических навыков по созданию архивов и их модификаций.

8.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 3, 7].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль:
 - ◆ общие сведения:
 - о сжатии информации,
 - о методах сжатия, их достоинствах и недостатках,
 - об архиваторах, их особенностях и возможностях;
 - ◆ основные понятия: файлы исходные, архивные, самораспаковывающиеся и др.;
 - ◆ работа с программой WinRAR (создание архивов, просмотр, удаление, извлечение файлов и др.).
4. Работа с программой-архиватором WinRAR.
5. Выполнение индивидуальных практических заданий: часть 1 (см. табл. Л-8.3) — дома и часть 2 (см. подраздел 8.6) — в лаборатории ПК.
6. Отчет о проделанной работе:
 - ◆ индивидуальное выполнение на ПК заданных преподавателем операций: архивации файлов, их извлечения из архивов и др;
 - ◆ оформление отчета (добавляется по результатам работы на ПК);
 - ◆ защита отчета.

8.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ краткое описание основных методов сжатия и их особенностей, основных архиваторов, достоинств и возможностей архиватора WinRAR.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ результат выполнения практического задания, часть 1 (подраздел 8.6);
- ◆ протокол выполнения заданий, часть 2 (подраздел 8.6);
- ◆ содержимое и размер двух архивов (см. задание 6, ч. 2) в виде табл. Л-8.1.

Таблица Л-8.1

Пример содержимого архива

Имя	Размер	Упак. размер	Тип	Дата изменения
Файл 1.txt	2010	1036	Текстовый документ	17-07-01 20:58
Файл 2.txt	4957	2423	Текстовый документ	17-07-01 20:59
Файл 3.txt	3572	1720	Текстовый документ	17-07-01 20:59

8.4. Теоретическая часть**ОСНОВНЫЕ ПОНЯТИЯ И МЕТОДЫ СЖАТИЯ ДАННЫХ**

Почти сразу после появления ПК возникла проблема нехватки памяти для хранения различных файлов в компьютере. Винчестеры несколько сняли остроту проблемы. Однако программисты стали создавать еще более сложные программы, и какую бы память не имел ПК, ее объем не всегда может удовлетворять пожеланиям всех пользователей. Но наряду с увеличением объема памяти актуальными являются способы эффективного хранения данных в ПК.

Во многих случаях информация, содержащаяся в файлах, избыточна. Для устранения избыточности используются специальные методы сжатия данных, основанные на поиске в файле избыточной информации и последующем ее кодировании с целью получения минимального объема.

Сжатие информации — это процесс сокращения числа битов, необходимых для хранения и передачи некоторого объема информации. Возможно *сжатие без потерь*, когда информация, восстановленная из сжатого сообщения, в точности соответствует исходной, и *сжатие с потерями* (необратимое, ущербное), когда восстановленная информация только частично соответствует исходной. Первое применяется при обработке текстов, записанных на естественном или искусственном языках, второе — при обработке изображений и звука, для цифровой записи аналоговых сигналов.

Различают два основных способа сжатия:

- ◆ *статистический*,
- ◆ *словарный*.

При первом способе каждому символу присваивается код, основанный на вероятности его появления в тексте. Высоковероятные символы получают короткие коды, и наоборот. Одним из самых ранних и широко известных статистических методов является *алгоритм Хаффмена*, предложенный в 1952 г. В нем символы заменяются кодом, состоящим из целого числа битов. В конце 70-х годов он был вытеснен *арифметическим кодированием*, имеющим схожую с кодом Хаффмена функцию и основанным на идее кодирования символов дробным числом битов. Арифметическое сжатие может быть использовано в тех случаях, когда степень сжатия важнее, чем временные затраты на сжатие информации.

При словарном способе группы последовательных символов, или фраз, заменяются кодом. Замененная фраза может быть найдена в некотором словаре. В 1977 г. *Лемпель и Зив* предложили свою модификацию словарного метода, отличающуюся от хаффменовского и арифметического методов, в которой сжатие основано на свойстве «потока символов» иметь повторяющиеся участки. *Поток символов* — это исходные данные для сжатия, например текстовый файл, массив. Основная идея алгоритма Лемпеля—Зива состоит в том, что второе и последующие вхождения некоторой строки символов в сообщении заменяются ссылкой на ее первое появление в сообщении.

На основании того, что любая практическая схема словарного сжатия может быть сведена к соответствующей статистической схеме сжатия, был предложен общий алгоритм преобразования словарного метода в статистический. Поэтому при поиске лучшего сжатия статистическое кодирование обещает быть наиболее плодотворным, хотя словарные методы и привлекательны своей быстротой.

Одним из наиболее простых и наглядных методов является метод сжатия последовательностей одинаковых символов, не относящийся, однако, к названным основным методам.

МЕТОД СЖАТИЯ ПОСЛЕДОВАТЕЛЬНОСТЕЙ ОДИНАКОВЫХ СИМВОЛОВ

Метод основан на представлении последовательности одинаковых символов в виде двух величин K и S , где K — число повторяющихся символов, S — код этого символа. Основным недостатком данного метода является то, что он обеспечивает сжатие лишь в случае, когда в исходном файле основную часть составляют повторяющиеся символы. В противном случае сжатый файл может занимать больше места, чем исходный неуплотненный файл.

Наиболее эффективно метод сжатия последовательностей одинаковых символов работает в случае двоичных файлов. Поскольку двоичный файл содержит только два символа — 1 и 0, то в сжатом файле можно не указывать сам сжимаемый символ, а указать лишь число повторений символа. Тогда каждая новая величина в сжатом файле будет означать изменение состояния битовой строки, но значение первого символа нужно указать обязательно.

Рассмотрим три примера, поясняющих работу этого алгоритма. Первый пример показывает случай сжатия исходного файла, второй — неудача сжатия, вследствие чего размер сжатого файла превышает размер исходного файла, третий пример — сжатие двоичного файла.

Пример 1. Исходный файл: MMMNNNKKNNNNNNNNNNLLLLL. Так как каждый символ при обычном представлении в ЭВМ занимает один байт памяти, то полная длина исходного файла $N_0 = 20$ байт. Сжатый файл: 3M3N1K8N5L. Общее число символов в сжатом файле $N_1 = 10$ байт. Коэффициент сжатия $K_{сж} = N_0/N_1 = 2$.

Пример 2. Исходный файл: Сжатие---информации. Сжатый файл:
1C1ж1a1t1и1e3-1и1н1ф1o1p1m1a1ц2и1.
 $N_0 = 20$ байт; $N_1 = 34$ байт; $K_{сж} = 0,59$.

Видим, что в данном примере сжатый файл получился больше исходного.

Пример 3. Исходный файл: 1100001111110100101111110000. Сжатый файл:

12471121174 $N_0 = 30$ байт; $N_1 = 11$ байт; $K_{сж} = 2,73$.

ПРОГРАММЫ-АРХИВАТОРЫ

На основе методов сжатия данных созданы различные программы, называемые архиваторами или упаковщиками. Существует много программ-архиваторов, имеющих различные показатели по степени и времени сжатия. Эти показатели могут быть разными для различных файлов (текстовых, графических, исполняемых и т.д.), т.е. один архиватор хорошо сжимает текстовый файл, а другой — исполняемый. Среди самых известных и часто используемых программ для Windows выделяются следующие: ARJ, PKZIP, RAR, LHA, HA и т.д. Зачастую упаковщики осуществляют сжатие сразу несколькими способами.

Обычно программы для архивации файлов позволяют помещать копии файлов на диске в сжатом виде в архивный файл, извлекать файлы из архива, просматривать оглавление архива и др. В настоящее время архиваторы, работающие под Windows, вытесняют конкурентов в основном за счет использования 32-разрядной шины данных, более удоб-

ного и интеллектуального интерфейса, расширенных возможностей и более совершенных алгоритмов сжатия. Кроме того, эти программы поддерживают не один, как раньше, а сразу несколько различных форматов архивных файлов. Для примера рассмотрим работу программы WinRAR, которая также отличается удобной встроенной оболочкой работы с файлами и папками.

8.5. Программа-архиватор WinRAR

ДОСТОИНСТВА И ОСНОВНЫЕ ВОЗМОЖНОСТИ WINRAR

WinRAR — 32-разрядная версия Windows архиватора RAR — мощное средство, которое позволяет создавать и управлять архивными файлами.

К достоинствам программы-архиватора *WinRAR* можно отнести следующее:

- ◆ интеллектуальный алгоритм сжатия (глубокий анализ данных, подлежащих сжатию, и др.);
- ◆ специальный алгоритм сжатия для мультимедиа-файлов;
- ◆ неархивное управление RAR;
- ◆ плотная архивация, которая может поднять коэффициент сжатия на 10—50% по сравнению с общими методами, особенно при упаковке большого числа аналогичных файлов (при этом файлы представляются как непрерывный поток данных).

Возможности WinRAR:

- ◆ создание архивов с различной степенью уплотнения;
- ◆ просмотр как архивных, так и обычных файлов;
- ◆ создание многотомных архивов;
- ◆ создание самораспаковывающихся архивов (в том числе и многотомных);
- ◆ восстановление физически поврежденных архивов.

Кроме того, WinRAR обладает рядом других возможностей, таких, например, как шифрование, архивные комментарии, регистрация ошибок и т.п.

РАБОТА С ПРОГРАММОЙ-АРХИВАТОРОМ WINRAR

Запуск WinRAR

Для запуска программы WinRAR необходимо выполнить одно из следующих действий:

- ◆ выбрать меню *Пуск | Программы | WinRAR* и щелкнуть на *WinRAR*;
- ◆ дважды щелкнуть мышью на файле архива;
- ◆ дважды щелкнуть мышью на ярлыке программы *WinRAR*.

Архивация файлов

Для создания архива необходимо сначала выбрать необходимые файлы и/или директории из списка файлов в основном окне WinRAR, что можно сделать, используя клавишу ПРОБЕЛ, [Shift] + клавиши позиционирования или щелкнув левой кнопкой мыши. Затем, выбрав один или более файлов, следует нажать кнопку *Add* наверху окна WinRAR или [Alt] + [A]. После этого появится диалоговое окно (рис. Л-8.1), в верхнем поле *Archive* которого надо ввести имя и расположение архивного файла или просто принять предложенное имя. В поле *Compression* можно выбрать степень уплотнения, что будет влиять на скорость архивации и размер архивного файла. Следующая строка *Dictionary size* дает возможность изменить размер внутреннего словаря архиватора. Этот параметр также будет влиять на размер конечного файла. С помощью диалогового поля *Volume size* можно разбить архивный файл на тома разного размера. Диалоговый блок *Archiving options* также предоставляет возможность изменять некоторые опции архивации, такие как:

- ◆ *Solid archive*/Плотный архив — при этом все файлы в архиве будут представляться как непрерывный поток данных, что позволит увеличить степень сжатия, но замедлит обращение к файлам внутри архива;
- ◆ *SFX archive*/Самораспаковывающийся архив — создается EXE модуль архива;
- ◆ *Multimedia compression*/Мультимедиа-уплотнение — в этом режиме RAR выполняет интеллектуальный анализ данных, направленный на улучшение архивации звука, видео и т.п.;
- ◆ *Put recovery record*/Добавление записи восстановления данных — облегчение восстановления данных при их порче или потере;
- ◆ *Put authenticity verification*/Добавлять проверку достоверности — WinRAR будет помещать в каждом новом и скорректированном архиве информацию относительно создателя, последнего времени коррекции и архивного имени;
- ◆ *Delete files after archiving*/Удалять файлы после их архивации — после помещения в архив файлы будут удалены.

После выбора всех необходимых параметров нажмите *ОК*. В течение процесса архивации будет отображено окно статистики. При необходимости прервать процесс архивации нужно щелкнуть мышью кнопку *CaFar Manager!* в командном окне. По завершении архивации окно статистики исчезнет, а вновь созданный архив будет выделен.

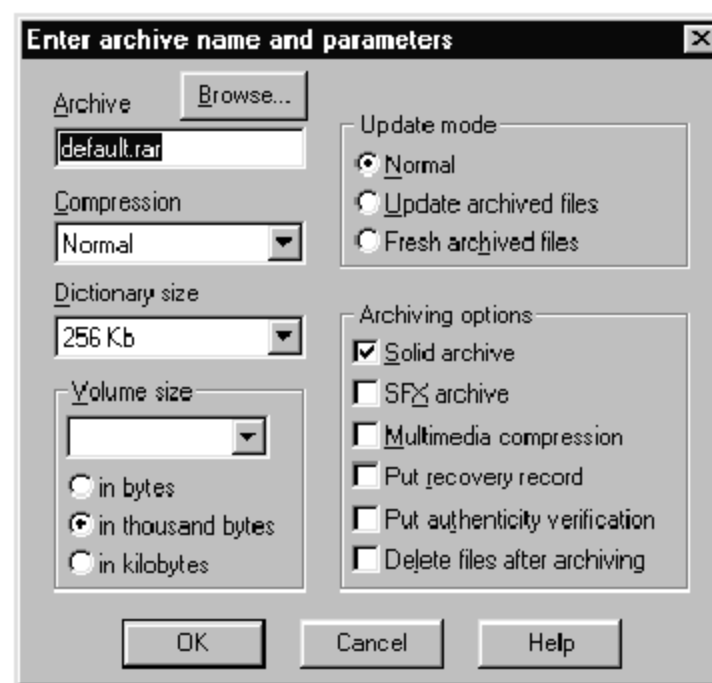


Рис. Л-8.1. Диалоговое окно

Можно также добавлять файлы к уже существующему архиву. Для этого надо выделить архив и нажать клавишу [Enter] (или дважды щелкнуть мышью), после чего WinRAR прочтает архив и отобразит его содержание. Далее можно просто перетаскивать мышью необходимые файлы из другого окна.

Просмотр файлов

Для просмотра файла его необходимо сначала *выделить* (это можно сделать, используя клавишу ПРОБЕЛ, [Shift] + клавиши позиционирования или щелкнув левой кнопкой мыши), а затем нажать кнопку

View .

С помощью кнопки View существует возможность просматривать как архивированные, так и нормальные файлы. Для просмотра могут использоваться как встроенные средства просмотра, так и другие программы. Если применяется встроенный просмотр, то можно просматривать несколько файлов одновременно. При использовании других средств просмотра можно просматривать только один файл. Команды встроенных средств просмотра показаны в табл. Л-8.2.

Таблица Л-8.2

Команды встроенных средств просмотра файлов

Меню/Пункт	Вызов	Функции
File/Exit	[Alt] + [F4]	Выход из просмотра
Edit/Copy	[Ctrl] + [C]	Копировать выделенный текст в буфер

Окончание

Меню/Пункт	Вызов	Функции
Edit/Select all	[Ctrl] + [A]	Выделить весь текст
Edit/Find	[Ctrl] + [F]	Поиск подстроки в тексте
Edit/Find next	[F3]	Поиск следующего появления строки
View as Windows	[Ctrl] + [W]	Использовать шрифты Windows
View as DOS text	[Ctrl] + [D]	Использовать DOS OEM шрифт
Help/View help	[F1]	Вызвать помощь

Удаление файлов

Для удаления файлов из архива или директории необходимо сначала выделить необходимые файлы (это можно сделать, используя клавишу ПРОБЕЛ, [Shift] + клавиши позиционирования или щелкнув левой кнопкой мыши), а затем нажать кнопку *Delete* .

Примечания:

1. Если из архива удаляются все файлы, то и пустой архив будет удален.
2. Удаление нельзя применять в отдельных томах архива.

Извлечение файлов

Для извлечения файлов из архива сначала нужно войти в архив, что можно сделать двойным щелчком мыши на его имени в окне WinRAR. В результате отображается содержание архива. Затем выбираются файлы/директории, которые должны быть извлечены, и необходимо щелкнуть мышью кнопку *Extract*  наверху окна WinRAR или нажать [Alt] + [E]. Если надо извлечь файлы в другую папку, а не в текущую, щелкните мышью кнопку *Extract to specified directory*  или нажмите [Alt] + [A], войдите в папку, расположенную в диалоговом блоке, и щелкните OK.

В течение процесса извлечения будет отображено окно статистики. При необходимости остановки процесса извлечения следует нажать кнопку *CaFar Manager!*. По завершении извлечения окно статистики исчезнет.

Другие возможности

Кроме рассмотренных выше кнопок* работы с архивами в верхнем меню оболочки WinRAR также находятся:

 — *test archiving files*/тестирование файлов архива — выполняет ложное файловое извлечение для проверки правильности извлечения файлов;

* Графическое изображение кнопок зависит от версии WinRAR и может быть разным.

-  — *add archive comment*/добавить к архиву комментарий. Максимальная длина комментария — 62 000 байт;
-  — *protect archive from damages*/добавление записи восстановления данных. К архиву может быть добавлена дополнительная, излишняя информация восстановления. За счет небольшого увеличения размера архивного файла она поможет восстановить архивные файлы в случае потери данных;
-  — *lock archive*/запирание архива. Эта команда предохраняет дальнейшие архивные модификации WinRAR;
-  — *convert archive to SFX*/преобразование архива в самораспаковывающийся. Эта команда создает self-extract архив (SFX), который самостоятельно извлекает содержащиеся в нем файлы (без использования архиватора);
-  — *show archive USER Sation*/показ информации об архиве. Показывается информация об архиве и его содержимом.

Справка

Для получения справки о работе программы-архиватора WinRAR необходимо выполнить одно из действий:

- ♦ находясь в окне WinRAR, нажать клавишу [F1];
- ♦ выбрать пункт меню Help/Contents (? / Содержание).

8.6. Практическая работа на ПК

Практическая работа состоит из двух частей. Первая часть выполняется дома без использования ПК в соответствии с номером варианта (табл. Л-8.3), а вторая содержит 10 заданий и выполняется с использованием программы-архиватора WinRAR.

ЧАСТЬ 1

Задание

В соответствии с заданным вариантом (соответствует номеру бригады), используя метод сжатия последовательностей одинаковых символов:

- а) найти сжатый файл для заданного исходного;
- б) найти двоичное представление заданного десятичного числа и сжать его.

Например, для двоичного представления десятичного числа $X = 871$ требуется 10 двоичных разрядов:

$$X = 871 = 1101100111_2,$$

и, следовательно, исходный файл занимает объем памяти $N_0 = 10$ байт, а сжатый файл $N_1 = 121\,223 - 6$ байт; $K_{сж} \approx 1,66$.

Для пунктов а и б рассчитать коэффициенты сжатия.

Таблица Л-8.3

Варианты заданий

Вариант	Исходный файл	Число
1	X=5.2*****Y=5.*****C=0.*****D=4.	513
2	*****ARCHIV*****	555
3	1000(3333+4444)+2223000=10000000	543
4	****STORAGE*****DATA****	520
5	СЖАТИЕ---ДАННЫХ	767
6	Z=100*****Y=0.001*****X=88	1020
7	<<<<<QWERTY>>>>>	800
8	(5555/1111+6661)*1000=6666000	903
9	+++++++COMPRESSION+++++++	511
10	223344+55+6600+770000=999999	544

ЧАСТЬ 2

Задания

1. В папке* создать две папки с именами DIR1 и DIR2 (что можно сделать, нажав правую клавишу мыши и выбрав в контекстном меню *Создать | Папка*), затем в папке DIR1 с помощью редактора *Блокнот* создать два текстовых файла: myfile1.txt и myfile2.txt объемом не менее 5 печатных строк, а в папке DIR2 создать файл myfile3.txt также объемом не менее 5 печатных строк (щелкнув правой клавишей мыши и выбрав в контекстном меню *Создать | Текстовый документ*).
2. Запустить программу-архиватор WinRAR (см. подраздел 8.5).
3. С помощью архиватора WinRAR создать архив arhiv1.rar, содержащий в себе файлы myfile1.txt и myfile3.txt, используя наилучший (best) метод уплотнения (см. подраздел 8.5. Архивация файлов).
4. Добавить в существующий arhiv1.rar файл myfile2.txt (см. подраздел 8.5. Архивация файлов).
5. Удалить из архива файл myfile1.txt (см. подраздел 8.5. Удаление файлов).
6. Создать архив arhiv2.rar, содержащий в себе файлы myfile2.txt и myfile3.txt, используя нормальную степень сжатия (normal).

* Вставить номер своей группы.

Сравнить и записать в отчет содержимое и размер двух полученных архивов.

7. Преобразовать файл myfile2.txt архива arhiv1.rar в SFX архив (см. подраздел 8.5. Другие возможности).
8. Создать архив arhiv3.rar из файлов myfile1.txt и myfile2.txt.
9. Преобразовать arhiv3.rar в самораспаковывающийся архив.
10. Из файла arhiv2.rar файл myfile2.txt распаковать в папку DIR2, а файл myfile3.txt удалить (см. подраздел 8.5. Извлечение файлов).

Контрольные вопросы

1. Что такое сжатие информации?
2. Какие виды и способы сжатия вы знаете и чем они отличаются?
3. Почему метод сжатия последовательностей одинаковых символов удобно применять для двоичных файлов?
4. Назовите основной недостаток метода сжатия последовательностей одинаковых символов.
5. Для чего применяется архивация данных?
6. Назовите наиболее часто используемые программы-архиваторы.
7. Что такое программа-архиватор WinRAR? Назовите ее основные достоинства и возможности.
8. Какие виды архивов вы знаете?
9. Для чего необходима дополнительная запись восстановления данных?
10. Для чего производится тестирование файлов архива?
11. Что такое SFX-архив?
12. Что такое многотомный архив и для чего он применяется?
13. Назовите особенности удаления файлов из архивов.

Лабораторная работа 9

ТАБЛИЧНЫЙ ПРОЦЕССОР MICROSOFT EXCEL (ОСНОВЫ РАБОТЫ)

9.1. Цель работы

1. Изучение основных приемов работы с табличным процессором MS Excel (размещение, хранение, модификация, обработка и удаление данных, а также построение простейших диаграмм).
2. Нарботка практических навыков по созданию и оформлению электронных документов с помощью табличного процессора MS Excel.

9.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [7, 8].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль. Основные сведения о табличном процессоре MS Excel:
 - ♦ возможности: ввод и составление формул и функций, использование списков, построение графиков, диаграмм и др.;
 - ♦ понятия: рабочий лист, рабочая книга, активная ячейка и др.;
 - ♦ операции: ввод данных, выделение блоков ячеек, присвоение им имен и т.д.
4. Практическая работа с процессором MS Excel (ввод числовой и текстовой информации в ячейки, редактирование, ввод и составление формул, построение диаграмм).
5. Выполнение индивидуального задания в соответствии с номером варианта.
6. Отчет о практической работе: выполнение на ПК операций по заданию преподавателя:
 - ♦ оформление отчета (с представлением готовых таблиц);
 - ♦ защита отчета.

9.3. Содержание отчета**ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:**

основные сведения о процессоре MS Excel:

- ♦ функции и понятия;
- ♦ перечень основных операций с таблицами (редактирование ячеек, ввод и составление формул, построение диаграмм) и их краткая характеристика;
- ♦ назначение элементов интерфейса (панели инструментов, ярлыки рабочих листов и т.д.).

ПРАКТИЧЕСКАЯ ЧАСТЬ:

подробный протокол выполнения индивидуального задания:

- ♦ модифицированная таблица в соответствии с индивидуальным заданием;
- ♦ перечень и порядок выполнения операций, соответствующих индивидуальному заданию;
- ♦ перечень действий, необходимых для выполнения каждой из операций.

9.4. Общие сведения о табличном процессоре MS Excel

Программные средства проектирования табличных форм документов в пакете Microsoft Office (MS Excel) получили название «табличный процессор». MS Excel — это компьютерная программа, предназначенная для обработки данных. Она позволяет не только создавать таблицы, но и автоматизировать обработку табличных данных.

К возможностям процессора относят:

- ◆ создание и редактирование электронных документов;
- ◆ получение выборки данных, удовлетворяющих определенным критериям;
- ◆ сортировку данных по определенному принципу;
- ◆ составление сводных отчетов с помощью нескольких способов определения итогов и объединения данных;
- ◆ построение графиков и диаграмм;
- ◆ обработку данных с использованием мощного аппарата функций и формул;
- ◆ исследование влияния различных факторов на данные (т.е. используя аналитические методы, можно вводить различные исходные данные в диаграммы рабочего листа и сравнивать получаемые результаты, давая тем самым ответ на вопрос, что будет, если...).

Основное достоинство и отличие MS Excel от других программ — простота использования средств обработки данных и их наглядность. Возможности электронных таблиц сравнимы с возможностями БД (базами данных). Работа с ними не требует от пользователя специальной подготовки в области программирования.

ОСНОВНЫЕ ЭЛЕМЕНТЫ MS EXCEL

Основными элементами MS Excel являются: рабочий лист, рабочая книга, ячейка.

Рабочий лист — множество *ячеек*, каждая из которых принадлежит некоторому столбцу и одновременно принадлежит некоторой строке. *Ячейка* — элементарная единица MS Excel. Листы используются для размещения, хранения, обработки и анализа данных. Информацию в рабочий лист легко вводить, изменять ее, удалять и добавлять новую. Введенные данные можно отредактировать. Строки и столбцы идентифицируются таким образом: столбцы именуются, а строки нумеруются. На рис. Л-9.1 показана часть рабочего листа. По умолчанию рабочий лист имеет 256 столбцов с именами от А до IV и 65 536 строк. Имя

столбца и номер строки однозначно определяют ячейку и являются ее именем. Имя также называется *адресом* или *ссылкой* на ячейку. Например, ячейка на рис. Л-9.1, выделенная рамочкой (*активная ячейка*), имеет адрес A1. Кроме того, ячейкам можно задавать собственные *имена* и использовать эти имена для ссылок на ячейки наряду с адресами. Файлы в MS Excel имеют расширение .xls. Команды работы с файлами собраны в пункте меню *Файл* основного меню. Основное рабочее пространство электронных таблиц — это *рабочая книга*, содержащая один или несколько рабочих листов.

	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						
7						
8						
9						
10						
11						
12						
13						
14						
15						

Рис. Л-9.1. Часть рабочего листа MS Excel

9.5. Работа с процессором MS Excel

ЗАПУСК ТАБЛИЧНОГО ПРОЦЕССОРА

Щелкните на командной кнопке *Пуск* панели задач Windows. В появившемся главном меню Windows выберите элемент *Программы* (появится еще одно меню), щелкните на элемент *Microsoft Excel* (рис. Л-9.2).



Рис. Л-9.2. Меню *Пуск* | *Программы* | *Microsoft Excel*

После запуска программы на экране появляется окно, представляющее собой пользовательскую оболочку или рабочую область MS

Excel (рис. Л-9.3). Окно программы разделено на две основные области: область управления (область команд) и рабочую область (область данных).

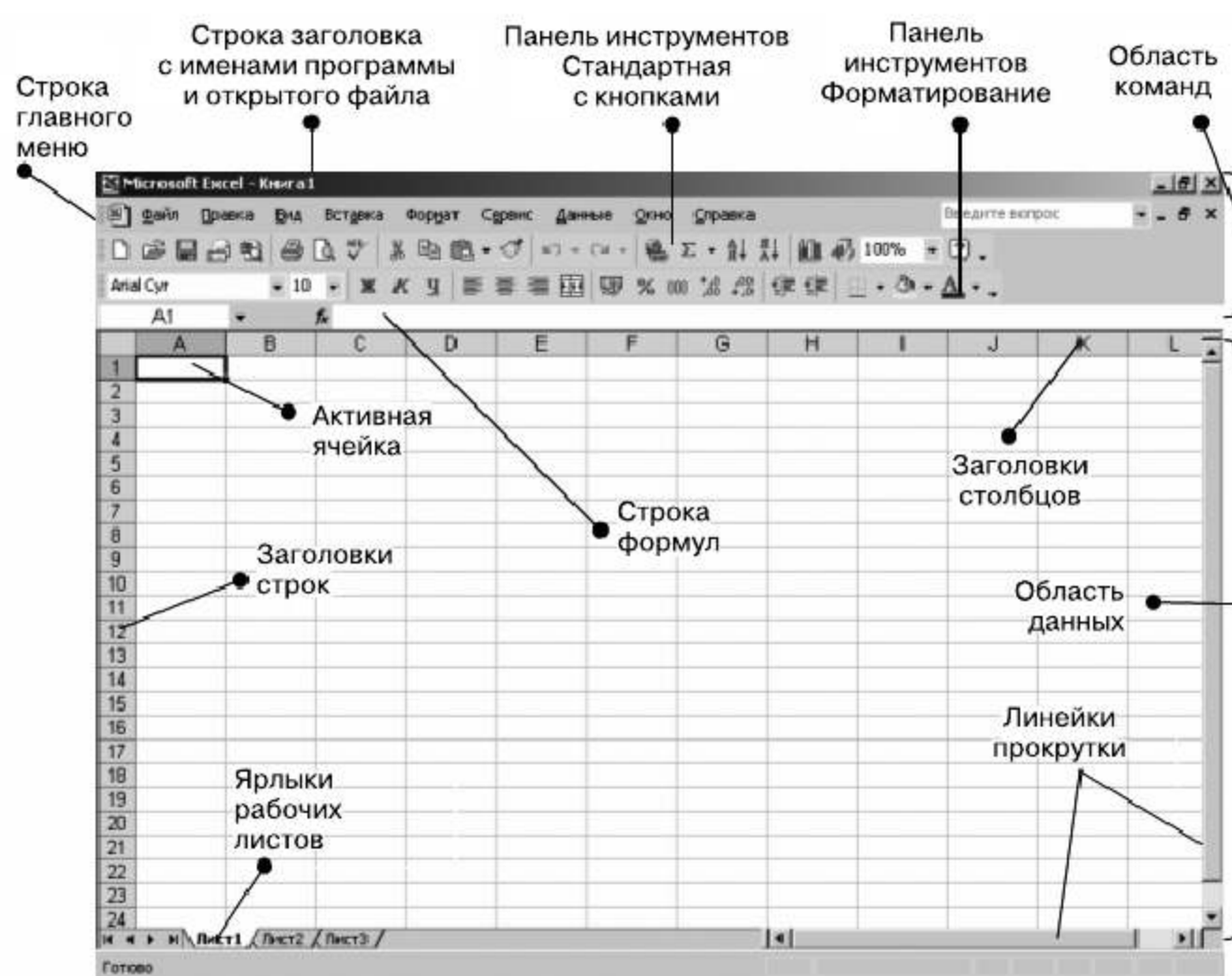


Рис. Л-9.3. Рабочая область MS Excel

ВВОД ИНФОРМАЦИИ

В любую ячейку рабочего листа можно вводить текст, числа, формулы.

Для ввода данных нужно выделить ячейку, в которую вводятся данные. Затем можно вводить данные двумя способами:

- 1) ввод данных в непосредственно выделенной ячейке. Курсор ввода находится внутри ячейки после ее выделения;
- 2) ввод данных в строке формул. Надо щелкнуть левой кнопкой мыши в строке формул и после этого вводить данные. Завершив ввод набранного текста в ячейку, можно нажать клавишу [Enter] или щелкнуть левой кнопкой мыши на любую другую ячейку.

По умолчанию текстовая информация ориентируется по левому краю ячейки, а числовые данные по правому.

ВЫДЕЛЕНИЕ БЛОКОВ

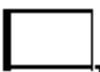
Для выделения блока из нескольких ячеек следует выполнить следующие действия:

- 1) поставить указатель мыши в левую верхнюю ячейку предполагаемого блока (курсор примет форму объемного крестика);
- 2) нажать левую кнопку мыши;
- 3) удерживая левую кнопку мыши, двигать указатель мыши по диагонали к правой нижней ячейке блока.

Первая ячейка остается светлой, остальные выделенные ячейки окрашиваются в темный цвет.

Для выделения строки щелкните левой кнопкой мыши на номере — заголовке строки.

Для выделения столбца щелкните левой кнопкой мыши на заголовке столбца.

Для выделения всего рабочего листа щелкните левой кнопкой мыши по кнопке , находящейся перед заголовком столбцов и над заголовком строк в левом верхнем углу рабочей книги.

9.6. Создание таблиц

Возможности MS Excel позволяют достаточно просто создавать весьма разные пользовательские таблицы. Создание пользовательских таблиц предполагает следующие этапы:

- 1) ввод шапки и данных таблицы;
- 2) форматирование шапки и данных таблицы.

Основная задача создания таблиц сводится к размещению данных в ячейках таблицы.

ВВОД ТЕКСТА ЗАГОЛОВКА ТАБЛИЦЫ

Условно все заголовки таблицы можно разбить на подуровни. Сначала нужно ввести общий заголовок шапки таблицы, а затем название полей уровня 1, затем уровня 2 и т.д. Заголовки одного уровня, как правило, должны находиться в одной строке и следовать друг за другом. Как бы ни была сложна устроена таблица, эти правила остаются неизменными (рис. Л-9.4).

Например:

Заголовок шапки таблицы (уровень 0)	Продано книг				Заголовок группы столбцов (уровень 1)
	В с в о	в том числе			
		Научных	Технических	Художест- венных	
Заголовок столбцов (уровень 2)	523	150	200	173	Область данных таблицы

Рис. Л-9.4. Заголовки таблицы

ФОРМАТИРОВАНИЕ ЯЧЕЕК

Общее правило форматирования выглядит следующим образом:

- 1) выделить область ячеек или одну ячейку;
- 2) выбрать из контекстного меню пункт *Формат ячеек* (появится окно формата ячеек с вкладками: *Число*, *Выравнивание*, *Шрифт*, *Граница*, *Вид*, *Защита*);
- 3) задать параметры форматирования на соответствующих вкладках и нажать кнопку *ОК*.

УСТАНОВКА ШРИФТА

Шрифт характеризуется следующими параметрами:

- 1) гарнитурой (комплект шрифтов, имеющих одинаковое начертание символов);
- 2) начертанием (обычный, курсив, полужирный и т.п.);
- 3) размером;
- 4) подчеркиванием;
- 5) цветом.

Задать параметры шрифта можно на вкладке *Шрифт* диалогового окна *Формат ячеек*.

ВЫРАВНИВАНИЕ И НАПРАВЛЕНИЕ СОДЕРЖИМОГО ЯЧЕЙКИ

Параметры выравнивания задаются на вкладке *Выравнивание* диалогового окна *Формат ячеек*. Под выравниванием содержимого ячейки будем понимать следующее:

- 1) выравнивание по горизонтали (например, по левому краю, правому краю, по центру и т.п.);
- 2) выравнивание по вертикали (например, по нижнему краю, по верхнему краю, по центру и т.п.);
- 3) ориентация (например, вертикальное расположение текста и т.п.);
- 4) отображение (например, переносить по словам и т.п.).

ОБЪЕДИНЕНИЕ ЯЧЕЕК

Чтобы объединить ячейки в одно целое, нужно выделить необходимый диапазон ячеек (см. подраздел 9.5. Выделение блоков). Затем вызвать правой клавишей мыши контекстное меню и выбрать пункт *Формат ячеек*, в появившемся окне щелкнуть левой клавишей мыши на вкладке *Выравнивание* и установить флажок на элементе *Объединение ячеек*.

ОБРАМЛЕНИЕ ЯЧЕЕК

Рамки могут быть хорошим средством оформления различных областей рабочего листа или привлечением внимания к важным ячейкам. MS Excel позволяет выбирать различные типы линий, их цвет и расположение. Можно применять несколько типов линий разных цветов к ячейке (диапазону ячеек). Например, сверху — двойная линия красного цвета, снизу — жирная линия синего цвета, справа и слева — тонкая сплошная линия зеленого цвета (рис. Л-9.5). Чтобы установить или изменить тип линии рамки или удалить рамку для выделенных ячеек, нужно воспользоваться вкладкой *Граница*. Для изменения типа рамки необходимо выбрать нужный тип линии и указать, к какой части диапазона он относится. Для удаления линии рамки необходимо отжать кнопку, соответствующую положению линии (например, слева

, справа , снизу , сверху  от ячейки).

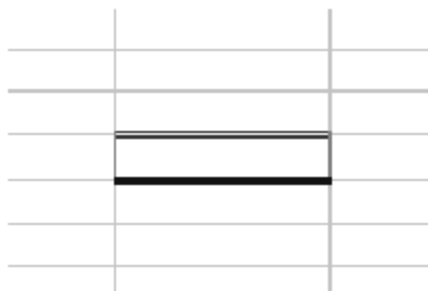


Рис. Л-9.5. Типы линий

ЗАПОЛНЕНИЕ ФОНА ЯЧЕЕК

Различные варианты заливки ячеек, предоставляемые вкладкой *Вид*, можно использовать для наглядного оформления рабочего листа. Например, заливку можно применять для выделения в листе итоговых данных или для привлечения внимания к предназначенным для ввода данных ячейкам рабочего листа. В качестве фона следует выбирать такой цвет, который позволяет легко читать текст и числовые значения.

Примечание. Все вышесказанное о форматировании ячейки распространяется на блоки ячеек.

ПРИСВОЕНИЕ ИМЕНИ ДИАПАЗОНУ ЯЧЕЕК

Чтобы присвоить имя какому-либо диапазону ячеек, нужно выделить необходимую область ячеек. Затем отметить в главном меню элемент *Вставка | Имя | Присвоить*, в появившемся диалоговом окне набрать имя области ячеек и нажать клавишу [Enter] или *OK*.

РЕДАКТИРОВАНИЕ ДАННЫХ

Редактировать данные можно двумя способами:

- 1) выделить ячейку и, щелкнув на *строке формул*, отредактировать данные в строке формул;
- 2) редактирование данных непосредственно в ячейке. Для этого надо щелкнуть в нужной ячейке, а затем, щелкнув еще раз, поместить текстовый курсор на то место, куда требуется внести изменения. Затем внести нужную информацию или удалить лишнее.

Примечание. Вызов контекстного меню происходит при нажатии правой кнопки мыши на выделенном объекте.

ВВОД ПОСЛЕДОВАТЕЛЬНОСТИ ДАННЫХ

Для ввода последовательности данных можно выделить блок ячеек и вводить отдельные элементы в эти ячейки.

Существует более простой способ для ввода в несколько ячеек последовательностей или прогрессий данных: достаточно ввести два первых элемента последовательности, а затем использовать *Автозаполнение* для ввода следующих элементов последовательности. Для этого нужно выделить блок ячеек с элементами последовательности, поместить курсор мыши в нижний левый угол блока (при этом курсор примет очертания черного крестика) и, удерживая левую клавишу мыши, распространить эту последовательность на необходимый диапазон ячеек. Еще один способ: выделить блок ячеек с элементами последовательности, затем в строке главного меню выбрать пункт *Правка*, в появившемся меню выбрать пункт *Заполнить | Прогрессия* и в появившемся диалоговом окне задать параметры прогрессии (рис. Л-9.6).

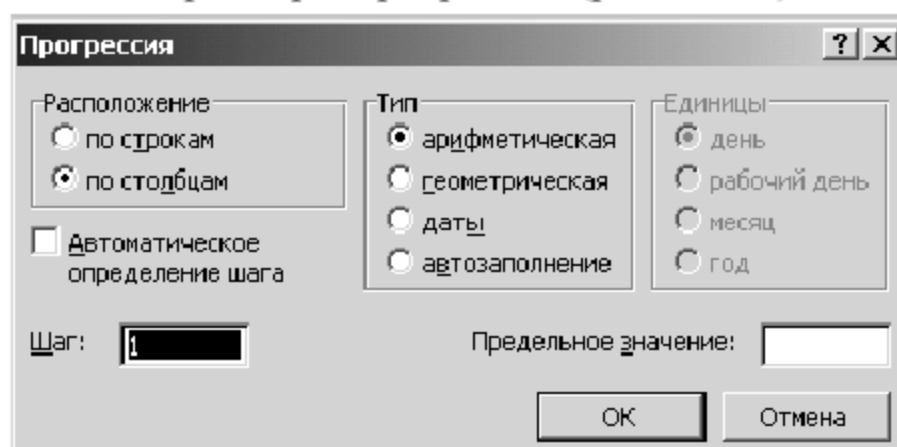


Рис. Л-9.6. Диалоговое окно *Прогрессия*

В MS Excel есть встроенный набор последовательностей, к которому можно добавить свои вновь создаваемые списки, такие как списки фамилий сотрудников фирмы или списки отделов.

9.7. Аппарат формул и функций MS Excel

Информацию, содержащуюся на рабочих листах, можно обрабатывать с помощью формул, состоящих из арифметических действий и встроенных функций.

Для упрощения работы с встроенными функциями при составлении формул используется *Мастер функций* — встроенный инструмент MS Excel. Для вызова мастера функций используется кнопка *Вставка функции*  на панели *Стандартная*.

АВТОМАТИЧЕСКОЕ СУММИРОВАНИЕ СТРОК И СТОЛБЦОВ

При работе часто возникает необходимость суммирования содержимого строк или столбцов. Кнопка *Автосумма* Σ на панели *Стандартная* автоматически применяет формулу суммирования к данным в строках и столбцах. Автосуммирование используется для следующих задач:

- ◆ суммирование данных в строках или столбцах ближайшего к текущей ячейке блока;
- ◆ суммирование данных в любом выделенном диапазоне ячеек;
- ◆ добавление итоговых сумм к ряду, содержащему частичные суммы.

Например, подсчитаем сумму чисел в ячейках.

1. Введем числа в ячейки (любые), например C7—C10, D7—D10.
2. Выделим ячейку D11 и нажмем Σ . Блок ячеек D7—D10 становится окруженным бегущей границей, а в строке формул появится формула =СУММ(D7:D10).
3. Для суммирования всех необходимых ячеек растянем границы рамки до нужного размера.
4. Результат вычисления находится в ячейке D11 (рис. Л-9.7).

12	545
43	454
343	5445
398	6444
=СУММ(C7:D10)	
СУММ(число1; [число2]; ...)	

Рис. Л-9.7. Результат вычислений

СОСТАВЛЕНИЕ ЭЛЕМЕНТАРНЫХ ФОРМУЛ

Для вычисления используются формулы, которые могут быть простыми, например сложение содержимого двух ячеек, а могут быть и более сложными, например определение отклонения какого-либо параметра от других значений последовательности.

Ввод формул в ячейках необходимо начинать с ввода знака равенства «=». Простейшие формулы состоят из двух частей: оператора и хотя бы одного адреса ячейки.

Примеры.

1. =A12+3 К содержимому ячейки A12 добавлено 3.
2. =(F23 – E40)*0.05 Из содержимого ячейки F23 вычитается содержимое ячейки E40, разность умножается на 0.05.
3. =МАКС(A12:A23) – МИН(G1:G10) Из максимального диапазона ячеек A12:A23 вычитается минимальное содержимое диапазона G1:G10.

СОСТАВЛЕНИЕ ФОРМУЛ С ПОМОЩЬЮ МАСТЕРА ФУНКЦИЙ

Любая функция представляет собой отдельную формулу, реализующую операцию над указанными данными. В одной формуле может использоваться несколько функций, обращение к одной функции может быть вложенным в обращение к другой функции. Функции могут иметь несколько аргументов (в таких случаях аргументы отделяются друг от друга запятыми).

Название функции можно набирать с клавиатуры или вводить функции автоматически с помощью *Мастера функций* — кнопка *Вставка функции*  на панели *Стандартная*. При работе с функцией, требующей несколько аргументов, трудно уследить за последовательностью ввода аргументов, поэтому *Мастер функций* поочередно запрашивает отдельные аргументы, необходимые для составления формулы, что значительно упрощает работу.

ИСПОЛЬЗОВАНИЕ ИМЕН В ФОРМУЛАХ

Имена блоков ячеек могут быть использованы в формулах вместо адресов ячеек. Имена делают формулы более удобными для понимания. Например, что вычисляет формула *МАКС [Имя диапазона (задается согласно п. 9.6, п/п Присвоение имени диапазону ячеек.)]*, более понятно, чем *МАКС(D12:D15)*.

Пример. Найдем максимальное значение.

1. Выделим диапазон ячеек C7:C10. Присвоим этому диапазону имя (п. 5).
2. Выделим ячейку C11 и щелкнем на кнопке *Мастер функций*. Откроется диалоговое окно.
3. В списке *Категория* выделим строку *Статистические*. Прокрутим список *Функция* и выделим имя функции *МАКС*. Эта функция определяет в выделенном блоке максимальное значение.

4. Щелчком *ОК*, откроется диалоговое окно *МАКС*.
5. В поле *Число 1* введем имя диапазона, в котором найдем максимальное значение. Щелчком *ОК*. Формула составлена, результат появился в ячейке *C11*.

ПЕРЕУПОРЯДОЧИВАНИЕ СОДЕРЖИМОГО ЯЧЕЕК

После ввода данных в ячейки рабочего листа как текста, так и чисел или формул внутри листа образуется некоторая структура, определяющая связь между ячейками.

Можно переупорядочивать данные: копировать данные из одних ячеек в другие, добавлять, удалять ячейки, строки, столбцы и перемещать данные между ячейками. Данные операции можно проводить двумя способами:

- 1) используя подпункты меню *Правка*;
- 2) с помощью контекстного меню, появляющегося при нажатии правой кнопкой мыши на выделенные ячейки.

9.8. Форматирование данных

Составляя документ, важно разместить в нем информацию с максимальной наглядностью. Оформление документов должно быть выполнено в едином стиле. Такие возможности предоставляют электронные таблицы.

АВТОМАТИЧЕСКОЕ ФОРМАТИРОВАНИЕ ДАННЫХ

Для включения режима автоформатирования следует:

- 1) выделить объекты (ячейку или блок ячеек), которые следует форматировать;
- 2) выбрать команду *Формат | Автоформат*.

Выбрать отдельные элементы форматирования можно с помощью команды *Формат ячеек* контекстного меню.

ФОРМАТИРОВАНИЕ

С ПОМОЩЬЮ ПАНЕЛИ ИНСТРУМЕНТОВ ФОРМАТИРОВАНИЕ

Если не подходит ни один из стандартных форматов или не хочется применять к части листа другие форматы данных, можно воспользоваться кнопками панели *Форматирование*.

♦ Как изменить формат чисел.

Задать стандартный формат для числа в ячейке можно с помощью кнопок панели инструментов *Форматирование*: *Денежный формат* , *Процентный формат* % и *Формат с разделителями* .

Примечание. При переходе к *Денежному формату* в ячейках вместо действительных величин могут появиться ряды значков #####. Появление этих значков указывает на то, что ширина столбцов таблицы недостаточна для записи числа в выбранном формате.

♦ *Как изменять шрифт.*

Можно изменять не только формат записи данных, но и стиль записи отдельных символов. Шрифты и их размер можно быстро изменять, используя поля *Шрифт* и *Размер шрифта* на панели инструментов.

Каждый шрифт имеет три модификации, которые задаются соответствующими кнопками этой же панели: *Полужирный*, *Курсив*, *Подчеркнутый*.

♦ *Как изменить выравнивание.*

По умолчанию в электронных таблицах происходит автоматическое выравнивание вводимого текста по левому краю, а числовых значений — по правому краю ячейки. Можно задать другие способы выравнивания. Текст в ячейках можно выравнивать с помощью кнопок панели инструментов.

♦ *Как добавлять рамки и изменять цвет фона ячеек.*

Отдельные ячейки или области листа можно выделить с помощью различных рамок, а также можно изменить цвет фона ячейки. Эти действия можно выполнить с помощью кнопок панели *Форматирование: Границы и Цвет заливки*. Границы позволяют выделять ячейки дополнительными линиями сверху, снизу, по бокам и вокруг ячейки или блока ячеек.

9.9. Построение диаграмм

Диаграммы можно построить как на одном листе с таблицей данных, так и на отдельном листе рабочей книги. Диаграмма, расположенная на листе с данными, называется *внедренной*. Размещение диаграммы на одном листе с данными позволяет одновременно видеть и данные, и их графическое представление.

Диаграмма, расположенная на отдельном листе, называется *листом диаграммы*.

КАК СОЗДАВАТЬ ДИАГРАММЫ

Для создания диаграммы на рабочем листе следует:

- 1) выделить данные, которые будут использованы при построении диаграммы;
- 2) вызвать *Мастера диаграмм*.

Вызов *Мастера диаграмм* осуществляется при нажатии кнопки  *Мастер диаграмм* панели инструментов *Стандартная*. Тип создаваемой диаграммы определяется пользователем и зависит от характера выделенных данных.

После выделения данных необходимо определить место, где будет расположена диаграмма. Если диаграмма будет внедренной, то следует предварительно определить размер области, в которую будет помещена диаграмма.

КАК УКАЗАТЬ ТИП ДИАГРАММЫ

Мастер диаграмм позволяет строить различные типы диаграмм, например круговые, линейчатые, гистограммы и т.д.

Линейчатые диаграммы и *гистограммы* удобны для сравнения величин в рядах данных. *Линейчатые* диаграммы обычно используются для сравнения данных по ряду параметров, *гистограммы* — для наблюдения изменений параметров во времени. *Круговые* диаграммы наглядно отображают соотношение частей и целого. *Диаграммы с областями и графики* позволяют наилучшим образом изобразить непрерывное изменение параметров во времени.

При создании диаграммы данные автоматически разделяются на *ряды данных* и *категории*. В диаграмму может быть внесена дополнительная поясняющая информация: метки легенды, обозначения осей, заголовки. Первый столбец и первая строка данных обычно принимаются за метки легенды. *Легенда* содержит пояснения (названия) к рядам данных.

ПРИМЕР ПОСТРОЕНИЯ ДИАГРАММЫ

1. Ввести в ячейки (любые) какие-нибудь числа, например 10, 20, 25, 40, 60 (рис. Л-9.8).

	А
1	
2	10
3	20
4	25
5	40
6	60

Рис. Л-9.8. Введение чисел в ячейки

2. Выделить ячейки с данными и нажать *Мастер диаграмм*.
3. В появившемся диалоговом окне *Мастер диаграмм* выбрать нужный Тип диаграммы (например, Гистограмма). Нажать *Далее* (рис. Л-9.9).



Рис. Л-9.9. Мастер диаграмм, шаг 1

4. В появившемся окне можно выбрать опцию диаграммы *Ряды в строках* или *столбцах*. В поле *Диапазон данных* автоматически заносятся абсолютные адреса выделенных данных. Нажать *Далее* (рис. Л-9.10).

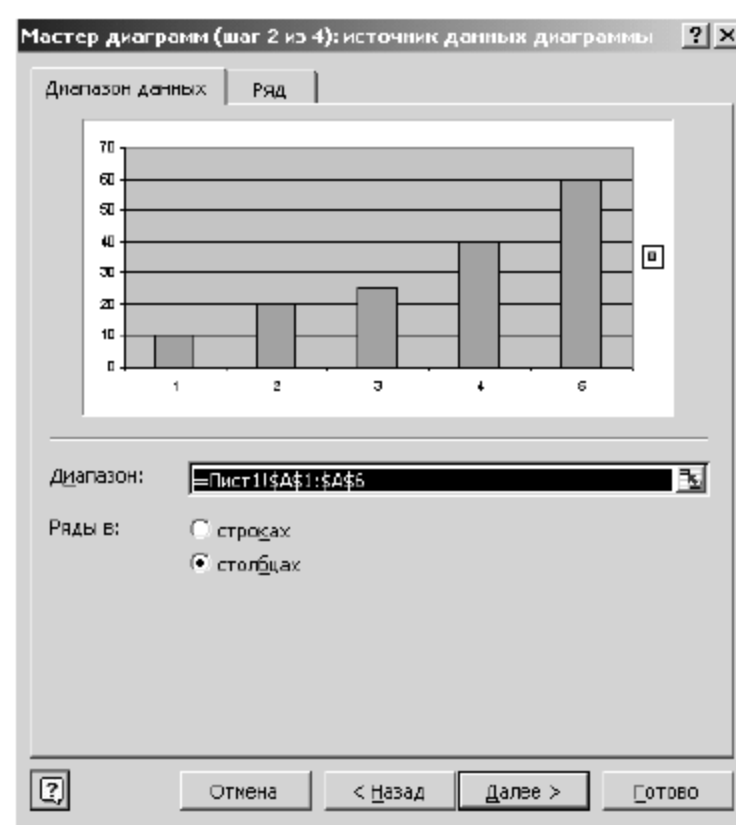


Рис. Л-9.10. Мастер диаграмм, шаг 2

5. В появившемся окне можно задать параметры диаграммы, такие как *Название диаграммы*, *Ось X* и т.д. (рис. Л-9.11).

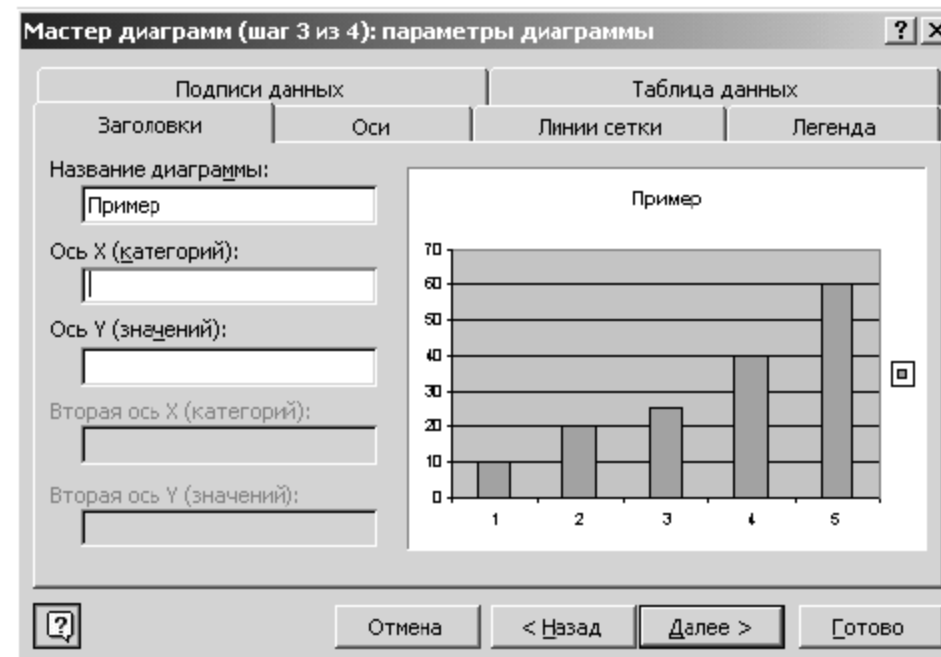


Рис. Л-9.11. Мастер диаграмм, шаг 3

6. Далее диаграмму можно разместить либо на отдельном листе, либо на листе с данными (рис. Л-9.12).

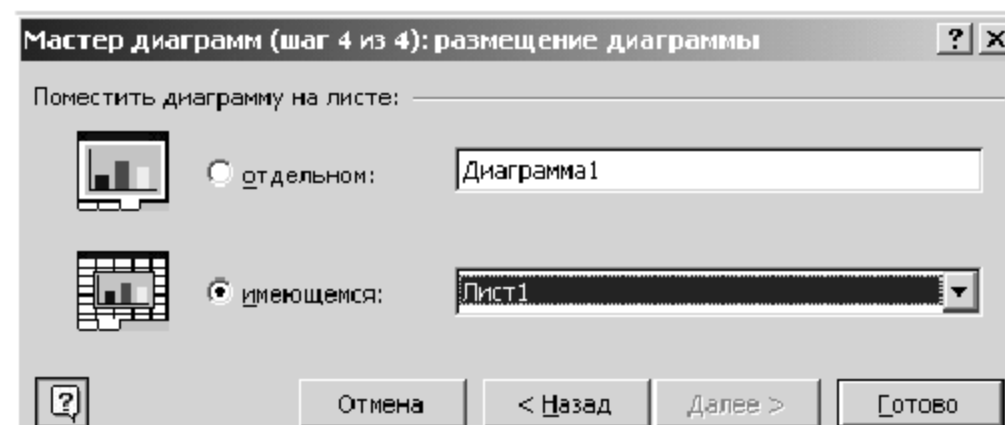


Рис. Л-9.12. Мастер диаграмм, шаг 4

Нажать кнопку *Готово*. На экране появится диаграмма (рис. Л-9.13).

Примечание. Готовую диаграмму можно редактировать.

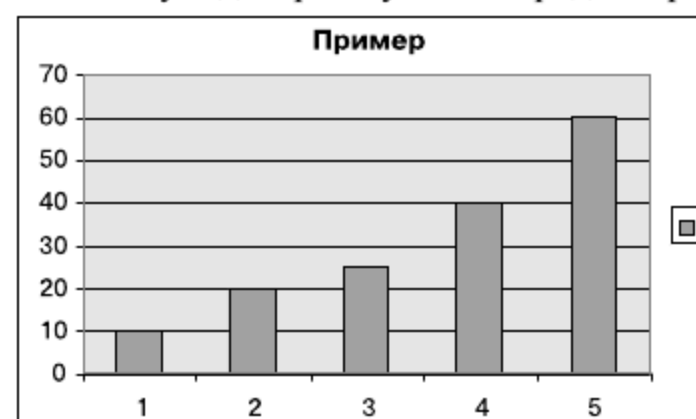


Рис. Л-9.13. Диаграмма

9.10. Сохранение файла

После выполнения всех заданий необходимо сохранить свой файл. Для этого нужно выполнить следующие действия:

- 1) выбрать в основном меню **Файл** пункт *Сохранить как...*;
- 2) в появившемся окне *Сохранение документа* выбрать: *Мой компьютер | Локальный диск (D:) | USERS | ** (папка с названием вашей группы);
- 3) в поле *Имя файла*: ввести название вашего файла;
- 4) нажать [Enter] или щелкнуть левой клавишей мыши по кнопке *Сохранить*.

9.11. Пример выполнения задания

Составить и отформатировать рабочий лист с данными о балансе какой-либо фирмы.

СОЗДАНИЕ РАБОЧЕГО ЛИСТА

1. Создать новый документ.
Меню **Файл** | **Создать** | *Чистая книга*.
2. Переименовать рабочий лист в *Баланс*. Правой кнопкой мыши щелкнуть на *Лист1* (в левом нижнем углу книги). В появившемся контекстном меню выбрать *Переименовать* и ввести название *Баланс* (рис. Л-9.14).

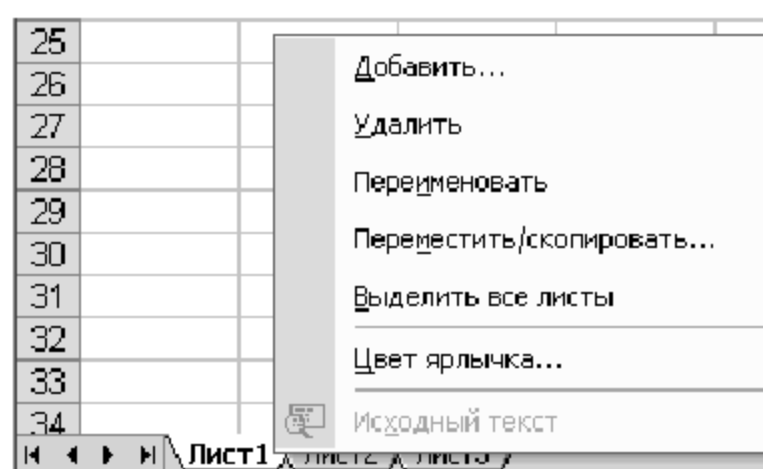


Рис. Л-9.14. Контекстное меню

3. Выделить ячейку A1 и ввести текст: «Фирма Р. годовой баланс на 1.01.2008» [Enter].
4. Выделить ячейку A3 и ввести: *Составил*, в ячейку B3 ввести свою фамилию, [Enter].
5. Выделить ячейку A4 и ввести: *Дата*, [Enter].

6. В ячейке A6 набрать: *Исходные*, [Enter].
7. В ячейке A9 набрать: *Баланс*, [Enter].
8. Выделить блок ячеек B6:C8.
Для этого поместите указатель (он имеет вид крестика) в ячейку B6 и, нажав левую кнопку мыши, перетащите указатель до ячейки C8.
Примечание. Между ячейками выделенного блока можно перемещаться с помощью клавиши [Tab].
9. В ячейке B6 набрать: *Темпы роста*, [Tab], [Tab], *Рост объема продаж*, [Tab], 1,5, [Tab], *Удорожание товаров*, [Tab], 0,9, [Tab], [Enter].
10. Снять выделение. Для этого щелкнуть на любое свободное место.
При правильно выполненных действиях должна получиться таблица, показанная на рис. Л-9.15.

	А	В	С	Д
1	Фирма Р. годовой баланс на 1.01.2004			
2				
3	Составил			
4	Дата			
5				
6	Исходные	Темпы роста		
7	данные	Рост объе	1,5	
8		Удорожан	0,9	
9	БАЛАНС			
10				
11				

Рис. Л-9.15. Пример составления таблицы для рабочего листа

11. В меню *Сервис | Параметры | Списки* выбрать строку *Новый список*.
12. Щелкнуть *Элементы списка* и набрать: *Актив*, [Enter], *Основные средства*, [Enter], *Материалы*, [Enter], *Касса*, [Enter], *Расчетный счет*, [Enter].
13. Нажать *Добавить*, после чего будет создан *Новый список* (рис. Л-9.16).
14. Повторив п. 11, в пункте *Элементы списка* набрать: *Пассив*, [Enter], *Уставный фонд*, [Enter], *Нерасп. прибыль*, [Enter], *Расч. по оплате труда*, [Enter], *Расч. с поставщиками*, [Enter], *Краткосроч. кредиты*, [Enter].
15. Нажать *Добавить* (появится новый список).
16. В ячейке B11 набрать *Актив*, [Enter].

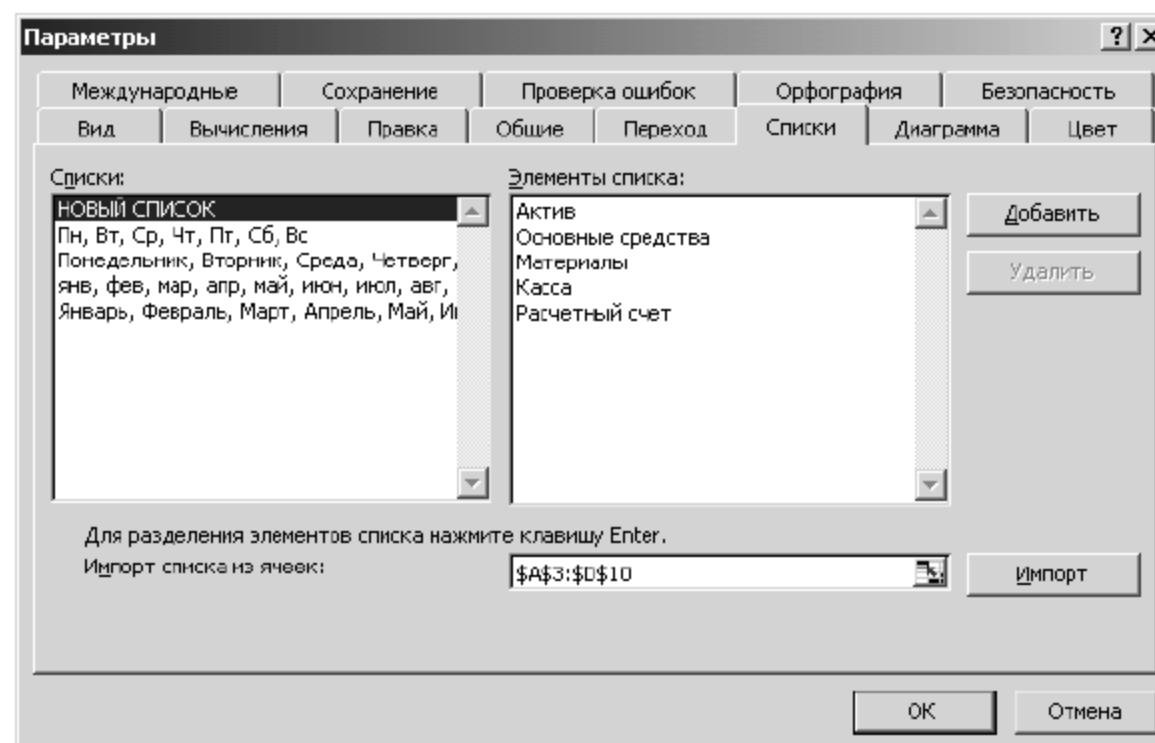


Рис. Л-9.16. Диалоговое окно *Параметры*.
Создание *Нового списка*

17. В ячейке E11 набрать *Пассив*, [Enter].
 18. Выделить ячейку B11 и перетащить маркер заполнения вниз на четыре строки к ячейке B15. Выделенные ячейки автоматически заполнятся введенной последовательностью заголовков.
 19. Выделить ячейку E11 и перетащить маркер заполнения вниз на пять строк к ячейке E16. Выделенные ячейки автоматически заполнятся введенной последовательностью заголовков.
- При правильном исполнении получится таблица, показанная на рис. Л-9.17.

	A	B	C	D	E	F
1	Фирма Р. годовой баланс на 1.01.2004					
2						
3	Составил					
4	Дата					
5						
6	Исходные	Темпы роста				
7	данные	Рост объе	1,5			
8		Удорожан	0,9			
9	БАЛАНС					
10						
11		1 Актив			Пассив	
12		2 Основные средства			Уставный фонд	
13		3 Материалы			Нерасп. прибыль	
14		4 Касса			Расч. по оплате труда	
15		5 Расчетный счет			Расч. с поставщиками	
16		6			Краткосроч. кредиты	
17						

Рис. Л-9.17. Пример дополнения таблицы для создания рабочего листа

20. В ячейке C11 набрать *Номер счета*.
 21. В ячейке D11 набрать *Сумма*.
 22. Выделить ячейки C11, D11. В контекстном меню выбрать команду *Копировать*. По контуру выделенных ячеек появится «бегущая» граница.
 23. Выделить ячейки F11, G11. В контекстном меню выбрать команду *Вставить*.
 24. В поля *Ном. счета* и *Сумма* ввести числовые данные (любые).
 25. Просуммировать значения в графах *Сумма*.
 26. Выделить ячейку D17 и нажать автосуммирование Σ , после чего результат должен появиться в ячейке D17.
 27. Пункт 26 повторить для ячейки G17.
- Результат показан на рис. Л-9.18.

	A	B	C	D	E	F	G
1	Фирма Р. годовой баланс на 1.01.2004						
2							
3	Составил	Фамилия					
4	Дата	08.08.2004					
5							
6	Исходные	Темпы роста					
7	данные	Рост объем	1,5				
8		Удорожани	0,9				
9	БАЛАНС						
10							
11	1	Актив	Ном.счета	Сумма	Пассив	Ном.счета	Сумма
12	2	Основные с	1	143200	Уставный	85	171900
13	3	Материалы	10	25200	Нерасп. п	88	900
14	4	Касса	50	20	Расч. по с	70	2400
15	5	Расчетный	51	9580	Расч. с по	60	1100
16	6				Краткосро	90	1700
17				178000			178000

Рис. Л-9.18. Результат создания рабочего листа

ФОРМАТИРОВАНИЕ СОЗДАННОГО ЛИСТА

1. Выделить ячейки A1:C4, выбрать команду *Формат | Автоформат*. В открывшемся диалоговом окне, в окне *Список форматов* выбрать строку *Цветной 2* и щелкнуть *ОК*.
2. Выделить ячейки B11:G18, в меню *Формат | Автоформат* в списке форматов выбрать *Классический 3* и щелкнуть на окне *Параметры*. В группе *Изменить* щелкнуть на флажках *формат чисел* и *ширину и высоту* (рис. Л-9.19).

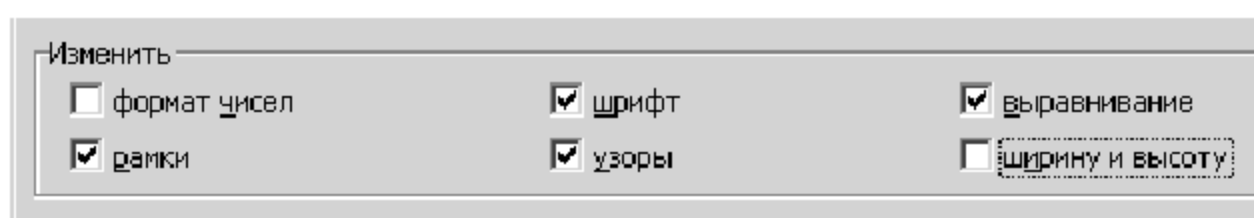
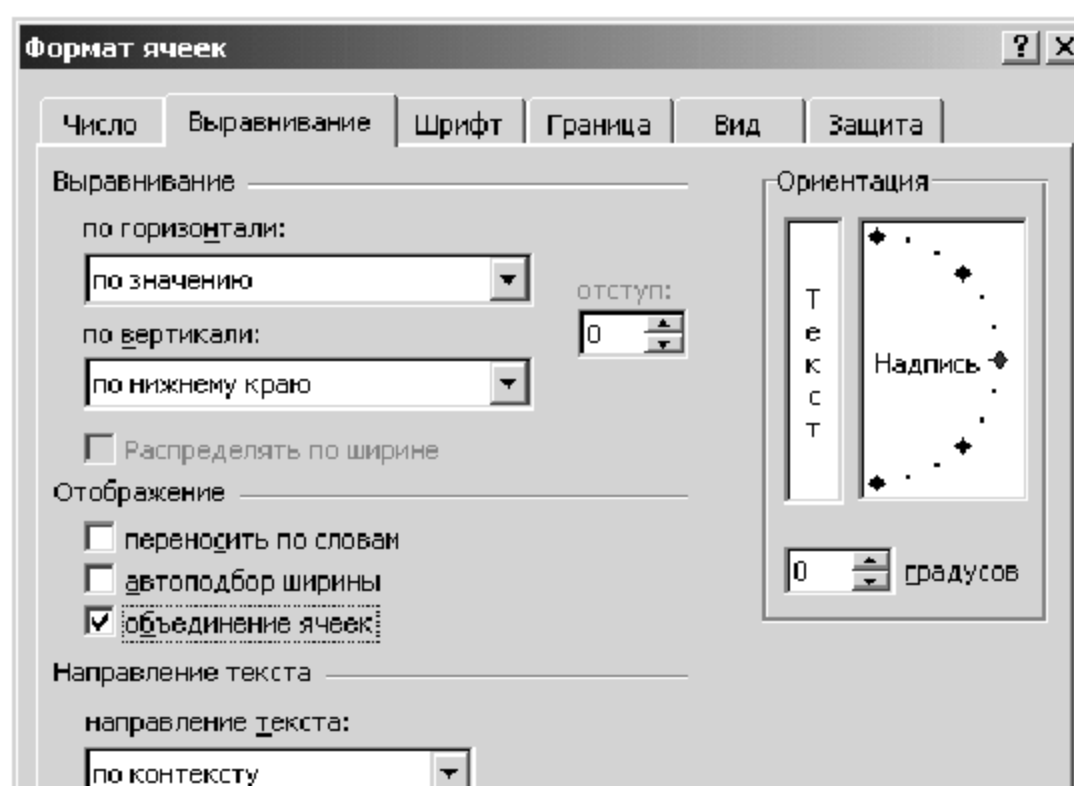


Рис. Л-9.19. Изменение параметров формата

3. Скопировать форматы на другие ячейки.
4. Выделить ячейку C11 (находится в выделенной области) и нажать кнопку  *Формат по образцу* (границы этой ячейки выделяются «бегущей» границей). Выделить блок ячеек A6:A9 (выбранный блок получает тот же формат, что и ячейка C11).
5. Удалить строки 6—8, для чего нажав [Ctrl], выделить заголовки этих строк. Далее в контекстном меню выбрать *Очистить содержимое*.
6. Слово «Баланс» поместить в середину листа. Для этого выделить ячейки A9:G9, в контекстном меню выбрать пункт *Формат ячеек*. В открывшемся окне выбрать пункт *Выравнивание* и поставить флажок в графе *Объединить ячейки* (рис. Л-9.20). Далее поставить *Выравнивание по центру*.

Рис. Л-9.20. Диалоговое окно *Формат ячеек*

7. Увеличить ширину 9-й строки. Для этого подвести курсор к верхней или нижней границе в заголовке строки (курсор

примет вид крестика) и, зажав левую клавишу, передвинуть границу.

8. Увеличить шрифт слова «Баланс». Для этого правой кнопкой мыши вызвать контекстное меню, выбрать пункт *Формат ячеек | Шрифт*. В появившемся окне можно менять стиль, размер и цвет текста. В графе *Размер* выбрать 14.
9. Увеличить ширину столбцов (чтобы все надписи были видны полностью). Для этого подвести курсор к любой границе каждого из столбцов и передвинуть границы до нужных размеров.

Получившаяся таблица представлена на рис. Л-9.21.

	A	B	C	D	E	F	G
1	Фирма Р. годовой баланс на 1.01.2004						
2							
3	Составил	Фамилия					
4	Дата	08.08.2004					
5							
6							
7							
8							
9	БАЛАНС						
10							
11		1	Актив	Ном.счета	Сумма	Пассив	Ном.счета
12		2	Основные средства	1	143200	Уставный фонд	85
13		3	Материалы	10	25200	Нерасп. прибыль	88
14		4	Касса	50	20	Расч. по оплате труда	70
15		5	Расчетный счет	51	9580	Расч. с поставщиками	60
16		6				Краткосроч. кредиты	90
17					178000		178000
18							
19							

Рис. Л-9.21. Окончательный вид таблицы

9.12. Практическая работа на ПК

Внимание! Если в процессе выполнения работы возникают вопросы, то их можно разрешить с помощью справки, которая вызывается клавишей [F1], или обратившись к литературе по данной теме.

Задание

Для каждой бригады:

1. Создать таблицу по образцу табл. Л-9.1.
2. Посчитать средний балл группы по каждому предмету с помощью функции *Среднее значение*.
3. Выполнить индивидуальные задания в соответствии с таблицей Л-9.2.

Таблица Л-9.1

Образец составления таблицы

Фамилии студентов группы	Предмет		
	Физика	Математика	Русский язык
1. Иванов			
2. Петров			
...			
26. Сидоров			
Средний балл			

Таблица Л-9.2

Индивидуальные задания

№ бригады	Номера индивидуальных заданий
1	1, 3, 5, 9, 10, 12, 16, 17, 18
2	2, 4, 6, 8, 9, 11, 14, 15, 18
3	1, 2, 6, 7, 10, 12, 16, 17, 18
4	3, 5, 7, 11, 14, 15, 16, 17, 18
5	1, 5, 6, 10, 12, 13, 14, 18
6	2, 4, 5, 7, 9, 11, 13, 15, 18
7	4, 6, 8, 9, 10, 12, 14, 16, 17, 18
8	3, 4, 6, 7, 9, 13, 14, 15, 18
9	5, 6, 7, 8, 9, 11, 12, 16, 18
10	1, 2, 5, 6, 9, 10, 12, 13, 18

ИНДИВИДУАЛЬНЫЕ ЗАДАНИЯ

1. Изменить стиль заголовка на *Классический 2*.
2. Выделить список имен красным цветом. Цвет текста в этих ячейках сделать синим.
3. Оценку по физике произвольно выбранного студента распространить на ячейки с другими предметами этого же студента с помощью функции *Автозаполнение*.
4. Изменить шрифт столбца фамилий с текущего на курсив.
5. Изменить стиль ячеек с названиями предметов на *Цветной 1*.
6. Средний балл по какому-либо предмету (для остальных это функция *Среднее значение*) посчитать с помощью функции *Автосумма*: результат автосуммирования поделить на число студентов.
7. Обрамить ячейку, в которой находится текст: *Предмет/Фамилии студентов группы*.
8. Поменять выравнивание в столбце с оценками по *Русскому языку* (например: выровнять по центру).
9. Поменять в столбце *Математика* ориентацию текста.

10. Увеличить ширину столбца с фамилиями в 2 раза.
11. Увеличить ширину строки с названиями предметов в 1,5 раза.
12. По оценкам какого-либо студента построить диаграмму (сохранить на том же листе).
13. Добавить к таблице столбец с информацией о наличии медали (*есть/нет*).
14. Переименовать *Лист1*.
15. По оценкам какого-либо студента построить диаграмму (сохранить на отдельном листе).
16. В поле *Ось X(категорий)* ввести: *Названия предметов*.
17. В поле *Ось Y(значений)* ввести: *Баллы*.
18. Сохранить файл под своим именем (действия выполняются по аналогии с действиями по сохранению файла в текстовом редакторе *Microsoft Word*).

Контрольные вопросы

1. Что такое табличный процессор?
2. Назовите возможности табличного процессора MS Excel.
3. В чем заключается основное достоинство MS Excel?
4. Что такое рабочий лист, ячейка, рабочая книга?
5. Какие элементы имеет окно программы MS Excel?
6. Для каких задач в MS Excel используется автосуммирование?
7. Как выделить одну или несколько ячеек таблицы?
8. Как выполняется форматирование шрифта в таблице?
9. Каким образом выполняется обрамление ячеек рамками?
10. Какие типы диаграмм вам известны? Охарактеризуйте их.

Лабораторная работа 10

РАБОТА В ИНТЕРНЕТЕ

(ПОИСК ИНФОРМАЦИИ. ЭЛЕКТРОННАЯ ПОЧТА)

10.1. Цель работы

1. Знакомство с услугами Интернета.
2. Практическая работа по использованию услуг Интернета.
3. Нарботка навыков работы в сети Интернета.

10.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1—3, 8].
2. Подготовка отчета (дома).

3. Индивидуальный входной контроль (цель работы, понятие Интернета, технология работы Internet Explorer).
4. Работа на компьютере:
 - ♦ освоение основных приемов по созданию электронного ящика и поиска информации;
 - ♦ выполнение индивидуального задания в соответствии с заданным вариантом.
5. Отчет о проделанной работе:
 - ♦ индивидуальная работа на ПК по заданию преподавателя;
 - ♦ оформление отчета (дорабатывается по результатам практической работы на ПК);
 - ♦ защита отчета.

10.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ♦ основные сведения об Интернете: определение, основные понятия (браузер, WWW, электронная почта и др.), основные услуги.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ♦ подробный протокол выполнения операций, особенности их выполнения.

10.4. Основные сведения об Интернете

ОПРЕДЕЛЕНИЕ ИНТЕРНЕТА

Интернет представляет собой глобальную компьютерную сеть, т.е. совокупность взаимосвязанных компьютерных сетей, разбросанных по всему миру. Само ее название означает «между сетей». Это сеть, соединяющая отдельные сети.

Если компьютер или какая-либо небольшая сеть может «говорить» с другими машинами на языке Интернета (или, выражаясь техническим языком, подключаться с применением протоколов), то этот компьютер считается включенным в сеть Интернета.

Интернет обеспечивает обмен информацией между всеми компьютерами, которые входят в сети, подключенные к ней.

К Интернету подключены практически все мировые организации (более трех тысяч университетов, Кремль, Белый Дом и т.д.).

Логическая структура Интернета представляет собой некое виртуальное объединение, имеющее свое собственное информационное пространство.

Схема подключения локальной сети (ЛС) к Интернету приведена на рис. Л-10.1, где в роли рабочей станции используется любой из компьютеров лаборатории.

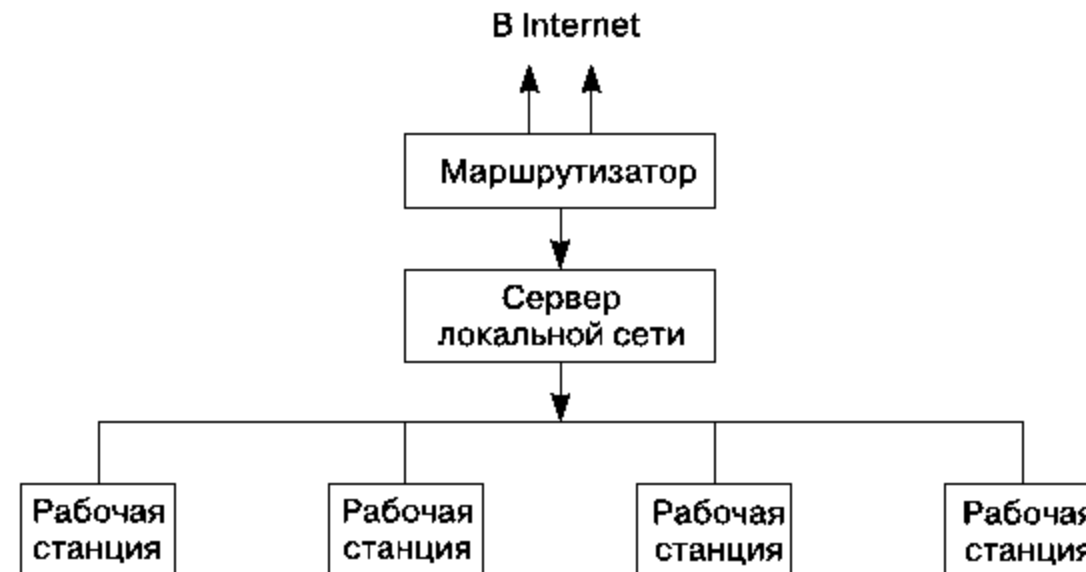


Рис. Л-10.1. Схема подключения ЛС к Интернету

WWW (World Wide Web — «всемирная паутина») — подсистема Интернета, являющаяся всемирной распределенной базой текстовых документов, хранящихся на компьютерах-серверах. Поставщиком документов может стать любой человек, имеющий компьютер, подключенный к Интернету.

Web-сервером является компьютер, хранящий на своих дисках Web-страницы.

Web-страница — это документ Web, содержащий информацию в виде текста, рисунков, фотографий, фрагментов аудио- и видеозаписей. Web-страницы записываются на языке программирования HTML (HyperText Markup Language) и сохраняются в виде файлов с расширением .html или .htm. Web-страницы обычно содержат гиперссылки — выделенные цветом и подчеркиванием фрагменты текста или графические изображения в цветных рамках, которые позволяют при нажатии на них переходить к другим страницам Web без ввода уникального адреса (*URL — Uniform Resource Locator*) этих страниц.

Для связи с Интернетом используется специальная программа — *браузер*. Первоначально браузеры предназначались для просмотра документов с Web-серверов, но конкуренция между производителями программного обеспечения привела к тому, что в них появилось много дополнительных возможностей. В результате в современных браузерах объединяются все возможные приложения для доступа в Интернет. Се-

годня наиболее популярными браузерами являются *Opera*, выпускаемый фирмой Opera Software, и *Internet Explorer* фирмы Microsoft. Оба пакета одинаково хороши для работы в Интернете, поддерживают различные службы Интернета, в том числе HTTP, FTP, программы чтения новостей, электронную почту, а также более старые протоколы.

Для обращения к серверу с помощью браузера необходимо ввести полное доменное имя этого сервера. Например, если нужно связаться с одним из известнейших российских поисковых серверов, необходимо ввести следующий адрес: <http://yandex.ru>.

Свой адрес в Интернете также имеет и Рязанский государственный радиотехнический университет: <http://rgrtu.ryazan.ru>.

УСЛУГИ ИНТЕРНЕТА

Интернет предоставляет пользователю более десяти основных услуг, каждая из которых имеет свой особый протокол. Все услуги предоставляются по архитектуре «клиент—сервер» (рис. Л-10.2). В распределенных локальных и глобальных вычислительных системах *сервер* — компьютер, предоставляющий те или иные общие ресурсы (БД, файловые системы, различные сервисы, принтеры и т.д.), *клиент* — компьютер, использующий общие ресурсы. Наиболее распространенными услугами считаются *обеспечение доступа к требуемой информации, электронная почта, передача, прием, обмен информацией*.

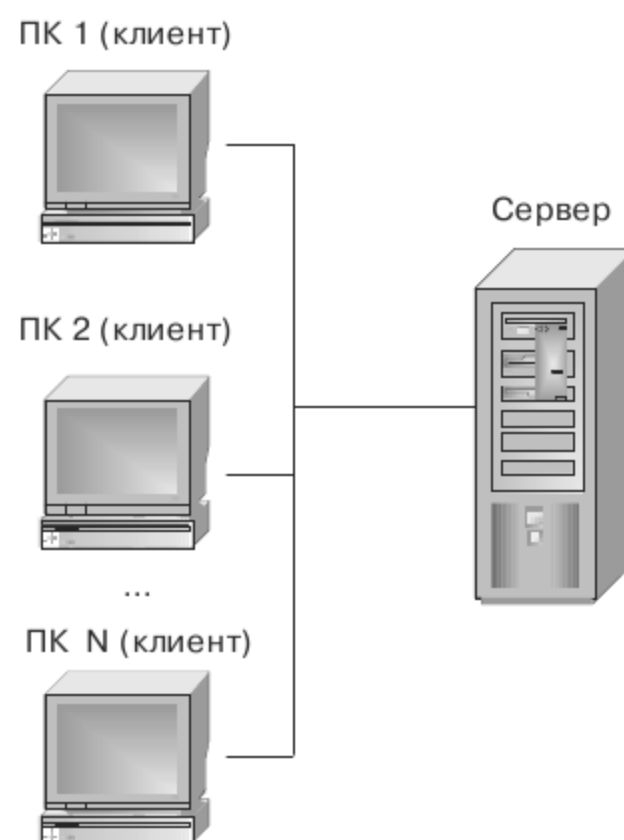


Рис. Л-10.2. Типовая архитектура клиент—сервер

Поиск информации. Поиск информации в Интернете осуществляется через службу WAIS (Wide Area Information Service — служба поиска распределенной информации). Эта служба обеспечивает поиск в архивах Интернета статей, содержащих указанные группы слов.

Электронная почта. Сегодня электронная почта похожа на обычную почтовую службу и обеспечивает:

- ◆ быструю доставку сообщений (минимальное время доставки достигает нескольких секунд); возможность общения в режиме on-line;
- ◆ аппаратную и программную независимость отправителя и получателя сообщений;
- ◆ возможность передачи не только обычных текстовых сообщений, но и файлов, содержащих Web-страницы, отформатированный текст, графику, звук и т.д.;
- ◆ доступ ко всем информационным услугам и базам данных Интернета (Usenet, World Wide Web, FTP, Gopher, WAIS и др.);
- ◆ доступ ко всем другим международным сетям (даже к тем, которые вообще не имеют on-line сервиса).

При работе с электронной почтой чаще всего выполняются следующие операции:

- ◆ запуск почтовой программы;
- ◆ отправка сообщений;
- ◆ чтение входящих сообщений;
- ◆ ответ на поступившие сообщения;
- ◆ удаление сообщений;
- ◆ выход из почтовой программы.

Передача, прием, обмен информацией. Данная услуга отличается от электронной почты тем, что при передаче, приеме и обмене информацией не обязательно иметь почтовый ящик. К разновидностям данной услуги относят: сохранение информации на жестком диске, чтение электронных газет, торги на бирже через Интернет и т.п.

10.5. Практическая работа на ПК

ОТКРЫТИЕ БРАУЗЕРА И ВЫХОД В СЕТЬ ИНТЕРНЕТА

Задание

Откройте браузер Internet Explorer, для чего выполните следующие действия:

- ◆ щелкните левой кнопкой мыши по кнопке *Пуск* и в появившемся меню выберите категорию *Программы*;

В поле *Адрес* (рис. Л-10.4) введите адрес сервера почтовых услуг (в рассматриваемом случае front.ru) и нажмите клавишу [Enter].

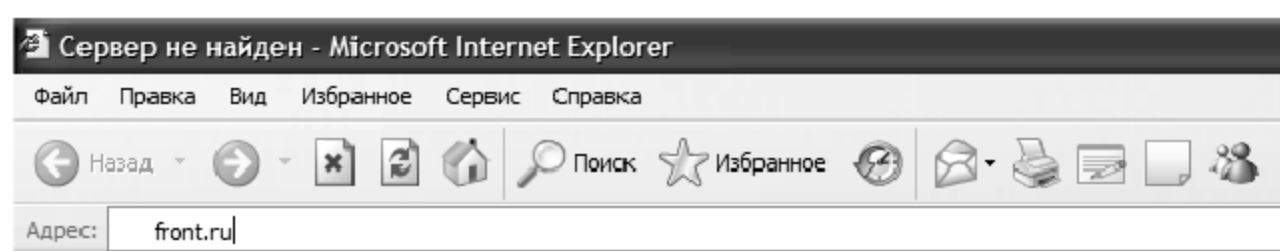


Рис. Л-10.4. Поле *Адрес*

2. Произведите регистрацию на сервере.

После загрузки сайта щелкните по надписи *Регистрация*. Регистрация на почтовом сервере front.ru происходит в три этапа. Сначала предлагается ввести последовательность цифр, которую пользователь видит на экране (процедура для защиты от автоматической регистрации). На втором и третьем этапах вводится имя почтового ящика (*login*), а также пароль доступа к нему (*password*). После выполнения их появится сообщение об успешном создании почтового ящика или сообщение о том, что почтовый ящик с таким именем существует (в этом случае будет предложено вновь выполнить два последних этапа).

На других серверах, предоставляющих услуги электронной почты, процедура регистрации происходит подобным образом.

ПОИСК ИНФОРМАЦИИ

В Интернете можно найти практически любую нужную информацию — от старых раритетных изданий до новых книг, рефератов и свежих новостей.

Для поиска информации необходимо:

1) *ввести адрес поисковой системы.*

Для этого в поле *Адрес* браузера нужно ввести адрес сайта, на котором существует поисковая система (rambler.ru, mail.ru, yandex.ru, yahoo.ru, google.ru и т.д.), затем нажать клавишу [Enter];

2) *осуществить поиск необходимой информации.*

На появившейся электронной странице в поле *Поиск* (*Найти*, *Искать* и т.п.) ввести ключевые слова для поиска. Через несколько секунд появится страница с результатами поиска, содержащая список сайтов, на которых содержится информация, имеющая связь с введенными словами.

Задание

Найдите реферат о Столыпинской аграрной реформе и сохраните его на жестком диске, а также сохраните любую страницу сайта, на котором содержится данный реферат.

Введите адрес поисковой системы rambler.ru в соответствующее поле браузера и нажмите [Enter]. На появившейся странице в поле *Искать* введите ключевые слова, позволяющие найти необходимый реферат: *реферат Столыпинская аграрная реформа*. Затем нажмите клавишу [Enter]. В появившемся списке (рис. Л-10.5) выполните щелчок по одной из ссылок на сайт, который содержит искомый реферат (один и тот же реферат может быть на нескольких сайтах), а также предоставляет возможность его сохранения на жестком диске пользователя — возможность «скачать».

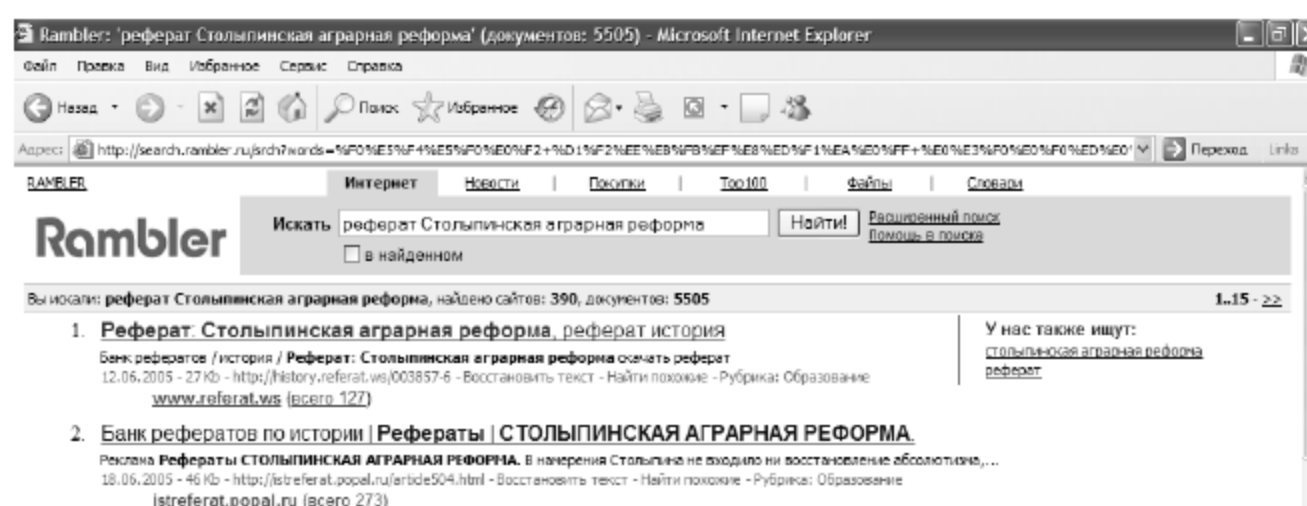


Рис. Л-10.5. Результаты поиска

В новом окне Internet Explorer, отображающем одну из страниц некоторого сайта (в рассматриваемом случае это www.referat.ws), щелкните левой клавишей мыши по кнопке *Скачать* (*Загрузить* и т.п.). Далее будет предложено указать папку для сохранения. Выберем D:\USERS* (номер группы). Чтобы сохранить страницу сайта, в меню браузера *Файл* выберите команду *Сохранить как...* (рис. Л-10.6), далее укажите путь сохранения. После сохранения реферата и Web-страницы на жестком диске закройте окно браузера.

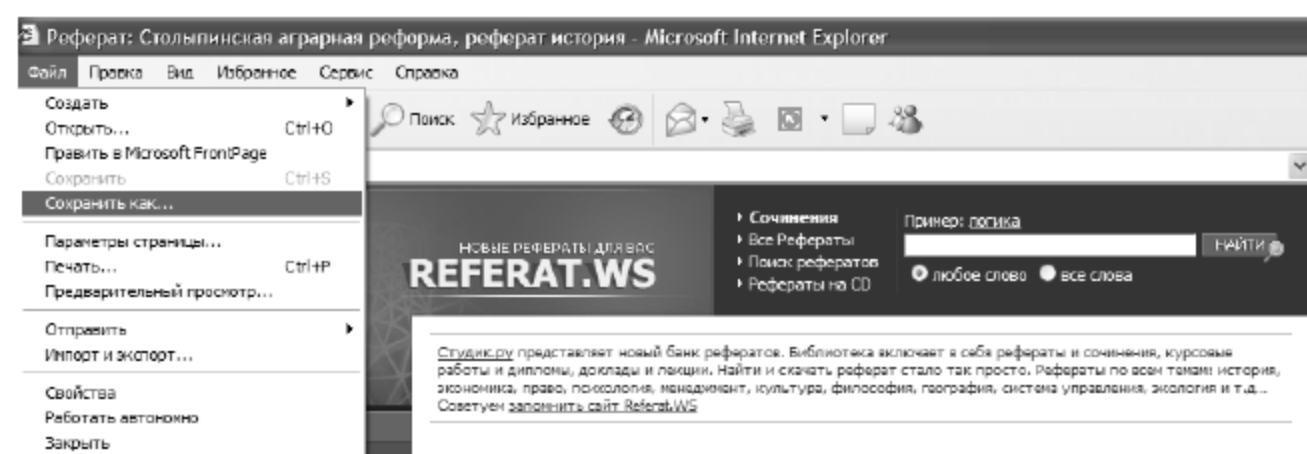


Рис. Л-10.6. Меню *Файл* браузера Internet Explorer

10.6. Индивидуальные задания

1. Найти и скопировать на жесткий диск реферат об академике А.Д. Сахарове, сохранить одну из страниц сайта, на котором он находится.
2. Создать почтовый ящик на rambler.ru.
3. Найти и скопировать на жесткий диск реферат о Ж. Алферове, сохранить одну из страниц сайта, на котором он находится.
4. Создать почтовый ящик на mail.ru.
5. Найти и скопировать на жесткий диск реферат о среде программирования Turbo-Pascal, сохранить одну из страниц сайта, на котором он находится.
6. Создать почтовый ящик на yandex.ru.
7. Найти и скопировать на жесткий диск реферат об академике В.М. Глушкове, сохранить одну из страниц сайта, на котором он находится.
8. Создать почтовый ящик на netman.ru.
9. Найти и скопировать на жесткий диск реферат об академике И.В. Курчатове, сохранить одну из страниц сайта, на котором он находится.
10. Создать почтовый ящик на hotmail.ru.

Контрольные вопросы

1. Что такое Интернет?
2. Что такое WWW?
3. Что такое браузер? Назовите наиболее распространенные браузеры.
4. Назовите наиболее распространенные поисковые системы.
5. Назовите наиболее распространенные услуги Интернета.
6. Как найти нужную информацию в Интернете?
7. Как создать электронный ящик?
8. Назовите компоненты, составляющие имя почтового ящика.
9. Назовите основные операции при работе с электронной почтой.

Лабораторные работы 11—12*
ЦЕЛЫЕ ЧИСЛА: СПОСОБЫ ПРЕДСТАВЛЕНИЯ
И ХРАНЕНИЯ В ЭВМ, ОСНОВНЫЕ ОПЕРАЦИИ
ОБРАЩЕНИЯ С ЧИСЛАМИ

11.1. Цель работы

1. Изучение типов числовых данных с фиксированной точкой (ФТ) и основных операций обращения с ними.
2. Нарботка практических навыков обращения с целыми числами на компьютере (запись, считывание, хранение).

11.2. Порядок выполнения работы

1. Изучение чисел с ФТ: представление в различных системах счисления и форматах, запись в память, считывание, диапазоны, организации хранения и др. (дома). Дополнительная литература [методические указания, 1, 2, 12].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (классификация числовых данных, перевод из одной системы счисления в другую, машинные коды, форматы, диапазон, хранение).
4. Работа на компьютере (см. подраздел 11.5).
5. Защита отчета.

11.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- 1) целые числа: способы представления и хранения (перевод чисел из одной системы счисления в другую — $10 \leftrightarrow 2$, $10 \leftrightarrow 16$, $16 \leftrightarrow 2$, машинные коды, форматы, диапазоны представимых чисел);
- 2) организация хранения данных:
 - ♦ элементы хранения данных: регистры, память (основные понятия, принятые обозначения, характеристика);
 - ♦ логическая организация памяти, принцип записи многобайтовых чисел в памяти компьютера.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- 1) исходное задание (из табл. Л-11.4);
- 2) таблица результатов по заданию 1 и пояснения к ней (заполняется *в ходе* выполнения *работы* по результатам считывания из памяти);

* Работа рассчитана на выполнение за 4 ч в отличие от двухчасовых работ 1—10. Поэтому работам 11—12, 13—14, 16—17 присвоены двойные номера.

- 3) таблица результатов по заданию 2 и пояснения к ней (заполняется дома);
- 4) фрагмент адресного пространства к заданию 2 (см. рис. Л-11.18).

11.4. Методические указания

ЦЕЛЫЕ ДВОИЧНЫЕ ЧИСЛА:

КЛАССИФИКАЦИЯ, ОСОБЕННОСТИ, ОСНОВНЫЕ ПОНЯТИЯ

В ЭВМ различают два основных типа числовых данных:

- ♦ целые двоичные числа (Integer) — числа с фиксированной точкой (ФТ);
- ♦ вещественные двоичные числа (Real) — числа с плавающей точкой (ПТ).

В данной работе рассматривается 1-й тип чисел, которые, в свою очередь, делятся на знаковые и беззнаковые (рис. Л-11.1).

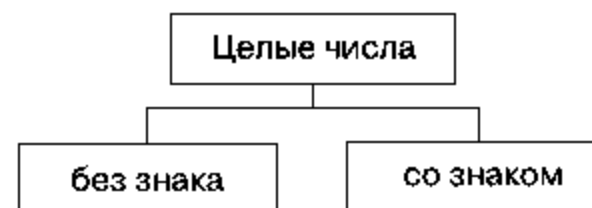


Рис. Л-11.1. Целые числа

При работе с числами на ЭВМ необходимо различать:

- ♦ значение числа;
- ♦ его внутреннее представление (в ЭВМ числа представляются в двоичной, 16-ричной либо двоично-десятичной форме);
- ♦ отображение внутреннего представления в памяти ЭВМ.

Целое число X , представленное в форме с ФТ (точка фиксируется после младшего разряда), например $X=100111_2$, может иметь различную интерпретацию, две из которых рассматриваются ниже.

Целое число без знака (все шесть двоичных разрядов числа являются значащими, т.е. имеющими соответствующий вес):

$$\begin{array}{ccccccc}
 \text{ст.} \downarrow & & & & & & \downarrow \text{мл.} \\
 & 5 & 4 & 3 & 2 & 1 & 0 \text{ (i-й разряд)} \\
 X = & 1 & 0 & 0 & 1 & 1 & 1_2 \\
 & 2^5 & 2^4 & 2^3 & 2^2 & 2^1 & 2^0 \text{ (} 2^i \text{ — вес i-го двоичного разряда)}
 \end{array}$$

и тогда десятичный эквивалент его (X_{10}) можно определить по формуле

$$X_{10} = \pm \sum_{i=0}^{n-1} a_i q_i. \quad (1)$$

Так как $q=2$, а коэффициенты двоичного ряда $a_i \in \{0, 1\}$, то

$$X_{10} = \underset{a_5}{1} * 2^5 + \underset{a_4}{0} * 2^4 + \underset{a_3}{0} * 2^3 + \underset{a_2}{1} * 2^2 + \underset{a_1}{1} * 2^1 + \underset{a_0}{1} * 2^0 = 32 + 4 + 2 + 1 = 39. \quad (2)$$

Целое число со знаком (старший бит не имеет веса и отображает знак). Единица в знаковом разряде — признак отрицательного числа (3) и, следовательно, приведенная последовательность — машинный код числа X (как правило, дополнительный). Значение X можно найти по аналогии с выражением (2), только после перехода от кода к числу, т.е. $[X]_{\text{дк}} \rightarrow X_2$ или $[X]_{\text{дк}} \rightarrow X_{16}$. В ЭВМ информация представляется в виде чисел, записанных в той или иной системе счисления, кратной степени двойки (двоичной, 16-ричной и др.). При этом, чем больше основание системы счисления q , тем короче запись числа, т.е. тем меньше разрядов требуется для его записи и хранения. Таким образом, ввод, вывод и обработка чисел на ЭВМ связаны с преобразованием их из одной системы счисления в другую ($10 \leftrightarrow 2$, $10 \leftrightarrow 16$, $16 \leftrightarrow 2$ и др.)

**ПЕРЕВОД ДЕСЯТИЧНЫХ ЧИСЕЛ
В СИСТЕМУ СЧИСЛЕНИЯ С ОСНОВАНИЕМ q (ПРЯМОЙ)
И ОБРАТНО ($X_{10} \leftrightarrow X_q$)**

Правило прямого перевода. Исходное число и последовательно получающиеся частные делятся на q до получения частного меньше q . Получаемые при делении остатки являются разрядами числа в новой q -ичной системе счисления. Последний остаток, за который принимается последнее частное, является старшим разрядом числа, т.е. для записи числа X_q остатки записываются в порядке, обратном их получению.

Пример. Десятичное число $X = 39$ перевести в двоичную и 16-ричную систему счисления, иначе говоря, найти двоичное и 16-ричное представление данного десятичного числа, т.е. $X = 39 = (?)_2 = (?)_{16}$.

Процесс перевода поясняется табл. Л-11.1 и Л-11.2 соответственно.

Таблица Л-11.1

Перевод $X_{10} \rightarrow X_2$

Исходное число и частные		q		Частное		Остаток	Разряд
39	:	2	=	19	→	1	Млад- ший
19	:	2	=	9	→	1	↑
9	:	2	=	4	→	1	
4	:	2	=	2	→	0	
2	:	2	=	1	→	0	
				1	→	1	Стар- ший

Таблица Л-11.2

Перевод $X_{10} \rightarrow X_{16}$

Исходное число		q		Частное		Остаток	Разряд
39	:	16	=	2	→	7	Младший
				2	→	2	Старший

$$\text{Таким образом, } X = 39 = \underbrace{100111}_2 = \underbrace{27}_{16}$$

Двоичное 16-ричное
представление представление

Двоичную запись числа можно получить, имея его 16-ричное представление, если каждой 16-ричной цифре ($a_i \in \{0, 1, \dots, 9, A, B, C, D, E, F\}$) поставить в соответствие тетраду — обычную 4-разрядную двоичную последовательность $\alpha_3\alpha_2\alpha_1\alpha_0$, где $\alpha_i \in \{0, 1\}$.

Например:

$$X = 27h = 0010\ 0111_{2-16} = 100111_2.$$

Для *обратного* перехода — от двоичного числа к 16-ричному — заданная двоичная последовательность разделяется на тетрады со стороны младших разрядов (недостающие дополняются нулями) и каждой из них ставится в соответствие 16-ричная цифра.

Например,

$$X = \underbrace{101}_5 \underbrace{1001}_9 \underbrace{1101}_d = \begin{matrix} 2 & 1 & 0 & i & \text{(разряд)} \\ 5 & 9 & D & h & \\ 16^2 & 16^1 & 16^0 & & \leftarrow \text{вес разряда} \end{matrix} = 5 \cdot 16^2 + 9 \cdot 16^1 + 13 \cdot 16^0 = 1437$$

ПРЕДСТАВЛЕНИЕ В ПАМЯТИ

Для представления целых чисел в памяти ПК используют три машинных формата: byte, word, long (рис. Л-11.2 и Л-11.3). *Форматом* числа называют представление его в конкретной *разрядной сетке* ЭВМ, под которой понимают набор двоичных разрядов для представления *машинного слова* в конкретной ЭВМ.

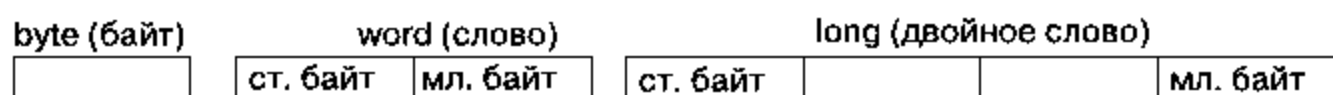


Рис. Л-11.2. Форматы беззнаковых чисел



Рис. Л-11.3. Форматы знаковых чисел

Под знак числа отводится *старший* бит. Знак кодируется:

$$\text{Sign}(\text{Sg}) = \begin{cases} 0, & \text{если } X \geq 0; \\ 1, & \text{если } X < 0. \end{cases} \quad (3)$$

(знак)

МАШИННЫЕ КОДЫ

ЭВМ работает не с числами, а с их кодами, т.е. с *машинными кодами* (МК). Для хранения отрицательных чисел и выполнения арифметических операций широко используется *дополнительный код* (ДК). Правило образования ДК для двоичных чисел имеет вид

$$[X]_{\text{ДК}} = \begin{cases} X, & \text{если } X \geq 0; \\ |\bar{X}| + 1, & \text{если } X < 0. \end{cases} \quad (4)$$

Если X — число со знаком, например $X = \pm 01011_2$, то ДК положительного и отрицательного X соответственно равны:

$$[+X]_{\text{ДК}} = 0.1011_2; [-X]_{\text{ДК}} = 10100_2 | \bar{X} |, \text{ т.е. } [-X]_{\text{ДК}} = \underbrace{1.0101_2}_{\text{хвост}}.$$

$$\begin{array}{c} + \\ 1 \\ \hline 1.0101_2 \end{array}$$

Длина числа в ДК может быть увеличена до любого числа разрядов путем копирования (тиражирования) его знакового разряда *слева*. При этом заданное число X оказывается «хвостом» двоичной последовательности требуемой длины в 8, 16, 32, ... бит. Так, рассмотренное число X в формате байта (8 бит) и слова (16 бит) имеет соответственно вид:

$$[X]_{\text{ДК}} = 1.\underbrace{1110101_2}_{\text{хвост}} = F5h = 1.111 \ 1111 \ \underbrace{0101_2}_{\text{хвост}} = FFF5h.$$

Заметим, что точка, отделяющая знаковый бит от значащих разрядов, в машинном представлении отсутствует и используется лишь при ручной записи, чтобы отличить код от числа.

ДИАПАЗОН ПРЕДСТАВИМЫХ ЧИСЕЛ

Диапазон целых чисел определяется неравенством

$$-X_{\text{max}} \leq X \leq X_{\text{max}} \quad (5)$$

и является одним из важнейших понятий при их рассмотрении. Рассмотрим диапазоны чисел для принятых форматов: байт и слово.

БАЙТ

- а) без знака: $X = 00h \div FFh = 0 \div 255$;
- б) со знаком: $X = 80h \div 7Fh = -128 \div +127$.

СЛОВО

- а) без знака: $X = 0000h \div FFFFh = 0 \div 65535$;
- б) со знаком: $X = 8000h \div 7FFFh = -32768 \div +32767$.

Следовательно, представление беззнакового числа $X > 65535$ либо $X > +32767$ приведет к *переполнению* 16-разрядной сетки.

ПЕРЕХОД ОТ КОДА К ЧИСЛУ, Т.Е. $[X]_{\text{ДК}} \rightarrow X$

Переход от ДК к числу выполняется по тому же правилу (выражение (4)), что и от числа к коду. Для отрицательных чисел $X = -([\bar{X}]_{\text{ДК}} + 1)$.

Пример. Найти десятичный эквивалент числа X , представленного в ДК, в формате байта (а) и слова (б).

Решение:

а) $[X]_{\text{ДК}} = A3h = 1.0100011_2$; $X = -01011101_2 = -5Dh = -93$.

б) $[X]_{\text{ДК}} = FFA3h = 1.1111111\ 10100011_2 = -0000\ 0000\ 0101\ 1101_2 = -005Dh = -93$.

ХРАНЕНИЕ ЧИСЛОВЫХ ДАННЫХ

Числа в компьютере хранятся либо в регистрах центрального процессора, например в регистрах общего назначения AX, BX, CX, DX, имеющих длину 16 бит в МП К1810 (рис. Л-11.4), либо в ОП. Регистр является устройством временного хранения данных и используется при выполнении арифметических (сложение, вычитание), логических (дизъюнкция, конъюнкция и др.) и операций пересылки ($AX \leftarrow [M]$, $AX \leftarrow (BX)$ и др.).



Рис. Л-11.4. Регистры данных

Длина регистра n является конструктивной единицей и определяется типом компьютера и его процессором. В общем случае РОНЫ могут использоваться для вычисления и модификации адресов и др. В n -разрядном регистре (рис. Л-11.5) биты нумеруются с 0 по $n - 1$. Бит 0 является самым младшим значащим битом. В РОНах, используемых для хранения данных, старший ($n - 1$) бит отводится под знак числа. Разрядность регистра n определяет длину разрядной сетки машины и соответственно длину ее машинного (двоичного) слова — упорядоченную последовательность бит, которыми ЭВМ оперирует одновременно.

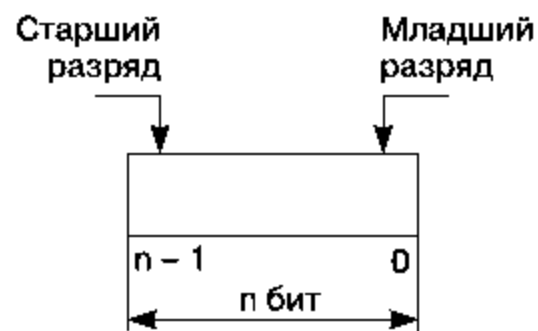


Рис. Л-11.5. Представление регистра

Для длительного хранения больших объемов данных используют «память» ЭВМ, которую разделяют на основную (ОП), внешнюю (ВП) и др. На программном уровне ОП представляют как линейную последовательность байтов (рис. Л-11.6). Адрес указывает конкретное место байта в среде хранения. В МП 80286, например, пространство линейных адресов простирается от 0 до 1 МБ (2^{20} байт). Из рисунка Л-11.6 видно, что для записи «слова» необходимо занять две последовательно расположенные ячейки памяти, т.е. два байта. Адресом слова в этом случае является адрес младшего байта. Принцип «младшее по младшему адресу» сохраняется и для представления других единиц данных: двойных слов, многобайтных команд и т.д.

Адрес	Память	
00000	1C	} байт = 1C по адресу 00000
00001	5A	
00002	7E	} слово = 7E5A по адресу 00001
00003	4B	
00004	50	} байт = 4B по адресу 00003
00005	6F	
		} слово = 6F50 по адресу 00004
⋮	⋮	
FFFFF		

Рис. Л-11.6. Адресное пространство памяти

Примечание. Поскольку для обращения к конкретному байту в адресном пространстве (см. рис. Л-11.6) требуется 5-разрядный адрес (представляется в 16-ричной системе счисления), а разрядность ЦП ($n = 16$ бит) ограничивает его лишь четырьмя, максимальный адрес, формируемый ЦП, равен $2^{16} = \text{FFFFh}$, что составляет 64 КБ = 65535 байт.

Адресное пространство *сегментируется*, т.е. разделяется на сегменты по 64 КБ, и ЦП вместо 5-разрядного физического адреса формирует *логический*, который хранится в двух регистрах (рис. Л-11.7). Один

из них, например DS, содержит адрес сегмента («базу»), а второй — смещение внутри сегмента и показывает, на сколько ячеек *адресуемый байт* смещен относительно базы.

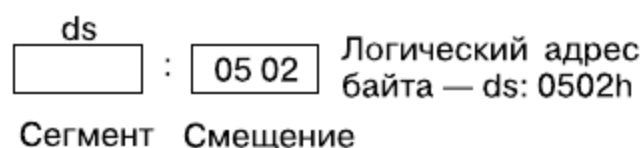


Рис. Л-11.7. Логический адрес

11.5. Практическая работа на ПК

Используя отладчик TD, выполнить упражнения с регистрами и памятью.

РЕГИСТРАЦИЯ:

- ♦ щелкнуть мышью по кнопке *Пуск*;
- ♦ в открывшемся меню выбрать пункт *Выполнить...*;
- ♦ в появившемся окне ввести cmd и нажать клавишу [Enter];
- ♦ в открывшемся окне ввести d: и нажать [Enter];
- ♦ ввести cd users* (номер группы, например 840) и нажать [Enter];
- ♦ после появления приглашения D:\INFORM\840 загрузить рабочую среду, набрав в командной строке lab11_12 [Enter] (для студентов кафедры ЭВМ РГРТУ). Другим пользователям необходимо перейти в рабочий каталог и запустить файл lab11_12.bat. На экране монитора появится окно центрального процессора (рис. Л-11.8).

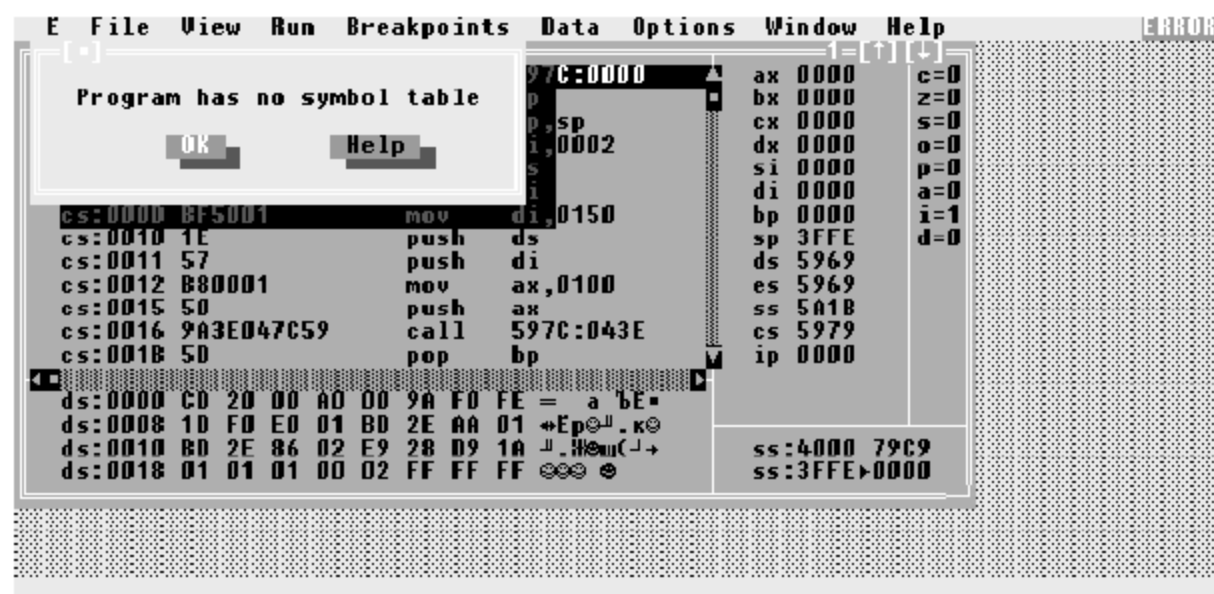


Рис. Л-11.8. Состояние экрана после загрузки рабочей среды

ЗАДАНИЯ**Задание 1**

По заданному адресу (см. табл. Л-11.4) считать из памяти (в режиме отображения byte) числа, представленные в форматах байта, слова и двойного слова, интерпретируя их как:

- а) знаковое число;
- б) беззнаковое число.

Результаты перехода от представления числа в памяти к его десятичному значению $(X:) \rightarrow [X]_{\text{дк}} \rightarrow X$ свести в табл. Л-11.5 (заполняется в лаборатории) и показать преподавателю.

Задание 2

Заданное в соответствии с № варианта число $\pm X$ (см. табл. Л-11.7) представить в форматах байта, слова и двойного слова и разместить в памяти, начиная с адреса ds:0030.

Результаты преобразований и перехода от значения числа к представлению его в памяти, т.е. $X \rightarrow [X]_{\text{дк}} \rightarrow (X:)$, в режиме отображения byte свести в табл. Л-11.6 (по образцу табл. Л-11.3). Привести фрагменты адресного пространства (см. рис. Л-11.18).

Результаты преобразований над числами и их представлениями перенести из табл. Л-11.6 в память ПК.

Таблица Л-11.3

Переход $X \rightarrow (X:)$

Число	Количество байтов	16-ричное представление	Адрес	Отображение в памяти
+51	1	33	ds:0030h	33
-51	1	CD	ds:0038h	CD
-51	2	FFCD	ds:003ah	CDFF

РАБОТА В TurboDebugger

После загрузки рабочей среды начинает свою работу отладчик TD. Это программа, которая находится на диске ПК и позволяет отображать и изменять содержимое памяти, регистров МП, отлаживать исполняемый модуль исходной программы в машинных кодах и др. Нажатием клавиши [Enter] снимается окно с сообщением *Program has no symbol table*, и на экране появляется окно отладчика центрального про-

цессора, состоящее из 5 подокон (рис. Л-11.9). Новое состояние экрана свидетельствует о том, что окно центрального процессора является активным, на что указывает находящийся в нем курсор [■].

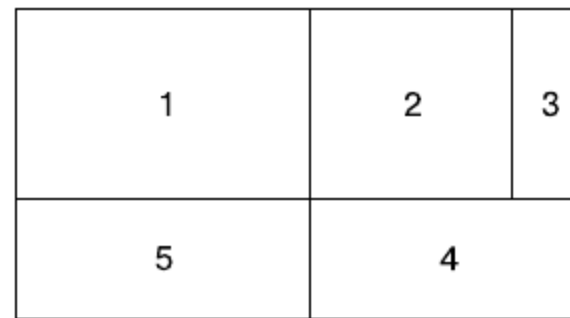


Рис. Л-11.9. Схема расположения подокон окна центрального процессора

1. Подокно, отображающее сегмент памяти, хранящий программу (рис. Л-11.10):

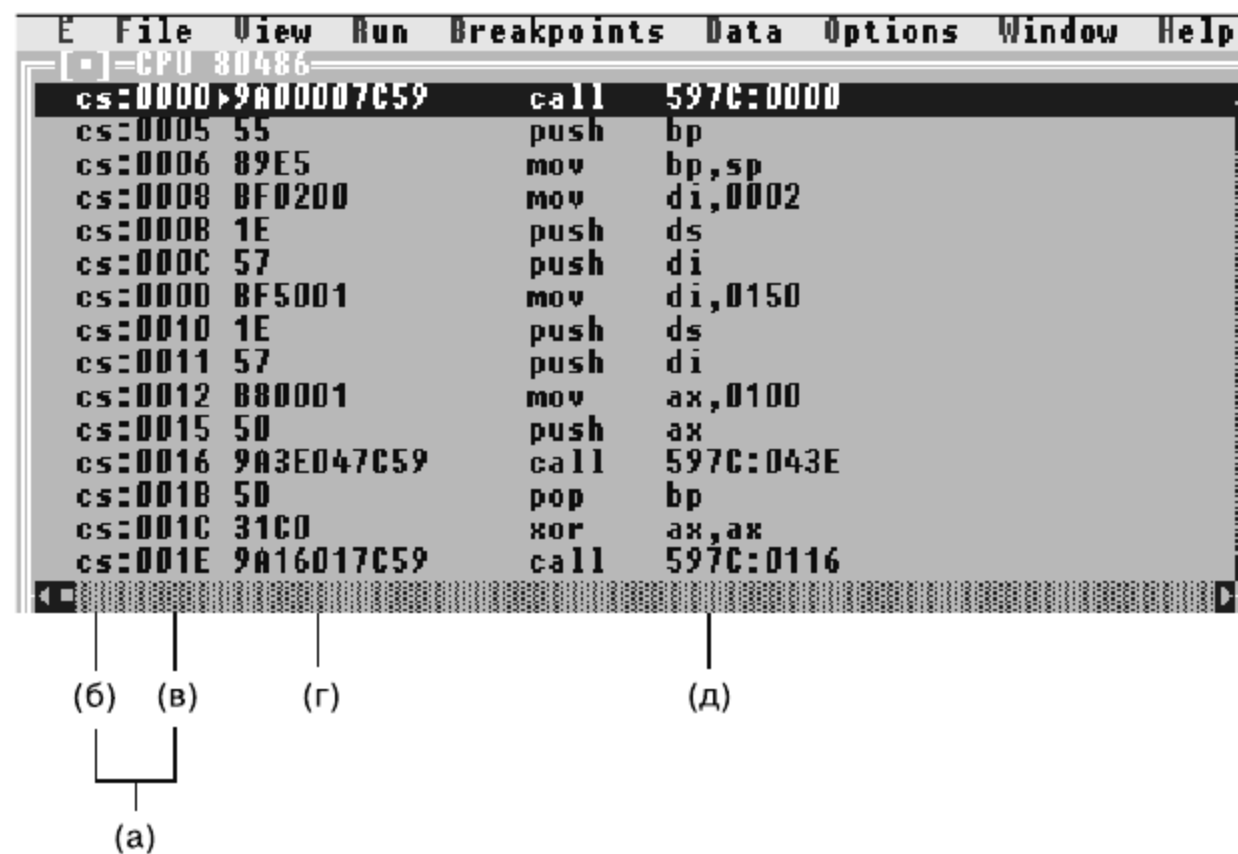


Рис. Л-11.10. Подокно, отображающее сегмент памяти

- а) полный адрес;
- б) база сегмента (cs указывает на то, что значение базы берется из регистра cs);
- в) смещение в сегменте;
- г) код команды;
- д) мнемоника команды на языке ассемблера.

2. Подокно, отображающее список регистров и их содержимое, представлено на рис. Л-11.11.



ax	0000
bx	0000
cx	0000
dx	0000
si	0000
di	0000
bp	0000
sp	3FFE
ds	5969
es	5969
ss	5A1B
cs	5979
ip	0000

Название регистра Содержимое регистра

Рис. Л-11.11. Подокно, отображающее список регистров и их содержимое

3. Подокно, отображающее флаги состояния процессора (не приводится).

4. Подокно, отображающее стек (не приводится).

5. Подокно, отображающее фрагмент памяти (дамп), хранящий данные (рис. Л-11.12):



ds:0000	CD 20 00 A0 00 9A F0 FE	= a bE
ds:0008	1D F0 E0 01 BD 2E AA 01	+Er .к
ds:0010	BD 2E 86 02 E9 28 D9 1A	Д.Иш(→
ds:0018	01 01 01 00 02 FF FF FF	???
ds:0020	FF FF FF FF FF FF FF	

(б) (в) (г)

(а)

Рис. Л-11.12. Подокно, отображающее фрагмент памяти, хранящий данные

- а) полный адрес;
- б) база сегмента (ds указывает на то, что значение базы берется из регистра ds);
- в) смещение в сегменте;
- г) содержимое памяти (каждая строка отображает содержимое восьми последовательных ячеек памяти. Адрес указан только

для первой из них. Для вычисления адресов последующих ячеек памяти к смещению необходимо прибавить соответственно 1, 2, 3 и т.д.).

Переход из одного подокна центрального процессора в другое выполняется нажатием клавиши [Tab] или [Shift] + [Tab]. При этом активность подокна отмечается ярким прямоугольником. Клавишей [F5] можно увеличить/уменьшить окно центрального процессора.

**ОСНОВНЫЕ КОМАНДЫ ОТЛАДЧИКА
ДЛЯ РАБОТЫ С РЕГИСТРАМИ И ПАМЯТЬЮ
С ПОМОЩЬЮ ОКНА ЦЕНТРАЛЬНОГО ПРОЦЕССОРА**

Работа с регистрами

Перейдите в подокно регистров и подведите маркер к регистру DX. Введите с клавиатуры код 30 и нажмите клавишу [Enter]. Тем самым вы ввели в DX операнд N = 0030h. (При вводе 16-ричных кодов *необходимо*, чтобы первый символ был цифрой, например 0A5 или 0FFFF. Если последний символ D или B, то после него обязательно вводится символ h.)

Упражнения:

- а) введите в регистр BL число X = -17h в формате *байта*.

Ваши действия:

- ◆ найдите [X]_{дек};
- ◆ подведите маркер к регистру BX;
- ◆ введите код [X]_{дек};
- ◆ [Enter].

	BH	BL
Результат: BX	00	E9

- б) введите в регистр BH число X = -17h в формате *байта*.

Ваши действия:

- ◆ найдите [X]_{дек};
- ◆ подведите маркер к регистру BX;
- ◆ введите код 0E900;
- ◆ [Enter].

	BH	BL
Результат: BX	E9	00

- в) введите в регистр BX число X = -17h в формате *слова*.

Последовательность действий определить *самостоятельно*.

	BH	BL
Результат: BX	FF	E9

г) измените содержимое регистра DX на N=01A9h.

Ваши действия:

- ◆ подведите маркер к регистру DX;
- ◆ введите код 1A9;
- ◆ [Enter].

Результат: DX

DH	DL
01	A9

.

Результаты упражнений показать преподавателю.

Работа с памятью

Упражнения:

а) запишите в память, начиная с адреса 502h, число X= -75h в формате слова (затем байта и двойного слова).

Для выбора требуемого режима отображения (byte, word или long):

- ◆ перейдите в подокно памяти;
- ◆ вызовите локальное меню нажатием клавиш [ALT] + [F10].

Экран примет вид, показанный на рис. Л-11.13;

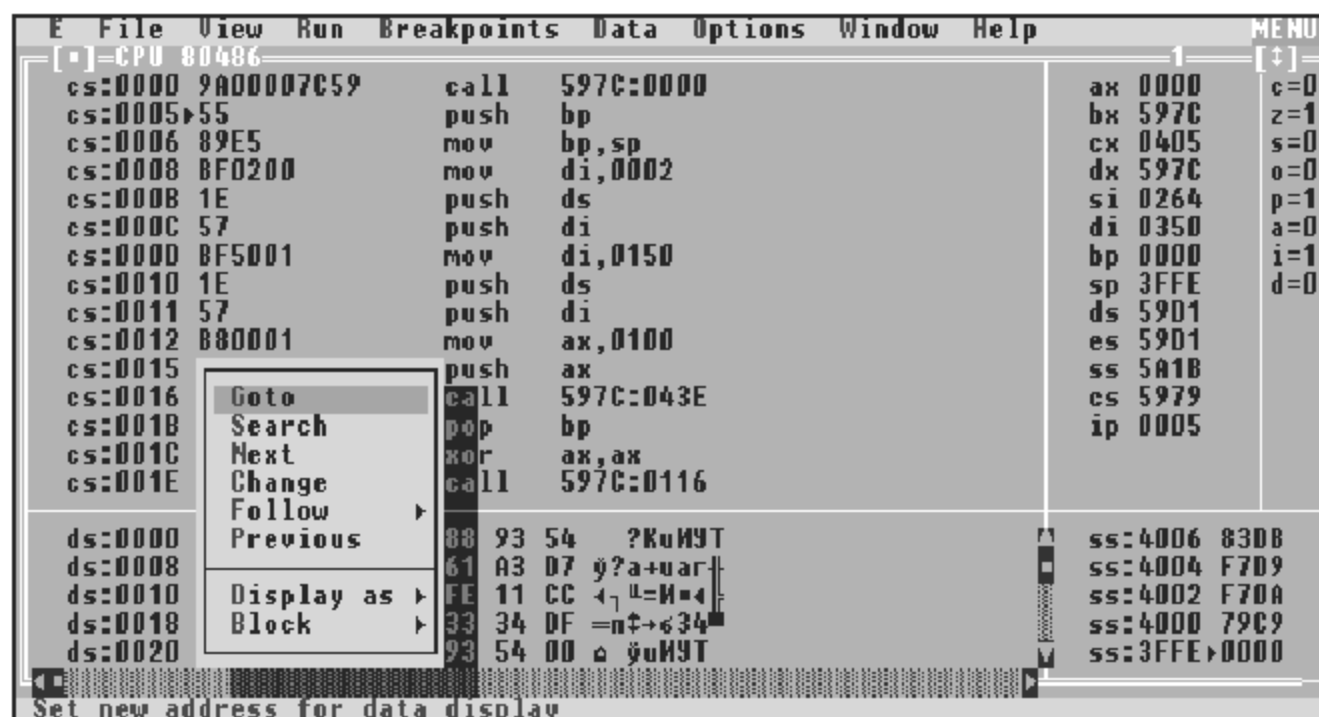


Рис. Л-11.13. Вид экрана Выбор локального меню

- ◆ переместите курсор на пункт *Display as* и нажмите [Enter]. Экран примет вид, показанный на рис. Л-11.14;



Рис. Л-11.14. Вид экрана при выборе отображения памяти

- ♦ выберите требуемый режим отображения (Byte, Word или Long) и нажмите [Enter];
- ♦ установите подокно с отображением ОП на область памяти, которая начинается с адреса 502h, для чего:
 - ♦ нажмите комбинацию клавиш [Ctrl] + [G] (нажать [Ctrl] и, не отпуская ее, клавишу [G]). При этом на экране раскроется диалоговое окно с приглашением для ввода нужного адреса: *Enter address to position to* (Введите адрес, на который надо установить окно) (рис. Л-11.15);



Рис. Л-11.15. Вид экрана при вводе адреса

- ◆ наберите адрес 502h;
- ◆ нажмите [Enter];
- ◆ найдите $[X]_{\text{дк}}^*$.

Подведите курсор к конкретному байту памяти, начните ввод нужного значения. В ответ, как показано на рис. Л-11.16, появится диалоговое окно с приглашением *Enter new data bytes* (Введите новые байты данных). Продолжите ввод значения, а для завершения ввода нажмите клавишу [Enter]. После этого происходит изменение содержимого памяти, как показано на рис. Л-11.17;



Рис. Л-11.16. Вид экрана при вводе данных в формате байта

- ◆ с клавиатуры наберите код 0FF8B;
 - ◆ нажмите [Enter].
- Результат: 0502 FF 8B 8B 00 FF FF FF 8B
: 050A ;
- б) загрузите $X = -75h$ в формате байта, слова и двойного слова по одному адресу (0502h);
- в) пункты а и б повторить для $X = +75h$.
- Результаты упражнений показать преподавателю.

* $[X]_{\text{дк}} = \underbrace{\text{FF8Bh}}_{\text{слово}} = \underbrace{\text{8Bh}}_{\text{байт}} = \underbrace{\text{FFFFFF8Bh}}_{\text{двойное слово}}.$

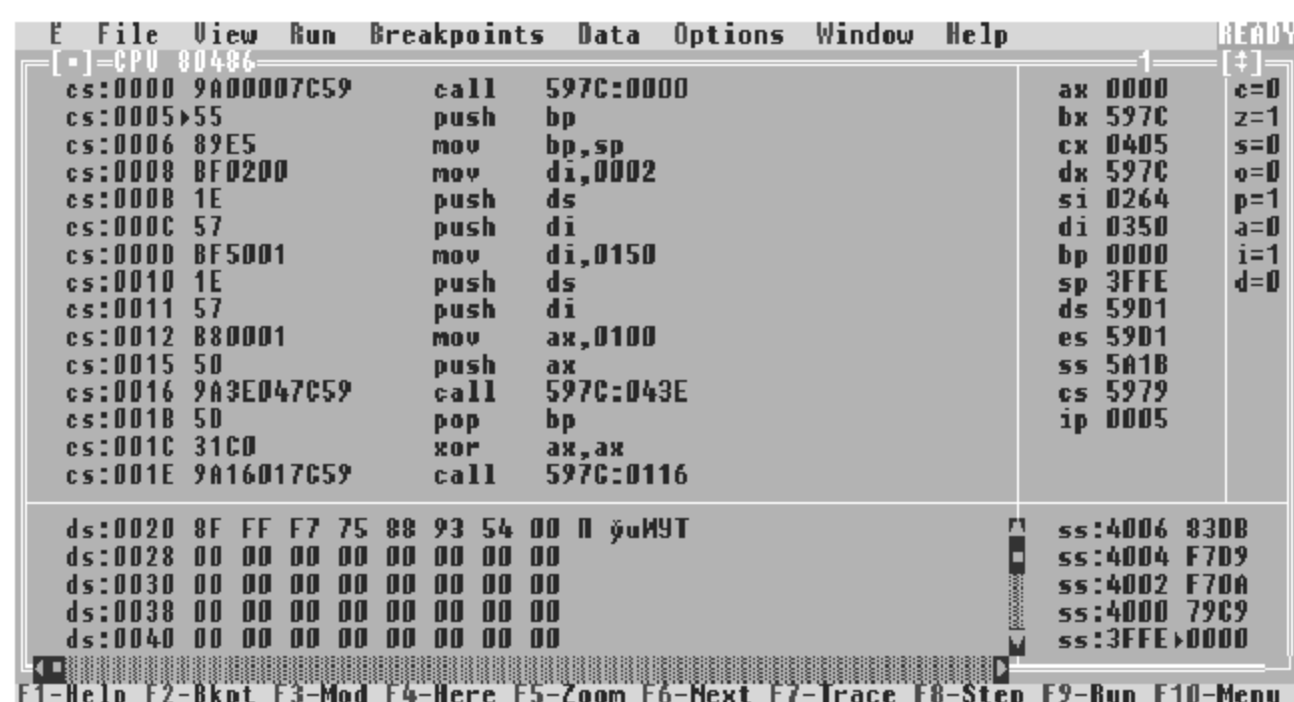


Рис. Л-11.17. Вид экрана после ввода данных

ПОРЯДОК ВЫПОЛНЕНИЯ ЗАДАНИЙ

Для выполнения задания 1 необходимо:

- ♦ установить подокно с отображением оперативной памяти (ОП) на область, начинающуюся с адреса, задаваемого табл. Л-11.4 в соответствии с номером варианта (соответствует номеру бригады);

Таблица Л-11.4

Варианты к заданию 1

Вариант	1	2	3	4	5
Адрес	ds:2	ds:4	ds:6	ds:8	ds:10
Вариант	6	7	8	9	10
Адрес	ds:12	ds:14	ds:16	ds:18	ds:20

- ♦ нажать комбинацию клавиш [Ctrl] + [G]. В открывшемся диалоговом окне (см. рис. Л-11.15) набрать нужный адрес и нажать [Enter];
- ♦ установить режим отображения Byte, после чего нажать [Enter];
- ♦ перенести представление числа в памяти, расположенного по заданному начальному адресу, в соответствующую строку третьего столбца табл. Л-11.5 (по образцу табл. Л-11.5*) и *показать преподавателю*.

Примечание. Для заполнения столбца 4 табл. Л-11.5 необходимо менять режимы отображения.

Таблица Л-11.5

Переход $(X:) \rightarrow [X]_{\text{дк}} \rightarrow X$

Адрес	Количество байт для представления числа	Представление числа в памяти $X: **$	16-ричное представление числа $[X]_{\text{дк}}$	Двоичное представление числа X_2	Десятичное представление числа X_{10}	
					без знака	со знаком
(1)	(2)	(3)	(4)	(5)	(6)	(7)
	1					
	2					
	4					

Таблица Л-11.5*

Пример заполнения таблицы Л-11.3

Адрес	Количество байт для представления числа	Представление числа в памяти $X: **$	16-ричное представление числа $[X]_{\text{дк}}$	Двоичное представление числа X_2	Десятичное представление числа X_{10}	
					без знака	со знаком
(1)	(2)	(3)	(4)	(5)	(6)	(7)
ds:0002	1	10	10	0001 0000	16	+16
ds:0002	2	10 00	0010	0...00010000	16	+16
ds:0002	4	10 00 00 00	00000010	0...00010000	16	+16

Для выполнения задания 2 необходимо заданное число (табл. Л-11.7), представленное в форматах байта, слова и двойного слова (см. табл. Л-11.6), записать в память, заполнив адресное пространство, начиная с адреса ds:0030, и *показать преподавателю*.

Таблица Л-11.6

Переход от числа к его отображению в памяти, т.е. $X \rightarrow [X]_{\text{дк}} \rightarrow (X:)$

Число	Количество байт для представления числа	16-ричное представление числа	Адрес	Отображение числа в памяти**
(1)	(2)	(3)	(4)	(5)
			ds:0030h	
			ds:0032h	
			ds:0034h	
			ds:0038h	
			ds:003ah	
			ds:003ch	

Таблица Л-11.7

Варианты к заданию 2

Вариант	1	2	3	4	5	6	7	8	9	10
Число	+23	+57	+41	+97	+34	+49	+19	+63	+78	+91
	-23	-57	-41	-97	-34	-49	-19	-63	-78	-91

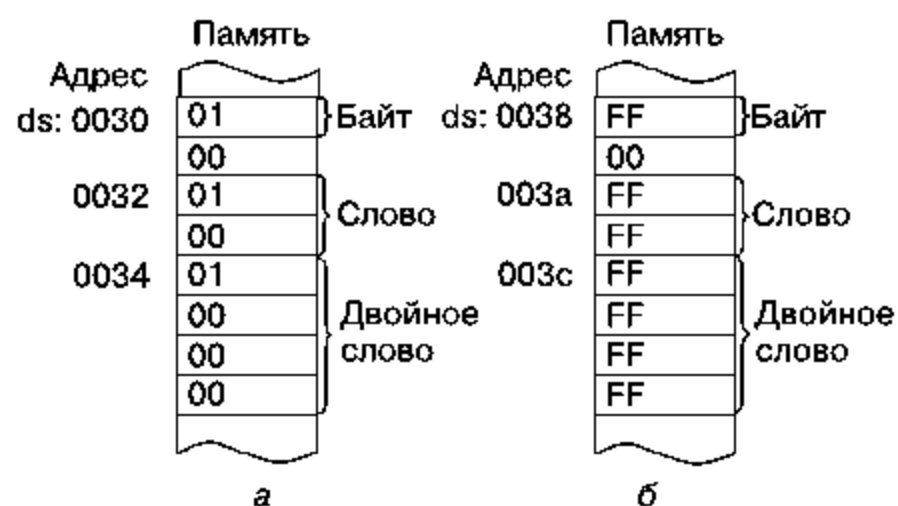


Рис. Л-11.18. Фрагмент адресного пространства:

а — область положительных чисел; б — область отрицательных чисел

Контрольные вопросы

1. Назовите основные форматы целых чисел.
2. Назовите число двоичных разрядов, необходимое для представления
 - а) $X = 418$;
 - б) $X = \pm 418$.
3. Назовите десятичный эквивалент числа, представленного в формате байта

$\begin{array}{c} 10010101 \\ \hline 7 \qquad 0 \end{array}$

если последний интерпретировать как:

 - ◆ число без знака;
 - ◆ число со знаком;
 - ◆ упакованное BCD-число.
4. Покажите представление числа $X = \pm 92$ в формате байта, слова, двойного слова.
5. Приведите 16-ричную запись числа $X = \pm 92$, представленного в форматах байта, слова, двойного слова.
6. Покажите представление $[X]_{\text{ПК}}$ и $[X]_{\text{ДК}}$ числа $X = \pm 92$ в формате байта и слова.
7. Назовите диапазон знаковых и беззнаковых чисел для 6- и 8-разрядной сеток.
8. Найдите десятичный эквивалент (X_{10}) числа, расположенного в памяти по адресу 01A7h (рис. Л-11.19).

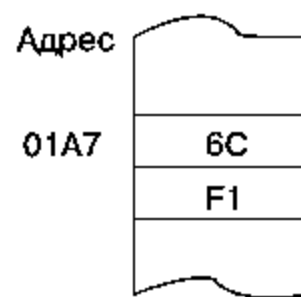


Рис. Л-11.19. Представление числа памяти

9. Найдите X_{10} , если его отображение в памяти $X:37FFh$.
10. Переведите число $X=463$ в двоичную систему тремя известными вам способами.
11. Переведите $X=571$ в 16-ричную и двоичную систему счисления: $X=571 = (?)_{16} = (?)_2$.
12. $[X]_{\text{дк}} = FF46h$. $X_{10} = ?$
13. Назовите основные элементы хранения числовых данных в ЭВМ и их основные характеристики.
14. Назовите номер ячейки памяти (номер байта) с адресом $A31Ch$.
15. Назовите максимальное число ячеек памяти, к которым можно адресоваться с помощью 16-разрядного регистра, и дайте пояснение.
16. Дайте понятие формата числа, разрядной сетки, машинного слова.
17. Какие средства позволяют отображать числовые данные и наблюдать за результатами выполнения операций над ними?

Лабораторные работы 13—14

ВЕЩЕСТВЕННЫЕ ЧИСЛА:

СПОСОБЫ ПРЕДСТАВЛЕНИЯ И ХРАНЕНИЯ В ЭВМ

Выполняется в течение двух занятий:

- ♦ 1-е занятие — работа с вещественными числами без компьютера;
- ♦ 2-е занятие — работа с вещественными числами на компьютере.

13.1. Цель работы

1. Изучение основных типов данных с плавающей точкой, принятых стандартов их представления в современных ЭВМ.
2. Нарботка практических навыков обращения с вещественными числами на компьютере (запись, считывание, хранение).

13.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 2, 12].

2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (цель работы, понятие вещественных чисел, основные типы вещественных чисел, принятые стандарты их представления в современных компьютерах, внутреннее представление вещественных чисел в принятых стандартах и их отображение в памяти компьютера — форматы, диапазон, хранение).
4. Работа на компьютере (см. подраздел 13.5).
5. Защита отчета.

13.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ♦ вещественные числа: представление в компьютере, типы, принятые стандарты их представления в современных компьютерах, внутреннее представление в принятых стандартах и их отображение в памяти компьютера.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ♦ конкретно сформулированное исходное задание. Например, число $x = -41$ (см. табл. 13.2, вариант 3) представить в ФПТ в форматах...;
- ♦ таблица результатов по заданию 1, а и пояснения к ней (заполняется дома);
- ♦ таблица результатов по заданию 2 и пояснения к ней (заполняется в лаборатории в ходе выполнения работы).

13.4. Методические указания

ВЕЩЕСТВЕННЫЕ ЧИСЛА: ОСНОВНЫЕ ПОНЯТИЯ, ТИПЫ, ОСОБЕННОСТИ, ПРЕДСТАВЛЕНИЕ В КОМПЬЮТЕРЕ

Вещественные числа (REAL) — числа, имеющие дробную часть. Для их представления в компьютере используется так называемое представление с плавающей точкой (ПТ), основанное на алгебраической записи чисел в виде

$$X = \pm M_x q^{P_x}, \quad (1)$$

где q — основание системы счисления;

P_x — порядок числа (целое);

M_x — мантисса (значащая часть числа), обычно представляется дробью:

$$M_x = \pm 0.d_1 d_2 \dots d_n.$$

Условие $d_1 \neq 0$ является характерным признаком нормализованного числа в ФПТ. Нормализованное представление используется для однозначной записи вещественных чисел.

Для нормализованной двоичной мантиссы ($q = 2$) диапазон определяется неравенством $2^{-1} \leq |M_x| < 1$.

Для примера целое число $X = -24 = -10101$. представим в ФПТ:

 ↑
Место точки

Место точки
↓

а) $X = -\underbrace{0.10101}_{M_x} \cdot 2^{+5} = -\underbrace{0.10101}_{n=5} \cdot \underbrace{10}_{q}^{+101}$, или $X = -0.10101e5$;

б) $X = -1.\underbrace{0101}_{\uparrow \text{Место точки}} \cdot 2^{+4} = -1.0101 \cdot 10^{+100}$, или $X = -1.0101e4$,

где e — степень двойки (знак перед e не ставится, если e — положительное число). Знак числа определяется знаком мантиссы.

Из примера видно, что в форме с ПТ точка переместилась («уплыла») на пять разрядов влево, чтобы оказаться перед старшей значащей единицей (а) либо сразу после нее (б), уменьшив тем самым число в 2^5 раз (а) и 2^4 раз (б) соответственно.

Мантисса числа X записывается соответственно как

а) $M_x = -0.10101_2$, или в общем виде $\overset{1}{\uparrow} \overset{0}{d_0} \overset{1}{d_1} \overset{0}{d_2} \overset{1}{d_3} \overset{1}{d_4}$,
Место точки

где $d_i \in \{0, 1\}$;

б) $M_x = -1,0101_2$, или $d0.\underbrace{d1d2d3d4}$.
 Место точки То, что хранится
 от мантиссы в памяти

Разряд d0 в нормализованных числах всегда равен 1 и в памяти компьютера не хранится (*скрытый бит*). Таким образом, мантисса, хранящаяся в памяти компьютера, всегда число дробное (рис. Л-13.1).



Рис. Л-13.1. Представление n-разрядной мантиссы

Форма представления чисел с ПТ позволяет значительно увеличить диапазон и точность представления чисел при той же длине разрядной сетки, что и для чисел с ФТ.

ПРЕДСТАВЛЕНИЕ В ПАМЯТИ

Вещественные числа в отличие от целых представляются и хранятся в памяти в прямом коде, их представление стандартизовано. Согласно стандарту IEEE-754 для ПК IBM PC вводят три машинных формата:

- ♦ одинарной точности (4 байт);
- ♦ двойной точности (8 байт);
- ♦ расширенной точности (10 байт).

Некоторые фирмы — разработчики ПО — используют нестандартные форматы, в частности, в компиляторе языка Pascal фирмы Borland используется формат длиной 6 байт.

С целью упростить изучение названных стандартов рассмотрим вначале представление вещественных чисел в классическом двоичном формате с 4 полями, который использовался в старых ЭВМ и стал каноническим (рис. Л-13.2).

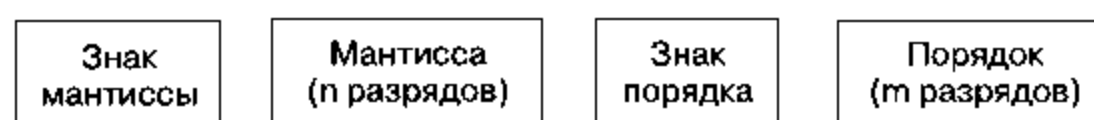


Рис. Л-13.2. Классический двоичный формат

Для примера представим в классическом двоичном формате с условно выбранной длиной полей ($n = 5$, $m = 3$) (рис. Л-13.3) число

$$X = -21 = -10101_2 = -0.10101 \cdot 2^{+5} = - \underbrace{0.10101}_{M_x} \cdot 10^{\overset{+101}{\underset{\uparrow \Pi_x}{5}}} = -0.10101e5.$$

S_g
 M_x
 S_g
 Π_x

Рис. Л-13.3. Классический двоичный формат в примере

С целью закрепить полученное представление о классическом двоичном формате предлагается выполнить два задания.

Задание 1

Считать из памяти (найти X_{10}) число, представленное в классическом двоичном формате (рис. Л-13.4).

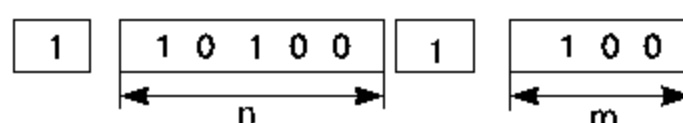


Рис. Л-13.4. Представление числа в двоичном формате (к заданию 1)

Задание 2

$X = +7\frac{5}{8}$ представить в классическом двоичном формате ($n = 5$, $m = 3$). Результаты заданий показать преподавателю.

МАШИННЫЕ ФОРМАТЫ ВЕЩЕСТВЕННЫХ ДАННЫХ

В различных алгоритмических языках принятые стандарты описываются разными ключевыми словами.

В языке Turbo-Pascal определены следующие типы вещественных чисел: single, real, double, extended и comp, из которых в данной работе рассматриваются только три (табл. Л-13.1).

Таблица Л-13.1

Типы вещественных чисел в языке Turbo-Pascal

Название типа	Длина в байтах	Мантисса десятичных значащих цифр	Диапазон десятичного порядка
Single	4	7...8	-45...+38
Real	6	11...12	-39...+38
Double	8	15...16	-324...+308

Во всех типах мантисса хранится в *прямом коде* (ПК). Порядок (или экспонента e) задается в так называемой смещенной форме. Смещение выбирается так, чтобы характеристика была целым положительным числом (тогда знак характеристики хранить не требуется), равным:

129 — для real,
127 — для single,
1023 — для double.

Смещенный порядок (характеристика E) равен истинному порядку Π , увеличенному на величину смещения, т.е. $E = \Pi + \text{смещение}$.

Мантисса всегда представляется в *прямом* двоичном коде в виде

$$\text{Мантисса} \\ d_0.\underbrace{d_1d_2d_3d_4}_{\text{Мантисса}}$$

То, что хранится в памяти

Если $0 < E < 255$ (формат single), то значение числа X определяется по правилу

$$X = f(S, E, M) = (-1)^S \cdot 2^{(E-127)} \cdot (1.M),$$

а для формата real по правилу

$$X = f(S, M, E) = (-1)^S \cdot 2^{(E-129)} \cdot (1.M),$$

где $S = 0$ или 1 .

ВНУТРЕННЕЕ ПРЕДСТАВЛЕНИЕ ВЕЩЕСТВЕННЫХ ДАННЫХ

Формат single (4 байта)

31	30	23	22	0
S	E (8 бит)	M (23 бита)		
знак числа	характеристика	мантисса		

$$X = f(S, E, M) = \begin{cases} \text{если } 0 < E < 255, \text{ то } (-1)^S \cdot 2^{(E-127)} \cdot (1.M); \\ \text{если } E = 0 \text{ и } M \neq 0, \text{ то } (-1)^S \cdot 2^{(E-127)} \cdot (1.M); \\ \text{если } E = 0 \text{ и } M = 0, \text{ то } (-1)^S \cdot 0; \\ \text{если } E = 255 \text{ и } M = 0, \text{ то } (-1)^S \cdot \text{бесконечность}; \\ \text{если } E = 255 \text{ и } M \neq 0, \text{ то } X \text{ не является числом}; \\ (\text{NaN, Not - a - Number}). \end{cases}$$

Пример.

$$X = -21 = -10101_2 = -1.0101 \cdot 2^4;$$

↑
Не хранится

$$E = \underbrace{4}_{P_x} + \underbrace{127}_{\text{Смещение}} = 131 = 128 + 3 = 83_{16};$$

1 10000011 010100...0

Sg E M

В итоге получаем:

1100 0001 1010 1000 0000 0000 0000 0000.

C 1 A 8 0 0 0 0

Таким образом, число $X = -21$ в формате single в 16-ричной системе счисления представляется как $X: C1A80000_{16}$ и отображается в памяти в обратном порядке:

$X: 00\ 00\ A8\ C1$.

47 46 8 7 0

S	M (39 бит)	E (8 бит)

↑
1.

$$X = f(S, M, E) = \begin{cases} 0, & \text{если } e = 0, \\ (-1)^S \cdot 2^{(E-129)} \cdot (1.M), & \text{если } 0 < E < 255. \end{cases}$$

$X = -21 = -10101_2 = -1.0101 \cdot 2^{+4}$;
 $\uparrow$
 Не хранится

1	0101000...0	10000101
Sg	M	E

```
1010 1000 0000 0000 0000 0000 0000 0000 0000 0000 0000 1000 0101
A      8      0      0      0      0      0      0      0      0      0      8      5
```

X: 85 00 00 00 00 A8.

63	62	52	51	0
S	E (11 бит)	Мантисса (52 бита)		

$$X = (-1)^S \cdot 2^{(E - 1023)} \cdot (1.M).$$

РЕГИСТРАЦИЯ:

- ◆ щелкнуть мышью по кнопке *Пуск*;
- ◆ в открывшемся меню выбрать пункт *Выполнить...*;
- ◆ в появившемся окне ввести cmd и нажать [Enter];
- ◆ в открывшемся окне ввести d: и нажать [Enter];
- ◆ ввести cd users\номер_группы, например 840, и нажать [Enter];
- ◆ после появления приглашения D:\INFORM\840 загрузить рабочую среду, набрав в командной строке lab13_14, нажать клавишу [Enter] (для студентов кафедры ЭВМ РГРТУ). Другим пользователям необходимо перейти в рабочий каталог и запустить файл lab13_14.bat.

Состояние экрана после загрузки рабочей среды приведено на рис. Л-13.5.

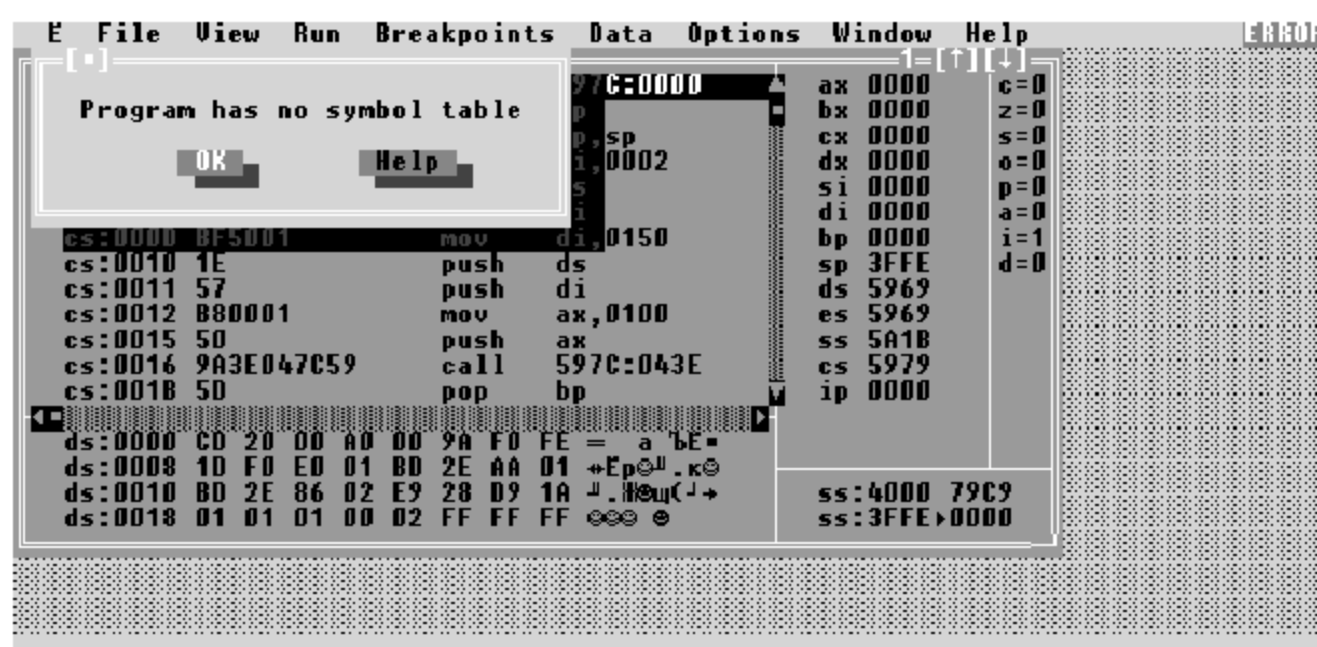


Рис. Л-13.5. Состояние экрана после загрузки рабочей среды

ЗАДАНИЯ

Задание 1:

а) заданное в соответствии с номером варианта число $\pm x^*$ (табл. Л-13.2) представить в ФПТ в трех форматах: single, real и double и разместить по начальным адресам ds: 00C0h, 00C8h, 00d0h соответственно (дома). Результаты преобразования и перехода от значения числа к отображению его в памяти свести в табл. Л-13.3. Пояснения к табл. Л-13.3 привести в отчете;

б) заданное номером варианта и представленное дома число в форматах single, real и double (табл. Л-13.3) *записать* (занести) в память ПК по заданным адресам (в лаборатории).

Таблица Л-13.2

Варианты к заданию 1

Параметр	Вариант									
	1	2	3	4	5	6	7	8	9	10
Число	-23	-57	-41	-97	-34	-41	-19	-63	-78	-91
Формат 1	single									
Формат 2	real									
Формат 3	double									

* Приведенные числа в табл. Л-13.2 изменить для всех групп, кроме гр. x40, следующим образом: гр. x41 — прибавить 10, гр. x42 — вычесть 20, гр. x43 — прибавить 15, гр. x47 — вычесть 15, гр. x48 — прибавить 5 (сложение и вычитание выполнить над модулем числа и восстановить знак).

Таблица Л-13.3

Переход от X_{10} к представлению в памяти

Число	Алгебраическая запись	Формат	16-ричное представление числа	Адрес	Отображение в памяти
1	2	3	4	5	6
-21	$1.0101 \cdot 10^{100}$	Single	C1A80000	ds:00C0h	0000A8C1
				ds:00C8h	
				ds:00D0h	

Задание 2:

по заданному в соответствии с номером варианта адресу (табл. Л-13.4) *считать* из памяти (в режиме отображения byte) числа, представленные в форматах single, real и double, и найти их десятичный эквивалент. Результаты представить в виде табл. Л-13.5, пояснения к табл. Л-13.5 привести в отчете.

Таблица Л-13.4

Варианты к заданию 2

Адрес	Вариант				
	1	2	3	4	5
1-й	ds:2h	ds:6h	ds:0ah	ds:0eh	ds:12h
2-й	ds:2ah	ds:30h	ds:36h	ds:3ch	ds:42h
3-й	ds:66h	ds:6eh	ds:76h	ds:7eh	ds:86h
Адрес	Вариант				
	6	7	8	9	10
1-й	ds:16h	ds:1ah	ds:1eh	ds:22h	ds:26h
2-й	ds:48h	ds:4eh	ds:54h	ds:5ah	ds:60h
3-й	ds:8eh	ds:96h	ds:9eh	ds:0a6h	ds:0aeh

Таблица Л-13.5

Форма для заполнения к заданию 2

Адрес	Формат	Отображение в памяти	16-ричное представление числа	Десятичный эквивалент числа X_{10}

13.6. Порядок выполнения заданий

1. Нажмите клавишу [Enter]. Окно с сообщением *Program has no symbol table* (рис. Л-13.5) пропадет.

Новое состояние экрана указывает на то, что активным является окно центрального процессора, о чем свидетельствует находящийся

в нем курсор (выделенный прямоугольник). Окно центрального процессора состоит из пяти подокон соответственно схеме, приведенной на рис. Л-13.6.

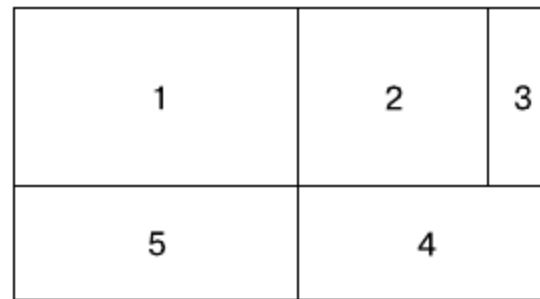


Рис. Л-13.6. Схема расположения подокон окна центрального процессора

Подокно 1 отображает сегмент памяти, хранящий программу (рис. Л-13.7).

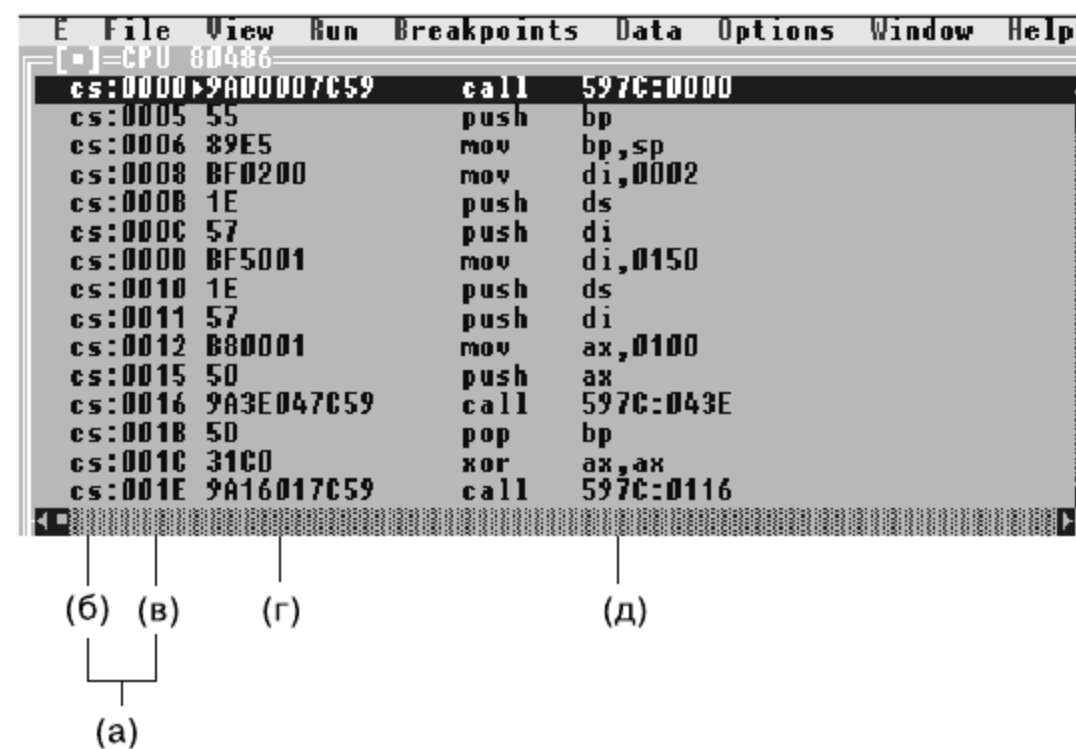


Рис. Л-13.7. Подокно 1 (сегмент памяти, хранящий программу)

На рисунке Л-13.7 приняты следующие обозначения:

- а) полный адрес;
- б) база сегмента (cs указывает на то, что значение базы берется из сегмента cs);
- в) смещение в сегменте;
- г) код команды;
- д) мнемоника команды на языке ассемблера.

Подокно 2 (рис. Л-13.8) отображает список регистров и их содержимое.



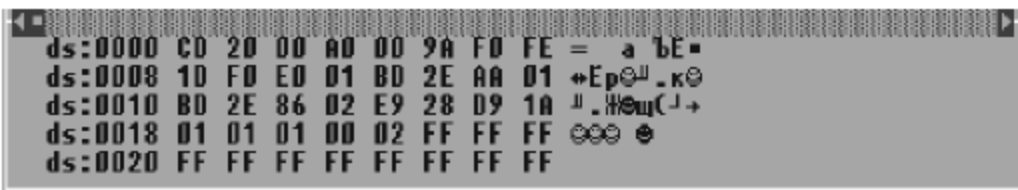
Название регистра	Содержимое регистра
ax	0000
bx	0000
cx	0000
dx	0000
si	0000
di	0000
bp	0000
sp	3FFE
ds	5969
es	5969
ss	5A1B
cs	5979
ip	0000

Рис. Л-13.8. Подокно 2 (список регистров)

Подокно 3 отображает флаги состояния процессора.

Подокно 4 отображает стек.

Подокно 5 (рис. Л-13.9) отображает фрагмент памяти (дамп), хранящий данные.



а)	б)	в)	г)
ds:0000	CD	20	00 00 9A F0 FE = a bE
ds:0008	10	F0	E0 01 8D 2E AA 01 +Er .к
ds:0010	BD	2E	86 02 E9 28 D9 1A .ЩCJ+
ds:0018	01	01	01 00 02 FF FF FF @@@
ds:0020	FF	FF	FF FF FF FF FF FF FF

Рис. Л-13.9. Подокно 5 (фрагмент памяти, хранящий данные)

На рисунке Л-13.9 приняты следующие обозначения:

- а) полный адрес;
- б) база сегмента (ds указывает на то, что значение базы берется из регистра ds);
- в) смещение сегмента;
- г) содержимое памяти [каждая строка отображает содержимое восьми последовательно расположенных ячеек памяти (байтов)]. Адрес указан только для первого из них. Для вычисления адресов последующих байтов памяти в каждой строке к смещению необходимо прибавить соответственно 1, 2, 3 и т.д).

2. Нажмите клавишу [F5], после чего окно центрального процессора займет весь экран.

3. Нажмите клавишу [F8]. При этом курсор должен опуститься на одну строку.

4. Нажимая клавишу [TAB], переместите курсор в подокно, отображающее фрагмент памяти. При этом активность подокна отмечается мерцающим прямоугольником:

- а) установите подокно на отображение области памяти с заданным адресом, для чего:
 - ◆ нажмите комбинацию клавиш [CTRL] + [G] (нажать [CTRL] и, не отпуская ее, клавишу [G]). На экране раскроется диалоговое окно с приглашением для ввода нужного адреса: *Enter address to position to (Введите адрес, на который установите окно)* (рис. Л-13.10);
 - ◆ введите с клавиатуры адрес ds: 00c0h, [Enter];



Рис. Л-13.10. Вид экрана Ввод начального адреса фрагмента памяти

- б) подведите курсор к конкретному байту памяти и начинайте ввод нужного значения *младшего* байта в формате single. В ответ, как показано на рис. Л-13.11, появится диалоговое окно с приглашением *Enter new data bytes (Введите новые байты данных)*, [Enter].

Продолжите ввод байтов, а по завершении ввода каждого из них нажимайте клавишу [Enter]. После ввода числа в формате single переходите к вводу его в форматах real и double соответственно, для чего необходимо повторить пп. а) и б) еще дважды. После этого происходит изменение содержимого памяти. Заполните столбец 6 табл. Л-13.3.



Рис. Л-13.11. Вид окна ввода данных

Так, например, число $X = -21$ в алгебраической форме имеет вид $X = 1.0101 \cdot 10^{+100}$ и в формате single в памяти отображается в виде $X: 00\ 00\ A8\ C1$ (см. табл. Л-13.3).

Для выполнения задания 2 повторите п. 4, а), изменяя соответственно значение адреса *ds*: на значения, заданные вариантом (см. табл. Л-13.4), и считайте числа из памяти в заданных форматах. Результаты сведите в табл. 13.5, пояснения к табл. Л-13.5 приведите в отчете.

Контрольные вопросы

1. Приведите формулу алгебраической записи вещественных чисел и охарактеризуйте ее составляющие (M , P , q).
2. Числа $X_1 = \pm 44$; $X_2 = \pm 5/128$; $X_3 = \pm 23.75$; $X_4 = 0.325$ представьте в классическом двоичном формате ($n = 6$, $m = 3$).
3. Определите диапазон чисел для формата, заданного в п. 2.
4. Найдите десятичный эквивалент числа (X_{10}), представленного в классическом двоичном формате:

- а)

1

1	0	0	0	1
n				

0

1	0	0
m		
- б)

0

1	0	0	0	1
n				

1

0	0	1
m		

5. Числа $X_1 \div X_4$ (см. п. 2) представьте в форматах single, real, double.

6. Найдите десятичный эквивалент числа, представленного в форматах single, real, double соответственно:
 - а) 000045C2;
 - б) 8600000000D4;
 - в) 00000000004025A0.
7. Нарисуйте формат single, real, double и дайте характеристику его полей.
8. Поясните понятие «смещенный порядок» и правило его вычисления для форматов single, real, double.
9. Запишите правило, позволяющее вычислить значение числа в форматах single, real, double.
10. Назовите достоинства и недостатки представления чисел в ФПТ.

Лабораторная работа 15

ФОРМАТЫ КОМАНД.

СПОСОБЫ АДРЕСАЦИИ ДАННЫХ

15.1. Цель работы

1. Изучение основных команд ПК на базе МП i286 и их форматов.
2. Изучение прямых способов адресации данных.
3. Приобретение практических навыков работы с командами.

15.2. Порядок выполнения работы

1. Изучение методических указаний к работе (дома). Дополнительная литература [1, 2, 10].
2. Подготовка отчета (дома).
3. Индивидуальный входной контроль (цель работы, понятие команды, формата команды, хранение данных в ЭВМ, команды МП i286, способы адресации операндов и команд, кодирование команд, команды отладчика TD).
4. Практическая работа на ПК:
 - а) регистрация;
 - б) загрузка в память компьютера (см. подраздел 15.5) программы реализации команд различных типов передач, подготовленной дома в соответствии с заданным вариантом задания (см. табл. Л-15.1);
 - в) выполнение и отладка исполняемого модуля программы с помощью TD (см. подраздел 15.5).
5. Отчет о проделанной работе:
 - ♦ индивидуальное выполнение на ПК заданной преподавателем команды передачи данных;

- ♦ корректировка отчета по результатам работы на ПК;
- ♦ защита отчета.

15.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ♦ понятие команды и формата команды;
- ♦ основные команды передачи данных в МП i286;
- ♦ понятие способа адресации, формирование $A_{исп}$ для рассмотренных способов адресации.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ♦ подробный протокол выполнения команд с различными способами адресации (в виде табл. Л-15.2);
- ♦ трасса исполнения команд.

15.4. Методические указания

ОСНОВНЫЕ ПОНЯТИЯ И ОПРЕДЕЛЕНИЯ

Информация, поступающая на вход ЭВМ, — числа, логические переменные, текст, графика и др., в том числе и команды (рис. Л-15.1) — вводится в компьютер в виде последовательности 0 и 1 и далее представляется и хранится в виде *двоичных слов фиксированной длины*: 8, 16, 32 бит.



Рис. Л-15.1. Информация в ЭВМ

Если в ЛР 11—12, 13—14 рассматривались способы представления (форматы) *числовых данных* с ФТ и ПТ, то в настоящей работе рассматриваются *форматы команд* и простейшие способы адресации данных, хранимых либо в регистрах ЦП, либо в ОП. Смысл двоичных слов может быть различным — либо число, либо адрес, либо команда.

Команда в ЭВМ, или машинная команда, — это упорядоченная последовательность бит (двоичное слово), с помощью которой указывают:

- ♦ наименование операции, инициируемой командой (код операции КОП хранится в ОЧ);
- ♦ адреса A_1 , A_2 операндов, участвующих в операции (хранятся в АЧ).

Формат команды представлен на рис. Л-15.2.

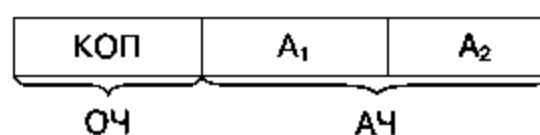


Рис. Л-15.2. Формат команды:

ОЧ — операционная часть; АЧ — адресная часть

Формат команды задает длину команды в байтах или словах, число полей для указания адреса, т.е. адресность команды, расположение полей в команде и т.д.

ТРЕБОВАНИЯ К ФОРМАТАМ КОМАНД

1. Краткость, так как длина команды влияет на объем памяти, отводимой для хранения команд.
2. Минимальное время выполнения.
3. Возможность обращения ко всей области памяти данных.

ХРАНЕНИЕ ДАННЫХ В ЭВМ

Исходные данные для решения задачи могут храниться либо в регистрах общего назначения (РОН) МП, либо в ОП (ячейки М). Ячейки (reg и М), хранящие исходную информацию, назовем *источником*, а регистры и ячейки памяти, принимающие информацию от источника, — *приемником*.

В МП i286 приемник расположен слева, источник — справа. Информация передается от источника к приемнику по стрелке (рис. Л-15.3, а).

Так, в байте команды, который содержит адреса регистров, хранящих данные при регистровом способе адресации, источник и приемник располагаются в соответствии с рис. Л-15.3, б; тип передачи: *регистр — регистр* или $\text{reg } 1 \leftarrow \text{reg } 2$.

(Рг $\leftarrow$ Рг). К другим типам относят передачи:

Приемник Источник

reg $\leftarrow$ М	регистр — память (Рг $\leftarrow$ П)
М $\leftarrow$ reg	память — регистр (П $\leftarrow$ Рг)
М ₁ $\leftarrow$ М	память — память (П $\leftarrow$ П)

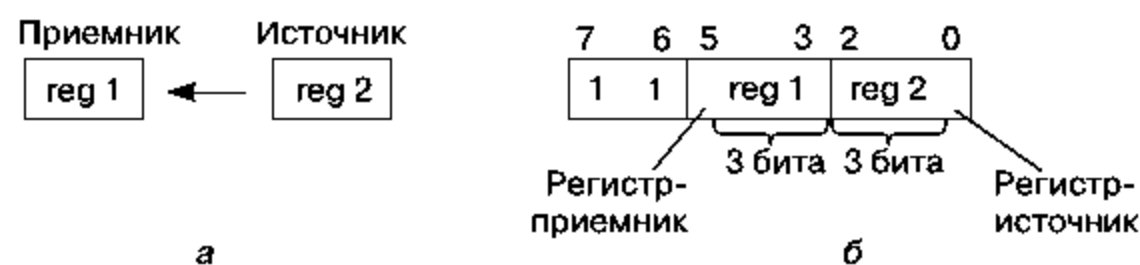


Рис. Л-15.3. Отношение источник-приемник:

а — направление передачи;

б — тип передачи: регистр—регистр

Пример. Если один из операндов, например К, хранится в регистре АХ, то в АЧ команды с регистровым способом адресации в поле (с 0 по 2 бит) будет записан код 000, т.е. адрес регистра АХ. Каждый из регистров имеет свой номер — *код* (см. табл. Л-15.3).

СПОСОБЫ АДРЕСАЦИИ

Данные (элементы информации), участвующие в операции, определяются с помощью адресов, указанных в командах. Существует большое число способов представления адресов в командах и их вычисления с целью доступа к операндам (данным) на основе информации, указанной в команде.

Правило вычисления адреса операнда на основе информации, указанной в команде, называют *способом адресации*. Таким образом, способ адресации определяет порядок выполнения действий над адресной частью команды и содержимым одного или нескольких регистров ЦП ПК для вычисления исполнительного адреса $A_{исп}$, по которому хранится операнд в памяти.

Под *исполнительным адресом* $A_{исп}$ понимается адрес операнда в памяти (М) или в блоке РОНов (АХ, ВХ, СХ, ДХ и др.). $A_{исп}$ — это целое двоичное число без знака, число разрядов n которого определяет возможное число ячеек памяти, к которым можно обращаться. Так, при $n = 16$ их число равно 2^{16} , что соответствует 64 КБ.

СПОСОБЫ ПРЯМОЙ АДРЕСАЦИИ

Регистровая — в АЧ команды задается номер регистра ЦП, в котором хранится операнд. Так, если это регистр СХ (рис. Л-15.4), то в поле (0—2) АЧ указывается его адрес (код) 001, который и является $A_{исп}$.

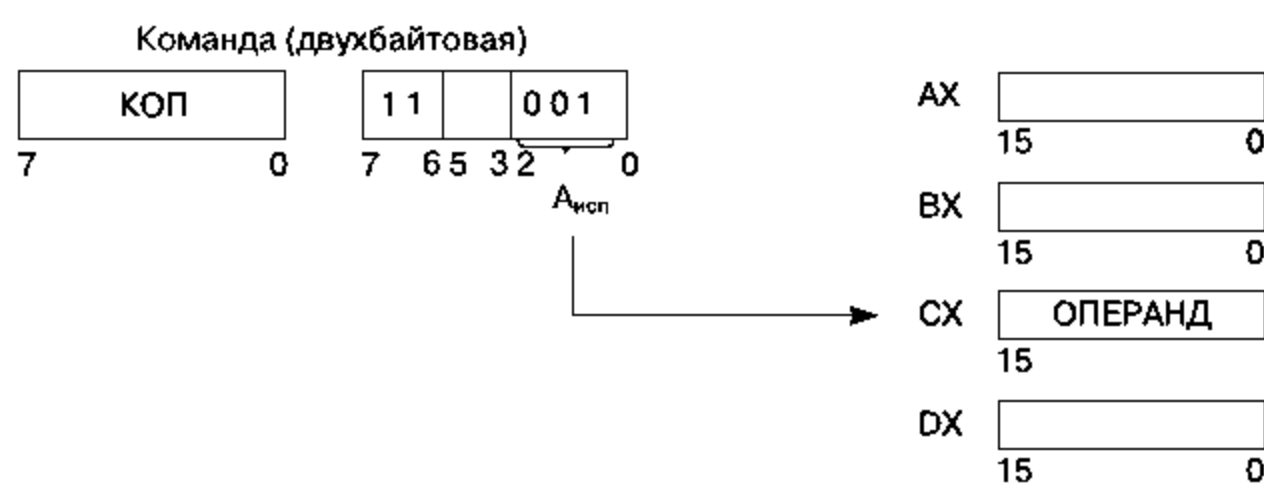
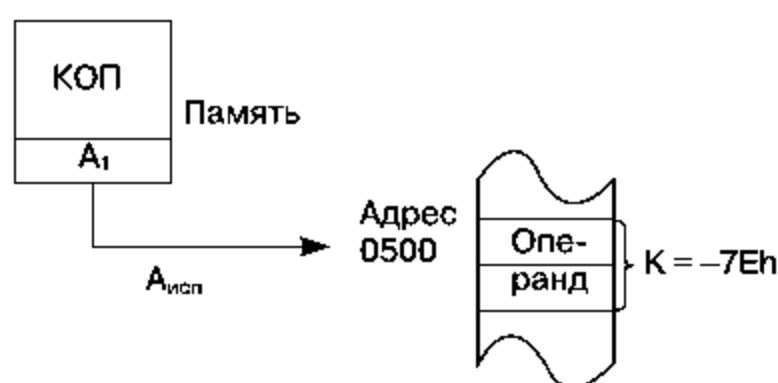


Рис. Л-15.4. Регистровая адресация

Абсолютная (прямая) — $A_{исп}$ задается в самой команде, обычно во 2-м или 3-м слове, операнд — в памяти. Структура определения $A_{исп}$ приведена на рис. Л-15.5.

Рис. Л-15.5. Структура вычисления $A_{исп}$

Пример. Команда с абсолютной адресацией $AX \leftarrow [M]500$ (в регистр AX переслать содержимое ячейки памяти с адресом 500, в которую занесена переменная, например $K = -7Eh$ в формате слова).

Команда, реализующая операцию, занимает в памяти ПК три ячейки (3 байта), (рис. Л-15.6, а), сам операнд K в ДК — 2 байта (рис. Л-15.6, б). Содержимое регистра AX до и после исполнения команды приведено на рис. Л-15.6, в.

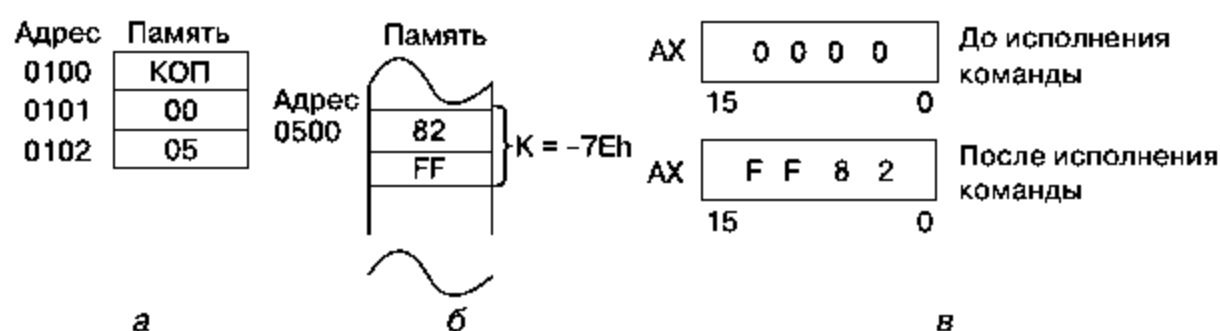


Рис. Л-15.6. Абсолютная адресация:

а — команда, реализующая операцию; б — операнд; в — содержимое регистра AX до и после исполнения команды

Непосредственная — операнд-константа задается непосредственно в самой команде (рис. Л-15.7) и может занимать 2 или 3 байта.

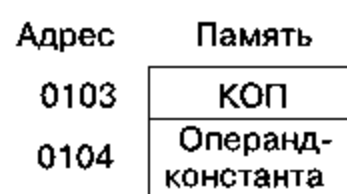


Рис. Л-15.7. Непосредственная адресация

КОДИРОВАНИЕ КОМАНД

Кодирование команд — процесс получения двоичного кода команды.

Ручное кодирование в общем случае достигается заменой каждого поля формата команды соответствующим двоичным полем и заме-

ной адресных полей двоичными значениями адресов. Далее каждый байт команды представляется в 16-ричной СС (два разряда).

В данной работе кодирование выполняется на основе усеченного набора команд, приведенного на рис. Л-15.11.

Пример. Закодировать команду, инициирующую операцию пересылки содержимого регистра ВХ в регистр АХ, т.е. $AX \leftarrow (BX)$.

Мнемоника команды: MOV AX, BX.

Команда передачи $reg1 \leftarrow reg2$ — двухбайтовая (см. рис. Л-15.11). Первый байт (КОП) ее приведен на рис. Л-15.8.

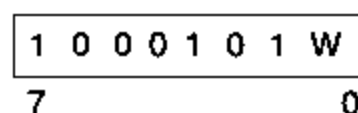


Рис. Л-15.8. Первый байт

Код первого байта 10001011 или 8Bh (в 16-ричной СС), так как регистры АХ, ВХ имеют длину слова $W = 1$. Код второго байта СЗ (рис. Л-15.9, а) получается в результате замены поля $reg1$ кодом регистра АХ = 000 и поля $reg2$ кодом регистра ВХ = 011 (см. табл. Л-15.3). Представление команды в памяти ПК приведено на рис. Л-15.9, б.

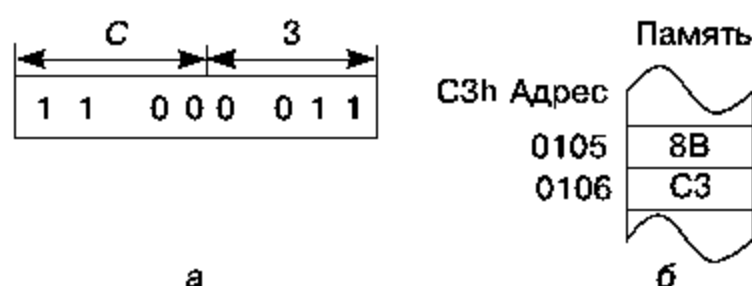


Рис. Л-15.9. Кодирование команды:

а — код второго байта; б — представление команды в памяти

15.5. Команды программы Far Manager и отладчика TD персональной ЭВМ

С помощью описанных ниже команд осуществляется ввод и отладка программ в машинных кодах микропроцессора МП86. В качестве примера используется программа вычисления выражения $M = 2K + N - R + 60$, приведенная в табл. Л-15.4.

СОЗДАНИЕ ИСХОДНОГО ФАЙЛА ПРОГРАММЫ

1. Для создания в системе Far Manager нового файла программы последовательно выполните пп. 2—4, при корректировке файла — пп. 5—7. Вызов системы производится следующим образом:

- ◆ щелкнуть мышью по кнопке *Пуск*;
- ◆ выбрать пункт *Выполнить...*;
- ◆ в появившемся окне ввести *far* и нажать [Enter].

2. При нажатой клавише [Shift] нажмите клавишу [F4], и программа Far Manager выдаст запрос. Введите в него имя файла и его расширение TXT и нажмите [Enter]. Например, PETROV.TXT, [Enter]. Еще раз нажмите [Enter] и попадете в редактор Far Manager.

3. С помощью команд встроенного редактора Far Manager наберите программу в машинных кодах:

A1 0005	↓	2BC2
03C0		05 6000
8BD8		A3 0405
A1 0205		90
03C1		

4. Чтобы сохранить созданный файл на диске, нажмите [F2], а затем [F10], и вы выйдете из редактора.

5. Если требуется скорректировать или дополнить известный файл, то в панели системы Far Manager курсором выделите имя этого файла и нажмите [F4] — попадете в редактор с вызванным файлом.

6. С помощью команд редактора Far Manager скорректируйте исходный файл. Например, перед машинной командой 90 введите еще несколько кодов 90 (пустая операция NOP).

7. Чтобы сохранить созданный файл на диске, нажмите [F2], а затем [F10], и вы выйдете из редактора.

Выделенный на панели Far Manager файл можно распечатать на принтере, если нажать [F5] — копировать, а затем при ответе на запрос ввести PRN (принтер) и нажать [Enter].

ПРЕОБРАЗОВАНИЕ ИСХОДНОГО МОДУЛЯ ПРОГРАММЫ В МАШИННЫХ КОДАХ В ИСПОЛНЯЕМЫЙ МОДУЛЬ ТИПА COM

С помощью команды

trans <имя файла> .TXT, [Enter]

запустите программу трансляции TRANS, которая преобразует коды команд МП86, представленные в виде ASCII-символов, в последовательность двоичных кодов команд, образующих COM-файл программы.

ВЫПОЛНЕНИЕ И ОТЛАДКА ИСПОЛНЯЕМОГО МОДУЛЯ ПРОГРАММЫ С ПОМОЩЬЮ TD

1. С помощью команды

TD <имя файла>.COM, [Enter]

запустите отладчик TD для работы с созданным COM-файлом программы. Отладчик загрузит в память исполняемый модуль типа COM

с адреса 100h, причем коды программы и данных разместятся в одном сегменте емкостью 64 Кбайт. После загрузки отладчик выдаст на экран окно центрального процессора CPU.

2. Нажав клавишу [Enter], снимите марку отладчика CPU. На экране отобразится окно отладчика CPU, состоящее из 5 подокон: кодового сегмента, содержащего коды программы; регистров; флажков; сегмента стека и сегмента данных. Клавишей [F5] увеличьте/уменьшите окно CPU. Двойной рамкой и находящимся в подокне маркером выделено активное подокно (или окно). Переход из одного подокна CPU в другое производится нажатием клавиши [TAB] или [Shift] + [TAB]. Можно перейти в подокно, нажимая [Shift] и одну из клавиш перемещения курсора.

Находясь в подокне, можно, нажав [ALT] + [F10], войти в локальное подменю и с помощью его команд изменить содержимое регистров и памяти. Нажав [F10], можно войти в главное меню отладчика и воспользоваться его командами для управления выполнением программы. Выход из меню производится клавишей [ESC].

Ниже рассматриваются основные команды отладчика для отладки программ с помощью окна CPU.

3. Нажав клавишу [TAB], перейдите в подокно регистров. Подведите маркер к регистру DX, введите код 30 и нажмите [Enter]. Тем самым вы ввели в DX операнд N = 0030h. (При вводе 16-ричных кодов необходимо, чтобы 1-й символ был цифрой, например 0FFFF).

4. Перейдите в подокно сегмента данных. Нажав [ALT] + [F10], вызовите для данного подокна локальное меню. Выберите в нем команды GOTO и нажмите [Enter]. Далее введите с клавиатуры начальный адрес данных (ds:500, [Enter]). Убедитесь в том, что маркер выделяет в подокне памяти байт с адресом 500, а затем введите с клавиатуры код 6000 0f0 0ff [Enter]. Тем самым вы ввели в память операнды K = +60 и K = -10.

Можно вводить данные в память в формате слова. Для этого с помощью команды [CTRL] + [D] войдите в подменю локального меню и задайте в нем формат Word — слово. Затем введите с клавиатуры необходимый операнд, например 60.

Отметим, что [CTRL] + [D] — это активная клавиша команд [ALT] + [F10]; Display as. Далее при работе с окнами отладчика будем в основном использовать активные клавиши, нажимая [CTRL] в сочетании с первой буквой команды локального меню.

5. Перейдите в подокно кода. В нем изображаются дизассемблированные команды выполняемой программы, причем текущая команда помечается стрелкой >. Для управления выполнением программы ис-

пользуют следующие основные команды, вызываемые активными клавишами:

- [F7] — выполнение одной команды;
- [F8] — выполнение одной команды с пропуском вызовов;
- [F9] — запуск программы в автоматическом режиме;
- [CTRL] + [F2] — установка программы в исходное состояние;
- [F2] — установка/отмена точки останова.

Нажимая клавишу [F7], выполните программу по шагам до команды NOP. При этом следите за содержимым регистров МП и памяти. Убедитесь в правильности полученного результата.

6. Нажав [CTRL] + [F2], установите программу в исходное состояние. Обратите внимание, что указатель > отмечает первую команду программы. Перегрузите исходные данные (см. п. 3—4). С помощью маркера выделите третью команду программы и нажмите [F2]: таким образом вы задали точку останова. Аналогичным образом установите точку останова на команде NOP. Для выполнения программы до 1-й точки останова нажмите [F9] (или [F4]). Далее выполните программу по шагам, последовательно нажимая [F7].

7. Нажав [CTRL] + [F2], установите программу снова в исходное состояние. Перезагрузите исходные данные (см. пп. 3—4). С помощью маркера и команды [F2] отмените точки останова. Маркером выделите команду программы ADD AX,0060, замените ее с клавиатуры на команду ADD AX,420 и нажмите [Enter]. Тем самым вы установили так называемую «заплату». Выполните модифицированную программу в пошаговом режиме.

РАБОТА С ГЛАВНЫМ МЕНЮ И ОКНАМИ ОТЛАДЧИКА

Выше были рассмотрены основные команды отладчика, вызываемые активными клавишами. В общем случае управление отладкой можно производить через главное меню отладчика. Это меню содержит позиции экрана: *File* (файлы), *View* (просмотр), *Run* (выполнение) и др. Вход в главное меню выполняется с помощью клавиш [F10] и выбора необходимой позиции. Вызов новых окон, например памяти (*Dump*) или регистров (*Registers*), производится через позицию *View*.

8. Выведите на экран окно памяти, для чего выполните команду [F10]; *View*; *Dump* [Enter]. Работа в этом окне аналогична подокну данных окна CPU.

9. Нажав [CTRL] + [F5], установите режим перемещения и изменения размеров текущего окна. Нажимая клавиши перемещения курсора, переместите окно. Этими же клавишами можно изменять размеры окна,

если держать нажатой клавишу [Shift]. Завершается этот режим нажатием клавиши [Enter].

10. Командой [F10]; *View; Registers* выведите на экран окно регистров *Registers*.

11. Нажимая последовательно [F6], можно перейти из одного окна в другое. Двойной рамкой или находящимся в нем маркером отмечается активное окно. Командой [ALT] + [F3] текущее окно удаляется. Ошибочно удаленное окно можно восстановить командой [ALT] + [F6]. Удалите все окна, кроме окна CPU.

12. Для выхода из отладчика нажмите [ALT] + [X].

15.6. Практическая работа на ПК (работа с командами)

Задание

1. Разработать в машинных кодах программу реализации команд *трех типов передачи данных* ($R_r \leftarrow R_r$, $R_r \leftarrow P$, $P \leftarrow R_r$) и *операции с константой* (сложить или вычесть). Коды команд, разрешенных к использованию в настоящей работе, приведены на рис. Л-15.10. Программу оформить *в виде* табл. Л-15.2 (*дома*).
2. Разработать регистровую модель реализации заданных команд. Источники хранения данных определяются вариантом задания (табл. Л-15.1). Исходя из заданного вариантом выражения, определить:
 - а) какие конкретно передачи вам необходимы (указать конкретные источники и приемники данных);
 - б) какая операция с константой выполняется. Тип операции с константой обусловлен вариантом задания. Значение константы в выражении задано в 16-ричной системе счисления;
 - в) операнды и результат представить в ДК в 16-ричной системе счисления. Значения операндов задаются преподавателем.
3. Изучить работу с командами программы Far Manager и отладчиком TD (*дома*). Описание программы (см. подраздел 15.5).

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Показать преподавателю разработанные *дома* программу реализации различных типов передач данных и операции с константой *в виде* табл. Л-15.2 и регистровую модель *в виде* рис. Л-15.11.
2. Регистрация.
3. Загрузить разработанную программу в память ЭВМ *по аналогии* с п. 1—7 (подраздел 15.5) и выполнить п. 1—7 раздела 15.5,

затем подраздел «Преобразование исходного модуля программы в МК в исполняемый модуль типа .COM», далее подраздел «Выполнение и отладка исполняемого модуля программы с помощью TD» и подраздел «Работа с главным меню и окнами отладчика».

4. Составить трассу программы, отображающую пошаговое изменение содержимого регистров, ячеек ОП и флажков (табл. Л-15.4). Число столбцов таблицы может быть и другим, что определяется числом источников и приемников информации, используемых при реализации заданной программы.

Таблица Л-15.1

Варианты заданий

№ варианта	Задание	Размещение операндов				Длина операнда	Начальный адрес программы
		M	K	R	N		
1	$M=K+R+2N-3$	Пам.	Пам.	Пам.	BH	Байт	100
2	$M=K-R-N+50$	Пам.	Пам.	DX	Пам.	Байт	100
3	$M=2K-N+R-6$	Пам.	CL	Пам.	Пам.	Байт	100
4	$M=K+3N-R+20$	Пам.	Пам.	Пам.	DX	Слово	100
5	$M=K-R-N-8$	Пам.	Пам.	CL	Пам.	Байт	100
6	$M=2K-R+N+18$	Пам.	DX	Пам.	Пам.	Слово	100
7	$M=2K-2R+N-10$	Пам.	Пам.	DL	Пам.	Байт	100
8	$M=120-K-R+N$	Пам.	CX	Пам.	Пам.	Слово	100
9	$M=20+K-2R+N$	Пам.	Пам.	Пам.	CH	Байт	100
10	$M=500+K-2R-N$	Пам.	BX	Пам.	Пам.	Слово	100

Таблица Л-15.2

Программа реализации команды типа $Rg \leftarrow \text{Пам}$

Адрес	Код	Мнемоника	Операция	Комментарий
CS:0100	A1	MOV AX,[0500h]	$AX \leftarrow [0500h]$	$AX \leftarrow K$
0101	00			
0102	05			
⋮	⋮	⋮	⋮	
⋮	⋮	⋮	⋮	
xxxx	90	NOP	Пустая операция	

Таблица Л-15.3

Адреса (коды) регистров МП

Адрес	Регистр	
	W = 1	W = 0
000	AX	AL
001	CX	CL
010	DX	DL
011	BX	BL
100	SP	AH
101	BP	CH
110	SI	DH
111	DI	BH

Таблица Л-15.4

Трасса исполнения команды MOV AX, [0500h]

Команда	Операция	AX	[M] ₅₀₀	[M] ₅₀₁	Флажки			
					C	S	O	Z
		0000*	A0*	FF*	0*	0*	0*	0*
1. MOV AX, [0500h]	AX ← [M] ₅₀₀	FFA0	A0	FF	0	0	0	0
2.	.							
3.	.							
.	.							
.	.							
N. NOP	Пустая операция							

* Исходное состояние.

1. Передача регистр-регистр MOV reg, reg

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
1	0	0	0	1	0	1		1	1	reg1						reg2							

2. Передача из памяти в регистр MOV reg, [EA]

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
1	0	1	0	0	0	0		Младшая часть EA								Старшая часть EA							

3. Передача из регистра в память MOV [EA], reg

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
1	0	1	0	0	0	1		Младшая часть EA								Старшая часть EA							

4. Сложить регистр-регистр ADD reg1, reg2

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
0	0	0	0	0	0	0	1	1	1	reg1						reg2							

5. Вычесть регистр-регистр Sub reg1, reg2

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
0	0	1	0	1	0	1		1	1	reg1						reg2							

6. Сложить константу с регистром ADD reg, data

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
0	0	0	0	0	0	1	0	data L								data H							

если W=1

7. Вычесть константу из регистра SUB reg, data

1-й байт								2-й байт								3-й байт							
7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0								
0	0	1	0	1	1	0		data L								data H							

если W=1

Рис. Л-15.10. Форматы команд с регистровой (1, 4, 5), непосредственной (6, 7) и прямой (2, 3) адресациями, используемые в данной работе

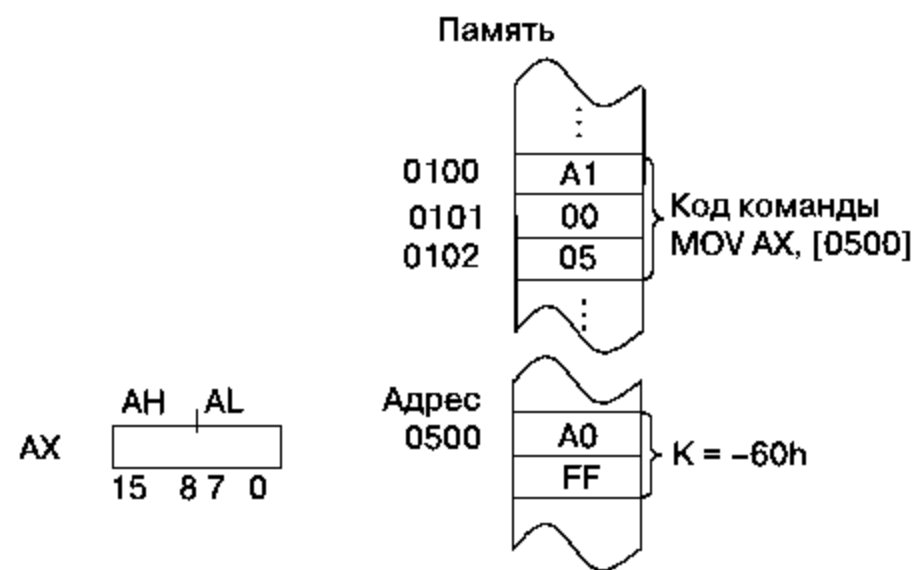


Рис. Л-15.11. Регистровая модель выполнения операции $AX \leftarrow [M]500$

Контрольные вопросы

1. Понятие машинной команды.
2. Что такое формат команды?
3. Требования к форматам команд.
4. Источники хранения данных в компьютере и типы передачи данных.
5. Способы прямой адресации в компьютере и их суть.
6. Понятие исполнительного адреса.
7. Нарисуйте структуру вычисления $A_{исп}$ для заданного способа адресации.
8. К чему сводится ручное кодирование команды? Привести пример.
9. Опишите порядок загрузки программ в машинных кодах в память компьютера.

Лабораторная работа 16

СТРУКТУРА КОМПЬЮТЕРА, ПРИНЦИП ФУНКЦИОНИРОВАНИЯ. ВЫПОЛНЕНИЕ АРИФМЕТИЧЕСКИХ ОПЕРАЦИЙ НА КОМПЬЮТЕРЕ

16.1. Цель работы

1. Изучение структуры компьютера и принципов его функционирования.
2. Выполнение арифметических операций в ЭВМ.
3. Разработка программы реализации арифметического выражения и ее отладка с помощью отладчика TD.

16.2. Порядок выполнения работы

1. Изучение методических указаний, подраздел 16.4 (дома). Дополнительная литература [1, 2, 10, 12].
2. Подготовка отчета (дома).

3. Индивидуальный входной контроль (цель работы, структурная организация ЭВМ, принцип функционирования ЭВМ, назначение основных узлов ЭВМ, основной цикл работы ЦП, исполнение команды, регистровая модель выполнения операции, организация памяти, принцип разработки программы в МК, отладчик TD).
4. Практическая работа на ПК:
 - ◆ регистрация;
 - ◆ создание исходного файла на диске (разработанную дома в соответствии с заданным вариантом программу вычисления арифметического выражения в МК ввести в ЭВМ);
 - ◆ коррекция исходного файла;
 - ◆ преобразование исходного модуля программы в МК в исполняемый модуль типа COM;
 - ◆ отладка исполняемого модуля с помощью отладчика TD;
 - ◆ загрузка исходных данных и исполнение программы (пошаговое и автоматическое);
 - ◆ снятие трассы отлаженной программы.
5. Работа с отлаженной программой по заданию преподавателя (смена исходных данных, модификация программы, различные варианты управления отладкой программы).
6. Коррекция отчета по результатам практической работы.
7. Защита отчета.

16.3. Содержание отчета

ТЕОРЕТИЧЕСКАЯ ЧАСТЬ:

- ◆ структурная организация ЭВМ (фоннеймановской и с системной шиной);
- ◆ принцип функционирования фоннеймановской ЭВМ;
- ◆ основной цикл работы ЦП;
- ◆ организация памяти.

ПРАКТИЧЕСКАЯ ЧАСТЬ:

- ◆ задание на разработку в соответствии с заданным вариантом;
- ◆ регистровая модель выполнения операции;
- ◆ программа вычисления заданного выражения (см. табл. Л-16.1);
- ◆ трасса программы.

16.4. Методические указания

СТРУКТУРНАЯ ОРГАНИЗАЦИЯ КОМПЬЮТЕРА

На рисунке Л-16.1 представлена структурная схема ЭВМ Дж. фон Неймана, включающая в себя устройства ввода-вывода (УВВ), основную память (ОП), внешнее запоминающее устройство (ВЗУ), центральное устройство управления (ЦУУ) и арифметико-логическое устройство (АЛУ). Все устройства совместно функционируют на основе принципа *программного управления*.

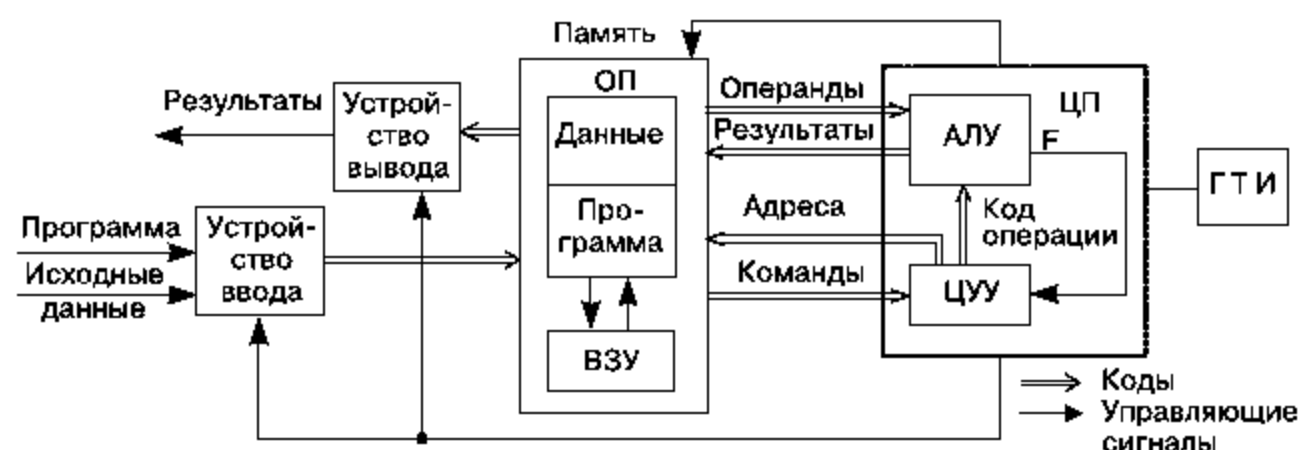


Рис. Л-16.1. Структура ЭВМ Дж. фон Неймана

Процесс решения любой задачи на ЭВМ определяется *алгоритмом* — однозначным предписанием последовательности операций, обеспечивающих решение задачи.

Устройство ЭВМ, непосредственно осуществляющее процесс переработки цифровой информации и управление им в соответствии с заданным алгоритмом, называется *центральным процессором* (ЦП). Для ЦП алгоритм представляется *программой* — последовательностью команд, реализующих алгоритм решаемой задачи. *Команда* имеет операционную (КОП) и адресную (АЧ) части (рис. Л-16.2).

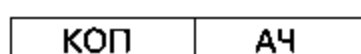


Рис. Л-16.2. Формат команды

Адресная часть (АЧ) указывает адреса ячеек ОП или регистров, в которых хранятся операнды (данные); операционная часть содержит код операции, характеризующий соответствующий тип операции, которую должен выполнить процессор.

ЦЕНТРАЛЬНЫЙ ПРОЦЕССОР (ЦП)

ЦП — основной блок ЭВМ (рис. Л-16.3). В нем можно выделить две основные части: управляющую (УЧ) и операционную (ОЧ). Операционная часть, в которой происходит преобразование информации (вы-

полняются элементарные операции из функционального полного набора), содержит необходимые для этого средства, в частности арифметико-логическое устройство (АЛУ) и различные регистры. Операции двоичной арифметики и булевой алгебры реализуются в АЛУ на уровне регистровых передач между регистрами-источниками операндов и регистром результата. Эти операции осуществляются за счет пересылки через комбинационную схему (КС), настроенную на определенную арифметическую операцию. УЧ формирует управляющие сигналы (УС) для настройки и коммутации узлов ОУ (рис. Л-16.5) в соответствии с выполняемой операцией, а также УС, необходимые для организации обмена информацией с ОП и внешними устройствами. Процессор реализует программное управление вычислительным процессом. Необходимую для этого управляющую информацию он получает в виде команд, хранимых в ОП. Программы вместе с исходными данными вводятся в ОП ЭВМ с помощью УВВ.

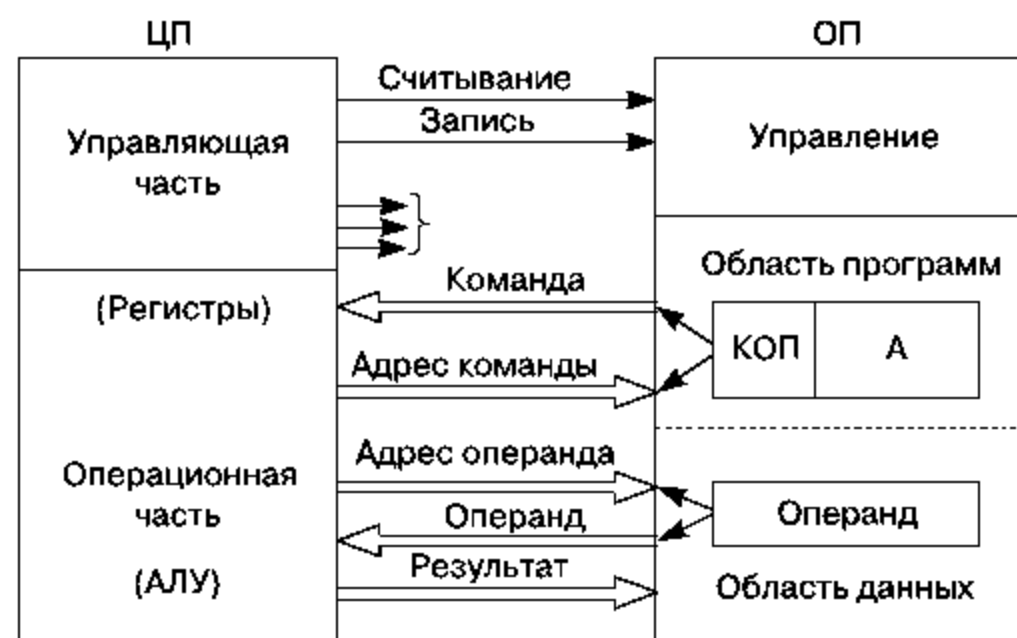


Рис. Л-16.3. Обобщенная структура процессора

Понятие «микропроцессор» (МП) в функциональном отношении совпадает с понятием ЦП и отражает лишь особенности, связанные с использованием технологии СБИС при его реализации.

РАБОТА ЦП

ЦП управляет процессом решения задачи на основе программы, считываемой из ОП, команда за командой, с последующим исполнением закодированных в них операций.

Выборка команд программы для исполнения, выборка операндов по адресам, указанным в команде, запуск операции на исполнение и запись результата в ОП поручаются управляющей части процессора, называемой центральным устройством управления (ЦУУ).

При обработке i -й команды программа ЦУУ реализует следующую *последовательность действий*:

- ◆ выборку очередной i -й команды;
- ◆ исполнение команды;
- ◆ выдачу и сохранение результата исполнения команды;
- ◆ подготовку следующей команды.

В состав ЦУУ входят счетчик команд (СК), хранящий адрес очередной команды, и регистр команд (РК). По адресу, содержащемуся в СК, из ОП извлекается очередная $i+1$ команда и записывается в РК.

Исполнение команды начинается с анализа двух полей команды — КОП и АЧ (см. рис. Л-16.2). Код операции настраивает АЛУ на выполнение заданной операции. Адреса поступают в ОП, по ним считываются операнды, участвующие в операции, и записывается результат. Заканчивается цикл, как правило, увеличением содержимого СК на количество байтов (длину) команды, подготавливая адрес следующей, $i+1$, команды. Результат операции в зависимости от типа команды может храниться в одном из регистров ЦП либо передаваться в ОП.

Способ адресации команд может быть *естественным* — адрес следующей команды формируется в СК путем увеличения адреса на длину текущей команды l , т.е. $СК \leftarrow СК + l$, либо *принудительным* (произвольным). При нарушении естественного порядка следования команд (разветвления, объединения, циклы) используют специальные команды управления переходами, в которых имеется *адрес перехода*.

СТРУКТУРА ПК

Структура ПК приведена на рис. Л-16.4 и является типовой для микропроцессорных систем и ПК на базе 16-разрядного МП.



Рис. Л-16.4. Типовая структура ПК на базе МП

Микропроцессорная система включает в себя ЦП на основе МП, память, подсистему ввода-вывода, системную шину (СШ), содержащую шину адреса (ША), шину данных (ШД) и шину управления (ШУ).

СТРУКТУРНАЯ СХЕМА МП

Основные этапы выполнения команды распределены внутри МП по двум сравнительно независимым устройствам — ОУ и УС (рис. Л-16.5).

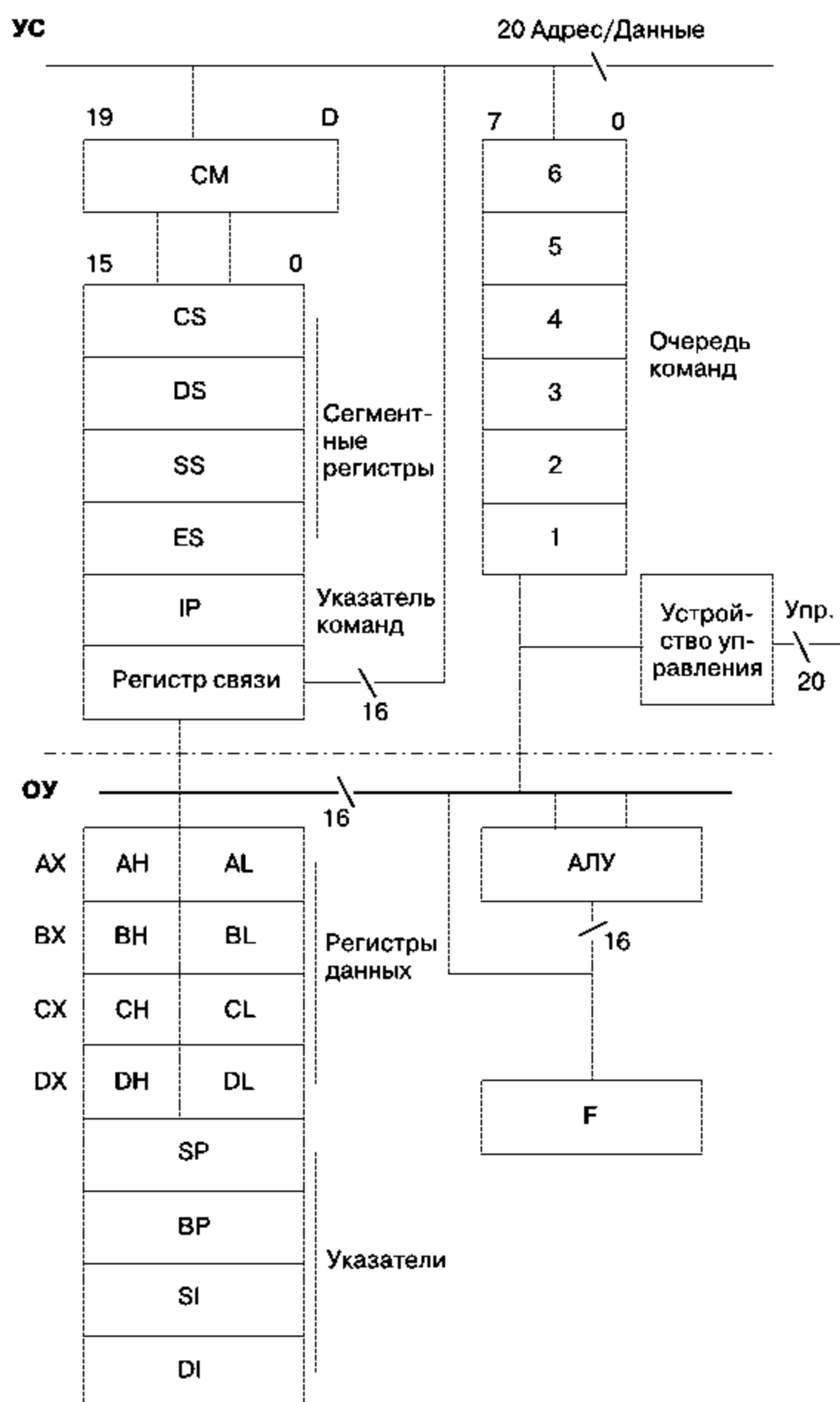


Рис. Л-16.5. Базовая структурная схема 16-разрядного МП

Операционное устройство (ОУ) содержит: 16-разрядные регистры данных AX, BX, CX, DX, указатели памяти SP, BP, SI, DI, арифметико-логическое устройство АЛУ и регистр признаков F (регистр флагов). Когда ОУ занято выполнением текущей команды, устройство сопряжения (УС) осуществляет опережающую выборку очередных команд из памяти. Команды хранятся во внутренней регистровой памяти, называемой очередью (буфером) команд, по существу выполняющей функцию регистра команд (РК) ЦП.

В состав УС входят 16-разрядные регистры CS, DS, SS, ES и сумматор CM, которые формируют 20-битный физический адрес памяти из двух 16-битных логических адресов: адреса сегмента (базы) и адреса смещения, называемого также *эффективным* (исполнительным) *адресом* EA. Схема вычисления физического адреса приведена на рис. Л-16.6. Значение EA складывается с содержимым сегментного регистра, сдвинутого относительно EA влево на 4 бита, как показано на рис. Л-16.6.

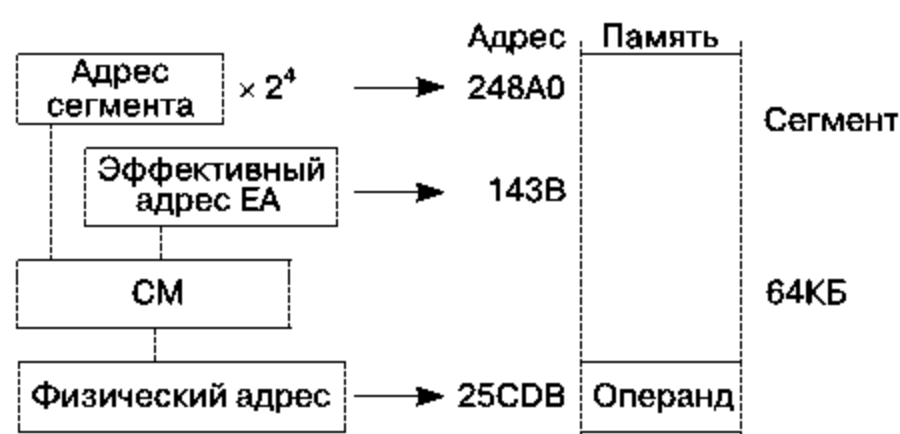


Рис. Л-16.6. Вычисление физического адреса

Если, например, содержимое сегментного регистра данных (DS)=248Ah, а указателя памяти (SI)=143Bh, то физический адрес будет равен 25CDBh:

$$\begin{aligned}
 &+ \quad 248A0 \text{ — начальный адрес сегмента данных } DS \times 2^4 \\
 &\quad \underline{143B} \text{ — эффективный адрес EA, хранящийся в регистре SI} \\
 &\quad 25CDB \text{ — физический адрес операнда в памяти.}
 \end{aligned}$$

16.5. Программная модель МП

В программной модели МП К1810ВМ86 (рис. Л-16.7) можно выделить четыре группы регистров: регистры данных, регистры сегментов, регистры указателей SP, BP и индексов SI, DI, регистр признаков F.



Рис. Л-16.7. Программная модель МП

Регистры указателей и регистры данных относят к регистрам общего назначения (РОН). В зависимости от того, чем оперирует команда: словами или байтами, регистры данных можно рассматривать как четыре 16-разрядных (AX, BX, CX, DX) или как восемь 8-разрядных регистров (AH, BH, CH, DH, AL, BL, CL, DL). L и H означают младшие (Low) и старшие (High) байты 16-разрядных регистров. Каждый из них помимо общих выполняет и специализированные функции: AX, AL — аккумулятор, BX — базовый регистр и т.д.

Память микросистемы на базе такого МП содержит сегменты по 64 Кбайт. МП может иметь дело одновременно с четырьмя типами сегментов: кода (команд) CS, данных DS, стека SS и дополнительного сегмента данных ES.

Регистр CS указывает на сегмент, содержащий текущую выполняемую команду. Для вычисления адреса следующей (с учетом очереди команд) исполняемой команды МП добавляет к содержимому CS, умноженному на 2^4 , содержимое указателя команд IP.

Регистр SS указывает на текущий сегмент стека, регистр сегмента данных DS — на текущий сегмент данных, обычно содержащий используемые в программе данные, а регистр ES — на текущий дополнительный сегмент, который используется для выполнения операций над цепочками.

Регистры указателей SP, BP и индексов SI, DI предназначены для хранения внутрисегментных смещений и обеспечивают косвенную адресацию данных в пределах текущего сегмента, но могут использоваться и как регистры данных.

Указатель команд IP адресует следующую команду программы (с учетом очереди команд) в сегменте кода CS (ведет себя как счетчик команд СК).

Регистр признаков F представлен на рис. Л-16.8.

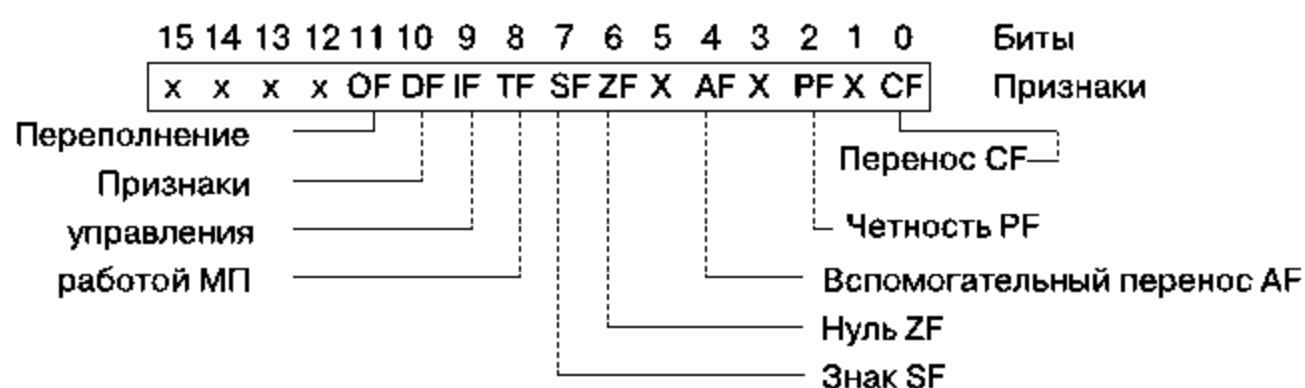


Рис. Л-16.8. Формат регистра признаков F

16.6. Логическая организация памяти

МП К1810 обеспечивает адресацию памяти емкостью 1 Мбайт (2^{20}).

На программном уровне память представляют как линейную последовательность из 2^{20} байт (рис. Л-16.9).

Адрес	Память	
00000	1C	Байт = 1C по адресу 00000
00001	5A	Слово = 7E5A по адресу 00001
00002	7E	
00003	4B	Байт = 4B по адресу 00003
00004	50	Слово = 6F50 по адресу 00004
00005	6F	
	.	
	.	
	.	
FFFFF		

Рис. Л-16.9. Логическая организация памяти

Два смежных байта образуют слово, адресом которого считается адрес младшего байта. Принцип «младшее по младшему адресу» сохраняется и для представления других единиц данных: двойных слов, многобайтных команд и т.д.

16.7. Состав системного ПО ПК

ПК с СК, например МП К1810 (i8086), является эффективным средством автоматизации процесса разработки и отладки прикладных программ микросистем, построенных на основе МП К1810.

Основные этапы разработки и отладки программы вычисления заданного выражения приведены на рис. Л-16.10.

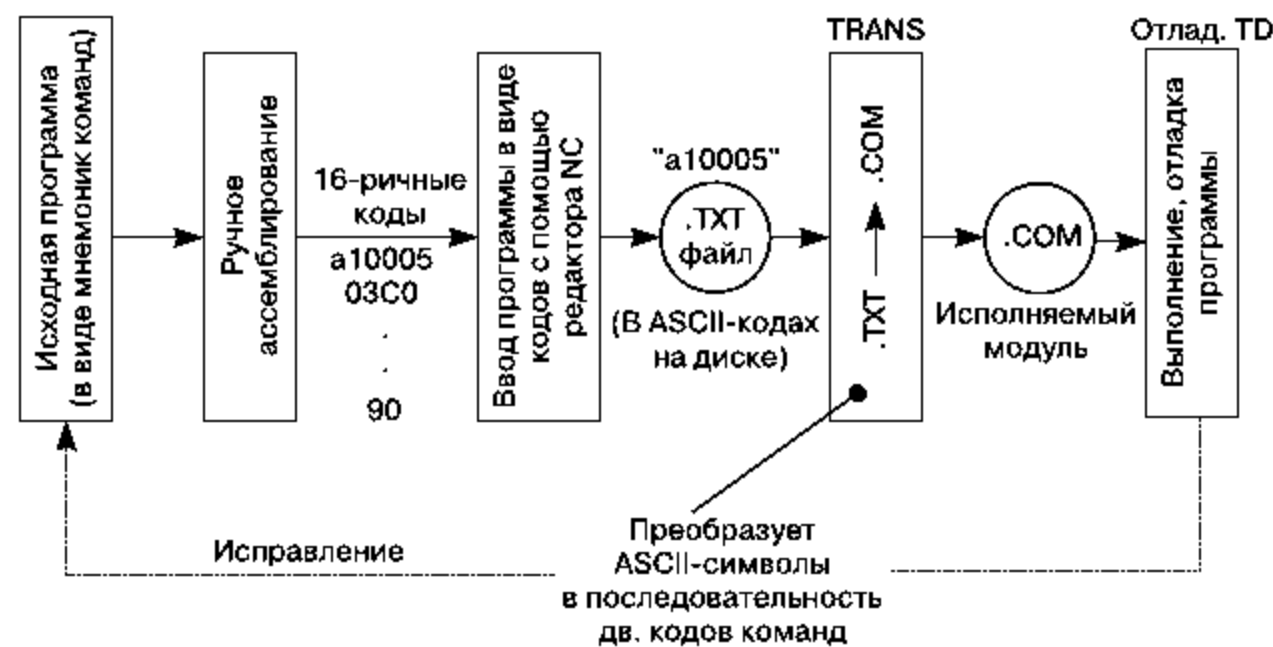


Рис. Л-16.10. Основные этапы разработки и отладки программ

16.8. Практическая работа на ПК

Реализация арифметического выражения (варианты заданий приведены в табл. Л-16.4).

Пример.

Разработать программу для вычисления выражения

$$M = 2K + N - R + 40, \quad (1)$$

в котором все операнды и результат — двухбайтные данные. Переменные K, N, M размещены в ОП, в сегменте данных, а R — в регистре DX. Числа со знаком представляются в дополнительном коде (ДК). Например, ДК чисел $K = +60h$, $R = -10h$, $N = -30h$ соответственно равны $[K]_{\text{ДК}} = 0060h$, $[R]_{\text{ДК}} = FFF0h$, $[N]_{\text{ДК}} = FFD0h$. Результат $[M]_{\text{ДК}} = 00E0h$.

1. РАЗРАБОТКА РЕГИСТРОВОЙ МОДЕЛИ

В регистровой модели (рис. Л-16.11) приводится перечень регистров и ячеек памяти для хранения исходных данных, промежуточных и окончательного результатов при реализации заданного выражения (1).

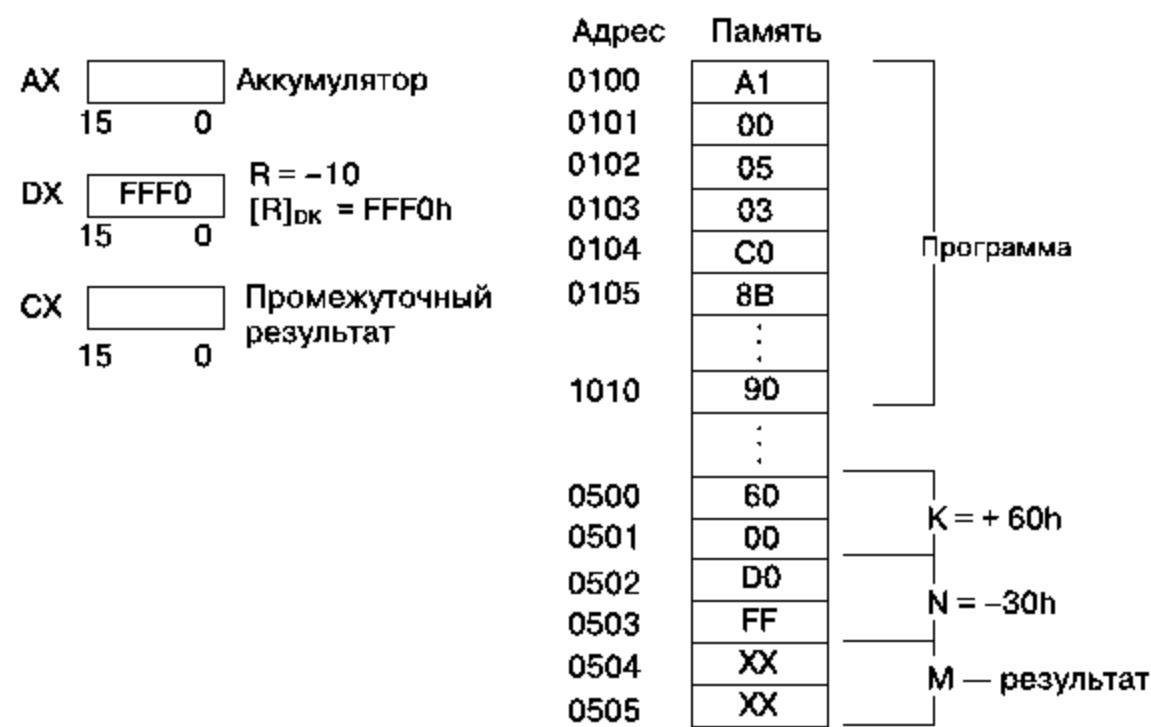


Рис. Л-16.11. Регистровая модель МП для реализации выражения (1)

1. Передача регистр—регистр MOV reg1, reg2	Биты
1-й байт 7 6 5 4 3 2 1 0	2-й байт 7 6 5 4 3 2 1 0
1000101W	11 reg1 reg2
2. Передача константы в регистр MOV reg, data	
1011W reg	data L data H, если W = 1
3. Передача из памяти в аккумулятор MOV acc, [EA]	
1010000W	Младшая часть EA Старшая часть EA
4. Передача из аккумулятора в память MOV [EA], acc	
1010001W	Младшая часть EA Старшая часть EA
5. Сложить регистр—регистр ADD reg1, reg2	
0000001W	11 reg1 reg2
6. Вычесть регистр—регистр SUB reg1, reg2	
0010101W	11 reg1 reg2
7. Сложить константу с аккумулятором ADD acc, data	
0000010W	data L data H, если W = 1
8. Вычесть константу из аккумулятора SUB acc, data	
0010110W	data L data H, если W = 1

Рис. Л-16.12. Форматы команд с регистровой (1, 5, 6), непосредственной (2, 7, 8) и прямой (3, 4) адресацией, используемые в данной работе

2. РАЗРАБОТКА ПРОГРАММЫ ВЫЧИСЛЕНИЯ ВЫРАЖЕНИЯ

Программа реализации выражения (1) оформлена в виде таблицы Л-16.1.

Табл. Л-16.1 разрабатывается на основе регистровой модели (см. рис. Л-16.11) и имеющегося заданного перечня команд с указанными форматами (см. рис. Л-16.12). При программировании на машинном

языке коды команд и данных записываются в 16-ричной СС. Адреса регистров МП приведены в табл. Л-16.2. Примеры кодирования команд приводились в лабораторной работе 15.

Таблица Л-16.1

Программа вычисления выражения $M = 2K + N - R + 40$

Адрес	Коды команд	Мнемоника	Операция	Комментарий
CS:0100	A1	MOV AX, [0500h]	$AX \leftarrow [M]_{500}$	$AX \leftarrow K$
0101	00	ADD AX, AX	$AX \leftarrow (AX) + (AX)$	$AX \leftarrow K + K$
0102	05			
0103	03			
0104	C0			
0105	8B	MOV CX, AX	$CX \leftarrow (AX)$	$CX \leftarrow 2K$
0106	C8	MOV AX, [0502h]	$AX \leftarrow [M]_{502}$	$AX \leftarrow N$
0107	A1			
0108	02			
0109	05			
010A	03	ADD AX, CX	$AX \leftarrow (AX) + (CX)$	$AX \leftarrow N + 2K$
010B	C1	SUB AX, DX	$AX \leftarrow (AX) - (DX)$	$AX \leftarrow N + 2K - R$
010C	2B			
010D	C2			
010E	05			
010F	40	ADD AX, [0040h]	$AX \leftarrow (AX) + 40$	Сл. с конст.
0110	00	MOV [0504h], AX	$[M]_{504} \leftarrow (AX)$	$[M]_{504} \leftarrow \text{результат}$
0111	A3			
1000	04			
1001	05			
1010	90	NOP	Пустая операция	
DS:0500	60	$K=0060$, операнд $K=+60$		
0501	00	$N=0030$, операнд $N=+30$		
0502	30			
0503	00			
0504	XX			
0505	XX	$M=XXXX$, результат M		
POH	Данные			
DX	FFF0	$R=FFF0$, операнд $R=10$		

Таблица Л-16.2

Адреса регистров МП

Адрес регистра (reg)	Регистр	
	W = 1	W = 0
000	AX	AL
001	CX	CL
010	DX	DL
011	BX	BL
100	SP	AH
101	BP	CH
110	SI	DH
111	DI	BH

Начальный эффективный адрес программы равен 100. Этому адресу соответствует физический адрес $(CS \times 2^4) + 100$. Далее будем обозначать его как CS:100. Эффективные адреса данных лежат в диапазоне DS:0500...DS:0505.

3. ТРАССА ПРОГРАММЫ

Трасса отлаженной программы отображает пошаговое изменение содержимого регистров, ячеек ОП и флажков при ее исполнении и может быть представлена в виде табл. Л-16.3.

Таблица Л-16.3

Трасса программы

Мнемо-ника	Операция	Регистры			Память (М)						Флажки			
		AX	DX	CX	500	501	502	503	504	505	C	S	O	Z
		0000*	FFF0*	0000*	60*	00*	00*	00*	00*	00*	0*	0*	0*	0*
MOV AX [500]	AX ← [M] ₅₀₀	0060	FFF0	0000	60	00	00	00	00	00	0	0	0	0
⋮														
NOP	—													

* Исходное состояние.

16.9. Отладчик TD

Отладчик TD — это программа, находящаяся на диске ПЭВМ. Отладчик позволяет управлять процессом исполнения пользовательской программы. Команды TD, вводимые с клавиатуры ПЭВМ, позволяют выводить на экран и изменять содержимое памяти и регистров МП, исполнять программу по шагам (командам) и др. Команды отладчика будут рассмотрены при выполнении работ.

16.10. Варианты заданий

Таблица Л-16.4

Варианты заданий

Выражение	Размещение операндов*				Длина операнда	Начальный адрес программы
	М	К	Р	N		
1. $M = K + R + 2N - 3$	Память			ВН	Байт	100
2. $M = K - R - N + 50$	Память		DX	Память	Байт	100
3. $M = 2K - N + R - 6$	Память	CL	Память		Байт	100
4. $M = K + 3N - R + 20$	Память			DX	Слово	100
5. $M = K - R - N - 8$	Память		CL	Память	Байт	100
6. $M = 2K - R + N + 18$	Память	DX	Память		Слово	100
7. $M = 2K - 2R + N - 10$	Память		DL	Память	Байт	100
8. $M = 120 - K - R + N$	Память	CX	Память		Слово	100
9. $M = 20 + K - 2R + N$	Память			CH	Байт	100
10. $M = 500 + K - 2R - N$	Память	BX	Память		Слово	100

* Значения переменных K, R, N задаются преподавателем.

16.11. Порядок выполнения работы

С помощью описанных ниже команд осуществляется ввод и отладка программы в машинных кодах микропроцессора МП86. В качестве примера используется программа вычисления выражения $M=2K+N-R+60$, приведенная в табл. Л-16.1.

СОЗДАНИЕ ИСХОДНОГО ФАЙЛА ПРОГРАММЫ

1. Для создания в системе Far Manager нового файла программы последовательно выполните пп. 2—4, при корректировке файла — пп. 5—7. Вызов системы производится командой Far Manager, [Enter].

2. При нажатой клавише [Shift] нажмите клавишу [F4] — программа FAR MANAGER выдаст запрос. Введите в него имя файла и его расширение TXT и нажмите [Enter]. Например, PETROV.TXT, [Enter]. Еще раз нажмите [Enter] и попадете в редактор Far Manager.

3. С помощью команд встроенного редактора Far Manager наберите программу (см. табл. Л-16.1) в машинных кодах:

```
a1 0005
03c0
0306 0205
2bc2
05 6000
a30405
90
```

4. Чтобы сохранить созданный файл на диске, нажмите [F2], а затем [F10], и вы выйдете из редактора.

5. Если требуется скорректировать или дополнить известный файл, то в панели системы FAR MANAGER курсором выделите имя этого файла и нажмите [F4], вы попадете в редактор с вызванным файлом.

6. С помощью команд редактора FAR MANAGER скорректируйте исходный файл. Например, перед машинной командой 90 введите еще несколько кодов 90 (пустая операция NOP).

7. Чтобы сохранить созданный файл на диске, нажмите [F2], а затем [F10], и вы выйдете из редактора.

Выделенный в панели FAR MANAGER файл можно распечатать на принтере, если нажать [F5] — копировать, а затем при ответе на запрос PRN (принтер) и нажать [Enter].

**ПРЕОБРАЗОВАНИЕ ИСХОДНОГО МОДУЛЯ ПРОГРАММЫ
В МАШИННЫХ КОДАХ В ИСПОЛНЯЕМЫЙ МОДУЛЬ ТИПА COM**

8. С помощью команды

```
trans <имя файла>.txt.␣
```

запустите программу трансляции TRANS, которая преобразует коды команд МП86, представленные в виде ASCII-символов, в последовательность двоичных кодов команд, образующих COM-файл программы.

**ВЫПОЛНЕНИЕ И ОТЛАДКА ИСПОЛНЯЕМОГО МОДУЛЯ ПРОГРАММЫ
С ПОМОЩЬЮ ТУРБО-ОТЛАДЧИКА TD**

9. С помощью команды

TD <имя файла>.COM ↵

запустите отладчик TD для работы с созданным COM-файлом программы. Отладчик загрузит в память исполняемый модуль типа COM с адреса 100h, причем коды программы и данных разместятся в одном сегменте емкостью 64 Кбайт. После загрузки отладчик выдаст на экран окно процессора CPU.

10. Нажав [Enter], снимите марку отладчика. На экране отобразится окно отладчика CPU, состоящее из 5 подокон: кодового сегмента, содержащего коды программы; регистров; флажков; сегмента стека и сегмента данных. Клавишей [F5] увеличьте/уменьшите окно CPU. Двойной рамкой и находящимся в подокне маркером выделено активное подокно (или окно). Переход из одного подокна CPU в другое производится нажатием клавиши [Tab] или [Shift] + [Tab]. Можно перейти в подокно, нажимая [Shift] и одну из клавиш перемещения курсора.

Находясь в подокне, можно, нажав [ALT] + [F10], войти в локальное подменю и с помощью его команд изменить содержимое регистров, памяти. Нажав [F10], можно войти в главное меню отладчика и воспользоваться его командами для управления выполнением программы. Выход из меню производится клавишей [ESC].

Ниже рассматриваются основные команды отладчика для отладки программ с помощью окна CPU.

11. Нажав клавишу [Tab], перейдите в подокно регистров. Подведите маркер к регистру DX, введите код 30 и нажмите [Enter]. Тем самым вы ввели в DX операнд N = 0030h. (При вводе 16-ричных кодов необходимо, чтобы 1-й символ был цифрой, например 0FFFF.)

12. Перейдите в подокно сегмента данных. Нажав ALT + [F10], вызовите для данного подокна локальное меню. Выберите в нем команду GOTO и нажмите [Enter]. Далее введите с клавиатуры начальный адрес данных ds:500, [Enter]. Убедитесь в том, что маркер выделяет в подокне памяти байт с адресом 500, а затем введите с клавиатуры коды 60 00 0F0 0FF, [Enter]. Тем самым вы ввели в память операнды K = +60 и R = -10.

Можно вводить данные в память в формате слова. Для этого с помощью команды [CTRL] + [D] войдите в подменю локального меню и задайте в нем формат WORD — слово. Затем введите с клавиатуры необходимый операнд, например 60.

Отметим, что [CTRL] + [D] — это активная клавиша команд [ALT] + [F10]; Display as. Далее, при работе с окнами отладчика будем в основном использовать активные клавиши, нажимая [CTRL] в сочетании с первой буквой команды локального меню.

13. Перейдите в подокно кода. В нем отображаются дизассемблированные команды выполняемой программы, причем текущая команда помечается стрелкой >. Для управления выполнением программы используют следующие основные команды, вызываемые активными клавишами:

- [F7] — выполнение одной команды;
- [F8] — выполнение одной команды с пропуском вызовов;
- [F9] — запуск программы в автоматическом режиме;
- [CTRL] + [F2] — установка программы в исходное состояние;
- [F2] — установка/отмена точки останова.

Нажимая клавишу [F7], выполните программу по шагам до команды NOP. При этом следите за содержимым регистров МП и памяти. Убедитесь в правильности полученного результата.

14. Нажав [CTRL] + [F2], установите программу в исходное состояние. Обратите внимание, что указатель > отмечает первую команду программы. Перезагрузите исходные данные (см. пп. 11—12). С помощью маркера выделите 3-ю команду программы и нажмите [F2]: таким образом вы задали точку останова. Аналогичным образом установите точку останова на команде NOP. Для выполнения программы до 1-й точки останова нажмите [F9] (или [F4]). Далее выполните программу по шагам, последовательно нажимая [F7].

15. Нажав [CTRL] + [F2], установите программу снова в исходное состояние. Перезагрузите исходные данные (см. пп. 11—12). С помощью маркера и команды [F2] отмените точки останова. Маркером выделите команду программы ADD AX,0060 и замените ее с клавиатуры на команду ADD AX,420 и нажмите [Enter]. Тем самым вы установили так называемую заилату. Выполните модифицированную программу в пошаговом режиме.

РАБОТА С ГЛАВНЫМ МЕНЮ И ОКНАМИ ОТЛАДЧИКА

Выше были рассмотрены основные команды отладчика, вызываемые активными клавишами. В общем случае управление отладкой можно производить через главное меню отладчика. Это меню содержит позиции экрана: File (файлы), View (просмотр), Run (выполнение) и др.

Вход в главное меню выполняется с помощью клавиши [F10] и выбора необходимой позиции. Вызов новых окон, например памяти (Dump) или регистров (Registers), производится через позицию View.

16. Выведите на экран окно памяти, для чего выполните команду [F10]; View; Dump [Enter]. Работа в этом окне аналогична подокну данных окна CPU.

17. Нажав [CTRL] + [F5], установите режим перемещения и изменения размеров текущего окна. Нажимая клавиши перемещения курсора, переместите окно. Этими же клавишами можно изменять размеры окна, если держать нажатой клавишу [Shift]. Завершается этот режим нажатием клавиши ↵.

18. Командой [F10]; View; Registers выведите на экран окно регистров Registers.

19. Нажимая последовательно [F6], можно перейти из одного окна в другое. Двойной рамкой или находящимся в нем маркером отмечается активное окно. Командой [ALT] + [F3] текущее окно удаляется. Ошибочно удаленное окно можно восстановить командой [ALT] + [F6]. Удалите все окна, кроме окна CPU.

20. Для выхода из отладчика нажмите [ALT] + [X].

21. Разработайте для заданного варианта задания программу вычисления выражения в машинных кодах МП86 и по аналогии с п. 1—20 введите ее в ЭВМ и выполните. Исходные данные и результат представьте в ДК.

Контрольные вопросы

1. Приведите структуру ЭВМ Дж. фон Неймана и поясните назначение входящих в нее блоков.
2. Приведите обобщенную структурную схему ЦП и поясните принцип его работы.
3. Поясните основной цикл работы ЦП.
4. Приведите типовую структуру ПК на базе МП К1810 и поясните ее.
5. Назовите способы адресации команд.
6. Поясните принцип программного управления.
7. Поясните принцип функционирования ЭВМ, используя разработанную вами программу реализации заданного выражения.
8. Поясните понятие «регистровая модель выполнения операции».
9. Поясните, как получен код одной из команд в разработанной вами программе реализации заданного выражения.
10. Поясните принцип логической организации памяти, формирования физического адреса операнда.

11. Поясните, как создали исходный файл и записали его на диск.
12. Назовите состав СПО и основные этапы разработки и отладки программы.
13. Что представляет собой трасса отлаженной программы?
14. Как выполняется отладка программы?
15. Поясните, как установились арифметические флажки в одной из строк трассы программы.

ЛИТЕРАТУРА

1. *Акулов О.А., Медведев Н.В.* Информатика: базовый курс : учебник для вузов. М. : Омега-Л, 2004.
2. *Бройдо В.Л.* Вычислительные системы, сети и телекоммуникации : учебник для вузов. 2-е изд. СПб. : Питер, 2005.
3. *Галкин В.А., Григорьев Ю.А.* Телекоммуникации и сети : учеб. пособие для вузов. М. : Изд-во МГТУ им. Н.Э. Баумана, 2003.
4. *Герасименко В.А.* Основы информационной грамоты. М. : Энергоатомиздат, 1996.
5. *Гульев И.* Компьютерные вирусы, взгляд изнутри. М. : ДМК, 1998.
6. *Игнатов В.А.* Теория информации и передачи сигналов : учеб. пособие для вузов. М. : Радио и связь, 1991.
7. Информатика : учебник. / под ред. Н.В. Макаровой. 3-е изд., перераб. М. : Финансы и статистика, 2003.
8. Информатика : Практикум по технологии работы на компьютере / под ред. Н.В. Макаровой. 3-е изд., перераб. М. : Финансы и статистика, 2005.
9. Информатика : Базовый курс / под ред. С.В. Симоновича. СПб. : Питер, 2004.
10. *Каган Б.М.* Электронные вычислительные машины и системы : учеб. пособие для вузов. 3-е изд., перераб. и доп. М. : Энергоатомиздат, 1991.
11. *Кузин Л.Т.* Основы кибернетики. Т. 1. Математические основы кибернетики : учеб. пособие для вузов. М. : Энергия, 1973.
12. *Пятибратов А.П., Гудыно Л.П., Кириченко А.А.* Вычислительные системы, сети и телекоммуникации : учебник. 3-е изд., прераб. и доп. / под ред. А.П. Пятибратова. М. : Финансы и статистика, 2005.
13. *Савельев А.Я.* Основы информатики : учебник для вузов. М. : Изд-во МГТУ им. Н.Э. Баумана, 2001.
14. *Советов Б.Я.* Теория информации. Теоретические основы передачи данных в АСУ. Л. : Изд-во Ленинградского ун-та, 1997.
15. *Столингс В.* Компьютерные системы передачи данных : пер. с англ. 6-е изд. М. : Вильямс, 2002.
16. <http://www.rsreu.ru/content/view/1729/1288> — Дополнение к лабораторному практикуму учеб. пособия Н.И. Иопы «Информатика». М. : Кнорус, 2011.