Министерство образования и науки Российской Федерации
Новосибирский государственный технический университет

ИНФОРМАТИКА И ПРОГРАММИРОВАНИЕ

Конспект лекций
для студентов 1 курса специальности 351400 «Прикладная информатика»
дневного и заочного отделения факультета бизнеса
(1 семестр)

Новосибирск
2007

Автор – к.т.н., доцент Кобылянский В.Г.

Работа подготовлена кафедрой программных систем
и баз данных

СОДЕРЖАНИЕ

1. Классификация аппаратных и программных средств ЭВМ

Предмет информатики. Машина Неймана: основные принципы и свойства Классификация аппаратных и программных средств. Форма представления информации в ЭВМ Краткая история развития средств вычислительной техники.. Классификация современных ЭВМ и их основные технические характеристики. Основные характеристики процессоров и устройств памяти. Классификация современного программного обеспечения (ПО). Способы распространения ПО: коммерческое, условно-бесплатное, свободное. ПО с открытым кодом..

1.1 Предмет информатики, основные термины и определения

Информатика — это дисциплина, изучающая структуру и общие свойства информации, а также закономерности и методы ее создания, поиска, хранения, преобразования, передачи, защиты и применения в различных сферах человеческой деятельности. Часто используется синоним - *computer science* (компьютерная наука). На методах информатики основаны современные информационно-коммуникационные технологии, использующие возможности компьютерной техники и каналов передачи данных. Практическое применение методов информатики охватывает области, связанные с разработкой, созданием, использованием и материально-техническим обслуживанием информационных систем различного назначения.

Информатика - научная дисциплина с широчайшим диапазоном применения. Её основные направления:

- разработка вычислительных систем и их программного обеспечения;
- теория информации, изучающая процессы, связанные с передачей, приемом, преобразованием и хранением информации;
- методы искусственного интеллекта, позволяющие создавать программы для решения задач, которые требуют интеллектуальных усилий (логический вывод, обучение, понимание речи, визуальное восприятие, игры и др.);

- методы и средства мультимедиа, включающие одновременное использование машинной графики, анимации, видео, звуковых эффектов, живой речи и т.д.;
- средства телекоммуникации, в том числе, глобальные компьютерные сети, объединяющие все человечество в единое информационное сообщество;
- разнообразные приложения, охватывающие бизнес, производство, науку, образование, медицину, торговлю, сельское хозяйство и другие виды хозяйственной и общественной деятельности.

Средства, на которых базируется информатика, обычно объединяют в следующие группы:

- технические средства;
- программные средства;
- алгоритмические средства.

Технические средства (hardware) — это аппаратная часть компьютеров и средств коммуникаций.

Программные средства (software) - это совокупность всех программ, используемых компьютерами, а также вся область деятельности по их созданию и применению.

Алгоритмические средства (brainware) – область информатики, связанная с разработкой алгоритмов решения задач и изучением методов и приемов их построения.

Алгоритм - это последовательность действий, приводящих к решению задачи.

Данные - представление информации в виде, пригодном для конкретной обработки.

Знания - данные, полученные из других данных путем логических рассуждений.

Информация передается в виде *сообщений* от источника к потребителю посредством канала связи. Источник посылает сообщение, которое кодируется в передаваемый сигнал. Этот сигнал посылается по каналу связи. В приёмнике он декодируется и обрабатывается.

Количество информации в передаваемом кодированном сообщении определяется минимальной длиной двоичного кода, необходимого для его передачи, и определяется по формуле Хартли:

$$\mathbf{I = \log_2 N}$$

где N - количество возможных вариантов сообщения.

Допустим, нужно передать сообщение о состоянии погоды в одном из 4-х заранее оговоренных вариантов: ясно, пасмурно, идет дождь, идет снёг. Количество информации в нашем сообщении оказывается равным $I = 1og_2 4 = 2$. Это означает, что сообщение будет представлено двузначным кодом в виде одного из следующих четырех двоичных чисел:

Код сообщения	Содержание
00	Ясно
01	Пасмурно
10	Идет дождь
11	Идет снег

Минимальная единица информации называется битом, она соответствует сообщению длиной один двоичный разряд (например, сообщение о результате жеребьевки с помощью подбрасывания монеты). В информатике устоялась более крупная единица измерения количества информации - *байт,* состоящий из восьми битов. Сообщение длиной в один байт несет количество информации, равное 8, и позволяет передать от одного до 2^8=256 возможных вариантов. На практике используются в более крупные единицы, производные байта:

1 Килобайт (Кб) = 1024 байт - 2^{10} байт,

1 Мегабайт (Мб) = 1024 Кб =, 2^{20} байт,

1 Гигабайт (Гб) «= 1024 Мб = 2^{30} байт,

1 Терабайт (Тб) = 1024 Гб = 2^{40} байт,

1 Петабайт (Пб) = 1024 Тб = 2^{50} байт.

Для вычисления количества информации в сообщении в общем случае американский ученый Клод Шеннон предложил формулу, в которой учитываются неодинаковые вероятности вариантов одного и того же сообщения, что характерно

для текстовых сообщений, ввиду того, что буквы алфавита имеют разную вероятность появления в речи:

$$\mathbf{I = -(\ p_1\ 1og_2\ p_1 + p_2\ 1og_2\ p_2 + \ldots\ p_N\ 1og_2\ p_N +)}$$

- где p_i, - вероятность того, что именно i-е сообщение выделено в наборе из N вариантов. Если вероятности p_i, ---,p_N вариантов сообщения одинаковы, то каждая из них равна 1/N, и формула Шеннона превращается свой частный случай — формулу Хартли.

Предметы, процессы, явления реального мира, рассматриваемые с точки зрения их информационных свойств, называются информационными *объектами.*

Обработка информации — получение одних информационных объектов из других информационных объектов путем выполнения некоторых преобразований. Обработка является одной из основных операций, выполняемых над информацией. Следствием обработки может быть увеличение объема и разнообразия информации.

Средства обработки информации - это всевозможные устройства и системы, основными из которых являются электронно-вычислительные машины (ЭВМ), представляющие собой универсальный аппаратно-программный комплекс для обработки информации. Вместо термина «ЭВМ» часто используется его синоним - ***компьютер.*** Компьютеры обрабатывают информацию путем выполнения программ, которые реализуют специально составляемые для этих целей алгоритмы.

1.2 Машина Неймана

Вероятно, первым человеком, сформулировавшим идею создания универсальной вычислительной машины, был Чарльз Бэббидж. В середине 19 века он детально проработал проект аналитической машины для проведения вычислений, который не был завершен по финансовым причинам. Его коллега Ада Лавлейс, дочь известного поэта Байрона, стала первым математиком – программистом для этой аналитической машины.

В 30-40-х годах 20 века необходимость создания универсальной вычислительной машины возникла особенно остро в связи с широким использованием вычислений в военных приложениях (артиллерия, навигация и т.д.). Основные идеи организации ЭВМ были выдвинуты в 1933- 1937 годах математиками Аланом Тью-

рингом и Джоном фон Нейманом в терминах гипотетической машины с достаточно простой структурой (рис.1).

Основные принципы машины Неймана могут быть сформулированы следующим образом:

управление осуществляется по заранее подготовленной программе - последовательности команд, исполняемой арифметико – логическим устройством; команды исполняются последовательно, естественный порядок выполнения команд может быть изменен командой перехода;

для выполнения программы ее надо разместить в памяти ЭВМ и инициировать выполнение первой команды;

для размещения программы и данных машина имеет прямоадресуемую линейную одномерную память, представляющую последовательность ячеек памяти, каждая из которых имеет уникальный номер (адрес); наименьшая адресуемая единица памяти – байт;

вся хранимая в памяти информация делится на команды и данные; команды указывают действие (что делать), а данные – это объекты, над которыми это действие совершается;

команды и данные хранятся в памяти без каких-либо опознавательных признаков, т.е. прочитав значение любого участка памяти, нельзя однозначно сказать, что хранится в этом участке – команда, число или строка символов.

Достоинством машины Неймана является простота реализации, а недостатком – высокая вероятность возникновения ошибок при выполнении программ из-за неправильной интерпретации данных. Для уменьшения этой вероятности современные языки программирования предусматривают необходимость описания типов всех используемых в программе переменных.

1.3 Краткая история развития ЭВМ

История развития вычислительной техники начинается с 1946 года, когда в США была создана первая электронно-вычислительная машина ENIAC, которая

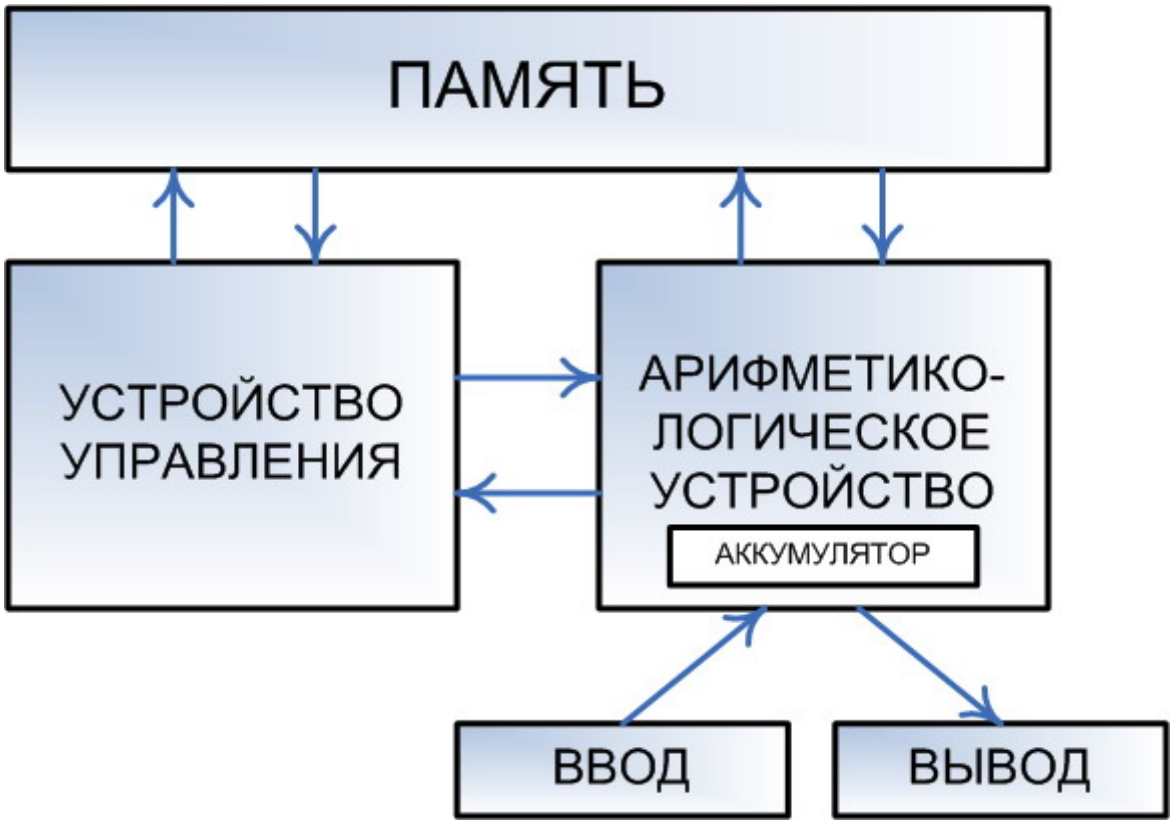


Рис. 1.1

имела по современным меркам совершенно «детские» технические характеристики – быстродействие процессора порядка 1000 операций в секунду, объем оперативной памяти – около 4 кбайт. Машина содержала около 18000 электронных ламп, имела объем 85 куб. м., вес около 30 тонн и имела среднее время наработки на отказ около 6 минут.

Первый персональный компьютер появился в 1977 году и был создан фирмой Apple, он весил около 7 кг и стоил 1350 долларов. К сожалению, невысокие технические характеристики не позволили ему выдержать конкуренцию с персональным компьютером фирмы IBM, появившемся в 1979 году и имевшим следующие характеристики: процессор Intel 8086 с тактовой частотой 4,77 Мгц; оперативная память 64 Кбайт; 1 дисковод для дискет 160 Кбайт; стоимость – 3000 долларов.

В таблице 1.3 приведены краткие данные по истории развития процессоров для персональных компьютеров на платформе Intel.

Таблица 1.3

Процессор	**Год выпуска**	**Частота на момент выхода**	**Количество транзисторов, тыс. шт.**	**Разрядность регистров**	**Размер кэш-памяти**
8086	1979	4,77 МГц	29	16-разрядные регистры общего назначения (GP)	-

80286	1982	12,5 МГц	134	16-разрядные GP	-
80386DX	1985	20 МГц	275	32-разрядные GP	-
80486DX	1989	25 МГц	1200	32-разрядные GP, 80-разрядные FPU	8 КБ L1
Pentium	1993	60 МГц	3100	32-разрядные GP, 80-разрядные FPU	16 КБ L1
Pentium Pro	1995	150 МГц	5500	32-разрядные GP, 80-разрядные FPU	16 КБ L1; 256, 512, 1К L2
Pentium II	1997	266 МГц	7000	32-разрядные GP, 80-разрядные FPU	32 КБ L1; 256, 512 КБ L2
Pentium III	1999	500 МГц	8200	32-разрядные GP, 80-разрядные FPU	32 КБ L1; 256, 512 КБ L2
Pentium IV	2005	1400 МГц	42000	32-разрядные GP, 80-разрядные FPU	32 КБ L1; 1024 L2
Pentium D	2005	2800 МГц	н/д	32-разрядные GP, 80-разрядные FPU	32 КБ L1; 2048 L2
Core Solo	2006	1660 МГц	н/д	32-разрядные GP, 80-разрядные FPU	32 КБ L1; 2048 L2

Здесь обозначено: GP – регистры общего назначения, FPU – регистры для выполнения действий с вещественными числами, L1 и L2 – кэш-память первого и второго уровня соответственно, н/д – нет данных.

В России также проводятся работы по созданию отечественных процессоров серии «Эльбрус». В Институте точной механики и вычислительной техники под руководством Б.А. Бабаяна еще в 1990 году был создан процессор Эльбрус-3 с лучшими техническими характеристиками, чем у процессора Intel Pentium, который появился в 1993 году только благодаря тому, что один из разработчиков Эльбруса В. Пентковский начал работать в фирме Intel. В 2000 году был создан процессор Эльбрус-2000, который по всем показателям превосходил аналогичную разработку Intel Itanium. К сожалению, проблемы финансового и технологического характера не позволили запустить этот процессор в серийное производство.

1.4 Классификация аппаратных средств ЭВМ

В настоящее время парк ЭВМ очень разнообразен – от малогабаритных карманных компьютеров до суперЭВМ. В основу классификации компьютеров могут быть положены следующие признаки:

- вид используемой для обработки информации;
- элементная база.
- габаритные характеристики;
- назначение;

По виду информации ЭВМ делятся на аналоговые и цифровые. В аналоговых машинах каждой переменной соответствует один электрический сигнал, который сколь угодно точно может представлять значение этой переменной. Преобразования значений переменной в основном выполняются аппаратно с помощью набора прецизионных устройств (усилителей, интеграторов, дифференциаторов, потенциометров и т.д.). Достоинством аналоговой ЭВМ является высокая скорость решения задач, для которых она предназначена (математические и имитационные задачи, описываемые системами сложных дифференциальных уравнений). Увеличение скорости достигается за счет использования параллельной обработки отдельных частей задачи отдельными комплектами устройств. Недостатки аналоговой ЭВМ – неспособность решения задач общего типа, низкая точность результатов, отсутствие возможности хранения больших объемов информации.

В цифровых ЭВМ значение каждой переменной представляется набором ограниченного числа сигналов (8, 16, 32 или 64), каждый из которых может принимать значение 0 или 1. Поэтому значение переменной всегда представлено с ограниченной точностью и всегда имеет ограничения по диапазону представляемых значений. Например, при использовании 8 сигналов переменная может принимать значение от 0 до 255.

Современные ЭВМ в подавляющем большинстве являются цифровыми. Аналоговые ЭВМ применяются только для решения специальных задач.

Классификация ЭВМ по элементной базе (по поколениям) приведена в таблице 1. Интегральные схемы (ИС) представляют собой электронные компоненты, на

подложке которых расположены несколько десятков транзисторов и полупроводниковых диодов, БИС имеют уровень интеграции порядка нескольких тысяч элементов, а СБИС – несколько миллионов элементов в одном кристалле. Например, ядро процессора Pentium IV включает более 6 млн. элементов.

Таблица 1.1

Номер поколения	Годы применения ЭВМ	Элементная база	Среднее быстродействие (оп/сек)
I	до 1960	электронные лампы	до 10 тыс.
II	до 1970	транзисторы, диоды	до 100 тыс.
III	до 1980	интегральные схемы (ИС)	до 1 млн
IV	до 1990	большие интегральные схемы (БИС)	до 5 млн
V	после 1990	сверхбольшие интегральные схемы (СБИС)	более 5 млн

По габаритным характеристикам ЭВМ делятся на карманные (КПК), переносные (notebook), настольные (desktop) и большие (mainframe).

По назначению ЭВМ могут быть персональными (домашними или профессиональными), рабочими станциями, серверами или суперЭВМ. Персональные компьютеры являются универсальными ЭВМ и обеспечивают, в основном, работу одного пользователя. Рабочие станции используются в составе автоматизированных рабочих мест проектировщиков и конструкторов; основные требования, предъявляемые к ним – мощная графическая подсистема вывода информации и высокая производительность. Серверы – это компьютеры, предназначенные для хранения больших объемов информации и ее обработки, поэтому они должны иметь большие объемы внешней памяти и также высокую производительность.

СуперЭВМ – компьютеры, имеющие очень высокую производительность и предназначенные для решения особо сложных задач. Это самая малочисленная группа ЭВМ из-за их очень высокой стоимости. Существует два типа суперЭВМ – одиночные многопроцессорные ЭВМ и распределенные кластерные системы. Ко-

личество процессоров в многопроцессорных ЭВМ может достигать нескольких тысяч.

В 2005 году первое место в рейтинге многопроцессорных суперЭВМ занимал компьютер Blue Gene/L фирмы IBM, показавший производительность 135,5 триллиона оп/сек. Он занимает 64 стойки (около 30 $м^2$) и объединяет в себе 130 тысяч процессоров с тактовой частотой 700 МГц, Основное назначение Blue Gene/L – решение задач моделирования физических явлений, которым требуется вычислительная мощность, намного превышающая возможности современных компьютеров, а также задач в области космологии, исследований поведения двойных звезд, взаимодействия лазера с плазмой и процесса старения ядерного оружия. Стоимость Blue Gene/L составляет около $100 млн.

Кластерная система представляет собой множество обычных персональных компьютеров, соединенных между собой высокоскоростной специализированной сетью и одновременно решающих одну задачу. Кластерная система уступает в производительности многопроцессорным суперЭВМ, но существенно выигрывает по стоимости.

1.5 Основные технические характеристики ЭВМ

Технические характеристики ЭВМ определяются характеристиками устройств, входящих в ее состав - центрального процессора, памяти, устройств ввода – вывода информации.

1.5.1 Характеристики процессора

Центральный процессор выполняет основную работу по выполнению программ пользователей в отличие от других процессоров, имеющихся в ЭВМ (например, собственные процессоры имеют видеокарта и клавиатура). Основными его характеристиками являются производительность и разрядность. Производительность определяет объем работы, выполненный в единицу времени, т.е. скорость решения задачи, а разрядность – точность полученного решения.

Производительность измеряется количеством выполняемых в единицу времени операций или путем использования специальных тестов, например SPECint2000 или SPECfp2000. SPEC - это корпорация, созданная в 1988 году, объединяющая

ведущих производителей вычислительной техники и программного обеспечения с целью разработки и стандартизации методов оценки производительности современных компьютеров. Разработанные SPEC тестовые пакеты являются стандартами для оценки производительности современных микропроцессоров, компьютеров и системного программного обеспечения. Тест SPECint2000 оценивает производительность на целочисленных операциях, а SPECfp2000 - на операциях с плавающей точкой.

Производительность компьютера зависит от многих факторов, в том числе от тактовой частоты процессора. Этот параметр часто используется на бытовом уровне для оценки производительности. Тактовые частоты современных процессоров могут достигать 3 ГГц.

Разрядность процессора определяет объем данных, передаваемых в процессор для выполнения команд. У современных процессоров разрядность может быть равна 32, 64 или 128 бита, более ранние процессоры имели разрядность 16 битов.

1.5.2 Характеристики памяти

Основными характеристиками устройств памяти являются объем, время доступа к информации и энергозависимость. Объем определяет максимальное количество информации, которое можно разместить в устройстве; время доступа – это интервал времени, в течение которого можно считать информацию из устройства; энергозависимость определяет, способно ли устройство хранить информацию при выключенном состоянии компьютера.

Память ЭВМ может быть двух типов – оперативные запоминающие устройства (ОЗУ) и внешние запоминающие устройства (ВЗУ). В качестве ОЗУ используются специальные микросхемы – модули памяти, а в качестве ВЗУ – различные виды дисков (магнитные, оптические, магнито – оптические и т.д.). Основные параметры этих устройств приведены в таблице 1.2

Таблица 1.2

	ОЗУ	**ВЗУ**
Объем	до 2 гбайт	до 500 гбайт
Время доступа	до 10 нсек	до 10 мсек
Энергозависимость	нет	да

Из таблицы 1.2 видно, что ОЗУ характеризуются относительно небольшим объемом по сравнению с ВЗУ, но по быстродействию существенно превосходят их (в миллион раз !) . Поэтому центральный процессор компьютера работает только с оперативной памятью, где размещаются программы и данные, которые необходимы для выполнения текущей работы. Все остальные программы и данные хранятся во внешней памяти.

На рис. 1.2 показана типовая схема обработки текстового документа, хранящегося в виде файла на магнитном диске:

- выполняется операция «Открыть документ», во время которой информация с диска копируется в оперативную память;
- проводится обработка документа с помощью текстового редактора в ОЗУ;
- выполняется операция «Сохранить документ», которая копирует измененный документ из оперативной памяти во внешнюю для долговременного хранения.

Рис 1.2 Типовая схема обработки текстового документа

1.6 Классификация программного обеспечения ЭВМ

Программное обеспечение (ПО) – это компонент, который «оживляет» аппаратную часть. Именно благодаря программному обеспечению компьютер может выполнять обработку различных видов данных (числовых, текстовых, звуковых, графических изображений и т.д.). Совокупность аппаратной части и ПО называется *вычислительной системой*.

Сегодня стоимость установленного ПО часто превышает стоимость аппаратной части ЭВМ, а общая стоимость ПО, разработанного в мире, исчисляется десятками миллиардов долларов.

Классифицировать ПО можно по назначению и по способу распространения. На рис. 1.3 приведена классификация по назначению

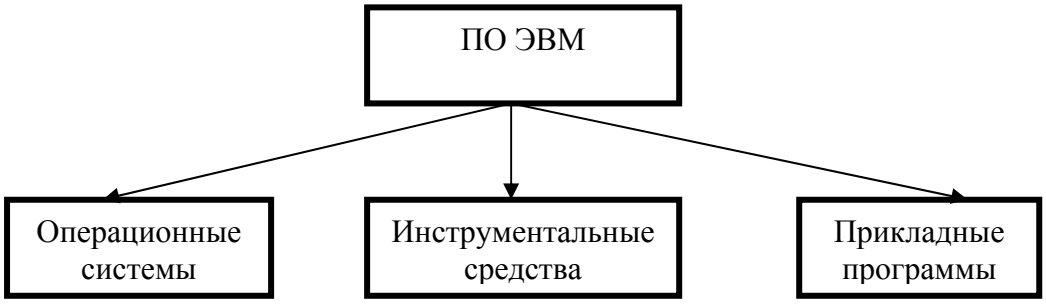


Рис. 1.3

Операционные системы – комплексы программ, предназначенных для эффективного управления всеми ресурсами ЭВМ и обеспечения интерфейса с пользователем. Ресурсы могут быть аппаратными, программными и информационными. К аппаратным ресурсам относятся процессор, память, принтер и.т.д.; программные ресурсы – набор программ, установленных в компьютере; информационные - информация, хранящаяся на устройствах внешней памяти в виде файлов. Примерами операционных систем являются MS DOS, MS Windows, Linux, OS/2 и др.

Инструментальные средства – это программы, предназначенные для решения типовых задач, возникающих у большинства пользователей. К ним относятся:

- файловые процессоры – программы, предназначенные для выполнения типовых операций при работе с файлами и каталогами (Norton Commander, FAR, Проводник и др.);
- редакторы – программы, предназначенные для создания и обработки различного рода документов (текстовых, графических, звуковых и т.д.) ; примерами редакторов являются MS Word, WordPad, Paint, Блокнот Windows и др.;
- электронные таблицы – программы, предназначенные для решения задач, у которых данные представлены в табличной форме (MS Excel);
- системы управления базами данных (СУБД) – программы, предназначенные для создания, ведения и обработки совокупности специальным образом организованных данных, которые называются базой данных; хранение данных в базе позволяет исключить дублирование и противоречивость данных. Примерами СУБД являются MS Access, MS SQL, Interbase, Oracle и др.;

- интегрированные системы – пакеты программ, состоящие из нескольких инструментов, например MS Office содержит текстовый редактор MS Word, графический редактор MS Power Point, электронную таблицу MS Excel, СУБД MS Access и другие программы;
- системы программирования – пакеты программ, предназначенные для разработки нового программного обеспечения и содержащие, как правило, текстовый редактор, компиляторы, компоновщик, отладчики и другие специальные программы. Примерами таких систем являются системы Delphi, C++ Builder, Visual Basic, Prolog и т.д.;
- системы автоматизированного проектирования, предназначенные для автоматизированного проектирования объектов в различных предметных областях: машиностроении (AutoCAD), архитектуре и строительстве (ArchiCAD), электронике (PCAD) и др.;
- системы моделирования и обработки данных - предназначены для математической обработки данных и моделирования (MatCAD, MatLab, Statistica и др.).

Прикладные программы – программы, предназначенные для решения частных прикладных задач (бухгалтерские, технологические, складские и т.д.).

По способу распространения программное обеспечение делится на коммерческое, условно - бесплатное (shareware), свободно распространяемое (freeware) и содержащее рекламу (adware). Коммерческое ПО распространяется только на платной основе, таким способом могут распространяться как программы, так и отдельные виды лицензий на эти программы. Такое ПО может поставляться с закрытым или с открытым исходным кодом.

Условно – бесплатное ПО обычно распространяется в два этапа: на первом этапе пользователь получает демонстрационную версию программы, которая может иметь ограниченный срок использования или ограниченный набор функций, а на втором этапе он уже может приобрести полнофункциональный продукт.

Свободно распространяемое ПО поставляется с открытым исходным кодом в соответствии с лицензией GNU GPL, согласно которой пользователь может изучать

код программы, модифицировать его, передавать другим пользователям и использовать его фрагменты в своих разработках, но только на условиях лицензии GPL.

Программы addware являются также бесплатными, но обычно содержат большое количество рекламных баннеров. Если пользователь такой программы хочет избавиться от этой рекламы, он должен связаться с разработчиком и отдельно оплатить работу по «очистке» программы.

2. Хранение данных в оперативной памяти

Логические и физические характеристики объектов в ОЗУ (идентификатор, значение, тип, область видимости). Статическое и динамическое выделение памяти. Виды объектов - простые (числовые, символьные, логические), сложные (массивы, записи, классы), многовариантные. Форматы представления целых чисел. Прямой, обратный, дополнительный коды представления целых чисел. 16- и 32- х разрядные типы данных. Экспоненциальная форма представления вещественных чисел, нормализованная дробь. Мантисса. Порядок. Экспонента. Внутренние форматы представления вещественных чисел. Основные операции над числами и символами. Представление символов в памяти ЭВМ. Кодировка символов (ASCII, ANSI, Unicode). Два варианта хранения символьных строк (String, PChar). Многовариантные типы объектов: объединения, данные типа Variant.

2.1 Классификация объектов в оперативной памяти

Вся информация, хранящаяся в оперативной памяти, делится на команды, исполняемые процессором, и данные. Команды формируются компилятором, а данные формируются и описываются программистом. Компилятор преобразует программу, написанную программистом на языке высокого уровня, в язык машинных команд, а также резервирует участки памяти для хранения данных. Далее будем понимать под объектом в ОЗУ только данные.

Любой объект в ОЗУ имеет логические и физические характеристики, которые приведены в таблице 2.1.

Имя объекта определяет физический адрес участка оперативной памяти, который выделяется для его хранения; значение адреса называется указателем на объект. Состояние – это значение, хранимое в ячейках ОЗУ, отведенных для объекта. Тип предполагает три свойства: единый внутренний формат (схему расположения битов внутри участка памяти), единый набор операций; диапазон допустимых значений. Область видимости определяет множество программ и подпрограмм, кото-

рые могут работать с объектом. В языках программирования объекту соответствует понятие типа данных

Таблица 2.1

Логические характеристики	Физические характеристики
Имя	Адрес в ОЗУ
Состояние	Значение
Тип	Внутренний формат
Область видимости	Область видимости

Классификация объектов в ОЗУ приведена на рис. 2.1

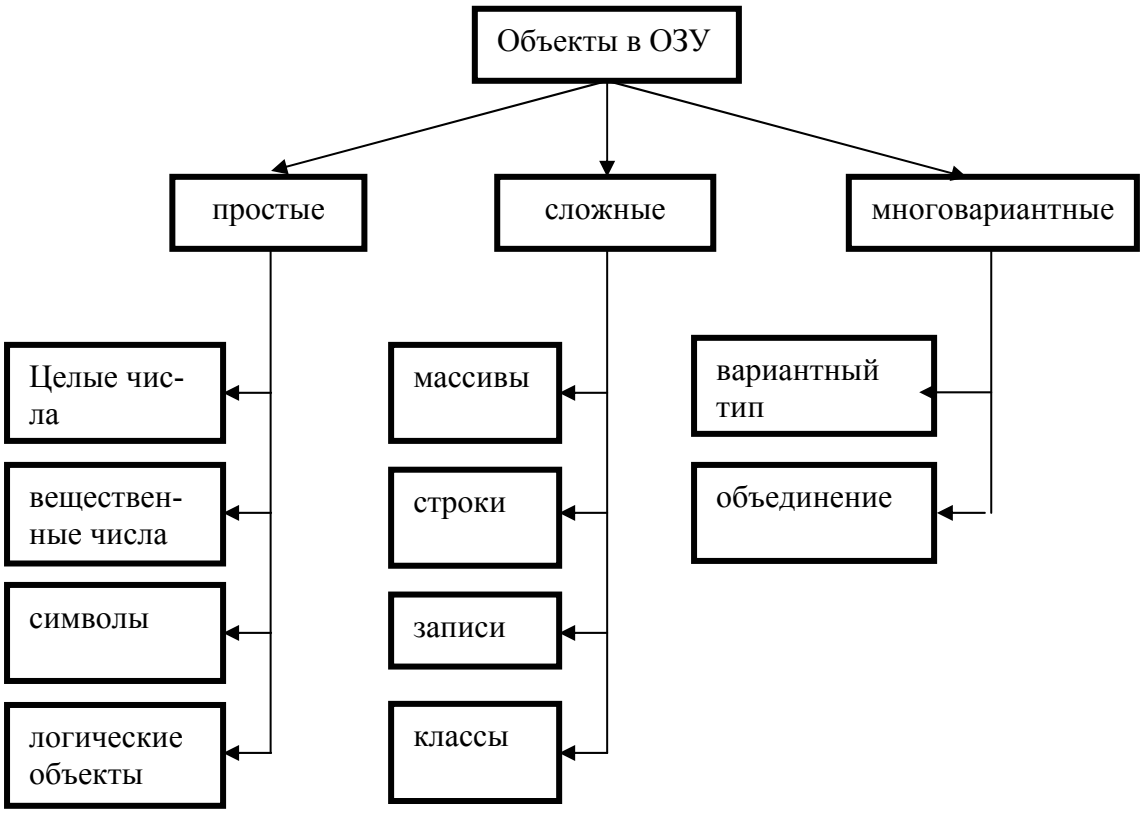


Рис. 2.1

2.2 Характеристики объектов в оперативной памяти

Исторически сложилось так, что в персональных ЭВМ логическая нумерация байтов на заданном участке памяти проводится справа налево. Например, если некоторый объект занимает в памяти байты с физическими адресами 15, 16, 17 и 18,

то логическое представление этого объекта будет записываться как содержимое байтов 18, 17, 16 и 15.

Все приведенные здесь схемы расположения полей в числовых форматах являются **логическими**, поэтому реально крайние левые байты этих форматов хранятся на старших адресах физического участка памяти. Далее везде мы будем записывать внутренние форматы только в логическом представлении.

2.2.1 Целые числа

Целые числа хранятся в виде двоичного представления в формате с фиксированной точкой. Знак числа обычно хранится в старшем (левом) разряде представления; для кодирования знака «плюс» используется 0, для знака «минус» - 1. Например, +10 и -15 в двоичном виде можно представить так:

Таблица 2.2

Число	Знаковый бит	Модуль числа
+10	0	0001010
-15	1	0001111

Для упрощения реализации вычислений в современных ЭВМ отрицательные целые числа представляются дополнительным двоичным кодом, неотрицательные – прямым двоичным кодом.

Прямой код. Получается путем непосредственного преобразования значения целого числа в двоичную систему счисления. Знак хранится в старшем разряде представления, остальные разряды содержат модуль числа в двоичном коде. Например, –5 в прямом коде равно 1000 0101. Данный код предельно прост, но неудобен для выполнения арифметических операций в ЭВМ. Для того, чтобы свести операцию вычитания к операции сложения используют обратный и дополнительный код.

Обратный код. Получается из прямого путем инвертирования каждого разряда двоичного представления модуля числа. Например, –5 в обратном коде будет равно 1111 1010. Обратите внимание: знаковый разряд в инвертировании не участвует.

Дополнительный код. Получается из обратного кода путем прибавления единицы. Данный код был предложен для удобства выполнения арифметических операций процессором. Число –5 в дополнительном коде будет равно 1111 1011.

Дополнительный код отрицательного числа можно получить другими способами:

а) перевести в двоичную систему положительное число, по модулю на единицу меньшее заданного, а затем полученное представление инвертировать;

б) перевести в двоичную систему положительное число, равное $(2^n – N)$, где n – количество разрядов числа в дополнительном коде, N – модуль числа, подвергаемого переводу. Например, дополнительный код числа –10 в однобайтовом представлении (n = 8) , может найти путем двоичного представления числа:

$$2^8 – 10 = 256 – 10 = 246 = 1111\ 0110_2$$

Основная информация по целочисленным форматам данных приведена в таблице 2.3, схема расположения битов приведена на рис. 2.2. Под «системой» в таблице 2.3 понимается вид системы программирования. Первые системы были 16-разрядными, т.к. были ориентированы на создание программ для 16-разрядных процессоров. К ним относятся, например, системы Турбо Паскаль и Турбо Си различных версий. Современные системы программирования (DELPHI, C++ Builder и др.) предназначены для создания программ для 32-разрядных процессоров.

Перевод чисел из внутреннего представления во внешнее проводится следующим образом:

- определяется значение знакового разряда;
- если знаковый разряд содержит 1, то число является отрицательным и его модуль необходимо перевести из дополнительного кода в прямой, а затем из прямого кода в десятичную систему счисления;
- если знаковый разряд содержит 0, то это положительное число в прямом коде, поэтому его сразу можно перевести в десятичное представление.

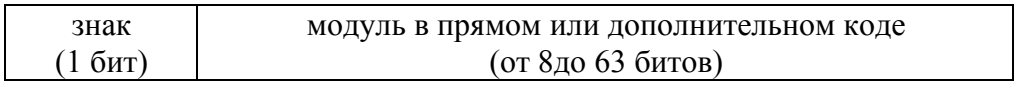


Рис. 2.2. Формат хранения целых чисел

Таблица 2.3

Система	Обозначение типа	Диапазон значений	Длина (байт)
16-разрядная	ShortInteger	-128 …127	1
	Integer	-32768 …32767	2
	LongInteger	-2147483648 …2147483647	4
	Byte	0 …255	1
	Word	0 …65535	2
32-разрядная	Integer	-2147483648 …2147483647	4
	Cardinal	0 …4294967295	4
	ShortInt	-128 …127	1
	SmallInt	-32768 …32767	2
	LongInt	-2147483648 …2147483647	4
	Int64	-2^{63} …$2^{63} - 1$	8
	Byte	0 …255	1
	Word	0 …65535	2
	LongWord	0 …4294967295	4

2.2.2 Вещественные числа

Для хранения вещественных чисел используется формат с плавающей точкой, основанный на экспоненциальной форме представления числа в виде нормализованной мантиссы (m) и порядка (k): $y = m * p^k$ Мантисса двоичного числа считается нормализованной, если ее значение находится в диапазоне $1 \le m < p$, где p – основание системы счисления.

В общем случае участок памяти, предназначенный для хранения вещественного числа, должен содержать четыре поля: знак мантиссы, значение мантиссы, знак порядка и значение порядка. Для экономии памяти в современных ЭВМ используются следующие приемы:

1. вместо порядка, который может быть как положительным, так и отрицательным, хранится экспонента e = k + const, которая всегда является целым беззнаковым числом; для этого значение константы всегда выбирается определенным образом;
2. в двоичной системе счисления целая часть нормализованной мантиссы всегда равна 1, поэтому достаточно хранить только дробную часть мантиссы.

Поэтому современные ЭВМ для хранения вещественных чисел в основном используют три поля: знак мантиссы (s), дробная часть мантиссы (f) и экспонента (e).

Существует несколько внутренних вещественных форматов, основные характеристики которых приведены в таблице 2.4 и на рис. 2.3 – рис. 2.8.

Обратите внимание:

- формат Extended хранит нормализованную мантиссу полностью (целая часть хранится в третьем поле, дробная – в четвертом).
- формат Comp является целочисленным знаковым форматом. Особенностью его является то, что он не относится к перечисляемым типам данных, а при вводе в него вещественного числа дробная часть просто отбрасывается.
- формат Real48 в 32-разрядных системах программирования используется для совместимости с форматом Real в 16-разрядных программах.

Таблица 2.4

Система	**Обозначение типа**	**Диапазон значений**	**Длина (байт)**	**Константа**	**Поля**
16-разрядная	Single	$1{,}5*10^{-45}$ …$3{,}4*10^{38}$	4	127	sef
	Real	$2{,}9*10^{-39}$ …$1{,}7*10^{38}$	6	129	sfe
	Double	$5*10^{-324}$ …$1{,}7*10^{308}$	8	1023	sef
	Extended	$3{,}4*10^{-4932}$ …$1{,}1*10^{4932}$	10	16383	sef
	Comp	$-2^{63}+1$…$2^{63}-1$	8	-	sd
32-разрядная	Single	$1{,}5*10^{-45}$ …$3{,}4*10^{38}$	4	127	sef
	Real48	$2{,}9*10^{-39}$ …$1{,}7*10^{38}$	6	129	sfe
	Real	$5*10^{-324}$ …$1{,}7*10^{308}$	8	1023	sef
	Double	$5*10^{-324}$ …$1{,}7*10^{308}$	8	1023	sef
	Extended	$3{,}6*10^{-4951}$ …$1{,}1*10^{4932}$	10	16383	sef
	Comp	$-2^{63}+1$…$2^{63}-1$	8	-	sd

Рис. 2.3 Формат Real и Real48 (6 байтов)

Рис 2.4. Формат Single (4 байта)

Рис 2.5. Формат Double (8 байтов)

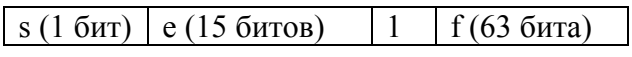


Рис. 2.6. Формат Extended (10 байтов)

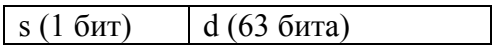


Рис. 2.7. Формат Comp (8 байтов)

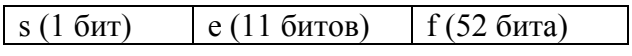


Рис. 2.8. Формат Real (8 байтов)

Значения всех вещественных чисел перед представлением во внутреннем формате предварительно представляются в двоичном виде

+/- мантисса $\cdot 2^{\text{порядок}}$,

где мантисса – двоичное вещественное нормализованное число, т.е. приведена к форме $1,f$ (имеет один бит слева от запятой, в котором должна быть единица, $1 \le$ мантисса < 2); порядок k – целое число со знаком.

Во внутреннем формате вещественные типы (Real, Single, Double, Extended) хранят двоичное представление знака s ($s = 0$ означает "+", $s = 1$ — "–"), экспоненты $e = k + const$ и дробной части мантиссы f (без первой единицы, кроме формата Extended). Положительное целое число *const* для каждого типа имеет заданное значение и служит для того, чтобы для любых допустимых k значение e было неотрицательным.

Внешнее представление числа определяется из внутреннего формата следующим образом:

$$(-1)^s \cdot (1,f)_2 \cdot 2^{e-const}.$$

При этом (за исключением Extended), если $e = 0$, то и значение принимается нулевым.

Вещественные числа в памяти всегда представлены с ограниченной точностью, которая определяется числом разрядов, выделенных для хранения дробной части мантиссы. Мерой точности является погрешность, которая может быть абсолютной и относительной. Абсолютная погрешность равна

$\Delta = y_{вн} - y_0$,

где $y_{вн}$ – значение числа, представленного во внутреннем формате; y_0 - истинное значение числа.

Относительная погрешность определяется следующим образом:

$\delta = \Delta / y_0 * 100$ (%).

Внутреннее представление вещественного числа в заданном формате формируется следующим образом:

а) число представляется в двоичном коде и нормализуется. При этом для чисел, больших единицы, плавающая точка переносится влево, определяя положительный порядок. Для чисел, меньших единицы, точка переносится вправо, определяя отрицательный порядок.

б) из таблицы 2.4 с учетом заданного формата определяется экспонента $e = k + const$.

в) определяется значение знакового разряда.

г) в отведенное в памяти поле в соответствии с типом числа записываются знак мантиссы, мантисса и экспонента.

Внешнее представление вещественного числа формируется следующим образом:

а) внутреннее представление разбивается на поля (знаковое, мантисса, экспонента);

б) определяется значение знакового разряда s;

в) определяется значение порядка (k = e - const);

г) записывается значение числа в двоичном коде

$$(-1)^s \cdot (1, f)_2 \cdot 2^k$$

д) запятая в полученном двоичном числе передвигается на k разрядов вправо (при k >0) или влево (при k <0);

е) полученное число переводится в десятичную систему счисления.

В таблице 2.5 приведены правила вычислений с вещественными числами.

Таблица 2. 5

a	**b**	**Умножение (a * b)**	**Деление (a / b)**	**Сложение (a + b)**
$m_1 * p^{k1}$	$m_2 * p^{k2}$	$m_1 * m_2 * p^{(k1+k2)}$	$(m_1 / m_2) * p^{(k1-k2)}$	$m_3 * p^{k3}$ где $k_3 = \max\{ k_1, k_2\}$, m_3 - значение мантиссы, приведенное к наибольшему порядку

2.2.3 Граничные значения числовых данных

Граничными значениями числовых данных называются значения, находящиеся на границе возможного диапазона представления определенного типа данных. В таблице 2.6 приведены граничные значения для целочисленных знаковых типов данных (на примере типа Integer), а в таблице 2.7 – для вещественных типов данных (на примере типа Single) [3]. Здесь x – любое число (0 или 1)

Таблица 2.6

№	**Значение диапазона**	**Внутренний формат**
1	нуль	00000000 00000000
2	наименьшее положительное число	00000000 00000001

3	наибольшее положительное число	01111111 11111111
4	наименьшее отрицательное число	10000000 00000001
5	наибольшее отрицательное число	11111111 11111111
6	неопределенность	10000000 00000000

Таблица 2.7

№	Значение диапазона	Внутренний формат (sef)
1	положительный нуль	0 00000000 000000….00
2	отрицательный нуль	1 00000000 000000….00
3	наименьшее положительное число	0 00000001 000000….00
4	наибольшее положительное число	0 11111110 111111….11
5	наименьшее отрицательное число	1 11111110 111111….11
6	наибольшее отрицательное число	1 00000001 000000….00
7	положительная бесконечность	0 11111111 000000….00
8	отрицательная бесконечность	1 11111111 000000….00
9	нечисло	1 11111111 xxxxxx….xx
10	неопределенность	1 11111111 100000….00

2.2.4 Символы

Процессор компьютера может выполнять только действия над числами, поэтому символы в памяти ЭВМ хранятся в виде числовых кодов. Каждый символ кодируется определенным целым беззнаковым числом. Множество упорядоченных пар символов и их кодов образуют кодировочную таблицу.

Каждая клавиша на клавиатуре имеет свой код сканирования, который при нажатии клавиши поступает в буфер клавиатуры. Микропроцессор блока клавиатуры при нажатии и отпускании клавиши формирует сигнал прерывания. По этому сигналу центральный процессор компьютера переключается на выполнение служебных процедур базовой системы ввода/вывода (BIOS) операционной системы, обслуживающих обработку прерываний от клавиатуры. Если процессор не успевает обработать коды клавиш, то эти коды запоминаются в буфере клавиатуры и по заполнении этого буфера раздается предупреждающий звуковой сигнал.

Система BIOS считывает код сканирования и преобразует его в символьный код нажатой клавиши или в расширенный код , отслеживая при этом нажатие клавиш смещения: Alt, Ctrl, Shift. Символьный код соответствует нажатию символьных клавиш, а расширенный код – нажатию специальных клавиш (Home, End, Tab, F1 – F12 и др.). Расширенный код является двухбайтовым: первый байт - нулевой,

второй байт содержит код. Значения второго байта расширенного кода обычно лежат в диапазоне 0-132.

Для формирования символьного кода могут использоваться различные кодировочные таблицы: ASCII , ANSI, UNICODE, KOI-8 и т.д. Основные характеристики некоторых таблиц приведены в таблице 2.8. Таблица ASCII является в настоящее время одной из самых применяемых, причем первая половина таблицы (коды 0…127) является стандартной для всех видов ЭВМ во всех странах мира и используется для хранения управляющих символов, цифр, латинских букв и знаков пунктуации, а вторая половина таблицы (коды 128…255) используется для хранения символов национальных алфавитов, символов псевдографики (для рисования рамок и таблиц) и научной графики.

Таблица 2.8

Основные характеристики кодировочных таблиц	ASCII	ANSI	UNICODE
Объем памяти для хранения одного символа, байт	1	1	2
Максимальное число символов	256	256	65536
Коды символов кириллицы	128 – 175, 224 - 239	192-255	1040 - 1103
Номер кодировочной таблицы для России	866	1251	отсутствует

Существует два способа ввода символов с клавиатуры: прямой и альтернативный. Прямой ввод проводится путем нажатия клавиши, закрепленной за данным символом. Альтернативный ввод может использоваться для ввода любых символов, в том числе и тех, которые отсутствуют на алфавитно-цифровой клавиатуре. Для этого необходимо при нажатой клавише ALT вводить числовое десятичное значение кода ASCII на дополнительной цифровой клавиатуре; после освобождения клавиши ALT на экране будет сгенерирован символ, соответствующий введенному значению кода.

Существенным недостатком однобайтовых кодовых таблиц (ASCII и ANSI) является относительно небольшое количество символов. Для расширения диапазона используемых символов была предложена новая кодовая таблица UNICODE, основной отличительной особенностью которой является выделение для каждого

символа 2 байт. Таким образом, максимальное количество символов, с которыми можно работать, увеличивается до 65536. Таблица UNICODE используется операционными системами семейства Windows. Шрифты Windows, базирующиеся на таблице UNICODE, содержат как символы латинского алфавита, так и многих других алфавитов (арабский , греческий, русский, немецкий и т.д.).

Просмотреть шрифты можно с помощью программы «Таблица символов», входящей в комплект поставки Windows. При выборе любого символа из таблицы в нижней строке окна программы будет выведен код символа. Обратите внимание: некоторые шрифты кодируются однобайтовыми кодировками ANSI (например, Courier, Fixedsys, MS Sans Serif). Эту программу можно также использовать для альтернативного ввода символов.

Описания символьных типов данных приведены в таблице 2.9.

Таблица 2.9

Тип данных	Размер (байт)	Кодировка
Char	1	ASCII
AnsiChar	1	ANSI
WideChar	2	UNICODE

2.2.5 Логические объекты

Логические объекты могут принимать только два значения – ЛОЖЬ или ИСТИНА (0 или 1, False или True). Для хранения такого объекта достаточно одного бита, но реально в памяти выделяется 1 байт, как минимально адресуемая единица.

В языках программирования для описания логических объектов используется тип Boolean.

2.2.6 Массивы

Массив – это одномерная или многомерная совокупность фиксированного числа однородных элементов. Элементы массива хранятся в памяти последовательно; если массив многомерный, то хранение элементов чаще всего проводится по строкам.

Каждый элемент массива имеет свой номер (индекс), представляемый целым числом, начальный индекс обычно может быть произвольным. Поиск элементов в

массиве проводится по индексу и размеру элемента. Например, для одномерного массива адрес i - го элемента определяется следующим образом:

$$A[i] = A_0 + S * (i - i_{нач}),$$

где S – размер одного элемента массива (байт), $i_{нач}$- начальный индекс массива, A_0 - адрес начального элемента массива в памяти.

Описание массива в алгоритмических языках обычно проводится с помощью ключевого слова ***array***.

Пример. Для одномерного массива, состоящего из пяти целых чисел и описанного следующим образом :

var mas: array[1..5] of integer

адрес четвертого элемента будет равен $A[4] = A_0 + 2*(4 - 1) = A_0 + 6$ *(байт).*

2.2.7 Символьные строки

Строки представляются в памяти ЭВМ в виде последовательности символов переменной длины, которые, по сути, являются одномерными массивами символов. Доступ к отдельным символам строки может проводиться по их номеру (индексу), как к элементам массива, или с использованием специальных функций, предназначенных для обработки строк.

Существуют два способа хранения строк: *статический* и *динамический*. Первый способ основан на том, что размер участка ОЗУ для хранения строки задается программистом во время написания программы. Второй способ предполагает выделение участка памяти на этапе выполнения программы, когда строковая переменная получает свое значение.

2.2.7.1 Статические строки

Для реализации статического способа в языке Паскаль используется тип данных String. Размер участка памяти в этом случае может задаваться в явном или неявном виде, причем явный вид использует синтаксис описания массива. Например, String[10] предполагает, что размер строки не будет превышать 10 символов. Если число символов превышает максимально возможное значение, то остальные символы отбрасываются. Если размер строки не задается, то для хранения строки вы-

деляется 255 байтов. Память для статических строк выделяется на этапе компиляции программы.

Информация о реальном размере строки содержится в начальном (нулевом) байте строки, в котором хранится символ, код которого равен текущей длине строки. Таким образом, физический размер участка памяти для хранения строки всегда превышает максимальную длину строки на один байт.

Например, для переменной типа String[7], имеющей значение «Зима», физический размер будет равен 8 байт, а содержимое этих байтов будет следующим: 04 87 A8 AC A0 00 00 00 (в кодировке ASCII). Если эта переменная описана как String, то ее физический размер будет равен 256 байт, из которых первые 5 байтов полностью повторяют предыдущий пример, а остальные байты будут содержать значение 00_{16}.

2.2.7.2 Динамические строки

Динамические способ основан на том, что объем памяти, выделяемой для хранения строки, всегда равен текущему размеру строки. Для распознавания конца строки после последнего ее символа записывается символ, код которого равен нулю. В языке Паскаль такие строки называются ASCIIZ-строками, которые хранятся в виде массивов, а число элементов массива не может превышать 65535.

Для описания динамических строк в Паскале используется тип PChar. При этом реально в программе формируется указатель на строку, который занимает 4 байта и является фактически адресом этой строки в ОЗУ. Например, для переменной типа PChar, которой присвоено значение «Зима», будет выделено 5 байтов, имеющих следующие значения: 87 A8 AC A0 00.

2.2.7.3 Строковые типы Object Pascal

Язык Object Pascal, являющийся базовым языком интегрированной 32-разрядной системы программирования DELPHI, имеет расширенные возможности по хранению строковых данных по сравнению с предыдущей версией ТурбоПаскаль 7.0. Это связано с тем, что с помощью Object Pascal создаются приложения, предназначенные для работы под управлением старших версий Windows, работающих на компьютерах с размером машинного слова 4 байта.

В таблице 2.10 приведены строковые типы данных Object Pascal.

Таблица 2.10

Виды данных	Тип данных	Способ выделения памяти	Максим. число символов	Кодировка символов
Стандартные	String	статический	255	ANSI (1 байт)
	PChar	динамический	65535	ANSI (1 байт)
Расширенные	ShortString	статический	255	ANSI (1 байт)
	AnsiString	динамический	$\sim 2^{31}$	ANSI (1 байт)
	WideString	динамический	$\sim 2^{30}$	UNICODE (2 байта)

При разработке новых приложений рекомендуется использовать только расширенные типы данных. Основными являются AnsiString и WideString, тип ShortString предназначен для совместимости со старыми 16-разрядными версиями программ.

Способ выделения памяти для переменной типа String зависит от значения директивы компилятора: если включен режим {H+}, то такая переменная интерпретируется как AnsiString; если режим отключен {H-}, то переменная считается ShortString. По умолчанию установлено значение {H+}.

В выражениях можно использовать строковые переменные различных типов, т.к. компилятор автоматически будет выполнять все преобразования, связанные с различными способами хранения этих строк. При вызове процедур или функций для обработки строковых переменных таких преобразований не проводится, поэтому необходимо учитывать тип строки.

2.2.7.4 Основные операции над строками

В языке Паскаль предусмотрены две операции над переменными строкового типа: конкатенация (сложение строк) и сравнение. Конкатенация используется для объединения строк нескольких строк в одну, причем результат зависит от типа переменной, в которую записывается этот результат. Сравнение строк осуществляется слева направо в соответствии с кодами символов, входящих в эти строки. Символ с меньшим значением кода считается меньше символа с большим значением кода, например ‘D’ < ’F’.

Для обработки строк имеется также большое число различных процедур и функций, сведения о некоторых из них приведены в таблице 2.11.

Таблица 2.11

Имя процедуры или функции	Назначение	Синтаксис
Concat	Функция возвращает строку, являющуюся объединением нескольких строк, например s1 и s2.	Concat(s1,s2: string): string; Возможен также следующий вариант: s3:=s1 + s2;
Copy	Функция выделяет из строки (s1) заданное число символов (m), начиная с заданной позиции (n).	Copy(s1: string; n: integer;m: integer): string;
Delete	Процедура удаляет из строки (s1) заданное числа символов (m), начиная с заданной позиции (n).	Delete(var s1:string; n: integer; m: integer);
Insert	Процедура вставляет подстроку (s1) в строку (s2) начиная с заданной позиции (m)	Insert(s1: string; var s2: string; m: integer);
Pos	Функция проводит поиск подстроки. Если подстрока найдена, то возвращает номер позиции первого символа подстроки s1 в строке s2; иначе возвращает 0.	Pos(s1, s2: string): Byte;
Str	Процедура преобразует вещественное или целое число x в символьную строку s	Str(x; var s: string);
Val	Процедура преобразует символьное представление числа (строка s) в числовую переменную целого или вещественного типа (v). Переменная code равна 0, если преобразование выполнено правильно.	Val(s: string; var v; code: integer);
UpCase	Функция преобразует строчный символ в прописной	UpCase(ch: char): char;
Ord	Функция возвращает код символа (его порядковый номер в кодировочной таблице)	Ord(x: char): longint;
Chr	Функция возвращает символ по заданному коду	Chr(x: byte): char;

2.2.8 Записи

Запись – это совокупность фиксированного числа полей, каждое из которых имеет собственный тип. Этот тип данных используется для хранения объектов, представленных набором атрибутов. Например, запись об объекте «Студент» представляется совокупностью таких атрибутов, как ФАМИЛИЯ, ГОД РОЖДЕНИЯ, АДРЕС, СРЕДНИЙ БАЛЛ.

Доступ к полям записи проводится на основе следующей информации:

- последовательность полей в записи;
- размер каждого поля.

В языке Паскаль для описания записи используется ключевое слово Record, в языке С – слово Structure. При описании записи обязательно указывается тип каждого поля. Обращение к полю записи проводится по имени переменной и имени поля

Пример:

type Student = Record

```
      FIO: string[30];
      GOD: word;
      ADRES: string[30];
      BALL:single;
     end;
var Stud: Student;

 // тело программы

 stud.ball := 4.58;
```

2.2.9 Классы

Класс – это объект, расширяющий возможности записей, и представляющий собой совокупность фиксированного числа полей, каждое из которых имеет собственный тип, и процедур (функций), предназначенных для обработки содержимого полей. Поля предназначены для хранения данных, а процедуры (функции) называются методами или свойствами класса.

Например, для записи «Студент», рассмотренной выше, мы можем хранить такие процедуры, как

ОЧИСТИТЬ_ВСЕ_ПОЛЯ – для очистки всех полей данных в текущей записи;

ИЗМЕНИТЬ_АДРЕС(Новый адрес) – для замены старого адреса на новый в текущей записи;

КОПИРОВАТЬ(Новая ФИО) – для копирования текущей записи в новую запись, у которой поле ФИО будет иметь значение Новая ФИО.

Обращение к полям и методам класса проводится с указанием имени переменной и имени поля или метода.

Пример.

```
Type Student = class(TObject)
      FIO: string[30];
      GOD: word;
      ADRES: string[30];
      BALL:single;

    Procedure ClearAll;
    Procedure ChangeAdres(NewAdres);
    Procedure CopyRecord(NewFIO);
  end;
 var MyStud: Student;

// тело программы
```

MyStud.CopyRecord('Петров А.П.')

2.2.10 Многовариантные объекты

Многовариантный объект – это объект, который может в любой момент времени хранить данные любого типа (целые или вещественные числа, символы, строки и т.д.). Этот вид объектов используется в том случае, когда тип того или иного объекта заранее не известен или когда какие-то функции и процедуры требуют именно такой тип аргументов.

Возможны два типа многовариантных объектов – вариантный тип и объединение. Вариантный тип занимает в памяти 16 байтов и содержат код текущего типа (2 байта) и значение переменной или указатель на это значение. Для описания переменных вариантного типа используется ключевое слово **variant**. В переменных этого типа могут храниться данные любых типов, кроме записей, массивов, файлов, классов, ссылок на классы и указателей. Кроме того, такие переменные могут хранить указатели на объекты COM и CORBA, обеспечивая доступ к их методам и свойствам. Примерами алгоритмических языков, поддерживающих переменные вариантного типа, являются Object Pascal и Visual Basic.

Недостатками использования вариантного типа являются увеличенные затраты памяти и времени на выполнение операций. Кроме того, недопустимые операции с переменными типа **variant** приводят к ошибкам времени выполнения, тогда как аналогичные недопустимые операции с переменными других типов выявляются на этапе компиляции.

Объединение – это объект, который может хранить данные только заранее описанных типов. Размер участка памяти при этом определяется размером самого громоздкого используемого типа, в любой момент времени в этом участке хранится переменная только одного типа. Этот вид объектов используется, например, в языке Си, где для описания ключевое слово **union**.

Пример описания объединения, предназначенного для хранения трех типов:

```
union stud {
      int digit;
      double chislo;
      char symbol; };
```

Далее в программе определяется переменная и проводится работа с ней:

```
union stud data;

data.digit=50;  data.symbol='S';   data.double=10.5;
```

Здесь компилятор выделяет участок памяти размером 8 байтов.

2.3 Доступ к объектам в памяти

Возможны два способа доступа к объектам, хранящимся в оперативной памяти ЭВМ: по имени (идентификатору) и по адресу (указателю). Первый способ можно использовать, например, следующим образом:

```
beta := alfa + brok[k] + student.ball;
```

Здесь переменной beta присваивается значение, равное сумме значений переменной alfa, k – го элемента массива brok и поля записи student.ball.

Второй способ предусматривает использование указателей на переменные. Для описания переменной типа указатель в Паскале используется символ '^', а в языке Си – символ '*'; адрес переменной определяется в Паскале с помощью операции @, а в Си – с помощью операции &. Например, в Паскале для определения адреса переменной prim можно использовать @prim, а в СИ - &prim.

Пример. Определим результат работы следующей программы:

```
Var B: byte;
     P1: ^byte;
begin
   B :=1;
   P1:= @B;
   Writeln('Значение переменной P1 равно: ', P1);
end.
```

На экране будет выведена строка: Значение переменной P1 равно:1

Доступ к объектам со стороны программ зависит от области видимости объектов. С этой точки зрения существуют локальные и глобальные объекты. Локальные переменные видны только внутри программ, в которых они описаны. Имена локальных переменных могут совпадать в нескольких программах, однако физически это будут разные объекты.

Глобальные переменные описываются в главном модуле программы и доступны в любой другом ее модуле как для чтения, так и для записи..

3. Хранение данных в внешней памяти

Классификация ВЗУ. Физическая модель диска. Дорожка, сектор, байт, физический адрес на диске, физическая запись. Форматирование дисков, классификация способов форматирования. Иерархическая схема хранения информации во внешней памяти. Файл. Основные соглашения об идентификации объектов управления: устройство, диск, каталог, маршрут, имя файла. Спецификация файлов в различных операционных системах. Логическая модель диска. Структура системной области и области данных. Кластер. Особенности файловых систем FAT и NTFS.

Внешние запоминающие устройства (ВЗУ) являются энергонезависимыми устройствами и предназначены для долговременного хранения информации. Основные отличительные признаки этих устройств – большой объем памяти и большое время доступа к информации. В современных ЭВМ в качестве ВЗУ используются устройства с последовательным или с прямым доступом.

Устройством с последовательным доступом является стример, у которого в качестве носителя информации используется магнитная лента. Стример обычно используется для резервного хранения информации.

Наиболее часто в качестве ВЗУ используются магнитные диски (жесткие или гибкие), оптические диски, магнитооптические диски и флэш - накопители.

3.1 Файлы и каталоги

На ВЗУ существует только два вида информационных объектов - файлы и каталоги. *Файл* - это поименованная совокупность информации, хранящаяся на ВЗУ. В виде файлов на диске хранятся программы и данные. *Каталог* – поименованная область диска, предназначенная для хранения метаданных о зарегистрированных в этом каталоге файлах и подкаталогах (имя, расширение, атрибуты, размер, дата и время создания или последнего изменения, адрес). В одном каталоге нельзя зарегистрировать файлы с одинаковыми именами, но один и тот же файл может быть зарегистрирован в разных каталогах.

В MS DOS имя файла может содержать до восьми, а расширение – до трех символов. Расширения имен файлов не являются необходимыми компонентами, но

они используются многими программами для обозначения и распознавания типа файла. В системе Windows имя файла может содержать до 255 символов.

Имена файлов не должны совпадать с именами устройств, зарезервированными в системе (табл. 3.1).

Таблица 3.1

Имя	**Назначение**
AUX	асинхронный интерфейс
CLOCK$	драйвер часов
COM1, COM2, COM3, COM4	последовательные интерфейсы
CON	консоль (клавиатура или монитор)
LPT1, LPT2, LPT3	параллельные интерфейсы
NUL	фиктивное устройство
PRN	принтер

Спецификация файла (полное имя файла) - описание местонахождения файла, имеет вид:

имя_устройства \ путь_доступа \ имя_файла. расширение

При обращении к группе файлов в имени допускается использование символов подстановки «*» и «?». Символ «*****» означает заменяет любое количество любых символов, символ «**?**» заменяет любой одиночных символ.

Файлы могут быть двух видов: исполняемыми и неисполняемыми. К первой группе относятся файлы, содержащие программы, исполняемые центральным процессором (расширение COM или EXE) или операционной системой (расширение BAT или CMD). Неисполняемые файлы содержат данные (текстовые, графические, звук и т.д.), тип данных можно ориентировочно определить по расширению в спецификации файла.

Требования к имени каталога те же, что и к именам файлов. Как правило, расширение имени для каталога не используется. В операционной системе имеется несколько зарезервированных имен для каталогов:

. – имя текущего каталога;

.. – имя родительского каталога;

… - имя прародительского каталога (поддерживается только в WINDOWS).

Любой файл или каталог имеют набор свойств - атрибутов, каждый из которых может принимать два значения – включен или выключен. Наиболее часто исполь-

зуются следующие атрибуты: read only (только для чтения), hidden (скрытый), system (системный) и archive (измененный).

Примеры описания групп файлов:

. *- все файлы, без исключения;*

**.txt* *- файлы с любыми именами, но с расширением .txt;*

AB.*- файлы, имена которых начинаются с цепочки символов AB и имеющие любое расширение;*

YE??. - файлы, имена которых начинаются с цепочки символов YE, два следующих символа НЕ имеют значения, расширение НЕ имеет значение.*

3.2 Физическая модель магнитного диска

Информация на МД размещается вдоль концентрических окружностей, называемых *дорожками*. Дорожки с одинаковыми номерами на различных *поверхностях* диска образуют *цилиндр*. Каждая дорожка содержит определенное количество *секторов* – это участок дорожки МД, хранящий минимальную порцию информации, которая может быть считана или записана за одно обращение к диску.

Для магнитных дисков объем сектора обычно составляет 512 байт (0,5 Кбайт). Для обеспечения постоянства размера сектора при изменении номера дорожки изменяется плотность записи, максимальная плотность записи используется на дорожках с минимальным радиусом.

Для получения доступа к информации, хранящейся на диске, необходимо указать физический адрес, который включает номер поверхности, номер дорожки и номер сектора на этой дорожке.

Таким образом, *физическая модель* МД включает следующие понятия: поверхность, дорожка, цилиндр и сектор. Все диски отличаются друг от друга по набору этих параметров.

3.3 Логическая модель магнитного диска.

С логической точки зрения все адресное пространство диска представляет собой набор последовательно пронумерованных секторов. Небольшая часть этих секторов выделяется для хранения служебной информации и называется системной областью диска, остальные сектора предназначены для хранения файлов и каталогов и образуют область данных.

Единицей дисковой памяти для размещения файлов является *кластер*. В один кластер могут входить от 1 до 128 смежных секторов. Все кластеры имеют сквоз-

ную нумерацию, причем файлы на диске могут храниться в виде фрагментов в несмежных кластерах, т.е. в различных частях диска.

Структура системной области диска зависит от типа файловой системы диска, которая определяет способы доступа к информации в области данных. Примерами файловых систем являются FAT, NTFS (в операционных системах семейства Windows), ext2, ext3 (в операционных системах семейства Linux), HPFS (в операционной системе OS/2).

Адресное пространство жестких дисков может быть разбито на несколько разделов (логических дисков), поэтому для жестких дисков в логической модели добавляется таблица разделов диска, содержащая информацию о размещении логических дисков и о том, какой из логических дисков будет использоваться для загрузки ОС.

3.3.1 Файловая система FAT

На рис. 3.1 приведена логическая модель диска с файловой системой FAT.

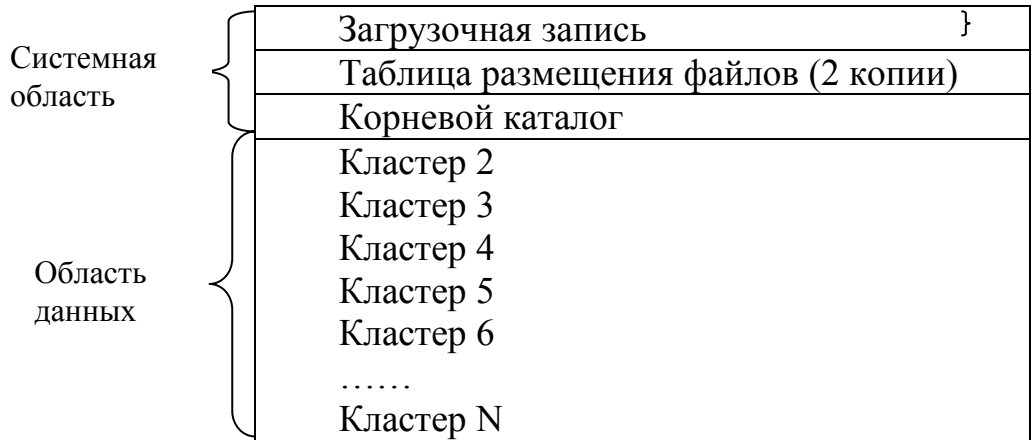


Рис. 3.1 Логическая структура диска

Загрузочная запись, иногда называемая начальным загрузчиком, имеет размер 512 байт, всегда хранится в нулевом секторе и используется в процессе загрузки операционной системы.

Таблица размещения файлов, в оригинальной литературе называемая FAT (File Allocation Table), содержит информацию о размещении файлов в области данных. Она всегда занимает сектора, начиная с первого. На любом диске всегда хранится две копии FAT для обеспечения надежного доступа к данным. Размер таблицы зависит от объема диска. Например, для гибкого магнитного диска 360 Кбайт

FAT занимает всего 4 сектора, а для жесткого диска объемом 8 Гбайт - 15799 секторов.

Таблица размещения файлов содержит информацию о номерах кластеров, выделенных для хранения каждого файла. Она представляет собой карту (образ) области данных, в которой описывается состояние каждого кластера диска. Каждый элемент таблицы соответствует одному кластеру в области данных. Если кластер свободен, то соответствующий ему элемент FAT имеет значение "0". Если кластер выделен для какого-либо файла, то возможны два варианта:

а) элемент содержит признак конца файла "EOF", если этот кластер является последним кластером, выделенным файлу;

б) элемент содержит значение номера следующего кластера, выделенного файлу.

Номер начального кластера, выделенного файлу, записывается в элемент каталога, в котором этот файл зарегистрирован. Таким образом элементы FAT, выделенные одному файлу, связываются в цепочки, позволяющие получить доступ к информации даже в том случае, если файл при записи разбивается на несколько фрагментов, хранящихся в несмежных кластерах диска.

Элементы FAT могут быть 12-ти, 16-ти и 32-х разрядными, в зависимости от объема диска, соответственно файловые системы называются FAT12, FAT16 или FAT32 . Для 12-ти разрядных элементов требуется полтора байта, поэтому в этом случае для уменьшения размеров FAT значения двух соседних элементов таблицы хранятся в трех байтах. FAT12 применяется на гибких магнитных дисках, FAT16 и FAT32 – на жестких дисках.

Корневой каталог – главный каталог диска, который занимает сектора, следующие за FAT. Количество секторов корневого каталога зависит от объема диска.

Записи корневого каталога имеют длину 32 байта, структура записей представлена в таблице 3.2

Таблица 3.2

Номер поля	**Длина (байт)**	**Назначение поля**
1	8	имя файла
2	3	расширение имени
3	1	атрибуты
4	10	резерв
5	2	время создания/модификации

6	2	дата создания/модификации
7	2	номер начального кластера
8	4	размер файла

Если файл не имеет расширения, то в соответствующем поле хранятся пробелы. Дата и время используются в виде четырехбайтового значения в операциях сравнения. Номер начального кластера определяет точку входа в FAT для данного файла и одновременно дисковый адрес собственно файла.

Байт атрибутов файла задает его статус в соответствии с таблицей 3.3. Если байт атрибута равен 8, то метка тома хранится в полях имени и расширения файла элемента корневого каталога. Если элемент каталога указывает на подкаталог, то используются все поля элемента каталога, при этом последнее поле (размер файла) имеет нулевое значение.

Таблица 3.3

Значение байта атрибутов	**Характеристика файла**
1	только чтение
2	скрытый файл
4	системный файл
8	элемент каталога хранит метку тома (11 байтов)
16	элемент каталога указывает на подкаталог

Файловая система Windows, способная хранить длинные имена (VFAT), для каждого файла и подкаталога хранит два имени – длинное и короткое. Хранение длинных имен файлов организуется в специально отформатированных записях каталога, у которых байт атрибутов содержит 0F. Экспериментальным путем было определено, что такие записи каталога не видны для программ DOS, работающих только с короткими именами. Структура записи каталога для хранения длинных имен приведена в таблице 3.4, откуда видно, что одна запись может хранить до 13 символов в кодировке Unicode.

Таблица 3.4

Номер поля	**Длина (байт)**	**Назначение поля**
1	1	порядок следования
2	10	первые 5 символов имени
3	1	Атрибуты (0F)
4	1	Указатель типа
5	1	Контрольная сумма
6	12	Следующие 6 символов имени
7	2	номер начального кластера (всегда 0)
8	4	последние 2 символа в имени

Для регистрации файла с длинным именем в каталоге выделяется необходимое количество специальных записей, а также одна стандартная запись для хранения короткого имени. Блок специальных записей всегда располагается в каталоге перед стандартной записью, поэтому если к каталогу обращается 16-разрядная DOS программа, то она будет видеть только короткое имя файла, а 32-разрядные Windows-приложения могут работать с длинными именами.

Короткое имя образуется из длинного следующим образом: оставляется 6 символов длинного имени и дописываются знак “~” (тильда) и порядковый номер в пределах каталога для файлов, у которых первые 6 символов имени совпадают.

Например, для регистрации файла с именем «Отчеты лаборатории за 3 квартал 2007 года», содержащим 41 символ, в каталоге будут выделены 4 специальные записи и одна стандартная, которая будет хранить имя “Отчеты~1”.

Каталоги нижнего уровня (подкаталоги) имеют структуру, аналогичную корневому каталогу, только в отличие от него они не имеют фиксированного размера и фиксированного дискового адреса, т.е. хранятся на диске в области данных как обычные файлы. Для того, чтобы в процессе работы существовала возможность перемещения по дереву каталогов как вниз, так и вверх, в структуре подкаталогов предусмотрено существование двух элементов каталога с именами "." и "..". Первый элемент каталога является указателем на данный каталог, а второй – на родительский каталог. Соответственно поле «Номер начального кластера» этих элементов содержит дисковые адреса этих каталогов и если это поле содержит нуль, то это означает, что каталог- родитель является корневым каталогом диска.

Логическое разбиение области данных на кластеры, как совокупность секторов, взамен использования одиночных секторов имеет следующий смысл :

- уменьшается размер FAT;
- уменьшается возможная фрагментация файлов;
- ускоряется доступ к файлу, т.к. в несколько раз сокращается длина цепочек фрагментов дискового пространства, выделенных для него.

Следует иметь ввиду, что при увеличении размера кластера ухудшается коэффициент использования дисковой памяти. Минимальный размер кластера на диске

с файловой системой FAT зависит об объема диска ($V_{диска}$) и разрядности элемента FAT (r):

$$V_{кл\_min} = V_{диска} / 2^{r}$$

3.3.1 Файловая система NTFS

Основным отличием системы NTFS от системы FAT является использование двух видов каталогов – базового и обычного. Каждый раздел диска имеет один базовый каталог, в котором регистрируются все файлы раздела, и который называется MFT (Master File Table - общая таблица файлов).

В записи MFT хранится вся информация о файле (имя, дата и время создания, размер, положение на диске отдельных фрагментов, и т.д). Если для информации не хватает одной записи MFT, то используются несколько, причем не обязательно подряд. При этом первая запись называется базовой.

Диск NTFS условно делится на две части. Первые 12% диска отводятся под так называемую MFT зону - пространство, в которое растет метафайл MFT. Запись каких-либо данных в эту область невозможна. MFT-зона всегда держится пустой - это делается для того, чтобы самый главный, служебный файл (MFT) не фрагментировался при своем росте. Остальные 88% диска представляют собой обычное пространство для хранения файлов. Кластер NTFS может иметь размер от 0,5 до 64 Кбайт, стандартное значение – 4 Кбайт.

Свободное место диска, однако, включает в себя всё физически свободное место - незаполненные куски MFT-зоны туда тоже включаются. Механизм использования MFT-зоны таков: когда файлы уже нельзя записывать в обычное пространство, MFT-зона просто сокращается (в текущих версиях операционных систем ровно в два раза), освобождая таким образом место для записи файлов. При освобождении места в области данных MFT зона может снова расшириться.

MFT представляет собой централизованный каталог всех остальных файлов диска, включая себя самого. MFT поделен на элементы фиксированного размера (обычно 1 Кбайт), каждая из которых соответствует какому либо файлу или каталогу. Каждый элемент имеет уникальный номер – индекс, общее количество эле-

ментов MFT – до 2^{48}. Первые 16 элементов описывают системные метафайлы, причем самый первый элемент описывает структуру самого MFT.

Эти первые 16 элементов MFT - единственная часть диска, имеющая фиксированное положение. Вторая копия первых трех элементов для надежности хранится ровно посередине диска. Остальной MFT-файл может располагаться, как и любой другой файл, в произвольных местах диска - восстановить его положение можно с помощью его самого, "зацепившись" за первый элемент MFT. На рис. 3.1 приведена начальная область MFT.

В NTFS определены 12 атрибутов, которые могут появляться в элементах MFT и перечень которых приведен в таблице 3.5

Для размещения файлов NTFS выделяет по возможности **серии последовательных кластеров** (**сегментов файлов**). Расположение сегментов описывается в элементе MFT в виде набора записей, каждая из которых содержит два 64-разрядных числа - номер начального кластера сегмента и количество кластеров в сегменте. Заголовок данных указывает общее количество блоков файла. Нефрагментированный файл будет иметь всего одну серию кластеров.

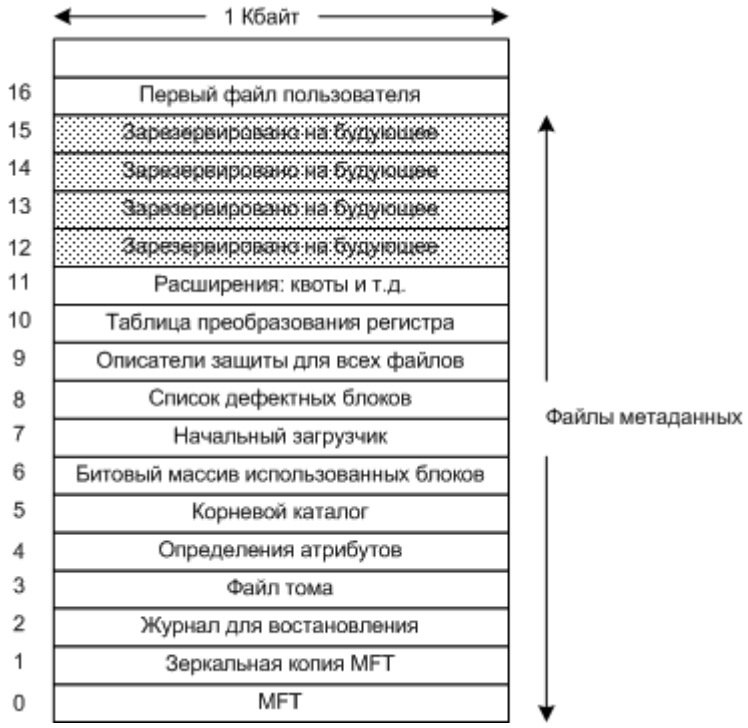


Рис. 3.2 Структура MFT

Таблица 3.5

Атрибут	Описание
Стандартная информация	флаговые биты (только чтение, архивный), временные штампы и т.д.
Имя файла	имя файла в кодировке Unicode, файлы могут повторяться в формате MS-DOS 8.3.
Список атрибутов	расположение дополнительных записей MFT
Идентификатор объекта	64-разрядный идентификатор файла, уникальный для данного раздела диска
Точка повторного анализа	используется для символьных ссылок и монтирования устройств.
Название тома	Название диска
Версия тома	Версия диска
Корневой индекс	используется для каталогов
Размещение индекса	используется для очень больших каталогов
Битовый массив	используется для очень больших каталогов
Поток данных утилиты регистрации	Управляет регистраций в $LogFile
Данные	поточные данные, может повторяться, используется для хранения информации о расположении файла на диске или для хранения данных файла (для небольших файлов)

Интересной особенностью NTFS является возможность размещения небольших по размеру файлов (несколько сотен байтов) внутри элемента MFT без выделения места в области данных, что позволяет существенно экономить дисковое пространство. В этом случае файл называется непосредственным.

На рис. 3.3 приведена структура записи MFT для файла, занимающего 9 кластеров и состоящего из трех сегментов: первый сегмент включает кластеры 20 -23, второй сегмент – кластеры 64, 65 и третий сегмент – кластеры 80 – 82.

Рис. 3.3.Элемент MFT для файла, состоящего из трех сегментов

Если файл сильно фрагментирован, то требуется несколько записей MFT (см. рис. 3.4)

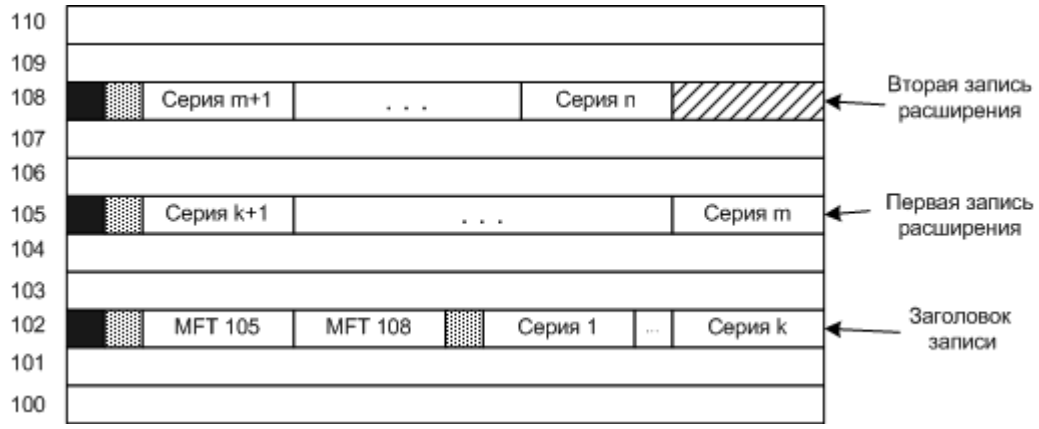


Рис. 3.4.Элемент MFT для сильно фрагментированного файла

Первые 16 файлов NTFS (метафайлы) носят служебный характер. Каждый из них отвечает за какой-либо аспект работы системы. Преимущество такого модульного подхода заключается в поразительной гибкости - например, в системе FAT физическое повреждение в самой области FAT фатально для функционирования всего диска, а NTFS может сместить или даже фрагментировать по диску все свои служебные области, обойдя любые неисправности поверхности - кроме первых 16 элементов MFT.

Метафайлы находятся корневом каталоге NTFS диска - они начинаются с символа имени "$", хотя получить какую-либо информацию о них стандартными средствами сложно. Ниже приведены используемые в данный момент метафайлы и их назначение.

Таблица 3.6

Имя файла	**Назначение**
$MFT	централизованный каталог всех файлов раздела
$MFTmirr	копия первых 16 записей MFT, размещенная посередине диска
$LogFile	файл поддержки журналирования
$Volume	служебная информация - метка тома, версия файловой системы, т.д
$AttrDef	список стандартных атрибутов файлов на томе
$.	корневой каталог
$Bitmap	карта свободного места тома
$Boot	загрузочный сектор (если раздел загрузочный)
$Quota	файл, в котором записаны права пользователей на использование дискового пространства
$Upcase	файл - таблица соответствия строчных и прописных букв в именах файлов на текущем томе; нужен потому, что в NTFS имена файлов записываются в кодировке Unicode, что составляет более 65 тысяч различных символов, искать большие и

малые эквиваленты которых достаточно сложно.

Каталог NTFS представляет собой специфический файл, хранящий ссылки на другие файлы и каталоги, создавая иерархическое строение данных на диске. Элемент MFT для небольшого каталога содержит несколько записей, каждая из которых описывает один файл или каталог, указывая при этом индекс соответствующего элемента MFT, длину имени и т.д. (рис. 3.5) .

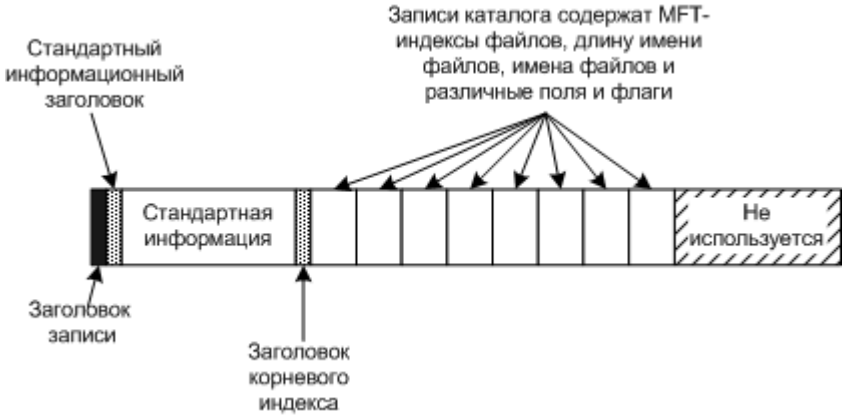


Рис. 3.5 Элемент MFT для небольшого каталога

Для больших каталогов вместо простого списка файлов используется бинарное дерево, обеспечивающее быстрый поиск файла. Вот что это означает: для поиска файла с заданным именем в линейном каталоге, таком, например, как у FAT, операционной системе приходится просматривать все элементы каталога до тех пор, пока она не найдет нужный. Бинарное же дерево располагает имена файлов таким образом, чтобы поиск файла осуществлялся более быстрым способом - с помощью получения двухзначных ответов на вопросы о положении файла.

Вопрос, на который бинарное дерево способно дать ответ, таков: в какой группе, относительно данного элемента, находится искомое имя - выше или ниже? Мы начинаем с такого вопроса к среднему элементу, и каждый ответ сужает зону поиска в среднем в два раза. Файлы, скажем, просто отсортированы по алфавиту, и ответ на вопрос осуществляется очевидным способом - сравнением начальных букв. Область поиска, суженная в два раза, начинает исследоваться аналогичным образом, начиная опять же со среднего элемента. Например, для поиска одного файла среди 1000 FAT придется осуществить в среднем 500 сравнений (наиболее

вероятно, что файл будет найден на середине поиска), а системе на основе дерева - всего около 12-ти ($2^{10} = 1024$).

Главный каталог диска - корневой - ничем не отличается об обычных каталогов, кроме специальной ссылки на него из начала метафайла MFT.

Для обеспечения повышенной надежности NTFS использует **журналирование** - ведение журнала транзакций, под которыми понимается последовательность действий, совершаемых целиком и корректно или не совершаемых вообще. У NTFS просто не бывает промежуточных (ошибочных или некорректных) состояний - квант изменения данных не может быть поделен на до и после сбоя, принося разрушения и путаницу - он либо совершен, либо отменен.

Важно понимать, что система восстановления NTFS гарантирует корректность только файловой системы, но не данных.

3.4 Подготовка дисков к работе

Процесс подготовки МД к работе называется *форматированием диска* и может включать следующие этапы:

1) низкоуровневое форматирование - формирование дорожек, разбиение дорожек на сектора, тестирование каждого сектора;

2) высокоуровневое форматирование - формирование и инициализация системной области (запись начального загрузчика, создание двух копий FAT и корневого каталога), запись метки диска, копирование файлов операционной системы.

Низкоуровневое форматирование может выполняться только для гибких дисков, высокоуровневое – для всех типов дисков. В терминах системы Windows низкоуровневое форматирование называется полным, а высокоуровневое – быстрым.

При форматировании гибкого диска в командном режиме пользователь может задавать количество дорожек и секторов на диске. Запись метки диска и копирование операционной системы не являются обязательными и выполняются только по дополнительному указанию пользователя, например при подготовке загрузочного диска.

Корневой каталог диска, в отличие от обычных каталогов, создается при форматировании диска и поэтому его размер всегда ограничен. Поэтому число файлов и подкаталогов, которые могут храниться в корневом каталоге, также огра-

ничено. Это следует учитывать особенно при использовании длинных имен файлов.

4. Основы технологии разработки программ.

Жизненный цикл программ. Этапы разработки программного обеспечения (ПО). Этапы подготовки программ к исполнению. Структура и основные функции интегрированной среды программирования DELPHI. Основные этапы разработки ПО и инструментальные средства для каждого из этапов. Программы, управляемые событиями. Интегрированные системы программирования: назначение, состав, интерфейс и основы функционирования. Консольное и графические приложения. Визуальные компоненты: методы, свойства, события. Ввод и редактирование исходного текста программы. Компиляция и способы построения загрузочного модуля. Отладка программы: средства временной остановки выполнения, средства для работы с окнами наблюдений. Настройка параметров интегрированной среды. Краткий обзор возможностей интегрированной системы Delphi. Классификация ошибок в программах, методы уменьшения вероятности возникновения ошибок. Стандартные управляющие структуры технологии структурного программирования .

4.1 Жизненный цикл программного обеспечения

Жизненный цикл программного обеспечения (ПО) – это период времени, который начинается с момента принятия решения о необходимости создания ПО и заканчивается в момент его полного изъятия из эксплуатации. Основной нормативный документ, регламентирующий состав процессов жизненного цикла ПО - международный стандарт ISO/IEC 12207:1995 “Information Technology – Software Life Cycle Processes” – определяет состав процессов, действий и задач, которые должны быть выполнены во время создания ПО. Российский аналог этого документа – ГОСТ Р ИСО/ МЭК 12207 – 99.

Существуют несколько моделей жизненного цикла:

модель черного ящика;

каскадная модель (водопадная);

итерационная модель (спиральная).

Модель черного ящика характеризуется полным отсутствием какого– либо технологического порядка, хотя юмор программистов выделяет в этой модели следующие стадии:

начало проекта;

безудержный энтузиазм;

разочарование;

хаос;

поиски виновных;

награждение непричастных;

определение требований к системе.

Каскадная модель (рис. 4.1) имеет следующие свойства:

переход на очередной этап проекта проводится только после завершения работ по предыдущему этапу без возврата на пройденные этапы;

требования к системе фиксируются до ее сдачи заказчику;

каждый этап завершается получением некоторых результатов, которые служат исходными данными для следующих этапов.

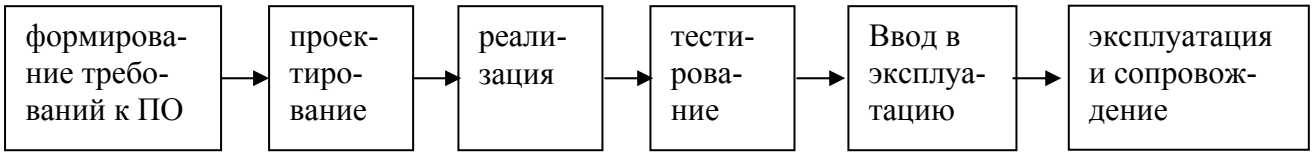


Рис. 4.1 Каскадная модель жизненного цикла ПО

Каскадная модель может использоваться при создании ПО, для которого в самом начале разработки можно точно и полно сформулировать все требования. Основными недостатками каскадной модели являются позднее обнаружение проблем, избыточное количество документации, высокий риск создания системы, не удовлетворяющей изменившимся требованиям пользователей.

В таблице 4.1 приведены характеристики этапов каскадной модели.

Таблица 4.1

№	Этап	Характеристики	Чем завершается
1	зарождение идеи и формулирование задачи	проводится формулирование и анализ требований к системе	техническое задание
2	проектирование	определяется архитектура системы, определяются структуры данных; разрабатываются алгоритмы обработки данных, выбираются инструментальные средства разработки, разрабатываются методики тестирования и т.д.	технический проект
3	реализация и тестирование	проводится кодирование программных модулей, их отладка и тестирование.	бета-версия системы, документация
4	опытная эксплуатация	система работает в режиме ограниченного использования под постоянным контролем	рабочая версия системы

		разработчиков	
5	функционирование	проводится основная работа пользователей с системой.	
6	старение	система морально устаревает.	система заменяется новой версией

Итерационная модель жизненного цикла ПО показана на рис. 4.2 и характеризуется следующими свойствами:

отказ от фиксации требований и назначение приоритетовпользовательским требованиям;

разработка последовательности прототипов системы, начиная с требований высшего приоритета;

анализ рисков на каждой итерации;

использование каскадной модели для реализации окончательного прототипа;

оценка результатов по завершении каждой итерации и планирование следующей итерации.

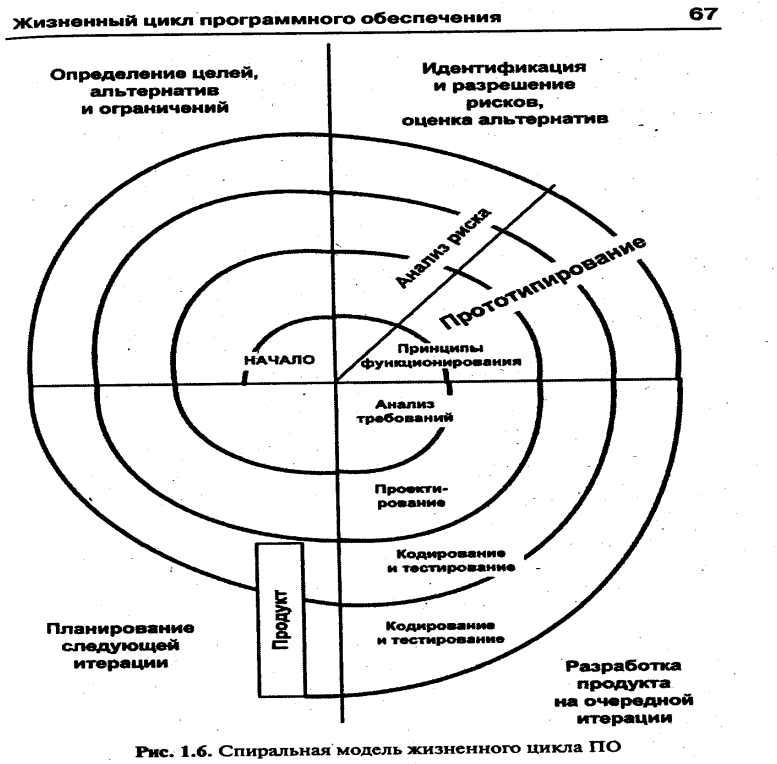


Рис. 1.6. Спиральная модель жизненного цикла ПО

Рис. 4.2 Итерационная модель жизненного цикла

4.2 Этапы подготовки программ к исполнению

Основными этапами подготовки программ к исполнению являются трансляция и организация межпрограммных связей (компоновка).

Целью трансляции является перевод программ, написанных на алгоритмическом языке (исходных модулей) в программы, представляющие собой набор ма-

шинных команд (объектные модули). По принципу действия трансляторы делятся на два вида: интерпретаторы и компиляторы. Интерпретатор переводит в последовательность машинных команд каждый оператор исходного модуля отдельно, причем эта последовательность команд сразу же выполняется. Достоинствами интерпретаторов являются удобство отладки программ и невысокие требования к объему ОЗУ, к недостаткам можно отнести худшее быстродействие по сравнению с компиляторами. Примерами интерпретаторов являются системы программирования Visual Basic и PHP.

Компилятор переводит в машинные команды весь исходный модуль и только после этого полученный объектный модуль передается компоновщику для дальнейшей обработки. Большинство современных систем программирования являются системами компилирующего типа (СИ, Паскаль, Пролог и др.)

Компилятор обрабатывает исходный модуль в несколько этапов:

а) препроцессорная обработка, когда в текст программы добавляются фрагменты из библиотечных файлов, хранящихся в исходном виде (обычно такие файлы называются файлами включения, описываются в программе предложением INCLUDE и имеют тип .H);

б) лексический анализ сформированной полной программы (выделение из программы стандартных языковых конструкций);

в) синтаксический контроль (проверка выделенных конструкций на соответствие синтаксису языка);

г) генерация машинного кода.

Сформированный объектный модуль записывается в файл, обычно имеющий тип .OBJ (только Паскаль - компиляторы сохраняют объектные модули в файлах типа .TPU или .DCU).

При разработке сложных программных систем иногда возникает необходимость написания модулей программы на различных алгоритмических языках. Для того, чтобы можно было объединить такие модули в единую программу, все компиляторы, используемые в операционной системе одного типа, должны формировать объектные модули стандартной структуры. Объектный модуль состоит из сло-

варя внешних символов, текста программы в виде двоичных команд и словаря перемещений.

Словарь внешних символов содержит информацию о всех внешних именах и ссылках, упоминаемых в данном модуле, и используется для организации межпрограммных связей. Каждая строка этого словаря включает одно имя функции или подпрограммы, используемое в модуле. В словаре перемещений хранится информацию о всех перемещаемых адресных константах, он используется при настройке загрузочного модуля по месту его расположения в ОЗУ.

Объектные модули могут храниться в последовательных или в библиотечных файлах. Последовательный файл обычно содержит один объектный модуль и имеет тип OBJ, библиотечный файл имеет тип LIB и представляет собой совокупность нескольких объектных модулей и оглавления.

Как правило, любая даже самая простая программа требует для своего выполнения дополнительных подпрограмм, реализованных в виде процедур и функций алгоритмического языка (ввод-вывод, обработка строк, математические функции и т.д.). Эти подпрограммы хранятся в виде объектных модулей в соответствующих библиотеках компилятора. Этап организации межпрограммных связей (компоновки) необходим для объединения независимо транслированных объектных модулей в единый загрузочный модуль (ЗМ) и реализуется с помощью специальных программы, которая называются компоновщиком.

На вход компоновщика подаются объектные модули в виде отдельных последовательных файлов (типа .OBJ) или подключаются библиотечные файлы (типа .LIB), содержащие объектные модули (рис. 4.3). Некоторые операционные системы предусматривают подключение на вход компоновщика дополнительных файлов (например, компоновщик Windows может обрабатывать файлы ресурсов типа .RES и файл описания модуля типа .DEF).

На выходе компоновщика формируются загрузочный модуль, который записывается на диск в файл типа .EXE или .COM и файл с картой памяти (типа .MAP). Карта памяти является обычным текстовым файлом и хранит структуру сформированного загрузочного модуля с указанием относительных адресов каждой функции внутри него.

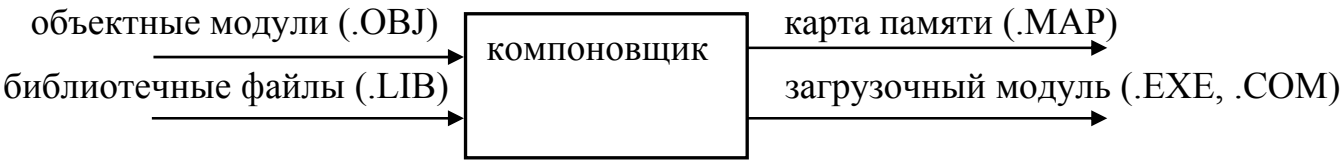


Рис. 4.3 Входные и выходные данные компоновщика

Возможны следующие варианты формирования загрузочного модуля:

а) *статический*, когда компоновщик включает в состав загрузочного модуля все дополнительные модули, требуемые для его выполнения;

б) *динамический*, когда компоновщик включает в состав загрузочного модуля только ссылки на дополнительные модули, требуемые для его выполнения; при этом сами дополнительные модули должны находится в специальных библиотеках динамической компоновки (файлы типа .DLL);

в) *оверлейный*, когда компоновщик включает в состав загрузочного модуля все дополнительные модули и ряд специальных команд, с помощью которых возможна загрузка такого модуля в ОЗУ отдельными сегментами.

Современные компиляторы после успешного формирования объектного модуля могут динамически вызывать компоновщик без дополнительных команд со стороны пользователя.

Из вышесказанного следует, что у программиста должен быть набор некоторых инструментальных средств, чтобы обеспечить успешную и быструю разработку программного обеспечения. В таблице 4.2 приведен перечень основных инструментов разработчика.

Таблица 4.2

№	Этап	Инструментарий
1	Разработка исходного модуля	Текстовый редактор
2	Формирование объектного модуля	Компилятор
3	Формирование загрузочного модуля	Компоновщик
4	Выполнение загрузочного модуля	Отладчик

4.3 Интегрированная среда программирования

Интегрированная Среда Разработки (Integrated Development Environment — IDE) - это пакет программ, в котором есть все необходимое для проектирования, запуска и тестирования приложений и где все нацелено на облегчение процесса создания программ. Впервые такие пакеты, разработанные фирмой Borland, появились в середине 80-х годов (система Турбо Паскаль). Для работы под управлением DOS было создано семь версий Турбо Паскаля, восьмая версия (Object Pascal) уже предназначалась для работы под управлением Windows и получила название Delphi. Система Delphi также прошла достаточно большой путь развития (к настоящему времени разработаны восемь версий).

IDE включает текстовый редактор кодов, редактор изображений, компилятор, компоновщик, отладчик, набор визуальных компонентов, инструментарий баз данных и другие программы. Все интегрированные системы программирования имеют очень похожие интерфейсы, поэтому дальнейшее их рассмотрение будем проводить на примере среды Delphi 7.

Основными объектами управления Delphi являются:

проект – набор модулей, форм, ресурсов и других файлов, относящихся к одному приложению; описание проекта хранится в текстовой файле с типом .DPR;

модуль – основная независимая программная единица проекта, может хранится в исходном или откомпилированном виде (тип .PAS или .DCU соответственно);

форма – основной визуальный объект приложения, выполняющий функции контейнера для хранения всех визуальных и невизуальных компонентов приложения, хранится в файле типа .DFM; каждому файлу формы всегда соответствует файл модуля .PAS;

настройки проекта (хранятся в файле типа .DOF);

библиотека компонентов – набор из нескольких сотен визуальных и невизуальных компонентов, используемых для организации интерфейсов, доступа к данным, для связи с другими приложениями и т.д.; имеется возможность добавления новых компонентов, которые обычно поставляются в виде файлов пакетов (тип .DPK) сторонними производителями.

В среде Delphi можно разрабатывать различные виды программного обеспечения - консольные или графические приложения, приложения для контрольной панели, динамические библиотеки (DLL), web – приложения, упакованные компоненты (DPK) и т.д.

Рассмотрим характеристики некоторых инструментов Delphi.

4.3.1 Текстовый редактор

Текстовый редактор Delphi является однооконным редактором со стандартным набором функций обработки текста (поиск, замена, работа с фрагментами через буфер, откат в случае выполнения ошибочных действий). Для одновременной обработки нескольких программных модулей используется стандартный блокнот с набором закладок, каждая из которых отображает программный код одного модуля.

Для работы с редактором предназначены пункты **Edit** и **Search** главного меню Delphi. Настройки редактора задаются в меню **Tools/Editor Options**.

4.3.2 Компилятор

Для управления режимами компиляции проекта используются пункты меню **Project/Compile**, **Project/Build** и **Project/Syntax Check**.

Режим **Compile** реализует следующие операции :

- проводится компиляция всех исходных модулей проекта, в которые были внесены изменения с момента последней компиляции;
- вызывается компоновщик, который формирует загрузочный модуль (.EXE).

Режим **Build** реализует следующие операции :

- проводится принудительная перекомпиляция всех исходных модулей проекта;
- вызывается компоновщик, который формирует загрузочный модуль (.EXE).

Режим **Syntax Check** используется для проверки всех модулей проекта, в которые вносились изменения, на наличие синтаксических ошибок, формирование объектных модулей и загрузочного модуля при этом не выполняется.

При работе с группой проектов используются меню **Project/Compile All Projects и Project/Build All Projects** для обработки всех проектов, включенных в группу. Настройка параметров компилятора проводится в меню **Project/Options/ Compiler** и **Project/Options/Compiler Messages**.

4.3.2 Отладчик

В ходе разработки программы могут возникать ошибки трех видов: синтаксические, ссылочные и семантические.

Синтаксические ошибки связаны с нарушением синтаксиса алгоритмического языка (например, пропущена скобка, не указан разделитель операторов, неправильно написано ключевое слово и т.д.) и обнаруживаются компилятором при компиляции программы.

Ссылочные ошибки связаны с нарушением ссылок на файлы, подключаемые к программе на этапе компоновки или с нарушением прав доступа при записи загрузочного модуля на диск и обнаруживаются компоновщиком (неправильно указан путь к библиотечным файлам, отсутствует нужная функция в библиотеке, запрещена запись в каталог, куда компоновщик пытается записать исполняемый файл).

Семантические ошибки проявляются только на этапе выполнения программы и связаны с делением на ноль, переполнением разрядной сетки, выходом за границы массива и т.д.; сообщения о таких ошибках выводятся операционной системой. К семантическим ошибкам относятся также ошибки, связанные с нарушением логики работы программы, когда программа работает, но формирует неправильные результаты. Для анализа причин возникновения семантических ошибок используется специальная программа – отладчик.

Отладчик включает средства временной приостановки выполнения программы и средства для контроля состояния переменных программы в момент приостановки. К средствам временной приостановки относятся:

выполнение программы до курсора (**Run / Run to Cursor**) – программа выполняется до строки, в которой установлен курсор;

пошаговый режим с заходом в подпрограммы (**Run /Trace Into**) – программа выполняется построчно, причем если какая-либо строка программы использует вызов подпрограммы, то эта подпрограмма также исполняется в пошаговом режиме;

пошаговый режим без заходам в подпрограммы (**Run /Step Over**) - программа выполняется построчно, все подпрограммы исполняются как один оператор.

включение точки остановки в произвольном месте программы (**Run /Add Breakpoints**).

Средства контроля состояния переменных в момент приостановки включают:

окно наблюдения за переменными программы (Watch);

окно редактирования значений переменных (Evaluate/Modify);

всплывающая подсказка.

Окно наблюдения применяется для вывода значения любых переменных программы и любых выражений с использованием этих переменных. Добавление идентификаторов переменных и выражений в это окно проводится через окно **Watch Properties** (меню **Run /Add Watch**). Для удаления переменных из окна укажите мышью имя переменной, с помощью правой кнопки мыши вызовите контекстное меню и выберите в нем пункт **Delete Watch**. Для удобства окно наблюдения можно сделать всплывающим (пункт контекстного меню **Stay On Top**).

Через окно редактирования значений переменных можно не только просматривать значения любых переменных и выражений в моменты остановки программы, но и присваивать им новые значения через поле **New Value** (меню **Run / Evaluate/Modify**).

Для контроля значений переменных программы в точке остановки удобно использовать всплывающую подсказку, отображающую значение переменной, на которую указывает курсор мыши.

Завершить выполнение приостановленной программы можно через меню **Run/ Program Reset**.

4.3.3 Средства настройки среды

Установка параметров интегрированной среды проводится в меню **Tools/ Environment Options** (общие параметры окружения) и **Tools/Editor Options** (параметры текстового редактора). Кроме того, отдельно устанавливаются параметры проекта (**Project/Options**)/

4.3.5 Методология разработки программ

Большинство программ, разрабатываемых в среде Delphi, управляются событиями, т.е. в первую очередь в системе возникает какое-либо событие и только после этого запускается некоторая последовательность действий, реагирующих на это событие. Такими событиями могут быть нажатия клавиш клавиатуры, перемещение мыши, срабатывания таймера.

С технологической точки зрения приложение Windows представляет собой набор объектов, каждый из которых является классом (см. 2.2.8) и имеет набор свойств и методов. Например, объект «Форма» имеет такие свойства, как размеры (высота и ширина), заголовок, цвет, расположение на экране и др. Методы формы – это процедуры, которые запускаются при возникновении различных событий (создание или активация формы, щелчок или перемещение мыши, нажатие клавиши и т.д.).

Для работы с объектами Delphi использует специальную программу – инспектор объектов. На закладке **Properties** устанавливаются значения различных свойств объекта, а закладка **Events** используется для ввода программного кода методов, реагирующих на различные события.

Основные приемы практической работы в среде Delphi Вы можете освоить, выполнив лабораторную работу (ссылка).

Для уменьшения вероятности возникновения ошибок и ускорения процесса разработки программ необходимо соблюдать следующие правила:

а) каждое функциональное требование к программе должно быть реализовано в виде отдельной подпрограммы (функции или процедуры);

б) каждая подпрограмма должна быть удобочитаемой, структурированной и иметь небольшой размер; структурированность предполагает запись текста программы с выделением отступами основных конструкций языка программирования, например, фрагмент на рис. 4.4 гораздо удобнее для понимания, чем фрагмент на рис.4.5, хотя оба выполняют одинаковые действия:

в) нежелательно использовать без крайней необходимости оператор безусловного перехода GOTO, который может чрезвычайно усложнить понимание логики работы программы;

```
If (alfa=1) then
   begin
      beta := 5;
      gamma := 10;
   end
  else
   begin
      beta:=0;
      gamma := 100;
   end;
```

Рис. 4.4

```
If (alfa=1)
then
Begin beta := 5; gamma := 10;
end
else begin beta:=0;
gamma := 100;
end;
```

Рис. 4.5

г) программа должна иметь подробные комментарии;

д) в программах желательно использовать только стандартные управляющие конструкции.

В теории программирования доказана теорема о том, что любая, сколь угодно сложная программа может быть разработана на основе семи стандартных управляющих структур – последовательности, трех видов ветвлений (if–then–else, if–then, case) и трех видов циклов (for-do, while-do, do-until).

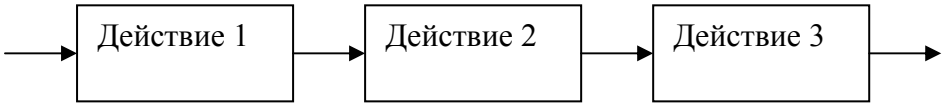
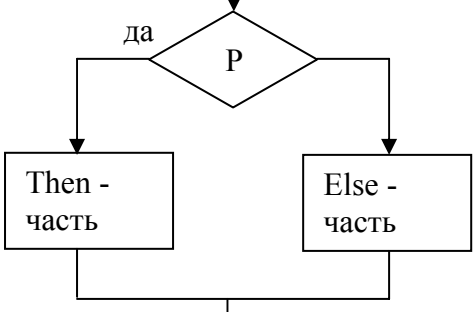


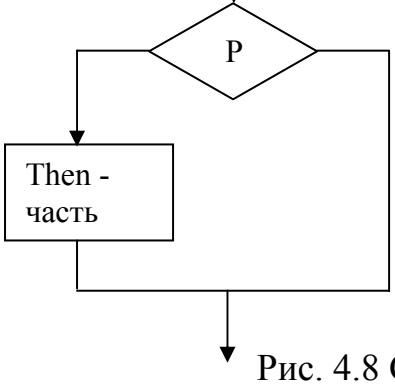
Рис. 4.6 Последовательность



Пример реализации:

```
If (a > b) then
     Max :=a
  else
     min :=b;
```

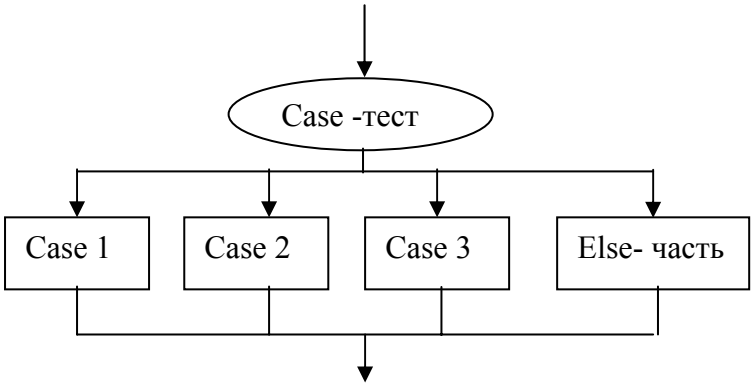
Рис. 4.7 Структура IF – THEN – ELSE



Пример реализации:

```
If  (beta >=0) then
   alfa := 100;
```

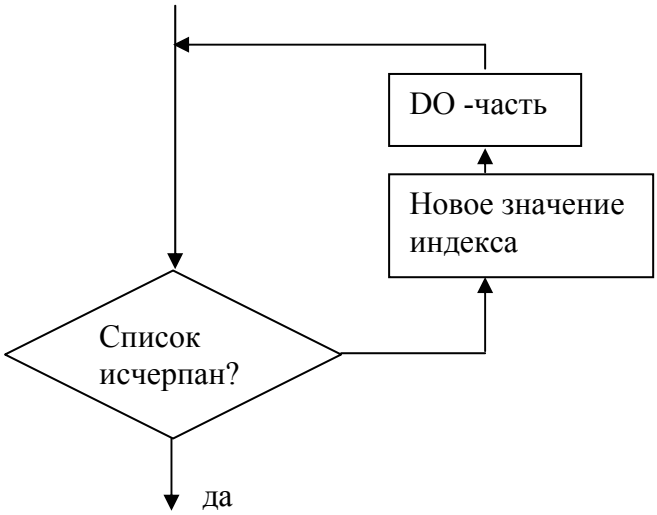
Рис. 4.8 Структура IF – THEN



Пример реализации:

```
Case symbol of
   ‘A’..’Z’: writeln(‘латинская буква);
   ‘A’..’Я’: writeln(‘буква кириллицы);
 else
    writeln(‘неизвестный символ);
end;
```

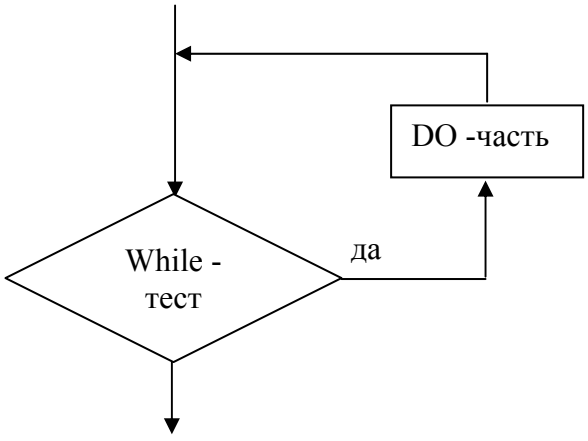
Рис. 4.9 Структура CASE



Пример реализации:

```
For i:=1 to 20 do
   a[i] :=i;
```

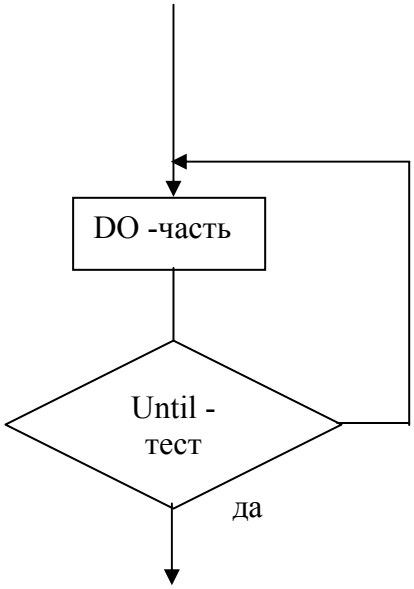
Рис. 4.10 Структура FOR-DO (предопределенный цикл)



Пример реализации:

```
While  (sum <=100) do
   Sum:=sum + 5;
```

Рис. 4.11 Структура WHILE-DO (цикл с предусловием)



Пример реализации:

```
Repeat
  sum :=sum + 5;
until sum >=100;
```

Рис. 4.12 Структура DO-UNTIL (цикл с постусловием)

На рис. 4.6 – 4 .12 показаны управляющие структуры и примеры их реализации на языке Паскаль. Здесь прямоугольник обозначает некоторую последовательность действий, P – предикатный узел, определяющий логическое условие, которое может принимать два значения – ИСТИНА или ЛОЖЬ. Каждая структура имеет один вход и один выход, а каждый прямоугольник может, в свою очередь, представлять собой произвольное количество стандартных управляющих структур.

Структура CASE реализует множественный выбор, в отличие от IF – THEN – ELSE, где выбор возможен только между двумя вариантами.

Предопределенный цикл всегда исполняется заданное число раз; возможна ситуация, при которой тело цикла не исполняется ни разу – если начальное значение индексной переменной превышает ее конечное значение.

Количество исполнений команд в теле условных циклов (с пред- или постусловием) зависит от значения условия, причем здесь возможно бесконечное исполнение команд, что называется зацикливанием. Тело цикла с предусловием может вообще не исполняться ни одного раза, если значение предиката при входе в цикл сразу принимает значение ЛОЖЬ. Тело цикла с постусловием всегда исполняется хотя бы один раз.

5. Инструментальные средства.

Текстовый процессор: основные операции и объекты. Принципы работы текстовых процессоров. Краткий обзор основных возможностей редактора MS Word. Табличный процессор: основные операции и объекты. Краткий обзор основных возможностей MS Excel. Назначение и классификация программ сжатия информации. Программы резервного копирования, архиваторы, динамические дисковые компрессоры. Понятие и структура архивного файла. Основные операции, выполняемые архиваторами. Алгоритмы сжатия информации: исключение повторяющихся последовательностей, Хаффмана. Краткий обзор возможностей современных архиваторов. Понятие компьютерного вируса. Классификация вирусов. Методы заражения файлов и загрузочной записи. Защита от проникновения вирусов. Классификация и алгоритмы антивирусных программ: ревизоры, фильтры, доктора, полифаги. Методика удаления вирусов. Краткий обзор возможностей антивирусных программ.

5.1 Текстовые процессоры

Текстовый процессор (или текстовый редактор) - программа, предназначенная для создания, редактирования, обработки и печати текстовых документов. Существует минимальный набор функций, выполняемых любым текстовым редактором:

- создание нового текстового документа;
- редактирование текстового документа (эта функция может выполняться параллельно с созданием документа);
- запись документа в виде файла на магнитный диск;
- считывание документа с магнитного диска в буферную память редактора;
- поиск в тексте заданной строки символов.

Минимальный набор функций характерен для редакторов, встроенных в файловые процессоры или в системы программирования (например, текстовые редакторы FAR или ТурбоПаскаль). Современные текстовые редакторы имеют, как правило, гораздо более широкие возможности:

- выполнение операций с фрагментами текста (вставка, замена, перемещение, удаление, копирование);
- форматирование текста (установка левой и правой границ текста, а также абзацного отступа, центрирование строк, автоматический перенос слова на следующую строку при выходе за границу абзаца и т.д.);
- поиск заданной строки символов и замена ее другой строкой;
- печать текста (разбивка на страницы, нумерация страниц, установка шрифта, качества печати и т.д.).;

- работа с таблицами и графическими изображениями;
- наличие режима отката;
- работа с шаблонами документов и т.д.

Работа с текстом может проводиться в двух режимах: вставки и замены. В режиме вставки вводимый символ появляется в позиции, отмеченной курсором, а оставшаяся часть строки от позиции курсора до конца строки сдвигается вправо. В режиме замены вводимый символ заменяет собой символ, находящийся в отмеченной курсором позиции, и сдвига строки не происходит.

Объектами управления текстового процессор являются: отдельный символ, слово, абзац, фрагмент текста, таблица, встроенный объект, документ. Операции с этими объектами выполняются в определенной последовательности – объект выделяется и только затем выполняется операция.

В настоящее время одним из распространенных редакторов является многооконный редактор MS Word, входящий в пакет программ MS Office. Он выполняет все типовые операции по обработке текста и имеет большой набор дополнительных функций. MS Word позволяет обрабатывать одновременно несколько документов, создавая для каждого из них собственное окно. Обмен информацией между документами проводится с помощью системного буфера (Clipboard). Заметим, что с помощью буфера текстовый редактор может включать в текстовый документ информацию из любого приложения Windows (например, из графического редактора) или файла.

Возможности современных текстовых редакторов и методика работы с ними описаны в большом количестве литературы, поэтому основное изучение этого вопроса проводится самостоятельно. Кроме того, по этой тематике выполняется лабораторная работа № 2.

5.2 Табличные процессоры

Табличная форма представления информации является одной из самых удобных и широко используемых при анализе управленческой, экономической и бухгалтерской информации. Для автоматизации обработки такой информации предназначены специально разработанные пакеты программ, которые называются таб-

личными процессорами. Табличные данные в памяти ЭВМ хранятся в виде электронных таблиц, с помощью которых пользователь может:

- создавать новые и редактировать уже существующие табличные формы документов;
- проводить обработку табличных данных в соответствии с заданными алгоритмами;
- выводить данные из таблицы на экран монитора или принтер в виде графиков и диаграмм;
- формировать на основе табличных данных отчеты произвольной формы;
- сохранять все результаты работы на внешних запоминающих устройствах.

Электронная таблица - это двумерные массивы строк и столбцов, размещаемые в памяти ЭВМ. Строки идентифицируются числами, столбцы - буквами латинского алфавита. Место пересечения строки и столбца называется клеткой (ячейкой) таблицы. В качестве идентификатора каждой клетки используются ее координаты (номера столбца и строки). Достоинствами электронных таблиц по сравнению с их бумажными аналогами являются:

более удобный и быстрый ввод данных за счет широкого использования операции копирования отдельных клеток таблицы и блоков, состоящих из нескольких клеток;

возможность с помощью формул задавать зависимость одних значений табличных данных от других;

возможность моделирования определенных ситуаций путем внесения изменений в исходные значения данных;

возможность многократного использования, модификации и распечатки в нужном количестве экземпляров.

В настоящее время наиболее широко применяется табличный процессор MS Excel, основным рабочим объектом которого является книга, состоящая из нескольких листов. На каждом листе имеется таблица, которая может содержать до 16536 строк и до 256 столбцов.

Таблицы Excel содержат два вида данных: константы и формулы (выражения). Константы могут содержать следующие значения: числовые, текстовые, дату/время. Числовые константы могут состоять только из цифр и символов “+”, “-“, “E”, “e”, “%”, “/”, ”точка”, “запятая”, “$”. Текстовые константы состоят из любых символов, причем если такая константа содержит только цифры, то перед ними необходимо поставить знак “апостроф” или заключить в кавычки и поставить перед ними знак равенства. Примеры текстовых констант:

осень, 46-11-11, ‘1234, =”1234”.

Константы типа “дата/время” содержат значения даты или времени, в качестве разделителей могут использоваться символы “/” или “точка”.

Формулы обязательно начинаются со знака равенства и могут содержать константы, знаки арифметических операций, идентификаторы ячеек таблицы, функции и т.д. Excel различает в выражениях два типа идентификаторов ячеек: абсолютный адрес ячейки и относительный адрес ячейки. При выполнении команды копирования содержимого ячейки, значение которой задано выражением с абсолютными адресами, в другую ячейку, абсолютные адреса не изменяются. Если в выражении использованы относительные адреса, то Excel автоматически проводит их коррекцию с учетом координат ячейки-приемника информации. Абсолютные адреса ячеек задаются с применением символа '$', относительные адреса - без использования этого символа. Возможно применение смешанных адресов, когда, например, номер столбца считается относительным, а номер строки – абсолютным. Например: A5 - относительный адрес ячейки, находящейся на пересечении первого столбца и пятой строки; A5 – абсолютный адрес этой же ячейки, A$5 – смешанный адрес.

Обратите внимание: идентификаторы ячеек, используемые в расчетных выражениях, должны содержать буквы только латинского алфавита!

Excel содержит более сотни встроенных функций для обработки как числовых, так и строковых данных. Наиболее часто используются функции суммирования, определения максимального и минимального значений, определения среднего значения, логические функции и т.д.

Ширина (размер) ячейки Excel определяется числом символов, выводимых в эту клетку на экране монитора и может быть задана пользователем. По умолчанию

размер ячейки составляет 9 позиций. Независимо от размера каждая ячейка может содержать информацию размером до 65535 символов.

Excel поддерживает также такие объекты, как блок и диапазон. Блоком называется совокупность произвольного числа ячеек электронной таблицы, образующих прямоугольник. Отдельную клетку таблицы также можно рассматривать как частный случай блока. Блок может участвовать в операции копирования в качестве как приемника информации, так и источника информации и обозначается следующим образом:

< адрес левой верхней ячейки > : < адрес правой нижней ячейки >

Например, блок A1:B2 содержит ячейки A1,A2,B1,B2.

Блок, состоящий из нескольких столбцов или строк таблицы, называется диапазоном и может быть описан, например, так: A:D (этот блок включает столбцы A,B,C,D)..

Кроме стандартного представления данных Excel дает возможность пользователю вывести информацию в графической форме путем построения различных диаграмм и графиков. Для представления данных в графической форме удобно использовать мастер диаграмм.

Основными понятиями, с которыми работает этот мастер, являются понятия ряда данных и категории. Ряд данных – это множество значений, которые наносятся на диаграмму, а категории задают положение конкретных значений в ряде данных. Если представить в графическом виде некоторую одномерную функцию, то значения ординат этой функции (ось Y) будут являться рядом данных, а значения абсцисс (ось X) - категориями.

На одной диаграмме можно отобразить до 255 рядов, каждый ряд может иметь до 4000 точек данных. При создании диаграммы необходимо задать расположение исходных данных, выбрать тип диаграммы (линейная, круговая, гистограмма и т.д.) и формат отображения данных, а также указать необходимые подписи к осям координат и диаграмме в целом.

Возможности современных табличных процессоров и методика работы с ними описаны в большом количестве литературы, поэтому основное изучение этого во-

проса также проводится самостоятельно. Кроме того, по этой тематике выполняется лабораторная работа № 3.

5.3 Программы сжатия данных

5.3.1 Классификация программ сжатия

Сжатие данных используется для следующих целей:

- создание резервных копий файлов и каталогов;
- сокращение трафика при передаче данных по каналам связи;
- уменьшение размеров исполняемых файлов;
- увеличение объема информации, хранящейся на ВЗУ.

На рис. 5.1 представлена классификация программ сжатия данных.

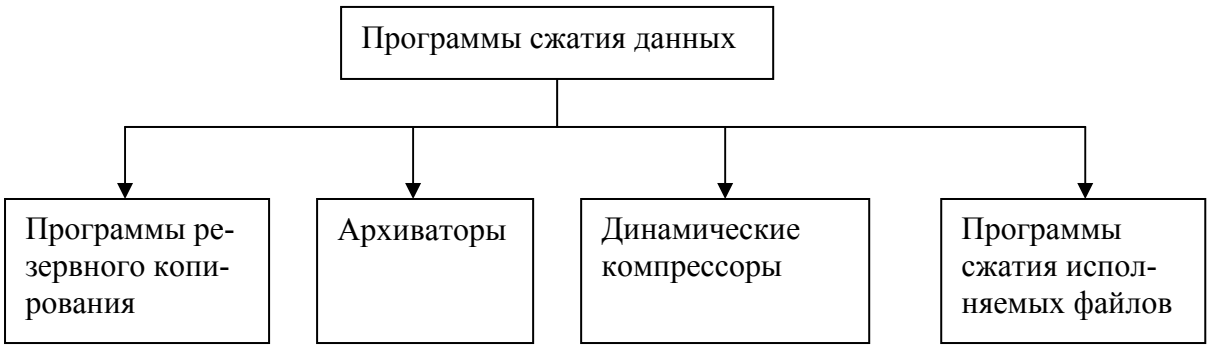


Рис. 5.1 Программы сжатия данных

Программы резервного копирования в основном предназначены для создания резервных копий файлов и каталогов. Основным достоинством таких программ является большой набор внешних устройств, с которыми они могут работать, а недостатком - невысокая степень сжатия. Примерами являются программы FastBack Plus, PC Backup. В состав ОС Windows XP включается программа резервного копирования NTBackup.

Архиваторы – программы, предназначенные для создания и обслуживания архивных файлов. Архивный файл - это набор из одного или нескольких файлов, помещенных в сжатом виде в единый файл. имеющий свое оглавление. Из всех программ сжатия данных архиваторы используются наиболее часто.

Динамический компрессор предназначен является резидентной программой и применяется для сжатия и распаковки без каких – либо дополнительных указаний со стороны пользователя. При любой записи на диск проводится сжатие данных, а

при чтении – распаковка. Для работы динамического компрессора на жестком диске создается специальный файл, называемый сжатым диском, размер которого задается пользователем во время начальной установки компрессора. Объем реальных данных, хранящихся на сжатом диске, может превышать его физический размер более, чем в два раза.

Достоинством дисковых компрессоров является увеличение эффективного объема жесткого диска и прозрачность работы для пользователя, а недостатком – некоторое снижение производительности компьютера. Кроме того, в случае порчи жесткого диска практически невозможно восстановить данные, хранящиеся на сжатом диске. Примерами дисковых компрессоров являются программы Stacker, DrvSpace, DoubleSpace.

Программы сжатия исполняемых файлов предназначены для уменьшения размера файлов, сформированных компоновщиком (типа COM и EXE). Примерами таких программ являются программы LZEXE, EXEPACK.

5.3.2 Алгоритмы сжатия информации

В настоящее время разработано достаточно большое количество алгоритмов сжатия информации, которые основаны на поиске и устранении избыточности. Все алгоритмы делятся на две группы:

- алгоритмы сжатия с потерями, которые удаляют из потока данных информацию, незначительно влияющую на точность передаваемых данных или вообще не воспринимаемую человеком (такие алгоритмы используются для обработки аудио- и видеоизображений);
- алгоритмы сжатия без потерь, при использовании которых данные можно восстановить абсолютно достоверно.

Программы сжатия данных используют алгоритмы второй группы, основными из которых являются:

- сжатие последовательности повторяющихся символов или групповое кодирование (RLC – Run Length Coding);
- кодирование Хаффмана (Haffman);
- кодирование Лемпеля – Зива (LZ77).

Используются также комбинации этих алгоритмов, например алгоритм Лемпеля – Зива – Хаффмана (LZH)

5.3.2.1 Алгоритм RLC

Алгоритм RLC основан на представлении последовательности одинаковых байтов в виде двух значений – значения байта и количества повторов этого байта. Например, последовательность символов NNNMMMMMPPPMMMMMM может быть закодирована в следующем виде: N3M5P3M6, т.е. вместо 17 байтов строка будет занимать 8 байтов.

Этот алгоритм эффективно сжимает графические и двоичные файлы, причем для двоичных данных можно вообще не хранить значения байтов, а сохранять только количество повторяющихся битов, например последовательность байтов 11111100 00000001 11111100 00000011 может быть записана так: 69782.

В текстовых файлах встречается много одиночных символов, поэтому в результате использования метода RLC можно получить архивный файл большего размера, чем исходный файл. Для устранения этого недостатка можно не кодировать последовательности, состоящие менее чем из трех символов.

5.3.2.2 Алгоритм Хаффмана

Алгоритм Хаффмана рассматривает входной поток данных, как последовательность несвязанных между собой байтов. Алгоритм основан на частотном анализе повторяемости каждого символа внутри входного потока и генерации для каждого символа кода, длина которого обратно пропорциональна частоте повторяемости символа. Т.е. для кодирования символов используется код переменной длины и чем чаще встречается символ в тексте, тем короче этот код.

Последовательность шагов алгоритма может быть представлена следующим образом:

- строится частотная матрица входного потока данных;
- проводится упорядочивание частотной матрицы по убыванию частоты повторения символов;
- строится кодировочное дерево;
- формируется кодировочная таблица.

Рассмотрим пример использования алгоритма Хаффмана для кодирования строки КОЛОКОЛ ОКОЛО КОЛОКОЛЬНИ.

а) определяем частоту повторения каждого символа и строим частотную матрицу:

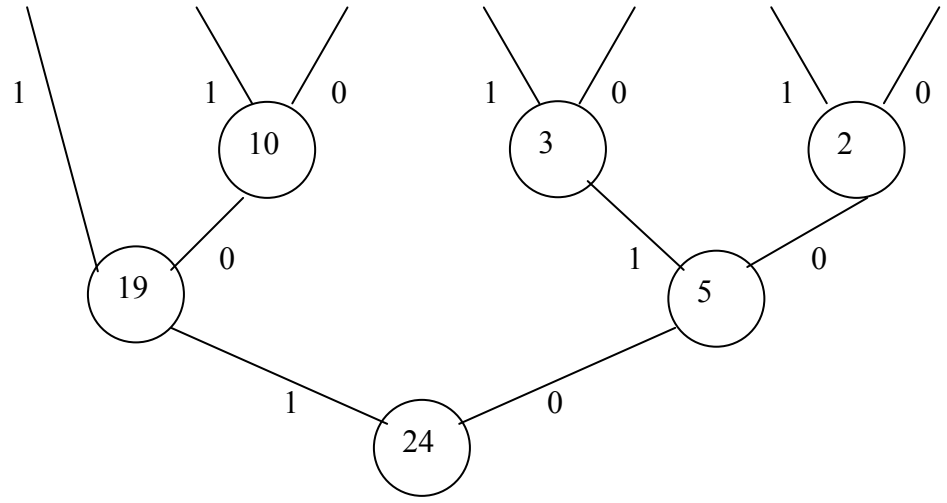
Символ	К	О	Л	пробел	Ь	Н	И
Кол-во	5	9	5	2	1	1	1

б) упорядочим матрицу по убыванию частоты повторения

Символ	О	К	Л	пробел	Ь	Н	И
Кол-во	9	5	5	2	1	1	1

в) строим кодировочное дерево:

Символ	О	К	Л	пробел	Ь	Н	И
Кол-во	9	5	5	2	1	1	1



соединяем попарно столбцы таблицы справа налево;

суммируем частоты повторения соответствующих символов;

обходим полученный граф от вершины к таблице, обозначаем ребра (0 - правое ребро, 1 – левое ребро) и формируем коды символов:

Символ	О	К	Л	пробел	Ь	Н	И
код	11	101	100	011	010	001	000

Обратите внимание: значение вершины дерева должно быть равно общему числу символов во входной строке.

г) кодируем строку:

10111100 11101111 00011111 01111001 11011110 01110111 10001000 1000

Таким образом длина результирующей строки получилась равной 7,5 байтов вместо 24 байтов у исходной строки. Полученную строку можно сжать еще больше, применив к ней алгоритм RLC.

В теории кодирования показывается, что код Хаффмана является префиксным, то есть код никакого символа не является началом кода какого-либо другого символа. Из этого следует, что код Хаффмана однозначно восстановим, даже если не сохраняется длина кода каждого символа. Для восстановления к сжатому файлу необходимо сохранить только кодировочное дерево в компактном виде. Для каждого сжатого файла будет храниться свое кодировочное дерево.

5.3.2.3 Алгоритм Лемпела - Зива

Алгоритм LZ77 предложен Лемпелом и Зивом в 1977 году и ориентирован на сжатие любых видов текстов, то есть использует факт неоднократного повторения "слов" – последовательностей байтов. Он может быть сформулирован следующим образом: «если в более раннем тексте уже встречалась подобная последовательность байтов, то в сжатый файл записывается только ссылка на эту последовательность в виде (смещение, длина), а не сам текст».

В этом случае фраза КОЛОКОЛ ОКОЛО КОЛОКОЛЬНИ будет закодирована в виде: КОЛО(-4,3)_(-5,4)О _(-14,7)ЬНИ, где символом «_» обозначен пробел.

5.3.3 Архиваторы

Существует достаточно большое количество программ для создания и ведения архивных файлов, отличающихся алгоритмами сжатия информации и, как следствие, степенью сжатия файлов и временем обработки информации. К ним относятся программы WINRAR, WINZIP, WINARJ и др. Они позволяют проводить сжатие текстовых файлов на 60-80 %, выполняемых файлов - на 20-30 %.

Архивный файл – это файл, представляющий собой совокупность сжатых файлов и оглавления, причем доступ к сжатым файлам проводится только через оглавление. Имя архивному файлу задает пользователь, а расширение обычно присваивается по умолчанию (.ARJ – для архиваторов WINARJ, .RAR - для архиваторов WINRAR и т.д.). В оглавлении для каждого сжатого файла хранится следующая информация:

- имя и расширение;

- размеры до и после сжатия;
- коэффициент или степень сжатия;
- дата и время записи в архив;
- атрибуты;
- контрольная сумма.

Если обозначить размер файла до сжатия V_1, а после сжатия – V_2, то коэффициент сжатия ($K_{сж}$) и степень сжатия ($S_{сж}$) будут соответственно равны:

$$K_{сж} = V_2 / V_1 \quad , \qquad S_{сж} = 1 - V_2 / V_1 \; .$$

Контрольная сумма файла используется для контроля целостности файла и представляет собой 16- или 32-разрядное целое беззнаковое число, значение которого зависит от значения каждого байта файла. Если по какой-то причине изменить содержимое хотя бы одного бита внутри файла, то контрольная сумма также изменится. Значение контрольной суммы рассчитывается при записи файла в архив после его сжатия. Контрольная сумма часто называется кодом циклического контроля (CRC - Cyclical Redundancy Check).

Современные архиваторы могут создавать архивы следующих видов: стандартный, самораспаковывающийся, многотомный, непрерывный. Самораспаковывающийся архив представляет собой исполняемый файл типа .EXE, который после запуска автоматически создает в текущем каталоге все файлы, включенные в этот архив в нормальном (несжатом) виде. При создании такого архива архиватор включает в него дополнительно небольшую программу распаковки.

При обработке больших файлов удобно использовать многотомный архив, представляющий собой архивный файл, разбитый на несколько фрагментов определенного размера. Каждый из таких фрагментов может быть сохранен на отдельном гибком магнитном диске.

Непрерывный архив – это архив, все сжатые файлы которого рассматриваются как единый поток данных. При этом существенно сокращается размер архива, но имеются следующие недостатки:

- обновление непрерывных архивов (т.е. добавление файлов в уже существующий архив или их удаление) происходит медленнее, чем обычных;

- чтобы извлечь один файл из непрерывного архива, приходится анализировать все предыдущие заархивированные файлы, поэтому извлечение отдельных файлов из середины непрерывного архива происходит медленнее, чем извлечение из обычного архива. Однако если из непрерывного архива извлекаются все или несколько первых файлов, то в этом случае скорость распаковки практически равна скорости распаковки обычного архива;
- если в непрерывном архиве какой-либо файл окажется поврежденным, то не удастся извлечь и все файлы, следующие после него. Поэтому при сохранении непрерывного архива на ненадежном носителе (например, на дискете) рекомендуется добавлять информацию для восстановления.

Непрерывные архивы предпочтительнее использовать в следующих случаях:

- архив предполагается редко обновлять;
- вы планируете чаще распаковывать весь архив, нежели извлекать из него один или несколько файлов;
- нужно достичь более плотной степени сжатия, даже в ущерб скорости упаковки.

Все архиваторы имеют типовой набор функций для работы с архивными файлами, основные из которых приведены в таблице 10

Таблица 5.1

Операция	**Код операции**	**Назначение**	**Результат операции**
Add	a	добавление файлов в архив	добавление файлов в архив (если указанный архивный файл не существует, то он создается)
Move	m	пересылка файлов в архив	перемещение файлов в архив (если указанный архивный файл не существует, то он создается)
Update	u	обновление файлов в архиве	добавление указанных файлов в архив (если эти файлы в архиве отсутствуют) или замещение указанных файлов, если в архиве существуют более ранние версии этих файлов.
Freshen	f	обновление файлов в архиве	замещение указанных файлов, если в архиве существуют более ранние версии этих файлов.
Test	t	тестирование архива	проверка целостности всех файлов архива.
Delete	d	удаление файлов из архива	удаление из архива указанных файлов.
Extract	e, x	извлечение файлов из архива	извлечение файлов из архивного файла.
List	l, v	просмотр оглавления архива	просмотр оглавления архивного файла
Print	p, c	просмотр файлов в архиве	вывод на монитор указанных файлов

Современные архиваторы могут работать в диалоговом и командном режимах. Диалоговый режим значительно упрощает работу с программой, но имеет ограниченный набор реализуемых функций. Командный режим рассчитан на опытных пользователей и здесь доступны все функции архиваторов.

Основные принципы работы с архиваторами изучаются в лабораторной работе № 4.

5.4 Программы защиты от компьютерных вирусов

5.4.1 Классификация компьютерных вирусов

Компьютерный вирус - это специально написанная небольшая программа, которая может присоединяться к другим программам и выполнять различные нежелательные для компьютера и его владельца действия. Употребление термина "вирус" вызвано многими сходствами компьютерного и биологического вирусов:

- большинство вирусов представляют опасность для системы, в которой они паразитируют;
- вирус проявляет себя не сразу, а через определенный интервал времени; причем в это время вирус продолжает распространяться;
- вирусы быстро размножаются и разносятся на большие расстояния;
- важную роль в борьбе с"заболеваниями" играет профилактика.

Когда "инфицированная" программа начинает работу, то сначала получает управление вирус, который находит и заражает другие программы и выполняет другие действия (портит файлы, засоряет оперативную или внешнюю память и т.д.). Никаких сообщений на экран при этом не выводится, поэтому действия вируса остаются незамеченными. Затем вирус передает управление программе, в которой он находится, и она работает, как и обычная незараженная версия. Обычно заражение вирус производит путем записи своей копии в конец программы и изменения стартовой точки таким образом, чтобы при запуске программы управление было передано вирусу.

Обязательным свойством любого вируса является возможность создавать свои дубликаты (не обязательно совпадающие с оригиналом) и внедрять их в вычислительные сети и/или файлы, системные области компьютера и прочие выполняемые объекты. При этом дубликаты сохраняют способность к дальнейшему распространению. Вирусы можно разделить на классы по следующим основным признакам: среда обитания, операционная система (ОС), особенности алгоритма работы, деструктивные возможности.

По среде обитания вирусы можно разделить на файловые, загрузочные, макро и сетевые. Файловые вирусы различными способами внедряются в выполняемые файлы, загрузочные вирусы записывают себя в загрузочный сектор диска, макровирусы заражают файлы с текстовыми документами и электронными таблицами, созданные в MS Office. Сетевые вирусы используют для своего распространения протоколы или команды компьютерных сетей и электронной почты.

Существуют различные сочетания - например, файлово-загрузочные вирусы, заражающие как файлы, так и загрузочные сектора дисков. Другой пример такого сочетания - сетевой макро-вирус, который не только заражает редактируемые документы, но и рассылает свои копии по электронной почте.

Заражаемая ОС является вторым уровнем деления вирусов на классы. Каждый файловый или сетевой вирус заражает файлы какой-либо одной или нескольких систем- DOS, Windows, Win95/NT, OS/2 и т.д. Загрузочные вирусы также ориентированы на конкретные форматы расположения системных данных в загрузочных секторах дисков.

Среди особенностей алгоритма вирусов выделяются следующие пункты: резидентность, использование стелс-алгоритмов, самошифрование и полиморфичность, использование нестандартных приемов. Резидентный вирус при инфицировании компьютера оставляет в оперативной памяти свою резидентную часть, которая затем перехватывает обращения ОС к объектам заражения и внедряется в них. Резидентные вирусы находятся в памяти и являются активными вплоть до выключения компьютера или перезагрузки ОС. Нерезидентные вирусы не заражают память компьютера и сохраняют активность ограниченное время. Некоторые вирусы оставляют в оперативной памяти небольшие резидентные программы, которые не

распространяют вирус, такие вирусы считаются нерезидентными. Резидентными можно считать макро-вирусы, поскольку они постоянно присутствуют в памяти компьютера в течение всего времени работы зараженного редактора.

Использование СТЕЛС-алгоритмов позволяет вирусам полностью или частично скрыть себя в системе. Наиболее распространенным стелс-алгоритмом является перехват запросов ОС на чтение/запись зараженных объектов. Стелс-вирусы при этом либо временно лечат их, либо "подставляют" вместо себя незараженные участки информации. В случае макро-вирусов наиболее популярный способ - запрет вызовов меню просмотра макросов.

Самошифрование и полиморфичность используются практически всеми типами вирусов для того, чтобы максимально усложнить процедуру обнаружения вируса. Полиморфик-вирусы - это достаточно труднообнаруживаемые вирусы, не имеющие сигнатур, т.е. не содержащие ни одного постоянного участка кода. В большинстве случаев два образца одного и того же полиморфик-вируса не будут иметь ни одного совпадения. Это достигается шифрованием основного тела вируса и модификациями программы-расшифровщика.

По деструктивным возможностям вирусы можно разделить на:

- безвредные, т.е. никак не влияющие на работу компьютера (кроме уменьшения свободной памяти на диске в результате своего распространения);
- неопасные, влияние которых ограничивается уменьшением свободной памяти на диске и графическими, звуковыми и пр. эффектами;
- опасные вирусы, которые могут привести к серьезным сбоям в работе компьютера;
- очень опасные, в алгоритм работы которых заведомо заложены процедуры, которые могут привести к потере программ, уничтожить данные или удалить необходимую для работы компьютера информацию, записанную в системных областях памяти.

Кроме традиционных вирусов, основным признаком которых является заражение других объектов, возможны и другие виды угроз. Таким угрозами могут быть например, программы – черви и троянские программы. Черви (Worms) используют для распространения в основном уязвимости операционных систем. На-

звание этого класса было дано исходя из способности червей "переползать" с компьютера на компьютер, используя сети, электронную почту и другие информационные каналы. Также благодаря этому многие черви обладают достаточно высокой скоростью распространения.

Черви проникают на компьютер, вычисляют сетевые адреса других компьютеров и рассылают по этим адресам свои копии. Помимо сетевых адресов часто используются данные адресной книги почтовых клиентов. Представители этого класса вредоносных программ иногда создают рабочие файлы на дисках системы, но могут вообще не обращаться к ресурсам компьютера (за исключением оперативной памяти).

Троянские программы (Trojans) выполняют на поражаемых компьютерах несанкционированные пользователем действия, т.е. в зависимости от каких-либо условий уничтожают информацию на дисках, приводят систему к "зависанию", воруют конфиденциальную информацию и т.д. Эти программы не являются вирусами в традиционном понимании этого термина (т.е. не заражают другие программы или данные) и не способны самостоятельно проникать на компьютеры, они обычно распространяются злоумышленниками под видом "полезного" программного обеспечения. При этом наносимый ими вред может во много раз превышать потери от традиционной вирусной атаки.

В последнее время наиболее распространенными типами вредоносных программ, портящими компьютерные данные, стали черви. Далее по распространенности следуют вирусы и троянские программы. Некоторые вредоносные программы совмещают в себе характеристики двух или даже трех из перечисленных выше классов.

5.4.2 Классификация антивирусных программ

Защита от воздействия вирусов может проводиться двумя методами:

механическая защита;

программная защита.

Механическая защита используется для полного запрета записи на диск. Это самый надежный метод, но он может использоваться только для гибких магнитных

дисков и некоторых типов USB флэш-накопителей. Кроме того, в любом случае на диск надо будет что-либо записывать и в этот момент на диск может попасть вирус.

Программная защита в настоящее время используется наиболее широко. Существуют различные типы антивирусных программ, отличающихся алгоритмами работы. К ним относятся: ревизоры, мониторы, сканеры.

Программы – ревизоры предназначены для контроля основных параметров заданных файлов. К таким параметрам относятся размер и контрольная сумма файла. Список файлов для проверки определяется пользователем обычно сразу после установки операционной системы, когда есть полная уверенность в том, что вирусы в системе отсутствуют. Ревизор сохраняет исходные параметры файлов и при запуске определяет текущие значения этих параметров. Запуск проводится по команде пользователя. Если текущие значения отличаются от исходных, то выводится предупреждающее сообщение о повреждении файла, идентификация вируса не проводится.

Мониторы – это резидентные программы, которые постоянно находятся в оперативной памяти и контролируют различные опасные ситуации (попытки форматирования дисков, прямой записи на диск, установку резидентных программ, изменение объектов файловой системы и т.д.). В случае появления опасной ситуации на экран выводится сообщение с просьбой подтвердить выполнение опасной операции. Идентификация вирусов стандартным монитором не проводится, хотя современные мониторы обычно обладают этой функцией..

Сканер (доктор) – программа, выполняющая следующие функции:

- поиск вирусов в заданном перечне объектов (каталогов, дисков и т.д.);
- идентификация (распознавание) вирусов;
- удаление вирусов.

Поиск и идентификация вирусов проводится на основе их сигнатур, под которыми понимаются участки программного кода, характерные для каждой конкретной вирусной программы. У каждого сканера имеется собственная библиотека сигнатур, регулярно обновляемая разработчиками, обычная периодичность обновления – еженедельная.

Для удаления вируса сканер выполняет все действия, сделанные вирусом при заражении, в обратном порядке. Если не удается удалить вирус, то необходимо удалить весь зараженный файл.

Таким образом сканер может обнаруживать только те вирусы, сигнатуры которых уже имеются в его библиотеке сигнатур. Современные сканеры имеют в своем составе блок эвристического анализа, позволяющий обнаруживать неизвестные вирусы. Принцип работы эвристического анализатора основан на анализе программного кода исполняемых файлов и эмуляции его исполнения, т.е. анализатор как будто начинает исполнять все команды исполняемого файла и , в случае возникновения потенциальной угрозы, выводит сообщение о том, что данный файл может быть заражен и несет опасность для компьютера (изменение файловой системы, встраивание модулей в другие процессы, скрытие процессов, изменение ключей системного реестра Microsoft Windows).

Проведение эвристического анализа ведет к дополнительным затратам времени, поэтому обычно в сканерах имеется возможность его отключения. В некоторых сканерах блок эвристического анализа называется блоком проактивной защиты.

В настоящее время производители обычно выпускают комплексы антивирусных программ. Наиболее часто используются комплексы Лаборатории Касперского (AVP или KAV) и Лаборатории Данилова (DrWEB). Например, в состав пакета KAV входят:

- центр управления (Control Center);
- монитор (Monitor);
- сканер (Scaner);
- ревизор (Inspector);
- программа проверки электронной почты (Mail Checker);
- монитор скриптов (Script Checker).

Для более лучшего усвоения материала по заданной теме выполняется лабораторная работа № 5.

6. Аппаратная часть ЭВМ

Состав и основные структуры ЭВМ: иерархическая и магистральная. Процессор: обобщенная структурная схема и взаимодействие основных узлов. Классификация команд процессора. Способы адресации в командах: прямая, косвенная, непосредственная, неявная. Архитектуры современных процессоров (CISC, RISC, EPIC). Архитектура IA-64. Классификация памяти ЭВМ: внешняя, оперативная, сверхоперативная. Доступ к памяти. Особенности различных видов памяти. Статическая и динамическая память: ROM, RAM, SRAM, DRAM. Виды динамической памяти: FPM, BEDO, SDRAM, DDR, RDRAM. Модули памяти SIMM, DIMM, SO DIMM. Логическая и физическая модели памяти. Интерфейсы современных ЭВМ: понятие, классификация, основные технические характеристики. Классификация и основные характеристики шин ПЭВМ (ISA, EISA, MCA, VESA, PCI). Периферийные устройства (мониторы, принтеры, сканеры)

6.1 Архитектура ЭВМ

Все ЭВМ, несмотря на большое количество их типов, принципиально состоят из небольшого набора функциональных устройств – процессоры, устройства памяти и устройства ввода-вывода. Один из процессоров является центральным (главным), именно он исполняет команды программ пользователей. Другие процессоры являются вспомогательными и используются в видеокарте, клавиатуре и других устройствах компьютера.

Под архитектурой вычислительной машины понимается совокупность ее функциональных устройств и способов их взаимодействия. В настоящее время используются, в основном, две архитектуры ЭВМ – иерархическая и магистральная.

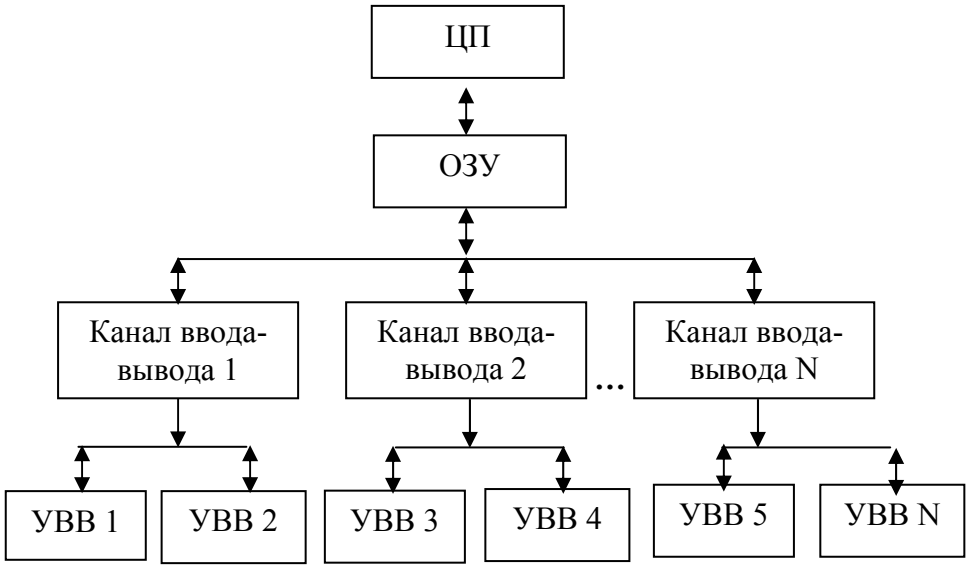


Рис. 6.1 Иерархическая структура ЭВМ

Иерархическая архитектура (рис. 6.1) используется в больших ЭВМ. Здесь обозначено: ЦП – центральный процессор, ОЗУ – оперативное запоминающее устройство, УВВ – устройство ввода-вывода.

Каналы ввода-вывода - это специализированные процессоры, работающие под управлением программ операционной системы. К каждому каналу может быть подключено несколько устройств ввода-вывода, каждое из которых имеет собственное устройство управления (контроллер). Особенностью иерархической архитектуры является то, что ЦП взаимодействует только с оперативной памятью.

Магистральная архитектура (рис. 6.2) используется в малых ЭВМ (рабочих станциях, серверах и персональных ЭВМ). В этом случае все функциональные устройства подключаются параллельно к одной числовой магистрали (общей шине), работающей в режиме разделения времени. В самом упрощенном варианте общую шину можно представить как набор проводов, по которым передаются команды, данные и сигналы управления.

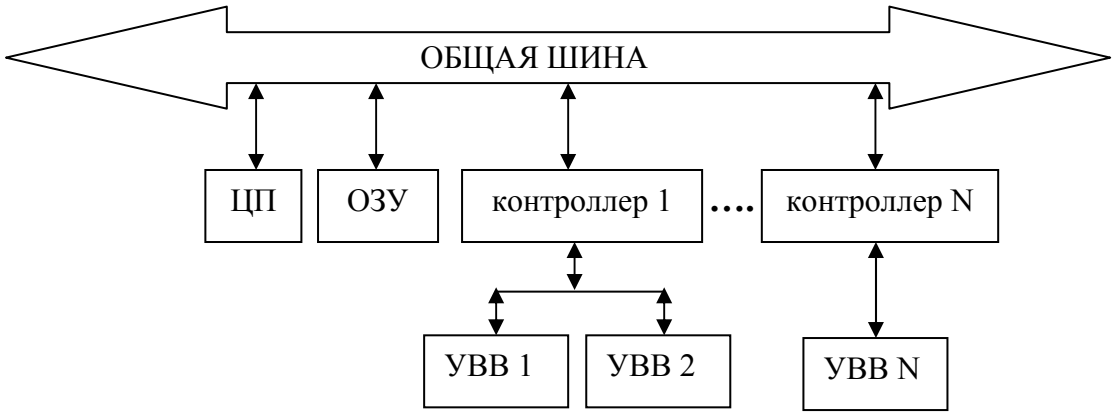


Рис. 6.2 Магистральная архитектура ЭВМ

Для исключения конфликтов между устройствами используется ряд определенных соглашений:

в каждый момент времени по шине обмениваются данными только два устройства, одним из которых должно быть ОЗУ или ЦП;

информация по шине передается машинными словами, размер которых определяется разрядностью ЦП;

каждое устройство, подключаемое к общей шине, имеет определенный приоритет, под которым понимается целое беззнаковое число, определяющее порядок

разрешения конфликтов; высший приоритет имеет самые быстродействующие устройства (ЦП и ОЗУ).

6.2 Структура персонального компьютера

Рассмотрим структурную схему персонального компьютера (рис. 6.3)

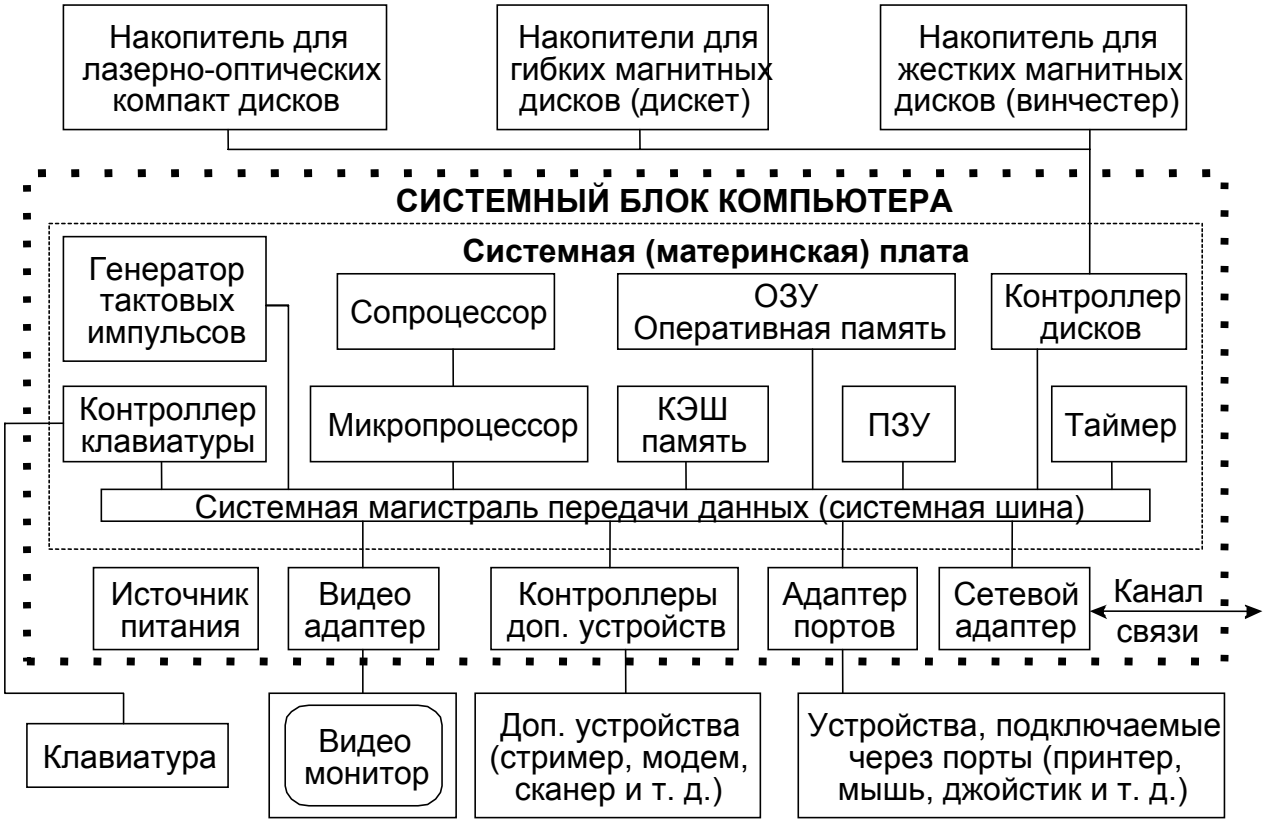


Рис. 6.3 Структурная схема персонального компьютера

Микропроцессор. Это центральный блок компьютера, предназначенный для управления работой всех блоков машины и для выполнения арифметических и логических операций над информацией. Микропроцессор производит сотни различных операций и делает это со скоростью в несколько десятков или даже сотен миллионов операций в секунду. В компьютерах типа IBM PC используются микропроцессоры фирмы Intel, а также совместимые с ними микропроцессоры других фирм (AMD, Cyrix и др.).

Сопроцессор. В тех случаях, когда на компьютере приходится выполнять много математических вычислений, к основному микропроцессору добавляют математический сопроцессор. Он помогает микропроцессору выполнять операции над ве-

щественными числами. В современных компьютерах сопроцессор входит в состав основного процессора.

Генератор тактовых импульсов. Он формирует последовательность электрических импульсов; частота генерируемых импульсов определяет тактовую частоту машины. Тактовая частота измеряется в мегагерцах (МГц).

Промежуток времени между соседними импульсами определяет время одного *такта работы машины*.

Системная шина. Для работы компьютера необходим обмен информацией между микропроцессором, оперативной памятью и внешними устройствами. Такой обмен называется вводом-выводом. *Шина* - основная интерфейсная система компьютера, обеспечивающая сопряжение и связь всех его устройств между собой. Она содержит провода и схемы сопряжения для параллельной передачи всех разрядов числового кода данных, инструкций, адресов ячеек памяти и портов ввода-вывода, а также для подключения всех блоков компьютера к системе электропитания.

Системная шина обеспечивает три направления передачи информации:

- между микропроцессором и основной памятью;
- между микропроцессором и портами ввода-вывода внешних устройств;
- между основной памятью и портами ввода-вывода внешних устройств.

Основная память. Она предназначена для хранения и обмена информацией с прочими блоками машины. Основная память содержит два вида электронных запоминающих устройств: постоянное запоминающее устройство (ПЗУ) и оперативное запоминающее устройство (ОЗУ) или *оперативная память*.

Постоянная память имеет собственное название, которое указывает на то, что ею обеспечиваются только режимы считывания и хранения.

ПЗУ служит для хранения неизменяемой (ROM - Read Only Memory) программной и справочной информации, позволяет оперативно только считывать хранящиеся в нем данные (изменить информацию в ПЗУ нельзя). Данные в ПЗУ сохраняются и при выключенном питании.

ОЗУ предназначено для оперативной записи, хранения и считывания информации (программ и данных), непосредственно участвующей в вычислительном

процессе, выполняемом компьютером. Название «оперативное» это устройство получило потому, что оно работает очень быстро и позволяет обращаться к произвольно выбранной ячейке памяти (RAM - Random Access Memory). Однако содержащиеся в нем данные сохраняются только пока компьютер включен, при выключении компьютера содержимое оперативной памяти стирается.

Кэш-память. Для быстрых компьютеров необходимо обеспечить быстрый доступ к оперативной памяти, иначе микропроцессор будет простаивать и быстродействие компьютера уменьшится. Для этого компьютеры оснащаются кэш-памятью, т. е. «сверхоперативной» электронной памятью относительно небольшого объема, в которой хранятся наиболее часто используемые участки оперативной памяти. Кэш-память располагается «между» микропроцессором и оперативной памятью, и при обращении микропроцессора к памяти сначала производится поиск нужных данных в кэш-памяти. Поскольку время доступа к кэш-памяти в несколько раз меньше, чем к обычной памяти, а в большинстве случаев необходимые микропроцессору данные содержатся в кэш-памяти, среднее время доступа к памяти уменьшается.

Внешняя память. Она используется для долговременного хранения любой информации, которая может потребоваться для решения задач. В частности, во внешней памяти хранится программное обеспечение компьютера.

Таймер. Это внутримашинные электронные часы, обеспечивающие при необходимости автоматический съем текущего момента времени (год, месяц, часы, минуты, секунды и доли секунд). Таймер подключается к автономному источнику питания – аккумулятору и при отключении машины от сети питания продолжает работать.

Контроллер прямого доступа к памяти. Это устройство освобождает микропроцессор от прямого управления накопителями на магнитных и оптических дисках, что существенно повышает быстродействие компьютера.

Контроллер прерываний. Он обслуживает процедуры прерывания, принимает запрос на прерывание от внешних устройств, определяет уровень приоритета этого запроса и выдает сигнал прерывания в микропроцессор. Микропроцессор, получив этот сигнал, приостанавливает выполнение текущей программы и переходит к вы-

полнению специальной программы обслуживания того прерывания, которое запросило внешнее устройство. После завершения программы обслуживания восстанавливается выполнение прерванной программы.

Адаптер портов ввода-вывода. Одним из контроллеров, которые присутствуют почти в каждом компьютере, является адаптер портов ввода-вывода.

6.3 Процессоры

6.3.1 Структура процессора

Структурная схема типового процессора показана на рис. 6.4.

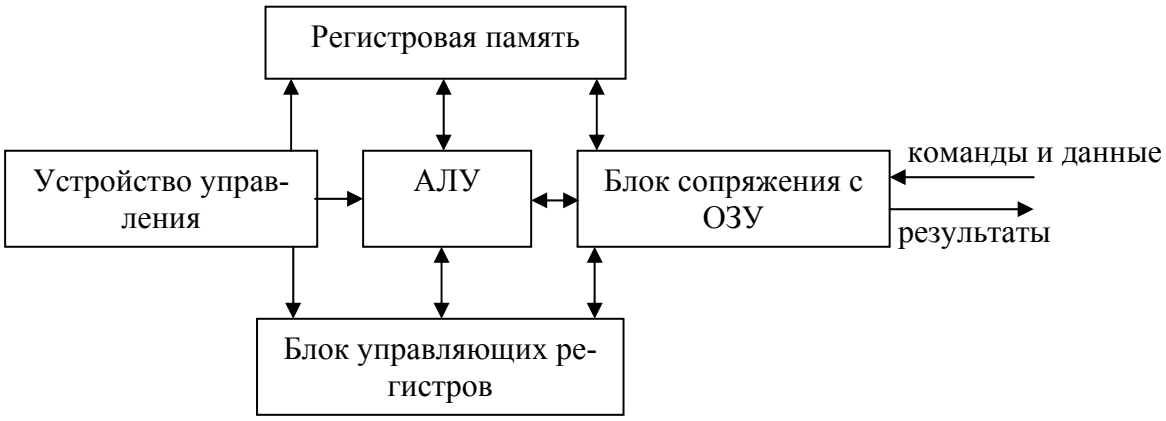


Рис. 6.4 Структура схема процессора

Здесь обозначено:

АЛУ (арифметико – логическое устройство) – блок, непосредственно выполняющий команды программы; может исполнять только арифметические и логические операции;

регистровая память – внутренняя память процессора, используемая для хранения команд и данных; представляет собой набор регистров, каждый из которых может хранить объем информации, равный одному машинному слову (32 или 64 бита);

блок управляющих регистров – набор регистров, используемых для хранения служебной информации, необходимой для организации вычислительного процесса;

устройство управления – предназначено для синхронизации работы всех компонентов процессора, одним из блоков устройства управления является тактовый генератор, задающий темп вычислительного процесса;

блок сопряжения с ОЗУ – предназначен для сопряжения компонентов процессора с оперативной памятью.

Команды и данные из ОЗУ считываются в регистровую память, АЛУ исполняет эти команды, а результаты выполнения возвращаются в ОЗУ по заданным адресам.

Современные процессоры имеют несколько АЛУ, выполняя одновременно несколько потоков команд. Такие процессоры называются суперскалярными.

6.3.2 Команды процессора

Единственное назначение процессора – это исполнение команд. Машинная команда состоит из кода операции и операндов, код операции указывает действие, а операнды задают имена объектов (или их значения), над которыми это действие производится. Например, в команде

СЛОЖИТЬ 10 5

код «СЛОЖИТЬ» является командой, а «10» и «5» являются операндами этой команды.

Каждое семейство процессоров имеет собственный набор команд, причем в качестве кода команды обычно используется целое беззнаковое число. Операндами команд могут быть адреса ячеек ОЗУ, номера регистров или значения констант.

Команды активной программы исполняются в естественном порядке, т.е. последовательно. Выборка команд из программы может проводиться двумя способами:

а) по номеру

$N_i = N_{i-1} + 1$, где N_i - номер следующей команды, N_{i-1} - номер предыдущей команды

б) по адресу

$A_i = A_{i-1} + L_{i-1}$, где A_i - адрес следующей команды, A_{i-1} - адрес предыдущей команды, L_{i-1} - длина предыдущей команды.

Существует два типа команд: распознаватели и преобразователи. Распознаватели изменяют естественный порядок выполнения команд и не влияют на состояние данных, преобразователи выполняют преобразование значений данных. При-

мерами распознавателей являются команды безусловного перехода (GO TO m1 – переход на заданную метку), и различных ветвлений (IF_THEN_ELSE, CASE).

Внутренний формат команды безусловного перехода показан на рис 6.5а, а команды ветвления – на рис.6.5б

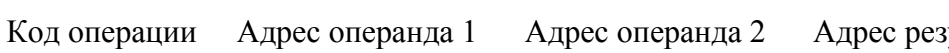
а) б)

Рис. 6.5

Здесь «условие» - логическое условие, при истинности которого выполняется переход на указанный «адрес».

Команды – преобразователи содержат информацию о коде операции и операндах и могут содержать несколько адресных полей. Примером двухадресной команды может служить команда инкрементного увеличения значения некоторой переменной a := inc(b) , а трехадресной команды – любая бинарная операция a := b+c.

В общем случае внутренний формат команды может быть записан в следующем виде



В наборе команд процессора могут быть и безадресные команды, например команда ОБНУЛИТЬ ВСЕ РЕГИСТРЫ.

Возможны несколько способов указания адреса операнда в адресном поле:

- прямая адресация;
- косвенная адресация;
- непосредственная адресация;
- неявная адресация.

При прямой адресации в адресном поле указывается адрес операнда, т.е. номер ячейки ОЗУ, в которой находится этот операнд. Недостаток этого способа состоит в том, что во время компиляции, когда формируются машинные команды, еще неизвестно, в каких физических адресах ОЗУ будет размещаться программа во время выполнения. Поэтому этот способ, несмотря на простоту, практически не используется в современных ЭВМ.

При косвенной адресации в адресном поле команды записывается адрес указателя на операнд, а указатель хранит адрес операнда (рис. 6.6). Для указания адреса операнда может использоваться несколько указателей, связанных в список, при этом количество указателей в цепочке называется кратностью косвенной адресации. Прямая адресация может рассматриваться как частный случай косвенной при кратности, равной единице.

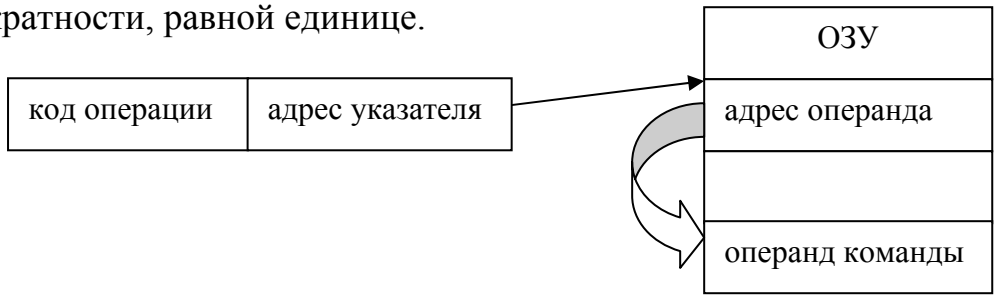


Рис. 6.6

Косвенная адресация требует нескольких обращений к памяти, т.е. замедляет работу программ по сравнению с прямой адресацией, но удобнее при использовании перемещаемых в памяти программ и сложных структур данных.

Непосредственная адресация используется, если в качестве одного из операндов используется константа. При этом значение константы записывается в адресное поле команды, а не в память, что исключает лишние обращения к ОЗУ.

При неявной адресации адрес операнда вообще не хранится в команде, а вычисляется по определенному алгоритму, или в качестве операнда используется содержимое определенного регистра процессора или содержимое участка памяти с фиксированным адресом.

6.3.3 Архитектуры процессоров

На рис. 6.7 приведена классификация процессоров по возможным архитектурам.

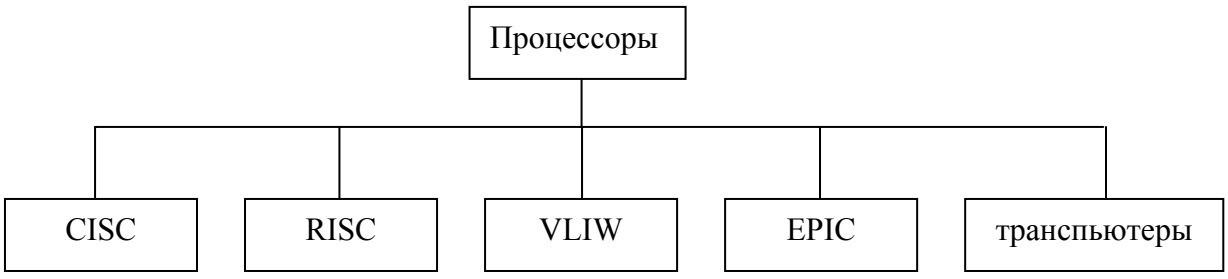


Рис.6.7 Архитектуры процессоров

CISC (Complete Instruction Set Computing), процессоры с полным набором команд) . Характеризуются следующими основными свойствами

- большое число команд (может достигать нескольких сотен);
- команды имеют различную длину;
- операнды команд могут располагаться в регистрах ЦП и в оперативной памяти;
- большинство команд требует для выполнения несколько тактов тактового генератора.

Недостатки – большая сложность процессора при малой производительности. Архитектуру CISC имели процессоры Intel 80486 и более ранние.

RISC (Reduce Instruction Set Computing), процессоры с сокращенным набором команд) . Основные свойства:

- небольшое число команд (несколько десятков);
- длина всех команд одинакова;
- операнды располагаются только в регистрах ЦП;
- все команды выполняются за один такт.

Процессоры с архитектурой RISC имеют производительность в 2 - 4 раза выше, чем с архитектурой CISC при равных тактовых частотах. Они имеют более высокую стоимость и используются в мощных серверах и рабочих станциях. Примерами RISC процессоров являются SPARC, Power PC, Alpha, MIPS.

VLIW (Very Long Instruction Word), процессоры с длинными командным словом. Используется в суперскалярных процессорах и характеризуется тем, что одна инструкция процессора содержит несколько команд, которые должны выполняться параллельно. Распределение работы между АЛУ решается во время компиляции и в инструкциях явно указано, какое вычислительное устройство должно выполнять какую команду. Упаковка команд в длинное машинное слово проводится также компилятором.

В процессоре команды выделяются и подаются каждая на свое АЛУ. Например, для процессора с двумя АЛУ для сложения четырех чисел, находящихся в регистрах R1, R2, R3 и R4, инструкции могут иметь следующий вид:

```
R5=R1+R2, R6=R3+R4 ; каждое АЛУ складывает свою пару чисел
```

```
R0=R5+R6, NOP          ; первое АЛУ находит сумму, второе простаивает
```

Первые VLIW-процессоры появились в конце 80-х и были разработаны компанией Cydrome. В чистом виде архитектуру VLIW имеют процессоры TriMedia фирмы Philips и семейство DSP C6000 фирмы Texas Instruments.

EPIC (Explicitly Parallel Instruction Computing), процессор с явным параллелизмом команд. Является дальнейшим развитием VLIW архитектуры и имеет следующие особенности:

- большое число регистров;
- упреждающая загрузка команд.
- явный параллелизм в машинном коде, причем значительную часть работы по организации параллелизма выполняет компилятор при подготовке программы;
- наличие режима предикации, который предполагает параллельное выполнение команд из разных ветвей условного оператора и вычисление значения логического условия; при этом каждый поток команд снабжается полями условия и результат выбирается из потока с соответствующим значением условия;

Концепция EPIC была разработана фирмами Intel и HP. Стандарт Intel Architecture -64 (IA-64) предусматривает наличие 64- разрядных регистров для выполнения операций с целыми числами (128 регистров) и 80 – разрядных регистров для выполнения операций с вещественными числами (128 регистров).

Примерами процессоров с архитектурой EPIC являются современные процессоры Pentium III, Pentium IV и более старшие версии.

Транспьютеры – микропроцессоры, предназначенные для работы в составе мультипроцессорных вычислительных систем с однотипными процессорами. Основная особенность транспьютеров – наличие высокоскоростных коммуникационных каналов связи, каждый из которых может одновременно передавать по одной магистрали данные в процессор, а по другой магистрали – данные из него. Т.е. здесь возможен прямой обмен данными между процессорами без использования ОЗУ. Производительность транспьютерной системы в 4-5 раз превышает производительность CISC системы при равных тактовых частотах.

6.4 Устройства памяти

6.4.1 Классификация памяти

Основными техническими характеристиками памяти являются объем и быстродействие (см. раздел 1.5.2). На рис.6.8 приведена классификация устройств памяти ЭВМ.

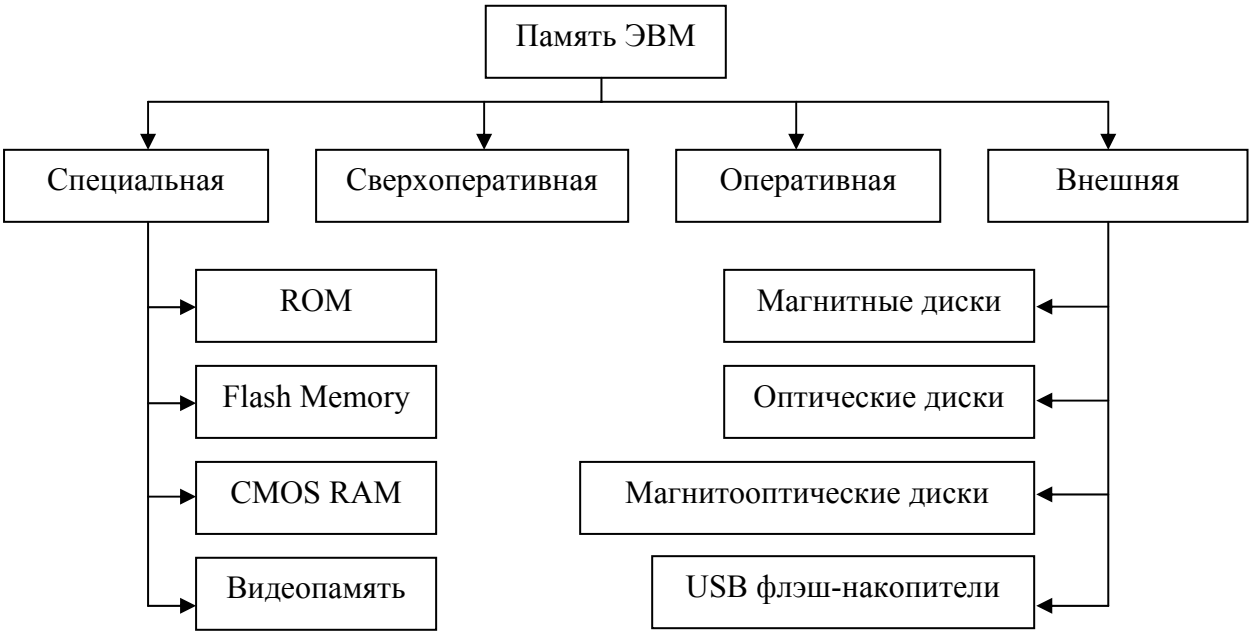


Рис. 6.8

Функции устройств памяти:

- прием информации от других устройств компьютера;
- хранение информации;
- выдача информации по запросу в другие устройства.

Оперативная память (RAM — Random Access Memory — память с произвольным доступом) — это быстродействующее запоминающее устройство с прямым доступом процессора, которое предназначено для записи, считывания и временного хранения выполняемых программ и данных. Обращение к ячейкам оперативной памяти проводится по адресу, все ячейки равнодоступны. Является энергозависимым устройством., т.е. при выключении компьютера все его содержимое пропадает. Объем обычно составляет 512—1024 Мб.

Сверхоперативная память (кэш –англ. *cache)* — очень быстрая память небольшого объема, которая используется в качестве промежуточной памяти при об-

мене данными между процессором и менее быстродействующей оперативной памятью для компенсации разницы в скорости обработки информации. Кэш-памятью управляет специальное устройство, которое, анализируя выполняемую программу, пытается предвидеть, какие данные и команды вероятнее всего понадобятся в ближайшее время процессору, и подкачивает их в кэш-память. Современные процессоры имеют встроенную кэш-память, так называемый кэш первого уровня размером до 512 Кб. На системной плате компьютера обычно устанавливается кэш второго уровня емкостью до 8 Мб.

Специальная память имеет следующие виды:

- постоянная память (ROM);
- память CMOS RAM, питаемая от батарейки;
- перепрограммируемая постоянная память *(Flash Memory);*
- видеопамять.

Постоянная память (ПЗУ, англ. ROM — *Read Only Memory* — память только для чтения) — энергонезависимая память, используемая хранения данных, которые никогда не потребуют изменения. Содержание памяти специальным образом «зашивается» в устройстве при его изготовлении. Из ПЗУ можно только читать однажды занесенные в нее данные. В ПЗУ находятся программы управления дисплеем, клавиатурой, принтером, внешней памятью, программы запуска и остановки компьютера, тестирования устройств. Важнейшая микросхема постоянной памяти — модуль BIOS (*Basic Input / Output System* — базовая система ввода-вывода). Это совокупность системных программ, предназначенных для автоматического тестирования устройств после включения питания компьютера и загрузки операционной системы в оперативную память.

Разновидность постоянной памяти — CMOS RAM. Это - память с невысоким быстродействием и минимальным энергопотреблением от батарейки. Она используется для хранения информации о конфигурации и составе оборудования компьютера, а также о режимах его работы. Содержимое CMOS изменяется специальной программой Setup, находящейся в BIOS.

Перепрограммируемая постоянная память *(Flash Memory)* — энергонезависимая память, допускающая многократную перезапись своего содержимого. Служит

для хранения обновляемых программ и данных в самых разных системах, включая сотовые телефоны, модемы, BIOS, системы управления автомобильными двигателями и многое другое.

Магнитные диски используют для хранения информации принцип магнитной записи. Информация на жесткий диск записывается по концентрическим дорожкам (трекам), которые делятся на секторы. Размер каждого сектора обычно составляет 512 байт, а количество секторов, записываемых на одну дорожку, зависит от физических размеров диска и плотности записи.

Данные записываются или считываются с пластин с помощью головок записи / считывания, по одной на каждую поверхность. Линейный двигатель представляет собой электро-механическое устройство, которое позиционирует головку над заданной дорожкой. Обычно головки крепятся на кронштейнах, которые приводятся в движение каретками. Цилиндр - это набор дорожек, соответствующих одному положению каретки. Накопитель на магнитных дисках (НМД) представляет собой набор пластин, магнитных головок, кареток, линейных двигателей плюс воздухонепроницаемый корпус.

Жесткий магнитный диск (ЖМД, англ. HDD — *Hard Disk Drive)* — это перезаписываемое запоминающее устройство большой емкости, в котором носителями информации являются тонкие алюминиевые диски диаметром около 3,5 дюймов, покрытые слоем магнитного материала. Пакет из нескольких таких дисков, закрепленных на общей оси, непрерывно вращается с большой скоростью. ЖМД используется для длительного хранения обновляемой информации (программ и данных) и обычно находится внутри системного блока компьютера. ЖМД имеют большой объем - более 150 Гб, среднее время поиска данных составляет 10 мс, максимальная скорость передачи данных до 40 Мб/с, скорость вращения до 10000 оборотов в минуту.

Гибкий магнитный диск или дискета (ГМД, англ. FDD - *floppy disk)* — съемное устройство для хранения небольших объемов информации, представляющее собой гибкий пластиковый диск диаметром около 3,5 дюймов, имеющий с обеих сторон магнитное покрытие. Диск помещен в пластмассовую кассету почти квадратной формы, которая вставляется в накопитель для ГМД. Накопитель вращает дискету

только при обращении к нему для чтения или записи данных. ГМД считаются малонадежным носителем информации, они используются для миграции данных небольших объемов с одного компьютера на другой и их резервного хранения. Дискеты имеют следующие характеристики: диаметр 3,5 дюйма (89 мм), емкость 1,44 Мб, скорость передачи данных - 100 Кб/с.

Оптические диски (компакт-диски) используют для хранения оптический принцип записи. Компакт-диск состоит из прозрачной полимерной основы диаметром 12 см и толщиной 1,2 мм. Одна сторона покрыта тонким алюминиевым слоем, защищенным от механических повреждений лаком. Принцип действия накопителей этого типа заключается во взаимодействии лазерного луча с основой диска и отражении его от противоположной поверхности. Информация на компакт-диске, в отличие от магнитного диска, размещается на единственной спиральной дорожке и организована специальным образом. Существует несколько вариантов оптических дисков и соответствующих им накопителей:

CD-ROM *(Compact Disk Read Only Memory* — компакт-диски только для чтения),

CD-R *(Compact Disk Recordable* — записываемые компакт-диски),

CD-RW *(Compact Disk Rewritable* — перезаписываемые компакт-диски);

DVD (*Digital Versatile Disc* - цифровой универсальный диск).

Информация, записанная на дисках CD-ROM и CD-R, не может быть изменена. Различие между ними в том, что первые «печатаются» массовым тиражом на фирмах-изготовителях, последние записываются индивидуально на компьютере пользователя. На диски CD-RW данные также записываются на компьютере пользователя, но могут быть впоследствии изменены. Количество перезаписей, которое выдерживает CD-RW, довольно велико - до 1000 раз.

Достоинства компакт-дисков:

- обладают достаточно большим объемом (650-700 Мб);
- удобны в работе, практически не изнашиваются;
- низкая удельная стоимость хранения данных.

- невозможно случайно стереть информацию с дисков CD-ROM и CD-R; изменить содержание дисков CD-RW и DVD можно только на специальных накопителях;

Скорость работы накопителей всех типов обычно измеряется в единицах относительно скорости аудио-CD. В настоящее время распространены накопители с 40—50-кратной скоростью чтения, которая составляет 2—4 Мб/с. Скорость записи существенно ниже и зависит от типов накопителя и носителя данных: от 10 до 20-кратной.

Емкость дисков DVD, в зависимости от количества используемых сторон и слоев, может варьироваться от 4,7 Gb до 17 Gb.

Например:

- Single Side/Single Layer (односторонний/однослойный) Это самая простая структура DVD диска. На таком диске можно разместить до 4,7 Гб данных.
- Single Side/Dual Layer (односторонний/двуслойный) Этот тип дисков имеет два слоя данных, один из которых полупрозрачный. Оба слоя считываются с одной стороны и на таком диске можно разместить 8,5 Гб.
- Double Side/Single Layer (двусторонний/однослойный) На таком диске помещается 9,4 Гб данных (по 4,7 Гб на каждой стороне).
- Double Side/Double Layer (двусторонний/двуслойный) Структура этого диска обеспечивает возможность разместить на нем до 17 Гб данных (по 8,5 Гб на каждой стороне).

Магнитооптические диски используют комбинацию магнитного и оптического методов записи информации. Запись на магнитооптические диски выполняется при взаимодействии лазера и магнитной головки. Луч лазера разогревает до точки Кюри (температуры потери материалом магнитных свойств) микроскопическую область записывающего слоя, которая при выходе из зоны действия лазера остывает, фиксируя магнитное поле, наведенное магнитной головкой. В результате данные, записанные на диск, не боятся сильных магнитных полей и колебаний температуры. Все функциональные свойства дисков сохраняются в диапазоне температур от -20 до +50 градусов Цельсия.

Время доступа к данным у магнитооптических дисков составляет около 20 мс, объем – более 1 Гб, скорость передачи данных - до 10 Мбайт/с.

USB-флэш-накопители – это портативный внешний накопитель информации размером с зажигалку и весом порядка 15 г, который не требует для его использования установки на компьютер дополнительных специальных устройств. Он подключается к компьютеру через USB-порт, поэтому не требует дисковода в отличие других сменных носителей. Кроме того, USB-интерфейс позволяет подключать устройство без необходимости выключения питания и перезагрузки компьютера. Объем флэш-накопителей составляет до 4 Гб. Информация может быть защищена от случайного стирания с помощью механического переключателя на корпусе. Флэш-накопитель безразличен к магнитным полям, встряскам, вибрациям, несильным ударам.

При подключении к USB-порту компьютер распознает его как новое устройство и открывает для него новый диск. Для подключения может потребоваться установка драйвера, который обычно входит в комплект поставки или может быть загружен из Internet. Благодаря специальным утилитам возможна парольная защита данных на этих накопителях, а также и загрузка компьютера с их использованием.

6.4.2 Оперативная память

Оперативная память (RAM) является основным системным ресурсом ЭВМ и прошла в своем развитии большую историю. На рис. 6.9 показана одна из возможных классификаций памяти персональных компьютеров, в основу которой положен исторический путь ее развития.

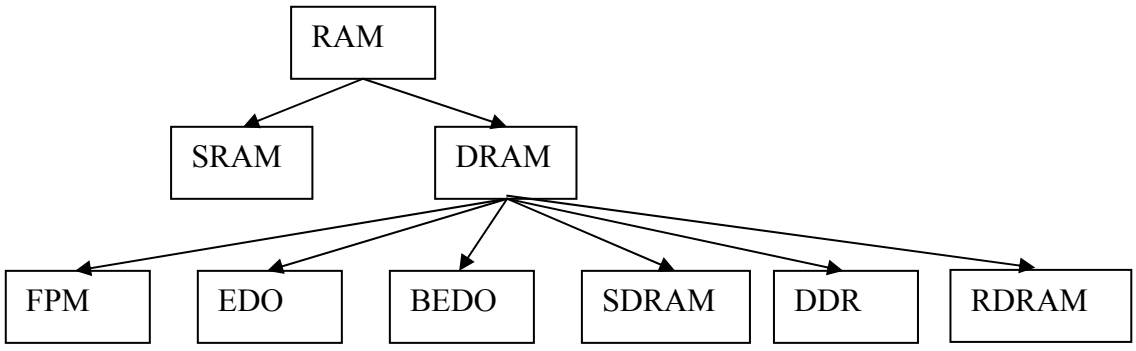


Рис. 6.9

Память бывает двух видов – статическая и динамическая. Статическая память (SRAM) хранит записанную информацию в течение всего времени включенного состояния, т.е. если некоторый байт содержит значение 10000001, то это значение будет храниться до тех пор, пока этот байт не будет перезаписан или не будет выключен компьютер. Динамическая память (DRAM) требует периодического обновления содержимого из-за того, что ее физическая реализация основана на использовании емкостных накопителей (конденсаторов), которые с течением времени теряют заряд и требуют подзарядки. Частота обновления данных в DRAM называется частотой регенерации (refresh rate).

Статическая память работает существенно быстрее, чем динамическая, но имеет более высокую стоимость, поэтому SRAM обычно используется в качестве кэш-памяти 2-го уровня, а DRAM применяется в качестве основной оперативной памяти компьютера.

Основными направлениями развития технологии оперативной памяти являются:

- увеличение объема отдельных микросхем и модулей памяти в целом;
- увеличение ее пропускной способности (напрямую зависящей от ее тактовой частоты);
- снижение задержек;
- уменьшение энергопотребления.

Исторически первым видом памяти, используемой в персональных компьютерах, была память FPM (Fast Page Mode), которая позволяла проводить выборку данных за три такта тактового генератора. Затем появились EDO (Extended Data Out) и BEDO (Burst Extended Data Out – улучшенное EDO), проводившие выборку за два такта. У синхронной памяти (SDRAM – Synchronous DRAM) данные доступны во время каждого такта, а память DDR (Double Data Rate) позволяет читать данные два раза за один так, т.е. при частоте шины данных модуля памяти 200 Мгц «эффективная» частота работы памяти будет равна 400 Мгц..

Память DDR прошла в своем развитии три поколения – DDR («эффективная» частота до 400 Мгц, напряжение питания 2,5 В), DDR2 («эффективная» частота до

1250 Мгц, напряжение питания 1,8 В) и DDR3 («эффективная» частота до 1866 Мгц, напряжение питания 1,5 В).

Альтернативой памяти DDR является память RDRAM, реализующая технологию обмена данными, предложенную фирмой RumBus (???) в середине 90-х годов. Для работы с RDRAM требуются специальные материнские платы.

Микросхемы памяти объединяются в модули памяти, которые могут быть выполнены в виде SIPP (Single In-line Pin Package), SIMM (Single In-line Memory Module), DIMM (Dual In-line Memory Module), SO DIMM (Small Outline DIMM) и RIMM (RumBus Memory Module). Наиболее употребительны сегодня модули DIMM и RIMM для настольных компьютеров, SO DIMM – для ноутбуков. Выводы (контакты) модулей памяти могут быть позолочены или с оловянным покрытием в зависимости от материала, из которого выполнен слот для памяти. Для лучшей совместимости следует стремиться использовать модули памяти и слоты с покрытием из одинакового материала.

Существует две разновидности модулей SIMM: 30-контактные или 72-контактные, в зависимости от числа выводов модуля. 30-контактные модули имеют ширину 9 бит (8 бит и бит контроля четности), а 72-контактные модули - ширину 32 бита (без контроля четности) или 36 бит (с контролем четности). Для 32-разрядной шины процессора необходимо использовать либо четыре 30-контактных модуля SIMM или один 72-контактный модуль. Системы на базе процессоров Pentium имеют 64-битную шину, что требует использования двух 72-контактных модулей SIMM или одного модуля DIMM, который имеет ширину 64 бита и 168 контактов. Необходимое число модулей памяти для заполнения шины называется «банком» памяти.

6.4.3 Порты

Внешние устройства подключаются к компьютеру через специальные интерфейсы - порты, которые обеспечивают согласование параметров и учет особенностей внешнего устройства. Основным параметром порта является максимальная скорость передачи данных, измеряемая в бит/сек, эта единица скорости называется также «бод».

По способу передачи данных порты делятся на последовательные и параллельные. При последовательной передаче устройства соединяются двухпроводной линией, по которой поочередно передаются все биты каждого байта данных (рис. 6.10). Для параллельной передачи устройства соединяются набором проводов, по которым одновременно передается один или несколько байтов данных (рис. 6.11).

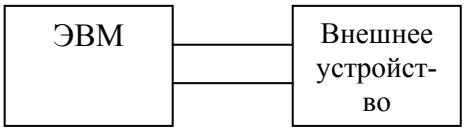


Рис.6.10

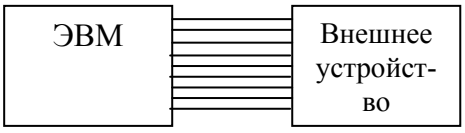


Рис.6.11

Достоинством параллельного способа передачи является высокая скорость обмена данными, а недостатками – более высокая конструктивная сложность и трудность обеспечения синхронного получения данных на приемном конце при высоких скоростях обмена. Параллельный порт (LPT) обычно используется для подключения к компьютеру быстрых устройств (принтер, сканер).

Последовательные порты могут быть нескольких типов – COM, PS/2, IEEE1394 и USB. COM-порт применяется для подключения медленных устройств (модем, мышь, технологическое оборудование). Он не обеспечивает гальванической развязки устройств, поэтому подключение устройств с независимым питанием должно проводиться только при отключенном питании компьютера. Недостаток – плохая помехозащищенность и , соответственно, короткие линии связи. Максимальная скорость передачи – 20 Кбит/сек, расстояние – до 15 м.

Порт PS/2 предназначен для подключения мыши, порт IEEE1394 (FireWare, iLink, скорость до 50 Мбит/с) - для подключения цифровых видеокамер и других мультимедийных устройств).

Стандарт USB был разработан в середине 90-х годов группой компаний IBM, DEC, Intel, Microsoft, NEC и Northern Telecom. Изначально в стандарт USB было заложено несколько удачных принципов. Во-первых, универсальный интерфейс, по которому может быть подключено любое из устройств: принтер, модем, клавиатура, мышь, колонки или целая аудиосистема, флэш-накопитель, а в последнее время еще и CD-ROM или жесткий диск. Определенные классы подключаемых устройств могут получать питание от встроенного в USB источника питания 5 В небольшой мощности. Существует также и отдельная субкультура устройств категории "USB

fun devices", вроде вентиляторов или зубных щеток, использующих USB только в качестве источника питания.

Второе "врожденное" свойство USB - это горячее подключение, не требующее установки дополнительных плат, отдельных драйверов и даже выключения компьютера. Это свойство, при наличии поддержки со стороны операционной системы, значительно повышает ценность интерфейса, в частности для мобильных пользователей.

Третье важное отличие - возможность каскадного подключения: некоторые устройства могут выступать как узлы (хабы), к которым можно подключить дополнительные устройства. Таким образом мышь можно присоединить к рассчитанной на это клавиатуре или, скажем, фотокамеру к принтеру, способному выполнять функции хаба. Обычно в качестве хабов выступают отдельные устройства типа "USB-тройник". Это использование имеет определенное ограничение: один USB-порт имеет совершенно небольшой запас по мощности встроенного источника питания, так что запитать от одного порта больше одного "пассивного" устройства без дополнительного питания будет проблематично.

Стандарт USB 2.0 решает проблему пропускной способности, устанавливая максимальную скорость до 60 Мбит/с. При каскадном подключении устройств пропускная способность порта распределяется между всеми подключенными устройствами, поэтому максимальную скорость можно получить лишь при подключении только одного устройства.

По способу соединения устройств порты бывают проводные и беспроводные. К беспроводным относятся инфракрасный порт (IrDA) и Bluetooth.- соединение. IrDA технология заключается в том, что она преобразует информацию в свет и передает ее от одного компонента устройства к другому. Недостаток IrDA заключается в том, что данная технология работает по принципу "точка-точка", то есть оба устройства, передающее и принимающее, должны находиться в прямой видимости друг друга.

Bluetooth – это многоточечный радиоканал, управляемый многоуровневым протоколом, аналогичным протоколу сотовой связи GSM. Частотный диапазон работы системы - 2,44 ГГц. Это можно считать одним из недостатков Bluetooth , по-

тому что на этой частоте работает большинство современных беспроводных систем ПНМ - промышленного, научного и медицинского назначения. Дело в том, что в случае возникновения так называемого "частотного конфликта" между каналом Bluetooth и каналом ПНМ сразу в обоих каналах могут возникнуть серьезные помехи. Принцип работы Bluetooth - скачкообразная перестройка частоты с расширением спектра, когда по псевдослучайному алгоритму радиопередатчик перескакивает с одной рабочей частоты на другую. Дальность действия Bluetooth – до нескольких десятков метров.

6.4.4 Шины

Шина в упрощенном варианте представляет собой набор проводов (линий), соединяющий различные компоненты компьютера для подвода к ним питания и обмена данными. В "минимальной комплектации" шина имеет три типа линий:

- линии управления;
- линии адресации;
- линии данных.

Устройства, подключенные к шине, делятся на две основных категории – bus masters и bus slaves. Первая категория - это устройства, способные управлять работой шины, т.е инициировать запись/чтение и т.д, вторая - устройства, которые могут только отвечать на запросы.

Первой шиной, используемой в персональных компьютерах, была шина разработанная фирмой IBM, которая работала на частоте 4,77 МГц , обеспечивала пропускную способность 1,2 Мбайт/сек, имела 8 разрядов для передачи данных и 20 разрядов для передачи адресов (адресное пространство - 1 МБайт). В настоящее время основными типами шин являются шины PCI и PCI-E.

Шина PCI (Peripheral Component Interconnect) разработана фирмой Intel в начале 90-х годов. Основные особенности этой шины следующие:

не зависит от архитектуры процессора;

используется синхронный 32-х или 64-х разрядный обмен данными;

для уменьшения числа линий в шине используется мультиплексирование, то есть адрес и данные передаются по одним и тем же линиям;

частота работы шины 33MHz или 66MHz (в версии 2.1) позволяет обеспечить широкий диапазон пропускных способностей (132 MB/сек при 32-bit/33MHz, 264 MB/сек при 32-bit/66MHz, 264 MB/сек при 64-bit/33MHz, 528 MB/сек при 64-bit/66MHz);

на шине PCI базируется технология автоматического конфигурирования устройств “Plag and Play”, предложенная фирмами Microsoft и Intel.

На базе шины PCI были разработаны шины AGP, PCI-X, mini-PCI. Шина AGP предназначена для использования с производительными графическими адаптерами, PCI-X - это ни что иное, как ускоренная до 133 МГц шина PCI 2.2 с обязательно 64-битной разрядностью интерфейса. PCI-X выпускалась и в более форсированных вариантах, с 266 МГц и 533 МГц тактовой частотой для использования в серверах. Менее заметна, но не менее важна шина mini-PCI, применяющаяся в портативных компьютерах для подключения различной «мелкой» периферии.

Шина PCI-E (PCI Express) появилась в 2002 году. Основная особенность этой шины – последовательная передача данных. PCI Express построен на принципах симплексной технологии, а это означает, что сигналы идут одновременно, в противоположных направлениях и по отдельным парам проводов - итого две пары, называемые линией. Стандарт декларирует пропускную способность симплексной линии на отметке 2,5 Гбит/с в одну сторону или, соответственно, 5 Гбит/с в обе стороны.

Единственное соединение, представляющее собой линию PCI Express, две пары проводов, - этого не достаточно для обеспечения высокой пропускной способности. Поэтому линии привычно выстраивают в ряд - их может быть 32, 16, 12, 8, 4 и 2. В итоге, вся последовательность данных, которую необходимо передать, распределяется на все имеющиеся линии «веером»- передача параллельная, но не синхронная. Если имеется 12 линий, то первый байт блока данных передается по первой линии, второй - по второй, и т. д., а тринадцатый байт - снова по первой. Теоретически шина с 32 линиями способна выдать пропускную способность 20 Гбит/с, от которых отнимаем 20% - 16 Гбит/с, или же по 8 Гбит/с в каждую сторону.

6.4.5 Мониторы

Монитор — устройство визуального отображения информации, представляемой в виде текста, таблиц, рисунков, чертежей, анимации и др. Основными потребительскими параметрами монитора являются размер экрана, частота кадровой развертки, способ вывода информации, разрешающая способность, шаг маски, , класс защиты.

Размер экрана измеряется по его диагонали в дюймах, возможные значения размеров от 15 до 29 дюймов. Частота, с которой меняются кадры изображения, называется частотой кадровой развертки; она не должна быть ниже 75 Гц, иначе глаз воспринимает изображение мерцающим. Мониторы особо высокого качества могут обеспечивать частоту развертки выше 100 Гц.

По способу вывода мониторы делятся на две группы – с электронно - лучевой трубкой (CRT) и жидкокристаллические (LCD). Принцип формирования изображения в CRT-мониторах аналогичен работе обычного цветного телевизора. Основной элемент монитора - электронно-лучевая трубка, передняя часть которой с внутренней стороны покрыта люминофором - веществом, способным излучать свет при попадании на него пучка электронов, испускаемого электронной пушкой.

Люминофор наносится в виде наборов RGB-точек трех основных цветов — красного (R — *red),* зеленого(O — *green)* и синего (B - *blue).* Эти цвета называют основными, потому что сочетаниями различных пропорций их яркости можно представить любой цвет спектра. Наборы точек люминофора располагаются по треугольным триадам. Триада образует *пиксел* — точку, из множества которых формируется изображение (англ, *pixel — picture element,* элемент картины). Расстояние между пикселами называется шагом маски дисплея. Это расстояние существенно влияет на четкость изображения; чем меньше шаг, тем выше четкость. В качественных мониторах шаг маски не должен превышать 0,26 мм. Количество точек на экране называется разрешающей способностью монитора или графическим разрешением.

Формирование изображения в LCD монитора проводится на основе свойств жидких кристаллов. Жидкие кристаллы — это особое состояние некоторых органических веществ, изменяющих структуру и светооптические свойства под действием

электрического напряжения. Большинство LCD дисплеев использует тонкую пленку из матрицы жидких кристаллов, помещенную между двумя стеклянными пластинами с электродами.

LCD мониторы занимают в 2—3 раза меньше места, чем CRT мониторы, потребляют гораздо меньше электроэнергии и не излучают электромагнитных волн, воздействующих на здоровье людей. Их недостатком является сравнительно невысокая надежность элементов матрицы, вышедшие из строя элементы остаются на экране как белые или черные точки.

Класс защиты дисплея определяется стандартом, которому соответствует дисплей. В настоящее время общепризнанными являются стандарты MPR-II, TCO-95, TCO-99.

Мониторы могут работать в одном из двух режимов: текстовом или графическом. В текстовом режиме экран монитора условно разбивается на отдельные участки - знакоместа, чаще всего на 25 строк по 80 символов (знакомест). В каждое знакоместо может быть введен один из 256 символов. В число этих символов входят большие и малые латинские буквы, символы кириллицы, цифры, знаки, а также псевдографические символы, используемые для вывода на экран таблиц и диаграмм, построения рамок вокруг участков экрана и т. д. На цветных мониторах каждому знакоместу может соответствовать свой цвет символа и фона.

В графическом режиме экран состоит из точек – пикселов, каждая из которых может быть темной или светлой на монохромных мониторах и одного или нескольких цветов - на цветном.

Графический режим предназначен для вывода на экран графиков, рисунков и так далее. Разумеется, в этом режиме можно выводить и текстовую информацию в виде различных надписей, причем эти надписи могут иметь произвольный шрифт, размер, цвет и др.

Работой монитора управляет специальное устройство – видеоадаптер, который может быть встроен в материнскую плату или может быть внешним, подключаемым по шине PCI, AGP или PCI-E. Наиболее распространенный видеоадаптер на сегодняшний день — адаптер SVGA (Super Video Graphics Array). Видеоадаптер способен воспроизводить любой из 65 миллионов цветов для каждого элемента

изображения. Основными параметрами видеоадаптера является поддерживаемое им разрешение и объем видеопамяти. Разрешение измеряется количеством элементов изображения, которое умещается на экране по горизонтали и по вертикали. Адаптер SVGA обеспечивает стандартные значения разрешения 640x480, 800x600, 1024x768, 1280x1024 и т.д. Объем памяти современных видеоадаптеров может достигать 512 МБайт.

При высоком экранном разрешении можно добиться очень высокой плотности информации в единице площади, но будет ли она при этом «читаемой», зависит от размера экрана.

Формула, определяющая ограничения на разрешение экрана и количество воспроизводимых цветов, очень проста: *(объем видеопамяти в байтах)* должен быть не меньше, чем (*число видимых точек в строке*) * (*число видимых строк на экране*) * (*число байт на точку*).

Например, если вы хотите выбрать разрешение экрана 1024 * 768 и иметь 16 миллионов цветов для каждой точки (4 байта на точку), то вам надо иметь 1024 * 768 * 4 = 3145728 байт памяти. Если же у вас всего 2 Мбайт памяти, то придется либо выбрать меньшее разрешение, либо меньшее количество цветов.

Для LCD мониторов основными характеристиками являются разрешающая способность, инерционность (2 – 10 мсек), яркость (до 255 кандел/кв. м), контрастность (от 500:1 до 3000:1), соотношение сторон (4:3 или 16:9) и угол обзора.

Наряду с традиционными видеоадаптерами широко используются разнообразные устройства компьютерной обработки видеосигналов:

- графические акселераторы (ускорители) — специализированные графические сопроцессоры, которые освобождают центральный процессор от большого объема операций с видеоданными;
- фрейм-грабберы, которые позволяют отображать на экране компьютера видеосигнал от видеомагнитофона, видеокамеры и т. п. с тем, чтобы скопировать нужный кадр в память и впоследствии сохранить его в виде файла;
- TV-тюнеры — видеоплаты, превращающие компьютер в телевизор; TV-тюнер позволяет выбрать любой канал телетрансляции и отображать видеосигнал на экране в масштабируемом окне со звуковым сопровождением.

6.4.6 Принтеры

Принтеры можно классифицировать по пяти основным позициям: принципу работы печатающего механизма, максимальному формату листа бумаги, использованию цветной печати, разрешающей способности а также по скорости печати.

По принципу печати различаются матричные, струйные и лазерные принтеры. Существуют другие технологии печати, например, сублимационная печать, светодиодная и другие, которые применяются гораздо реже.

Формат листов бумаги у наиболее распространенных моделей принтеров А3 и А4.

По гамме воспроизводимых цветов принтеры делятся на черно-белые, черно-белые с возможностью цветной печати (такие модели есть среди матричных и струйных принтеров) и цветные. Принтеры с возможностью цветной печати, как правило, плохо воспроизводят страницы, на которых цветная графика соседствует с черным фоном. Этот фон получается путем смешения чернил нескольких основных цветов. В итоге черный цвет оказывается недостаточно насыщенным, а стоимость печати такой страницы - весьма высокой.

Скорость печати современных принтеров обычно находится в диапазоне от 12 до 30 стр./мин. Скорость при цветной печати, как правило, значительно ниже, чем при печати одним черным цветом.

Матричный принтер использует комбинацию маленьких штырьков (иголок), которые бьют по красящей ленте, благодаря чему на бумаге остается отпечаток символа. Достоинства таких принтеров - способностью работать с любой бумагой, а также низкая стоимость печати. Недостаток – повышенная шумность в работе и низкая разрешающая способность — около 100 точек на дюйм.

Существуют 3 вида матричных принтеров: 9-, 18- и 24-игольчатые. С увеличением количества иголок возрастает качество печати.

К параметру «скорость печати» надо относиться осторожно, так как изготовители указывают максимальную скорость печати в черновом режиме. Высококачественная печать для игольчатых принтеров длиться значительно дольше. Еще медленнее осуществляется печать графики потому, что при этом положение каждой

печатаемой точки рассчитывается, в отличие от считывания знаков из внутренней памяти принтера при алфавитно-цифровой печати.

Струйный принтер создает символы в виде последовательности чернильных точек. Печатающая головка принтера имеет крошечные сопла, через которые на страницу выбрызгиваются быстросохнущие чернила. Эти принтеры требовательны к качеству бумаги, их чернила со временем выцветают.

Первой технологией, сделавшей струйную печать доступной и относительно дешевой, была технология *сухих чернил* (Dry Ink Jet). Под воздействием высокой температуры частицы твердого красителя (чаще всего графита) расплавлялись и под давлением наносились на бумагу. Современное развитие этого метода получило название сублимационной печати.

Другая разновидность струйной печати – *спарк-технология* - в целом аналогична предыдущей, но использует жидкие чернила.

Два других типа струйной печати представляют ее современное лицо. Это *пьезоэлектрическая* и *пузырьковая* технологии.

Первая использует пьезоэлектрический эффект для нанесения чернил на бумагу или пленку. Это позволяет очень точно позиционировать частицы красителя, однако требует сложного и дорогого устройства печати - *картриджа*.

Пузырьковая технология (Bubble Ink Jet Printing) осуществляет нанесение красителя путем выталкивания частиц чернил из емкости при помощи пузырька газа, образующегося внутри картриджа в результате резкого локального повышения температуры и давления. Эта технология позволяет реализовать печатающий узел устройства в виде дешевого съемного картриджа, она неприхотлива к качеству используемых чернил. И главное – пузырьковая технология обладает свойством «масштабируемости». Иными словами, увеличение истинного разрешения печати для технологии Bubble Ink Jet есть проблема технологическая, но не принципиальная.

Качество струйной печати зависит, главным образом, от трех основных факторов: качества печатающего узла (разрешение), качества чернил (передача полутонов и цвета), типа используемого носителя (насколько хорошо данные чернила сочетаются с данным типом бумаги или пленки).

Струйные принтеры практически бесшумны, с легкостью осуществляют цветную печать высокого разрешения (до 1200 точек на дюйм). При печати высокого качества скорость вывода достигает 20 стр./мин.

Лазерный принтер. Принцип работы основан на том, что информация о странице проецируется с помощью лазерного луча на красящий вращающийся барабан, с которого красящий порошок — тонер переносится на бумагу и «вплавляется» в нее, оставляя стойкое высококачественное изображение с разрешением до 1200 точек на дюйм.

Качество изображения, получаемого с помощью цветного лазерного принтера, приближается к фотографическому, однако цена такого принтера достаточно высока. Для получения высококачественной черно-белой распечатки следует отдать предпочтение лазерному принтеру по сравнению со струйным. Если вы желаете получить цветное изображение, то в большинстве случаев можете быть удовлетворены цветным струйным принтером.

Уровень шума при работе лазерного принтера составляет в среднем 40 дБ.

Скорость печати лазерного принтера определяется двумя факторами. Первый - время механической протяжки бумаги, второй - скорость обработки данных, поступающих от компьютера, и формирования растровой страницы для печати.

Обычно лазерный принтер оборудован собственным процессором. Для черно-белых принтеров часто используется микропроцессор Motorola 680000. В высокопроизводительных принтерах, например HP, используют процессор Intel 80960, имеющий тактовую частоту 33 МГц и сокращенный набор команд (RISC-архитектура).

Лазерный принтер является страничным принтером (формирует для печати полную страницу, а не отдельные строки, как игольчатый или струйный). Современные лазерные принтеры могут печатать до 20 и более страниц в минуту.

Скорость печати определяется не только быстродействием процессора, но и существенно зависит от памяти, которой оборудован принтер. Объем памяти лазерных принтеров достигает 160 Мбайт (для цветных принтеров).

Разрешение лазерного принтера по горизонтали и по вертикали определяется различными факторами. Вертикальное разрешение соответствует шагу барабана и

для большинства принтеров составляет 1/600 дюйма (для дешевых 1/300 дюйма). Горизонтальное разрешение определяется числом точек в одной «строке» и ограничено точностью наведения лазерного луча.

Как правило, большинство лазерных принтеров могут печатать на бумаге формата А4 и меньше, правда, в последнее время появились принтеры, способные печатать на листах формата А3 и на рулонной бумаге.

Некоторые лазерные принтеры могут печатать на обеих сторонах листа, а во многих дорогих моделях предусмотрена возможность их дооборудования для двусторонней печати.

6.4.7 Сканеры

Сканер - это устройство, позволяющее вводить в компьютер изображения, представленные в виде текстов, рисунков, слайдов, фотографий, штрих-кода и отпечатков пальцев и другой графической информации. Он создает оцифрованное цветное изображение документа и помещает его в память компьютера.

В сканированных штриховых изображениях каждому пикселу соответствует 1 бит - черный или белый. Шкала полутонов использует 8 бит на один пиксел. При воспроизведении оттенков цветовой гаммы используются три уровня 8-разрядного сканирования (по одному уровню на красную, зеленую и синюю составляющие цвета) для создания 24-разрядных изображений.

Для связи с компьютером сканеры используют 8- или 16-разрядную интерфейсную плату. Широко используются стандартные интерфейсы (последовательный, параллельный и USB порты, а также интерфейс SCSI).

Сканер достаточно полно можно охарактеризовать тремя параметрами: это оптическая разрешающая способность, глубина цвета и диапазон оптических плотностей.

В *планшетном* цветном сканере над сканируемым изображением перемещается флуоресцентная лампа. Свет лампы отражается от сканируемого документа, затем проходит через линзу и фокусируется на линейке *приборов с зарядовой связью* (ПЗС), которая воспринимает изображение.

Элементы ПЗС с фильтрами красного, зеленого и синего цветов в однопроходных сканерах считывают соответствующие цветовые составляющие изображе-

ния. В трехпроходных сканерах элементы ПЗС в каждом проходе фильтруют разный цвет. Однопроходные сканеры быстрее, а трехпроходные позволяют добиться большей точности. Истинное оптическое разрешение сканера, как и качество сканированного изображения, прямо пропорционально числу элементов ПЗС-матрицы сканера.

В качестве единицы измерения разрешения используется показатель количества точек, которые сканер в состоянии воспринять на одном дюйме оригинала – dpi (dot per inch). В спецификации часто указывается два параметра - горизонтальное разрешение и вертикальное, например 600×1200 dpi.

В данном случае имеется в виду не квадратный дюйм, а линейный. Сканер с указанным в спецификации оптическим разрешением 600×1200 при установке в режим «600 dpi» способен выдать изображение оригинала, в котором на каждом дюйме по горизонтали и вертикали будет размещено 600 цветовых точек. При установке большего, чем 600 dpi разрешения, драйвер сканера, используя методы математического увеличения разрешения, выравнивает горизонтальное и вертикальное разрешение. Следует иметь в виду, что именно показатель горизонтального разрешения характеризует оптические возможности сканера. Вертикальный же параметр - не более чем показатель прецизионности механического устройства перемещения каретки.

Надо иметь в виду, что вводимый с помощью сканера текст передается в компьютер в графическом формате. Для преобразования такого изображения в символьный текст, пригодный для редактирования, используют специальные программы оптического распознавания образов, например FineReader.

В полиграфии используют *барабанные* сканеры большой производительности и высокого разрешения.

Главные отличия в качестве изображений между барабанными и планшетными сканерами возникают из-за принципиально разных типов используемых ими электронных «глаз». Почти во всех барабанных сканерах используются *фотоэлектронные умножители* (ФЭУ), которые гораздо чувствительнее, чем линейки ПЗС. Барабанные сканеры обычно захватывают более широкий диапазон оптических плотностей. Преимущество широкого диапазона плотностей проявляется в самых

светлых и самых темных участках изображения: ФЭУ могут различать очень темные или светлые цвета, которые сканеры на основе ПЗС воспринимают как белый или черный.

Ручные сканеры используются в быту, офисах, в магазинах. Они не занимают места на рабочем столе, но выполняют все необходимые действия.

Аппаратные и программные средства позволяют сканеру автоматически настраиваться, минимизируя потери качества. Наиболее ценятся сканеры, обладающие способностью автокалибровки, то есть настройки на динамический диапазон плотностей оригинала, а также компенсации цветовых искажений.

Первая половина кодировочной таблицы:

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
00	<управляющие символы>							<B>			<LF>			<CR>		
10	<управляющие символы>											<ESC>				
20	<SP>	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
30	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
40	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
50	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
60	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
70	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	□

Вторая половина кодировочной таблицы: кодировка DOS (ASCII).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
80	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
90	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
A0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
B0	░	▒	▓	│	┤	╡	╢	╖	╕	╣	║	╗	╝	╜	╛	┐
C0	└	┴	┬	├	─	┼	╞	╟	╚	╔	╩	╦	╠	═	╬	╧
D0	╨	╤	╥	╙	╘	╒	╓	╫	╪	┘	┌	█	▄	▌	▐	▀
E0	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я
F0	Ё	ё	Є	є	Ї	ї	Ў	ў	°	∙	·	√	№	¤	■	

Вторая половина кодировочной таблицы: кодировка Windows (ANSI).

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
80	Ђ	Ѓ	‚	ѓ	„	…	†	‡	€	‰	Љ	‹	Њ	Ќ	Ћ	Џ
90	ђ	‘	’	“	”	•	–	—	□	™	љ	›	њ	ќ	ћ	џ
A0		Ў	ў	Ј	¤	Ґ	¦	§	Ё	©	Є	«	¬		®	Ї
B0	°	±	І	і	ґ	µ	¶	·	ё	№	є	»	ј	S	s	ї
C0	А	Б	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П
D0	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Ь	Э	Ю	Я
E0	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F0	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	ь	э	ю	я

Список использованных источников

1. Современные операционные системы. 2-е изд./ Э. Таненбаум – СПб.: Питер, 2004. – 1040 с.

2. Вендров А. М. Проектирование программного обеспечения экономических информационных систем: Учебник. – М. Финансы и статистика, 2006. – 544 с.,

3. Фролов А. В., Фролов Г. В. Аппаратное обеспечение IBM PC. Том 2, часть 2. – М. «Диалог МИФИ», 1992.

4. Таненбаум Э. Архитектура компьютера. 4-е изд. – СПб, Питер, 2005. – 899 стр.

5. Каган Б.М. Электронные вычислительные машины и системы. М. , 1985, 324 стр.

6. Вильковский В.А., Малышкин В.Э. Элементы современного программирования и суперЭВМ - Новосибирск: Наука, 1990. – 143 стр.

7. Рабинович Е. В. Информатика. Уч. пособие. - НГТУ, Новосибирск

8. Беляев А. В. Методы и средства защиты информации. Конспект лекций , ЧФ СПбГТУ.- 2000 г. www/citforum.ru/internet/infsecure /