Б.С. Богумирский

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ ПЭВМ

1

Б. С. Богумирский

РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ ПЭВМ

В двух частях

Часть 1

Санкт–Петербург
Ассоциация «OILCO»
1992

УДК 681.3

Богумирский Б.С.

Руководство пользователя ПЭВМ: В 2-х ч. Ч. 1. -- Санкт-Петербург: Ассоциация OILCO, 1992. — 357 с.: ил.

ISBN 5-87738-105-9

Книга содержит обозрение аппаратных средств и системного программного обеспечения персональных ЭВМ (ПЭВМ). Исчерпывающим образом описываются пользовательские интерфейсы операционной системы DOS версий 3.3, 4.0 и 5.0. Изложение сопровождается множеством примеров и практических советов. Затрагиваются вопросы управления ресурсами ПЭВМ, в том числе периферийными устройствами. Рассматриваются системные программные продукты, необходимые для эффективной работы на ПЭВМ, и даются рекомендации по их использованию.

В часть 1 входят разделы 1 — 5.

Для пользователей IBM-совместимых ПЭВМ.

ISBN 5-87738-105-9

ОГЛАВЛЕНИЕ

ПЕРЕЧЕНЬ СОКРАЩЕНИЙ

АИ — адаптер интерфейса
АПУ — адаптер периферийного устройства
АРМ — автоматизированное рабочее место
БД — банк данных
БзД — база данных
БМ — базовый модуль
ВДП — внешняя дублирующая память
ВЗУ — внешнее запоминающее устройство
ВОДП — внешняя оперативно-доступная память
ГД — гибкий диск
ЗУПВ — запоминающее устройство с произвольной выборкой
ИИ — искусственный интеллект
ИМС — интегральная микросхема
КП — командный процессор
ЛВС — локальная вычислительная сеть
МД — магнитный диск
МЛ — магнитная лента
МП — микропроцессор
МПЗУ — масочное ПЗУ
МР — модуль расширения
НГМД — накопитель на гибких магнитных дисках
НЖМД — накопитель на жестких магнитных дисках
НМД — накопитель на магнитных дисках
НМЛ — накопитель на магнитных лентах
НОД — накопитель на оптических дисках
ОД — оптический диск
ОЗУ — оперативное запоминающее устройство
ОП — основная память
ОС — операционная система
ПВВ — порт ввода-вывода
ПЗУ — постоянное запоминающее устройство
ПО — программное обеспечение
ППЗУ — программируемое ПЗУ
ППЭВМ — профессиональная персональная ЭВМ
ПУ — периферийное устройство
ПЭВМ — персональная ЭВМ
РОН — регистр общего назначения
СВТ — средства вычислительной техники
СД — словарь данных
СОП — сверхоперативная память
СП — система прерываний
СПДП — система прямого доступа к памяти
СПО — системное программное обеспечение
СППЗУ — стираемое программируемое ПЗУ
СУБД — система управления базой данных
СШ — системная шина
УВВ — устройство ввода-вывода
ЦМД — цилиндрический магнитный домен
ЭС — экспертная система
ЭСППЗУ — электрически стираемое программируемое ПЗУ

ANSI — Американский Национальный институт стандартов
ASCII — американский стандартный код для обмена информацией
BIOS — базовая система ввода-вывода
BR — загрузочная запись
CISC — компьютер со сложной системой команд
DPB — блок параметров диска
DRAM — динамическое RAM (ЗУПВ, ОЗУ)
EEPROM — электрически стираемое программируемое ROM (ПЗУ)

EISA — расширенная ISA
EMB — блок расширенной памяти
EMM — менеджер отображаемой памяти
EMS — спецификация отображаемой памяти
EOF — маркер конца файла
EOL — маркер конца строки
EPROM — стираемое программируемое ROM (ПЗУ)
ESC/P — стандартный код фирмы Epson для принтера
FAT — таблица размещения файлов
FCB — блок управления файлом
HMA — область верхней памяти
IRQ — линия запроса прерывания
ISA — стандартная промышленная архитектура
LDT — таблица логического диска
MCB — блок управления памятью
MBR — главная загрузочная запись
MCA — архитектура «Микроканал»
MROM — масочное ROM (ПЗУ)
NSB — внесистемный загрузчик
P-SRAM — псевдостатическое RAM (ЗУПВ, ОЗУ)
PROM — программируемое ROM (ПЗУ)
PSP — префикс программного сегмента
PT — таблица разделов
RAM — запоминающее устройство с произвольной выборкой
RDir — корневой каталог
RISC — компьютер с сокращенной системой команд
ROM — постоянное запоминающее устройство
RSec — зарезервированный сектор
SB — системный загрузчик
SCRAM — статическое векторное RAM (ЗУПВ, ОЗУ)
SMBR — вторичная MBR
SRAM — статическое RAM (ЗУПВ, ОЗУ)
TSR — резидентная программа
UMB — блоки верхней памяти
XMM — менеджер расширенной памяти
XMS — спецификация расширенной памяти

Названия фирм:

AMD — Advanced Micro Devices
CDC — Control Data Corp.
DEC — Digital Equipment Corp.
HP — Hewlett-Packard
IBM — International Business Machine Corp.
ITT — Integrated Information Technology

ВВЕДЕНИЕ

В настоящее время мы являемся свидетелями бурного развития вычислительной техники и ее внедрения во многие сферы человеческой деятельности. Особенно отчетливо эти тенденции проявляются, когда речь заходит о персональных ЭВМ (ПЭВМ).

Сами ПЭВМ и их программное обеспечение (ПО) совершенствуются столь стремительно, что отнюдь не каждому специалисту оказывается под силу ориентироваться в этой области, не говоря уже о рядовых пользователях. Особенно серьезные проблемы возникают при покупке ПЭВМ, в частности, при выборе периферийного оборудования. Типична ситуация, когда человек еще не успел в совершенстве освоить какой-либо программный продукт, а уже появилась его новая версия.

Интерес к персональным компьютерам постоянно растет, а круг их пользователей непрерывно расширяется. В число пользователей ПЭВМ вовлекаются как новички в компьютерном деле, так и специалисты по другим классам ЭВМ. Персональная машина становится надежным помощником человека в его профессиональной деятельности, а недалек тот день, когда многие соотечественники будут иметь ПЭВМ не только на рабочем месте, но и дома. Об этом говорит опыт промышленно развитых стран с высоким уровнем компьютеризации. Однако и у нас в стране ПЭВМ уже используются в сфере досуга, развлекая детей и взрослых компьютерными играми. Широкое внедрение ПЭВМ радикально изменит стиль работы и быт человека, обеспечив автоматизацию рутинных вычислительных работ и оперативный доступ к разнообразной информации. Последнее, конечно, может быть полностью реализовано только при объединении компьютеров в глобальную сеть.

Высокая популярность ПЭВМ привела к появлению на отечественном рынке множества изданий как по техническим средствам персональных машин, так и по их ПО.

Наряду с заслуживающими внимания книгами, к сожалению, есть и такие, которые, не отличаясь высоким качеством, нацелены лишь на то, чтобы «снять сливки».

Переводные книги в большинстве случаев не могут конкурировать с отечественными по причине того, что содержащийся в них материал устаревает еще до их издания. Задержка в появлении перевода на русский язык пусть даже очень хорошей книги составляет обычно не менее 3-х лет, а это очень большой срок для такой динамичной области, как ПЭВМ. К примеру, новые версии программных продуктов поставляются на мировой рынок раз в год, а то и чаще.

Отечественным изданиям, ориентированным на пользователей ПЭВМ, присущи следующие недостатки:

1) отсутствие систематизированного материала учебного характера;
2) ориентированность на пользователя-новичка, вследствие чего:
— отсутствие некоторых важных тем;
— поверхностное освещение ряда вопросов;
3) наличие погрешностей методического характера, ошибок и неточностей;
4) описание порой устаревших версий программных продуктов.

Некоторые из перечисленных недостатков нейтрализуются тем, что фирмы-разработчики в последнее время стали предлагать документацию к своим программным продуктам на русском языке. Но документация сама по себе не решает всех проблем (иначе книги вообще были бы не нужны) и в силу специфики нашего экономического положения малодоступна.

В предлагаемой Вашему вниманию книге автор попытался хотя бы частично устранить названные недостатки существующих изданий. Она ориентирована в первую очередь на пользователей-профессионалов машин типа IBM PC (лиц, использующих ПЭВМ для разработки ПО), а также на обслуживающий персонал и преследует следующие *цели*:

1) систематизацию сведений по техническим средствам и системному ПО ПЭВМ;
2) рассмотрение широкого круга тем, важных для пользователя;
3) полноту освещения затрагиваемых вопросов;
4) изучение последних версий популярных системных программных продуктов.

Книга содержит все сведения и для начинающего пользователя, но с его точки зрения окажется чрезмерно перегруженной материалом. Если такого пользователя данное обстоятельство не испугает, то он сможет заметно повысить свою квалификацию.

Книга может использоваться в *качестве*:

1) учебного пособия;

2) руководства по работе с наиболее известными системными программными продуктами нижнего слоя;

3) справочника по оборудованию и системному ПО ПЭВМ.

Проработка книги *позволит*:

— систематизировать имеющиеся у читателя, возможно, обрывочные сведения по ПЭВМ и их ПО;

— осуществить в зависимости от своих потребностей квалифицированный выбор ПЭВМ, периферийного оборудования и системных программных продуктов;

— научиться работать на ПЭВМ и действовать в нештатных ситуациях (в частности, при разрушении информации на магнитных дисках или появлении компьютерных вирусов).

Книга состоит из 10 разделов и 5 приложений.

Разделы 1 — 3 посвящены ознакомлению с техническими средствами ПЭВМ.

Первый раздел содержит начальные сведения о персональных компьютерах. Определяется место и роль ПЭВМ в широком разнообразии средств вычислительной техники, приводятся исторические сведения, рассматривается структура ПЭВМ, дается их классификация и характеризуются области применения персональных машин.

Во *втором разделе* описываются технические средства ПЭВМ, включая микропроцессоры и периферийные устройства. Приводятся классификация, а также технические характеристики; дается сравнительная оценка устройств различных типов и моделей. Рассматривается архитектура микропроцессора 8086/88, который эмулируется более совершенными, чем он, изделиями при работе под управлением операционной системы MS-DOS.

Третий раздел является справочником по наиболее распространенным моделям ПЭВМ (главным образом — IBM-совместимым). Описываются машины всевозможных классов, причем с различными вариантами конструктивного исполнения.

В *четвертом разделе* приводится классификация системного ПО ПЭВМ, даются необходимые определения. Кратко описываются и сравниваются представители различных типов системного ПО ПЭВМ.

Остальные разделы посвящены подробному рассмотрению системных программных продуктов нижнего слоя, которые являются неотъемлемой составной частью любой IBM-совместимой ПЭВМ.

В *пятом разделе* содержится описание операционной системы MS-DOS версий 3.3 и 4.0. Это позволяет использовать материал как пользователями наиболее распространенной версии 3.3, так и более совершенной версии 4.0. Обсуждаются принципы построения и функционирования DOS, описывается программный интерфейс DOS на понятийном уровне. Рассматривается кодирование символов в ПЭВМ и приводятся исчерпывающие сведения о возможностях взаимодействия пользователя с выполняющимися программами посредством клавиатуры. Со всеми деталями, порой отсутствующими даже в фирменной документации, описываются пользовательский интерфейс DOS (команды пользователя), методы и средства разработки командных файлов, а также методы и средства конфигурирования DOS. Приводятся сведения по управлению посимвольными устройствами (дисплеем, клавиатурой и принтером). Излагаются вопросы, касающиеся размещения информации на магнитных дисках.

Шестой раздел посвящен рассмотрению оболочки Norton Commander 3.0, являющейся надстройкой над DOS и существенно упрощающей работу на ПЭВМ. Приводятся также особенности оболочки Pie Commander.

В *седьмом разделе* описывается специализированная оболочка SHEZ 5.5, ориентированная на сжатие и распаковку файлов.

В *восьмом разделе* рассматривается комплект утилит Norton Utilities 6.0, без которых немыслима нормальная эффективная работа на ПЭВМ.

Девятый раздел посвящен описанию утилиты Norton Backup 1.1, облегчающей и автоматизирующей процессы резервирования файлов с жесткого диска и их восстановления.

В *десятом разделе* приводятся базовые сведения о компьютерных вирусах и рассматриваются наиболее совершенные антивирусные средства, а именно: пакет фирмы McAfee Associates версии 74-В и полидетектор-дезинфектор AIDSTEST Д.Н. Лозинского.

Приложение 1 содержит описание сообщений DOS и требуемой реакции пользователя при их возникновении.

В *приложении 2* описываются основные особенности операционной системы DOS версий 5.0 и 6.0.

В *приложении 3* приводятся все необходимые сведения по эксплуатации магнитных дисков и в особенности жестких.

Приложение 4 содержит перечень сообщений утилиты Norton Backup.

Приложение 5 представляет собой каталог копьютерных вирусов.

Книга содержит большое число иллюстраций и практически полезных примеров.

Трудности технического характера не позволили издать книгу одним томом. Поэтому она состоит из двух частей, причем в *первой части* содержатся разделы 1 — *5*, а во *второй* — весь оставшийся материал. Нумерация страниц в книге сквозная.

Автор отдает себе отчет в том, что когда читатель возьмет книгу в руки, часть содержащихся в ней сведений уже устареет. Но вместе с тем общие идеи, принципы и концепции останутся.

Преемственность же версий программных продуктов позволит перейти к работе с новой версией безболезненно. Автор сделал все от него зависящее, чтобы уменьшить разрыв между достигнутым уровнем развития технических средств, а также системного ПО ПЭВМ и его освещением, постоянно внося изменения в рукопись. Достаточно сказать, что он осваивал комплект Norton Utilities по версии 4.5, описывал версию 5.0, а затем переработал материал применительно к версии 6.0. Второе же приложение переделывалось три раза, не говоря уже о внесении множества поправок и дополнений.

Автор выражает признательность редакциям журналов «КомпьютерПресс» и «Мир ПК», сведения из которых использованы при подготовке первых разделов настоящей книги.

1. ОБЩИЕ СВЕДЕНИЯ О ПЕРСОНАЛЬНЫХ ЭВМ

Данный раздел является введением в одну из наиболее популярных и динамичных областей вычислительной техники, в центре которой находятся ПЭВМ.

Читатель ознакомится с классификацией ЭВМ, различными их типами, местом ПЭВМ в иерархии средств вычислительной техники (СВТ).

В разделе дается экскурс в историю создания и развития персональных машин, перечисляются базовые семейства и модели ПЭВМ, а также основные их производители.

После этого рассматриваются структура и состав ПЭВМ, а также обсуждаются функции, выполняемые ее устройствами.

Затем проводится классификация ПЭВМ по совокупности значений основных параметров, целевому назначению и конструктивному исполнению, перечисляются требования, предъявляемые к их комплектации.

Раздел завершает характеристика областей применения ПЭВМ в профессиональной деятельности человека, типов возлагаемых на них задач, а также социальных последствий широкомасштабного их внедрения.

1.1. Место ПЭВМ в иерархии средств вычислительной техники

ЭВМ классифицируются по различным признакам, в частности, по способам организации вычислительного процесса, функциональным возможностям, способности к параллельному выполнению программ и др. Однако чтобы определить место ПЭВМ в широком разнообразии СВТ, следует рассмотреть классификацию вычислительных машин по таким показателям, как габариты и производительность (см. рис. 1.1).

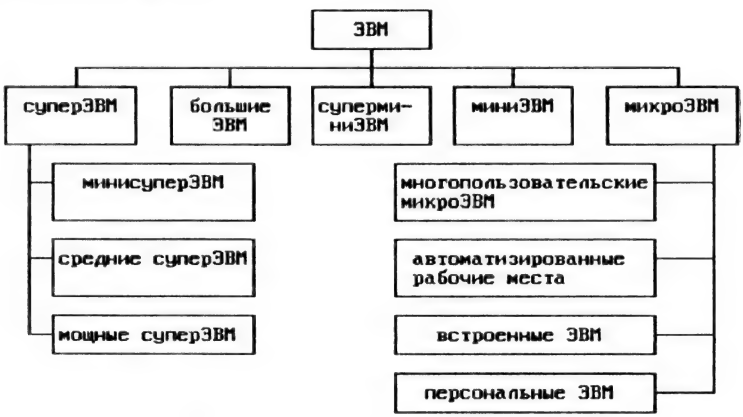


Рис. 1.1. Классификация ЭВМ

Исторически первыми появились *большие ЭВМ*, элементная база которых прошла путь от электронных ламп до интегральных схем со сверхвысокой степенью интеграции. В настоящее время применяются большие ЭВМ четвертого поколения и ведутся интенсивные работы по созданию ЭВМ пятого поколения. ЭВМ этого класса, как правило, используются в режиме разделения времени, одновременно обслуживая многих пользователей. Большие ЭВМ конструктивно выполнены в виде нескольких стоек, включая устройства ввода-вывода, а также внешние запоминающие устройства на магнитных дисках и лентах. Для установки машин требуется достаточно большое помещение, оборудованное средствами обеспечения заданного температурного режима. Обслуживание больших ЭВМ трудоемко, зато их производительность лежит в пределах от нескольких сот тысяч до нескольких миллионов команд в секунду. К большим машинам относится подавляющее большинство моделей IBM 360/370 и их отечественных аналогов — ЕС ЭВМ.

Для иллюстрации возможностей больших ЭВМ в табл. 1.1 представлены основные технические характеристики моделей ЕС ЭВМ Ряда-3. В настоящее время создаются технические и программные

средства четвертой очереди, а также идет подготовка к разработке основных концепций пятой очереди ЕС ЭВМ. Среди новинок назовем модель ЕС 1130 Ряда-4, обладающую производительностью около 2,5 млн. команда/с и емкостью оперативного запоминающего устройства (ОЗУ) до 8 Мбайт.

Таблица 1.1

Основные технические характеристики моделей ЕС ЭВМ Ряда-3

Модель ЭВМ	Производительность(*), млн. команда/с	Емкость ОЗУ (**), Мбайт	Число каналов ввода-вывода	Стоимость (***), тыс. руб.	Занимаемая площадь, м²
ЕС 1036	0,4	2(4)	5	516,7-700,0	56-96
ЕС 1046	1,3	2(8)	6	650,0-850,0	100-127
ЕС 1066	5,5	8(16)	12	1600,0-2100,0	200-260
ЕС 1068	10,5	32	24	н/д	н/д
ЕС 1087	14	8(16)	12	н/д	н/д

н/д - нет данных.

* На смесях команд для научно-технических задач максимальная производительность в 2 раза выше, а на смесях команд для планово-экономических задач - наоборот, в 2 раза ниже приведенной.

** В скобках указана максимальная емкость ОЗУ.

*** По ценам 1990 г.

Производительность больших ЭВМ оказалась недостаточной для ряда приложений — таких, как прогнозирование метеообстановки, моделирование и др., что явилось стимулом для создания *суперЭВМ*. Появляются все новые и новые области их применения, а поэтому потребность в машинах данного класса непрерывно растет. Производительность современных ЭВМ не соответствует многим из таких областей, что обуславливает улучшение показателей суперЭВМ. Под суперЭВМ понимают вычислительную систему, относящуюся к классу самых мощных систем в данное время. Они имеют большие габариты, требуют для своего размещения специальных помещений и весьма сложны в обслуживании. Одной из основных проблем проектирования и эксплуатации является эффективный отвод тепла. Производительность суперЭВМ в настоящее время составляет десятки и сотни млн. команда/с. Две наиболее известные серии суперЭВМ — это Cray (Cray-1, Cray-2 и Cray-3) корпорации Cray Research и Cyber 205 фирмы Control Data Corp. (CDC). Отметим, что Cray-3 способна выполнять 16000 млн. команд с плавающей точкой в секунду. Стоимость отдельных суперЭВМ достигает 10 млн. долл. Из отечественных ЭВМ к данному классу можно отнести машину с динамической архитектурой (МДА) В.А.Торгашева.

В 70-е гг. появился еще один класс ЭВМ — *миниЭВМ*, что обусловлено, с одной стороны, прогрессом в области элементной базы, а с другой — избыточностью ресурсов больших ЭВМ для ряда приложений. МиниЭВМ используются как в режиме разделения времени, так и для управления технологическими процессами. Они конструктивно выполнены в виде одной или нескольких малогабаритных стоек (без учета устройств ввода-вывода) и имеют более низкие по сравнению с большими ЭВМ быстродействие и стоимость. ЭВМ данного класса не требуют специально оборудованных помещений. К миниЭВМ относятся машины серии PDP-11 фирмы Digital Equipment Corp. (DEC) и их отечественные аналоги — большинство моделей СМ ЭВМ. Выпускаются также миниЭВМ «Электроника 79» (СНГ); ИЗОТ 1016С, 1016М (Болгария); КОРАЛЛ 4001, 4011, 4030 (Румыния), ЯНУС (Венгрия) и др. Практически все миниЭВМ являются 16-разрядными. Основные технические характеристики отечественных ЭВМ этого класса приведены в табл. 1.2.

Таблица 1.2

Основные технические характеристики миниЭВМ

Модель ЭВМ	Производительность, млн. команда/с	Емкость ОЗУ, Мбайт	Стоимость(*), тыс. руб.	Занимаемая площадь, м²
СМ 1420	0,4	до 2	80-140	15-30
СМ 1425	3,0	до 4	25-30	680x550x180 мм (**)
ЕС 1007	0,133	1	131	25

* По ценам 1990 г.

** Габаритные размеры основной стойки.

Дальнейшие успехи в области элементной базы и архитектурных решений привели к возникновению *суперминиЭВМ*. СуперминиЭВМ — это вычислительная машина, относящаяся по архитектуре, размерам и

стоимости к классу миниЭВМ, но по производительности сопоставимая с большой ЭВМ. СуперминиЭВМ используются, как правило, в режиме разделения времени. Наиболее яркими их представителями являются ЭВМ семейства VAX-11 фирмы DEC. Это семейство послужило прототипом отечественной ЭВМ СМ 1700. Кроме того, выпускаются следующие суперминиЭВМ: «Электроника-82» (СНГ), К1840 (Восточная Германия), СМ 52/12 (Чехо-Словакия), ИЗОТ 1055С (Болгария) и др. Все ЭВМ данного класса являются 32-разрядными. Основные технические характеристики моделей отечественных суперминиЭВМ приведены в табл. 1.3.

Таблица 1.3

Основные технические характеристики суперминиЭВМ

Модель ЭВМ	Производительность, млн. команда/с	Емкость ОЗУ, Мбайт	Стоимость(*), тыс. руб.	Занимаемая площадь, м²
СМ 1700	1,5	до 5	350-500	20-25
СМ 1702	4,5	до 64	н/д	640x220x726 мм (**)
ЕС 1705(***)	3,75	до 64	н/д	1024x1200x600 мм(**)

н/д – нет данных.
* По ценам 1990 г.
** Габаритные размеры основной стойки.
*** Перспективная модель.

Изобретение в 1969 г. микропроцессора (МП) привело к появлению в 70-х гг. еще одного класса ЭВМ — *микроЭВМ*. Именно наличие МП служит определяющим признаком микроЭВМ. Эти ЭВМ, в свою очередь, делятся на многопользовательские микроЭВМ, автоматизированные рабочие места (АРМ), встроенные ЭВМ и ПЭВМ.

Многопользовательские микроЭВМ — это микроЭВМ, оборудованные несколькими видеотерминалами и работающие в режиме разделения времени. Они выполняются, как правило, в одной малогабаритной стойке и изредка — в настольном варианте.

АРМ, или *рабочая станция* (workstation), представляет собой ЭВМ, оборудованную всеми средствами, необходимыми для выполнения работ определенного типа. Различают технические (инженерные) АРМ, графические АРМ, АРМ для автоматизированного проектирования, АРМ для издательской деятельности (настольные издательские системы) и др. В классе микроЭВМ АРМ наряду с многопользовательскими микроЭВМ имеют самое высокое быстродействие. Существуют как настольные АРМ, так и АРМ, выполненные в виде малогабаритной стойки.

Термин АРМ (рабочая станция) неоднозначен и часто употребляется в других смыслах, а именно:

1) для именования ПЭВМ, снабженной специальным ПО, необходимым для решения задач определенного класса;

2) для именования терминальных узлов вычислительных сетей.

Встроенные ЭВМ представляют собой вычислители, используемые для управления (например, станком или боевым средством) и обработки измерений. Конструктивно они выполняются в виде одной или нескольких плат и не обеспечивают реализацию широкого спектра вычислительных функций, а также стандартного взаимодействия с пользователем.

Персональной называется *универсальная однопользовательская* микроЭВМ.

Определение ПЭВМ в значительной степени расплывчато. Для его уточнения выделяют следующие характеристики персональных машин:

1) невысокую стоимость;

2) наличие периферийных устройств (ПУ), необходимых для ввода-вывода и хранения информации;

3) наличие аппаратных ресурсов, достаточных для решения реальных задач (в частности, достаточной емкости ОЗУ);

4) поддержку языков программирования высокого уровня;

5) наличие операционной системы (ОС), которая упрощает взаимодействие пользователя с ПЭВМ;

6) «дружественность» по отношению к пользователю.

Мощные ПЭВМ способны обеспечить работу нескольких пользователей одновременно, что размывает границу между ними и многопользовательскими микроЭВМ.

С другой стороны, в настоящее время стирается граница между ПЭВМ и инженерными АРМ. Причины этого кроются в следующем:

1) АРМ становятся «дружественными» и более дешевыми, в результате чего появился даже термин «персональные АРМ»;

2) АРМ можно снабдить дополнительными ПУ, превращающими его в универсальную ЭВМ;

3) технические характеристики ПЭВМ приближаются к техническим характеристикам АРМ;

4) на базе ПЭВМ можно построить АРМ, снабдив ее специальным оборудованием и соответствующим программным обеспечением.

Интересен тот факт, что мы являемся свидетелями рождения нового класса вычислительных машин — *супермикроЭВМ.*

На долю больших и персональных ЭВМ сейчас приходится примерно половина объема сбыта СВТ в стоимостном выражении, которая делится также примерно поровну между названными классами ЭВМ. Динамика роста числа установленных ПЭВМ подчиняется экспоненциальному закону. Однако относительный рост объема сбыта АРМ в 1991 г. превысил относительный рост объема сбыта персональных машин.

1.2. Эволюция ПЭВМ

История зарождения и развития вычислительной техники довольно коротка. Ее принято исчислять с 1833 г., когда английский математик Чарльз Беббидж впервые проникся идеей создания механического «вычислительного помощника», используя принцип *программного управления.* Потребовалось более 100 лет, чтобы эта идея, обогащенная американским математиком Дж. фон Нейманом в 1945 — 47 гг. и базирующаяся на появившихся к тому времени электронных лампах, положила начало эры ЭВМ. С момента создания в 1947 г. первой программно-управляемой цифровой ЭВМ начался бурный прогресс вычислительной техники. Совершенствование элементной базы привело к существенному уменьшению размеров, стоимости и энергопотребления, а также к повышению быстродействия и надежности ЭВМ. Эволюция архитектурных решений способствовала еще большему улучшению последних двух показателей. Большие успехи достигнуты также в области периферийного оборудования, что существенно облегчило общение пользователей с ЭВМ и повысило емкость накопителей информации.

Еще в 1982 г. Х.Тунг и А.Гупта привели следующее яркое сравнение, иллюстрирующее высокие темпы развития СВТ: «Если бы за последние 25 лет авиационная промышленность развивалась столь же стремительно, как и вычислительная техника, то Боинг-767 можно было бы приобрести сегодня за 500 долл. и облететь на нем земной шар за 20 мин., израсходовав при этом 19 л горючего». За этот период скорость вычислений возросла в 20 раз, а размеры и энергопотребление ЭВМ стали в 10000 раз меньше, чем у машин сравнимой производительности 25-летней давности. В последние годы отмеченные тенденции не только сохранились, но и усилились.

Исходя из этого вполне закономерным явилось появление МП и создание на их основе микроЭВМ, венцом которых стали ПЭВМ. Первая персональная машина была сконструирована американской фирмой MITS в 1975 г. и названа Altair 8800. По сегодняшним меркам она, ощетинившаяся индикаторными лампочками и переключателями, выглядит довольно странно. Цена ее составляла около 6 тыс. долл. Эта машина давно уже не выпускается.

Следующая ПЭВМ была создана в буквальном смысле в гараже двумя молодыми американцами С.Возняком и С.Джобсом в 1976 г. Она получила название Apple-I. Весной 1977 г. ими же был изготовлен относительно дешевый и вместе с тем вполне законченный персональный компьютер Apple-II. Две другие компании — Commodore и Radio Shack (филиал корпорации Tandy) — уже выпустили в продажу похожие машины, однако роль детонатора, вызвавшего взрыв в области ПЭВМ, сыграла именно Apple-II. Вскоре в эту область ринулось множество других конкурентов, а миллионы нетерпеливых покупателей поспешили обзавестись персональным компьютером. В результате домашняя мастерская С.Возняка и С.Джобса превратилась в процветающую фирму Apple Computer, которая и в настоящее время занимает достойное место на рынке ПЭВМ. Первые персональные компьютеры были 8-разрядными и по своим возможностям напоминали больше игрушки, нежели инструмент профессионала. Наиболее популярные ПЭВМ конца 70-х гг., такие, как TRS-80, Apple-II и PET, еще доживают свой век, но по сегодняшним меркам они уже безнадежно устарели.

В начале 80-х гг. в число производителей ПЭВМ влились компьютерные гиганты International Business Machine Corp. (IBM), DEC и Hewlett-Packard (HP). Это не могло не привести к структурным изменениям на рынке персональных компьютеров. Так, в 1981 г. IBM выпустила свою первую удачную 16-разрядную модель PC (Personal Computer) и с этого момента стала флагманом в производстве не только больших, но и персональных ЭВМ. В 1983 г. и 1984 г. появились новые модели машин этой же фирмы, а именно: PC XT (eXtended Technology) и PC AT (Advanced Technology) соответственно. Они стали неписаными стандартами в области ПЭВМ. Многие фирмы-производители освоили выпуск двойников этих изделий, улучшая некоторые из их характеристик или снижая стоимость. Наиболее в этом отношении преуспели компании Compaq и Tandy. В последние годы набрали силу также фирмы AST Research, Dell Computer, ZEOS International и ряд других. Единственной фирмой с существенной долей на мировом рынке ПЭВМ, не свернувшей с намеченного пути и не поддавшейся «давлению» IBM, была и по сей день остается компания Apple Computer, выпускающая все новые и новые конкурентоспособные модели машин. Интенсивные попытки завоевать рынок ПЭВМ были предприняты фирмами Commodore и Atari выпуском моделей Amiga и Atari 1040 ST соответственно. Несмотря на то, что базовые конфигурации этих машин имели лучшие параметры, чем базовые комплекты корпорации IBM, этим фирмам не удалось добиться сколько нибудь заметных успехов в конкурентной борьбе. Причины этого кроются в слишком запоздалом выпуске данных моделей, в

несовместимости с семейством PC IBM и замкнутости архитектуры, не допускающей наращивания периферийного оборудования.

В 1987 г. на рынке ПЭВМ произошло новое потрясение — фирма IBM объявила о выпуске следующего семейства ПЭВМ — PS/2 (Personal System/2), в которое включены не только 16-ти, но и 32-разрядные модели машин. Подавляющее большинство выпускаемых до этого ПЭВМ были 16-разрядными, а наиболее простые и устаревающие — 8-разрядными. Однако, несмотря на усиленные попытки IBM закрепить свое положение на рынке ПЭВМ и уйти вперед от своих конкурентов, фирма Compaq без лишнего шума на год раньше, чем IBM, приступила к выпуску 32-разрядных машин и опережает ее с тех пор в данном секторе рынка. Компания Apple Computer также успешно сбывает свои 32-разрядные модели семейства Macintosh II (сокращенно — Mac II).

В 1989 г. произошло еще одно важное событие — британская компания Apricot приступила к производству 32-разрядной ПЭВМ на базе нового МП фирмы Intel — 80486. Она получила название Apricot VX FT Server. Сейчас уже ряд производителей, включая IBM, предлагает машины этого класса.

Таким образом, всего лишь 15 лет понадобилось для того, чтобы ПЭВМ прошли путь от 8-разрядных «игрушек» до 32-разрядных ЭВМ, обладающих высокими техническими характеристиками и находящих все новые и новые области применения.

Сейчас на рынке ПЭВМ доминируют следующие 3 сектора:

1) *IBM PC AT* и их аналоги (доля машин этого класса составляет около 40%);
2) *32-разрядные ПЭВМ*, совместимые с компьютерами фирмы IBM;
3) *машины семейства Mac II* фирмы Apple Computer.

ПЭВМ типа IBM PC XT по-прежнему используются, но многими фирмами уже сняты с производства.

В настоящее время мировой рынок ПЭВМ вступает в пору зрелости. Этого, к сожалению, никак нельзя сказать о нашей стране, где он находится только лишь в зародышевом состоянии. Мы существенно отстаем не только от развитых капиталистических, но и от многих развивающихся стран как по техническим характеристикам ПЭВМ, так и по объему их производства.

Отечественная промышленность выдает нам за новинки вычислительной техники те ПЭВМ, прототипы которых уже практически не выпускаются даже в странах Юго-Восточной Азии. Наиболее совершенная отечественная ПЭВМ ЕС 1842 находится по своим характеристикам между IBM PC XT и AT.

В США, например, ПЭВМ имеет в среднем каждая семья, не говоря уже о пока сказочной для нас компьютеризации различных организаций и фирм. В СНГ же к 1990 г. было намечено выпустить только 1,1 млн. ЭВМ данного типа.

При написании этого раздела автор не ставил перед собой целью полностью охарактеризовать наиболее популярные модели ПЭВМ, считая это пока преждевременным. Неудовлетворенный читатель может найти интересующие его сведения в разделе 3.

1.3. Структура и состав ПЭВМ

Типовая структура ПЭВМ изображена на рис. 1.2, где использованы следующие еще не введенные сокращения:

ОП — основная память;
ПВВ — порт ввода-вывода;
АПУ — адаптер периферийного устройства;
АИ — адаптер интерфейса.

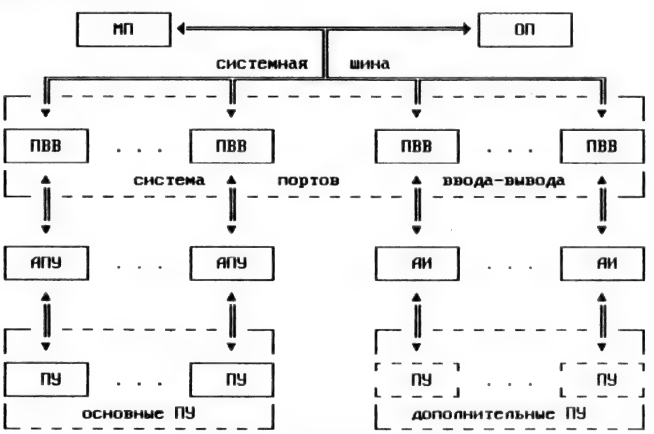


Рис. 1.2. Структура ПЭВМ

На представленной схеме не учтены многие детали, однако это не мешает иллюстрации основных архитектурных особенностей и состава ПЭВМ. Главная отличительная черта структуры персонального компьютера состоит в наличии *системной шины* (СШ), посредством которой взаимодействуют и обмениваются информацией все его устройства. Архитектура с общей СШ обеспечивает простоту и дешевизну ЭВМ, а также унифицирует алгоритмы взаимодействия ее устройств, облегчая программирование; однако наряду с этим общая СШ является узким местом ПЭВМ, потенциально ограничивая ее производительность. Последнее объясняется тем, что в каждый момент времени посредством СШ могут обмениваться информацией только два устройства; остальные же вынуждены простаивать. И тем не менее такое архитектурное решение в ПЭВМ себя полностью оправдывает.

Помимо СШ, в состав ПЭВМ входят следующие устройства: МП, ОП, ПУ, АПУ, ПВВ и АИ.

Важными компонентами любой ЭВМ, в частности, ПЭВМ, являются также система прерываний (СП) и система управления шиной (контроллер системной шины). Кроме того, в состав ПЭВМ зачастую входит система прямого доступа к памяти (СПДП). Перечисленные системы на рис. 1.2 не показаны.

Кратко охарактеризуем основные устройства ПЭВМ, обращая особое внимание на выполняемые ими функции. Взаимодействие устройств при выполнении команд рассматриваться не будет, так как этот вопрос достаточно полно освещен в литературе, раскрывающей принципы построения ЭВМ, и в полной мере относится к ПЭВМ.

МП — это «сердце» ПЭВМ. Он осуществляет вычисления по хранящейся в ОП программе и обеспечивает общее управление компьютером. МП, как минимум, содержит:

— *арифметико-логическое устройство,* предназначенное для выполнения арифметических и логических операций (т.е. являющееся собственно вычислителем ЭВМ);

— *устройство управления,* обеспечивающее общее управление вычислительным процессом по программе и координацию работы всех устройств ПЭВМ.

ОП — это запоминающее устройство, напрямую связанное с процессором и предназначенное для хранения выполняемых программ и данных, непосредственно участвующих в операциях. Она имеет достаточно высокое быстродействие, но ограниченный объем. ОП делится на различные виды, основными из которых являются ОЗУ и постоянное запоминающее устройство (ПЗУ).

ОЗУ служит для приема, хранения и выдачи информации. В нем содержатся программы и данные, доступные для использования процессором, а также промежуточные и окончательные результаты вычислений. Процесс выполнения программы сводится к преобразованию исходного состояния памяти в заключительное (конечное). ОЗУ в ПЭВМ, как и в других классах машин, является энергозависимым, что означает исчезновение информации при отключении питания (если, конечно, отсутствуют встроенные элементы питания). Однако может использоваться и энергонезависимая память на новых физических принципах.

ПЗУ, являясь энергонезависимым, обеспечивает надежное хранение и выдачу информации. Содержимое ПЗУ не может быть изменено. В нем хранятся часто используемые (универсальные) программы и данные, такие, как программы ОС и ее информационные структуры, а также интерпретаторы и компиляторы языков программирования. Существуют и полупостоянные запоминающие устройства, или стираемые ПЗУ.

Еще раз подчеркнем, что МП в своей работе использует только информацию, хранимую в ОП. Если же программы и/или данные находятся в другом устройстве, то они должны быть предварительно размещены в ОЗУ. Логически ОП можно представить в виде совокупности ячеек, доступ к каждой из которых осуществляется путем указания ее адреса.

Под *периферийным* понимают любое устройство, конструктивно отделенное от центральной части ПЭВМ (МП и ОП), имеющее собственное управление и выполняющее запросы МП без его непосредственного вмешательства. По функциональному признаку ПУ делятся на две основные группы:

1) *внешние запоминающие устройства* (ВЗУ), служащие дополнительным энергонезависимым, более медленным, но и более емким полем памяти машины для долговременного хранения программ и данных;

2) *устройства ввода-вывода* (УВВ), обеспечивающие общение пользователя с ПЭВМ.

В качестве ВЗУ в ПЭВМ обычно используются накопители на магнитных дисках (НМД) — как гибких, так и жестких, накопители на магнитных лентах (НМЛ), а также накопители на оптических дисках (НОД). Внедрение последних началось совсем недавно.

Набор УВВ для ПЭВМ существенно богаче: дисплеи, клавиатуры, различные манипуляторы (типа «мышь», джойстики и типа «шар»), печатающие устройства (принтеры), устройства ввода и вывода графической информации и др. Этот список можно продолжить вплоть до синтезаторов речи и даже устройств ввода речевой информации.

Состав ПУ может сильно меняться от модели к модели ПЭВМ. Он определяется, главным образом, ее назначением. Так, например, ПЭВМ для дома может содержать единственное ПУ — клавиатуру, при этом в качестве устройства вывода информации используется телевизор, а в качестве накопителя — бытовой магнитофон.

По степени важности для ПЭВМ ПУ делятся на основные и дополнительные (факультативные). *Основные* ПУ являются неотъемлемой составной частью всех ПЭВМ или каких-либо их типов. *Факультативные* же ПУ могут поставляться и подключаться по специальным заказам. Без

дополнительных ПУ персональная машина хотя и может решать возложенные на нее задачи, но порой с меньшей эффективностью. Без основных ПУ работа на ПЭВМ, как правило, становится невозможной. К основным ПУ относятся дисплей, клавиатура и, как минимум, один накопитель информации (ВЗУ). Остальные ПУ считаются факультативными.

Основные ПУ подключаются к СШ не непосредственно, а через цепочку АПУ — ПВВ.

АПУ выполняет две основные функции:

1) осуществляет непосредственное *управление* ПУ по запросам от МП, освобождая тем самым последний от выполнения рутинных операций;

2) обеспечивает *согласование* интерфейса ПУ с СШ.

Остановимся несколько подробнее только на второй функции. Периферийное оборудование выпускается многими производителями, зачастую без ориентации на то или иное семейство ПЭВМ. Кроме того каждый тип ПУ обладает своей спецификой. Другими словами, ПУ имеют различные интерфейсы (соглашения о связях с другими устройствами). Поэтому очевидно то, что для подключения таких ПУ к общей СШ ПЭВМ нужны согласующие устройства, роль которых и выполняют АПУ. Для подсоединения нового ПУ при этом достаточно разработать для него соответствующий адаптер, совместимый с той или иной СШ. Иногда в таких случаях можно обойтись даже без какой-либо модернизации системного ПО, если ПУ данного типа уже предусмотрено в системе.

Понятие «адаптер периферийного устройства» можно считать синонимом термина «контроллер», однако последний употребляется чаще для устройств, реализующих более сложные функции по управлению ПУ. Развитые АПУ включают в свой состав специализированные МП и память. Это же относится и к ПУ со сложными алгоритмами работы, требующими наличия совершенных блоков управления.

ПВВ обеспечивает непосредственное подключение АПУ к СШ (т.е. является по сути «точкой» такого подключения). Каждый ПВВ имеет свой адрес, аналогичный адресу в ОП, но содержащийся в другом адресном пространстве. Одному ПУ может быть приписано несколько ПВВ. Упрощенно ПВВ можно считать регистром, в который записывается информация для передачи в ПУ или с которого считывается полученная из ПУ информация. Каждое стандартное ПУ для унификации ПО закреплено за ПВВ с определенными адресами. Можно провести аналогию ПВВ с морским портом, объясняющую происхождение этого термина. Достаточно считать, что грузы, привозимые в морской порт и отправляемые на судах в различные концы света, а также грузы, разгружаемые с пришедших судов и доставляемые получателям, — это информация, передаваемая через ПВВ. Однако последние, в основном, являются однонаправленными. Совокупность ПВВ образует *систему портов ввода-вывода.*

АИ выполняют роль согласующих звеньев для сопряжения центральной части ПЭВМ с дополнительными ПУ, интерфейсы которых для универсальности стандартизованы. Иными словами, АИ — это в определенном смысле универсальный адаптер. Более строго АИ — это средство сопряжения центральной части ПЭВМ с дополнительным ПУ, в котором все физические (электрические) и логические параметры отвечают предварительным соглашениям (стандартный интерфейс) и широко используются в других устройствах. К таким АИ могут подключаться ПУ со своими контроллерами, модемы для внедрения в телефонную сеть и т.п.

Примером интерфейса, стандартизованного на уровне производителя, используемого для подключения принтера и совместимого со многими интерфейсами других производителей печатающих устройств, является параллельный интерфейс фирмы Centronics. В качестве другого примера можно привести последовательный интерфейс RS232C, который используется для подсоединения многих типов относительно медленных ПУ и модемов. Спецификация данного интерфейса опубликована Ассоциацией электронной техники министерства торговли США и при этом полностью соответствует рекомендациям Международного консультативного комитета по телеграфии и телефонии в Западной Европе. Взамен RS232C недавно предложен интерфейс EIA-232-D. В СНГ с интерфейсами Centronics и RS232C совместимы интерфейс ИРПР-М и стык С2 соответственно.

АИ зачастую называют просто интерфейсами, что, учитывая полисемию (многозначность) этого термина, представляется нецелесообразным. Действительно, при такой трактовке два понятия — устройство для согласования и соглашения о связях — будут называться одинаково в одном и том же контексте, что может привести к путанице. Используется и другой синоним АИ — порт (параллельный для Centronics или последовательный для RS232C), чего также, на наш взгляд, придерживаться нежелательно.

МП должен оперативно реагировать на различные события, происходящие в ПЭВМ в результате действий пользователя или без его ведома. В качестве примеров таких событий можно привести нажатие клавиши на клавиатуре (и другие события в ПУ), попытка деления на нуль, переполнение разрядной сетки, сбой питания (а также иные нарушения в работе оборудования), запланированные в программе обращения к ядру ОС и т.п. Необходимую реакцию на события обеспечивает система прерываний. *Прерыванием* называется ситуация, требующая каких-либо действий (реакции) МП при возникновении определенного события. Под *системой* же *прерываний* понимают комплекс аппаратных и программных средств, обеспечивающих выявление и обработку прерываний.

Обработка прерываний сводится к приостановке исполнения текущей последовательности команд (программы), вместо которой начинает интерпретироваться другая последовательность инструкций, соответствующая данному типу прерывания и называемая обработчиком прерывания. После ее реализации исполнение прерванной программы может быть продолжено, если это

возможно и/или целесообразно, что зависит от типа прерывания. Реакция на прерывание может состоять, например, в обработке введенного с клавиатуры символа.

СПДП является факультативным устройством, служащим для того, чтобы разгрузить МП при обмене информацией между ОЗУ и быстродействующими ПУ (такими, как НМД), а также увеличить скорость обмена. Без СПДП в обмене информацией между ОЗУ и ПУ непосредственно участвует МП, «пропуская» через себя эту информацию довольно маленькими порциями (пословно). При использовании СПДП МП только инициирует операцию обмена целым блоком информации, активизируя СПДП. Сам же обмен происходит напрямую между ОЗУ и ПУ под ее управлением. МП в это время может взять на себя другую работу. СПДП подключается к ПЭВМ аналогично ПУ (через цепочку АПУ — ПВВ).

Контроллер системной шины управляет СШ ПЭВМ в зависимости от состояния МП. Этот контроллер может входить в состав МП или выполняться в виде отдельного устройства.

Сама *СШ* представляет собой совокупность одно- и двунаправленных линий, логически объединяемых в следующие группы:

1) *шину данных*, служащую для передачи информации в оба направления (от МП к ОЗУ или ПУ и обратно, либо между ОЗУ и ПУ при задействовании СПДП);

2) *шину адреса*, с использованием которой адресуются ОП и ПВВ;

3) *шину управления*, предназначенную для передачи управляющих сигналов, таких, как «запись в память», «чтение из памяти», «запись в порт», «чтение из порта», сигналы прерываний и т.п.

Физически шины адреса и данных могут мультиплексироваться (совмещаться). Такая идеология принята, например, в семействах PC IBM и ЕС (шина Multibus).

Более детально многие устройства ПЭВМ рассматриваются в разделе 2.

1.4. Классификация ПЭВМ и требования к их комплектации

В соответствии с ГОСТом 27201-87 ПЭВМ подразделяют на типы в зависимости от совокупности значений базовых параметров, определяющих основные функциональные возможности ПЭВМ и тем самым сферы их применения. Условные обозначения типов и свойственные им значения основных параметров ПЭВМ приведены в табл. 1.4. Сделаем ряд пояснений, касающихся этой таблицы:

1) 1 Кбайт равен 1024 байт, а 1 Мбайт — 1024 Кбайт, где байт — восьмиразрядный (восьмибитовый) код;

2) емкость НМД проставлена без учета форматирования, которое «съедает» часть объема диска;

3) для ПЭВМ типов ПМ 4 и ПМ 5, предназначенных для работы с дополнительными устройствами отображения графической информации, количество адресуемых точек должно быть не менее 1024x1024;

4) потребляемая мощность и масса установлены для центральной части ПЭВМ с клавиатурой и соответствующими НМД, но без других ПУ, в частности, дисплея;

5) в числителе указаны значения параметров на 1987 — 1990 гг., а в знаменателе — на 1991 — 1995 гг.;

6) значения параметров ПЭВМ для типа ПМ 2 установлены для одного рабочего места ученика; значения же параметров для рабочего места преподавателя должны соответствовать нормам, установленным для типов ПМ 2 — ПМ 5;

7) в скобках указаны значения параметров, которые могут быть удовлетворены по требованию заказчика;

8) для ПЭВМ типов ПМ 1 и ПМ 2 по согласованию с заказчиком (потребителем) допускается замена накопителей на гибких магнитных дисках накопителями на кассетных магнитных лентах или магнитофонами.

В научно-технической литературе в связи с классификацией ПЭВМ обычно употребляются не условные наименования типов ПМ 1 — ПМ 5, которые не несут никакой смысловой нагрузки, а названия типов компьютеров в соответствии с их целевым назначением. Последнее, естественно, определяется техническими характеристиками и приведено в только что рассмотренной таблице. Таким образом, по **целевому назначению** (или **сфере применения**) ПЭВМ делятся на следующие типы:

1) *бытовые* ПЭВМ, которым соответствует тип ПМ 1;

2) *учебные* ПЭВМ (типы ПМ 2 — ПМ 3);

3) *профессиональные* ПЭВМ (ППЭВМ), которым соответствуют типы ПМ 3 — ПМ 5.

ППЭВМ имеют лучшие по сравнению с другими типами ПЭВМ характеристики; бытовые же ПЭВМ, наоборот, обладают наиболее ограниченными ресурсами.

В соответствии с требованиями упомянутого ГОСТа ПЭВМ должна состоять из базового комплекта, периферийных устройств, а также других технических и программных средств, необходимых для удовлетворения запросов пользователей.

Базовый комплект ПЭВМ должен включать:

— основной и, при необходимости, дополнительный микропроцессоры;

— оперативное и, при необходимости, постоянное запоминающие устройства;

— основные ПУ — клавиатуру и устройство отображения информации (как правило, дисплей);

— средства подключения ПУ (адаптеры, контроллеры);

— базовое ПО (ОС, программы контроля работоспособности ПЭВМ и пакеты прикладных программ общего назначения);

— средства подключения устройств сопряжения с локальной сетью;

— средства подключения устройств, расширяющих функциональные возможности ПЭВМ;

— источник питания.

Таблица 1.4

Типизация ПЭВМ

Наименование параметра	Типы и значения параметров для них				
	ПМ 1	ПМ 2	ПМ 3	ПМ 4	ПМ 5
Разрядность основного МП, разряды	8; 16			16; 32	не менее 32
Быстродействие, млн. коротких операций в секунду (типа "регистр-регистр"), не менее	0,5		1,0	1,0 / 2,0	2,0 / 4,0
Емкость ОЗУ, Мбайт, не менее	0,0625	0,0625 / 0,125	0,5(0,125) / 1,0(0,25)	1,0(0,25) / 2,0(0,25)	2,0(1,0) / 8,0(2,0)
Емкость накопителя на гибком магнитном диске, не менее	360 Кбайт	360 Кбайт / 720 Кбайт	360 Кбайт – до 1.01.88; 1 Мбайт – до 1.01.91 / 3 Мбайт		
Емкость накопителя на жестком магнитном диске, Мбайт, не менее	–		10 / 20(10)	20(10) / 40(20)	40 / 80
Количество адресуемых точек на экране дисплея, точек, не менее	640x200	512x256 / 640x200	640x200	640x200 / 720x512	720x512
Цветность дисплея	одноцветный, многоцветный				многоцветный
Потребляемая мощность, ВхА, не более	30 / 20	40 / 30	100 / 70	150 / 100	200 / 150
Масса, кг, не более	5 / 3	7 / 5	10 / 7	15 / 12	20 / 15
Основная рекомендуемая область применения ПЭВМ	Индивидуальное применение в бытовых условиях	Массовое обучение (рабочие места учеников)	Профессиональное обучение, профессиональная деятельность (обработка текстов, планирование, экономические и инженерные расчеты)	Профессиональная деятельность (образование, здравоохранение, научная, инженерная, административно-управленческая, финансовая, экономическая и др.)	

В соответствии с запросами пользователей ПЭВМ могут быть укомплектованы дополнительно к базовому комплекту различными *ПУ*. Стандартно обычно предусматривается следующий минимальный набор:

— накопители на гибких и жестких магнитных дисках (последние — не всегда);

— печатающее устройство.

Отметим, что НМД для типов ПМ3 — ПМ *5* являются основными, а для других — факультативными ПУ.

ПЭВМ должна обеспечивать возможность расширения ее ресурсов путем подключения дополнительных плат, модулей или устройств (модули ОЗУ и ПЗУ, специализированные процессоры, устройства связи с другими ЭВМ, устройства связи с объектом управления, манипуляторы и др.).

Базовое ПО ППЭВМ должно включать минимальный набор пакетов прикладных программ общего назначения, выполняющих следующие функции:
— обработку текстовой информации;
— обработку табличной информации;
— обработку графической информации;
— работу в локальной сети;
— управление базами данных;
— трансляцию с языков программирования Бейсик, Фортран, Паскаль и Си.

Кроме целевого назначения, в зависимости от **конструктивного исполнения** ПЭВМ делятся на стационарные и портативные.

Стационарные ПЭВМ обычно устанавливаются на рабочем столе. Значительно реже они выполняются в виде малогабаритной стойки, размещаемой на полу.

Портативные ПЭВМ делятся на следующие категории:

1) *переносные* (portable), которые, имея сравнительно небольшие массу и габариты, поддаются транспортировке одним человеком;

2) *наколенные* (laptop), выполненные в виде «дипломата»;

3) *блокнотные* (notebook), имеющие габариты большого блокнота;

4) *карманные* (pocket), которые помещаются в карман.

Категории портативных машин перечислены в порядке уменьшения их габаритов и массы.

Переносные ПЭВМ, как правило, не имеют автономного питания, в то время как портативные персональные компьютеры других классов способны питаться от батарей или аккумуляторов.

Деление ПЭВМ по конструктивному исполнению в определенной степени условно. Имеется четкая граница только между стационарными и портативными машинами. В нашей стране наибольшее распространение получили лишь настольные ПЭВМ.

Настольная ППЭВМ конструктивно содержит:
— системный блок, включающий МП, ОП, адаптеры, порты, а также НГМД и НЖМД;
— дисплей;
— клавиатуру.

Системный блок ПЭВМ отечественного производства, как правило, разделен на 2 отдельных части.

Примеры конструктивного исполнения ПЭВМ приведены на рис. 1.3.

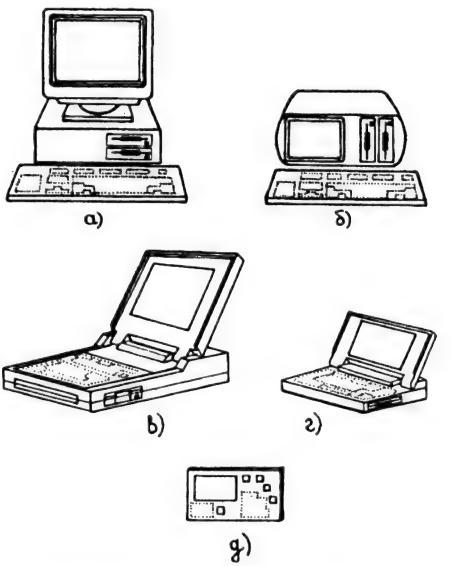


Рис. 1.3. Варианты конструктивного исполнения ПЭВМ:
а — настольная ПЭВМ, б — переносная ПЭВМ, в — наколенная ПЭВМ, г — блокнотная ПЭВМ, д — карманная ПЭВМ

1.5. Сферы применения ПЭВМ

ПЭВМ могут найти и уже находят применение во многих *областях* человеческой деятельности, за исключением тех из них, которые связаны преимущественно с физическим трудом. Сказанное не означает, что в последних не используются СВТ вообще. В этих областях применяются, главным образом, микроЭВМ, встроенные в роботы.

Области, в которых оказывается целесообразным использование ПЭВМ, покрываются тремя перечисленными в предыдущем подразделе *сферами* — бытом, обучением и профессиональной деятельностью. Мы заострим внимание читателя только на профессиональных приложениях персональных машин. В этом плане ПЭВМ обеспечивают автоматизацию труда той или иной категории работников, освобождая последних от выполнения рутинных операций. ППЭВМ с системами искусственного интеллекта, в частности, экспертными системами, помогают даже принимать решения в различных предметных областях.

Приведем далеко неполный перечень областей человеческой деятельности с ярко выраженной потребностью использования ПЭВМ:

— делопроизводство;
— бухгалтерская деятельность;
— медицина;
— научные исследования;
— инженерная деятельность;
— научная деятельность;
— издательская деятельность;
— социология;
— военное дело.

С помощью ППЭВМ задачи в различных областях можно классифицировать по следующим *типам* (в скобках указаны примеры областей):

1) обработка текстов (делопроизводство, издательская деятельность);
2) обработка графики (издательская деятельность);
3) работа с электронными таблицами (бухгалтерская деятельность, социология);
4) обслуживание баз данных (бухгалтерская деятельность);
5) научные и инженерные расчеты (соответствующие области);
6) разработка программного обеспечения для различных областей;
7) задачи искусственного интеллекта (медицина, военное дело).

Как уже отмечалось, путем оборудования ПЭВМ специализированными ПУ и применения соответствующего ПО ее можно превратить в АРМ низшего класса для автоматизации деятельности определенного типа, а также в управляющую машину для работы в реальном масштабе времени.

ПЭВМ в зависимости от потребностей и возможностей пользователей может использоваться либо автономно, либо в составе вычислительной сети, либо в качестве интеллектуального терминала более мощной ЭВМ.

Сеть обеспечивает общение ее пользователей (электронная почта), а также совместное использование дорогостоящих и/или уникальных аппаратных, программных и информационных ресурсов. Например, не выходя из кабинета, можно узнать о последних новостях, поработать с тем или иным изданием, обратиться к той или иной базе данных или распечатать большой объем информации на дорогостоящем лазерном принтере, установленном в другом помещении.

Играя роль интеллектуального терминала, ПЭВМ выполняет, главным образом, интерфейсные функции (взаимодействие с пользователем), а также предварительную обработку вводимой и окончательное формирование выводимой информации.

Широкое внедрение ПЭВМ как дешевых, компактных и надежных СВТ, не может не иметь важных *социальных последствий*. К ним, в первую очередь, относятся:

1) резкое увеличение творческого содержания человеческой деятельности;
2) нереализуемые ранее возможности распространения знаний и информации посредством, главным образом, сетей с подключенными к ним ПЭВМ;
3) уменьшение доли непосредственного человеческого общения как на работе, так и в быту;
4) повышение уровня сервиса в бытовой сфере (например, выбрать и купить тот или иной товар можно будет не выходя из дома);
5) всесторонний контроль и фиксация результатов профессиональной деятельности;
6) «раскрепощение» творческой деятельности, в частности, отсутствие необходимости работать на рабочем месте.

2. УСТРОЙСТВА ПЕРСОНАЛЬНЫХ ЭВМ

Этот раздел посвящен рассмотрению различных устройств ПЭВМ — как центральных, так и периферийных. Изложенный здесь материал, конечно, не претендует на исключительную полноту. Некоторые устройства, в частности, СПДП, конспективно описаны в подразделе 1.3 и автор считает этого достаточным. Ряд устройств, особенно периферийных, не упоминается вовсе, так как невозможно объять необъятное, что в полной мере относится к аппаратным средствам персональных компьютеров. Среди таких устройств, например, часы-календарь, которыми снабжаются ПЭВМ типа IBM PC AT. Их наличие обеспечивает постоянный счет и сообщение процессору текущего времени и даты, что позволяет при включении ПЭВМ производить установку времени автоматически. Часы-календарь питаются от аккумулятора, который автоматически подзаряжается при работе компьютера.

Ознакомившись с материалом данного раздела, читатель узнает, какие устройства могут быть в составе ПЭВМ, сможет понять, какие устройства ему нужны, а также будет в состоянии осуществить их выбор.

2.1. Микропроцессоры

Неформально, в развернутом виде, определение МП приводилось в подразделе 1.3. Более строго **микропроцессором** называют полупроводниковый кристалл или комплект кристаллов, на которых реализуется центральный процессор ЭВМ, т.е. совокупность арифметико-логического устройства и центрального устройства управления.

Остановимся сначала на исторических аспектах, связанных с появлением и развитием МП. Затем перечислим основные показатели качества МП и дадим определения, необходимые для дальнейшего изложения материала. После этого рассмотрим базовые модели МП, сосредоточив внимание главным образом на изделиях фирмы Intel, используемых в IBM-совместимых ПЭВМ. Более подробно в качестве примера разберем архитектуру МП 8086/88, на которую ориентирована ОС MS-DOS. Этот материал полезен не только узким специалистам-профессионалам, но и пользователям средней квалификации, так как большинство систем программирования для ПЭВМ обеспечивает доступ к аппаратным средствам машины. Потом кратко остановимся на перспективных RISC-микропроцессорах и специализированных МП, играющих роль акселераторов для основных МП в ПЭВМ. В конце подраздела наметим пути дальнейшего развития МП.

2.1.1. Эволюция микропроцессоров

Создание МП явилось следствием бурного прогресса в области электронной техники.

Первоначально в ЭВМ для передачи, хранения и преобразования информации, представленной двоичным кодом в виде электрических сигналов, использовались *электронные лампы*. Они обладали многими недостатками: выделяли большое количество тепла, потребляли много энергии, были громоздкими, дорогими и ненадежными.

Проблемы, связанные с недостатками электронных ламп, были решены в 1947 г. У.Шокли, Дж.Бардином и У.Браттейном. Работая в лаборатории компании Bell, они изобрели *транзистор*. Транзисторы выполняли те же функции, что и электронные лампы, но использовали электрические свойства полупроводников. В связи с появлением транзисторов отпала необходимость в приборе, в котором электрический сигнал передается с помощью заряженных частиц, движущихся в вакууме. Благодаря этому снизилось энергопотребление, уменьшились тепловыделение и размеры электронных узлов. Изобретение транзистора позволило сконструировать первую миниЭВМ.

В 1958 г. два американских инженера сделали следующий важный шаг в развитии элементной базы — создали *интегральную микросхему* (ИМС), содержащую на одном кремниевом кристалле соединенные между собой транзисторы, резисторы и конденсаторы. Этими двумя, причем независимыми авторами ИМС были Д.Килби из компании Texas Instruments и Р.Нойс, который основал впоследствии корпорацию Intel, являющуюся с тех пор флагманом в производстве МП. Первоначально ИМС включали только по несколько транзисторов; однако технология производства электронных компонентов развивалась столь стремительно, что сначала десятки, а затем сотни и более транзисторов стали размещаться на кремниевой пластине размером всего лишь с ноготь. С момента появления первой интегральной схемы была отлажена технология производства ИМС средней, большой и сверхбольшой степени интеграции. В 1987 г. фирма IBM открыла эру ИМС ультравысокой степени интеграции.

В 1969 г. компании Intel удалось создать комплект из четырех ИМС с полным набором элементов, характерным для процессора. Длина слова первого МП составляла всего 4 бита. В 1971 г. был выпущен 4-разрядный коммерческий МП Intel 4004, который стал применяться в микрокалькуляторах. Уже в 1972 г. появился 8-разрядный МП Intel 8008, а в 1974 г. — его улучшенный вариант Intel 8080. Он нашел применение в первых встроенных ЭВМ для управления производственными процессами. Вскоре появились и конкурирующие изделия — МП Z80 фирмы Zilog (его система команд включала набор команд 8080), 6802 фирмы MOS Technology, 6800, а затем и 6809 фирмы Motorola.

Развитие МП шло по разным направлениям, важнейшее из которых — увеличение разрядности. В этой области за довольно сжатый срок были достигнуты существенные результаты. Так, в 1976 г. фирма Texas Instruments выпустила 16-разрядный МП TMS9900, а в 1980 г. рынок МП пополнился 32-разрядными приборами компании Motorola MC68000. В настоящее время 32-разрядные МП производятся несколькими фирмами, причем на передовых рубежах находится фирма Intel, конкурируя с компанией Motorola. Наряду с 32-разрядными выпускаются и 64-разрядные МП для мощных АРМ. Первый однокристальный 64-разрядный МП 80860 создан фирмой Intel в 1989 г.

Подавляющее большинство производимых в настоящее время МП являются однокристальными, за исключением ряда 64-разрядных комплектов (разрядно-модульных МП).

2.1.2. Перечень основных технических характеристик и классификация микропроцессоров

МП характеризуются:
1) тактовой частотой;
2) разрядностью;
3) архитектурой.

Тактовая частота МП (более строго — тактовая частота, при которой способен работать МП) определяется максимальным временем выполнения элементарного действия в МП. Работа МП синхронизируется импульсами тактовой частоты от задающего генератора. Чем выше тактовая частота МП (при прочих равных условиях), тем выше его быстродействие. Более того, некоторые авторы под быстродействием понимают именно тактовую частоту, что совершенно неверно. Тактовые частоты современных МП колеблются в пределах единиц — десятков МГц.

Разрядностью МП называют максимальное количество разрядов двоичного кода, которые могут обрабатываться или передаваться одновременно. Понятие «разрядность» включает:
— разрядность внутренних регистров МП (m);
— разрядность шины данных (n);
— разрядность шины адреса (k).

Исходя из этого разрядность МП будем обозначать в виде m/n/k. Определяющую роль в принадлежности МП к тому или иному классу играет разрядность внутренних регистров (*внутренняя длина слова*). Мы уже использовали и будем иногда использовать только этот показатель. От разрядности шины данных *(внешней длины слова)* зависит скорость передачи информации между МП и другими устройствами. Например, для МП с разрядностью 16/16/20 скорость передачи информации (если нет других ограничений) в два раза выше, чем для МП с разрядностью 16/8/20. Однако в последнем случае упрощается управление шиной и предоставляется возможность использовать более простые ПУ. Разрядность шины адреса определяет *адресное пространство* МП, т.е. максимальное количество полей (обычно байтов, где 1 байт = 8 бит) памяти, к которым можно осуществить доступ. Очевидно, адресное пространство составляет 2^k, где ^ — операция возведения в степень, и при k=20 получим 1 Мбайт (1 Мбайт = 2^10 Кбайт = 1024 Кбайт, 1 Кбайт = 2^10 байт = 1024 байт). Реальная память машины может иметь меньший объем.

Архитектура МП является емким понятием, имеющим при этом, как и архитектура ЭВМ, неоднозначное толкование. **Архитектурой** часто называют организацию (в данном случае МП) с точки зрения пользователя. Описание архитектуры МП в таком понимании включает описание пользовательских возможностей программирования (в частности, состава регистров МП), системы команд, способов адресации, (логической) организации памяти, средств ввода-вывода и типов обрабатываемых данных. С этой точки зрения архитектуры МП считаются одинаковыми, если последние способны выполнять одни и те же программы. Реализации же одной и той же архитектуры могут отличаться одна от другой как на уровне физических компонентов аппаратных средств, так и на уровне способов реализации узлов МП. Иными словами, детали реализации, невидимые для пользователя, не оказывают влияния на архитектуру. В контексте же аппаратных средств под термином **«архитектура МП»** понимают принцип действия МП, конфигурацию (состав) и взаимное соединение основных его узлов.

Мы не будем отдавать предпочтение ни одному из альтернативных определений архитектуры, понимая ее в широком смысле. Ограничим наше рассмотрение следующими элементами архитектуры МП:

1) системой команд и способами адресации;

2) возможностью совмещения выполнения команд во времени;
3) наличием дополнительных устройств и узлов в составе МП;
4) режимами работы МП.

Система команд представляет собой совокупность команд, которые способен выполнять МП. Она включает полный список кодов операций, для каждой из которых указывается число операндов и допустимые способы их адресации. *Способы адресации* определяют технику вычисления адресов ячеек памяти (в которых хранятся операнды) и выполнения операций над адресными регистрами. Способы адресации разработаны и используются с целью обеспечения компактных адресных ссылок для случаев, когда машинный адрес имеет слишком большую длину и его неудобно включать в таком виде в команду, либо когда невозможно (например, при переборе последовательных ячеек памяти в цикле) или просто нет необходимости использовать явный адрес.

В соответствии с составом системы команд различают:

1) МП с *CISC-архитектурой* (CISC — complex instruction set computer — компьютер со сложной системой команд);

2) МП с *RISC-архитектурой* (RISC — reduced instruction set computer — компьютер с сокращенной системой команд).

МП первого типа являются традиционными, а их система команд включает большое количество команд для выполнения арифметических и логических операций, команд управления, пересылки и ввода-вывода данных.

МП второго типа появились сравнительно недавно. Их система команд упрощена и сокращена до такой степени, что каждая команда выполняется за единственный такт (по единственному тактовому импульсу). Такой подход позволяет упростить структуру МП и тем самым повысить его быстродействие. Традиционные операции при этом реализуются последовательностями элементарных машинных команд.

По функциональному назначению, определяемому системой команд, МП делятся на универсальные и специализированные.

Универсальные МП способны реализовывать любой алгоритм, который предварительно кодируется в системе команд данного МП. Они используются в качестве *основных процессоров* микроЭВМ. Следует отметить, что большинство универсальных МП аппаратно поддерживает только целочисленную арифметику. Арифметика же с плавающей точкой реализуется на них программно.

Специализированные МП служат для решения задач определенного класса. Приборы этого типа используются в качестве *сопроцессоров*, дополняющих основные процессоры, и выполняют роль акселераторов (ускорителей). Сопроцессор расширяет набор команд ЭВМ. Когда основной процессор получает команду, которая не входит в его рабочий набор, он передает управление сопроцессору с целью ее выполнения.

Подчеркнем, что деления МП на CISC и RISC, а также на универсальные и специализированные, находятся в разных плоскостях и никоим образом между собой не связаны. Однако в ПЭВМ в качестве основных используются универсальные CISC-МП, в то время как мощные АРМ оснащаются МП с RISC-архитектурой.

Некоторые наиболее развитые МП обеспечивают *совмещение выполнения нескольких последовательно расположенных команд во времени*, организуя конвейерную обработку. Эта архитектурная особенность оказывает заметное влияние на скорость выполнения линейных участков программ.

На кристалле МП могут быть размещены *дополнительные устройства*, в том числе:

1) система управления шиной;
2) кэш-память;
3) средства поддержки виртуальной памяти;
4) средства защиты памяти.

Дополнительные устройства способствуют повышению производительности ПЭВМ, уменьшению ее габаритов, предотвращению несанкционированного доступа и расширению функциональных возможностей МП (в том числе улучшению отладочных возможностей и обеспечению дополнительных режимов работы).

Кэш-память используется в качестве буфера между процессором и менее быстродействующей ОП. Действительно, при несогласованности быстродействия МП и ОП в случае обращения к последней МП будет простаивать один или несколько тактов, что снизит производительность ПЭВМ. При наличии быстродействующей кэш-памяти МП непосредственно обменивается информацией именно с ней. Эта память, в свою очередь, осуществляет обмен с ОП. При такой организации простой МП возможен только в случаях, когда требуемая им информация в кэш-памяти отсутствует (нужна подкачка в кэш из ОП) или когда выводимая им информация не может быть размещена в кэш-памяти по причине ее заполненности (требуется выгрузка содержимого кэша в ОП). Свойство локальности данных позволяет сократить время простоя МП. В простейшем случае в кэш-памяти сохраняется информация, извлеченная при последних обращениях к ОП.

Большинство современных МП обеспечивает управление *виртуальной памятью*. *Виртуальным* называется ресурс, который пользовательской программе представляется обладающим свойствами,

отличными от тех, которые он в действительности имеет. В случае виртуальной памяти пользовательской программе предоставляется возможность работать с адресным пространством, существенно превышающим объем реальной адресуемой памяти. Виртуальная память реализуется «совместными усилиями» аппаратных средств и ОС. При этом содержимое виртуальной памяти распределяется между внешней памятью (например, магнитными дисками) и ОЗУ. В последнем находится только часть содержимого виртуальной памяти, используемая в данный момент процессором. Если запрашиваемой области виртуальной памяти в данный момент в ОЗУ нет, то осуществляется подкачка из внешней памяти с возможной выгрузкой в нее части содержимого ОЗУ. Интересно то, что обо всем этом пользовательская программа ничего не «знает» (с точки зрения программиста виртуальная память существует в «действительности»). Такая организация памяти позволяет существенно ослабить требования к размеру программы и области ее данных, которые должны помещаться в ОП. Обмен между ОЗУ и ВЗУ может осуществляться *страницами* (областями фиксированного размера) или *сегментами* (областями переменной длины).

Средства защиты памяти позволяют предотвратить некорректное вмешательство пользовательской программы в области ОЗУ, занятые другими программами. Эти средства наряду с другими средствами являются крайне желательными в случаях, когда в ОЗУ требуется хранить одновременно несколько программ.

Архитектуры МП и ПЭВМ в целом определяют допустимые *режимы* работы машины. Важнейшим фактором при этом являются возможности МП. Использование же предоставляемых аппаратными средствами режимов зависит от свойств ОС. В настоящее время МП поддерживают широкий спектр *режимов работы*, среди которых:

1) однопрограммный режим;
2) многопрограммный режим;
3) система виртуальных машин.

В *однопрограммном* режиме в каждый момент времени может находиться в ОЗУ и выполняться только одна пользовательская программа. В *многопрограммном* режиме обеспечивается хранение в ОЗУ нескольких программ и попеременное их выполнение с той или иной дисциплиной обслуживания, что целесообразно главным образом (но не только) при возможности совмещения во времени счета в МП и операций ввода-вывода. На основе многопрограммного режима работы МП могут быть организованы *однопользовательский многопрограммный*, а при наличии соответствующих аппаратных средств — и *многопользовательский многопрограммный* режимы работы ПЭВМ. *Система виртуальных машин* является дальнейшим развитием мультипрограммирования, основной признак которого — возможность одновременной работы нескольких ОС.

В данном пункте мы неоднократно употребляли термин *«быстродействие МП»*. К определению этой характеристики в настоящее время относятся с большой осторожностью. Общепризнанной интегральной оценки быстродействия МП не существует. Часто для количественного выражения быстродействия приводят число коротких операций, которые могут быть выполнены в единицу времени. Под короткой операцией понимается простейшая команда типа сложения содержимого двух регистров. Реальные МП с CISC-архитектурой выполняют такие инструкции за 4 — 5 тактов (конечно, без учета конвейеризации). Поэтому, зная тактовую частоту МП, можно ориентировочно оценить его быстродействие. Мы будем использовать именно такую трактовку быстродействия.

В заключение отметим, что быстродействие МП определяется сочетанием почти всех рассмотренных его характеристик.

2.1.3. Основные модели микропроцессоров

Ограничим наше рассмотрение 16- и 32-разрядными МП с CISC-архитектурой, так как именно они используются в качестве основных МП ППЭВМ. Только лишь благодаря разработке 16-разрядных приборов микропроцессорная техника в основном достигла уровня, необходимого для построения ПЭВМ массового спроса. Появление 32-разрядных МП позволило создать ПЭВМ высшего класса для приложений, характеризующихся большим объемом вычислений.

Основные модели МП ведущей в этой области фирмы Intel и их технические характеристики приведены в табл. 2.1. Эти приборы используются в IBM-совместимых ПЭВМ. Все МП выпускаются на различные тактовые частоты, причем предельные тактовые частоты постоянно повышаются за счет совершенствования технологии. Чем выше тактовая частота МП, тем выше его стоимость и энергопотребление. Имеются также МП с переключаемой тактовой частотой. Оценки быстродействия весьма относительны и представлены для модификаций с максимально возможной для данной модели тактовой частотой.

Популярность МП фирмы Intel объясняется их высокими техническими характеристиками и поддержкой корпорацией IBM. В настоящее время аналогичные приборы выпускаются и другими производителями, среди которых достойны упоминания американские компании Harris Semiconductor и Advanced Micro Devices (AMD), а также японская корпорация NEC Information Systems.

Приборы 8086 и 8088 различаются только разрядностью шины данных. Они не обеспечивают в полной мере асинхронную работу ПУ и не содержат средств защиты памяти для разграничения доступа, а следовательно, используются главным образом в однопрограммном режиме. Отсутствуют и средства

Таблица 2.1

Основные технические характеристики микропроцессоров фирмы Intel

Модель (отечественный аналог)	Год начала выпуска	Количество транзисторов на кристалле, тыс.шт.	Максимальная тактовая частота, МГц	Быстродействие, млн. команд/с	Разрядность	Адресное пространство физической памяти	Наличие средств защиты памяти	Виртуальное адресное пространство на задачу	Наличие конвейеризации	Емкость кэш-памяти, Кбайт	В каких ПЭВМ используется (отечественные аналоги)
8086 (К1810ВМ86)	1978	70	16	3	16/16/20	1 Мбайт	нет	-	нет	-	IBM PC (8088), IBM PC XT (8088), IBM PS/2 Model 30 (8086) (ЕС1840, ЕС1841, СМ1810, ИСКРА 1030, НЕЙРОН И9.66, Истра 4816 - все на МП К1810ВМ86)
8088 (К1810ВМ88)	1979	70	10	2	16/8/20	1 Мбайт	нет	-	нет	-	
80286 (К1810ВМ86М)	1983	140	12	2,5	16/16/24	16 Мбайт	есть	1 Гбайт	есть	-	IBM PC AT, IBM PS/2 Model 50 и 60 (ЕС1842)
80386	1985	275	33	8	32/32/32	4 Гбайт	есть	64 Тбайт	есть	-	IBM PS/2 Model 80, IBM PS/2 Model 70, Compaq DESKPRO 386/xx (ЕС1851 - в перспективе)
80386SX	1988	275	20	5	32/16/24	16 Мбайт	есть	1 Гбайт	есть	-	IBM PS/2 Model 55SX
80386SL	1991	275	25	6	32/16/24	16 Мбайт	есть	1 Гбайт	есть	-	Zenith MastersPort 386SL
80486	1989	1160	50	27	32/32/32	4 Гбайт	есть	нет данных	есть	8	Apricot VX FT Server, IBM PS/2 Model 90 XP 486 и 95 XP 486, Compaq DESKPRO 486/50L
80486SX	1991	нет данных	25	14	32/32/32	4 Гбайт	есть	нет данных	есть	8	IBM PS/2 Model 90 XP 486SX и 95 XP 486SX

поддержки виртуальной памяти. Известны приборы с пониженным энергопотреблением (80C88 и 80C86), что важно для портативных ПЭВМ, а также с переключаемой тактовой частотой (8088-2). Аналогами МП 8086/88 являются отечественные изделия К1810ВМ86 и К1810ВМ88, а также приборы MSM80C86 и MSM80C88 американской фирмы Oki Semiconductor. Корпорация NEC выпускает МП V20 и V30, которые способны выполнять программы для приборов 8088/86, однако имеют по сравнению с ними расширенную систему команд и оригинальную двухшинную архитектуру, что обеспечивает более быстрое выполнение большинства команд. Улучшенными вариантами МП V20 и V30 являются приборы V40, V50 и V35. На их кристаллах дополнительно размещены базовые периферийные схемы.

Переходным между МП 8086/88 и 80286 был прибор **80186/80188**, который не получил широкого распространения. Он применялся лишь в нескольких моделях ПЭВМ.

МП 80286 свободен от упомянутых недостатков своих предшественников и имеет следующие режимы работы:

1) *незащищенный (реальный)* режим работы с физической памятью, в котором этот прибор эмулирует (имитирует) МП 8086, возможно, повышенного быстродействия, выполняя подготовленные для него программы;

2) *защищенный* режим (формально — защищенный режим виртуальной адресации), когда используются виртуальная память и механизм защиты памяти.

В защищенном режиме допустимо мультипрограммирование. Имеется возможность непосредственного подключения к ПЭВМ типа PC AT двух дополнительных консолей, чем обеспечивается возможность работы в многопользовательском режиме. Сменный же модуль ATtain 286 американской фирмы Corollary в состоянии обслужить 8 консолей. Этот модуль содержит МП 80286, двухпортовую память емкостью 1 Мбайт и два последовательных АИ. К ПЭВМ PC AT можно подсоединить до четырех подсистем ATtain 286. При этом она превращается в многопользовательскую мультипроцессорную микроЭВМ, способную одновременно обслуживать до 33 пользователей. Виртуальная память изделия 80286 имеет сегментную организацию.

Существует модификация прибора 80286 с пониженным потреблением энергии — 80C286. Фирмы Harris Semiconductor и AMD выпускают аналоги МП 80286, но с лучшими по сравнению с ним характеристиками (на 25 и 20 МГц соответственно). Корпорация NEC предлагает прибор V33, который характеризуется пониженным энергопотреблением, способен выполнять программы для МП 80286, но имеет расширенную систему команд, что не обеспечивает обратную совместимость. Отечественный прибор К1810ВМ86М также является аналогом изделия 80286, но неполным, так как выполняет некоторые команды последнего в режиме программной эмуляции.

В 1988 г. японская фирма VM Technology выпустила МП VM860S, в котором используется архитектура и технология виртуальных процессоров (VM). Он способен заменить 16-разрядные изделия фирмы Intel либо реализовать полностью или частично оригинальные наборы команд. Этот прибор имеет повышенное быстродействие и не содержит микрокода. Гибкость его архитектуры (в частности, системы команд) достигается программируемыми логическими матрицами, что обеспечивает аппаратную реализацию команд. Адресное пространство физической памяти МП VM860S составляет 512 Мбайт, виртуальное адресное пространство — 4 Гбайт, а разрядность — 16/16/29.

Среди производителей 16-разрядных МП с архитектурой, отличной от приборов фирмы Intel, можно назвать следующие американские компании:

1) National Semiconductor, выпускающую прибор NS32016;
2) Texas Instruments, предлагающую изделие TMS9900;
3) Zilog, поставляющую МП Z8000 и Z8003.

Шины адреса и данных 16-разрядных МП, как правило, являются *мультиплексированными*, т.е. совместно используемыми.

Большинство моделей 32-разрядных МП различных производителей характеризуются:

1) развитой конвейерной структурой;
2) наличием устройства управления виртуальной памятью на кристалле или средств поддержки внешнего аналогичного устройства;
3) схожестью структуры, несмотря на различия в системах команд;
4) наличием очереди команд либо кэш-памяти команд и/или данных.

Некоторые наиболее совершенные модели 32-разрядных МП имеют средства обработки чисел с плавающей точкой (вещественных чисел), а также немультиплексированные шины адреса и данных, что способствует повышению быстродействия.

Как уже отмечалось, первый 32-разрядный МП, но с 16-разрядной шиной данных, был создан фирмой Motorola в 1980 г. и назывался MC68000. Первый же полностью 32-разрядный однокристальный прибор NS32032 разработала компания National Semiconductor в 1982 г. Вскоре (в 1984 г.) появился и аналогичный прибор MC68020 фирмы Motorola. Фирма же Intel заметно отставала до тех пор, пока не выпустила **изделие 80386**. Желание и необходимость обеспечить его совместимость с приборами 8086 и 80286 не позволили разработчикам реализовать радикальные технические решения. Поэтому по ряду показателей МП 80386 уступает своим конкурентам. Несмотря на это, он пользуется высоким спросом на рынке, так как накоплен большой объем ПО для его предшественников. Кроме того, этот прибор, как и все МП фирмы Intel (что уже отмечалось), поддерживается фирмой IBM.

Итак, изделие 80386 отличается от прибора 80286 не столь радикально, как последний от 8086/88, и имеет развитую конвейерную структуру, средства поддержки виртуальной памяти с возможностью выбора сегментной, страничной или комбинированной организации, а также усовершенствованный механизм защиты памяти, который назван системой защиты конфигурации. Защита конфигурации гарантирует надежную работу машины при выполнении сложных прикладных программ (в частности, уменьшение степени повреждений, вызванных ошибками в ПО). Средства защиты МП 80386 проверяют каждую машинную команду на соответствие определенному набору требований. Степень необходимой защиты зависит от типа выполняемой прикладной программы. Разработчик пользовательской программы ограничен в выборе режимов защиты. Изделие 80386 характеризуется также аппаратным, а не программным переключением задач и расширенными средствами отладки программ за счет введения дополнительных команд, но отсутствием аппаратных средств для обработки чисел с плавающей точкой и встроенной кэш-памяти, а также мультиплексированностью шин адреса и данных.

Максимальный размер сегмента увеличен с 64 Кбайт до 4 Гбайт. Размер страницы составляет 4 Кбайт. В МП 80386 реализованы следующие виды адресации памяти: сегментно-страничная, сегментная, линейная и странично-линейная, причем механизм страничной памяти является подчиненным по отношению к механизму сегментной организации.

В МП 80386 предусмотрена возможность динамического изменения ширины шины данных.

Аналогично прибору 80286 МП 80386 может функционировать в реальном и защищенном режимах.

В *реальном* режиме процессор имитирует МП 8086, выполняя соответствующие программы с большей скоростью. При этом он (как и прибор 80286 в реальном режиме) работает в однопрограммном режиме и его адресное пространство также ограничено объемом 1 Мбайт.

Защищенный режим раскрывает все возможности МП 80386. Этот режим, в свою очередь, включает три подрежима:

— режим виртуального процессора 8086;
— защищенный режим МП 80286;
— собственно защищенный режим прибора 80386.

В *режиме виртуального процессора 8086* обеспечивается мультипрограммная работа с выполнением программ для МП 8086. Это по сути поддержка системы виртуальных машин, в чем и состоит главное качественное отличие прибора 80386 от 80286. Данный режим обеспечивается средствами «среды многопроцессорного исполнения» (multiple-execution environment), позволяющей параллельно выполнять разнородные программы, в том числе различные ОС. Адресное пространство каждого виртуального процессора 8086 ограничивается объемом 1 Мбайт. *Защищенный режим МП 80286* эмулирует работу

высокоскоростного процессора 80286, обеспечивая многопрограммную обработку и 16-Мбайт адресное пространство. *Собственно защищенный режим МП 80386* характеризуется 4-Гбайт адресным пространством, а также поддержкой многопрограммной работы.

В 1987 г. фирма Motorola выпустила новый МП MC68030 и довела его тактовую частоту в 1989 г. до 50 МГц. Он, безусловно, составляет серьезную конкуренцию прибору 80386.

Обратим внимание на выпуск в 1988 г. фирмой Intel гибридной модели 80386SX. Это вызвано тем, что компании Harris Semiconductor и AMD заметно опередили фирму Intel по производству высокоскоростных модификаций МП 80286. Прибор 80386SX является быстродействующей заменой для изделия 80286 практически при той же стоимости ПЭВМ. МП 80386SX имеет ту же внутреннюю архитектуру, что и 80386, и отличается от последнего только разрядностями двух шин. AMD в 1991 г. уже выпустила 25-МГц вариант МП 80386SX.

В том же 1991 г. фирма Intel стала предлагать изделие 80386SL, в основе которого лежит МП 80386SX. Прибор 80386SL предназначен для использования в блокнотных ПЭВМ и имеет следующие особенности:

— в его состав входят схемы управления периферийным оборудованием;
— габариты МП 80386SL на 40% меньше прибора 80386SX;
— МП 80386SL обладает пониженным энергопотреблением благодаря «управлению» питанием.

Очередной виток конкурентной борьбы в области 32-разрядных МП связан с появлением в 1989 г. новых приборов — 80486 (Intel) и MC68040 (Motorola). По степени готовности этих МП фирма Intel шла впереди.

Изделие 80486 является дальнейшим развитием МП 80386. Новое детище Intel дополнительно содержит:

— кэш-память команд и данных на 8 Кбайт;
— средства выполнения операций над числами с плавающей точкой (сопроцессор плавающей точки);
— средства для многопроцессорной обработки, в том числе несколько новых команд.

Устройство управления виртуальной памятью ориентировано на страничную организацию. Шины адресов и данных не мультиплексированы. В 1991 г. Intel сообщила, что ей удалось разогнать прибор 80486 до 100 МГц.

На одной и той же тактовой частоте МП 80486 работает примерно в 2 раза быстрее прибора 80386.

Недавно фирме Intel пришлось сделать откат назад и выпустить изделие 80486SX. МП 80486SX полностью повторяет прибор 80486, но не содержит сопроцессора плавающей точки. Причиной разработки 80486SX явилось то, что AMD стала предлагать аналоги изделия 80386 с повышенной тактовой частотой (до 40 МГц). 20-МГц МП 80486SX полностью заменяет 80386 (это относится как к функциональным возможностям, так и к стоимости).

Чтобы можно было легко отличить МП 80386/486 от 80386SX/486SX, к обозначению первых теперь добавлены справа буквы DX (80386DX и 80486DX).

В августе 1989 г. вице-президент фирмы Intel Д.Хаус сообщил о разработке МП 80586, который будет содержать 4 млн. транзисторов и появится на рынке в 1993 г. На конец 1995 г. планируется выпуск МП 80686 (22 млн. транзисторов), а к 2000 г. — прибор 80786, который будет работать с тактовой частотой 250 МГц и содержать около 100 млн. транзисторов. По словам Д.Хауса, в последнем будет параллельно «функционировать четыре обычных и два векторных процессора».

Доминирующая на японском рынке корпорация NEC проводит свою политику в области 32-разрядных МП. Ее прибор V60, выпущенный в 1986 г., был первым в мире изделием, оснащенным средствами обработки чисел с плавающей точкой. Модели же V70 (1987 г.) и особенно V80 (1989 г.) имеют очень хорошие характеристики.

Таким образом, в области 32-разрядных МП в настоящее время идет острая конкурентная борьба без отчетливо выраженного лидера. В области же 16-разрядных МП большой отрыв имеют приборы 8086, 80286 и их аналоги.

По объему сбыта 32-разрядных МП места различных фирм распределяются следующим образом:

1) Motorola;
2) Intel;
3) National Semiconductor (приборы NS32032, NS32332 и NS32532);
4) американская компания AT&T (изделие WE32100);
5) Zilog (МП Z80000).

Кроме того, в данном секторе рынка фигурируют следующие производители:

— японская фирма Hitachi (МП Micro 32, являющийся аналогом прибора MC68020);
— VM Technology (изделие VM8600S, имеющее те же свойства, что и VM860S).

Подчеркнем еще раз, что 32-разрядные МП различных фирм несовместимы между собой (имеют различные системы команд).

Группы европейских и японских производителей вынашивают планы создания мировых стандартов МП, каковыми стали изделия фирм Intel и Motorola (последние используются в ПЭВМ компании Apple Computer). Для этого необходимо, как минимум:

1) создать прибор с самыми высокими техническими характеристиками;
2) обеспечить перенос накопленного ПО;

3) убедить заказчиков в том, что все это всерьез и надолго, т.е. что они в обозримом будущем не сдадут свои позиции и не уступят лидерство другим фирмам.

В качестве японского вызова в связи с этим можно упомянуть проект TRON (The Real-time Operating-system Nucleus — ядро ОС реального времени) компаний Fujitsu, Hitachi и Mitsubishi, разработавших МП с быстродействием до 20 млн. команда/с. МП этой группы хорошо совместимы с ОС UNIX.

По сферам применения 32-разрядные МП можно разделить на следующие группы:

1) для АРМ и многопользовательских микроЭВМ (приборы WE32100, 80386, 80486, MC68020, MC68030, MC68040, NS32332, Micro 32, V70 и V80);
2) для систем реального времени (изделия NS32032 и NS32532);
3) для контроллеров, ПУ и встроенных микроЭВМ (модели NS32532, Z80000 и др.);
4) для многопроцессорных систем (МП 80486, MC68040 и NS32532);
5) для военной аппаратуры (прибор Z80000);
6) для ПЭВМ (изделия фирм Intel, Motorola и NEC).

2.1.4. Архитектура микропроцессора 8086/88

В состав прибора 8086/88 входят:
— микропрограммное устройство управления;
— арифметико-логическое устройство;
— программно-доступные регистры;
— регистр-указатель команд IP (Instruction Pointer) и другие внутренние регистры;
— блок формирования адресов;
— блок очереди команд;
— средства для поддержки прямого доступа к памяти.

МП обеспечивает 24 способа (режима) адресаций и выполнение 134 различных команд. Однако действия над числами с плавающей точкой аппаратно не поддерживаются и поэтому должны производиться программно или на дополнительном оборудовании (сопроцессоре плавающей точки).

Времена выполнения основных команд характеризуют следующие величины:
— минимальный цикл шины (последовательность микроопераций, необходимых для передачи порции информации между МП и ПУ или ОП) — 4 такта;
— копирование содержимого одного регистра в другой — 2 такта;
— сложение (при условии, что оба операнда находятся в регистрах МП и результат также помещается в регистр) — 3 такта;
— сложение (в случае, когда один операнд находится в регистре, другой — в ОП, а результат записывается в регистр) — 14 тактов;
— умножение (регистровое — аналогично третьему пункту) — 144 такта;
— деление (аналогично) — 177 тактов.

Состав *программно-доступных регистров* МП 8086/88 представлен на рис. 2.1. Всего имеется 13 таких регистров, разрядность которых равна 16 (2 байта). По функциональному назначению они делятся на:

1) регистры общего назначения (РОН);
2) сегментные регистры;
3) регистр флагов.

РОНы в принципе допускают произвольное использование в программах, однако обычно они имеют вполне определенное назначение, соответствующее их названию. Это связано с тем, что некоторые команды задействуют те или иные регистры без их явного указания; кроме того, используются стандартные соглашения для связи программных модулей. В РОНах могут храниться *данные* или *указатели адресов* (мы вскоре рассмотрим технику формирования 20-разрядного исполнительного адреса из 16-разрядных указателей адресов, т.е. из номера первого параграфа в сегменте и относительного адреса в пределах сегмента).

Информационные регистры стандартно выполняют следующие функции:

1) *регистр-аккумулятор* AX используется для временного хранения данных и промежуточных результатов;
2) *базовый регистр* BX служит для хранения указателя адреса области памяти, который участвует в формировании смещения при определенных режимах адресации;
3) *регистр-счетчик* CX предназначен для организации циклов (хранит число повторений тела цикла);
4) *регистр данных* DX служит в качестве вторичного аккумулятора для временного хранения данных и промежуточных результатов.

Любой информационный регистр доступен под тремя именами, например:

AX — имя всего 16-разрядного регистра;
AH — имя старшего байта регистра AX;
AL — имя младшего байта регистра AX.

Содержимое *индексных регистров* SI и DI при определенных режимах адресации (возможно, совместно с содержимым регистра BX) используется при формировании относительного адреса

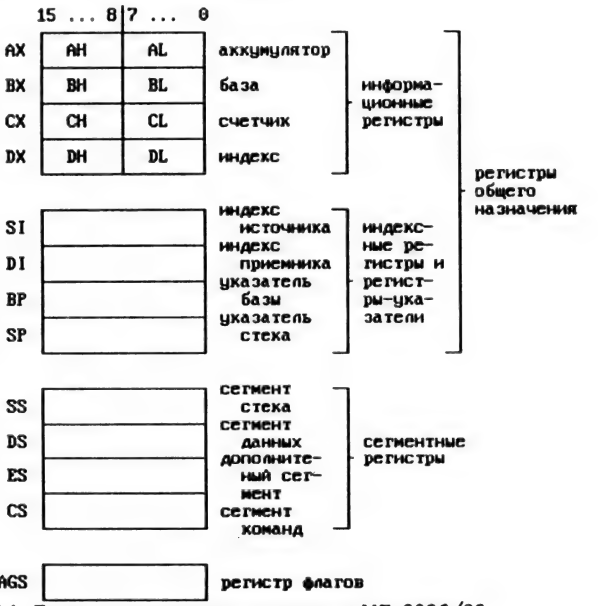


Рис. 2.1. Программно-доступные регистры МП 8086/88

для обращения к памяти. Разделение относительного адреса на две составляющие (базу и индекс) осуществлено в целях повышения эффективности выполнения операций над содержимым последовательных ячеек памяти (например, над элементами массива в цикле). Код в регистрах SI и DI при выполнении команды может автоматически возрастать или уменьшаться в режиме автоиндексации. Регистр SI применяется для индексации операнда, а регистр DI — результата.

Регистры-указатели BP и SP используются для работы со стеком, который служит, в частности, для передачи аргументов подпрограмме, а также возврата значений и управления в основную программу. Следует иметь в виду, что стек не организован аппаратно, а реализуется в определенной области ОЗУ. Содержимое регистра SP указывает на адрес элемента в вершине стека; регистр же BP используется аналогично регистру BX (в том числе совместно с регистрами SI и DI), только указывает на адрес в стеке, а не в области данных. Это необходимо для организации произвольного доступа к содержимому стека, что удобно, например, для выборки из него аргументов подпрограммы. Кроме того, содержимым регистра BP можно отмечать дно стека, если его область пересекается с областью данных.

Индексные регистры и регистры-указатели могут применяться и по другому назначению, если в данный момент они не используются стандартным образом.

Сегментные регистры играют важную роль в формировании физических адресов ОП (т.е. исполнительных адресов). В связи с тем, что внутренние регистры МП 8086/88 16-разрядные, напрямую можно адресовать только 2^16 = 64 Кбайт памяти (минимальная адресуемая единица — байт), этого для реальных приложений явно недостаточно. Для расширения адресного пространства до 1 Мбайт (поддерживается 20-разрядным адресом) в МП 8086/88 используется сегментная организация памяти. В соответствии с этим 1-Мбайт адресное пространство логически делится на 64 К **параграфов** по 16 байт. Поэтому для хранения номера параграфа оказывается достаточным 16-ти разрядов, причем этот номер содержится в сегментном регистре. **Сегмент** представляет собой логическую область памяти, состоящую из целого числа параграфов, начинающуюся с какого-либо параграфа и содержимое которой можно адресовать 16-разрядным кодом. Следовательно, размер сегмента не может превышать 64 Кбайт (4 К параграфов). *Исполнительный адрес* (см. рис. 2.2) формируется путем сложения двух составляющих:

1) 20-разрядного *адреса сегмента*, представляющего собой номер первого параграфа в сегменте, умноженный на 16;

2) 16-разрядного *относительного адреса* в рамках сегмента.

В действительности умножение номера первого параграфа в сегменте на 16 осуществляется путем сдвига содержимого сегментного регистра на 4 разряда влево. Очевидно, адрес сегмента всегда кратен 16-ти.

Сегментную организацию памяти иллюстрирует рис 2.3.

Сам же относительный адрес внутри сегмента формируется в соответствии с заданным в команде режимом адресации из указателей адресов, хранящихся в РОНах, и содержимого адресного поля команды.

МП имеет доступ одновременно к четырем сегментам:

1) *сегменту команд (программному сегменту)* CS, содержащему последовательность команд программы;

2) *сегменту данных* DS, хранящему данные для выполняемой программы;

3) *дополнительному сегменту* ES, обычно используемому для запоминания промежуточных результатов, т.е. в качестве дополнительной рабочей памяти;

4) *сегменту стека* SS, содержащему стек, в котором размещаются значения локальных переменных и через который осуществляется связь по информации между основной программой и подпрограммой, что уже отмечалось.

Рис. 2.2. Формирование 20-разрядного адреса

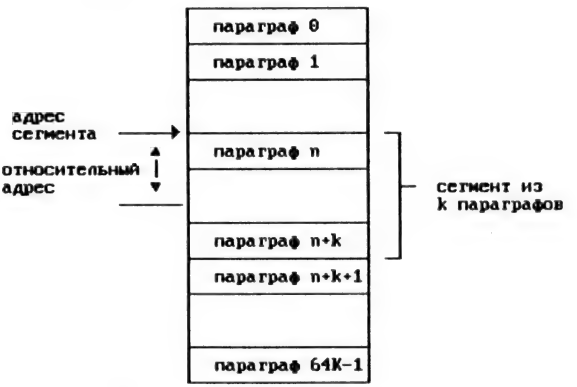


Рис. 2.3. Сегментная организация памяти

Тот или иной сегментный регистр задается в команде явно или предполагается неявно.

Различные сегменты могут перекрываться, совпадать или занимать непересекающиеся области памяти. На это нет никаких ограничений. Так, например, для простой программы все перечисленные сегменты могут начинаться с одного адреса. Для объемных программ, наоборот, может потребоваться использование большего количества сегментов. Поэтому система команд МП обеспечивает смену содержимого сегментных регистров и передачу управления из одного программного сегмента в другой (*дальнюю* передачу управления наряду с *ближней*). За счет сегментирования поддерживается также простая реализация перемещаемости программ в ОЗУ. Однако не следует считать, что сегментная организация памяти является панацеей от всех бед: в действительности именно она доставляет много хлопот программистам.

С точки зрения аппаратуры размер любого сегмента составляет 64 Кбайт. Меньший же размер может фиксироваться только в голове программиста путем ограничения области относительных адресов, за счет чего неиспользуемая часть сегмента может быть отведена под другой сегмент.

Иногда удобно считать, что память имеет *страничную организацию* с размером страницы, равным 64 Кбайт. Тогда старшие 4 бита 20-разрядного адреса указывают **страницу**, а остальные — относительный адрес в рамках страницы.

Как уже говорилось, минимальной адресуемой областью памяти является байт. *Слово* содержит 2 байта, причем младший байт слова помещается в однобайтовую ячейку с четным номером, а старший — в следующую однобайтовую ячейку с на единицу большим нечетным номером (учтите то, что байты адресуются с нуля). Аналогично размещается в памяти и 32-разрядное *двойное слово* (сначала — младшее слово, а затем — старшее). Это нужно всегда иметь в виду при низкоуровневой работе с памятью.

Последний из программно-доступных регистров — *регистр флагов* FLAGS — предназначен для фиксации различных сигналов, вырабатываемых в МП в ходе интерпретации команд (к примеру, знака результата выполненной в арифметико-логическом устройстве операции). Эта информация может использоваться для организации разветвлений в программе.

Регистр-указатель команд IP (традиционное название — *счетчик команд*) служит для хранения адреса очередной и формирования адреса следующей инструкции (конечно, в форме относительного адреса в пределах программного сегмента).

Блок формирования адресов обеспечивает аппаратное формирование исполнительного адреса по описанным выше принципам.

Для повышения производительности ПЭВМ МП 8086/88 содержит *блок очереди команд* на 6 байт. В этот блок осуществляется упреждающее считывание кодов команд, текстуально следующих за выполняемой. Подкачка очереди команд происходит асинхронно и одновременно с выполнением текущей команды. Это в совокупности с большой вероятностью последовательного выполнения команд дает ощутимый выигрыш по быстродействию. Описанная техника не работает только при реализации команды, передающей управление (очередь команд оказывается бесполезной).

Назначение *средств поддержки прямого доступа к памяти* описано в подразделе 1.3.

2.1.5. RISC-микропроцессоры

Долгое время считалось, что чем сложнее система команд процессора, тем лучше. Однако очевидно, что такой подход ведет к его усложнению и, следовательно, к усложнению технологии его производства. Это обстоятельство косвенно оказывает отрицательное влияние на быстродействие.

С середины 70-х гг. фирмой IBM, а затем и другими фирмами стали проводиться авангардистские работы совершенно в противоположном направлении — по упрощению системы команд с тем, чтобы каждая из них выполнялась за один такт. Эти исследования принесли положительные результаты, и в настоящее время выпускается ряд МП, построенных на RISC-основе. Идеал RISC-архитектур *(инструкция за один такт)* обычно не достигается. За выигрыш же в быстродействии приходится расплачиваться сложностью ПО. Достоинства RISC-архитектуры состоят в следующем:

— в упрощении МП, а следовательно, в снижении его стоимости;

— в повышении быстродействия МП благодаря его простоте и однородности структуры команд, что позволяет выполнять комбинацию из нескольких команд быстрее, чем одну эквивалентную этой комбинации сложную команду;

— в уменьшении размеров МП.

RISC-МП обычно используются в АРМ, а также иногда монтируются в платы-акселераторы для ПЭВМ. Кроме того, такие МП находят применение в ПУ и адаптерах.

Самым мощным в настоящее время кремниевым RISC-МП является уже упоминавшийся прибор 80860 фирмы Intel. Он представляет собой фактически микропроцессорный вариант суперЭВМ Cray-1 и содержит на одном кристалле 32-разрядный целочисленный процессор, 64-разрядный процессор плавающей точки, трехмерный графический процессор, кэш-память данных и команд, а также устройство управления памятью. Тактовая частота этого изделия составляет 40 МГц, а быстродействие — 33 млн. команд ЭВМ VAX фирмы DEC в секунду и 10 MFLOPS (Million Floating Point Operations per Second — млн. операций с плавающей точкой в секунду). Пиковое быстродействие МП 80860 характеризуется выполнением трех команд за один такт (в различных асинхронно функционирующих процессорах). Уже объявлен 50-МГц вариант данного изделия. Прибор 80860 несовместим с семейством МП 80x86 той же фирмы, но это не исключает возможности их совместного использования с общим полем памяти. Сейчас уже просматриваются следующие области применения МП 80860:

1) многопроцессорные системы, где приборы 80860 будут работать параллельно под управлением ОС UNIX;

2) высокопроизводительные графические АРМ, где МП 80860 будет использоваться в качестве основного процессора;

3) ПЭВМ на базе МП 80386 и 80486, где прибор 80860 будет играть роль акселератора, обеспечивая быстродействие большой ЭВМ.

Так, уже предлагается сопроцессорная плата Number-Smasher-860, содержащая МП 80860 и ОЗУ емкостью 8 Мбайт. Эта плата предназначена для ПЭВМ на МП 80286 — 80486 и имеет быстродействие 80 MFLOPS.

Среди других МП с RISC-архитектурой можно назвать:

1) 32-разрядный прибор Am29000 фирмы AMD, имеющий быстродействие 17 млн. команда/с и работающий на частоте 25 МГц;

2) 64-разрядный прибор Sparc-H (Sparc — Scalable Processor ARChitecture — расширяемая архитектура процессора) фирмы Fujitsu Microelectronics, характеризующийся быстродействием 25 млн. команда/с при тактовой частоте 25 МГц (полностью реализован RISC-подход);

3) 32-разрядный прибор 88100 из микропроцессорного набора 88000 фирмы Motorola, работающий на частоте 20 МГц и содержащий на кристалле целочисленный процессор, процессор плавающей точки, а также устройство управления памятью (кэш-память реализована в отдельной ИМС).

Интересен тот факт, что фирма AMD сообщила об отказе от выпуска МП Am29000 (видимо, в связи с жесткой конкуренцией в этой области).

Уникальными характеристиками обладает разработанный в 1988 г. 32-разрядный арсенид-галлиевый МП компании Texas Instruments, способный работать на частоте 100 МГц и имеющий шестиуровневую внутреннюю конвейерную архитектуру. Он создан по заказу Управления перспективных НИОКР Министерства обороны США (DARPA) для военных приложений. Планируется увеличение тактовой частоты данного прибора до 200 МГц.

2.1.6. Специализированные микропроцессоры

Существующие в настоящее время специализированные МП можно условно разделить на нижеприведенные группы:

1) *функционально-ориентированные* МП, предназначенные для реализации тех или иных функций;

2) *векторные* и *матричные* процессоры, обеспечивающие параллельное выполнение операций над регулярными (однородными) структурами данных;

3) МП, поддерживающие тот или иной *язык программирования* (возможно, параллельный).

Первые две группы специализированных МП используются главным образом в качестве сопроцессоров. Приборы же третьей группы могут применяться как в качестве сопроцессоров, так и в роли основных процессоров специализированных микроЭВМ.

Среди *функционально-ориентированных* МП выделяют:

— математические процессоры;
— графические процессоры;
— процессоры для поддержки баз данных;
— процессоры для ПУ и адаптеров.

Наибольшее распространение получили *математические* специализированные МП и особенно — приборы для выполнения операций над вещественными числами, используемые в качестве сопроцессоров плавающей точки. Сведения об основных математических специализированных МП представлены в табл. 2.2. Эти приборы позволяют существенно (в несколько десятков раз) снизить время выполнения операций, на которые они ориентированы.

Таблица 2.2

Основные математические специализированные МП

Модель основного МП	Математические МП: фирма, модель
8086/88	Intel, 8087
80286	Intel, 8087; Intel, 80287; Integrated Information Technology (ITT - США), 2C87; Zaiaz (США), плата 933 Computer Engine
80386	Intel, 80287; Intel, 80387; ITT, 2C87; ITT, 3C87; Weitek (США), Abacus 3167; Togai InfraLogic (США), FC110
80386SX	Intel, 80287; Intel, 80387SX
80486	Weitek, Abacus 4167
80486SX	Intel, 80487; Weitek, Abacus 4167
MC68020	Motorola, MC68881; Weitek, Abacus 3168
MC68030	Motorola, MC68882; Weitek, Abacus 3168
WE32100	AT&T, WE32106

Перечисленные в этой таблице сопроцессоры фирм Intel, Weitek, Integrated Information Technology (ITT) и AT&T служат для поддержки арифметики с плавающей точкой. Данные приборы характеризуются различными функциональными возможностями, быстродействием и стоимостью. Сопроцессоры фирмы Weitek дополняют соответствующие сопроцессоры фирм Intel и Motorola и могут работать во взаимодействии с ними. Следует отметить, что сопроцессоры Abacus функционируют в 1,7 — 4 раза быстрее аналогичных изделий фирмы Intel. Серьезный вызов фирме Intel сделала и созданная в 1987 г. компания ITT, выпустив совместимые с МП 80287 и 80387 изделия 2C87 и 3C87 той же стоимости, но с бо́льшим в 2 раза быстродействием на тех же тактовых частотах.

Плата 933 Computer Engine содержит 30-МГц RISC-процессор Clipper фирмы Fairchield Semiconductor и ОЗУ емкостью 4 — 32 Мбайт. Она обеспечивает быстродействие 5 млн. команда/с и 1,5 MFLOPS. Широкое распространение получили сопроцессоры фирм Intel, Motorola и Weitek. Несмотря на то, что МП 80486 оперирует с вещественными числами, компания Weitek все же предложила для него математический сопроцессор, который обеспечивает повышение быстродействия в 5 — 6 раз. Стоимость установки в ПЭВМ МП фирмы Weitek достаточно высока и составляет 1000 — 31300 долл., что превышает цену компьютера IBM PC XT.

Для отечественного МП К1810ВМ86 разработан сопроцессор плавающей точки К1810ВМ87.

Особое место среди математических специализированных МП занимает прибор FC110, предназначенный для решения задач нечеткой логики. Он позволяет примерно в 10 раз ускорить выполнение ключевого сегмента алгоритма логического вывода по сравнению с МП 80386.

Среди функционально-ориентированных специализированных МП других типов можно упомянуть графический сопроцессор 82786 фирмы Intel, аппаратно реализующий полиэкранный режим работы, когда каждое приложение может иметь окно на дисплее с текстовой и/или графической информацией, а также плату N:vector фирмы Nucleus International (США) для поддержки реляционной базы данных. Последняя позволяет получить 10-кратный выигрыш по быстродействию в сравнении с чисто программными решениями.

Группу *векторных* и *матричных* специализированных МП составляют следующие изделия:

1) плата Zip 3232-20 для семейства PC IBM фирмы Mercury Computer Systems (США), обеспечивающая быстродействие 20 MFLOPS;

2) матричный процессор Vortex компании Sky Computers для IBM PC AT, также обеспечивающий быстродействие 20 MFLOPS;

3) плата MC3200 для IBM PC AT фирмы Mercury Computer Systems с быстродействием 10 MFLOPS (содержит кэш-память, целочисленный МП и МП плавающей точки);

4) матричные процессоры RL800 и PL2500 для IBM PC AT компании Eighteen Eight Laboratories (США) для работы в режиме с плавающей точкой, обеспечивающие быстродействие до 25 MFLOPS;

5) плата акселератора с 64-разрядными сопроцессорами плавающей точки TMS320C30 фирмы Texas Instruments для ПЭВМ семейства PC IBM, обеспечивающая быстродействие 35 MFLOPS.

В качестве примеров специализированных МП, поддерживающих *языки высокого уровня*, можно назвать нижеперечисленные изделия:

1) *транспьютер* T800 британской фирмы Inmos, аппаратно реализующий параллельный язык программирования Occam и имеющий быстродействие 4 MFLOPS, что в 12 раз превышает быстродействие пары 80386 — 80387; его предшественник — T414 (1985 г.) — стал первым коммерческим МП, сделавшим реальностью параллельную обработку;

2) однокристальный LISP-процессор фирмы Texas Instruments, предназначенный для портативных микроЭВМ военного назначения с искусственным интеллектом;

3) плата PC400 фирмы Silicon Composers (США) для IBM PC XT/AT, выполняющая программы на языке FORTH и имеющая производительность 4 млн. команда/с.

2.1.7. Перспективы развития микропроцессоров

Создание 32-разрядных универсальных МП с CISC-архитектурой не привело к ликвидации спроса на приборы меньшей разрядности, так как они в достаточной степени удовлетворяют ряду приложений, в частности, обработке текстов. Поэтому было бы ошибочным утверждать, что перспективы развития микропроцессорной техники сводятся только к совершенствованию 32-разрядных приборов.

Эти перспективы характеризуются следующими тенденциями:

1) прекращением эволюции 8- и тем более 4-разрядных МП;

2) стабилизацией архитектур 16-разрядных МП и совершенствованием технологии их производства для повышения тактовой частоты (эти приборы достигли своей зрелости);

3) развитием архитектур и совершенствованием технологии производства 32-разрядных МП;

4) проведением работ по созданию 64-разрядных CISC-МП;

5) разработкой новых RISC-МП;

6) созданием и совершенствованием специализированных МП различных типов.

2.2. Основная память

ОП в плане ее назначения и обобщенной классификации уже кратко характеризовалась в подразделе 1.3. Здесь мы более подробно изучим эти вопросы.

На логическом уровне память представляет собой совокупность ячеек определенной разрядности (в настоящее время, как правило, 1 байт), к содержимому которых можно обращаться (по чтению или по записи) путем указания их адресов. ОП ПЭВМ состоит из одного или нескольких *устройств памяти*. Устройства памяти характеризуются следующими основными показателями:

1) временем доступа (быстродействием);

2) емкостью;

3) стоимостью;

4) потребляемой мощностью (энергопотреблением).

Время доступа — это промежуток времени, за который может быть записано или прочитано содержимое ячейки памяти после подачи ее адреса и соответствующего управляющего сигнала. **Емкость** определяет количество ячеек или битов в устройстве памяти. Однако этим показателем емкость ОП ПЭВМ не ограничивается, так как она может содержать несколько устройств памяти. **Стоимость** измеряется денежными затратами в расчете на единицу емкости памяти.

Классификация устройств ОП ПЭВМ по функциональному признаку представлена на рис. 2.4. Все устройства памяти базируются на полупроводниковой технологии и выполняются в виде ИМС. В связи с этим показатель «емкость» сводится по сути к удельной емкости (количеству ячеек или битов на кристалле).

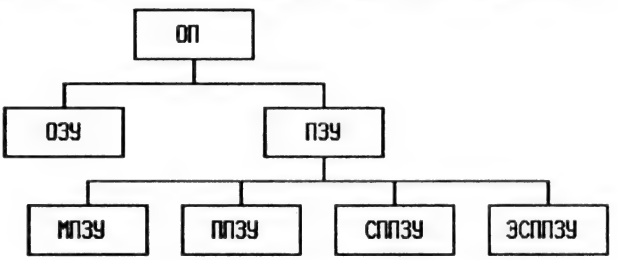


Рис. 2.4. Устройства основной памяти

В англоязычной литературе вместо ОЗУ используется термин RAM (Random-Access Memory — запоминающее устройство с произвольной выборкой — ЗУПВ). Под произвольностью выборки в данном случае понимают возможность непосредственного доступа к любой (произвольной) заданной ячейке памяти, причем время доступа для любой ячейки одинаково. Но термин ЗУПВ ничего не говорит о функциональных возможностях, а именно о том, что оно используется в качестве временной рабочей памяти машины, обеспечивая как запись, так и считывание информации (программ и данных).

В зависимости от способа хранения информации ОЗУ подразделяют на:

1) статические ОЗУ (SRAM — Static RAM);
2) динамические ОЗУ (DRAM — Dynamic RAM);
3) статические векторные ОЗУ (SCRAM — Static-Column RAM);
4) псевдостатические ОЗУ (P-SRAM — Pseudo-Static RAM).

В *SRAM* каждый бит информации (1 или 0) хранится на элементе типа электронной защелки (обобщение триггера), состояние которого остается неизменным до тех пор, пока не будет сделана новая запись в этот элемент или не будет выключено питание.

В *DRAM* же каждый информационный бит хранится в виде заряда конденсатора. Из-за наличия токов утечки заряд конденсатора необходимо с определенной периодичностью регенерировать (обновлять), причем во время регенерации запись новой информации должна быть запрещена. Обновление информации как правило проводится каждую миллисекунду (мс).

По сравнению со статическими, динамические ОЗУ имеют более высокую удельную емкость и меньшую стоимость, но большее энергопотребление и меньшее быстродействие. В настоящее время выпускаются кристаллы DRAM емкостью до 4 Мбит (512 Кбайт) и даже 16 Мбит (2 Мбайт) — эта заслуга принадлежит корпорации NEC, а SRAM — до 1 Мбит (128 Кбайт). Отечественные же кристаллы DRAM имеют емкость всего лишь до 256 Кбит. Стоимость одного Мбайт DRAM составляет 250, а SRAM — 500 долл. Время доступа для DRAM лежит в пределах 70 — 150 наносекунд (нс), в то время как SRAM — 45 — 55 нс. В связи с таким соотношением характеристик в качестве ОЗУ ПЭВМ сейчас используются, как правило, устройства динамической памяти. Устройства же статической памяти применяются главным образом для создания кэшей, устраняющих диспропорции по быстродействию между различными устройствами высокопроизводительных ПЭВМ.

Так, кэши могут создаваться для обмена информацией между скоростным МП, с одной стороны, и DRAM-ОЗУ, НМД, а также ПЗУ, с другой стороны. *Кэш* логически представляет собой промежуточный буфер, через который перекачиваются данные. Кэширование может осуществляться как по чтению, так и по записи. Выигрыш по быстродействию достигается благодаря тому, что часто используемые данные находятся в кэше, а поэтому доступ к ним со стороны МП ускоряется.

SCRAM занимает промежуточное положение между SRAM и DRAM, имея быстродействие динамической памяти при произвольном и статической памяти при последовательном доступе, а также среднюю между этими двумя типами ОЗУ стоимость.

P-SRAM также созданы для того, чтобы в определенной степени скомпенсировать недостатки как DRAM, так и SRAM. P-SRAM представляет собой DRAM, имеющее встроенную схему регенерации заряда каждой ячейки памяти. Это позволяет снизить энергопотребление по сравнению с DRAM при стоимости меньшей, чем SRAM. Такие характеристики позволяют использовать P-SRAM в качестве основной памяти ПЭВМ, и особенно портативных. Снижение энергопотребления ОЗУ при невысокой стоимости, конечно, очень актуально для последних при питании от батарей, поскольку *основная память* потребляет около четверти мощности, необходимой для работы компьютера. Удельная емкость P-SRAM эквивалентна DRAM. Так, фирма Hitachi уже представила микросхемы P-SRAM емкостью 4 Мбит.

ОЗУ всех четырех рассмотренных типов являются *энергозависимыми*. Тем не менее имеется и в настоящее время иногда используется способ устранения этого недостатка, состоящий в применении батарейного (или аккумуляторного) питания. Конечно, есть и альтернатива — использовать в качестве ОЗУ устройства памяти, основанные на других физических принципах.

ПЗУ (ROM — Read-Only Memory) делятся на следующие типы:

1) неперепрограммируемые ПЗУ — масочные ПЗУ (MROM — Masked ROM — МПЗУ);
2) программируемые ПЗУ (PROM — Programmable ROM — ППЗУ);
3) стираемые программируемые ПЗУ (EPROM — Erasable Programmable ROM — СППЗУ);
4) электрически стираемые программируемые ПЗУ (EEPROM — Electrically Erasable ROM — ЭСППЗУ).

В дословном переводе ROM означает запоминающее устройство только для чтения, что полностью соответствует выполняемым им функциям. Но, кроме того, ПЗУ является и памятью с произвольной выборкой в рассмотренном выше смысле. Поэтому, на наш взгляд, сочетание терминов ROM и RAM в англоязычной литературе не совсем удачно, однако широко распространено по историческим причинам. Все рассматриваемые типы ПЗУ являются *энергонезависимыми*.

МПЗУ — это запоминающее устройство, информация в котором «зашита» по специальному шаблону или маске при его изготовлении и изменению не подлежит.

ППЗУ же предоставляет пользователю возможность самостоятельно его запрограммировать. Содержимое ППЗУ формируется после того, как устройство памяти изготовлено. Процесс занесения информации в ППЗУ осуществляется специальным устройством, называемым *программатором*. Обычно этот процесс основан на пережигании плавких перемычек и необратим в том смысле, что после такого программирования содержимое памяти не может быть изменено. ПЗУ данного типа получили широкое распространение.

СППЗУ обеспечивает возможность неоднократного изменения своего содержимого путем стирания информации с помощью интенсивного ультрафиолетового излучения и последующей записи новой информации посредством специального программатора. Стирание СППЗУ осуществляется за 10 — 15 мин. В качестве примеров СППЗУ можно назвать кристаллы емкостью 4 Мбит (512 Кбайт) фирм Toshiba и NEC, обеспечивающие время доступа 150 и 120 нс соответственно. Очевидно, эти характеристики сопоставимы с DRAM.

Дальше всего от традиционного понимания ПЗУ отстоит *ЭСППЗУ*. Оно предоставляет возможность стирать свое содержимое не ультрафиолетовыми лучами, а электрическими сигналами. Но, конечно же, время перезаписи информации в этом устройстве существенно выше, чем в ОЗУ, что не позволяет его использовать в качестве последнего. Однако ЭСППЗУ можно считать заменителем НМД небольшой емкости. К примеру, еще в 1987 г. фирма Seeq Technology выпустила кристаллы ЭСППЗУ емкостью 1 Мбит (128 Кбайт).

2.3. Системные шины

СШ называется среда передачи сигналов (группа линий электрических соединений), к которой могут параллельно подключаться несколько устройств ЭВМ и через которую осуществляется передача данных, а также управляющих сигналов между ними. Синонимом термина «шина» является термин «магистраль». Как уже отмечалось в подразделе 1.3 и пункте 2.1.3, СШ содержит мультиплексированные или немультиплексированные линии адресов, данных и управления. СШ обеспечивает три вида передачи данных:

1) между МП и ОП;
2) между МП и ПВВ;
3) между ОП и ПВВ (при использовании СПДП).

Широко используемой в ПЭВМ СШ является шина **Multibus I,** первоначально разработанная фирмой Intel для машин на базе МП 8086/88. Ее адресные линии задействуются одновременно и как линии данных. Позднее эта шина была доработана для использования в ПЭВМ на базе МП 80286 и получила название **ISA** (Industry Standard Architecture — стандартная промышленная архитектура), или просто AT-шина потому, что прибор 80286 применяется в ПЭВМ IBM PC AT. Отечественным аналогом шины Multibus I является СШ И-41.

Для машин с МП 80386 пока нет общепризнанного стандарта СШ. Действительно, ряд производителей разработали собственные шины (в частности, фирма Intel — **Multibus II**). Фирма же IBM в 1987 г. предложила свою шину **MCA** (Micro Channel Architecture — архитектура «Микроканал») и стала использовать ее в своем новом семействе ПЭВМ — PS/2. Шина MCA защищена блоком патентов, которые трудно обойти, а лицензии фирмой IBM на ее производство другим изготовителям практически не выдаются. Наличие этой шины как раз и является основной отличительной особенностью семейства PS/2. Данная шина несовместима с шиной семейства машин PC IBM, что требует определенных затрат по совместимости на более высоком уровне. В январе 1989 г. группой известных фирм окончательно сформулированы технические условия на новую 32-разрядную шину под названием **EISA** (Extended ISA — расширенная промышленная стандартная архитектура), которая является просто дальнейшим расширением шины для МП 80286. Это как раз и обеспечивает хорошую совместимость ПЭВМ, а также возможность использования различного периферийного оборудования. Какая из MCA и EISA шина будет

доминировать в будущем — сказать пока трудно. В настоящее же время между приверженцами этих шин наблюдается жесткая конкуренция. Из-за неочевидности будущего стандарта на 32-разрядную шину ряд производителей, в том числе хорошо известные американская фирма Tandy и итальянская компания Olivetti, выпускают ПЭВМ, совместимые с PS/2, и одновременно работают над созданием машин на базе EISA. Первую ПЭВМ с шиной EISA в конце 1989 г. выпустила фирма HP. Она называется Vectra 486 PC и выполнена на базе МП 80486. Уже в 1990 г. ряд производителей предложили ПЭВМ с шиной EISA.

Шины MCA и EISA характеризуются высоким быстродействием, повышенной надежностью и простотой задания конфигурации ПЭВМ программными средствами вместо микропереключателей. Дополнительно к этому шина EISA позволяет использовать существующие для ПЭВМ типа PC XT и AT платы расширения ресурсов.

В 1989 г. фирма Wells American предложила шину **CompuStar**, которая может быть сконфигурирована для любого из четырех МП: 8086, 80286, 80386SX или 80386. Это еще более усилило конкурентную борьбу.

2.4. Внешние запоминающие устройства

ВЗУ являются важной составной частью ПЭВМ, обеспечивая долговременное хранение программ и данных на различных носителях информации. Наибольшее распространение в ППЭВМ получили НМД (дисководы), которые являются основными ПУ. Любая ППЭВМ содержит как минимум один накопитель на *гибких магнитных дисках* (НГМД) и, зачастую, накопитель на *жестких магнитных дисках* (НЖМД). Если же НЖМД отсутствует, то обычно имеется два НГМД. При наличии в составе ПЭВМ только НГМД работа на ней становится неудобной из-за ограниченности емкости *гибких дисков* (ГД). Для отечественных же ПЭВМ встраивание НЖМД считается почти непозволительной роскошью.

НГМД и НЖМД имеют много общего, так как используют один и тот же носитель информации — *магнитный диск* (МД). Поэтому мы вначале достаточно полно обсудим устройство НГМД и ГД и их характеристики; а затем остановимся только лишь на особенностях НЖМД. Будут представлены сведения и о магнито-оптических запоминающих устройствах.

После этого рассмотрим дополнительные ВЗУ, такие, как НОД, использование которых в ПЭВМ наконец-то стало реальностью, а также НМЛ, которые применяются давно, но не получили в персональных машинах широкого распространения.

Потом кратко охарактеризуем устройства памяти на новых физических принципах.

В завершение подраздела раскроем иерархию памяти ПЭВМ и приведем обобщенные сравнительные характеристики различных ее видов.

2.4.1. Накопители на гибких магнитных дисках

НГМД является устройством со сменным носителем информации. Он был впервые разработан в фирме IBM для загрузки диагностического ПО, причем диаметр ГД составлял 8 дюймов (203 мм). В настоящее время НГМД используются во всех ППЭВМ, да и не только профессиональных, а также в мини- и суперминиЭВМ.

В ПЭВМ ГД *(дискеты)* служат для долговременного хранения программ, которые при необходимости загружаются в ОЗУ для выполнения, и данных, используемых или формируемых выполняемыми программами. В связи с появлением НЖМД эта функция приобрела характер вспомогательной. Вместе с тем ГД как сменные носители информации продолжают играть в ПЭВМ важнейшую роль. Сменные носители информации необходимы для решения таких задач, как:

— *резервирование* (*дублирование*) информации;
— *обеспечение конфиденциальности* данных;
— *транспортирование* информации;
— *распространение* ПО.

Более емкие и быстродействующие НЖМД, как правило, не обеспечивают смену МД, а следовательно, не могут использоваться для выполнения перечисленных функций.

НГМД называют ПУ, в которое устанавливается ГД и которое обеспечивает как считывание, так и запись информации на него. Принцип записи заключается в намагничивании участков поверхности диска, что распознается при считывании.

НГМД состоит из следующих узлов:

1) *механического привода*, обеспечивающего вращение ГД;
2) *блока магнитных головок* чтения/записи;
3) *системы позиционирования* магнитных головок, которая служит для их перемещения относительно поверхности дискеты в радиальном направлении;
4) *электронного блока*, обеспечивающего управление накопителем и преобразование сигналов.

Двигатель привода включается только при доступе к накопителю по чтению или по записи. Следовательно, диск непрерывно не вращается.

НГМД имеет щель для установки дискеты. После ввода ГД эта щель обычно закрывается откидной заслонкой или дверцей. НГМД традиционно монтируются в системный блок ПЭВМ, однако выпускаются и такие дисководы, которые подключаются к машине с помощью кабеля (внешние НГМД).

Количество магнитных головок зависит от обслуживаемого числа рабочих поверхностей дискеты. В данный момент большинство НГМД являются двухсторонними. Имеется возможность в двухстороннем дисководе работать с односторонней дискетой, а в одностороннем — с двухсторонней, но только с единственной ее поверхностью.

Для подключения НГМД к ПЭВМ служит устройство, обычно называемое **контроллером**, а не адаптером (хотя мы об этом в подразделе 1.3 и не говорили).

Устройство дискеты показано на рис. 2.5. Она содержит гибкий пластмассовый диск (1) с магнитным покрытием, который помещен в пластиковый конверт (2) квадратной формы толщиной 2 — 3 мм. В этом конверте обеспечивается возможность вращения диска, сам же конверт выполняет защитные функции. На внутренние стороны конверта наклеены специальные подкладки, которые препятствуют повреждению ГД, уменьшают трение при его вращении, а также очищают поверхность диска от пыли и остающихся частиц магнитного покрытия. Информация считывается с диска (записывается на диск) через окно (3) овальной формы, вырезанное в конверте (для двухсторонних дисков — с двух сторон). Магнитные головки непосредственно соприкасаются с поверхностью диска, что сокращает срок службы последнего. На одной из боковых сторон конверта имеется прямоугольный вырез (4) разрешения записи. Открытый вырез означает, что для диска разрешены как чтение, так и запись. При заклеивании выреза светонепроницаемой полоской запись на диск аппаратно запрещается, что препятствует непреднамеренному, ошибочному или умышленному изменению информации на нем. Обойти эту защиту невозможно, если не использовать специальный дисковод. Многие производители поставляют ПО на дискетах без вырезов. Диск и конверт имеют круглые отверстия (5) для привода и круглые отверстия (6) небольшого диаметра для определения начала оборота. Начало оборота ГД распознается НГМД при совмещении отверстий 6 диска и конверта. Для придания большей механической прочности диск вокруг внутреннего отверстия усилен специальными металлическими кольцами (7). Две небольшие выемки (8) в конверте служат для фиксации дискеты в дисководе.

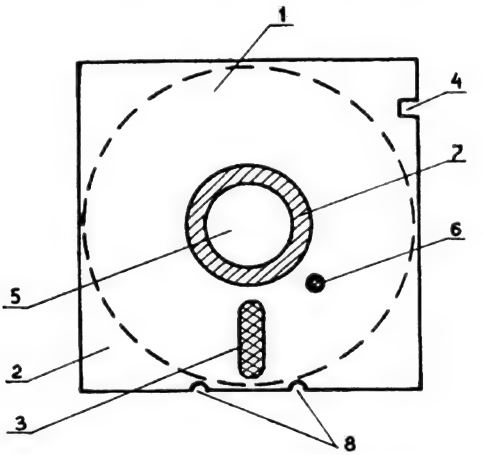


Рис. 2.5. Устройство дискеты

Информация на ГД размещается вдоль концентрических окружностей, называемых **дорожками**. Дорожки с одинаковыми номерами на различных поверхностях диска (в общем случае — пакета дисков) образуют **цилиндр**. Доступ к информации, записанной в одном цилиндре, осуществляется без перемещения магнитных головок. Каждая дорожка содержит определенное число секторов. Под **сектором** понимают участок дорожки МД, хранящий минимальную порцию информации, которая может быть считана с диска или записана на него. Каждый сектор имеет уникальный адрес. Между секторами имеется межсекторный интервал. Для ГД емкость сектора обычно составляет 512 байт. Описанную схему размещения информации на дисках иллюстрирует рис. 2.6. Более детально этот вопрос будет обсуждаться в подразделе 5.12.

В настоящее время используются четыре различных метода записи информации на МД.

Качество НГМД определяется совокупностью нижеприведенных характеристик:

1) поддерживаемой *емкостью* носителя информации (дискеты);

2) *диаметром* носителя информации;

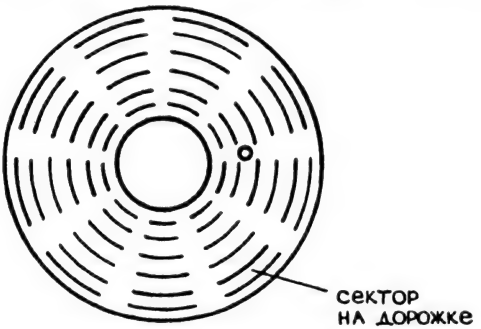


Рис. 2.6. Схема размещения информации на дисках

3) *продольной* (вдоль дорожки) *плотностью* записи информации, измеряемой числом бит на единицу длины или количеством секторов стандартного размера на дорожке;

4) *радиальной* (по радиусу диска) *плотностью* записи информации, измеряемой числом дорожек на единицу длины или количеством дорожек на поверхности диска определенного диаметра;

5) *средним временем доступа*, являющимся суммой среднего времени позиционирования (установки) блока магнитных головок на дорожке и среднего времени ожидания, требуемого для подхода к головкам нужного сектора;

6) *временем перемещения блока магнитных головок на соседнюю дорожку*;

7) *скоростью чтения/записи*;

8) *габаритными размерами*;

9) *массой*;

10) *энергопотреблением*;

11) *стоимостью*.

Конечно, многие из перечисленных технических характеристик не являются независимыми.

Различают *емкость неформатированного диска* и *емкость после форматирования* (разметки), которая несколько меньше предыдущей. Мы будем говорить, как правило, о той емкости, которая доступна пользователю, т.е. о емкости отформатированного диска.

Характеристики, указанные в пп. 8 — 10, для стационарных ПЭВМ решающей роли не играют. Однако они чрезвычайно важны для портативных машин.

В настоящее время в ПЭВМ широко используются НГМД, поддерживающие дискеты, характеристики которых представлены в табл. 2.3. Основные параметры дискеты (тип и диаметр в дюймах) указываются на ее конверте.

Таблица 2.3

Основные характеристики дискет

Тип	Диаметр, мм	Емкость	Число дорожек (цилиндров)	Число секторов на дорожке
DS/DD	133	360 Кбайт	40	9
DS/DD	133	720 Кбайт	80	9
DS/HD	133	1,2 Мбайт	80	15
DS/DD	89	720 Кбайт	80	9
DS/HD	89	1,44 Мбайт	80	18

DS - двухсторонние (double sided)
DD - двойной плотности (double density)
HD - высокой плотности (high density)

Следует отметить, что фирма IBM игнорирует стандарт дискет DS/DD диаметром 133 мм (5,25 дюйма) емкостью 720 Кбайт. Однако дисководы такого типа широко распространены. Так, производители IBM-совместимых ПЭВМ (в том числе в СНГ) предлагают модели, оборудованные японскими НГМД для работы с этим форматом дискет.

Дискета меньшей емкости, как правило, может использоваться в дисководе, рассчитанном на большую емкость (но нужны соответствующие драйверы); обратное же утверждение в общем случае неверно. Например, не удастся использовать диск HD в НГМД типа DD.

Подчеркнем, что для первых двух форматов записи применяются одни и те же дискеты DS/DD, но различные дисководы.

Наибольшей популярностью в мире пользуются НГМД для 89-мм (3,5-дюймовых) дисков, первоначально разработанные японской фирмой Sony. Фирма IBM считает их стандартными для моделей PS/2. На отечественные ПЭВМ устанавливаются пока только 133-мм дисководы.

Фирма Sony уже выпустила 51-мм НГМД, которые начинают применяться в портативных ПЭВМ. Такой дисковод работает с дисками повышенной емкости (800 Кбайт на 50-ти дорожках при односторонней записи), имеет массу 200 г и размер 66x25x96 мм. Время доступа невелико благодаря повышенной скорости вращения диска (3600 оборотов в мин. вместо стандартной скорости 360 оборотов в мин.). Каждая дорожка имеет 4 физических сектора по 4096 байт, которые могут использоваться как логические секторы размером 256, 512 или 4096 байт. Масса диска составляет всего навсего 8,7 г.

Находят применение также 63,5-мм дисководы.

Среднее время доступа для ГД зависит от скорости перемещения магнитных головок, размера диска и скорости вращения диска. Оно намного выше аналогичного показателя для ОП и лежит в пределах 200 — 1000 мс. *Скорость чтения/записи* невысока и составляет 30 — 40 Кбайт/с.

Габариты НГМД имеют немаловажное значение при размещении их в системном блоке ПЭВМ. Так, например, 133-мм дисководы обладают длиной 203 мм, шириной 146 мм и высотой 82 мм. В последние годы появились устройства с так называемой половинной высотой (их высота в два раза меньше, чем у обычных устройств). Это позволило размещать два дисковода один над другим вместо одного.

Емкость ГД не удовлетворяет пользователей ПЭВМ, в связи с чем фирмы-разработчики НГМД предпринимают усиленные попытки ее увеличения, которые дали ощутимые результаты. Определились следующие два основных направления работ: *увеличение емкости за счет совершенствования подсистем НГМД* и *использование новых принципов записи информации*.

В рамках *первого* направления можно выделить два следующих подхода.

Известно, что в настоящее время применяются дискеты, радиальная плотность записи на которых ограничивается не их возможностями, а возможностями НГМД, в частности системы позиционирования и блока магнитных головок. Если бы удалось устанавливать магнитные головки с большей точностью, то удалось бы и увеличить емкость существующих ГД. Ряд производителей дисководов решили эту задачу. Так, японская фирма Y-E Data еще в 1987 г. выпустила 133-мм НГМД емкостью 2,4 Мбайт информации, и 89-мм НГМД емкостью 2 Мбайт для обычных 133-мм и 89-мм дискет DS/HD соответственно. Очевидно, такой подход не сможет привести к качественному скачку в повышении емкости НГМД.

Специалистам фирмы Iomega удалось повысить емкость диска на другой основе. В разработанном ими накопителе быстро вращающийся ГД огибает магнитную головку, что обеспечивается возникающими при этом потоками воздуха. Уравнения для описания последних предложил швейцарский математик Д.Бернулли еще в XVIII веке. В результате такого аэродинамического эффекта между головкой и поверхностью ГД образуется зазор. Именно стабильность зазора и позволила повысить плотность записи, а следовательно, емкость диска. Данный подход уже позволил создать 133-мм накопитель Bernoulli Box II 44, поддерживающий хранение на специальной кассете 42,4 Мбайт информации, а также дисковод Bernoulli Transportable 90 на 90 Мбайт. Накопители данного типа обладают повышенной надежностью в работе и высоким быстродействием (приближающимся или сравнимым с быстродействием НЖМД), а сам диск — долговечностью, так как он никогда не соприкасается с магнитной головкой. Однако дисководы типа Bernoulli весьма дороги. Так, изделие Transportable 90 стоит 1099 долл.

В рамках *второго* направления разрабатываются дисководы, основанные на *магнитооптическом методе записи*, который состоит в следующем. На ГД располагаются нестираемые серводорожки, при помощи которых оптическая сервосистема с замкнутой обратной связью осуществляет точное позиционирование магнитных головок на дорожке (даже при деформации диска). Запись же и считывание информации осуществляется обычным способом. Такие дисководы позволяют существенно повысить радиальную плотность записи, но при этом нужны специальные гибкие магнитооптические диски. По описанному пути пошла фирма-разработчик стандартной 133-мм дискеты Insite Peripherals, выпустив 89-мм НГМД емкостью 20,8 Мбайт со средним временем доступа 65 мс. Скорость вращения диска составляет 720 оборотов в мин., а скорость чтения/записи — 1,6 Мбайт/с. В недалеком будущем планируется выпуск и 100-Мбайт НГМД. Уже в 1989 г. американская фирма Brier Technology объявила о разработке 89-мм магнитооптического дисковода, на котором можно записать 40 Мбайт информации.

2.4.2. Накопители на жестких магнитных дисках

В ПЭВМ используются главным образом НЖМД с *несъемными* МД. Поэтому накопители данного типа служат исключительно для хранения подлежащих выполнению программ и информационных массивов без возможности их непосредственного переноса на другие машины (требуется предварительная перезапись на дискеты). Не обеспечивается в должной мере и конфиденциальность информации. Однако НЖМД существенно превосходят НГМД как по емкости, так и по быстродействию.

НЖМД с несъемным носителем информации имеет ту особенность, что МД, система позиционирования и блок магнитных головок помещены в герметично закрытый корпус. Сам диск обычно имеет металлическую основу. Такой НМД был впервые применен фирмой IBM в 1973 г.

Герметизация диска в накопителе позволила добиться качественного улучшения его характеристик благодаря идеальной чистоте рабочих поверхностей носителя информации. С учетом того, что в НЖМД устанавливаются 2 — 4 жестких МД с металлической основой, их суммарная емкость

колеблется в пределах 5 Мбайт — 2 Гбайт, среднее время доступа составляет 4 — 100 с, а скорость чтения/записи — 0,5 — 1 Мбайт/с. Скорость вращения МД велика — как правило, 3600 оборотов в мин. Диаметр дисков обычно составляет 133 или 89 мм. Существуют также 63- и даже 46-мм НЖМД, предназначенные для портативных машин.

Пакет жестких МД имеет 306, 612 или более цилиндров (дорожек на каждой рабочей поверхности), а на каждой дорожке размещается 17 или большее количество секторов.

НЖМД с несъемным носителем информации часто называют *винчестерскими накопителями* или просто *винчестерами*. Причина этого кроется в том, что первые такие устройства рассчитывались на два жестких диска по 30 Мбайт, а емкость этих дисководов обозначалась как 30/30, подобно калибру старинного охотничьего ружья «Винчестер».

Нумерация секторов на дорожке жесткого диска не соответствует последовательности их физического размещения. Дело в том, что большая скорость вращения диска не позволяет передать в ПЭВМ содержимое считанного сектора до подвода к головкам физически следующего сектора. Поэтому сектор с номером, на единицу большим предыдущего, размещается за ним через определенное количество секторов *(коэффициент чередования)*. Естественно, в этом случае информация, содержащаяся на дорожке, не может быть прочитана за один оборот МД. Непоследовательная нумерация секторов устанавливается посредством процедуры, называемой **форматированием нижнего уровня**. Для каждого типа НЖМД имеется своя программа такого форматирования. Более подробно понятие чередования обсуждается в п. 8.5.1.

Основным производителем, обеспечивающим более 50 % потребности мирового рынка НЖМД, является американская фирма Maxtor, разместившая свои предприятия в Сингапуре. Именно она и лидирует в улучшении характеристик дисководов этого типа. За ней по пятам следуют компании Micropolis, Imprimus, Conner Peripherals и Newbury Data. Фирма IBM в настоящее время работает над созданием 380-Мбайт 89-мм НЖМД.

Винчестерские накопители устанавливаются, как правило, в системный блок ПЭВМ и невидимы для пользователя, за исключением лампочки HD (Hard Disk), которая загорается при обращении к накопителю. В отличие от гибкого жесткий диск вращается непрерывно. До недавнего времени все НЖМД изготавливались только стандартной высоты. Однако одной из новинок 1989 г. стал НЖМД половинной высоты. По сравнению с НГМД винчестерские накопители имеют повышенную долговечность в связи с тем, что в них отсутствует непосредственный контакт магнитных головок с поверхностью дисков. Стоимость НЖМД большой емкости высока и составляет несколько тысяч долларов.

Контроллер винчестерского накопителя может поддерживать один из следующих интерфейсов:

1) *интерфейс* ST506/412, разработанный фирмой Seagate Technology еще в начале 1980-х гг., в настоящее время преобладающий в ПЭВМ, но не отличающийся высокими характеристиками;

2) *усовершенствованный интерфейс малых дисковых накопителей* (ESDI), предложенный в середине 1980-х гг. и представляющий собой усовершенствованную версию интерфейса ST506/412; обслуживает диски емкостью до 760 Мбайт и имеет повышенное быстродействие;

3) *интерфейс малых вычислительных машин* (SCSI), в отличие от предыдущих являющийся параллельным, а не последовательным интерфейсом, что дополнительно повышает его быстродействие;

4) *интерфейс встроенных дисковых накопителей* (IDE), предложенный пользователям ПЭВМ IBM PC XT/AT в 1988 г. как недорогая альтернатива интерфейсам ESDI и SCSI; основная его особенность — реализация функций контроллера в накопителе.

Ожидается, что Американским национальным институтом стандартов (ANSI) в качестве стандартного будет принят интерфейс ESDI, хотя его набор команд еще не устоялся.

Наряду с попытками повысить емкость НГМД предпринимаются усилия по созданию компактных и вместительных НЖМД, обеспечивающих *смену носителей информации*. На этом пути созданы *накопители со сменными кассетами*, содержащими жесткие МД, а также *съемные накопители*. НЖМД со съемными кассетами требуют дополнительных средств герметизации и очистки поверхности дисков, а емкость кассет относительно невелика, да и остальные характеристики отнюдь не идеальны. Правда, недавно компании Proteus Technology и Sysgen выпустили 150-Мбайт НЖМД со сменными дисками и временем доступа всего 28 мс. А фирма Data Express превзошла все ожидания, разработав 450-Мбайт НЖМД со сменными кассетами. Съемные же накопители, естественно, имеют хорошие технические характеристики, но крайне дороги (ведь кроме пакета дисков приходится платить и за сам привод).

По сравнению с НГМД дисководы со съемными жесткими дисками стоят дороже, но имеют повышенную емкость, быстродействие и долговечность.

Перспективным материалом для замены металлической основы жестких дисков является стекло. Одна из основных проблем при этом — прочность стеклянных дисков. Но зато гладкая поверхность стеклянных дисков позволяет обеспечивать «плавание» магнитных головок с уменьшенным зазором, благодаря чему повышается плотность записи, а следовательно, и емкость диска. Несмотря на ряд технологических трудностей, в 1988 г. американская фирма Areal Technology выпустила 89-мм НЖМД с одним стеклянным диском емкостью до 100 Мбайт и средним временем доступа 28 мс. Он имеет также пониженное по сравнению с аналогами энергопотребление.

Недавно фирма IBM сообщила о создании новых НМД с потрясающей плотностью записи, составляющей 1 млрд. бит на квадратный дюйм. Для этого применены тонкопленочные головки, а зазор уменьшен в 3 раза. Однако коммерческие изделия, использующие эту технологию, появятся только через несколько лет. В ходе эксперимента была достигнута скорость чтения/записи данных 3,5 Мбайт/с.

2.4.3. Накопители на оптических дисках

Долгие теоретические изыскания наконец-то привели к созданию коммерческих образцов НОД, которые в настоящее время начинают внедряться в ПЭВМ, обладая чрезвычайно большой информационной емкостью. До использования их в качестве основных ВЗУ для загрузки выполняемых программ еще далеко, однако уже наметился путь к этому. Так, ультрасовременная ПЭВМ NeXT (IBM-несовместимая) одноименной компании в стандартной комплектации имеет НОД на сменных дисках вместо НМД.

Техника НОД берет начало в области методов бытовой звуковой оптической записи. Разработчиков НОД преследовали неудачи с самого начала исследований — с середины 70-х гг. Решив ряд серьезных проблем, им все-таки удалось в 1983 г. предложить работоспособные образцы НОД, но последние не получили коммерческого внедрения из-за отсутствия оптических дисков (ОД). В тот же период фирмой Matsushita был изготовлен опытный образец НОД с возможностью перезаписи. Только лишь в конце 80-х гг. «призрак» НОД стал реальностью.

Принцип работы всех существующих ныне оптических дисководов основан на использовании луча лазера для записи и чтения информации в цифровом виде. В процессе записи модулированный цифровым сигналом лазерный луч оставляет на активном слое оптического носителя след, который затем можно прочитать, направив на него луч меньшей интенсивности и проанализировав изменение характеристик отраженного луча.

По функциональному признаку НОД делятся на три категории:

1) НОД *без возможности записи* (только для чтения);

2) НОД *с однократной записью и многократным чтением;*

3) НОД *с возможностью перезаписи* (перезаписывающие оптические дисководы).

Первая категория НОД использует технологию CD-ROM (ПЗУ на компакт-диске), которая возникла как продолжение технологии CD для цифровой записи звука. Компакт-диски, подобно грампластинкам, записываются на заводе-изготовителе и используются для распространения больших объемов информации, предназначенной только для считывания. Пользователь не имеет возможности ни стереть, ни записать информацию на таком диске.

Затем появились оптические дисководы второй категории, основанные на технологии WORM («однократная запись, многократное чтение»). На ОД в таких дисководах можно самостоятельно записывать, однако однажды записанную информацию ни стереть, ни перезаписать не удастся. Поэтому НОД данной категории удобны только для архивирования и особенно полезны в областях, где принципиально важно хранить единожды записанную информацию в неизменном виде. Кроме того, на ОД с однократной записью можно помещать ряд последовательных версий файлов с данными, причем с возможностью последующей выборки нужного варианта. Однако оптическая память при этом становится обычным расходным невосстанавливаемым материалом.

Поистине революцией явилась технология перезаписываемых ОД. НОД с возможностью перезаписи — это функциональные эквиваленты, а следовательно, и потенциальные конкуренты НМД.

В качестве носителя информации в НОД применяются жесткие диски, покрытые специальным оптическим материалом. Сами диски обычно изготавливаются из поликарбоната, хотя некоторые разработчики предпочитают использовать стекло. Существенным технологическим достижением является создание английской фирмой Imagedata *цифровой бумаги*, представляющей собой новый тип WORM-носителя. Она обладает достаточной гибкостью, чтобы ее можно было намотать на катушку как фотопленку. Путем резки или штамповки из такой бумаги удается получить носители различного вида — такие, как гибкие диски, ленты или карточки.

По сравнению с дисковыми оптические накопители имеют следующие характеристики:

1) более высокое среднее время доступа, чем НЖМД, но более низкое в сравнении с НГМД (35 — 100 мс для лучших образцов);

2) высокую стоимость, доходящую до 10 тыс. долл.;

3) более высокую плотность записи, а следовательно, емкость (благодаря возможности точной фокусировки лазерного луча);

4) повышенные надежность и долговечность из-за отсутствия контакта носителя с головками (по сравнению с НГМД) и большим зазором между диском и головками (в сравнении с НЖМД);

5) возможность смены ОД без специальных ухищрений.

Быстродействие НОД принципиально ограничивается необходимостью предварительного стирания информации перед перезаписью, что сложно реализовать за один проход. Один из путей преодоления этого недостатка состоит в использовании магнитооптического метода записи.

Соотношение характеристик НОД и НМД определяет использование первых в ближайшем будущем только *в качестве дублирующих накопителей* (для долговременного хранения информации без возможности загрузки программ в ОЗУ для выполнения) или *вместо НГМД* (в случае снижения стоимости оптических дисководов), но не взамен НЖМД.

Для примера охарактеризуем перезаписывающие НОД Tahiti I и Fiji I уже известной нам фирмы Maxtor. 133-мм оптический дисковод Tahiti I обладает средним временем доступа 43,5 мс и обеспечивает емкость 1 Гбайт. 89-мм оптический накопитель Fiji I имеет емкость 160 Мбайт и среднее время доступа 100 мс. Диски в обеих моделях съемные.

В 1991 г. IBM предложила для своих новых моделей ПЭВМ семейства PS/2 89-мм 128-Мбайт НОД с возможностью перезаписи.

2.4.4. Накопители на магнитных лентах

НМЛ стали применяться в ЭВМ в 50-х гг., когда магнитная лента (МЛ) была уже широко распространена в звукозаписи и измерительной технике. Уже не раз предсказывалось ослабление роли МЛ как машинного носителя информации, однако она в ближайшем будущем сохранит свои позиции, по крайней мере, на больших ЭВМ.

В ПЭВМ НМЛ не получили широкого распространения. Они предлагаются в качестве факультативных ПУ и используются только в роли *дублирующих накопителей* для архивирования (резервирования) информации. Синонимом (на жаргоне) НМЛ для ПЭВМ является термин *стример* (strimer).

НМЛ состоит из лентопротяжного механизма, блока магнитных головок (для чтения, записи и стирания), а также электронного блока. МЛ представляет собой гибкую подложку в виде ленты, на которую нанесено магнитное покрытие. Покрытие обычно состоит из мельчайших частиц окисла железа, взвешенных в инертном связующем веществе, а подложка традиционно изготавливается из полиэфира. Информация записывается с помощью магнитного кодирования (как и в НМД) на несколько дорожек одновременно при движении МЛ вдоль магнитных головок. Считывание осуществляется аналогично. Таким образом, к ленте может быть организован только последовательный доступ, что принципиально ограничивает быстродействие НМЛ. МЛ поставляется намотанной на бобины (катушки) или в кассетах. В соответствии с этим различаются и НМЛ.

В ППЭВМ используются только *кассетные* НМЛ с шириной ленты 6,35 мм (0,25 дюйма). Кассета с такой лентой внешне очень похожа на видеокассету. Емкость кассеты стандартного размера может достигать 1 Гбайт, особенно если применяется сжатие данных. Преуспели в этом фирмы Gigatek Memory Systems и Tecmar. Имеются и миникассеты с емкостью 40 — 80 Мбайт, которую планируется довести до 110 Мбайт. Скорость передачи данных для лучших изделий может достигать 10 Мбайт/мин. Среднее же время доступа очень велико — несколько минут. Стоимость 150-Мбайт кассетного НМЛ составляет примерно 1 тыс. долл. Габаритные размеры кассетного НМЛ совпадают с габаритами 133-мм НГМД стандартной высоты.

В бытовых ПЭВМ в качестве НМЛ используются обычные кассетные магнитофоны либо специальные устройства для работы с бытовыми компакт-кассетами (ширина ленты равняется 3,8 мм, т.е. 0,15 дюйма). Выпускаются и специальные компакт-кассеты улучшенного качества для цифровой записи.

Несмотря на имеющиеся недостатки, по сравнению с НМД ленточные накопители имеют и неоспоримое достоинство — более высокую надежность хранения информации. Серьезными же конкурентами НМЛ в этом плане в настоящее время выступают НОД.

2.4.5. Запоминающие устройства на новых физических принципах

Для устранения недостатков существующих устройств памяти многими фирмами и научно-исследовательскими организациями проводятся работы по созданию накопителей информации на основе новых физических принципов. Наиболее проработанными в настоящее время являются следующие виды запоминающих устройств:

— *память на цилиндрических магнитных доменах* (ЦМД);

— *голографическая память*.

В памяти *первого* вида биты представляются цилиндрическими магнитными доменами («элементарными» кольцевыми магнитиками), способными перемещаться в стационарной плоской среде. Можно провести аналогию между памятью на ЦМД и НМД. Разница состоит в том, что в НМД движется носитель информации, а не сама информация; в памяти же на ЦМД все организовано наоборот, что обеспечивает ее более высокую надежность благодаря отсутствию механических подвижных частей. Емкость одного устройства памяти на ЦМД обычно составляет 1 Мбит (128 Кбайт), а время доступа — 30 мс. Для увеличения общей емкости можно объединить несколько устройств. Являясь *энергонезависимой*, по быстродействию и стоимости память на ЦМД занимает промежуточное положение между памятью на полупроводниках и магнитных дисках. Она уже находит применение в портативных ПЭВМ и в перспективе, видимо, способна вытеснить НМД.

Использовать *голографию* для запоминания информации предложили А.Л.Микаэлян и В.И.Бебринев еще в 1966 г.

Первоначально голография применялась главным образом только для фиксации изображений в объемном виде. *Голограмма* формируется на поверхности кристалла в результате интерференции двух лазерных лучей — сигнальной волны как результата отражения луча от голографируемого объекта и опорной волны (того же луча, направленного на кристалл под определенным углом). Изменяя угол падения опорного луча, можно записать на один и тот же кристалл множество изображений. Считывание изображения осуществляется таким же лазерным лучом.

Только лишь в 1991 г. идея применения голографии для хранения информации стала реальностью. Заслуга в этом принадлежит фирме Microelectronics and Computer Technology. Сотрудникам данной фирмы удалось решить две задачи:

— *повысить надежность* хранения информации в голограмме (раньше она разрушалась после нескольких считываний);

— *снизить стоимость* голограммы в результате применения множества мелких кристаллов вместо одного большого.

В ближайшее время ожидается создание опытного образца накопителя емкостью 1 Гбайт, которая в дальнейшем может быть увеличена в 1000 раз.

2.4.6. Иерархия памяти ПЭВМ

Настало время подвести итоги рассмотрения различных устройств памяти, используемых в ПЭВМ. По функциональному признаку память персональных машин делится на следующие иерархические уровни:

1) *сверхоперативную память* (СОП) для хранения указателей адресов и промежуточных результатов вычислений;

2) *кэш-память*, выполняющую роль буфера между МП и ОП, а также другими устройствами в случае дисбаланса в их быстродействии;

3) *ОП*, хранящую выполняемые программы, данные для них и результаты вычислений (как промежуточные, так и окончательные перед выдачей пользователю);

4) *внешнюю оперативно-доступную память* (ВОДП), служащую для хранения программ, которые могут быть непосредственно загружены в ОЗУ для выполнения, а также массивов данных, доступных выполняемым программам и формируемых этими программами;

5) *внешнюю дублирующую память* (ВДП), предназначенную для создания резервных копий (архивов) программ и данных, а также для связи с внешним миром и обеспечения максимально возможной конфиденциальности.

Уровни 4 и 5 объединяются общим названием «внешняя память», или ВЗУ. ОП, как мы уже говорили, делится на ОЗУ и ПЗУ. Каждый следующий уровень иерархии памяти характеризуется по сравнению с предыдущим, как правило, меньшим быстродействием и удельной стоимостью, но большей емкостью и надежностью хранения информации. Распределение различных типов устройств памяти по приведенным уровням иерархии представлено на рис. 2.7.

Рис. 2.7. Уровни иерархии и типы устройств памяти

Сравнительные характеристики различных устройств памяти иллюстрируются табл. 2.4. Цены в ней для НГМД и НОД проставлены исходя из стоимости носителей информации, а не самих накопителей.

Таблица 2.4

Характеристики различных устройств памяти

Устройство	Среднее время доступа	Скорость чтения/записи, Мбайт/с	Объем памяти, байт	Стоимость хранения 1 Мбайт информации, долл.
регистр МП	5 нс	нет данных	2-4	2400
SRAM	45-55 нс	нет данных	128К	500
DRAM	70-150 нс	5000-50000	32К-512К	150
НГМД	200-1000 мс	0,030-0,040	360К-1,44М	3
ЖМД	4-100 мс	0,5-1,0	5М-2Г	100
НОД	35-175 мс	0,1-0,8	100М-5,6Г	3,5-5,5
НМЛ	15 с	0,06-0,08	40М-1Г	0,75

2.5. Устройства ввода информации

Пользователям ПЭВМ предлагается широкий спектр различных устройств ввода информации. И все же зачастую ПЭВМ поставляется с единственным устройством ввода — *клавиатурой*, являющейся основным ПУ. За клавиатурой по популярности следуют различные манипуляторы, функционально ее дополняющие, и особенно *манипуляторы* типа «мышь», а также шаровые манипуляторы. Остальные устройства ввода применяются значительно реже, так как полезны только лишь для некоторых приложений.

В данном подразделе рассматриваются различные устройства ввода информации в ПЭВМ.

2.5.1. Клавиатуры

Единственным основным устройством ввода информации в ПЭВМ и управления ее работой была, есть и в обозримом будущем останется *клавиатура*. Она представляет собой *матрицу клавиш*, объединенных в единое целое, и *электронный блок* для преобразования нажатия клавиши в двоичный код.

Клавиатура должна быть эргономичной, т.е. удобной для работы. К основным эргономическим показателям клавиатуры относят:

— общекомпоновочные решения клавиатуры в ПЭВМ;
— толщину клавиатуры и угол ее наклона относительно горизонтали;
— схему расположения клавиш, их цвет, форму и размеры;
— необходимое усилие для нажатия клавиши и ее свободный ход;
— коэффициент отражения света клавишами и всей поверхностью клавиатуры;
— легкость чтения надписей на клавишах.

Мы не будем подробно характеризовать эти показатели, но все же отметим следующее. Во-первых, существуют два основных общекомпоновочных принципа, касающихся исполнения клавиатуры. При *моноблочном* построении ПЭВМ (главным образом, бытовых) клавиатура является неотъемлемой составной частью системного блока. При *полиблочном* построении ПЭВМ она выполнена в виде отдельного устройства, соединенного с системным блоком посредством гибкого кабеля. Последнее, конечно, представляется более удобным. Во-вторых, схема расположения клавиш, особенно алфавитно-цифровых, стандартизована, чтобы каждый человек мог одинаково хорошо общаться с любой ПЭВМ. И в-третьих, особое значение имеет необходимое усилие для нажатия клавиш. Слишком «тугие» клавиши замедляют работу и ведут к быстрому утомлению пользователя. Слишком же «слабые» клавиши (что, к сожалению, характерно для отечественных ПЭВМ) приводят к большому количеству ошибок, обусловленных случайным их касанием, а следовательно, в конечном счете, также замедляют работу.

Быстродействие клавиатуры всегда таково, что независимо от скорости нажатия клавиш вся вводимая информация успевает передаваться в ПЭВМ, если она там ожидается.

В клавиатурах ПЭВМ используются клавиши различных типов, из которых наиболее широкое распространение получили емкостные и контактные.

Емкостные клавиши состоят из подвижной металлической пластины (подвижного электрода), прикрепленной к кнопке, и двух металлических выступов на печатной плате, образующих неподвижные электроды конденсатора переменной емкости. При нажатии на клавишу подвижная пластина приближается к этим выступам, что приводит к изменению емкости конденсатора, а этого достаточно для фиксации нажатия электронной схемой. Помимо простоты устройства емкостные клавиши имеют высокую надежность. Они выдерживают до 100 и более миллионов нажатий и отпусканий.

Контактные клавиши могут изготавливаться в различных вариантах, но всегда в основе их работы лежит принцип непосредственного механического контакта между двумя гибкими металлическими пластинами при нажатии клавиши. В местах соприкосновения пластины обычно имеют специальное покрытие, обеспечивающее малое сопротивление контакта. Срок службы контактных кнопок характеризуется числом срабатываний в несколько десятков миллионов.

Характерной особенностью клавиатур ПЭВМ является допустимость перекрытия нажатий, т.е. случайного или преднамеренного нажатия одновременно двух и более клавиш. В этом случае клавиатура обеспечивает правильную выдачу кодов всех клавиш в порядке их нажатия. Кроме того, есть множество комбинаций клавиш, одновременное нажатие которых приводит к выдаче в ПЭВМ определенных кодов, называемых **расширенными кодами клавиш**.

Другая особенность клавиатур ПЭВМ заключается в том, что при нажатии клавиши в ПЭВМ передается один код, а при отпускании — иной, которые называются **кодами сканирования (опроса) клавиш**. В ПЭВМ эта кодировка преобразуется в другую при помощи соответствующей таблицы. Такая организация ввода символов придает дополнительную гибкость и мобильность при использовании различных фонетических систем, а также позволяет перепрограммировать клавиши. Дополнительно к этому в клавиатурах реализуется так называемая *функция повторения*, обеспечивающая многократную выдачу кода нажатия клавиши при ее удержании в утопленном состоянии. Периодическое повторение выдачи кода нажатия прекращается в случае нажатия другой клавиши или отпускания нажатой.

Таким образом, клавиатура в ПЭВМ является весьма совершенным устройством ввода.

Любая клавиатура ПЭВМ имеет четыре группы клавиш:

1) *клавиши пишущей машинки* для ввода прописных и строчных букв, цифр и специальных знаков;

2) *служебные клавиши*, меняющие смысл нажатия остальных клавиш и осуществляющие другие действия по управлению вводом с клавиатуры;

3) *функциональные (программируемые) клавиши*, смысл нажатия которых зависит от используемого программного продукта;

4) *клавиши двухрежимной малой цифровой клавиатуры*, обеспечивающие быстрый и удобный ввод цифровой информации, а также управление курсором и переключение режимов работы клавиатуры.

Стандартами де-факто клавиатур для ПЭВМ являются предложенные фирмой IBM 83- (84-), 101- и 122-клавишные пульты. Первыми двумя из них обычно комплектуются машины IBM PC XT и AT соответственно, но первоначально IBM PC AT поставлялась с 84-клавишной клавиатурой. Аналогичные клавиатуры выпускаются рядом других фирм. Портативные ПЭВМ, как правило, имеют клавиатуры с меньшим количеством клавиш. Отечественные ПЭВМ поставляются с клавиатурами, очень напоминающими 83-клавишный пульт. Подробное описание ряда клавиатур будет приведено в п. 5.4.1.

2.5.2. Манипуляторы

Манипуляторы (координатно-указательные устройства, устройства управления курсором) являются дополнительными ПУ для ввода информации. Совместно с клавиатурой они повышают удобство работы пользователя с рядом диалоговых программных продуктов, где требуется быстро перемещать курсор по экрану дисплея и выбирать пункты меню, а также выделять фрагменты экрана. Итак, главная функция манипуляторов состоит в облегчении перемещения курсора (который может принимать различную форму) по экрану и отметки при необходимости точки экрана, которая указывается курсором. Ряд манипуляторов обеспечивает также возможность вычерчивания на экране изображений. Основной особенностью работы с ними является обязательное наличие обратной связи с пользователем путем отображения действий, производимых посредством манипуляторов, на экране дисплея.

В настоящее время в ПЭВМ используются следующие разновидности манипуляторов:

1) джойстик;

2) световое перо;

3) манипулятор типа «мышь»;

4) шаровой манипулятор (манипулятор типа «шар»);

5) манипулятор Isopoint Control.

Джойстик (от Joystic — рукоятка, рычажный указатель) обеспечивает перемещение курсора на экране в одном из четырех направлений. Он представляет собой (см. рис. 2.8, а) рычаг, установленный на соответствующем корпусе. Корпус с помощью присосок фиксируется на неподвижной поверхности вблизи ПЭВМ. Сам рычаг, шарнирно соединенный с преобразователями углов, может совершать движения (в результате воздействий руки пользователя) вдоль координат x и y в пределах некоторого телесного угла. На рычаге может находиться одна или несколько кнопок. Джойстик используется в бытовых ПЭВМ для взаимодействия с игровыми программами. При этом курсор может принимать форму какого-либо движущегося объекта.

Световое перо (light pen) может применяться для указания точки на экране дисплея (участка экрана) или для формирования изображений. Оно конструктивно напоминает ручку (см. рис. 2.8, б), внутри которой находится фотоэлемент. Когда световое перо приставлено к экрану, световой поток, образуемый светящейся его точкой, через отверстие в ручке поступает к фотоэлементу. Нажатие имеющейся на пере кнопки приводит к передаче соответствующего сигнала в ПЭВМ по шнуру для указания точки на экране. Совместно со световым пером используется специальная программа, которая получает сигнал, идущий от него, и сигнал синхронизации дисплея, вычисляет временную задержку и исходя из этого определяет затем координату указанной световым пером точки. При использовании светового пера возникают трудности указания из-за параллакса, обусловленного толщиной экрана, и из-за слишком большой площади наконечника пера. Они устраняются благодаря следящему перекрестию, формируемому из тонких линий и размещаемому на экране так, что его центр указывает точку наибольшей чувствительности приставленного к экрану светового пера. При перемещении пера по экрану следящее перекрестие движется вместе с ним, указывая текущее положение пера. Работа со световым пером приводит к быстрому утомлению пользователя.

Манипулятор типа «мышь» (mouse) представляет собой приспособление для указания нужных точек на экране путем перемещения его вручную по плоской поверхности. Координаты местоположения «мыши» передаются в ПЭВМ и вызывают соответствующее перемещение курсора в виде точки или стрелки (обычно говорят — указателя «мыши») по экрану дисплея. Иногда допускается и вычерчивание изображений. Это устройство изобретено Д.Энджеблатом в 1964 г., впервые выпущено в начале 70-х гг. корпорацией Xerox и приобрело в последующем большую популярность. Конструктивно (см. рис. 2.8, в) «мышь» представляет собой небольшую пластмассовую коробку, которая подсоединяется к ПЭВМ обычно посредством шнура.

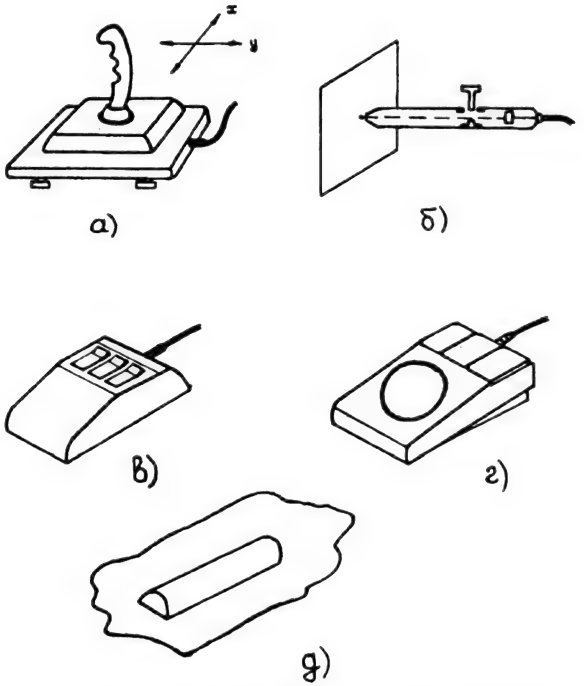


Рис. 2.8. Манипуляторы для ввода информации:
а – джойстик, б — световое перо, в — «мышь»
г — «шар», д — Isopoint Control

В соответствии с принципом действия различают механические и оптические «мыши».

Основным узлом *механической «мыши»* является шар, выступающий из основания корпуса и соприкасающийся с поверхностью стола. При перемещении этого прибора по столу вращение шара преобразуется электронным блоком в соответствующие электрические сигналы, передаваемые в ПЭВМ и приводящие к перемещению курсора на экране. На верхней крышке манипулятора расположены одна или несколько кнопок для указания позиции на экране и редактирования выбранной информации.

Оптическая «мышь» работает иначе, хотя и похожа на механическую. Она перемещается по специальному планшету, на поверхность которого нанесена сетка из разноцветных линий. Специальный оптоэлектронный узел фиксирует направление движения манипулятора и расстояние, им пройденное, посылая на планшет луч от светодиода и принимая отраженный сигнал. Существенную роль при этом играет имеющаяся на планшете сетка.

Оптическая «мышь» является более сложным и дорогим устройством, требует наличия специального планшета, но более долговечна и надежна в работе (из-за отсутствия сложных механических узлов), а также имеет существенно меньшую массу (так как шар становится ненужным).

Обычно «мыши» обладают постоянной чувствительностью, не зависящей от скорости перемещения манипулятора по рабочей поверхности. Это не всегда удобно: лучше иметь «мышь», чувствительность которой изменяется обратно пропорционально скорости перемещения манипулятора. Реализация данной зависимости позволит быстро перемещать курсор по экрану на большие расстояния, но точно его позиционировать в нужной точке экрана. Примером «мыши» с переменной чувствительностью является устройство LogiMousePilot фирмы Logitech.

Стандартные «мыши» требуют наличия на рабочем столе специальной рабочей поверхности. Лучше всего использовать гладкий (но не скользкий) планшет. Однако фирмой Honeywell выпускается манипулятор, который с состоянии великолепно работать не только на поверхности с любым качеством, но и под любым углом к горизонту (даже вверх ногами).

Еще одно немалое неудобство в работе с «мышью» доставляет кабель, соединяющий ее с ПЭВМ. Этот недостаток устранен в предлагаемых на рынке *беспроводных манипуляторах типа «мышь»*. Информация о перемещении манипулятора в таких случаях передается инфракрасными лучами или радиосигналами посредством передатчика, встроенного в манипулятор. Эти сигналы фиксируются специальным приемником и поступают в ПЭВМ. При использовании инфракрасного диапазона «мышь» должна находиться в зоне прямой видимости приемника. Радиус действия беспроводных манипуляторов достигает нескольких метров, питаются они от батареек или аккумуляторов.

Еще один ощутимый недостаток «мыши» состоит в необходимости иметь на рабочем столе дополнительное место. От этого недостатка свободен *манипулятор типа «шар»* (trackball), первоначально использовавшийся только в больших ЭВМ по причине его дороговизны. В 1987 г. американская компания Itac Systems выпустила манипулятор типа «шар» и для ПЭВМ. Теперь такие манипуляторы получили широкое распространение и способны вытеснить манипуляторы типа «мышь».

Манипулятор типа «шар» (см. рис. 2.8, г) представляет собой просто перевернутую, или стационарную, механическую «мышь» с теми же функциональными возможностями. При использовании шарового манипулятора достаточно вращать шар, а не перемещать весь прибор. Новейшее шаровое устройство американской фирмы Marconi Electronic Devices стоит всего 198 долл. Если «шар» разместить на клавиатуре, то не будет необходимости переключать внимание и переносить руку с манипулятора на клавиатуру и обратно. Следовательно, встроенный в клавиатуру «шар» устраняет сразу два недостатка «мыши».

Вообще же «шар» позволяет позиционировать курсор точнее «мыши», так как сам шар может иметь бо́льшие размеры, да и вращать его удобнее.

Однако шаровой манипулятор все еще слишком громоздок, что затрудняет его использование в портативных ПЭВМ. В 1989 г. К.Калвером именно для таких применений был создан новый манипулятор под названием *Isopoint Control*. Этот термин пока не имеет аналога в русском языке, а его дословный перевод (что-то вроде «равноточечного управления») тоже ничего не дает. Поэтому мы оставили наименование прибора в оригинале. Это устройство очень компактно и может размещаться рядом с клавишей пробела на клавиатуре или выполняться автономно. Оно представляет собой (см. рис. 2.8, д) цилиндр, вращая который можно перемещать курсор по осям x и y. Нажатие на цилиндр соответствует нажатию кнопки «мыши» или шарового манипулятора. Несмотря на то, что Isopoint Control не является полным функциональным аналогом этих устройств (не позволяет перемещать курсор в произвольном направлении), он устраивает многих пользователей, подходит для ряда приложений и поэтому, видимо, вскоре получит широкое распространение.

Еще одной новинкой в области манипуляторов явилась «мышь» в форме авторучки, выпущенная фирмой International Machine Control System (США) для машин семейств PC и PS/2 и предлагаемая за 129 долл. Это устройство отличают малые габариты, удобство в работе и низкая чувствительность к качеству рабочей поверхности.

2.5.3. Сканеры

Сканером (от англ. scanner) называется устройство ввода, позволяющее вводить в ЭВМ изображения. Ввод изображений может потребоваться при размножении документов, для их редактирования с последующей выдачей, а также в системах хранения и поиска изображений. Сканеры являются дополнительными ПУ ПЭВМ. При комплектации сканером и высококачественным печатающим устройством ПЭВМ превращается в АРМ для подготовки и издания различных информационных материалов.

Использование сканеров для ввода в ПЭВМ изображений (текстовой и графической информации) уже имеет как минимум пятилетнюю историю. Сейчас на рынке представлено не менее 150 различных устройств — от ручных портативных сканеров по цене 200 долл. до сложных систем стоимостью свыше 16000 долл. Ведущую роль в производстве сканеров играют японские фирмы. С ними не без успеха соперничают американские компании. Жесткая конкуренция обеспечивает неуклонное улучшение характеристик и качества наряду со снижением стоимости сканирующих устройств.

Сканеры характеризуются:

— разрешающей способностью (разрешением);
— количеством воспринимаемых оттенков;
— возможностью ввода цветных изображений или отсутствием таковой;
— быстродействием;
— размером обрабатываемых изображений;
— стоимостью.

Большинство из этих показателей рассмотрим на фоне изложения принципов работы сканирующих устройств.

Аналогично копировальному устройству сканер освещает оригинал, а светочувствительный датчик сканера с определенной частотой производит замеры интенсивности отраженного оригиналом света. Разрешающая способность сканера прямо пропорциональна частоте замеров. В процессе сканирования устройство выполняет преобразование величины интенсивности в двоичный код, который передается в ЭВМ для дальнейшей обработки. Если сканер при каждом замере регистрирует всего один бит информации, то он распознает только два цвета — черный и белый. В зависимости от количества битов, соответствующих одному замеру, сканер может распознавать большее или меньшее количество оттенков от черного до белого. Так, при 4-битовом кодировании имеется возможность распознавания 16 различных оттенков. Восьмибитовые же сканеры обеспечивают регистрацию 256 градаций серого цвета. В зависимости от спектра сканирующего луча сканер может не реагировать на изображения, выполненные тем или иным цветом.

Большинство выпускаемых сканеров являются черно-белыми, т.е. даже при считывании цветных изображений происходит их «превращение» в черно-белые. Однако в 1989 г. на рынке появились и цветные сканеры. Возможность цветного сканирования не исключалась и раньше, но соответствующее оборудование стоило слишком дорого (обычно десятки тысяч долларов). И только недавно выпущенные устройства JX-450 фирмы Sharp, Scanmaster фирмы Howtek, Epson ES-300C компании Seiko Epson, а также некоторые другие оказались доступными по цене.

Программные средства, управляющие работой сканера, с учетом возможностей самого сканера могут обеспечивать один из трех режимов сканирования:

1) сканирование штрихового рисунка;
2) сканирование изображения с полутоновой интерпретацией;
3) сканирование шкалы яркости (серой шкалы).

Для обеспечения первых двух режимов достаточно иметь сканер с однобитовым кодированием, причем второй режим реализуется исключительно программными средствами. Третий же режим требует устройства с многобитовым кодированием. *Штриховой рисунок* представляет собой изображение, содержащее только черные и белые участки, без каких-либо промежуточных оттенков. Сканирование такого изображения не требует дорогостоящих сканеров, сложного программного обеспечения и большого объема памяти для хранения изображения.

Два последних режима применяются для сканирования полнокровных изображений, в том числе, конечно, и штриховых рисунков.

Если поближе рассмотреть иллюстрацию в газете, то можно увидеть, что она не содержит непрерывных полутоновых переходов, а представляет собой множество точек. Такое изображение называется *полутоновым*. При просмотре с большего расстояния точки полутонового изображения сливаются вместе и создают имитацию оттенков. Для воспроизведения различных оттенков применяется следующая техника. Расстояние между центрами точек по вертикали и горизонтали остается постоянным и измеряется количеством линий на дюйм (точек на дюйм), где 1 дюйм = 2,54 см. Размеры же точек изменяются, причем более крупные точки создают впечатление темного цвета, а точки с меньшими размерами делают изображение более светлым. Описанное «растровое» представление для газетных фотографий формируется обычно 65 точечными линиями на дюйм. Для журналов с хорошим качеством иллюстративного материала этот показатель равен 133 или 155.

Работа большинства систем сканирования основана именно на принципе *полутоновой интерпретации*. Такое сканирование по-прежнему является однобитовым, но введенное изображение подвергается затем программно-реализованной процедуре фильтрования с целью получения *«смазанного»* изображения. Термин *«смазанное»* в данном случае связан с методом имитации промежуточных оттенков серого цвета посредством изменения размеров точек. Если введенное и обработанное таким образом изображение распечатать, то получится полутоновый рисунок, как в газете или журнале. Для хранения отфильтрованного изображения требуется больший объем памяти, а для его формирования необходимо соответствующее ПО. Кроме того, нет гарантии отсутствия ошибок в интерпретации изображения.

Для получения лучшего качества копии введенного изображения следует выбрать сканер и ПО, обеспечивающие работу в режиме *воспроизведения шкалы яркости*, или *серой шкалы*. В этом случае используется непосредственно многобитовое сканирование без какой-либо последующей обработки изображения. Воспроизведение 256 оттенков серого цвета оказывается максимально достаточным, так как человеческий глаз не в состоянии различить более «тонкую» градацию. В случае обеспечения такого уровня переходы между участками изображения с различной яркостью становятся плавными и выглядят вполне естественно. Однако при использовании 8-битового кодирования процесс сканирования фотографии размером 8x10 см может потребовать 5 Мбайт дисковой памяти. Процесс сканирования с 16, 32 или 64 уровнями требует меньших ресурсов, но качество изображения при этом снижается.

В соответствии с конструктивным исполнением сканеры делятся на настольные и портативные (ручные).

Настольные сканеры представляют собой высококачественные устройства, большинство из которых стоит не менее 1500 долл. Настольные сканеры бывают планшетного типа, либо похожими на фотоувеличитель, либо с роликовыми направляющими, а также с другими средствами подачи бумаги. Сканеры с подачей бумаги напоминают печатающие устройства.

В качестве примеров настольных сканеров охарактеризуем три изделия, выпускаемые фирмами Kurzweil Computer Products и Datacopy, являющиеся дочерними компаниями фирмы Xerox Imaging Systems.

Модель Datacopy 730GS стоимостью 1300 долл. представляет собой сканер планшетного типа, предназначенный для ввода как графической, так и текстовой информации. Помимо обеспечения интерпретации 16 или 64 уровней серой шкалы устройство имеет 27 различных значений разрешения в диапазоне 60 — 450 точек на дюйм. Однако при максимальных значениях двух этих параметров потребуется более 23 мин. для ввода информации и порядка 20 Мбайт на диске для ее записи. Для ввода фотографии с 16 уровнями серой шкалы при разрешении 60 точек на дюйм требуется 1 мин. Программой ввода изображений PC Image обеспечивается возможность записи файлов в четырех форматах. Дополнительно к этому она предоставляет широкий спектр возможностей для редактирования изображений, хотя больших успехов можно добиться с использованием независимых графических редакторов PC Paintbrush IV Plus и Publisher's Paintbrush. Сканером распознается большое количество стандартных шрифтов и, кроме этого, имеются возможности его «обучения».

Устройство Kurzweil Discover 73200 модели 5 стоимостью 4995 долл. представляет собой сканер, который ориентирован на ввод только текстовой информации, при этом распознаются различные шрифты и обеспечиваются три уровня управления контрастностью. Интерпретация букв осуществляется методами искусственного интеллекта, реализованными в программных средствах. Обеспечивается запись информации в файлы 16-ти текстовых форматов. Сканер поставляется с сопроцессорной платой, содержащей 2-Мбайт ОЗУ.

Если уместны автомобильные аналоги, то модель Kurzweil K-5000 можно назвать Кадиллаком как по стоимости (15950 долл.), так и по быстродействию и качеству. В его состав входит плата сопроцессора, содержащая МП MC68020 и 4-Мбайт ОЗУ. Для ввода фотографии при разрешающей способности 400 точек на дюйм в режиме серой шкалы требуется менее 5 мин. Имеется возможность ввода текстовой информации. В программных средствах распознавания символов при этом использованы методы искусственного интеллекта. В графическом режиме данная модель обеспечивает наилучшие по качеству параметры, но для реализации ее возможностей требуется печатающее устройство с аналогичными характеристиками.

Портативные сканеры стоят, как правило, дешево. Но по сравнению с настольными они обладают весьма скромными возможностями. Кроме того, малейшая вибрация в процессе ручного сканирования приводит к искажению изображения. Портативный сканер похож на большую «мышь» с длинным «хвостом», который подключается к ПЭВМ. Комплект поставки, аналогично настольным сканерам, включает необходимое ПО. Работа с таким аппаратом состоит в том, что оригинал помещается на плоскую поверхность, сканер устанавливается на одной из его сторон и, после нажатия кнопки пуска, медленно перемещается по оригиналу вручную. Неоспоримым достоинством ручного сканера является возможность обработки не только плоских изображений.

О вводе текстов в ПЭВМ необходимо сказать особо. Мы определили сканер как устройство ввода графической информации. Введенный рисунок записывается на внешний носитель информации в специальном формате и может быть отредактирован посредством графического редактора и/или отпечатан принтером в медленном графическом режиме. Текст же, хотя по внешнему виду и является частным случаем изображения, в ПЭВМ обрабатывается иначе. Эти особенности перечислены ниже:

— тексты хранятся в текстовых форматах, которые требуют существенно меньше внешней памяти (байт на символ, а не байт на точку при 256-битовом сканировании);

— тексты обрабатываются текстовыми, а не графическими редакторами;

— текст может быть напечатан принтером в гораздо более быстром текстовом режиме;

— с текстом можно производить манипуляции, недоступные для изображений, например, понимание (уяснение) и умозаключения.

Обеспечить ввод напечатанного или рукописного текста можно одним из следующих способов:

1) использовать специальное устройство оптического распознавания символов;

2) применить сканер с программными средствами для распознавания символов.

Устройства оптического распознавания символов появились около пяти лет назад. Однако они весьма дороги и обладают ограниченными возможностями. Проблемы возникают даже при вводе текстов с пропорциональными шрифтами (с переменной шириной символа), не говоря уже о рукописных текстах.

При втором способе сканер, как обычно, вводит изображение. Затем оно читается специальными программными средствами и преобразуется в текстовый формат. Здесь не обойтись без методов искусственного интеллекта, в частности, теории распознавания образов. Такое ПО достаточно сложно, но в этой области достигнуты заметные успехи. В качестве образца для подражания можно назвать систему The Typist американской фирмы Caer, включающую ручной сканер, программный распознаватель символов и программу обеспечения целостности образа.

Одним из основных показателей качества системы ввода текстов является точность идентификации вводимых символов, или вероятность ошибок при вводе.

Как мы уже видели, многие устройства ввода сочетают в себе возможности сканеров и устройств оптического распознавания символов.

2.5.4. Графические планшеты

В то время как сканеры обеспечивают ввод в ПЭВМ готовых изображений, *графические планшеты*, или *диджитайзеры* (от англ. digitizer — цифровой преобразователь), автоматизируют их создание. Следовательно, графический планшет (частично) является альтернативой манипуляторам, а именно тем из них, которые способны формировать изображения. Работа с графическим планшетом аналогична рисованию карандашом или ручкой, а поэтому более удобна, чем с манипулятором (конечно, это замечание касается только создания рисунков). Графические планшеты являются факультативными ПУ. Они намного упрощают (по сравнению с написанием специальных программ) ввод в ПЭВМ графической информации, состоящей из линий, т.е. штриховых рисунков.

Графический планшет состоит из прямоугольного корпуса, на котором расположены наклонная рабочая поверхность и панель управления, а внутри — электронный блок. Для формирования рисунка служит специальное перо, подключаемое к планшету при помощи гибкого шнура. Сам

же планшет обычно подсоединяется к ПЭВМ через АИ RS232C. Для облегчения ввода сложных изображений на рабочую поверхность может быть нанесена вспомогательная координатная сетка.

Наиболее распространены графические планшеты, ввод информации в которых основан на использовании *пьезоэлектрического эффекта*. В этом случае под пластиной рабочей поверхности находится пластина пьезоэлектрика, к которой приложена сетка из тонких проводников. При касании пером рабочей поверхности на ближайшем пересечении проводников возникает разность потенциалов, в результате чего электронным блоком обнаруживаются координаты касания, которые с помощью программного драйвера вводятся в ПЭВМ и отображаются в виде точки на экране дисплея. Разрешающая способность простых графических планшетов составляет около 100, а профессиональных моделей — 400 и более линий на дюйм. Именно разрешение наряду с размерами рабочей поверхности и стоимостью являются основными показателями качества данных устройств.

С помощью дополнительного ПО пользователю могут быть предоставлены различные сервисные функции, например, закрашивание сформированных фигур, их штриховка и т.п.

Возможны планшеты для ввода текстовой информации, которые могут быть специализированными или обычными графическими, но способными совместно с соответствующими программными средствами распознавать символы.

Планшеты, обеспечивающие ввод текстовой информации, могут прийти на смену клавиатуре ПЭВМ, когда будет окончательно решена задача распознавания не только печатных, но и рукописных символов.

Так, американская компания GRiD Systems уже выпустила блокнотную ПЭВМ GRiDPad, которая вместо клавиатуры оборудована электронным пером. Конечно, пользователям-непрофессионалам работать на такой машине удобнее. Однако пока ею могут восприниматься только стилизованные «квадратные» символы, а точность их восприятия составляет 95%. Так что пользователь, в совершенстве владеющий клавиатурой, сможет вводить информацию в машину с ее помощью существенно быстрее, чем посредством электронного пера. Вообще печатать на клавиатуре можно быстрее, чем писать, но для этого необходимы устойчивые навыки.

На пути ввода в ПЭВМ текстов посредством их написания заметных успехов добилась фирма Microsoft, создав опытный образец системы Pen Windows. Данная система обеспечивает распознавание обычных рукописных символов.

В этом же направлении, причем весьма успешно, работает и совместное предприятие «ПараГраф».

2.5.5. Сенсорные экраны

Сенсорный экран (touch screen) является функциональным аналогом светового пера в смысле указания точки на экране, но имеет гораздо меньшую разрешающую способность, так как вместо пера используется палец. Сенсорные экраны часто применяются при выборе пунктов меню и, таким образом, могут использоваться совместно с интерактивными (диалоговыми) программами вместо манипуляторов и клавиатуры.

В общем случае при работе с сенсорным экраном пользователь ПЭВМ касается пальцем курсора, буквы, числа или другой высвеченной на экране фигуры. Вне зависимости от физической природы принципов, положенных в основу функционирования сенсорного экрана, с его поверхностью связывается прямоугольная система координат. Координаты точки касания экрана фиксируются и передаются в ПЭВМ. Точкам координатной сетки тем или иным программным продуктом ставятся в соответствие какие-либо функции, или действия, выполнение которых инициируется при касании точек.

К основным характеристикам сенсорных экранов относят скорость преобразования касания в цифровую форму, разрешение и стоимость. Первые два показателя при ручном указании особого значения не имеют. При этом на передний план выступают такие характеристики, как удобство пользования экраном, его практичность и долговечность. Последние показатели в большой мере зависят от физических принципов, положенных в основу функционирования сенсорного экрана.

В соответствии с этим различают следующие типы сенсорных экранов:

1) резистивного типа;
2) с емкостными датчиками;
3) с акустическими датчиками;
4) с оптическими датчиками.

Сенсорный экран *резистивного типа* чаще всего выполняется в виде двух прозрачных майларовых пленок, размещаемых на внешней поверхности экрана дисплея. На каждой пленке имеются параллельные проводники одной из координат. Таким образом, относительно внешнего механического воздействия образуется резистивное матричное поле. Когда палец или какой-либо другой указатель прижимает одну пленку к другой, сопротивление между двумя ближайшими перпендикулярно расположенными проводниками изменяется, что фиксируется и передается в ПЭВМ.

В сенсорных экранах *с емкостными датчиками* на экран дисплея наносятся с определенной топологией методом вжигания тонкие прозрачные слои токопроводящего материала. В момент касания одного из таких участков происходит изменение емкости и координаты касания передаются в ПЭВМ.

В сенсорных экранах *с акустическими датчиками* вдоль двух перпендикулярных границ экрана расположены передатчики, излучающие акустические волны в неслышимом диапазоне, которые распространяются вдоль поверхности экрана. Любой касающийся экрана объект отражает эти волны, и они фиксируются акустическими приемниками, размещенными рядом с передатчиками. Таким образом однозначно определяются координаты касания. В других разновидностях акустических сенсорных экранов приемники могут находиться в плоскости экрана напротив передатчиков. В этом случае приемники реагируют на ослабление акустического сигнала, вызванного появлением касающегося экрана объекта.

В сенсорных экранах *с оптическими датчиками* вдоль двух перпендикулярных границ экрана располагаются светоизлучающие диоды, а напротив них, вдоль противоположных границ экрана — фотоприемники. Тем самым над поверхностью экрана формируется ортогональная сетка инфракрасных лучей. Если палец или другой указатель соприкасается с поверхностью экрана, то он пересекает определенные лучи, что идентифицируется электронным блоком и передается в ПЭВМ.

Сенсорные экраны с оптическими датчиками обладают высокой надежностью и долговечностью, хорошей разрешающей способностью и отсутствием каких-либо конструктивных элементов, ухудшающих видимость изображения. Однако в ранних моделях проблемой был параллакс, обусловленный выпуклостью экрана. Последние достижения технологии в значительной степени устранили эту проблему. Аналогичными характеристиками обладают и акустические сенсорные экраны.

В принципе сенсорный экран можно оформить в виде клавиатуры и использовать в качестве ее заменителя. Но тогда любое случайное касание клавиши приведет к ошибочному вводу символа.

2.5.6. Средства речевого ввода

Средства речевого ввода (устройства речевого ввода и необходимые программные средства) открывают широкие возможности и повышают удобство общения с ПЭВМ для руководителей высокого ранга. В будущем, видимо, эти средства вытеснят клавиатуры в силу гораздо большей их скорости и удобства ввода информации в ПЭВМ. Пока же возможности средств речевого ввода весьма узки и они используются, в основном, для ввода ограниченного набора команд.

Средства речевого ввода оцениваются и (частично) классифицируются по следующим параметрам:

1) возможности распознавать слитную речь;
2) степени зависимости от диктора;
3) быстродействию;
4) объему словаря;
5) вероятности ошибок интерпретации слов;
6) стоимости.

По первому параметру средства речевого ввода делятся на две основные группы: средства, обеспечивающие распознавание *непрерывной, слитной речи*, и средства для распознавания *изолированных слов* (команд), разделенных искусственными паузами. Решение первой задачи существенно сложнее, так как необходимо не только идентифицировать последовательность звуков, но и подвергать предложение анализу (в том числе и смысловому) для разделения его на слова.

По второму параметру средства речевого ввода подразделяются на *зависимые* и *не зависимые от диктора*. Зависимость от диктора практического интереса не представляет. Не зависимые от диктора средства, в свою очередь, делятся на средства без подстройки под диктора и средства с подстройкой. Первые характеризуются низкой точностью распознавания речи, ресурсоемки и не обеспечивают большой словарный запас. Вторые же являются наиболее перспективными. Они осуществляют «привыкание» к тому или иному пользователю путем неоднократного повторения эталонных слов и запоминания его особенностей. Приступая к использованию настроенной таким образом системы, пользователь вводит в нее свой идентификатор или фамилию, при помощи чего осуществляется автоматическая адаптация к нему.

В настоящее время различными изготовителями выпускается широкая номенклатура средств речевого ввода.

Так, например, американская фирма Logical Business Machines предлагает систему Voicecraft для семейства PC IBM. Она способна распознавать до 32000 слов, которые размещаются в наборе словарей емкостью 500 слов каждый. Словари хранятся в запоминающем устройстве, и любой из них включается в работу по мере необходимости. На распознавание слова затрачивается 200 мс. Для семейства PS/2 IBM фирмой Dragon Dictate предлагается система Dragon Dictate-MCA, которая может распознавать одновременно до 30000 слов при словаре в 80000 слов, причем на идентификацию слова требуется не более 2 с. Стоит такая система 9000 долл.

Устройство речевого ввода для отечественной учебной ПЭВМ «Агат» представляет собой одноплатный модуль размером 250x125 мм, вставляемый в гнездо расширения ПЭВМ. Оно позволяет распознавать отдельные слова или короткие словосочетания и является адаптируемым к диктору. Максимальная длительность речевого сигнала, воспринимаемого устройством, сотавляет 1,5 с. Минимальная длительность паузы между отдельными словами — 0,21 с. Объем словаря — 64 слова, время же распознавания слова не превышает 1 с. Точность распознавания лежит в пределах 85 — 100 %.

2.6. Устройства вывода информации

Данный подраздел посвящен рассмотрению устройств вывода информации, наиболее часто используемых в ПЭВМ, а именно: дисплеев, принтеров, графопостроителей и синтезаторов звука. *Дисплей* является неотъемлемым атрибутом любой ПЭВМ. Без *принтера* в принципе можно обойтись, если не требуется долговременно хранить (документировать) выданную компьютером информацию, а достаточно лишь увидеть ее на экране дисплея. Однако такие ситуации крайне редки и поэтому ПЭВМ, как правило, комплектуются и принтерами. Если же машина редлагается без данного устройства, то покупатель всегда может приобрести его дополнительно у этого же или у любого из многих других поставщиков. *Графопостроители* же пользователи ПЭВМ приобретают значительно реже: только при потребности в получении высококачественных рисунков для различных приложений. Простейшие *синтезаторы звука* встраиваются во все современные ПЭВМ. За дополнительную плату можно получить более совершенные синтезаторы вплоть до синтезаторов речи.

2.6.1. Дисплеи и дисплейные адаптеры

Дисплеем называют устройство визуализации (отображения) текстовой и графической информации без ее долговременной фиксации. Отсутствие долговременной фиксации информации означает ее исчезновение при выключении питания или при выводе новой информации.

Дисплей является основным ПУ ПЭВМ и служит как для отображения информации, вводимой посредством клавиатуры или других устройств ввода, так и для выдачи пользователю сообщений, а также для вывода полученных в ходе выполнения программ результатов.

В бытовых ПЭВМ в качестве дисплеев, как правило, используются обычные телевизоры. В ППЭВМ же применяются специальные устройства. Независимо от физических принципов формирования изображения дисплей состоит из двух основных частей — экрана и электронного блока, размещенных в одном корпусе. Подключается дисплей к ПЭВМ через *дисплейный адаптер (видеоадаптер*, или *видеоконтроллер)*.

Часто вместо термина «дисплей» употребляют термины *«монитор» («видеомонитор»)* или *«терминал» («видеотерминал»)*. **Монитором** называют устройство, применяемое для контроля какого-либо процесса и управления системой. Конструктивно — это либо совокупность дисплея и клавиатуры, либо просто дисплей. **Терминалом** же называется (обычно удаленное) устройство ввода-вывода данных для взаимодействия пользователя с системой. Так как в ПЭВМ функции управления и контроля, а также ввода-вывода данных совмещены в одних и тех же устройствах, то монитор, терминал и дисплей можно считать синонимами, хотя в общем случае эти термины не эквивалентны.

В дальнейшем мы будем употреблять еще один термин — *консоль*. **Консолью** называется рабочее место (совокупность УВВ), с которого осуществляется контроль и управление функционированием вычислительной системы. Понятие «консоль» аналогично понятию «монитор», но консоль, в первую очередь, предназначена именно для управления. В данном случае предполагается, что процесс контроля является подчиненным по отношению к процессу управления. Монитор же выполняет главным образом функции контроля. В ПЭВМ стандартной консолью является совокупность дисплея и клавиатуры, причем клавиатура — обязательный компонент.

По функциональному назначению (функциональным возможностям) дисплеи подразделяются на *алфавитно-цифровые* и *графические*. Первые способны воспроизводить только ограниченный набор символов. Вторые же являются гораздо более гибкими. Они в состоянии отображать как графическую, так и, что вполне естественно, текстовую информацию. В настоящее время графические дисплеи в ПЭВМ практически вытеснили алфавитно-цифровые.

По количеству воспроизводимых цветов различают монохромные (одноцветные) и цветные дисплеи. *Монохромные* устройства способны воспроизводить информацию только в каком-либо одном цвете, возможно, с различными градациями яркости. Широко распространены черно-белые экраны, а также зеленые и желтые. *Цветные* дисплеи обеспечивают выдачу на экран информации одновременно в нескольких цветах.

По физическим принципам формирования изображения существуют:

1) дисплеи на базе электронно-лучевой трубки;
2) жидкокристаллические дисплеи;
3) плазменные (газоразрядные) дисплеи;
4) электролюминесцентные дисплеи.

Дисплеи на базе электронно-лучевой трубки традиционны, а принцип их работы аналогичен бытовому телевизору. В электронно-лучевой трубке формируется луч (или три луча для цветных трубок), управляя перемещением и интенсивностью которого можно получить изображение на люминофорном экране. Для дисплеев данного типа графические изображения могут формироваться двумя способами. В *векторном* дисплее электронный луч непрерывно «вырисовывает» контур изображения. Само изображение формируется из отдельных элементарных отрезков (векторов). В *растровых* же дисплеях изображение получается с помощью матрицы точек, которые могут «светиться», а могут быть невидимыми: электронный луч пробегает по строкам экрана, подсвечивая требуемые зерна (точки) люминофора. В этом случае и небольшом разрешении при

воспроизведении ряда фигур хорошо заметен эффект «мозаичности». Цветные экраны имеют зерна трех цветов: красного, зеленого и желтого, собранные в триады. Каждый из трех электронных лучей отвечает за свой цвет, подсвечивая при необходимости «свои» зерна. Манипулируя яркостью зерен, можно сформировать точку любого цвета. Первоначально дисплеи на базе электронно-лучевой трубки в отличие от бытовых телевизоров имели цифровой видеовход. Сейчас же в наиболее совершенных моделях дисплеев осуществлен возврат к аналоговым видеовходам (имеется в виду стандарт VGA). Дисплеи на базе электронно-лучевой трубки громоздки, потребляют много энергии, но имеют хорошие технические характеристики.

Жидкокристаллический экран (индикатор) представляет собой совокупность сегментов для воспроизведения элементарных частей изображения (в частности, точек). Каждый сегмент состоит из нормально прозрачной анизотропной жидкости, заключенной между двумя прозрачными электродами. При подаче на электроды напряжения коэффициент отражения жидкости меняется и сегмент при освещении его внешним источником света темнеет. Индикаторы данного типа в отличие от других являются не активными, а пассивными (изображение «проявляется» только при внешнем освещении). По сравнению с другими жидкокристаллические индикаторы характеризуются малыми потребляемой мощностью и массой. Основная проблема для них — невысокая контрастность изображения. К настоящему времени предложены не только монохромные, но и цветные жидкокристаллические дисплеи. В производстве цветных устройств преуспели японские фирмы. Индикаторы данного типа часто применяют в электронных часах и калькуляторах.

В ПЭВМ в последнее время широкое распространение получили жидкокристаллические индикаторы *с обратной (задней) подсветкой* (backlit). Их конструктивная особенность заключается в том, что за экраном размещается источник света, а сам экран состоит из жидкокристаллических ячеек, которые в нормальном состоянии являются непрозрачными. При приложении к такой ячейке напряжения она начинает пропускать свет, что и приводит к получению изображения на экране. Такой принцип формирования изображения облегчает создание цветных дисплеев. Действительно, достаточно на экране иметь тройки жидкокристаллических ячеек, обеспечивающие на просвет воспроизведение основных цветов (красного, зеленого и синего).

В 1990 г. японская фирма Dainippon Inc. & Chemicals завершила разработку полимерной сети, которой можно обвить жидкий кристалл как паутиной. Такой экран не требует поляризаторов и подсветки, а также потребляет меньше энергии.

Экран *плазменного дисплея* представляет собой матрицу газоразрядных элементов. При приложении к электродам газоразрядного элемента напряжения возникает электрический разряд красного или оранжевого свечения в газе, которым этот элемент заполнен. По сравнению с жидкокристаллическими плазменные индикаторы имеют более высокую контрастность, однако обладают и повышенным энергопотреблением.

Экран *люминесцентного дисплея* состоит из матрицы активных индикаторов, дающих яркие изображения с высокой разрешающей способностью. Они имеют высокую механическую прочность и надежность, однако отличаются большим энергопотреблением и высокой стоимостью. Наряду с монохромными имеются и цветные люминесцентные дисплеи.

В стационарных ПЭВМ в настоящее время применяются дисплеи на базе электронно-лучевой трубки. Переносные ПЭВМ снабжаются такими же устройствами или плазменными дисплеями. В наколенных и более компактных ПЭВМ используются главным образом жидкокристаллические и изредка плазменные индикаторы. Электролюминесцентные дисплеи перспективны для использования в различных классах малогабаритных ПЭВМ.

В зависимости от степени универсальности дисплеи подразделяются на однорежимные и многорежимные.

Однорежимный дисплей способен работать только совместно с видеоадаптером одного типа.

Многорежимные дисплеи совместимы с видеоадаптерами различных типов (см. ниже).

Основными техническими характеристиками дисплеев являются:

1) разрешающая способность;
2) количество воспроизводимых цветов или градаций яркости;
3) размер экрана (как правило, по диагонали);
4) масса и габариты;
5) стоимость.

Разрешение дисплея измеряется в различных единицах. Для алфавитно-цифровых устройств указывается число воспроизводимых символов в строке и строк на экране. Для графических дисплеев указывается количество высвечиваемых точек по горизонтали и по вертикали. В дальнейшем разрешение будет указываться в виде m x n, где m относится к горизонтали, а n — к вертикали. Это же справедливо и для матрицы точек при представлении символа на графическом дисплее. Альтернативной единицей измерения разрешающей способности является количество воспроизводимых точек (по вертикали или горизонтали) на единицу длины.

Есть еще один немаловажный показатель качества дисплеев на базе электронно-лучевой трубки, а именно, *частота сканирования* (частота вертикальной и частота горизонтальной развертки). Чем выше разрешение дисплея, тем выше должна быть и частота (скорость) сканирования для обеспечения приемлемого качества изображения (без мерцания). Поэтому от частоты сканирования во многом зависит степень универсальности монитора.

Возможность многорежимной работы дисплея на базе электронно-лучевой трубки определяется его способностью воспринимать синхроимпульсы для горизонтальной и вертикальной развертки с различной частотой. В соответствии с этим различают:

— *дисплеи с фиксированной частотой*;

— *мультичастотные дисплеи*, способные работать на нескольких фиксированных частотах для горизонтальной и вертикальной развертки;

— *мультисканирующие дисплеи*, обеспечивающие работу в диапазонах частот для горизонтальной и вертикальной развертки.

Дисплеи с фиксированной частотой могут быть только однорежимными; другие же поддерживают несколько режимов работы.

При формировании изображения на основе других физических принципов многорежимность определяется возможностями управления индикаторами.

Разрешающая способность высококачественных графических дисплеев на базе электронно-лучевой трубки достигла величины, позволяющей получить изображение фотографического качества.

Количество воспроизводимых цветов или *градаций яркости* зависит от возможностей по управлению интенсивностью электронных лучей, прозрачностью жидкокристаллических индикаторов или яркостью других (активных) индикаторов.

Для цветных мониторов удобно пользоваться двумя понятиями — базовая и рабочая палитры. *Базовая палитра* представляет собой совокупность цветов, которые могут отображаться на экране. Но цвета базовой палитры в общем случае нельзя отобразить на дисплее одновременно. Обычно из базовой палитры формируется *рабочая палитра*, цвета которой могут сочетаться на дисплее одновременно и в любой комбинации. Как правило, рабочая палитра существенно уже базовой. Менять рабочую палитру можно программными средствами.

Кратко охарактеризуем некоторые из наиболее совершенных мониторов на базе электронно-лучевой трубки.

Черно-белый дисплей UHR-2007 фирмы MegaScan имеет разрешение 7,9 точка/мм (2560x2048 точек на экране) и позволяет воспроизвести 256 оттенков серой шкалы. Размер экрана по диагонали — 48,3 см. Качество изображения выше качества фотографии, полученной с 35-мм пленки.

Черно-белый монитор UHR-3000 той же фирмы обеспечивает воспроизведение всего двух оттенков серой шкалы, но зато обладает разрешением 11,8 точка/мм (4096x3300 точек на экране), что полностью совпадает с разрешающей способностью лазерных принтеров.

Цветной дисплей Lundy 1612 американской фирмы Lundy Electronics & Systems имеет разрешение 1600x1200 точек при рабочей палитре 16 цветов и базовой палитре из 4096 оттенков. Размер экрана по диагонали — 19 дюймов. Программные средства позволяют получить рабочую палитру из 256 при базовой палитре из 16 млн. цветов, но разрешение в этом случае снижается до 1024x768 точек. Цена монитора вместе с адаптером составляет 9950 долл.

Наилучший монохромный дисплей этой же фирмы обладает разрешением 4096x4096 точек.

Лучшим многорежимным монитором с поддержкой всех видеостандартов (вплоть до разрешения 1024x768 точек) в настоящее время является изделие MyltiSync 5D фирмы NEC. Его стоимость составляет 3699 долл.

Конечно, в ПЭВМ обычно используются дисплеи среднего качества по цене в несколько сот долларов.

Интересны изделия и другого типа, основной особенностью которых являются малые масса-габаритные показатели.

Так, например, американская фирма Reflection Technology выпустила устройство размером 31x28x81 мм с разрешением 720x280 (стандартно) или 1024x280 точек. Его масса составляет всего лишь 60 г. Такой дисплей можно прикрепить к головным телефонам (наушникам). Он располагается на расстоянии 2 — 3 см от глаз, а у наблюдателя создается впечатление, что изображение находится на расстоянии около метра.

Возможности ПЭВМ по отображению информации определяются совокупностью и совместимостью технических характеристик дисплея и его адаптера (т.е. *видеосистемы* в целом). В настоящее время различными производителями предлагается широкий спектр видеоадаптеров. Любой **адаптер** содержит *видеопамять*, хранящую воспроизводимую на экране информацию. Ее объем может достигать нескольких Мбайт. Каждой точке экрана или знакоместу соответствует поле видеопамяти (несколько бит или байт), в котором хранится элемент отображения, или изображения (pixel — сокращение от англ. picture element). *Элемент отображения* определяет режим высвечивания и цвет точки либо символа. Видеопамять логически содержится в одном адресном пространстве с ОП. Допускается записывать данные в видеопамять и считывать информацию из нее программными средствами. То, что находится в видеопамяти, немедленно отображается на экране.

Видеоадаптеры делятся на две большие группы — *алфавитно-цифровые* и *графические*. Они управляют соответствующими типами дисплеев.

Графический адаптер обычно может работать в нескольких текстовых и нескольких графических режимах, которые различаются разрешением, а также цветовыми (яркостными) возможностями.

В текстовых режимах имеющаяся видеопамять полностью не используется, поэтому можно организовать в ней несколько *страниц*, что позволяет ускорить смену изображений на экране путем предварительного заполнения страниц требуемой информацией и последующего переключения воспроизведения со страницы на страницу. Страничная организация видеопамяти допустима также в графических режимах с пониженным разрешением при избыточности объема видеопамяти.

Основные технические характеристики базовых моделей видеоадаптеров, ставших стандартными, представлены в табл. 2.5.

Таблица 2.5

Основные технические характеристики распространенных видеоадаптеров

Тип	Разрешение	Число цветов (градаций яркости) в рабочей палитре	Число цветов (градаций яркости) в базовой палитре	Матрица символов, точек	Объем видеопамяти, Кбайт
MDA	80x25 симв.	2	2	9x14	4
CGA	80x25 симв.	16	16	8x8	16
	40x25 симв.	16			
	320x200 точек	4			
	640x200 точек	2			
EGA	80x25 симв.	16	64	8x14	64-256
	80x35 симв.	16			
	80x43 симв.	16			
	640x200 точек	16			
	640x350 точек	16			
	640x350 точек	4			
	640x350 точек	2			
VGA	80x25 симв.	16	256K	8x16	256-512
	80x40 симв.	16			
	80x50 симв.	16			
	640x200 точек	16			
	640x350 точек	16			
	640x480 точек	16			
8514/A	80x25 симв.	256	нет данных	12x20	512-1024
	146x51 симв.	256			
	1024x768 точек	16			
	1024x768 точек	256			
XGA	80x25 симв.	256	нет данных	12x20	512-1024
	132x25 симв.	256			
	132x51 симв.	256			
	146x51 симв.	256			
	1024x768 точек	16			
	1024x768 точек	256			
	640x480 точек	64K			

Адаптеры MDA, CGA и EGA разработаны фирмой IBM для использования в семействе ПЭВМ PC.

Адаптер монохромного дисплея MDA (Monochrome Display Adapter) был применен в модели IBM PC. Он обеспечивает только воспроизведение алфавитно-цифровой (текстовой) информации в монохромном режиме. Предполагается, что дисплей все-таки растровый, с разрешением 720x350 точек, причем на экран можно выдать 25 строк текста по 80 символов в строке. Каждый символ представляется матрицей размером 7x9 точек в ячейках поля 9x14 точек. Количество воспроизводимых символов для MDA (как и для других адаптеров) составляет 256, включая символы псевдографики, в частности, линии. Можно обеспечить мигание, подчеркивание, утолщение и инверсную выдачу каждого символа. В настоящее время MDA практически не используется. Отечественные ППЭВМ вначале снабжались видеоадаптером стандарта MDA. MDA предлагаются сейчас за 60 — 80 долл.

С целью использования графических возможностей фирма Hercules Computer Technology в 1982 г. разработала *монохромный графический адаптер* HGA (Hercules Graphics Adapter), или HGC (Hercules Graphics Controller), с разрешением 720x348 точек. Он обеспечивал число градаций яркости, равное двум, и имел матрицу символа размером 9x14 точек (включая промежутки). Его дополнительные текстовые режимы характеризуются разрешением 132x25 и 132x44 символов. Несколько забегая вперед, чтобы не возвращаться к изделиям этой фирмы,

отметим, что цветной вариант HGA, созданный в 1984 г., являлся аналогом адаптера CGA. После этого в 1986 г. фирма Hercules предложила монохромный режим RamFont (реализован в плате Graphics Plus), состоящий в том, что для указания символа используется не 8, а 12 бит. Это увеличивает число воспроизводимых символов с 256 (=2^8) до 4 К (=2^12). Поэтому появляется возможность имитировать графический режим работы посредством более быстрого текстового режима с большим количеством символов псевдографики. В том же 1986 г. Hercules предложила новую плату InColor, реализующую режим RamFont (с 12 К символами) в цветном варианте. Рабочую палитру составляли 16 цветов, а базовую — 64 цвета. Эта плата совместима с адаптером EGA. Сравнительно недавно фирма Hercules взялась за производство недорогих адаптеров стандарта VGA. Несмотря на все новшества определенную популярность приобрел только первоначальный HGA, предлагаемый сейчас за 80 — 100 долл.

Первым из получивших распространение цветных графических адаптеров является CGA (Color Graphics Adapter — *цветной графический адаптер*), используемый главным образом в модели ПЭВМ IBM PC XT и в совместимых с ней, в частности, в выпускаемых в настоящее время ПЭВМ ЕС. Он обеспечивает максимальное разрешение 640x200 точек, а максимальное число цветов в рабочей палитре составляет 16 (но не в графических режимах — см. табл. 2.5). Этот адаптер способен работать в двух текстовых и двух графических режимах, причем все они дополнительно подразделяются на подрежимы в зависимости от того, какой дисплей используется — монохромный или цветной. В случае монохромного дисплея каждому цвету соответствует своя градация яркости. Таким образом, в общей сложности имеется 7 режимов работы. Существенные недостатки CGA состоят в небольшой разрешающей способности при выводе текста (всего 7x7 точек на символ в матрице 8x8), приводящей к нечетким изображениям, а также в низком разрешении при выводе цветных графических изображений (320x200 точек). Данный адаптер совместим с программными средствами, разработанными для MDA, а его организация подробнее рассматривается ниже. Стоимость CGA составляет 80 — 100 долл.

Созданный в 1984 г. для использования в модели ПЭВМ IBM PC AT *усовершенствованный графический адаптер* EGA (Enhanced Graphics Adapter) имеет различные графические режимы работы в зависимости от размера видеопамяти и потребного количества страниц (см. табл. 2.5). Максимальное разрешение составляет 640x350 точек, а максимальное число цветов в рабочей палитре (или градаций яркости в монохромном варианте) — 16 из общего количества в 64 цвета. В текстовом режиме данный адаптер обеспечивает при желании вывод информации в 43 строки, но символы при этом становятся более мелкими. Адаптер EGA совместим с ПО, разработанным в расчете на MDA и CGA. Цена EGA сейчас не превышает 500 долл.

Расширением стандарта EGA явился адаптер EGA Plus (разработчик — не IBM), отличающийся от своего прототипа только повышенной разрешающей способностью по вертикали (640x480 точек вместо 640x350).

В том же 1984 г. фирма Vermont Microsystems предложила еще один видеоадаптер — PGA (Professional Graphics Adapter — *профессиональный графический адаптер*), выпущенный и фирмой IBM. Он является развитием EGA Plus только в смысле числа одновременно воспроизводимых цветов, которое для него составляет 256. Однако эта попытка предложить новый графический стандарт осталась незамеченной, и данный адаптер не получил широкого распространения. Его цена составляет 2000 — 3000 долл.

В связи с выпуском в 1987 г. нового семейства ПЭВМ PS/2 фирма IBM разработала и новые графические адаптеры, среди которых VGA, MCGA и IBM 8514/A.

VGA (Video Graphics Adapter или Array — *видеографический адаптер* или *массив*) аналогичен устройству EGA Plus, но имеет гораздо большее количество цветов для выбора (см. табл. 2.5). Режимы с пониженным разрешением характеризуются наличием страничной организации видеопамяти. В монохромном варианте (режиме) VGA обеспечивает 64 градации яркости. Допустимо отображение текста в 50 строк. Данный адаптер эмулирует более ранние форматы CGA и EGA. VGA может применяться во всех моделях PS/2, за исключением самых младших.

Вариантом адаптера VGA явилась плата MCGA (Multi Color Graphics Adapter или Array — *многоцветный графический адаптер* или *массив*), используемая в младших моделях PS/2. Этот адаптер обеспечивает одновременное воспроизведение 256 цветов при разрешении 320x200 точек или двух цветов при разрешении 640x480 точек. Несмотря на разрешающую способность, всего лишь равную CGA, 256 цветов MCGA производят эффект фотографии благодаря множеству оттенков. Матрица символов MCGA, как и в VGA, составляет 8x16 точек, что обеспечивает высокое качество воспроизведения текстовой информации.

VGA является первым видеоадаптером, имеющим квадратную, а не прямоугольную точку. Это означает, что правильная окружность, отображенная на экране, будет действительно окружностью, а не эллипсом.

Видеоформат VGA быстро приобрел популярность и получил дальнейшее развитие другими фирмами. Среди расширений VGA отметим следующие:

1) усовершенствованный VGA (EVGA — Enhanced VGA), имеющий разрешение VGA и цветовые возможности MCGA (т.е. 640x480 точек при 256 цветах одновременно);

2) Super VGA (VESA), имеющий повышенное по сравнению с VGA разрешение (800x600 и даже 1024x768 точек) при тех же 16 цветах одновременно;

3) дополнительно усовершенствованный VGA (Further Enhanced VGA), обладающий разрешением Super VGA (800x600) и цветовыми возможностями EVGA (256 цветов одновременно).

Терминология для расширений стандарта VGA не устоялась. Часто используются и такие трактовки:

1) VGA Plus для разрешения 800x600 точек при 16 цветах или 640x400 точек при 256 цветах;
2) Super VGA для разрешения 1024x768 точек при 16 цветах или 800x600 точек при 256 цветах.

Под Super VGA иногда понимают любое расширение стандарта VGA.

Разрешение 800x600 точек обеспечивает возможность вывода текста в 132 колонки.

Для старших моделей PS/2 в качестве альтернативы VGA предложен *видеоадаптер IBM 8514/A*, который по сравнению с VGA обладает существенно бо́льшим разрешением (1024x768 точек) при 256 цветах одновременно и поэтому позволяет воспроизводить текст на 51 строке по 146 символов в каждой. На плате 8514/A имеется свой собственный графический контроллер для повышения скорости работы. Адаптеры с такими возможностями довольно дороги: цены большинства из них лежат в пределах от 1100 до 2400 долл.

В 1990 г. фирма IBM предложила еще один графический стандарт — XGA (eXtended Graphics Adapter или Array — *расширенный графический адаптер* или *массив*), который уже устанавливается в старшие модели семейства PS/2 IBM. Основные особенности XGA состоят в ориентации на интерфейсную систему MS Windows и в повышенном быстродействии.

Наиболее популярны сейчас стандарт VGA и его расширения.

Для высококачественных дисплеев разработан ряд видеоадаптеров с еще более высокими техническими характеристиками. Среди них адаптер CAD-Genius западногерманской фирмы Projekt-Team Electronic GmbH, обеспечивающий разрешение 2048x2048 точек при 16 цветах или 2048x1024 точек при 256 цветах одновременно. Другим примером является плата UDC-800 американской фирмы Univision Technologies, поддерживающая разрешение 2048x1536 точек, имеющая видеопамять емкостью 4 Мбайт и стоящая 6995 долл.

Фирма Apple Computer придерживается своих высококачественных видеоформатов.

Несмотря на наличие высококачественных дисплеев и видеоадаптеров разработка программных средств для них заметно отстает, что затрудняет наиболее полное использование предоставляемых ими возможностей. Так, например, большинство версий ОС MS-DOS поддерживают лишь адаптер CGA и только самые последние версии — EGA/VGA. В связи с этим разработчики тех или иных программных продуктов учитывают новые видеоадаптеры непосредственно в своих программных изделиях.

В качестве примера несколько подробнее рассмотрим режимы работы и организацию адаптера CGA. Изложенные при этом принципы справедливы и для других видеоадаптеров (имеются главным образом только количественные отличия).

Режимы работы адаптера CGA представлены в табл. 2.6.

Таблица 2.6

Режимы работы адаптера CGA

Номер режима	Вид режима	Разрешение, символов или точек	Число цветов или градаций яркости	Тип дисплея
0	текстовый	40x25	16	монохромный
1	текстовый	40x25	16	цветной
2	текстовый	80x25	16	монохромный
3	текстовый	80x25	16	цветной
4	графический	320x200	4	цветной
5	графический	320x200	4	монохромный
6	графический	640x200	2	монохромный

В случае *текстовых* режимов каждый отображаемый на экране символ представляется словом (двумя байтами). Младший (четный) байт слова содержит *код символа* в ASCII, а старший (нечетный) — *атрибуты*. Атрибуты указывают, в каком виде символ должен отображаться на экране.

Структура байта атрибутов изображена на рис. 2.9. *Цвет фона* определяет цвет поля, на котором выводится данный символ. Допустимы 8 фоновых (более темных) из общего набора 16 цветов. *Цвет символа* задается аналогично, но к его коду относится дополнительный бит I, определяющий *интенсивность* указанного цвета. При единичном значении бита I интенсивность цвета повышается, а при нулевом — совпадает с набором фоновых цветов. Поэтому в качестве цвета символа оказывается допустимым любой из 16 возможных цветов. Установка в единичное состояние бита B задает *мерцание* символа с определенной частотой. Исходя из рассмотренного можно сделать вывод о том, что в текстовых режимах допустимо управление представлением каждого отдельного символа.

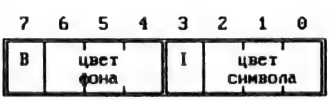


Рис. 2.9. Структура байта атрибутов для адаптера CGA

Таким образом, размер элемента отображения CGA в текстовых режимах составляет 16 бит. Эту величину иначе именуют *глубиной*.

Логическая организация видеопамяти адаптера CGA для текстовых режимов представлена на рис. 2.10. При разрешении 80x25 на экране отображаются 2000 символов, а при разрешении 40x25 — 1000 символов. Поэтому для запоминания содержимого экрана требуется соответственно 4 Кбайт или 2 Кбайт. Остающаяся от 16 Кбайт незанятая область видеопамяти при этом может использоваться в качестве дополнительных страниц, о чем упоминалось раньше.

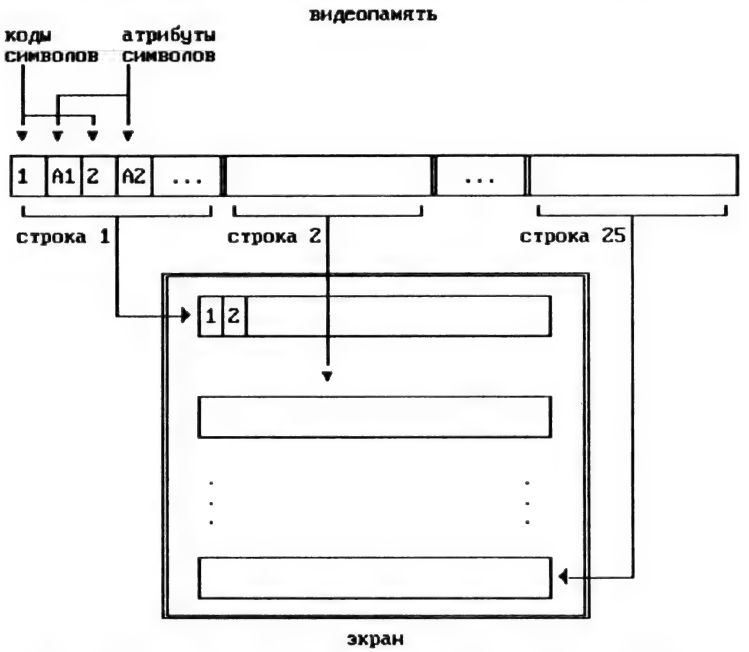


Рис. 2.10. Организация видеопамяти адаптера CGA в текстовых режимах

Информация о правилах отображения каждого символа с тем или иным кодом хранится в специальной памяти адаптера, называемой знакогенератором. Соответствие между кодами символов и их внешним представлением в виде совокупности точек, определяемым знакогенератором, можно изменить, начиная с адаптеров EGA (для отечественных ПЭВМ — с CGA).

Цветные *графические режимы* CGA подразделяются на подрежимы в соответствии с цветовыми параметрами. Рабочая палитра CGA представляет собой упорядоченный перечень четырех цветов, первый из которых считается фоновым. Эти четыре цвета можно использовать на экране одновременно. При необходимости сменить цвета нужно переключиться (программно) на другую рабочую палитру, однако при этом изменится оцветовка всей информации, выведенной на экран. Дело в том, что каждая точка, отображаемая на экране дисплея, представляется кодом в видеопамяти, соответствие между которым и цветом точки как раз и определяется рабочей палитрой. Фон в данном случае — понятие условное и имеет смысл только при очистке видеопамяти. Адаптер CGA имеет несколько (2 или 4) заготовок рабочих палитр, содержащих три цвета (за исключением фонового). Фоновый цвет при выборе палитры устанавливается программистом.

Для хранения кода точки достаточно иметь элемент отображения длиной всего 2 бита, так как по сути код точки обозначает ее цвет, а возможны всего 4 цвета. Поэтому коды всех 320x200 точек на экране укладываются в объем видеопамяти, составляющий 16 Кбайт. При разрешении же 640x200 точек глубина составляет 1 бит, что не позволяет указать более двух кодов. Код числа 0 соответствует фоновому цвету (первому цвету в палитре), а код числа 3 — последнему цвету в палитре.

Логическая организация видеопамяти адаптера CGA в цветных графических режимах изображена на рис. 2.11. Информация хранится в ней в виде двух больших блоков, в первом из которых содержится изображение, создаваемое на экране четными линиями, а во втором — нечетными. При этом для каждой линии сохраняется по 640 бит независимо от того или иного графического режима.

И, наконец, кратко рассмотрим принцип, в соответствии с которым осуществляется вывод на экран текста в графическом режиме. В ПЗУ ПЭВМ записывается представление изображения в виде точек для первых 128 символов кода ASCII. Пользователю же предоставляется возможность записать в определенную область ОЗУ «таблицу» соответствия для оставшихся 128 символов. При выводе на экран символа по его коду осуществляется вход в одну из этих таблиц. Из таблицы выбирается последовательность элементов отображения для представления данного символа, и эта последователь-

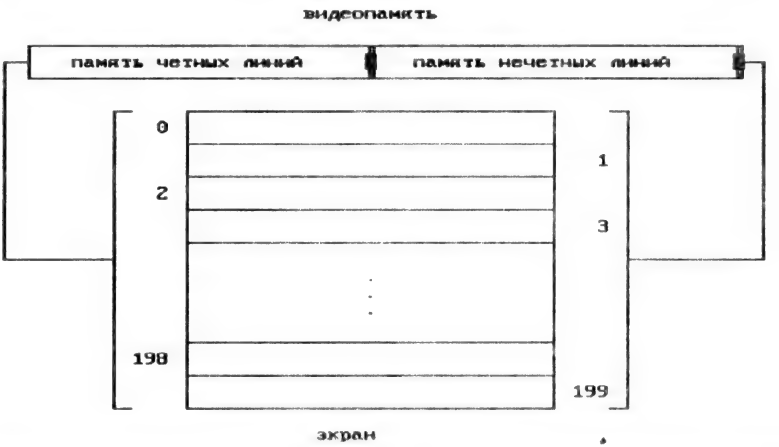


Рис. 2.11. Организация видеопамяти адаптера CGA в цветных графических режимах

ность записывается в видеопамять. Реализация аналогичного принципа программными средствами позволяет запрограммировать любые шрифты, что и используется во многих графических пакетах.

2.6.2. Печатающие устройства

Печатающие устройства, или *принтеры* (от англ. printer) предназначены для вывода алфавитно-цифровой (текстовой) и графической информации на бумагу или подобный ей носитель. Следовательно, принтер в отличие от дисплея позволяет получить *твердую копию* изображения практически с неограниченным временем хранения. Несмотря на то, что печатающие устройства в ПЭВМ считаются дополнительными ПУ, они, как правило, входят в комплект поставки и подключаются через параллельный АИ.

За довольно короткий срок принтер из простого печатающего устройства, которое лет тридцать назад примитивно стучало через красящую ленту молоточками по бумаге, выводя из огромной ЭВМ столбцы цифр вперемежку с малопонятными словами, превратился в многофункциональный инструмент для документирования текстовой и графической информации как в черно-белом, так и цветном виде.

Классификация выпускаемых для ПЭВМ принтеров по технологии печати приведена на рис. 2.12. Принтеры *ударного типа* характеризуются тем, что изображение на бумагу наносится механическим способом. Из них в ПЭВМ применяются устройства *с литерной печатью (литерные принтеры)* и *точечно-матричные принтеры*. В *безударных* принтерах передвижение бумаги и печатающей головки по-прежнему осуществляется механическим способом, но для формирования изображения на бумаге используются немеханические принципы. Наибольшее распространение в ПЭВМ получили следующие виды безударной технологии печати: *струйная*, *термографическая* и *электрофотографическая (лазерная)*. По причине высоких технических характеристик считаются перспективными и уже начали использоваться *электростатическая* и *магнитографическая* технологии. *Электрочувствительные принтеры* используются редко. Безударная технология бурно развивается, и именно с ней связывается достижение в области технических характеристик принтеров нового качественного уровня. Основные ее преимущества — высокая скорость печати и низкий уровень шума.

Качество черно-белой печати принтеров уже достигло фотографического уровня. Для цветной печати этот уровень будет достигнут в ближайшие годы.

По степени параллелизма в работе принтеры подразделяются на устройства *последовательного действия* (печатают посимвольно), *построчно печатающие* устройства (выводят строки целиком) и *постранично печатающие* устройства (сразу формируют страницу). В ПЭВМ наибольшее распространение получили устройства первого класса благодаря их простоте, компактности и дешевизне. Но, конечно, они обладают меньшей производительностью.

Конструктивно принтеры для ПЭВМ выполняются достаточно малогабаритными, что позволяет размещать их на столе, рядом с ПЭВМ. Выпускаются еще более компактные устройства для портативных ПЭВМ.

Основными техническими характеристиками принтеров являются:

1) принцип действия (в соответствии с только что рассмотренной классификацией);
2) цветовые возможности (черно-белые или цветные принтеры);
3) графические возможности или их отсутствие;
4) разрешающая способность;
5) качество печати, тесно связанное с предыдущим показателем и обобщающее его;
6) скорость печати (быстродействие);
7) стоимость.

Рис. 2.12. Классификация принтеров для ПВМ

Вместо *быстродействия* принтера лучше говорить о *производительности* печати, учитывающей не только собственно скорость печати, но и время выполнения других операций, в частности, время заправки бумаги. Некоторые модели принтеров осуществляют последнюю операцию автоматически.

Дополнительно к этому принтеры характеризуются следующими показателями:

1) емкостью буферной памяти;
2) стандартным набором шрифтов и возможностями формирования новых шрифтов;
3) форматом используемого листа бумаги, в частности, шириной каретки;
4) габаритными размерами и массой;
5) энергопотреблением;
6) уровнем акустического шума.

Ниже мы изучим все типы принтеров, приведем их сравнительную оценку и рассмотрим характеристики лучших моделей печатающих устройств. Особое внимание будет уделено точечно-матричным принтерам, получившим в настоящее время наибольшее распространение. Причины этого кроются в хорошем сочетании их технических характеристик. Точечно-матричные принтеры позволяют выводить как текст, так и графику в черно-белом или цветном виде, имеют низкую стоимость и относительно высокое быстродействие, а также не требуют высококвалифицированного обслуживания.

Литерные принтеры

Первой реализованной в коммерческих принтерах технологией печати была именно техника *литерной* печати. В больших ЭВМ используются высокоскоростные литерные печатающие устройства *параллельного* действия. В ПЭВМ же нашли применение главным образом только устройства *последовательного* действия.

Последовательная литерная технология печати заимствована, по сути дела, у пишущих машинок. Она состоит в печати сформированными символами — *литерами*. При этом способе печати производится удар по бумаге литерой через красящую ленту, в результате чего на бумаге остается контур символа. *Печатающие элементы (шрифтоносители)*, на которых размещены литеры всех печатных знаков, могут выполняться цилиндрическими (в виде барабана), шарообразными, лепестковыми (типа «ромашка»), ленточными или наперсткообразными (напоминающими волан для игры в бадминтон). Зачастую эти элементы делают съемными, что позволяет изменять виды шрифтов, наборы символов и языки. Однако такую смену нельзя осуществить оперативно (в ходе печати).

В настоящее время принтеры с лепестковым шрифтоносителем типа «ромашка» — самые распространенные среди литерных печатающих устройств. Каждая литера в таком принтере отформована на одном из концов металлического упругого лепестка, другой конец которого вмонтирован во втулку. Собранный таким образом шрифтоноситель представляет собой колесо, напоминающее ромашку. Нужный для печати символ выбирается путем поворота колеса, размещенного параллельно бумаге. Затем специальный молоточек ударяет по подведенному к нему лепестку, в результате чего символ через красящую ленту, находящуюся между колесом и бумагой, наносится на бумагу. После удара благодаря упругости лепестка он возвращается в исходное состояние.

Литерные принтеры обладают высокой надежностью, обеспечивают типографское качество печати и допускают смену шрифтов, хотя последнее не является удобным и простым. Однако

они имеют низкую скорость печати (10 — 60 символ/с), высокий уровень шума и сравнительно высокую стоимость (порядка 2000 долл., а иногда и выше), а также характеризуются отсутствием графических возможностей. Исключение составляют некоторые виды графических работ, если шрифт содержит символы псевдографики. Цветовые возможности также ограниченны, однако в принципе реализуемы путем использования многоцветной ленты и ее смещения относительно шрифтоносителя. Только ведь печать литерами, а не точками различного цвета, почти ничего не дает. В ряде типов принтеров используются механизмы для изменения интервалов печати (между символами в строке) и механизмы печати в обоих направлениях при перемещении печатающей головки.

Устройства литерного типа сейчас находят в ПЭВМ весьма ограниченное применение.

Приведем в качестве примера основные характеристики некоторых моделей литерных принтеров.

Принтер с лепестковым шрифтоносителем PR340 итальянской фирмы Olivetti имеет ширину каретки 330 мм и обеспечивает скорость печати 45 символ/с. Цена его составляет около 2500 долл.

Устройство Juki 6000 американской фирмы Juki Industries of America стоит всего 295 долл., весит 5,9 кг и имеет габариты 400x228x140 мм. Уровень шума составляет 62 дБ, а скорость печати — 10 символ/с. Он содержит 100-литерный лепестковый шрифтоноситель. Способен воспроизводить текст с плотностью 10, 12 и 15 символ/дюйм. Ширина каретки составляет 229 мм.

Лепестковый принтер Alphapro 101 фирмы Alphacom (США) стоит 399 долл., печатает со скоростью 20 символ/с и имеет каретку шириной 364 мм.

Точечно-матричные принтеры

Основным узлом *точечно-матричного принтера* является *печатающая головка*, которая перемещается по специальным направляющим вдоль печатаемой на бумаге строки, «вырисовывая» выводимую информацию по точкам через красящую ленту. После печати строки бумага продвигается и описанный процесс повторяется.

Печатающая головка содержит несколько *игл (штифтов)*, расположенных вертикально. Каждая игла управляется собственным электромагнитом. При необходимости отпечатать точку в ходе движения головки соответствующий электромагнит срабатывает, игла ударяет по красящей ленте и точка наносится на бумагу. Следовательно, принцип формирования изображений в точечно-матричных принтерах логически эквивалентен способу вывода информации на экран дисплея.

Первоначально принтеры данного типа имели головки с 7 иглами, а символы формировались матрицей размером 5x7 точек в ходе горизонтального перемещения головки. Затем их вытеснили 9-игольчатые устройства, и среди дешевых принтеров они сейчас наиболее распространены. Матрица символов в таких принтерах составляет 7x9 или 9x9 точек (возможны и другие варианты), что повышает качество печати. Разработанные после этого 24-игольчатые принтеры обеспечивают лучшее качество печати, но стоят заметно дороже. Недавно стали выпускаться 18-игольчатые устройства. Диаметр игл, используемых в современных точечно-матричных принтерах, лежит в диапазоне 0,25 — 0,35 мм. В перспективе планируется довести эту величину до 0,02 мм, что существенно повысит разрешающую способность печатающих устройств, которая как раз и ограничивается диаметром иглы. Иногда вместо круглых игл применяются штифты с квадратным сечением. Использование последних позволяет, по данным ряда изготовителей, заметно улучшить качество печати символов, но, по всей видимости, не графики.

В точечно-матричных принтерах применяются устройства подачи красящей ленты кассетного и бобинного типа. Устройство *кассетного* типа характеризуется простотой процедуры заправки кассеты с лентой в принтер. Пользователь даже не касается красящей ленты руками при ее смене, так как извлекает и вставляет целую кассету. В устройствах *бобинного* типа замена ленты сопряжена с определенными трудностями и выполняется вручную.

Различают устройства с обычной и широкой кареткой. Принтеры с *обычной (узкой) кареткой* обеспечивают печать на носителе, имеющем ширину стандартного листа писчей бумаги. *Широкая* же *каретка* позволяет заправлять бумагу с удвоенной шириной. Принтеры обычно способны работать как с рулонной, так и с листовой бумагой. В последнем случае в простых устройствах листы бумаги заправляются вручную. Высококачественные же принтеры обеспечивают автоматическую подачу даже листовой бумаги, что существенно повышает производительность печати. Есть и промежуточные варианты, при которых заправка бумаги автоматизируется лишь частично.

Точечно-матричные принтеры имеют буферное ОЗУ той или иной емкости для того, чтобы разгрузить МП в ходе печати.

Аналогично графическим дисплеям принтеры данного типа могут работать в двух режимах — текстовом и графическом. *Текстовый* режим характеризуется существенно большей скоростью печати, так как при этом выводится выводится сразу строка символов, а не строка точек. В случае текстового режима в принтер пересылаются коды символов, которые следует распечатать, причем матрицы точек, которые нужно нарисовать, выбираются из знакогенератора принтера. При *графическом* режиме в печатающее устройство пересылаются коды, определяющие последовательность и местоположение точек изображения.

Качество печати точечно-матричного принтера определяется его разрешающей способностью, а также возможностями вывода точек с частичным перекрытием (в том числе за несколько проходов печатающей головки). Для текстового режима в общем случае различают следующие подрежимы, характеризующиеся различным качеством печати:

1) *режим черновой печати* (Draft);

2) *режим печати, близкий к типографскому* (NLQ — Near Letter Quality), или *режим делового письма* (Correspondence Quality);

3) *режим с типографским качеством печати* (LQ — Letter Quality);

4) *сверхкачественный режим* (SLQ — Super Letter Quality).

В принтерах с различным числом игл эти режимы реализуются по-разному.

Так, 9-игольчатые устройства обеспечивают печать в режиме Draft за один проход печатающей головки по строке. Режим NLQ реализуется за два прохода: после первого прохода головки бумага протягивается на расстояние, соответствующее половинному размеру точки; затем совершается второй проход (с частичным перекрытием точек). При этом скорость печати уменьшается в два раза. В некоторых принтерах режим двойного прохода совмещается с режимом горизонтального смещения, что еще больше повышает качество печати. Иногда применяется и трехпроходная техника.

18-игольчатые печатающие головки содержат два столбца по 9 игл с вертикальным сдвигом относительно друг друга на половину размера точки. Поэтому режимы печати повышенного качества реализуются за один проход, а скорость черновой печати может быть увеличина в 2 раза за счет одновременной печати двух столбцов точек, вследствие чего можно повысить скорость перемещения пишущей головки.

24-игольчатые принтеры обеспечивают наилучшее качество печати с наивысшей для данного типа устройств скоростью и имеют два ряда по 12 игл с относительным смещением.

Во многих моделях принтеров увеличение скорости печати достигается путем реализации вывода как при прямом, так и при обратном ходе печатающей головки.

Дополнительного повышения качества печати можно добиться, применив печатающие головки с «плавающими» в плоскости бумаги штифтами, но такие устройства не отличаются высокой надежностью.

Точечно-матричные принтеры, как правило, поддерживают несколько шрифтов и их разновидностей, среди которых получили широкое распространение Roman (мелкий шрифт пишущей машинки), italic (курсив), bold-face (полужирный), expanded (растянутый), elite (полусжатый), condenced (сжатый), pica (прямой шрифт — цицеро), Courier (курьер), Sans Serif (рубленый шрифт сансериф), Serif (сериф), Prestige Elite (престиж-элита) и пропорциональный шрифт (ширина поля, отводимого под символ, зависит от ширины символа).

Переключение режимов работы точечно-матричных принтеров и смена шрифтов может осуществляться как программно, так и аппаратно путем нажатия имеющихся на устройствах клавиш или соответствующей установки переключателей.

Принтеры рассматриваемого типа надежны, экономичны, просты в обслуживании, недороги и обладают достаточным быстродействием, приемлемым качеством печати, сравнительно невысоким уровнем шума, а также графическими возможностями. Цена 9-игольчатых устройств колеблется в пределах 150 — 800 долл., а 24-игольчатых — обычно несколько превышает 700 долл. Быстродействие точечно-матричных принтеров лежит в диапазоне 80 — 400 символ/с, но обычно составляет 160 символ/с. Разрешение высококачественных устройств уже сейчас достигло 14,2 точка/мм, но пока они не получили широкого распространения.

Цветная печать реализуется достаточно просто. Каждая строка цветного изображения формируется за четыре прохода печатающей головки с помощью поднятия или опускания кассеты с цветной лентой при каждом проходе, в результате чего иголки ударяют по полосе другого цвета на ленте. Как мы уже знаем, цветной RGB-дисплей для производства различных цветов смешивает три основных цвета: красный (Red), зеленый (Green) и синий (Blue). В отличие от этого цветной принтер выполняет ту же работу с помощью других трех цветов, которые легче смешивать на бумаге: бирюзового, яркокрасного и желтого. Если добавить черный цвет, то получится квартет, который формирует основу всей цветной печати — даже в книгах и в журналах применяется именно такое четырехцветное разделение. Принтер наносит эти цвета на бумагу либо по отдельности, либо один поверх другого. При смешении бирюзовый и яркокрасный цвета воспринимаются как синий, бирюзовый и желтый — как зеленый, а яркокрасный и желтый — как красный цвет. Таким образом образуется RGB-палитра. Перечисленные три комбинации плюс четыре исходных цвета дают уже семь цветов, которые охватывают палитру точечно-матричных принтеров. Более прогрессивные струйные и термографические технологии могут воспроизводить более широкий диапазон оттенков.

Признанным лидером в производстве точечно-матричных принтеров является японская фирма Seiko Epson, хотя в последнее время объем продаж выше у компании Panasonic. Основные технические характеристики базовых моделей изделий фирмы Epson сведены в табл. 2.7. Для сравнения в табл. 2.8 приведены более полные технические характеристики принтеров другого известного производителя — компании Star Micronics (США).

Лучшим в настоящее время 9-игольчатым черно-белым точечно-матричным принтером является изделие Microline 320 (ML-320) фирмы Okidata. Этот принтер — единственное устройство, позволяющее печатать одновременно (под копирку) до пяти экземпляров (остальные принтеры — обычно три). ML-320 обладает наилучшими по сравнению с аналогичными принтерами скоростью и качеством печати (разрешение при работе в графическом режиме составляет 288 точка/дюйм). Доступны три текстовых режима (скоростной черновой, улучшенный черновой и NLQ) и два дополнительных шрифта (Courier и Sans Serif) наряду с Roman. Стоит ML-320 499 долл.

Таблица 2.7

Основные технические характеристики точечно-матричных принтеров фирмы Seiko Epson

Модель	Количество игл	Ширина каретки, дюймов	Быстродействие, символов/с			Интерфейс
			Draft	NLQ	LQ	
Epson LX-850	9	10	200	30	–	Centronics
Epson FX-850	9	10	220	45	–	Centronics
Epson FX-1050	9	15	220	45	–	Centronics
Epson LQ-850	24	10	264	–	88	Centronics, RS232C
Epson LQ-1050	24	15	264	–	88	Centronics, RS232C
Epson LQ-2550	24	15	400	–	133	Centronics, RS232C

Таблица 2.8

Основные технические характеристики точечно-матричных принтеров фирмы Star Micronics

Характеристика	Модель									
	LC-20	LC-15	FR-10	FR-15	ZA-200	ZA-250	XB24-10	XB24-15	XB24-200	XB24-250
Число игл	9						24			
Режимы печати	Draft, NLQ				HS-draft, Draft, NLQ		Draft, LQ, SLQ		HS-draft, Draft, LQ, SLQ	
Скорость печати: Elite, Draft Elite, NLQ (LQ) Pica, Draft Pica, NLQ (LQ)	180 45 150 37		300 78 250 63		Elite, HS-draft: 420 Elite, Draft: 336 Elite, NLQ: 84 Pica, HS-draft: 372 Pica, Draft: 280 Pica, NLQ: 70		240 80 200 67		Elite, Draft: 300 Elite, LQ: 100 Pica, HS-draft: 332 Pica, Draft: 250 Pica, LQ: 83	
Совместимость	ESC/P, IBM Proprinter II				ESC/P, IBM Proprinter III		ESC/P, IBM Proprinter X24/XL24, NEC PC (частично)		ESC/P, IBM Proprinter X24E, NEC (graphic)	
Емкость буфера печати, Кбайт	4	16	31		32		27	41	29	76
Число встроенных шрифтов	4		8		7		13		8	
Ширина каретки, дюймов	10	15	10	15	10	15	10	15	10	15
Интерфейс	Centronics									
Разрешение, точка/дюйм	до 240		до 120		до 240		до 360			
Матрица символов, точек	Draft: 9x12 NLQ: 18x24	Draft: 9x9 NLQ: 18x23	Draft: 9x11 NLQ: 18x23		HS-draft: 9x9 Draft: 9x12 NLQ: 18x24		Draft: 9x24 LQ: 35x24 SLQ: 35x48		HS-draft: 8x24 Draft: 12x24 LQ: 36x24 SLQ: 36x48	
Стоимость, долл.	н/д	536	701	809	н/д	н/д	867	974	н/д	н/д

HS-draft - высокоскоростной режим Draft

Лучшим из существующих цветных точечно-матричных принтеров является Epson LQ-2550 (см. табл. 2.7), стоящий 1499 долл. Он имеет следующие встроенные (стандартные) шрифты: Courier, Prestige Elite, Roman, Sans Serif, Script, OCR-A, OCR-B и Draft. При этом обеспечивается пропорциональная разрядка, разрешение составляет 142 точка/см, а емкость буфера печати равна 8 Кбайт с возможностью расширения до 32 Кбайт. В режиме LQ качество печати достигает значений данного показателя для многих лазерных принтеров.

Фирма Star выпускает цветной принтер Star LC-10 Color, а ее высококачественные устройства индустриального класса FR и XB могут быть снабжены устройством цветной печати.

В заключение приведем характеристики ряда черно-белых точечно-матричных принтеров других изготовителей.

9-игольчатый принтер PR 15 фирмы Ing. C. Olivetti & Co печатает с максимальной скоростью 120 символ/с, имеет 2 уровня качества печати и включает наборы знаков для 13 национальных алфавитов. Ширина каретки составляет 20 см. Модель PR 17 имеет те же характеристики, за исключением ширины каретки в 330 мм.

Принтер PR 19 (снова Olivetti) снабжен той же печатающей головкой, но работает со скоростью 300 символ/с (в режиме NLQ — 150 символ/с).

Принтер PR 1580 той же фирмы с широкой кареткой (330 мм) обеспечивает 7 вариантов качества печати в диапазоне скоростей от 100 до 480 символ/с. Его головка содержит матрицу игл 9x9. Поддерживается печать шрифтами цицеро, элита, микрон и пропорциональным. Максимальное число копий составляет 4. Стоимость этого устройства высока — около 3000 долл.

18-игольчатая модель фирмы Genicom (США) обладает быстродействием 400 символ/с (Draft) и 100 символ/с (LQ). Разрешение в графическом режиме составляет 2,8 или 5,7 (при двойном проходе) точка/мм. Этот принтер стоит 2450 долл.

Особенностью модели 850XL американской фирмы Output Technology является высокая скорость печати — до 850 символ/с.

Отличительная черта 24-игольчатого изделия SL-210AI фирмы Seikosha — большой выбор шрифтов и наличие вместо переключателей кассеты для конфигурирования принтера.

Имеются также построчно-печатающие точечно-матричные принтеры, в которых иглы расположены равномерно вдоль всей строки печати, что существенно повышает быстродействие. Возможны и промежуточные варианты (пример — уже охарактеризованная модель Olivetti PR 1580).

В связи с большим разнообразием точечно-матричных принтеров немаловажной является проблема их совместимости. Принтеры считаются *совместимыми*, если их наборы команд совпадают. В настоящее время стандартными считаются система команд принтеров фирмы Seiko Epson (ESC/P) и система команд принтеров корпорации IBM. Эти системы команд различны. Другие производители обычно обеспечивают совместимость своих изделий одновременно с двумя названными семействами принтеров, чтобы не ограничивать множество потенциальных покупателей. Конкретный режим работы (совместимость с Epson или совместимость с IBM) устанавливается имеющимися на принтерах переключателями. Примерами таких устройств являются изделия фирмы Star Micronics, представленные в табл. 2.8.

Струйные принтеры

Струйная технология впервые была разработана в начале 60-х гг. годов учеными Стенфордского университета (США). Широко внедряться в печатающие устройства она стала только с конца 70-х гг. Первопроходцами в доведении научных разработок до коммерческого использования были фирмы IBM и Siemens AG. Первым удачным струйным принтером явилось устройство ThinkJet, выпущенное фирмой HP. В настоящее время производится множество таких устройств, различающихся как принципом печати, так и техническими характеристиками.

Струйная технология печати, абстрагируясь от деталей, состоит в том, что изображение наносится на бумагу путем «выстреливания» (под давлением) красителя из крохотного *сопла*. Одно или несколько сопел устанавливаются на печатающей головке, которая аналогично точечно-матричным принтерам в процессе работы устройства перемещается относительно бумаги.

Различают два основных типа струйных принтеров:

1) с непрерывной подачей красителя;

2) с капельным микродозатором.

В устройствах *первого типа* формируется непрерывный поток из маленьких капель, которые заряжаются и, пролетая через электрическое поле, отклоняются в вертикальной плоскости пропорционально их заряду. Вспомним, что горизонтальное отклонение обеспечивается перемещением печатающей головки. Капли, которые не должны делать точку на бумаге, отклоняются в специальный желоб, по которому краска возвращается в резервуар для последующего использования. Отклонение капель может быть бинарным, при котором капля попадает либо в определенную точку по вертикали на бумаге, либо в желоб возврата. Такой принцип используется для печатающих головок с несколькими вертикально расположенными соплами. Имеются и устройства с мультиотклонением, используемым при недостаточном количестве сопел, в частности, когда печатающая головка имеет единственное сопло.

Принтеры *второго типа* (с капельным микродозатором) содержат матрицу или столбец вертикально расположенных сопел, и принцип формирования изображений в них аналогичен точечно-матричным печатающим устройствам. При горизонтальном движении печатающей головки из сопел в нужные моменты времени «выстреливаются» капли, которые попадают на бумагу. В этом случае отпадает необходимость отклонять поток капель.

Принтеры с непрерывной подачей красителя, по сравнению с устройствами с капельным микродозатором, имеют большее быстродействие, но и являются более сложными.

Струйным принтерам присущи низкие уровень шума и энергопотребление, графические возможности, вполне доступная стоимость и достаточно высокое качество печати. Малая потребляемая мощность обеспечивает возможность их использования в портативных ПЭВМ с батарейным питанием.

Струйная технология печати порождает и ряд проблем, среди которых основной является проблема предотвращения засыхания чернил в соплах и одновременно с этим обеспечения быстрого их высыхания при попадании на бумагу. Она решается либо путем погружения сопел в резервуар с красителем, либо автоматизацией очистки сопел, либо благодаря использованию красителя, расплавляющегося при нагревании и затвердевающего при остывании. Последний способ решения проблемы представляется наиболее перспективным. Для его реализации достаточно подогреть сопла и, возможно, резервуар с красителем. Иногда используется и нормально твердый краситель в таблетках.

Струйная технология является одним из основных видов получения высококачественной цветной печати. Для цветной печати, как правило, используются красители уже названных четырех цветов. Попарное их смешение до нанесения капель на бумагу дает еще три цвета. Чтобы выйти за семицветное ограничение, струйные принтеры используют прием, известный как *подмешивание*: печать смежных (возможно, с наложением) точек разными цветами, которые глаз воспринимает как одноцветный блок. Однако из-за того, что подмешивание заменяет одну точку определенного цвета несколькими точками разных цветов, изображения, напечатанные методом подмешивания, получаются несколько размытыми.

Охарактеризуем ряд моделей черно-белых струйных принтеров. Уже известный нам малогабаритный принтер с капельным микродозатором ThinkJet фирмы HP обладает быстродействием 40 символ/с, стоит 495 долл. и имеет контейнер с красителем, рассчитанный на 250 млн. символов.

Серьезным конкурентом высококачественным точечно-матричным принтерам является струйное печатающее устройство DeskJet той же фирмы, стоящее 995 долл. В режиме Draft оно печатает со скоростью 240 символ/с, а в режиме LQ — 120 символ/с, что выше аналогичного показателя для большинства точечно-матричных принтеров. Это изделие обладает разрешением 11,8 точка/мм, работает достаточно тихо (уровень шума — 44 дБ) и имеет небольшие габариты (17,3x14,8 дюйма). Печатающая головка DeskJet содержит 50 сопел с насадками, предотвращающими засыхание красителя, и является съемной. Одна головка рассчитана на печать 1100 или 450 страниц текста в режимах Draft и LQ соответственно. Она стоит 18,95 долл. Принтер обеспечивает печать шрифтами Courier, Courier Bold и Courier Compressed. Существуют 12 дополнительных кассет со шрифтами. В каждой кассете записаны по меньшей мере 4 шрифта, различающихся по типу, размеру и стилю. Буфер принтера имеет емкость 16 Кбайт. Поддерживаются как параллельный (Centronics), так и последовательный (RS232C) интерфейсы. Имеется устройство автоматической подачи бумаги формата A4 (шириной 10 дюймов).

Высокоскоростной струйный принтер Digit 1 фирмы Deconix (США) с непрерывной подачей красителя содержит в печатающей головке матрицу из 64 сопел и осуществляет печать графики, а также текста со скоростью 20 страница/мин. (1 страница/мин. примерно равна 50 — 60 символ/с) и разрешением 12 точка/мм. Используется принцип бинарного отклонения. Каждое сопло выбрасывает 75 тыс. капель в секунду. Уникальной является возможность печати на обеих сторонах листа.

Портативный струйный принтер Diconix 150 той же фирмы весит 2 кг при габаритах 50x165x275 мм, стоит 479 долл., печатает со скоростью 150 символ/с, имеет 12 встроенных шрифтов; резервуар с красителем рассчитан на 500 страниц текста.

Недорогой цветной принтер PaintJet фирмы HP поддерживает стандартные шрифты Courier и Letter Gothic, обеспечивает скорость печати 88 символ/с, обладает разрешением 7,1 точка/мм (в семицветном режиме), способен формировать 330 цветов (при разрешении 3,5 точка/мм), содержит буфер емкостью 8 Кбайт и стоит 1395 долл. Автоматическая загрузка бумаги отсутствует, а длина строки составляет 20,3 см.

Высококачественный цветной струйный принтер Pixelmaster фирмы Howtek ценой 7900 долл. способен печатать в 250000 цветах при разрешении 9,5 точка/мм.

Термографические принтеры

Между принципом действия *термографических* и точечно-матричных принтеров можно провести вполне определенную параллель. Отличия состоят лишь в том, что для нанесения точек в первых принтерах используется свойство некоторых материалов изменять свой цвет при нагревании (или расплавляться), а вместо обычных металлических игл применяются тонкие нагреваемые электроды. Таким образом, в термографических принтерах для формирования изображения на бумаге используется не удар, а нагрев. Иногда эти устройства называют *химическими* принтерами, так как в них используется одноименная реакция, вызванная нагреванием.

Термографические печатающие устройства подразделяются на два типа:

1) принтеры с прямым нагревом;
2) принтеры с переносом.

В устройствах *первого* типа используется бумага со специальным химическим покрытием. Нагретый электрод непосредственно касается такой бумаги, и в результате химической реакции точка «проявляется», приобретая синий или черный цвет.

В принтерах *второго* типа используется специальная красящая лента, краситель которой, расплавляясь от касания нагретым электродом, переносится на бумагу, отпечатывая точку.

Достоинство принтеров с передачей состоит в том, что им не требуется специальной бумаги, однако сама красящая лента довольно дорога. Кроме того, при передаче точки через ленту возникает ряд дополнительных технических проблем.

Термографические принтеры почти бесшумны, просты по конструкции, недороги и, хотя обладают малым для большинства моделей быстродействием (40 — 80 символ/с), дают довольно высокое качество печати, естественно, предоставляя и графические возможности. Простота конструкции привела к тому, что устройства этого типа часто используются в портативных ПЭВМ.

Фирма IBM разработала демонстрационный образец высокоскоростного термографического принтера (450 символ/с), но в производство его запустить трудно. Более того, имеются даже термографические страничные принтеры, «выпекающие» сразу целую страницу. Как промежуточные варианты, существуют и построчно печатающие устройства.

Технология цветной термографической печати достаточно проработана, однако независимо от типа устройства (с прямым нагревом или с переносом) она требует нескольких проходов (по одному на каждый основной цвет).

Цветной принтер Bluechip объединения Calcomp фирмы Sanders Associates использует трехцветную слоистую конденсаторную бумагу.

Цветные принтеры Colorscript 100 фирмы QMS и Phaser CP фирмы Tektronix (США) являются постранично печатающими устройствами с переносом. Они обеспечивают разрешение 11,8 точка/мм и используют трех- или четырехпроходную ленту — бирюзово-яркокрасно-желтую (CMY) или бирюзово-яркокрасно-желто-черную (CMYB). Эти устройства предлагают почти неограниченную палитру цветов. Colorscript стоит 15995, а Phaser CP — 12995 долл.

Более дешевый цветной принтер 4693D фирмы Tektronix реализует технологию прямого нагрева вощеной термографической бумаги за четыре прохода (по одному для каждого из трех основных цветов и один проход для черного) и обладает разрешением 12 точка/мм. Предусмотрено также 2 режима монохромной печати: режим с 256 градациями серого цвета и двухцветный черно-белый режим. Цветовая палитра составляет 16 млн. цветов. В стандартной конфигурации это устройство стоит 7995 долл.

Электрофотографические (лазерные) принтеры

В основе большинства *лазерных* принтеров лежит *электрофотографический принцип печати*, заимствованный из ксерографии, где используется свойство фоточувствительных материалов изменять свой поверхностный заряд в зависимости от освещенности.

Пионером в области производства лазерных принтеров является фирма Xerox. Ее изделия были довольно громоздки и стоили несколько десятков, а то и более сотни тысяч долларов. Кроме того, их было не так просто обслуживать. По этим причинам лазерные принтеры не могли использоваться в ПЭВМ.

В 1984 г. фирма Canon USA (США) предложила лазерный принтер LBP-CX, имеющий радикально новую конструкцию. Основное новшество состояло в размещении всего того, что подлежит частой замене, в сменной кассете. Дополнительно к этому была усовершенствована оптика. Стоило данное устройство существенно дешевле, но и имело заметно худшие характеристики по сравнению с изделиями фирмы Xerox. Именно принтер LBP-CX был первым лазерным принтером, доступным для ПЭВМ. Его конструкция легла в основу популярных ныне печатающих устройств LaserJet фирмы HP, LaserWriter компании Apple Computer и 8/300 фирмы Imagen.

Лазерный принтер содержит вращающийся *барабан* (реже — *ленту*), покрытый фоточувствительным (светочувствительным) материалом. В исходном состоянии поверхность барабана электрически нейтральна или имеет электрический заряд, равномерно по ней распределенный (в зависимости от разновидности принтера). В процессе работы устройства при помощи *сканирующего зеркала* осуществляется растровая развертка луча от *лазерного диода* по поверхности барабана. После множества коротких вспышек этого диода, выполняемых в соответствии с выводимым изображением, на барабане засвечиваются все требуемые участки и электрический заряд их изменяется. После засветки на барабан наносится порошок определенного цвета, называемый *тонером*, частицы которого обладают заданным электрическим зарядом. В результате электростатического взаимодействия частицы тонера прилипают к барабану только в тех местах, которые были освещены или не были освещены, что зависит от системы окрашивания (разновидности принтера). Затем рисунок переносится на бумагу путем ее прижатия к барабану и последующего приложения электрического поля. Наконец, тонер фиксируется на бумаге (чаще всего путем термосилового закрепления, состоящего в прокатке разогретым валиком). Иногда фиксация осуществляется вследствие воздействия паров какого-либо растворителя.

Изображение формируется по точкам, однако за счет высокого разрешения лазерными принтерами обеспечивается типографское качество печати текстов и возможность воспроизведения высококачественных рисунков, что позволяет размещать на одной странице как графические изображения, так и текстовую информацию с широким диапазоном размеров букв и множеством различных шрифтов.

Для лазерных, да и ряда других типов принтеров разработаны и используются различные языки описания страниц, среди которых наибольшей известностью пользуется язык PostScript. Он создан несколько лет назад фирмой Adobe Systems. Этот язык может быть реализован как

программно, так и аппаратно оборудованием принтера. Конечно, аппаратная реализация обходится дороже, но и является более эффективной. Фирма HP для своих лазерных принтеров использует собственный язык PCL, также весьма популярный, и одновременно обеспечивает возможность работы на языке PostScript, если установить в устройство специальную кассету.

Лазерные принтеры отличаются высокими быстродействием, разрешающей способностью и соответственно качеством печати, а также великолепными графическими возможностями и низким уровнем шума. Низкоскоростные устройства обеспечивают печать со скоростью 6 — 8 страница/мин., а высокоскоростные — 20 и более страница/мин. В ближайшем будущем планируется довести их быстродействие до 50 страница/мин. Обеспечивается автоматическая подача бумаги. К недостаткам лазерных принтеров следует отнести низкую надежность из-за большой сложности и высокую стоимость.

Для вывода цветного изображения достаточно пропустить через лазерный принтер одну и ту же страницу четыре раза, обеспечив смену тонера, чтобы разные области страницы получили бирюзовый, яркокрасный, желтый и черный цвета.

Наряду с цветными фотокопировальными устройствами, реализующими эту технику, уже появились и цветные лазерные принтеры, но стоят они непомерно дорого. (К примеру, цветное множительное устройство японской корпорации Canon стоит 42000 долл., а принтер обойдется еще дороже).

Лидером в производстве электрофотографических принтеров для ПЭВМ является фирма HP. Ее изделия LaserJet Series II, а теперь и LaserJet III, пользуются заслуженной популярностью из-за удачного сочетания технических характеристик (см. табл. 2.9). Принтер LaserJet IIP (от англ. Personal) весьма компактен и печатает на обычных стандартных листах формата A4. Его габариты составляют 368x457x254 мм, а масса — 10 кг. Устройство LaserJet IIIP при той же массе имеет несколько меньшие габариты — 350x400x200 мм. Остальные модели, оставаясь напольными, имеют большие размеры. Дополнительно к встроенным шрифтам фирма за отдельную плату поставляет широкое разнообразие факультативных шрифтов на кассетах. Так, недавно выпущена кассета с 65 наиболее распространенными шрифтами.

Таблица 2.9

Основные технические характеристики лазерных принтеров фирмы Hewlett-Packard

Модель	Быстродействие, страница/мин.	Разрешение, точка/мм	Стандартная емкость буфера, Кбайт	Число встроенных шрифтов	Интерфейс	Стоимость, долл.
HP LaserJet IIP	4	11,8	512	7	Centronics, RS232C	1495
HP LaserJet II	8	11,8	512	6	Centronics, RS232C	2395
HP LaserJet IID	8	11,8	640	24	Centronics, RS232C	3595
HP LaserJet IIIP	4	11,8	1024	н/д	Centronics, RS232C	1595
HP LaserJet III	8	11,8	1024	36	Centronics, RS232C	2395
HP LaserJet IIID	8	11,8	н/д	н/д	Centronics, RS232C	3595
HP LaserJet IIISi	16	11,8	2048	н/д	Centronics, RS232C	5495

н/д - нет данных

В острую конкурентную борьбу с HP вступила фирма IBM, предложив электрофотографический принтер Laserprinter 4019, который в сравнении с LaserJet II на 40% меньше по размерам, имеет быстродействие 10 страница/мин., обладает той же стоимостью, на треть дешевле в эксплуатации, имеет стандартно ту же емкость буфера с возможностью расширения до 4 Мбайт и 10 встроенных шрифтов.

Японская фирма Sharp предлагает лазерный принтер JX-9500E, габариты которого примерно на треть меньше, чем у HP LaserJet IIP. JX-9500E стоит всего 1340 долл., а печатает со скоростью 6 страница/мин. Остальные технические характеристики совпадают с LaserJet IIP.

Уже упомянутый принтер LBP-CX фирмы Canon USA имел характеристики, аналогичные устройству LaserJet II. Его сменная кассета содержит сухой тонер, устройство проявления, барабан, блоки электризации и др. элементы. Оптика в нем никогда не заменяется.

Новый принтер LBP-20 той же фирмы обладает быстродействием 20 страница/мин., имеет разрешение 19,2 точка/мм и позволяет печатать на обеих сторонах листов бумаги (как, впрочем, и LaserJet IIISi).

Некоторые лазерные принтеры имеют разрешающую способность много ниже 11,8 точка/мм и стоят значительно меньше 1000 долл. Другие же, разрешение которых достигает 96,5 точка/мм, имеют цену порядка 80000 долл.

Электростатические принтеры

Технология *электростатической* печати является близкой родственницей электрофотографии и разработана сотрудниками фирмы Delphax Systems.

Вместо источника света и сложной оптики с подвижными частями для переноса изображения на барабан в электростатических принтерах используется *принцип ионного осаждения* (электронная печать). Он реализуется за счет того, что над *барабаном* устанавливается *управляющий электрод*, а между ними — *сменная кассета для ионного осаждения*. Барабан и кассета, в свою очередь, разделены *экранирующим электродом* с отверстиями, который воздействует на ионы в качестве удерживающего и фокусирующего элемента. При приложении к барабану и управляющему электроду напряжения между ними возникает коронный разряд, в результате чего ионы, «хранящиеся» в кассете, ускоряются и переносятся через экранирующий электрод на барабан. Потенциал же экранирующего электрода управляет засветкой барабана в соответствии с выводимым изображением. Далее процесс печати повторяет технологию, реализованную в лазерном принтере.

Из-за отсутствия подвижных деталей электростатические принтеры обладают большей надежностью и долговечностью.

В настоящее время фирма Delphax Systems предлагает два варианта электростатических принтеров:

1) устройство 2460;

2) устройство S6000, имеющее быстродействие 60 страница/мин.

В среднем электростатические принтеры обладают быстродействием 20 — 40 страница/мин. (выше, чем у лазерных) и есть резервы его увеличения до 200 — 300 страница/мин. Их стоимость колеблется в пределах 15 — 48 тыс. долл.

Электрочувствительные принтеры

В *электрочувствительном* печатающем устройстве изображение формируется в результате протекания тока по поверхности специальной бумаги. В наиболее распространенной конструкции используется *бумага с цветным покрытием*, поверх которой наносится тонкая *алюминиевая пленка*, придающая листу бумаги белый цвет. Печать производится аналогично точечно-матричным принтерам с помощью ряда *игл*, к которым приложено напряжение. При касании иглами алюминиевой пленки по ней протекает ток и локально испаряет ее участки. Через образующиеся отверстия в пленке становится видна подложка (покрытие бумаги, обычно темного цвета), за счет чего и «проявляется» изображение. Существуют как принтеры последовательного действия, так и построчно печатающие устройства данного типа.

Благодаря малым размерам электрочувствительные устройства могут встраиваться в дисплеи и использоваться в портативных ПЭВМ.

Магнитографические принтеры

Магнитография в какой-то мере аналогична электрофотографии и электростатике, но в ней используется *магнитная запись*. Барабан имеет магнитное покрытие, а над ним располагаются магнитные головки, которые записывают на этот барабан «невидимое» изображение. Тонер обладает ферромагнитными и термопластическими свойствами. После намагничивания барабана тонер переносится на него, «прилипая» к определенным его областям. Проявленное таким образом изображение закрепляется на бумаге путем теплового сплавления.

Уникальность данной технологии в том, что она позволяет воспроизводить копии одного и того же изображения без его регенерации на барабане.

Настольная модель 800 магнитографического принтера фирмы Ferix (США) содержит матрицу головок 8x16, имеет быстродействие 10 — 14 страница/мин., обладает разрешением 9,5 точка/мм и стоит 2000 — 3000 долл.

Фирма Cynthia Peripheral выпускает устройства с тем же разрешением, но быстродействием, равным 50 и 90 страница/мин.

Наряду с рассмотренными типами принтеров имеются и другие, в частности, *гибридные*. Так, например, фирма Brother International предлагает устройство с двумя печатающими головками: одна — лепесткового типа, а другая — точечно-матричная. Это обеспечивает типографское качество печати наряду с графическими возможностями. В некоторых разновидностях принтеров для нанесения изображения используются перья, что ставит такие устройства в промежуточное положение между типичными принтерами и графопостроителями.

Дорогие высокопроизводительные принтеры целесообразно использовать совместно несколькими пользователями. Это можно реализовать одним из следующих способов:

— простым переключением принтера между ПЭВМ;

— одновременным подсоединением единственного принтера к нескольким ПЭВМ через интеллектуальный многопортовый буфер;

— путем объединения ПЭВМ в простую сеть на базе интерфейса RS232C;
— посредством комплектования ПЭВМ в более сложную сеть типа Ethernet;
— в результате переноса ГД с данными на другую ПЭВМ и печати содержащейся на нем информации.

2.6.3. Графопостроители

Графопостроитель, или **плоттер** (от англ. plotter), — это устройство вывода, представляющее выводимые из ЭВМ данные в форме рисунка или графика на бумаге или другом подобном ей носителе информации. Графопостроители являются высококачественной альтернативой принтерам при выводе изображений. Графопостроители — это дополнительные ПУ, которые используются при автоматизированном проектировании, в частности, с применением ПЭВМ.

В зависимости от конструктивного исполнения плоттеры делятся на устройства планшетного и рулонного (барабанного) типа.

В графопостроителе *планшетного* типа лист бумаги закрепляется на *рабочей плоскости*, над которой движется *перо* (или несколько перьев), перемещающееся по двум координатам x и y. Пишущий узел приводится в движение *сервоприводом*, обеспечивающим высокую точность его установки. Точности позиционирования пера, как правило, достаточно, например, для вычерчивания в реальном масштабе рисунков сложных печатных плат. Роль перьев в простых моделях плоттеров играют обычные шариковые ручки, а в совершенных моделях — специальные тонкопишущие фломастеры. Изображение формируется на листе бумаги при перемещении опущенного пера. Если нужно просто установить перо в другую исходную точку, то оно автоматически приподнимается и переводится в требуемое место. В ПЭВМ применяются главным образом устройства в настольном исполнении и намного реже — в напольном.

Графопостроители *рулонного* типа отличаются большей компактностью и могут использоваться при выводе на рулонную бумагу. В них бумажный лист протягивается *транспортирующим валиком* для обеспечения вертикальных перемещений; перо же может перемещаться только в горизонтальной плоскости.

В графопостроителе могут использоваться одно или несколько перьев. Применение нескольких перьев обеспечивает вычерчивание цветных изображений или повышение скорости вывода. Дополнительные цветовые оттенки можно получить путем смешения на бумаге основных цветов. Наиболее совершенные устройства имеют отдельный привод для каждого пера. Другие же просто обеспечивают автоматическую смену перьев.

Плоттеры также могут применяться и для печати текстовой информации, однако при этом они работают очень медленно, так как последовательно вырисовывают контуры символов, а при выдаче текстов шрифтами большого размера дополнительное время требуется для закрашивания символов. Нередко встречаются и гибридные устройства, сочетающие возможности как принтеров, так и графопостроителей, поскольку печатающие устройства и плоттеры функционально и конструктивно близки.

Из-за низкого быстродействия плоттеры, как правило, достаточно подключать к адаптеру последовательного интерфейса.

Основными техническими характеристиками графопостроителей являются:
1) размер чертежной поверхности (формат листа бумаги);
2) быстродействие;
3) разрешающая способность;
4) точность установки пера;
5) количество перьев, что определяет число воспроизводимых цветов;
6) масса, габариты и энергопотребление;
7) стоимость.

Быстродействие плоттера измеряется скоростью вычерчивания линии. Под *разрешением* понимают минимальное расстояние между двумя адресуемыми точками. По разрешающей способности графопостроитель может в несколько раз превосходить даже лазерный принтер. *Точность установки пера* определяет абсолютную погрешность, с которой перо может быть перемещено к точке с указанными координатами. Стоимость графопостроителей лежит в пределах 1500 — 20000 долл.

Для иллюстрации возможностей плоттеров в табл. 2.10 сведены характеристики различных устройств, выпускаемых фирмой Houston Instrument. Эта же фирма за 2100 долл. предлагает принтер-плоттер JetPro 360, обладающий аналогичными лазерным устройствам графическими возможностями и типографским качеством печати, размером чертежной поверхности A2 — A4, разрешением 0,07 мм (14,2 точка/мм), а также интерфейсами RS232C и Centronics.

Высокими техническими характеристиками обладают плоттеры семейства Draftmaster фирмы HP. Они снабжены 12 разноцветными перьями, имеют быстродействие 1100 мм/с и стоят до 17000 долл. Вообще плоттеры HP считаются стандартными, особенно изделие HP 7475A.

Принтер же FP 6302 фирмы Graphtec, имея 24 пера, способен одновременно воспроизводить 24 цвета.

Таблица 2.10

Основные технические характеристики графопостроителей фирмы Houston Instrument

Модель	Формат	Быстродействие, мм/с	Разрешение, мм	Точность, мм	Количество перьев	Стоимость, долл.	Интерфейс
HI-1117	A3-A4	400	0,025	0,05	8	1540	RS232C
DMP-60MP	A3-A4	800	0,0127	0,05	6	нет данных	RS232C
DMP-61	A1-A4	800	0,0127	0,05	1	5600	RS232C
DMP-61MP	A1-A4	800	0,0127	0,05	6	6440	RS232C
DMP-61DL	A1-A4	800	0,0127	0,05	8	6860	RS232C
DMP-62	A0-A4	800	0,0127	0,05	1	7350	RS232C
DMP-62MP	A0-A4	800	0,0127	0,05	6	8190	RS232C
DMP-62DL	A0-A4	800	0,0127	0,05	8	8890	RS232C

2.6.4. Синтезаторы звука

Простыми встроенными *синтезаторами звука* снабжаются, как правило, все ПЭВМ, они используются для выдачи компьютером звукового сигнала при возникновении какого-либо события. Синтезаторы звука управляются программными средствами. Различными производителями факультативно предлагается в виде ПУ широкий спектр синтезаторов звука — вплоть до синтезаторов речи.

Примитивные синтезаторы являются по сути просто электронными генераторами звука, способными выдавать однотональный сигнал определенной частоты (она обычно лежит в пределах 800 — 1000 Гц).

В более совершенных устройствах имеется возможность программирования частоты выдаваемого звукового сигнала. Например, в IBM-совместимых ПЭВМ используется генератор с частотой, изменяемой от 37 до 32767 Гц, что позволяет исполнять на машине музыкальные произведения. Но этот генератор является *одноголосным*.

Дальнейшее развитие синтезаторов звука привело к появлению *многоголосных* устройств, имеющих несколько (обычно 3 или 4) независимых каналов программирования тональных сигналов, способных функционировать одновременно. Это позволяет имитировать звучание сложных музыкальных инструментов и даже оркестров. Такими возможностями, к примеру, обладают ПЭВМ фирмы Atari.

Для профессиональных музыкантов выпускаются АРМ с расширенными возможностями синтеза и программирования звуковых сигналов. Клавиатура ПЭВМ в этом случае превращается в клавиатуру музыкального инструмента.

Венцом звуковых синтезаторов являются синтезаторы речи. Выводить информацию в речевой форме намного проще, чем вводить. Выпускаются и гибридные устройства, осуществляющие как ввод, так и вывод речи.

2.7. Общие сведения о системе прерываний

Понятие прерывания мы уже ввели в подразделе 1.3.

Первоначально *системы прерываний* ЭВМ обеспечивали главным образом асинхронную работу процессора и ПУ, чтобы отказаться от непрерывного опроса процессором периферийного оборудования. Однако концепция прерываний оказалась полезной и для вызова системных подпрограмм, так как в этом случае легко изменить или заменить подпрограмму вследствие того, что обращение к ней производится не по адресу, а по номеру прерывания.

В IBM-совместимых ПЭВМ прерывания делятся на следующие категории:

1) *внешние аппаратные прерывания*, возникающие при событиях вне МП;

2) *внутренние аппаратные прерывания*, вырабатываемые самим МП;

3) *программные прерывания*, инициируемые выполняемой программой, как правило, по специальной команде.

Сигналы *внешних аппаратных прерываний* вырабатываются периферийными и другими устройствами, выставляются на системной шине и перед передачей в МП предварительно обрабатываются специальным устройством — *контроллером прерываний*. В качестве примеров ситуаций, вызывающих прерывания данной категории, можно назвать нажатие клавиши на клавиатуре, обнаружение ошибки памяти, сбой питания, сигнал от таймера. Внешние аппаратные прерывания делятся на *маскируемые* и *немаскируемые*. Первые могут быть заблокированы, а вторые — нет. Немаскируемые прерывания возникают в случае событий с катастрофическими последствиями.

Примерами событий, приводящих к *внутренним аппаратным прерываниям*, являются попытка деления на нуль, управление пошаговым режимом работы МП (для отладки программ), достижение контрольной точки, переполнение разрядной сетки.

Программные прерывания задаются программистом в программе путем записи специальной команды и используются для получения сервисных услуг ОС.

IBM-совместимые ПЭВМ поддерживают 256 типов прерываний, определяемых источником и причиной прерывания. Для их распознавания и разделения во времени каждому типу прерывания присваивается определенный *код* (от 0 до 255) и *приоритет (уровень)*. Число уровней может варьироваться от 4 до 15. Одновременное поступление в МП сигналов прерываний одного уровня обычно не допускается. Поэтому прерывания от ПУ, которые могут функционировать одновременно, должны иметь различный уровень.

Для обработки прерываний в начальных ячейках ОЗУ располагается *таблица векторов прерываний* размером 254x4=1024 байта. Каждое четырехбайтовое поле этой таблицы называется **вектором прерывания** и содержит адрес обработчика прерываний в виде номера первого параграфа в сегменте (старшее слово) и относительного адреса (младшее слово). Каждый вектор прерывания однозначно адресуется по сигналу прерывания путем сдвига его кода на два разряда влево, что эквивалентно умножению на 4.

Обработка прерывания состоит в следующем:

1) формируется его код, который поступает в МП;

2) в стеке сохраняется текущее состояние МП (содержимое регистров CS и IP) с целью возможного последующего восстановления;

3) по соответствующему вектору прерывания осуществляется переход на программу обработки прерывания данного типа;

4) выполняется собственно обработка прерывания;

5) восстанавливается исходное состояние МП, вследствие чего либо продолжается выполнение прерванной программы, либо завершается ее выполнение, либо происходит останов МП (в зависимости от типа прерывания).

2.8. Средства объединения ПЭВМ

ПЭВМ могут использоваться в четырех режимах (четырьмя способами):

1) как автономные вычислительные установки;

2) в качестве интеллектуальных терминалов больших и миниЭВМ;

3) в составе сетей ЭВМ;

4) в качестве пассивных терминалов многопользовательских систем.

Наиболее простым и доступным является *автономный режим*, когда пользователь работает на ПЭВМ независимо от кого-либо и решает свои задачи без использования посторонних (дополнительных) вычислительных ресурсов. В этом случае пользователь «варится в своем собственном соку», а обмен информацией с другими машинами осуществляется только с помощью съемных (сменных) внешних носителей информации. Такой режим использования ПЭВМ отвечает отнюдь не всем приложениям, а только наиболее простым и не являющимся ресурсоемкими.

Второй вариант использования ПЭВМ состоит в подключении последних к большим или миниЭВМ *в качестве интеллектуальных терминалов*. В этом случае на ПЭВМ возлагается реализация функций взаимодействия с пользователем, первичной обработки вводимых данных и окончательной подготовки результатов решения задач к выдаче пользователю. Центральная же ЭВМ будет выполнять остальные ресурсоемкие задачи, в частности требующие большого объема вычислений или большого объема внешней памяти. Использование ПЭВМ в качестве интеллектуальных терминалов в последнее время стало очень популярным, что объясняется тремя факторами:

1) дружественностью интерфейса ПЭВМ с пользователем;

2) возможностью разгрузки центральной ЭВМ от выполнения вспомогательных действий;

3) низкой стоимостью ПЭВМ, сравнимой с ценой обычных терминалов.

К одной центральной ЭВМ, естественно, целесообразно подключить несколько интеллектуальных терминалов.

Наиболее перспективным для многих приложений является использование ПЭВМ *в составе вычислительной сети*. При этом несколько ПЭВМ, а возможно, и ЭВМ других классов, соединяются вместе посредством каналов связи и аппаратуры сопряжения с ними для обмена информацией. Объединение ПЭВМ в сеть служит достижению следующих целей:

1) повышению эффективности использования дорогостоящих и уникальных аппаратных, программных и информационных ресурсов (путем их разделения между различными пользователями);

2) обеспечению взаимодействия различных пользователей сети (электронная почта).

Так, например, пользователь сети может получить доступ к любому изданию крупной библиотеки вместо того, чтобы хранить ее на своей ПЭВМ или реально ходить в библиотеку.

Вычислительные сети делятся на глобальные и локальные.

Каналами связи в *глобальных* сетях служат в основном телефонные линии связи. Так как по телефонным линиям могут передаваться только сигналы с частотой звукового диапазона, необходимо обеспечить преобразование цифровых сигналов (логические уровни 0 и 1) в сигналы звуковой частоты. Преобразование цифровой информации в аналоговую форму производится специальным устройством, называемым *модулятором*. Обратное преобразование, которое производится при приеме на другом конце линии, т.е. восстановление информации в ее первоначальной цифровой форме, осуществляется другим устройством, называемым *демодулятором*.

Поскольку при двусторонней связи нужны оба вида преобразования, то для их выполнения используются устройства, сочетающие в себе функции модулятора и демодулятора, для краткости именуемые *модемами*.

Локальная вычислительная сеть (ЛВС) связывает ряд ЭВМ, в частности ПЭВМ, находящихся в одной локальной зоне, ограниченной, например, одним или несколькими рядом расположенными зданиями, каким-либо радиусом или одной организацией. Существенное отличие ЛВС от глобальной сети состоит в том, что при передаче информации не требуется преобразование из цифровой формы ее представления в аналоговую и наоборот, т.е. информация между ПЭВМ передается непосредственно в цифровом виде по кабелям. Длина кабелей может достигать нескольких километров. Линии связи ЛВС обладают гораздо большей пропускной способностью по сравнению с телефонными линиями. ПЭВМ подключаются к сети обычно через адаптер последовательного интерфейса RS232C. ЛВС могут подсоединяться к глобальной сети через машины-шлюзы.

Целесообразность использования ЛВС объясняется тем, что персональный помощник человека, работающего в коллективе, должен входить в «коллектив» помощников. В данном случае немаловажное значение имеет электронная почта. Совместное использование ресурсов сохраняет свое значение. Например, одному пользователю слишком дорого обойдется наличие в ПЭВМ НЖМД емкостью 1 Гбайт, высококачественного лазерного принтера и высокоточного графопостроителя. Коллектив же пользователей может и должен позволить себе такую роскошь, обеспечив доступ к этим ресурсам каждого своего члена.

ЛВС на базе ПЭВМ выгодно отличаются от сетей мини- и больших ЭВМ по ряду технико-экономических показателей, таких, как стоимость, простота внедрения и эксплуатации и др. Это объясняется, в первую очередь, тем, что ПЭВМ по своим возможностям начинают превосходить мини- и даже большие ЭВМ, а по стоимости они на один — два порядка дешевле последних.

К числу важнейших характеристик ЛВС относятся:

1) *топология сети*;

2) *количество ПЭВМ*, способных работать в сети;

3) максимально допустимое *удаление ПЭВМ* друг от друга;

4) *типы ПЭВМ*, входящих в сеть (в зависимости от этого различают однородные и неоднородные ЛВС);

5) *используемое ПО*;

6) *надежность ЛВС*, определяемая ее способностью сохранять работоспособность при выходе из строя отдельных ПЭВМ и линий связи, что во многом зависит от топологии сети и ПО;

7) *передающая среда*, под которой понимают физическую среду, обеспечивающую передачу информации с помощью электрических или световых сигналов;

8) *метод доступа*, представляющий собой по сути дела совокупность принципов функционирования ЛВС, выбор которого определяется топологией сети;

9) *протокол*, являющийся совокупностью правил, регламентирующих формат и процедуры обмена информацией между узлами сети.

Топология определяет структурную организацию связей между узлами ЛВС (ПЭВМ). Различают шинную (магистральную), звездообразную, древовидную, кольцевую и полную (многосвязную) топологию (см. рис. 2.13). Возможны и смешанные (гибридные) варианты. В ЛВС ПЭВМ обычно используется кольцевая или шинная топология.

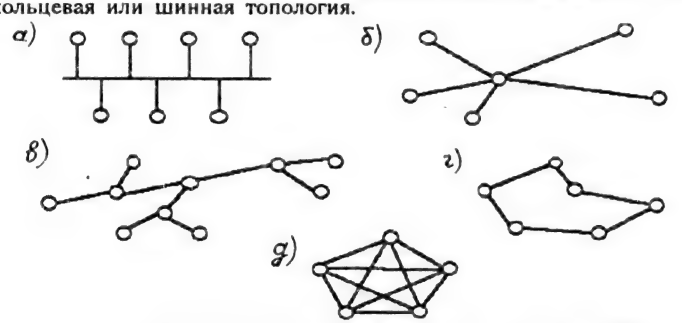


Рис. 2.13. Основные топологические структуры ЛВС:
а — шинная (магистральная); б — звездообразная;
в — древовидная; г — кольцевая; д — полная (многосвязная)

В качестве примера охарактеризуем наиболее совершенную локальную сеть Ethernet, имеющую шинную топологию. Спецификация этой сети разработана совместно фирмами DEC, Intel и Xerox. В ней обеспечивается скорость передачи данных 10 Мбит/с. В качестве линий связи применен коаксиальный кабель с волновым сопротивлением 50 Ом. Он обычно прокладывается в желобах или фальшпотолках отрезками длиной не более 500 м. Одна сеть Ethernet может содержать любое число таких отрезков, соединенных повторителями, при условии, что между двумя ПЭВМ существует только один соединяющий их путь и этот путь содержит не более двух повторителей (не более 1500 м кабеля).

Другими известными ЛВС ПЭВМ с шинной топологией являются:
— Cluster/One фирмы Nestar Systems (США) — одна из первых ЛВС ПЭВМ;
— Omninet фирмы Corvus Systems — одна из наиболее популярных в настоящее время ЛВС ПЭВМ;
— PC NET фирмы IBM;
— Cheapernet — упрощенный вариант сети Ethernet.

Применяются также следующие ЛВС ПЭВМ со звездообразной топологией:
— PLAN 4000 фирмы Nestar Systems, позволяющая объединять ПЭВМ различных типов и обеспечивающая выход на глобальные сети и большие ЭВМ;
— Star Lan фирмы AT&T Information Systems (США).

Наиболее типичным представителем ЛВС ПЭВМ с кольцевой топологией является сеть Token-Ring Network фирмы IBM.

Каждый узел ЛВС ПЭВМ является рабочей станцией, в качестве которой выступает персональный компьютер. Он должен иметь свою ОС, а также копию сетевой ОС. Каждая из ПЭВМ, объединенных в сеть, способна принимать участие в обработке информации. Обычно наиболее мощная машина в сети играет роль *файл-сервера*, храня все используемое многими пользователями ПО на жестком диске в единственном экземпляре. Каждый из пользователей может загрузить на выполнение в свою рабочую станцию любой из программных продуктов, записанных на файл-сервере.

Вообще **сервером** называют узел вычислительной сети, обычно локальной, в котором обеспечивается обслуживание сетевых терминалов (рабочих станций) путем управления распределением дорогостоящего ресурса совместного пользования. Для *файл-сервера* роль такого ресурса играет внешняя память и хранимая в ней информация.

В отличие от ЛВС *многопользовательская система* представляет собой единственную (обычно мощную) ПЭВМ и несколько пассивных терминалов, способных только вводить и выводить информацию. Конечно, роль таких терминалов могут играть и ПЭВМ, но тогда их ресурсы будут использоваться неэффективно. Вся обработка в многопользовательской системе осуществляется на единственной машине.

Предпочтение многопользовательской системе можно отдать в тех случаях, когда:
— экономия денежных средств является важным фактором;
— общее число пользователей невелико и в последующем меняться не будет;
— требуется только централизованная обработка данных.

3. ОСНОВНЫЕ МОДЕЛИ ПЕРСОНАЛЬНЫХ ЭВМ

В предыдущем разделе мы рассмотрели различные устройства ПЭВМ (классифицировали их, определили и сравнили основные технические характеристики).

Здесь же ПЭВМ будет представлена как единое целое. Этот раздел может служить в качестве справочника по существующим моделям персональных машин.

Вначале будут определены основные понятия, такие, как производительность и совместимость ПЭВМ, а также перечислены наиболее важные технические характеристики персональных компьютеров с точки зрения пользователя. Затем читатель ознакомится с различными классами ПЭВМ, выпускаемыми ведущими фирмами, и с персональными машинами, производимыми в восточноевропейских странах. Раздел завершается описанием основных тенденций развития ПЭВМ.

3.1. Слагаемые производительности ПЭВМ

Производительность — важнейший показатель качества ЭВМ, но, к сожалению, его затруднительно определить строго. При сравнении различных ПЭВМ по производительности зачастую можно установить, какая из них и примерно насколько лучше по данному показателю. Иногда же (если машины совершенно различны) провести сравнительную оценку их производительности можно лишь по набору тестов, которые все равно не дадут исчерпывающей объективной картины. Абсолютной же оценки производительности ПЭВМ, равно как и других типов ЭВМ, не существует. Можно получить только ряд косвенных оценок, в той или иной степени ее характеризующих.

Производительность ПЭВМ определяется:

1) производительностью МП;
2) наличием состояний ожидания МП при обращении к ОП и ПУ;
3) быстродействием и типом ОЗУ;
4) быстродействием ПУ;
5) эффективностью ОС.

Поскольку основную роль в обработке информации компьютером играет МП, он оказывает наибольшее влияние на производительность ПЭВМ. *Производительность МП* определяют пять факторов:

1) тактовая частота;
2) пропускная способность (разрядность) шины данных;
3) набор команд;
4) эффективность микрокода;
5) параллелизм при выполнении последовательности команд.

Рассмотрим подробнее перечисленные факторы, несмотря на то, что они в той или иной мере уже упоминались в п. 2.1.2.

МП — приборы последовательного действия. Для соблюдения очередности действий используются *синхронизирующие импульсы*, называемые **тактовыми**. За время каждого такта по очередному тактовому импульсу МП реализует одну элементарную операцию *(микрооперацию)*. Для выполнения команды или какой-либо другой функции требуется несколько тактов. Чем выше тактовая частота, тем быстрее при прочих равных условиях МП выполняет микрооперации, а следовательно, и команды в целом.

Каждый МП имеет максимальную расчетную тактовую частоту. Когда тактовая частота повышается сверх нее, то гарантия правильной работы МП отсутствует. На слегка завышенной тактовой частоте МП обычно сохраняют работоспособность. Это обстоятельство используется некоторыми производителями ПЭВМ, выжимающими из МП «все соки».

Влияние *пропускной способности шины данных* на производительность ПЭВМ очевидна. Фирма Intel, например, утверждает, что использование прибора 8086 вместо 8088 ведет к увеличению производительности в среднем на 40% при работе на той же частоте.

Команды — это элементарные «кирпичики», из которых складывается программа. От того, какие именно команды имеются в *системе команд* МП, зависит время выполнения программы, а следовательно, и его производительность. Например, если в набор команд МП входит команда умножения, которая позволяет найти произведение двух 16-разрядных чисел, скажем, за 20 тактов, то это дает программисту возможность не писать выполняющую умножение подпрограмму, которая выполнялась бы за сотни тактов. Однако усложнение системы команд имеет и отрицательные последствия, в связи с чем существует альтернативный подход (см. п. 2,1.5).

МП имеют *микропрограммное управление*, т.е. каждая команда реализуется *микропрограммой*, хранящейся в ПЗУ. *Эффективность микрокода* определяется числом микрокоманд в микропрограмме той или иной команды, или числом тактов, за которое команда выполняется. При разработке новой модели МП изготовитель обычно оптимизирует микрокод, по крайней мере, для некоторых, наиболее употребимых команд. Так, например, упоминавшиеся ранее изделия серии V фирмы NEC Information Systems имеют более эффективный микрокод, чем соответствующие им МП фирмы Intel, что обеспечивает бóльшую производительность первых на одной и той же тактовой частоте.

Развитые МП имеют *конвейерную архитектуру*, а следовательно, последовательность команд может выполняться параллельно. Это дополнительно повышает их производительность.

Роль *состояний ожидания* рассмотрим на примере МП 8088.

Обычно прибору 8088 требуется четыре такта для выполнения операций чтения из памяти или записи в нее. Следовательно, цикл доступа к памяти составляет 4 такта. Доступ к ПУ осуществляется аналогично.

Если цикл доступа к памяти или ПУ превышает время, соответствующее четырем тактам, по причине недостаточного их быстродействия, то МП один или несколько тактов будет простаивать. Таким образом, состояние ожидания — это дополнительный «холостой» такт, необходимый для организации доступа к памяти или ПУ. Состояния ожидания позволяют продлить цикл обращения настолько, насколько это необходимо. В ПЭВМ IBM PC, например, автоматически добавляется одно состояние ожидания для доступа к любому ПУ.

Состояния ожидания могут значительно снизить производительность ПЭВМ. Для исключения состояний ожидания в машинах с высокоскоростными МП в настоящее время используется быстродействующая кэш-память.

Быстродействие ОП как раз и влияет на производительность ПЭВМ через состояния ожидания. Тип ОЗУ также оказывает воздействие на этот показатель. Действительно, DRAM, как уже отмечалось в подразделе 2.2, требует *регенерации* своего содержимого. Для этого в ПЭВМ IBM PC и PC XT периодически осуществляется цикл прямого доступа к памяти. Эта операция отвлекает МП от полезной работы и тем самым снижает производительность компьютера примерно на 8%. Аналогичная ситуация наблюдается и в IBM PC AT, хотя регенерация там осуществляется иначе. Дополнительно к проблеме инициирования регенерации неизбежны конфликты при попытке МП осуществить доступ к ОЗУ во время цикла восстановления его содержимого. В этом случае ПЭВМ вынуждена добавлять состояния ожидания для завершения регенерации памяти, прежде чем открыть доступ к ней со стороны МП. Это дает уровень дополнительных потерь порядка 5 — 10%. ОЗУ, выполненное на SRAM, полностью снимает описанные проблемы, а кэш-память — только частично.

Быстродействие ПУ влияет на производительность ПЭВМ естественным образом. Так, например, для дисководов (НМД) важны среднее время доступа и скорость чтения/записи. Аналогична ситуация и для принтера. С видеосистемой имеется дополнительная проблема, определяемая конфликтами, возникающими при одновременном обращении видеоблока и МП к видеопамяти. Первый должен иметь приоритет перед вторым. Следовательно, во время кадровой развертки (во время вывода изображения на экран) МП не сможет ничего записать в видеопамять и поэтому вынужден будет ждать. Запись в видеопамять возможна только во время обратного хода луча — промежутка времени, в течение которого луч электронной пушки дисплея переводится в левый верхний угол экрана. Это обстоятельство, естественно, снижает производительность ПЭВМ. В этом отношении наихудшим является адаптер CGA, а лучшим по сравнению с ним и EGA — VGA.

Для согласования быстродействия МП и ПУ в наиболее совершенных моделях ПЭВМ используют кэши (буферы), встроенные в соответствующие адаптеры или в сами устройства. Так, например, некоторые адаптеры НЖМД снабжены кэш-памятью; практически все принтеры имеют встроенный буфер; клавиатура также оборудована небольшим буфером.

ОС, являясь связующим звеном между пользователем, программами и оборудованием ПЭВМ, требует для своей деятельности ресурсов компьютера, в том числе времени МП. Следовательно, от *эффективности* использования ресурсов ПЭВМ операционной системой зависят накладные расходы, связанные с ее функционированием. Они уменьшают долю полезной работы ПЭВМ в общем объеме вычислений и поэтому отрицательно влияют на ее производительность. Кроме ухудшения производительности ПЭВМ, ОС обеспечивает и противоположный эффект в результате создания в ОЗУ кэшей для обмена информацией с ПУ, в частности, НМД. Важное значение при этом приобретает рациональная установка параметров ОС.

Многозадачные ОС способны существенно повысить производительность ПЭВМ на основе распараллеливания работ.

Как уже отмечалось, существующие методики оценки производительности ПЭВМ базируются на использовании программных тестов. Один из ведущих в области ПЭВМ американский журнал BYTE, например, использует следующие две группы *тестовых индексов (показателей) производительности*:

1) низкоуровневые тестовые индексы;
2) прикладные тестовые индексы.

Индексы *первой* группы в относительных единицах характеризуют быстродействие основных устройств ПЭВМ, среди которых МП, сопроцессор плавающей точки (если он имеется), НЖМД и дисплейный адаптер.

Индексы *второй* группы, также в относительных единицах, характеризуют производительность ПЭВМ в целом при решении задач, различающихся соотношением использования ресурсов (основных устройств) ПЭВМ. При этом выделяют следующие типы задач:

— обработка текстов;
— обработка табличной информации;
— обслуживание баз данных;
— научные и инженерные расчеты;
— разработка программного обеспечения (компиляция и компоновка).

Путем суммирования значений прикладных тестовых индексов определяют *обобщенный индекс производительности*. Таким образом, чем больше значение этого индекса, тем выше производительность данной ПЭВМ. Именно он и является конечным показателем производительности (в соответствии с методикой журнала BYTE).

3.2. Понятие совместимости ПЭВМ

Большое количество моделей ПЭВМ различных классов и различных производителей определяет интерес пользователей к проблеме их совместимости.

Термин *«совместимость»* в области вычислительной техники имеет, как минимум, три различных толкования — аппаратная совместимость, программная совместимость ЭВМ и совместимость программ.

Под *аппаратной совместимостью* понимают способность одного устройства логически заменять другое устройство того же типа или способность одного устройства как физически, так и логически сопрягаться с другими. В последнем случае в качестве синонимов аппаратной совместимости используются также термины *«полная (аппаратная) совместимость»* и *«совместимость по разъемам»*.

Под *программной совместимостью* одной ЭВМ с другой понимают способность первой выполнять программы, которые были разработаны для второй ЭВМ. Различные модели одного и того же семейства ЭВМ имеют, как правило, *«одностороннюю»* совместимость, поскольку компьютеры более поздних (старших) моделей обычно являются более мощными (т.е. способны исполнять дополнительные команды, имеют больший объем памяти и т.д.). В этом случае говорят, что ЭВМ старшей модели *совместима снизу вверх* с ЭВМ младшей модели, подчеркивая тот факт, что первая может выполнять программы, подготовленные для второй, но не наоборот.

Под *совместимостью программ* (аналогично аппаратной совместимости) понимают либо способность одной программы воспроизводить поведение другой программы (т.е. полностью ее заменять), либо способность одной программы взаимодействовать с другими программами, в частности, путем передачи данных в определенном формате.

Несколько обособленно, но ближе к аппаратной, стоит *совместимость ЭВМ в сети*.

Более подробно мы остановимся только на программной совместимости ПЭВМ, так как, с одной стороны, она может обеспечиваться различными способами и быть полной или неполной, а с другой стороны, эти вопросы важны при выборе ПЭВМ, чтобы приобрести компьютер, способный выполнять требуемые системные и прикладные программы.

Строго говоря, для программной совместимости требуется совместимость как по информации, так и по исполнению.

Под **совместимостью по информации** будем понимать способность переноса информации, т.е. программ и данных, с одной машины на другую. Для этого обе ПЭВМ должны включать однотипные ВЗУ со съемными носителями информации. Например, если ПЭВМ различаются лишь тем, что одна из них укомплектована 133-мм, а другая — 89-мм НГМД, причем иные ВЗУ со сменными носителями отсутствуют, то эти машины уже несовместимы по информации. Проблему совместимости данного вида несложно решить, дополнительно укомплектовав компьютер соответствующим ПУ.

Более фундаментальна **совместимость по исполнению**, т.е. способность ЭВМ выполнять уже перенесенные на нее программы, разработанные для другой ЭВМ.

Зачастую программную совместимость отождествляют с совместимостью по исполнению, так как совместимость по информации, как уже отмечено, обеспечивается достаточно просто. Кроме того, производители ПО поставляют свою продукцию на различных носителях информации по выбору покупателя.

Для совместимости по исполнению требуется либо неразличимость устройств ЭВМ (как центральных, так и периферийных) с точки зрения выполняемых программ, либо программная эмуляция (имитация) одних устройств другими, если это возможно.

В первом случае говорят о *полной программной совместимости* ПЭВМ, хотя они могут быть и не идентичны на аппаратном уровне.

Во втором случае — о различных *уровнях совместимости* ПЭВМ. Среди них выделим два основных уровня: совместимость на уровне ОС и совместимость на уровне базовой системы ввода-вывода (BIOS — Basic Input-Output System).

Совместимость на уровне ОС определяет способность программ, разработанных в среде данной ОС, выполняться на любой ПЭВМ, на которой эта ОС установлена. При этом роль эмулятора выполняет сама ОС, написанная отдельно для каждого типа ПЭВМ. Данный уровень совместимости наиболее слабый, так как, во-первых, нельзя выполнить на другой машине саму ОС без ее «переделки», а во-вторых, обход в программах услуг ОС делает эти программы невыполнимыми на других типах ПЭВМ, даже работающих под управлением той же ОС. Различия же в ПУ зачастую создают дополнительные проблемы, если в программе не предусмотрена адаптация к различной периферии.

Две ПЭВМ проблематично совместить на уровне ОС, если они не поддерживают одно и то же ядро системы команд.

ПЭВМ обычно поставляются с прошитой в ПЗУ BIOS, т.е. с программой, обеспечивающей непосредственное управление ПУ и скрывающей их аппаратные особенности. *Совместимость на уровне BIOS* обеспечивает выполнение программ, не использующих непосредственное управление ПУ. Этот тип совместимости занимает промежуточное положение между совместимостью на уровне ОС и полной программной совместимостью ПЭВМ, не поддерживая выполнение на другой ПЭВМ тех программ, которые действуют в обход BIOS. Остаются также проблемы, связанные с различными возможностями периферии. Когда говорят о совместимости без конкретизации уровня, имеется в виду именно совместимость на уровне BIOS, которая распространена наиболее широко.

Между двумя рассмотренными уровнями совместимости по исполнению могут быть всевозможные промежуточные варианты.

При оценке совместимости ПЭВМ, в первую очередь, обращают внимание на тип МП, в нее установленного. Если в двух ПЭВМ МП одинаковы или совместимы снизу вверх, то в большинстве случаев эти машины окажутся совместимыми. Конечно, ПЭВМ с совершенно различными МП можно сделать совместимыми путем программной эмуляции одного МП другим (иногда говорят «на другом»), однако реально так не поступают из-за больших накладных расходов.

Затем следует выяснить, какая СШ использована в ПЭВМ. Однако различия в СШ не столь серьезны, как различия МП, и могут быть устранены на уровне BIOS.

Таким образом, МП, СШ и BIOS являются тем водоразделом, который определяет совместимость ПЭВМ по исполнению.

Немаловажным является также тип дисплейного адаптера. Так, если программа жестко привязана к EGA, то она не будет выполняться на ПЭВМ с CGA. Обратный же переход возможен, так как EGA имитирует все режимы работы CGA.

При выборе ПЭВМ следует обратить внимание и на другую периферию, в частности, на тип НГМД.

3.3. Перечень технических характеристик ПЭВМ

К важнейшим техническим характеристикам ПЭВМ относятся:

1) модель МП, его тактовая частота и число состояний ожидания при обращении к памяти;
2) наличие средств поддержки сопроцессора плавающей точки определенной модели или самого сопроцессора;
3) стандартная емкость ОЗУ и возможности его расширения;
4) наличие кэш-памяти;
5) характеристики основных ПУ — дисплея (дисплейного адаптера), НМД и клавиатуры;
6) наличие и характеристики дополнительных ПУ;
7) наличие и число адаптеров интерфейсов;
8) наличие и число гнезд расширения, служащих для подключения дополнительных устройств;
9) стоимость.

Этот перечень характеристик охватывает важнейшие показатели качества стационарных ПЭВМ. Для портативных машин немаловажны также следующие характеристики:

1) габариты;
2) масса;
3) тип дисплея;
4) потребляемая мощность;
5) возможность автономного питания;
6) максимальная продолжительность автономной работы, если она допустима.

Что касается возможностей расширения ОЗУ, то они могут обеспечиваться двумя способами:

— поставкой ПЭВМ с увеличенной емкостью ОЗУ на основной (системной) плате, но возможность этого ограниченна;

— использованием дополнительных плат, подсоединенных к соответствующим гнездам расширения, если они имеются.

Мы будем указывать максимальную емкость ОЗУ с учетом двух этих возможностей.

Иногда быстродействие памяти на дополнительной плате оказывается меньшим, чем быстродействие ОЗУ на системной плате. Этот фактор следует учитывать.

Вероятно, Вам интересно узнать, в каких странах производятся наиболее качественные, а в каких — самые дешевые ПЭВМ.

В начале «списка качества» (по мере снижения качества) фигурируют следующие страны:
1) Германия;
2) Япония;
3) США и Тайвань;
4) Южная Корея;
5) Сингапур;
6) Гонконг.

Если Вы ищете самые дешевые машины или хотите, чтобы Вас не обвели вокруг пальца, то ознакомьтесь со «списком цен» (по мере роста стоимости):
1) Гонконг и Сингапур;
2) Южная Корея;
3) Тайвань;
4) США;
5) Япония;
6) Германия.

Поэтому, например, при условии одинаковых технических характеристик и цен ПЭВМ лучше приобрести сингапурскую машину вместо гонконговской. Если же Вам предложат на выбор американскую и тайваньскую машину с одинаковыми техническими характеристиками, то лучше приобрести вторую (конечно, за меньшую плату).

3.4. ПЭВМ фирмы IBM

В настоящее время весь рынок ПЭВМ делится на следующие три сектора:
1) машины фирмы IBM и их аналоги;
2) изделия фирмы Apple Computer;
3) ПЭВМ других типов независимых производителей.

Наиболее крупным является первый из перечисленных секторов, а за ним следует второй. Третий же сектор наименее представителен. Указанное соотношение наиболее ярко проявляется в области ППЭВМ. Изменить эту пропорцию достаточно сложно из-за большого объема накопленного ПО для IBM-совместимых ПЭВМ. Большинство выпускаемых в СНГ ПЭВМ совместимы с изделиями фирмы IBM.

По отмеченным причинам имеет смысл подробно охарактеризовать компьютеры, выпускаемые корпорацией IBM, и лишь после этого рассмотреть другие типы ПЭВМ (как совместимые, так и несовместимые с ними).

Фирма IBM остается крупнейшим производителем ПЭВМ, несмотря на наличие множества других компаний, предлагающих аналогичные и даже более совершенные изделия. Причины такого положения в том, что изделия корпорации IBM отличаются высокой надежностью и зачастую агрессивно низкими ценами. Кроме того, IBM обеспечивает высокое качество технического обслуживания в сжатые сроки. Наконец, делает свое дело и репутация фирмы. Многое из того, что поддерживается или предлагается IBM, впоследствии обретает роль стандарта.

Корпорация IBM специализируется на производстве стационарных ППЭВМ. Она предпринимала попытки освоить рынок других типов ПЭВМ как по функциональному назначению, так и по конструктивному исполнению, но все они оказались неудачными. Возможно, новые модели семейства PS/2 изменят существующую расстановку сил.

Схематично генеалогия ПЭВМ корпорации IBM рассматривалась в подразделе 1.2. Полные же сведения представлены на рис. 3.1. На этом рисунке для каждой модели в скобках указана дата ее анонсирования (объявления). Дуги же иллюстрируют программную совместимость моделей по исполнению. Если модель A соединена дугой с моделью B, то это означает совместимость модели B с моделью A снизу вверх. Возможна и двухсторонняя совместимость, в том числе, на различных уровнях. Этот факт отмечается направленностью дуги в обе стороны. Отношение совместимости является транзитивным: если модель A совместима с моделью B, а модель B совместима с моделью C, то и модель A совместима с моделью C.

Изделия корпорации IBM характеризуются открытостью и модульностью архитектуры и делятся на два *семейства*:
1) PC (Personal Computer — персональный компьютер);
2) PS/2 (Personal System 2 — персональная система 2).

Название первого семейства неудачно, так как под PC можно понимать и просто ПЭВМ.

Основные технические характеристики базовых представителей семейства PC в порядке увеличения вычислительной мощности приведены в табл. 3.1.

В этой и всех последующих таблицах использованы следующие обозначения:

— за типом МП через тире указана тактовая частота в МГц; если же допускается ее переключение, то возможные значения отделены наклонной чертой;

— две точки (..) разделяют минимальное и максимальное значения для данной характеристики, указывая тем самым диапазон значений;

— запятая разграничивает альтернативные значения (за ней в отличие от разделителя целой и дробной частей числа должен следовать пробел);

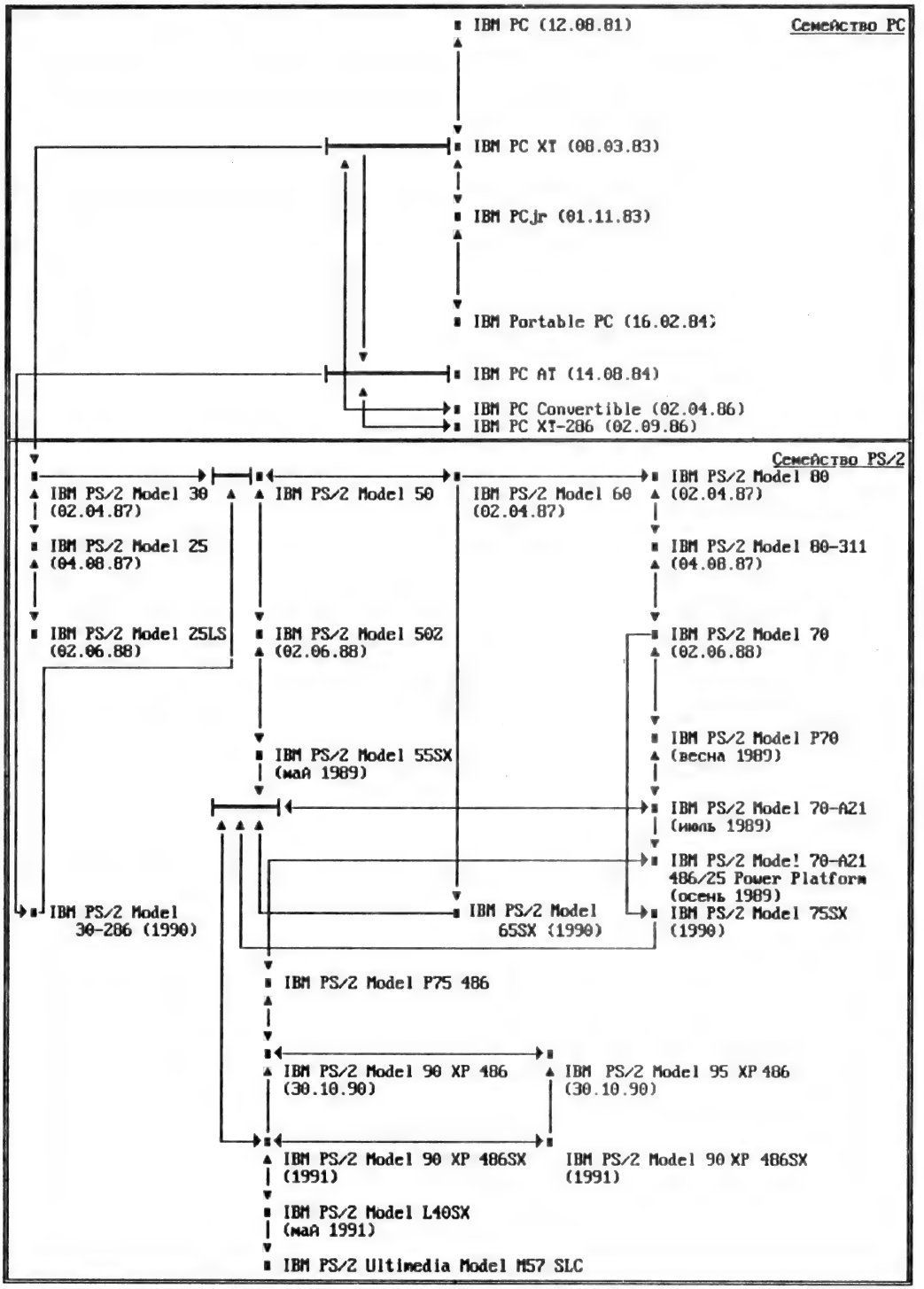


Рис. 3.1. Генеалогия ПЭВМ фирмы IBM

Таблица 3.1

Технические характеристики ПЭВМ семейства PC IBM

Модель	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Количество НГМД х емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт	Дисплейный адаптер	АИ	Количество гнезд расширения	ОС
IBM PC	8088-4,77	8087	64K..544K	2x160K-133	-	MDA	1s&1p	5_8	PC DOS, CP/M-86
IBM PC XT	8088-4,77	8087	256K..640K	1x360K-133	10[20]	CGA	1s&1p	8_8	PC DOS, CP/M-86
IBM PC XT-286	80296-6	80287	640K	1x360K-133	20	CGA	1s&1p	5_{16}&3_8	PC DOS
IBM PC AT	80286-8	80287	512K..10,5M	1x1,2M-133	20[30,40]	EGA[CGA]	1s&1p	6_{16}&2_8	PC DOS, Xenix, OS/2

— если одно из альтернативных значений является стандартным, то оно указано первым, а остальные перечислены в квадратных скобках;

— в качестве характеристик НГМД проставлено их количество, а также емкость и диаметр диска;

— в качестве характеристик НЖМД, если имеются соответствующие сведения, указаны емкость и среднее время доступа;

— знак & (амперсанд) интерпретируется как «И»;

— «м» и «ц» указывают на наличие монохромного и цветного дисплея соответственно;

— буквы s, p, m как подстрочные индексы чисел в столбце АИ обозначают адаптеры последовательного (RS232C), параллельного (Centronics) интерфейса и интерфейса для «мыши» соответственно; например, 1s&1p говорит о наличии одного адаптера последовательного и одного адаптера параллельного интерфейсов;

— подстрочные числовые индексы указывают разрядность гнезд расширения, если она нам известна;

— н/д является сокращением от «нет данных».

Все представленные модели семейства PC имеют настольное исполнение.

Родоначальником семейства PC IBM является модель IBM PC. В настоящее время она не пользуется сколь-нибудь заметным спросом из-за отсутствия контроллера НЖМД. Современные ее модификации снабжаются НГМД для дисков диаметром 133 мм и емкостью 360 Кбайт. Возможно также расширение ОЗУ до 640 Кбайт. Клавиатура IBM PC по нынешним меркам недостаточно удобна и имеет 84 клавиши. В настоящее время IBM PC стоит менее 1000 долл.

Основной отличительной особенностью ПЭВМ IBM PC XT является наличие встроенного НЖМД. В ней применен также адаптер CGA и имеется больше гнезд для расширения ресурсов. ПЭВМ может оборудоваться как монохромным, так и цветным монитором. К моменту появления IBM PC XT фирма Corona Data Systems уже выпускала аналогичную машину с НЖМД емкостью 10 Мбайт, так что пальма первенства в использовании в ПЭВМ дисковода типа «винчестер» корпорации IBM не принадлежит. В настоящее время IBM PC XT продолжает пользоваться определенным спросом и стоит около 1200 долл.

Попытка корпорации IBM выйти на рынок учебных ПЭВМ с моделью IBM PCjr на МП 8088 окончилась полной неудачей, и в апреле 1985 г. выпуск этой машины был прекращен. В данном секторе рынка традиционно лидирует фирма Apple Computer.

Последовав примеру компании Compaq, разработавшей первую переносную ПЭВМ, корпорация IBM предложила модель IBM Portable PC (МП 8088). Однако и это изделие постигла неудача.

Собравшись с силами, IBM опередила своих конкурентов по внедрению в ПЭВМ МП 80286, разработав модель IBM PC AT. Эта машина уже допускает возможность многозадачной и (в принципе) многопользовательской работы; обладает более высокой производительностью, а также более емкими НГМД и НЖМД; использует новый дисплейный адаптер (EGA); оборудована 84-клавишной клавиатурой, вскоре замененной на 101-клавишный пульт, и встроенными часами-календарем с автономным питанием. 101-клавишные клавиатуры широко используются в современных моделях ПЭВМ. В настоящее время IBM PC AT наиболее распространена, но спрос на нее начинает падать. Стоимость этой модели составляет 1,5 — 2 тыс. долл.

Модель IBM PC Convertible, использующая МП 8088, существенным спросом не пользовалась.

Первоначально машина IBM PC AT с трудом завоевывала рынок. Поэтому была предложена модель IBM PC XT-286 на МП 80286, но с характеристиками, близкими к XT. Это, конечно, был шаг назад, сделанный, видимо, с целью оживления спроса на IBM PC AT.

Все модели PC IBM имеют одно состояние ожидания МП.

В настоящее время фирма IBM делает ставку на другое свое семейство ПЭВМ — PS/2. Отличительными особенностями первых представителей этого семейства являются:

— переход на новую шину (MCA) для большинства моделей;

— возможность использования НОД наряду с НМЛ для резервирования информации;

— введение трех новых графических стандартов (MCGA, VGA и 8514/A);

— использование (начиная с 50-й модели) 15-ти каналов прямого доступа к памяти, четыре из которых могут работать одновременно;

— встраивание в системную плату адаптеров интерфейсов, ранее размещавшихся на отдельных платах расширения;

— использование нового (89-мм) формата ГД;

— улучшенный дизайн.

Шина MCA и 89-мм НГМД создают трудности по обеспечению программной совместимости моделей PS/2 с моделями PC. Совместимость по исполнению реализуется благодаря наличию в ПЗУ моделей семейства PS/2 двух BIOS, одна из которых обеспечивает совместимость с семейством PC на уровне BIOS, а другая наиболее полно использует возможности шины MCA. Совместимость по информации может быть достигнута путем подключения к моделям PS/2 внешних 133-мм НГМД. Несмотря на наличие в моделях 25 — 30 PS/2 шины AT двухсторонняя совместимость между ними и соответствующими представителями семейства PC не обеспечивается из-за различных дисплейных адаптеров.

Основные технические характеристики большинства моделей семейства PS/2 IBM в порядке возрастания их номеров сведены в табл. 3.2. Каждая модель может иметь несколько модификаций, в зависимости от значений некоторых параметров. Приведенные цены весьма условны и не включают стоимость дисплеев, принтеров и других не указанных в таблице ПУ.

Модели 25 и 25LS отражают еще одну попытку корпорации IBM внедриться в сферу учебных ПЭВМ. В модели 25LS используется адаптер ЛВС Token Ring.

Модель 30, популярная в 1987 — 88 гг., имеет производительность в 2 — 2,5 раза бóльшую, чем исходная IBM PC XT.

Модель 30-286 отличается от модели 30 типом МП и близка к модели 50, за исключением того, что имеет шину AT. Выпуск модели 30-286 обусловлен тем, что шина MCA не стала стандартной и не известно, будет ли она таковой. Появление этой модели выражает в определенной степени «отступление» корпорации IBM.

Модель L40SX является первой наколенной ПЭВМ в семействе PS/2. Эта машина снабжена жидкокристаллическим дисплеем и весит 3,49 кг. Возможно автономное питание.

Модели 50 и 60 — это MCA-аналоги машины IBM PC AT, имеющие вдвое более высокую производительность. Модель 50 подвергалась критике за низкое быстродействие. Поэтому выпущена модель 50Z, устраняющая этот недостаток, в том числе путем ликвидации состояний ожидания МП (Z — от англ. Zero — нуль).

Модель 55SX представляет собой недорогую машину на МП80386SX, являющуюся переходной между моделями 50, 60 и моделями 70, 80. Модели 65SX и 75SX аналогичны.

Модели 70 и 80 выполнены на базе МП 80386, причем первая первоначально была настольным аналогом второй и обеспечивала производительность, в 3,5 раза бóльшую, чем AT. Но затем была выпущена более мощная 25-МГц модификация модели 70 (A21) с кэшем. Она отличается малыми габаритами, и единственный ощутимый ее недостаток состоит в малом количестве гнезд расширения. Потом IBM предложила плату 486/25 Power Platform, превращающую модель 70-A21 в ПЭВМ с МП 80486. Производительность машины повысилась при этом в два раза.

Модель P70 является портативным вариантом модели 70 и снабжена VGA-совместимым монохромным плазменным дисплеем с 16 градациями яркости, а также полной клавиатурой. Возможно подключение внешнего VGA-дисплея. Эта машина весит 9,5 кг и имеет габариты 465x305x127 мм. Автономное питание не обеспечивается. Выпуск модели P70 ознаменовал третью попытку корпорации IBM освоить рынок компактных ПЭВМ.

Позже появилась еще одна переносная машина — модель P75 486 с таким же дисплеем, массой 10 кг и размерами 300x184x63мм, что позволило ей свободно помещаться в портфеле. Обратите внимание на то, что она в отличие от своей предшественницы имеет новый процессор.

Модели 90 и 95 можно считать принадлежащими новому поколению ПЭВМ в семействе PS/2, причем это связано не только с применением МП 80486 и видеосистемы XGA.

Все варианты моделей 90 и 95 могут быть снабжены вторым НГМД, в качестве которого допускается установка как 89-мм 1,44-Мбайт, так и 133-мм 1,2-Мбайт накопителя. Вместе с тем 133-мм дисковод — это не шаг назад. Просто в ПЭВМ данного типа есть место для монтирования НОД, которые пока не поставляются. 1,2-Мбайт НГМД имеет улучшенные характеристики и высоту, равную 1/3 от стандартной.

Модель Ultimedia Model M57 SLC, выполненная на базе МП80386, — это первая ПЭВМ, поддерживающая комплексное представление информации (multimedia).

Кроме моделей, указанных на рис. 3.1 и в табл. 3.2, фирмой IBM выпускается «японоязычная» модель 55 с жидкокристаллическим дисплеем и объявлены (в конце 1989 г.) модели 120, 130 и 135. О последних моделях достоверной информации у нас нет. Лишь известно, что модель 120 будет представлять собой многопроцессорную ЭВМ на МП 80486 и иметь ОЗУ емкостью 48 — 60 Мбайт, а также до восьми терминалов. Поэтому к классу ПЭВМ ее отнести уже трудно.

Таблица 3.2

Технические характеристики ПЭВМ семейства PS/2 IBM

Модель (IBM PS/2 Model ...)	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Емкость кэш-памяти, Кбайт	Шина	Количество НГМД x емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер-дисплей	АИ	Количество гнезд расширения	ОС	Цена, долл.	Конструктивное исполнение
1	2	3	4	5	6	7	8	9	10	11	12	13	14
25-001 25-004 25-G01 25-G04	8086-8	8087	512K 512K 640K 640K	–	AT	1x720K-89 [2x720-89]	–	MCGA-м MCGA-ц MCGA-м MCGA-ц	1s&1p&1m	2_8	PC DOS	969 н/д	настольное
25LS-L01 25LS-L04	8086-8	8087	640K	–	AT	1x720K-89	20:80	MCGA-м MCGA-ц	1s&1p&1m	2_8	PC DOS	н/д	настольное
30-001 30-021	8086-8	8087	640K	–	AT	2x720K-89 1x720K-89	– 20:80	MCGA	1s&1p&1m	3_8	PC DOS	1084 1549	настольное
30-286-E01 30-286-E21 30-286-E31	80286-10	80287	1M..4M	–	AT	2x1,44M-89 1x1,44M-89 1x1,44M-89	– 20:80 30:39	VGA	1s&1p&1m	3_{16}	PC DOS, OS/2, Xenix	1412 н/д н/д	настольное
L40SX	80386SX-20	80387SX	2M..16M	–	MCA	1x1,44M-89	60	VGA-м	н/д	н/д	PC DOS, OS/2	5995	наколенное
50-021	80286-10	80287	1M..7M	–	MCA	1x1,44M-89	20:80	VGA[8514/A]	1s&1p&1m	3_{16}	PC DOS, OS/2	2449	настольное
50Z-031 50Z-061	80286-10	80287	1M..16M	–	MCA	1x1,44M-89	30:39 60:27	VGA[8514/A]	1s&1p&1m	3_{16}	PC DOS, OS/2	2376 2719	настольное
60-041 60-071	80286-10	80287	1M..15M	–	MCA	1x1,44M-89	44:40 70:30 [115:28]	VGA[8514/A]	1s&1p&1m	7_{16}	PC DOS, OS/2	3279 3495	напольное (стойка)
55SX-031 55SX-061	80386SX-16	80387SX	2M..16M	–	MCA	1x1,44M-89	30:39 60:27	VGA[8514/A]	1s&1p&1m	3_{32}	PC DOS, OS/2	3895 4238	настольное
70-E61 70-121 70-A21 70-A21 486/25 Power Platform	80386-16 80386-20 80386-25 80486-25	80387 80387 80387 –	1M..16M 1M..16M 2M..16M 2M..16M	– – 64 64	MCA	1x1,44M-89	60:27 120:27 120:27 120:27	VGA[8514/A]	1s&1p&1m	1_{16}&2_{32} 1_{16}&2_{32} 3_{32} 3_{32}	PC DOS, OS/2, AIX	5995 7995 11295 12990	настольное
P70-061 P70-121	80386-20	80387	4M..8M	–	MCA	1x1,44M-89	60:27 120:27	VGA	н/д	1_{16}&1_{32}	PC DOS, OS/2, AIX	7695 9400	переносное

Таблица 3.2 (окончание)

1	2	3	4	5	6	7	8	9	10	11	12	13	14
P75 486	80486-33	-	8M..16M	-	MCA	1x1,44M-89	160 [400]	VGA-и	н/д	н/д	PC DOS, OS/2	15990	переносное
80-041 80-071 80-111 80-311	80386-16 80386-16 80386-20 80386-20	80387	1M..22M 2M..22M 2M..22M 2M..22M	-	MCA	1x1,44M-89	44:40 60:27 120:27 314[600]	VGA[8514/A]	1sälpälm	7	PC DOS, OS/2, AIX	4374 5069 7099 8264	напольное (стойка)
90 XP 486- 0J5 0J9 0KD	80486-25 80486-25 80486-33	-	4M..32M 4M..32M 4M..32M	[256]	MCA	1x1,44M-89	80:17 160:16 320:12,5	XGA	н/д	н/д	PC DOS, OS/2	12495 13095 16695	настольное
90 XP 486SX	80486SX-20	н/д	4M..32M	8 [512]	MCA	1x1,44M-89	80:17	XGA	н/д	н/д	PC DOS, OS/2	8345	настольное
95 XP 486- 0J9 0JD 0KD	80486-25 80486-25 80486-33	-	4M..32M 4M..32M 4M..32M	[256]	MCA	1x1,44M-89	160:16 320:12,5 320:12,5	XGA	н/д	н/д	PC DOS, OS/2	14145 16095 17745	настольное (башня)
95 XP 486SX	80486SX-20	н/д	4M..32M 4M..32M	8 [512]	MCA	1x1,44M-89	160:16	XGA	н/д	н/д	PC DOS, OS/2	н/д	настольное (башня)

С 1986 г. IBM выпускает рабочую станцию PC RT на RISC-МП. Для данной машины разработана UNIX-подобная ОС AIX, использующаяся и в старших моделях PS/2. В связи с тем, что IBM PC RT не является ПЭВМ, мы ее рассматривать не будем.

3.5. IBM-совместимые ПЭВМ

IBM-совместимой будем называть ПЭВМ, удовлетворяющую следующим двум условиям:

1) она программно совместима по исполнению (по крайней мере снизу вверх) с одной из моделей корпорации IBM;

2) уровень этой совместимости не выше BIOS.

Таким образом, если ПЭВМ работает под управлением той же ОС, что и изделие IBM, то это еще не означает ее IBM-совместимости.

Определяющими факторами для IBM-совместимости являются:

1) использование МП фирмы Intel или его аналога;

2) использование или эмуляция посредством BIOS шины, совместимой с одной из двух шин, применяемых фирмой IBM.

В зависимости от используемого МП различают следующие *классы* IBM-совместимых ПЭВМ:

1) XT-совместимые машины, т.е. ПЭВМ, совместимые с IBM PC XT или IBM PC (МП 8088/86);

2) AT-совместимые компьютеры (МП 80286 или 80386SX);

3) ПЭВМ на базе МП 80386;

4) ПЭВМ на базе МП 80486 (в том числе МП 80486SX).

Поскольку модели последних двух классов могут быть совместимы с различными изделиями IBM, в названии классов фигурируют просто типы МП.

Представители перечисленных классов выпускаются в различных конструктивных исполнениях. Мы сочли более целесообразным разделить IBM-совместимые ПЭВМ на *группы*, в зависимости от конструктивного исполнения, и рассмотреть различные классы машин в рамках каждой группы. Это вызвано прежде всего тем, что от конструктивного исполнения зависит степень важности тех или иных характеристик.

Для обеспечения конкурентоспособности своей продукции производители IBM-совместимых ПЭВМ ставят перед собой одну или несколько из следующих целей:

1) улучшение технических характеристик и/или расширение функциональных возможностей ПЭВМ;

2) снижение стоимости;

3) увеличение гарантийного срока эксплуатации;

4) повышение качества технического обслуживания.

Перейдем теперь к рассмотрению конкретных моделей IBM-совместимых ПЭВМ.

3.5.1. Стационарные ПЭВМ

В группе стационарных ПЭВМ имеются представители всех четырех классов IBM-совместимых машин. Лидерами в производстве стационарных IBM-совместимых ПЭВМ являются следующие фирмы:

— Compaq Computer, выпускающая дорогие и высококачественные ПЭВМ экстракласса;

— Tandy, в последнее время несколько сдавшая свои позиции;

— AST Research, набирающая силу;

— ZEOS International, предлагающая высококачественные ПЭВМ по относительно низким ценам;

— Dell Computer, изделия которой также пользуются хорошей репутацией и имеют относительно невысокую стоимость;

— Advanced Logic Research, изготавливающая ПЭВМ с высокими техническими характеристиками.

Имеются и другие фирмы, предлагающие великолепные изделия в том или ином классе IBM-совместимости.

Крупными производителями дешевых и достаточно хороших IBM-совместимых ПЭВМ являются тайваньские и южнокорейские фирмы, слабо представленные в настоящем обзоре.

Здесь мы проведем последовательное рассмотрение всех четырех классов IBM-совместимых ПЭВМ в порядке увеличения вычислительной мощности. В представленных ниже таблицах модели отсортированы в алфавитном порядке фирм изготовителей, а для каждой фирмы — в порядке увеличения вычислительной мощности машин.

XT-совместимые ПЭВМ

XT-совместимые машины в настоящее время дослуживают свой век. Основные технические характеристики наиболее известных представителей данного класса приведены в табл. 3.3. Дополнительно к этому отметим следующее:

— многие модели имеют гнездо для подключения сопроцессора 8087, а ПЭВМ Global/XT стандартно поставляется с ним;

Таблица 3.3

Технические характеристики стационарных XT-совместимых ПЭВМ

Производитель: фирма, страна	Модель	МП	Емкость ОЗУ, байт	Количество НГМД x емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт	Дисплейный адаптер-дисплей
Amstrad, Великобритания	Amstrad PC 1640	8088-8/4,77	640K	2x360K-133	20[32]	CGA-ц
BEST Computer, США	BEST VTI-33	8088-10	640K	2x360K-133	н/д	CGA-м
Global Computer Network, США	Global/XT	8088-12/8	640K	1x360K-133, 1x720K-89	20	EGA-ц
Ing. C. Olivetti & Co, Италия	Olivetti M24	8006-8	128K..640K	1x360K-133 [1x720K-133]	10[20]	EGA-м, EGA-ц
Jameko Electronisc, США	JE 3002 JE 3003	8088-8/4,77 8088-8/4,77	256K..640K 640K	по выбору по выбору	по выбору по выбору	CGA CGA
OWL Computer Services, США	OWL Turbo	V20, 8088-8, 10, 12	до 640K	по выбору	30	CGA-м
PAO-KU International, США	MYODA MD2000	8088-10/4,77	256K..640K	1x360K-133	по выбору	по выбору
Swan Technologies, США	Swan XT/10	8088-10/4,77	640K	по выбору	по выбору	по выбору
Tandy, США	Tandy 1000 Tandy 1200	8088-4,77 8088-4,77	до 640K до 640K	2x360K-133 1x360K-133	- 10[20]	MDA CGA
The Ultimate Business Machines, США	Club Model 110	8088-10/4,77	256K..640K	по выбору	20, 40	по выбору
Zenith Data Systems, США	Zenith Z-150 Zenith Easy PC Zenith Z-158	8088-4,77 8088-4,77 8088-8/4,77	до 640K 512K 256K..640K	1x360K-133 1x720K-89 [2x720K-89] 1x360K-133 [2x360K-133]	10[20] -[20] -[10]	CGA CGA CGA

— ряд моделей поставляется с усовершенствованной 101-клавишной клавиатурой вместо 84-клавишной;

— большинство перечисленных моделей имеет по одному адаптеру последовательного и параллельного интерфейсов;

— большинство представленных изделий имеет несколько 8-разрядных гнезд расширения;

— цена XT-совместимых ПЭВМ, в зависимости от комплектации, колеблется в пределах от нескольких сот до 2 тыс. долл. (в последнем случае имеется в виду полный комплект с винчестером, дисплеем, принтером, «мышью» и сопроцессором);

— подавляющее большинство моделей имеет обычное настольное исполнение, однако MYODA MD2000 выполнена в виде малогабаритной стойки, размещающейся на столе (tower — башня);

— у всех моделей шина аналогична шине IBM PC XT.

В XT-совместимых ПЭВМ достигнута тактовая частота 12 МГц; иногда используется адаптер EGA и 720-Кбайт НГМД. Емкость НЖМД невелика.

ПЭВМ класса XT способны работать только в *однопользовательском однозадачном* режиме и предназначены для приложений, не требующих большой вычислительной мощности. Основной ОС для них является MS-DOS (PC DOS); иногда применяется и CP/M-86.

AT–совместимые ПЭВМ

В момент написания этой книги AT-совместимые ПЭВМ пользовались наибольшим спросом. Модели данного класса строятся на базе МП 80286 или 80386SX. Использование последнего имеет преимущества для 32-разрядных приложений. При обработке же 16-разрядных слов никакого выигрыша не наблюдается, налицо даже небольшой проигрыш по быстродействию (при одной и той же тактовой частоте). Вообще ПЭВМ на МП 80386SX являются переходными между машинами класса AT и компьютерами на процессоре 80386. По цене они недалеко отстоят от AT, но могут выполнять программы, созданные специально для МП 80386.

Если Вы смотрите в будущее, но стеснены в денежных средствах, то наилучший выбор для Вас — машина на базе МП 80386SX. Первой фирмой, использовавшей в 1988 г. МП 80386SX, была компания Compaq.

Основные технические характеристики ряда наиболее известных моделей AT-совместимых ПЭВМ представлены в табл. 3.4. В колонке «МП» этой таблицы за тактовой частотой в круглых скобках указано число тактов ожидания, если оно фирмой сообщается. Буква Н обозначает обычное настольное исполнение, а Б — настольное в виде башни. Если дисплей не указан, то он полностью соответствует адаптеру.

В таблице не отражено следующее:

— все модели поставляются с усовершенствованной 101-клавишной клавиатурой;

— ПЭВМ снабжены одним — двумя адаптерами последовательного интерфейса и, как правило, одним адаптером параллельного интерфейса;

— некоторые модели имеют адаптер игрового интерфейса или адаптер «мыши»;

— компьютеры имеют как 8-, так и 16-разрядные гнезда для расширения ресурсов;

— шина у всех моделей аналогична шине IBM PC AT;

— цены колеблются в пределах 1200 — 6000 долл., в зависимости от комплектации ПЭВМ и фирмы-производителя;

— ПЭВМ Global/286 и CompuStar 286 поставляются с сопроцессором;

— машины ALR FlexCache SX386Z, ALR VIP SX386 и AGC Powerlay SX+ снабжены 16-Кбайт, а Proteus 286 STX — 32-Кбайт кэшем;

— изделия фирмы PC BRAND содержат аппаратно реализованную спецификацию отображаемой памяти LIM EMS 4.0.

В AT-совместимых ПЭВМ уже начали применяться 25-МГц МП (модель ACCESS 286/25), нередко используются графические стандарты VGA — Super VGA и НЖМД большой емкости (до 344 Мбайт). Довольно распространено комплектование ПЭВМ монохромным VGA-монитором с моделированием цветов различными градациями яркости.

ПЭВМ класса AT способны работать как в *однопользовательском однозадачном*, так и в *однопользовательском многозадачном* режимах. Последняя возможность должна быть реализована посредством ОС. Возможно также построение на базе ПЭВМ этого класса многопользовательских микроЭВМ.

Основными ОС для компьютеров класса AT являются MS-DOS (PC DOS), OS/2, а также различные варианты ОС UNIX и особенно Xenix.

К сожалению, в табл. 3.4 из-за отсутствия у автора информации не представлены новые AT-совместимые модели известных фирм Tandy, Zenith Data Systems и Ing. C. Olivetti & Co. Известно лишь, что последняя выпускает модель M250 стоимостью 3800 долл. с великолепной эргономикой, а также модели PSC-286 и M290 с достаточно высоким быстродействием.

Среди изделий фирм, еще не упоминавшихся в этом разделе в связи с высокими характеристиками своих ПЭВМ, в классе AT следует отметить машины Everex STEP 286/16 и 286/20, а также Club AT 316SX и Acer 1100SX.

Как уже указывалось, цены на AT-совместимые ПЭВМ колеблются в достаточно широком диапазоне. Так, например, машина MYODA MD3410 без периферии, за исключением НГМД и клавиатуры, стоит всего 685 долл., а изделие ALR FlexCache SX386Z, полностью укомплектованное,

Таблица 3.4

Технические характеристики стационарных AT-совместимых ПЭВМ

Производитель: фирма, страна	Модель	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Количество НГМД x емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер-дисплей	Исполнение
1	2	3	4	5	6	7	8	9
ACCESS Micro Systems, США	ACCESS 286/12	80286-12/6	80287	512K..4M	1x1,2M-133	40:28, 80:28, 159:23	моно, EGA, VGA	Н
	ACCESS 286/20	80286-20/10 (1&0)	80287	1M..5M	1x1,2M-133	40:28, 80:28, 159:23	моно, EGA, VGA	Н
	ACCESS 286/25	80286-25/10 (1&0)	80287	1M..5M	1x1,2M-133	40:28, 80:28, 159:23	моно, EGA, VGA	Н
Acer Technology, Тайвань	Acer 1100SX	80386SX-16	80387SX	8M	по выбору	до 338	VGA	Н
AGC Electronics, Тайвань	AGC Powerlay SX+	80386SX-20	н/д	2M	по выбору	по выбору	VGA	Б
Advanced Logic Research, США	ALR FlexCache SX3062	80386SX-16(0)	80387SX	1M..8M	1x1,2M-133 [1x1,2M-133& 1x1,44M-89]	40	VGA	Н
	ALR VIP SX386	80386SX-16(0)	80387SX	1M..8M	1x1,2M-133 [1x1,2M-133& 1x1,44M-89]	40	VGA	Б
Applied Microsystems Technology, Великобритания	AMT 286xii	80286-12/8/6	н/д	1M..н/д	1x1,2M-133	20[40, 80]	CGA, EGA, HGA, PGA	Н
Aquarius Systems Integral, СНГ	ASI-286-12	80286-12	80287	1M..4M	1x1,2M-133 [1x1,2M-133& 1x1,44M-89]	20:40, 40:19	VGA	Н
	ASI-386/SX-DT	80386SX-16/8	80387SX	1M..8M	1x1,2M-133 [1x1,2M-133& 1x1,44M-89]	40:19, 80:19, 170	VGA	Б
AST Research, США	AST Bravo/286	80286-8	80287	512K	1x1,2M-133	н/д	н/д	Н
	AST Bravo/286/16	80286-16	80287	1M	1x1,44M-89	по выбору	Super VGA	Н
	AST Bravo/386SX	80386SX-16	80387SX	2M..8M	1x1,44M-89	до 320	VGA	Н
	AST Bravo/386SX/20	80386SX-20	80387SX	2M..8M	1x1,44M-89	по выбору	Super VGA	Н
	AST Premium Workstation/286	80286-10/6(1)	80287	1M..4M	по выбору	40	HGA, EGA, Enhanced EGA	Н
	AST Premium/286	80286-10/8/6(0)	80287	1M..13M	по выбору	20, 40, 70	HGA, EGA	Н
BEST Computer, США	BEST VTI-55	80286-12/8	н/д	640K	1x1,2M-133	40	EGA	Н
BUSH, Индия	BUSH PC-AT	80286-10	н/д	640K	1x1,2M-133	40	EGA	Н

Таблица 3.4 (продолжение)

1	2	3	4	5	6	7	8	9
Clone Computers, США	Clone 286-10	80286-10	-	640K..1M	1x360K-133, 1x1,2M-133, 1x1,44M-89	30 [32..122]	HGA, CGA, EGA, VGA-м, VGA-ц	H
	Clone 286-12	80286-12	80287	640K..1M	1x360K-133, 1x1,2M-133, 1x1,44M-89	40 [32..122]	HGA, CGA, EGA, VGA-м, VGA-ц	H
	Clone 286-16	80286-16	80287	н/д	1x360K-133, 1x1,2M-133, 1x1,44M-89	65 [32..122]	HGA, CGA, EGA, VGA-м, VGA-ц	H
Compaq Computer, США	Compaq DESKPRO 286	80286-8	80287	до 8,2M	1x1,2M-133	до 70	н/д	H
	Compaq DESKPRO 286e	80286-12	80287	1M..13M	по выбору	20, 40, 110	VGA	H
	Compaq DESKPRO 386s	80386SX-16	80387SX	1M..13M	1x1,2M-133	40..320	VGA	H
	Compaq DESKPRO 386s/20	80386SX-20	80387SX	2M	1x1,44M-89	-, 60, 120	VGA	H
Corporate Computers of Iowa, США	CCI AT 286/20	80286/20/10	80287	7M..8M	1x1,2M-133& 1x1,44M-89	80:28	Enhanced VGA	н/д
Dell Computer, США	Dell System 210	80286-12,5(0)	80287	512K..16M	1x1,2M-133, 1x1,44M-89	20, 40:29, 100:25	VGA plus- (VGA м, VGA ц)	H
	Dell System 220	80286-20	80287	1M..16M	1x1,44M-89 [2x1,44M-89]	40:29, 100:25	VGA plus- (VGA м, VGA ц)	H
	Dell System 316SX	80386SX-16(0)	80387SX	512K..16M	1x1,2M-133, 1x1,44M-89	20, 40:29, 100:25, 190	Super VGA- (VGA м, VGA plus ц, Super VGA ц)	H
	Dell System 320LX	80386SX-20	80387SX	512K..16M	1x1,2M-133, 1x1,44M-89	20, 40:29, 100:25	Super VGA- (VGA м, VGA plus ц, Super VGA ц)	H
Everex Computer Systems, США	Everex STEP 286/8	80286-8	н/д	512K-н/д	1x1,2M-133	по выбору	по выбору	н/д
	Everex STEP 286/12	80286-12	н/д	1M-н/д	1x1,2M-133	по выбору	по выбору	н/д
	Everex STEP 286/16	80286-16/8/5(0)	80287-10	до 4M	1x1,2M-133	40..300	VGA	н/д
	Everex STEP 286/20	80286-20	80287	н/д	1x1,2M-133	40..300	VGA	н/д
	Everex STEP 386s	80386SX-16	80387SX	2M..4M	1x1,2M-133	40..300	VGA	н/д
Gateway 2000, США	Gateway 286/12	80286-12	н/д	2M	1x1,2M-133& 1x1,44M-89	40, 65:28	Super VGA	H
	Gateway 286/16	80286-16	н/д	2M	1x1,2M-133& 1x1,44M-89	40, 65:28	Super VGA	H
	Gateway 286/20	80206-20	н/д	2M	1x1,2M-133& 1x1,44M-89	40, 65:28	Super VGA	H
	Gateway 386/SX	80386SX-16	н/д	2M	1x1,2M-133& 1x1,44M-89	40, 65:28	Super VGA	H

Таблица 3.4 (продолжение)

1	2	3	4	5	6	7	8	9
Global Computer Network, США	Global/286-12	80286-12/8	80287	1M..4M	1x1,2M-133& 1x1,44M-89	40	Super EGA [VGA]	Н
	Global/286-16	80286-16/12	80287	1M..4M	1x1,2M-133& 1x1,44M-89	40	Super EGA [VGA]	Н
Hewlett-Packard, США	HP Vectra	80206-8	80287	н/д	1x1,2M-133	20..40	EGA	Н
	HP Vectra 286/12	80286-12	80287	1M..8M	1x1,2M-133, 1x1,44M-89	20, 42:19, 84:19	Super VGA	Н
Hitachi, Япония	B16EX-III	80286-12,5	н/д	1M..5M	2x1,44M-89	20, 40	EGA	Н
	B16HX-II	80286-12,5	н/д	1M..5M	2x1,44M-89	20, 40	1120x720	Н
Jameko Electronics, США	JE3008	80286-12/8	80287-8	512K..1M	по выбору	по выбору	по выбору	Н
	JE3013	80286-16/8	80287-10	2M..8M	по выбору	по выбору	по выбору	Н
Micronics, США	FIVESTAR 286/10	80286-10(0)	80287	512K..4M	1x1,2M-133	20..320	EGA, VGA	Н
	FIVESTAR 286/14	80286-14(0)	80287	1M..4M	1x1,2M-133	20..320	EGA, VGA	Н
	FIVESTAR 386/16SX	80386SX-16(0)	80387SX	1M..4M	1x1,2M-133	20..320	EGA, VGA	Н
NEC, Япония	APC IV	80286-8	н/д	640K	1x360K-133, 1x1,2M-133	20, 40	1120x750	Н
OWL Computer Services, США	OWL ATOM-12/6	80286-12	н/д	512K	1x1,2M-133	30	по выбору	Н
	OWL ATOM-16/6	80286-16	н/д	512K	1x1,2M-133	30	по выбору	Н
PAO-KU International, США	MYODA MD3410	80286-12(0)	80287	1M..4M	1x1,2M-133	20:65, 40:28, 80:20, 120:28	VGA, VGA plus	Б
	MYODA MD5030	80386SX-16	80387SX	1M..8M	1x1,2M-133	20:65, 40:28, 80:20, 120:28	VGA, VGA plus	Б
PC BRAND, США	PCBRAND 286/12	80286-12(0)	80287	512K..8M	1x1,2M-133, 1x1,44M-89 [1x1,2M-133& 2x1,44M-89]	20:40, 40:25, 66:25, 100:25	Super VGA-(м, VGA м, VGA ц, Super VGA ц)	Н, Б
	PCBRAND 286/16	80286-16	80287	512K..8M	1x1,2M-133, 1x1,44M-89 [1x1,2M-133& 2x1,44M-89]	20:40, 40:25, 66:25, 100:25	Super VGA-(м, VGA м, VGA ц, Super VGA ц)	Н, Б
	PCBRAND 286/20	80286-20	80287	512K..8M	1x1,2M-133, 1x1,44M-89 [1x1,2M-133& 2x1,44M-89]	20:40, 40:25, 66:25, 100:25	Super VGA-(м, VGA м, VGA ц, Super VGA ц)	Н, Б
	PCBRAND 386/SX-16	80386SX-16	80387SX	512K..8M	1x1,2M-133, 1x1,44M-89 [1x1,2M-133& 2x1,44M-89]	20:40, 40:25, 66:25, 100:25	Super VGA-(м, VGA м, VGA ц, Super VGA ц)	Н, Б

Таблица 3.4 (окончание)

1	2	3	4	5	6	7	8	9
Proteus, США	Proteus 286GTX	80286-12(0)	н/д	640K..16M	1x1,2M-133	20, 40, 80	моно, VGA, VGA plus	н/д
Seiko Epson, Япония	Epson EL2	80286-10	80287	640K..4,6M	1x1,44M-89	40	VGA	Н
	Epson EL3	80386SX-16	80387SX	1,6M..5,6M	1x1,44M-89	40	VGA	Н
	Epson Equity 386SX	80386SX-16	80387SX	до 14M	по выбору	до 100	н/д	Н
Swan Technologies, США	Swan 286/12	80286-12,5/6,25	80287	512K..5M	по выбору	32:40, 48:28, 80:28	моно, VGA-м, VGA-ц	Н
	Swan 386SX	80386SX-16/8	80387SX	1M..8M	по выбору	32:40, 48:28, 80:28	моно, VGA-м, VGA-ц	Н
Tandy, США	Tandy 3000HL	80286-8/4	80287	512K..640	1x1,2M-133	10, 20	CGA	Н
	Tandy 3000	80286-8	80287	н/д	1x1,2M-133	20..40	EGA	Н
	Tandy 3000NL	80286-10	80287	до 16M	1x1,44M-89	н/д	EGA	Н
	Tandy 4016SX	80386SX-16	80387SX	4M	1x1,44M-89	до 344	н/д	Н
The Ultimate Business Machines, США	Club Model 200	80286-10/8	80287	н/д	1x1,2M-133	по выбору	EGA, VGA	Н
	Club AT 316SX	80386SX-16	80387SX	2M..16M	1x1,2M-133	40..300	VGA	Н
Wells American, США	CompuStar 286	80286-20	80287-10	1M..16M	2x1,44M-89& 1x1,2M-133	150	VGA	Н
Zenith Data Systems, США	Zenith Z-200	80286-6	80287	512K..16M	1x1,2M-133	-, 20	EGA	Н
	Zenith Z-286LP	80286(0)	80287	1M..16M	н/д	н/д	н/д	Н
ZEOS International, США	ZEOS 286/12	80286-12/6(0)	80287	512K..16M	1x1,2M-133	32	HGA[EGA,VGA]	н/д
	ZEOS 286/16	80286-16	80287	512K..16M	1x1,2M-133	32	HGA[EGA,VGA]	н/д
	ZEOS 286/20	80286-20	80287	512K..16M	1x1,2M-133	32	HGA[EGA,VGA]	н/д
	ZEOS 386SX	80386SX-16/8	80387SX	512K..16M	1x1,2M-133	32	HGA[EGA,VGA]	н/д

но без принтера, — 5337 долл. Чтобы получить более полное представление о стоимости AT-совместимых ПЭВМ, достаточно взглянуть на табл. 3.5, в которой приведены действовавшие весной 1990 г. цены в долларах на машины PCBRAND 286/12, если емкость ОЗУ составляет 512 Кбайт. Чтобы получить цены на модели 286/16 и 286/20, следует добавить 150 и 300 долл. соответственно. Стоимостная характеристика ПЭВМ PCBRAND 386/SX-16 (также в долларах) дана в табл. 3.6.

Таблица 3.5

Цены на ПЭВМ PCBRAND 286/12

НЖМД / Дисплей	–	20:40	40:25	66:25	100:25
–	599	929	1029	1209	1349
монохромный	719	1049	1149	1329	1469
VGA (монохромный)	899	1229	1329	1509	1649
VGA (цветной)	1029	1459	1559	1739	1879
Super VGA (цветной)	1139	1569	1669	1849	1989

Таблица 3.6

Цены на ПЭВМ PCBRAND 386/SX-16

НЖМД / Дисплей	–	20:40	40:25	66:25	100:25
–	899	1229	1329	1509	1649
монохромный	1019	1349	1449	1629	1769
VGA (монохромный)	1199	1529	1629	1809	1949
VGA (цветной)	1429	1759	1859	2039	2179
Super VGA (цветной)	1539	1869	1969	2149	2289

Вообще средние базовые цены на ПЭВМ с МП 80286-12, 80386SX-16 и 80386SX-20 составляют 1248, 1775 и 2219 долл.

ПЭВМ класса AT вполне подходят для большинства профессиональных приложений, если не требуется производить большие объемы вычислений.

ПЭВМ на базе микропроцессора 80386

ПЭВМ на базе МП 80386 догоняют по объему сбыта AT-совместимые машины и в недалеком будущем станут наиболее массовыми. Напомним, что первой ПЭВМ, построенной на базе МП 80386, была 16-МГц машина Compaq DESKPRO 386, уже снятая с производства.

На вопрос о том, с какой моделью IBM совместимы ПЭВМ с МП 80386, однозначно ответить нельзя. Все зависит от того, какая шина использована в том или ином компьютере. Если применена шина MCA (или совместимая с ней), то ПЭВМ совместима как с семейством PC, так и с семейством PS/2. В противном случае обеспечивается совместимость только с семейством PC (точнее — совместимость снизу вверх с IBM PC AT). При этом говорят о классе AT-386.

Первой ПЭВМ с MCA-совместимой шиной была Tandy 5000 MC. Сейчас же с шиной MCA выпускаются компьютер ALR MicroFlex 7000 и ряд других. Стала использоваться и шина EISA (ПЭВМ Compaq SYSTEMPRO, Zenith Z-386/33E и др.). Большинство же машин имеют свои собственные шины, созданные путем расширения шины AT.

Основные технические характеристики широкого спектра моделей ПЭВМ, построенных на базе МП 80386, сведены в табл. 3.7. Напольное исполнение в виде стойки отражено буквой «С» в соответствующем столбце.

Стандартной для компьютеров данного класса считается усовершенствованная 101-клавишная клавиатура. Практически все модели снабжены одним — двумя адаптерами последовательного и одним адаптером параллельного интерфейсов. Не редкость также адаптеры игрового интерфейса и адаптеры «мыши». Используются 8-, 16- и 32-разрядные гнезда для расширения ресурсов. К приведенным ценам нужно относиться осторожно, так как они указаны для различных комплектаций ПЭВМ и к «общему знаменателю» их привести трудно; кроме того, ярко выражена тенденция снижения цен. Отметим также, что ПЭВМ CAF Master 386/25T, а также Global/386-33 в стандартной конфигурации поставляются с сопроцессором. Для остальных же моделей он действительно является факультативным. Ряд ПЭВМ предусматривают в качестве факультативных устройств НМЛ и реже — НОД.

ПЭВМ рассматриваемого класса могут работать на частотах от 16 до 33 МГц, имеют большую емкость ОЗУ и НЖМД, а также развитые графические возможности. В качестве сопроцессоров изредка предусматриваются МП 80287 и Weitek Abacus 3167 (последний — для высококачественных моделей), но наиболее часто применяется МП 80387. Отнюдь не редкость конструктивное исполнение мощных машин в виде башни.

Все указанные в табл. 3.7 компьютеры могут использоваться в качестве ПЭВМ в традиционном понимании, т.е. в роли *универсальных однопользовательских* вычислительных машин (*однозадачных*

Таблица 3.7

Технические характеристики стационарных ПЭВМ на базе МП 80386

Производитель: фирма, страна	Модель	МП	Факультативный сопроцессор	Емкость кэш-памяти, Кбайт	Емкость ОЗУ, Мбайт	Количество НГМД х емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер-дисплей	Исполнение	Цена, долл.
1	2	3	4	5	6	7	8	9	10	11
ACCESS Micro System, США	ACCESS 386/16	80386-16/8(0)	80387	н/д	1..10	1x1,2M-133	[80:28, 159:23, 338:18]	моно, EGA, VGA	Н	от 1695
	ACCESS 386/20	80386-20/8(0)	80387	н/д	1..10	1x1,2M-133	[80:28, 159:23, 338:18]	моно, EGA, VGA	Н	от 1995
	ACCESS 386/20 Tower	80386-20/8(0)	80387	н/д	1..10	1x1,2M-133	[80:28, 159:23, 338:18]	моно, EGA, VGA	Б	от 2099
	ACCESS 386/33	80386-33/8/6(0)	80387	н/д	1..10	1x1,2M-133	[80:28, 159:23, 338:18]	моно, EGA, VGA	Н	от 3795
Acer Technology, Тайвань	Acer 1100/25	80386-25	80387	н/д	до 40	1x1,2M-133	н/д	н/д	н/д	9673
Advanced Logic Research, США	ALR FlexCache 20/386	80386-20	80387	н/д	1..16	по выбору	н/д	н/д	Н	н/д
	ALR FlexCache 25/386	80386-25	80387	64	1..16	2 шт. по выбору	н/д	VGA	Н	9517
	ALR MicroFlex 7000	80386-25	80387	64	2..н/д	1x1,44M-89 [1x1,44M-89& 2x1,2M-133]	120, 300	Super VGA	Б	9499
	ALR FlexCache 33/386Z	80386-33	80387	32	1..16	1x1,44M-89, 2x1,2M-133	150:17 [300:18, 330:16, 340, 650:16]	Super VGA	Н	от 3995
	ALR FlexCache 33/386DT	80386-33	80387& Weitek 3167	128	2..16	1x1,2M-133	-, 120, 340	VGA	Н	4490-8490
	ALR FlexCache 33/386	80386-33	80387& Weitek 3167	128	2..16	1x1,44M-89 [1x1,44M-89& 1x1,2M-133]	150:17 [300:18, 330:16, 340, 650:16]	Super VGA	С	9990-15490
	ALR MicroFlex 3300	80386-33	80387& Weitek 3167	128	4..64	1x1,44M-89	120, 642:16	VGA	Б	14006
AGC Electronics, Тайвань	AGC Powerlay 386+	80386-33	н/д	64	2	по выбору	по выбору	VGA	Н	н/д

Таблица 3.7 (продолжение)

1	2	3	4	5	6	7	8	9	10	11
Ami, США	Dyna Cache 386	80386-33	80387& Weitek 3167	64	4..16	1x1,2M-133	147:18	VGA	Н	9993
Aquarius Systems Integral, СНГ	ASI 386/33-DT	80386-33/8	н/д	н/д	1..н/д	1x1,44M-89& 1x1,2M-133	40	VGA	Б	н/д
Arche Technologies	Arche Legacy 386-33	80386-33	80387, Weitek 3167	н/д	8..32	1x1,44M-89& 1x1,2M-133	380	н/д	Н	5606-12285
AST Research, США	AST Premium 386/16	80386-16	н/д	-	н/д	по выбору	н/д	н/д	Н	н/д
	AST Premium/386	80386-20(0&1)	н/д	-	2..13	по выбору	40, 90, 150, 320	по выбору	Н	н/д
	AST Premium/386C	80386-20	н/д	-	н/д	по выбору	н/д	н/д	Н	н/д
	AST Premium 386/25	80386-25	н/д	н/д	до 36	до 3x1,2M-133	110:16	VGA	н/д	н/д
	AST Premium 386/33	80386-33(0)	80387& Weitek 3167	32	2..36	1x1,2M-133& 1x1,44M-89	110:16, 330:16	VGA plus-по выбору	н/д	8495-11195
BEST Computer, США	BEST 386/20	80386-20	80287, 80387, Weitek 3167	-	1..8	1x1,2M-133	68	VGA	Н	н/д
	BEST 386/20 Workstation	80386-20	н/д	-	1..8	1x1,2M-133& 1x1,44M-89	80	VGA plus	Б	2695
	BEST 386/25 Workstation	80386-25	80387-25	-	4..16	1x1,2M-133& 1x1,44M-89	150	Super VGA	Б	4395
CAF, Тайвань	CAF Master 386/25T	80386-25	80387	32	1..н/д	н/д	70	VGA	н/д	8310
	CAF Master 386C/33T	80386-33	80387	64	4..8	1x1,2M-133& 1x1,44M-89	150	Super VGA	Б	от 3179
Compaq Computer, США	Compaq DESKPRO 386	80386-16	80287	-	1..14	1x1,2M-133	20, 40:30, 130:25	EGA	Н	4499
	Compaq DESKPRO 386/20e	80386-20(0)	н/д	32	1..н/д	1x1,2M-133	40	VGA	Н	н/д
	Compaq DESKPRO 386/25	80386-25(0)	80387& Weitek 3167	32	до 16	1x1,2M-133	300	VGA	Н	н/д
	Compaq DESKPRO 386/33	80386-33(0)	80387& Weitek 3167	64	2..16	1x1,2M-133	84:25,7, 110, 320, 650	VGA-по выбору	Н	10499-17999
	Compaq SYSTEMPRO	2x80386-33(0)	80387& Weitek 3167	64	4..256	по выбору	4x210 [8x210]	по выбору	Б	от 16000

Таблица 3.7 (продолжение)

1	2	3	4	5	6	7	8	9	10	11
Dell Computer, США	Dell System 316	80386-16	80387	-	1..16	1x1,2M-133, 1x1,44M-89	40:29 [100:25, 150:18, 322:18]	VGA plus-(VGA м, VGA plus ц)	H	3000-6000
	Dell System 320	80386-20	80387, Weitek 3167	32	1..16	1x1,2M-133, 1x1,44M-89	40:29 [100:25, 150:18, 322:18]	Super VGA-(VGA м, VGA plus ц, Super VGA)	H	н/д
	Dell System 325	80386-25	80387, Weitek 3167	32	1..16	1x1,2M-133, 1x1,44M-89	100:25 [40:29, 150:18, 322:18]	Super VGA-(VGA м, VGA plus ц, Super VGA)	H	4199-6299
	Dell System 333D	80386-33	80387, Weitek 3167	есть	1..16	1x1,2M-133, 1x1,44M-89	до 650	Super VGA-(VGA м, VGA plus ц, Super VGA)	H	4659
Everex Computer Systems, США	Everex STEP 386/16	80386-16	н/д	64	1..н/д	1x1,2M-133	по выбору	по выбору	н/д	н/д
	Everex STEP 386/20	80386-20	н/д	64	1..н/д	1x1,2M-133	по выбору	по выбору	н/д	н/д
	Everex STEP 386/25	80386-25/12/8	н/д	64[256]	2..16	по выбору	150	EGA	н/д	9347
	Everex STEP 386/33	80386-33	80387& Weitek 3167	64[256]	2..16	1x1,2M-133	160:18, 330:18	VGA	н/д	от 7599
Gateway 2000, США	Gateway 386/20	80386-20	-	[64]	1..н/д	1x1,2M-133& 1x1,44M-89	65:28, 80:28	VGA	н/д	от 2695
	Gateway 386/25	80386-25	н/д	[64]	2..н/д	1x1,2M-133& 1x1,44M-89	150:17	VGA	н/д	от 3395
	Gateway 386/33	80386-33	80387& Weitek 3167	64	2..32	1x1,2M-133& 1x1,44M-89	150:17	VGA, Super VGA	н/д	от 4395
Global Computer Network, США	Global/386-20	80386-20/16	80387	-	4	1x1,2M-133& 1x1,44M-89	120	Super VGA	н/д	3960
	Global/386-33	80386-33	80387	-	4	1x1,2M-133& 1x1,44M-89	120	Super VGA	н/д	4800
Haupauge, США	PC LInk 386/33	80386-33	80387, Weitek 3167	64	4..64	1x1,2M-133	159:17	VGA-по выбору	н/д	5995
Hertz Computer, США	Hertz 386/25	80386-25	н/д	64	2..24	5 отсеков	160, 300	VGA	н/д	9053-10375

Таблица 3.7 (продолжение)

1	2	3	4	5	6	7	8	9	10	11
Hewlett-Packard, США	HP Vectra RS/16	80386-16	н/д	-	16..32	1x1,2M-133	40	н/д	н/д	6495
	HP Vectra RS/20	80386-20	н/д	-	20..36	1x1,2M-133	103, 155, 310:17	н/д	н/д	7495-11995
	HP Vectra RS/25C	80386-25	н/д	н/д	н/д	н/д	н/д	н/д	н/д	14705
	HP Vectra 386/25 PC	80386-25	80387	32	2..н/д	н/д	42, 84, 170, 340:17	Super VGA	н/д	от 9145
Ing. C. Olivetti & Co, Италия	Olivetti M380 XP9	80386-33	80387	32	до 64	1x1,2M-133, 1x1,44M-89	135:20, 300:20	VGA	н/д	н/д
Jameko Electronics, США	JE 3550	80386-20/16/8	80387-20	-	1..16	1x1,2M-133& 1x1,44M-89	60	н/д	н/д	от 1900
	JE 3555	80386-25	80387-25, Weitek 3167	-	4..16	1x1,2M-133& 1x1,44M-89	120	н/д	н/д	н/д
Micro Express, США	ME 386-20	80386-20	-	н/д	н/д	н/д	н/д	н/д	н/д	н/д
	ME 386-25	80386-25(0)	н/д	64	до 32	3 шт. по выбору	150	по выбору	н/д	6324
	ME 386-33	80386-33(0)	80387, Weitek 3167	64	до 40	3 шт. по выбору	по выбору	по выбору	н/д	н/д
Micro 1, США	Micro 1 Power 386-20	80386-20	-	-	1	1x1,2M-133, 1x1,44M-89	30	HGA	н	1995
Micronics, США	FIVESTAR 386/16	80386-16/8	80387	-	1..10	по выбору	по выбору	по выбору	н/д	от 1995
	FIVESTAR 386/20	80386-20/6	80387	64	1..16	по выбору	по выбору	по выбору	н/д	от 2495
	FIVESTAR 386/33	80386-33	80387, Weitek 3167	32	1..16	1x1,2M-133	по выбору	по выбору	н/д	от 3395
	Blackship 386/33	80386-33	80387, Weitek 3167	32	1..16	1x1,2M-133	40:28	моно	н/д	4195
	Matrix MDP 386-33	80386-33	80387, Weitek 3167	32	4..16	1x1,2M-133	по выбору	по выбору	н/д	от 5585
	National Microsystems Flash 386-33	80386-33	80387, Weitek 3167	64	4..16	1x1,2M-133	по выбору	по выбору	н/д	н/д
Mylex, США	Tangent 333	80386-33	80387, Weitek 3167	128	4..32	1x1,2M-133	100:18	VGA	н/д	6995
NEC Home Electronics, США (филиал NEC, Япония)	NEC ProSpeed 386	80386-16	н/д	-	до 10	по выбору	40, 100	EGA	н/д	н/д
	NEC PowerMate 386/33E	80386-33	н/д	н/д	до 16	по выбору	до 300	по выбору	н/д	5108

Таблица 3.7 (продолжение)

1	2	3	4	5	6	7	8	9	10	11
Northgate Computer Systems, США	Elegance 386/20 Elegance 386/25 Elegance 386/33	80386-20 80386-25 80386-33(0)	н/д н/д 80387, Weitek 3167	н/д н/д 64[256]	н/д н/д 4	н/д н/д 1x1,2M-133	н/д н/д 150:16	н/д н/д VGA	н/д н/д Б	н/д н/д 3395
OWL Computer Services, США	OWL SUPER-ATOM-386	80386-20	80287, 80387	64	1..2	1x1,2M-133	40	EGA-м	н/д	2995
PAO-KU International, США	MYODA MD7240	80386-25(0)	80387	64	4..16	1x1,2M-133	20:65, 40:28, 80:28, 120:28	VGA, VGA plus	Б	2399
PC BRAND, США	PCBRAND 386/20	80386-20(0)	80287, 80387, Weitek 3167	–	1..16	1x1,2M-133, 1x1,44M-89	20:40, 40:25, 66:25, 100:25	Super VGA –(м, VGA м, VGA ц, Super VGA ц)	Н, Б	1779–3189
	PCBRAND 386/25	80386-25(0)	80287, 80387, Weitek 3167	[32, 64]	1..16	1x1,2M-133, 1x1,44M-89	20:40, 40:25, 66:25, 100:25	Super VGA –(м, VGA м, VGA ц, Super VGA ц)	Н, Б	1929–3339
	PCBRAND 386/33 Cache	80386-33(0)	80387, Weitek 3167	32[64]	1..16	1x1,2M-133, 1x1,44M-89	20:40, 40:25, 66:25, 100:25	Super VGA –(м, VGA м, VGA ц, Super VGA ц)	Н, Б	2679–4089
Platform Operation, США (филиал Intel)	Platform 302-20	80386-20	н/д	н/д	2..16	по выбору	по выбору	по выбору	Н	н/д
	Platform 303	80386-33(0)	н/д	64	8..40	по выбору	по выбору	по выбору	С	н/д
Proteus, США	Proteus 386-16	80386-16(0)	–	–	0,625	1x1,2M-133	40:28, 80, 100, 150, 340	моно, EGA plus, VGA plus	н/д	2899–6374
	Proteus 386-20	80386-20(0)	–	32	1..16	1x1,2M-133	40:28, 80, 100, 150, 340	моно, EGA plus, VGA plus	н/д	н/д
	Proteus 386-25	80386-25(0)	80387	32	1..32	1x1,2M-133	40..700	моно, EGA plus, VGA plus	н/д	5635–9174
	Proteus System 3400ME	80386-25 [80386-33]	80387	32	4..н/д	1x1,2M-133, 1x1,44M-89	150 [до 700]	моно, EGA plus, VGA plus	н/д	8795
Simple-Net Systems, США	NetPro 386/25	80386-25	80387	64	н/д	1x1,44M-89	120	VGA	н/д	1350

Таблица 3.7 (окончание)

1	2	3	4	5	6	7	8	9	10	11
Swan Technologies, США	Swan 386/20D	80386-20/8	80287, 80387	–	1..16	по выбору	48:28, 80:28, 150:18	моно, VGA-(м, ц)	н/д	н/д
	Swan 386/33	80386-33/8	80387, Weitek 3167	32	1..16	1x1,2M-133д 1x1,44M-89	150:18, 300:17, 600:17	моно, VGA-(м, ц)	н/д	н/д
Systems Integration Associates, США	SIA 386/32	80386-25 (на частоте 32,5)	80387, Weitek 3167	64	1..16	по выбору	по выбору	по выбору	С	13100-19800
	SIA 386/33	80386-33	80387, Weitek 3167	64	1..16	1x1,2M-133	по выбору	по выбору	н/д	от 6490
Tandy, США	Tandy 4000	80386-16	80387	–	1..н/д	1x1,44M-89	по выбору	VGA	н/д	н/д
	Tandy 4000LX	80386-20	80387	–	н/д	1x1,44M-89	по выбору	VGA	Н	н/д
	Tandy 5000MC	80386-20	80387	32	2..н/д	1x1,44M-89	80..н/д	VGA	Н	н/д
	Tandy 4033LX	80386-33	80387	н/д	до 16	1x1,44M-89	до 440	VGA	н/д	7072
Toshiba America, США (филиал Toshiba, Япония)	Toshiba T8500	80386-25	80387	есть	до 8	2x1,44M-89	100	VGA	Н	н/д
The Ultimate Business Machines, США	Club 386	80386-16	н/д	н/д	н/д	н/д	н/д	н/д	н/д	н/д
	Club Model 30	80386-20/8 [80386-16/8]	80387	н/д	1..н/д	1x1,2M-133	40, 70, 130	EGA, VGA	н/д	н/д
Wang Microsystems, США	Wang PC/381	80386-16	80387	–	2..16	1x1,2M-133 [2x1,2M-133, 1x1,2M-133д 1x1,44M-89]	20, 42, 68, 143, 321	VGA-(м, VGA plus ц)	Н	н/д
	Wang PC/382	80386-20	80387	–	2..16	1x1,2M-133 [2x1,2M-133, 1x1,2M-133д 1x1,44M-89]	20, 42, 68, 143, 321	VGA-(м, VGA plus ц)	Н, Б	6875
Zenith Data Systems, США	Zenith Z-386/25	80386-25	80387	н/д	н/д	н/д	н/д	н/д	н/д	н/д
	Zenith Z-386/33	80386-33	80387	16	2..64	1x1,44M-89	150:18, 320:14	VGA-по выбору	н/д	11499-13499
	Zenith Z-386/33E	80386-33	80387	н/д	4..20	1x1,44M-89 [1x1,2M-133]	150:18, 320:14	VGA	н/д	11999-13799
ZEOS International, США	ZEOS 386/16	80386-16	80287, 80387	–	1..16	1x1,2M-133	80:28	HGA, EGA, VGA	Н, Б	2495-3190
	ZEOS 386/20	80386-20(0)	80287, 80387	64	1..16	1x1,2M-133	80:28	HGA, EGA, VGA	Б	2995-3690
	ZEOS 386/25	80386-25/8(0)	80387	64	1..16	1x1,2M-133	80:28	HGA, EGA, VGA	Б	3495-4190
	ZEOS 386/33	80386-33(0)	80387	128	1..24	1x1,2M-133	80:28, 160, 337, 674, 2x674	HGA, EGA, VGA	Б	3495-4190

или *многозадачных*). Как автономные вычислительные установки компьютеры на базе МП 80386 целесообразно применять для обработки изображений, автоматизированного проектирования и выполнения расчетов, связанных с большим объемом вычислений. Однако ресурсы памяти (как основной, так и внешней) некоторых из них настолько велики, что не имеет смысла отдавать компьютер целиком одному пользователю (последнее напоминает «стрельбу из пушки по воробьям»). Поэтому компьютеры данного класса часто используются для создания *многопользовательских* систем или в качестве *файл-серверов* в ЛВС, если машина оборудована НЖМД большой емкости.

Компьютеры на базе МП 80386 могут работать под управлением ОС MS-DOS (PC DOS), OS/2, UNIX-подобных систем (в частности Xenix) и PC MOS. Для ряда моделей разработаны новые версии DOS, в которых устранены некоторые присущие ей ограничения. Так, например, DOS 3.3 Plus для Zenith Z-386 устраняет 32-Мбайт ограничение на размер логического диска на винчестере; ОС Compaq DOS 3.3 способна адресовать внешнюю память объемом до 300 Мбайт.

Информация, содержащаяся в табл. 3.7, конечно же не дает полного представления о качестве, в том числе производительности ПЭВМ. При выборе машины можно интуитивно полагаться на репутацию фирмы, однако и это не всегда срабатывает. В частности, хорошими характеристиками обладают изделия еще не прославившихся фирм, таких, как Everex Computer Systems, Hertz Computer, Micro Express, PC Brand, Arche Legacy и Systems Integration Associates. Отметим тот факт, что модель SIA 386/32 построена на 25-МГц МП 80386, который используется на частоте 32,5 МГц и специально для этого охлаждается. Надежность работы машины при этом не снижается.

Особый интерес представляют изделия фирмы Compaq. При использовании МП с повышенной тактовой частотой она не просто дорабатывает старую модель машины, а проектирует ее заново для устранения диспропорций в быстродействии устройств. Чуть ли не уникальной является возможность одновременного подключения сопроцессоров 80387 и Weitek Abacus 3167 с обеспечением программного переключения между ними в процессе работы. Компьютер Compaq DESKPRO 386/33 может иметь внешнюю память до 2,6 Гбайт за счет двух внутренних НЖМД по 650 Мбайт и дополнительного подключения внешних НЖМД. Машина же Compaq SYSTEMPRO обладает возможностью расширения внешней памяти до 4,28 Гбайт. SYSTEMPRO создана специально для многопользовательских применений и для использования в качестве файл-сервера ЛВС. Подчеркнем, что она является многопроцессорной машиной, так как содержит два МП 80386. Возможна их замена на 33-МГц МП 80486. Очевидно, не случайно Compaq SYSTEMPRO признана лучшим компьютером 1990 г.

Средние базовые цены на ПЭВМ с МП 80386 таковы: 16 МГц — 2570 долл., 20 МГц — 3210 долл., 25 МГц — 3590 долл. и 33 МГц — 4421 долл.

ПЭВМ на базе микропроцессора 80486

Компьютеры, построенные на МП 80486, стали анонсироваться, начиная с июня 1989 г. Здесь опередила всех английская фирма Apricot. Компания же Compaq несколько запоздала в связи с тем, что проектирование ее машины (как обычно) велось полностью заново. Зато Compaq первой создала ПЭВМ с 50-МГц МП 80486. В 1991 г. появились и ПЭВМ с МП 80486SX.

Для ПЭВМ данного класса справедливы все замечания по совместимости, сделанные в предыдущем подпункте, только при совместимости с IBM PC AT говорят о классе AT-486.

Между машинами на базе МП 80386 и 80486 много общего. Главное отличие состоит в том, что на той же тактовой частоте ПЭВМ с МП 80486 работает примерно в два раза быстрее (в первую очередь благодаря более глубокой конвейеризации).

Основные технические характеристики анонсированных компьютеров с МП 80486 и 80486SX представлены в табл. 3.8. Если в ПЭВМ используется шина, отличная от MCA и EISA, то мы проставили «AT», имея в виду ее совместимость с шиной IBM PC AT. Все модели снабжены 16- и 32-разрядными гнездами для расширения ресурсов. Отметим, что контроллер НЖМД компьютера Cheetah Gold 425/D снабжен кэш-памятью емкостью 512 Кбайт, что косвенно увеличивает быстродействие НЖМД. Кэши ПЭВМ фирмы ALR способны не только временно хранить прочитанную из ОЗУ информацию, но и переписывать в память информацию, выработанную микропроцессором. Такой кэш называется «read and write».

Машины данного класса выполняются в обычном настольном варианте, в виде башни или как напольные стойки.

В ПЭВМ с МП 80486 используются те же ОС, что и в компьютерах на базе прибора 80386. Области применения также аналогичны. Однако больший упор делается на *многопользовательские* приложения и использование в качестве *файл-серверов* ЛВС.

В табл. 3.8 не отражен тот факт, что изделие Apricot VX FT Server, в зависимости от комплектации, служит для сетевых применений (модели ряда 400/xx) или для многопользовательских применений в расчете на 64, 96 или 128 пользователей (модели ряда 800/xx).

Отличительной особенностью модели Compaq DESKPRO 486/50Z является использование скоростного 50-МГц сопроцессора, совместимого с прибором 80387. Внешнюю память этой машины можно нарастить до 20 Гбайт.

В дополнение к сведениям, представленным в табл. 3.8, отметим также, что фирма Advanced Logic Research разработала компьютер Multiaccess Series 3000, который может содержать до шести МП 80486. Это серьезная альтернатива модели Compaq SYSTEMPRO.

Средняя базовая цена ПЭВМ с МП 80486-25 составляет 5953долл., с МП 80486-33 — 7599 долл., а с МП 80486SX-20 — 4546 долл.

Таблица 3.8

Технические характеристики стационарных ПЭВМ на базе МП 80486

Производитель: фирма, страна	Модель	МП	Факультативный сопроцессор	СШ	Емкость кэш-памяти, Кбайт	Емкость ОЗУ, Мбайт	Количество НГМД х емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер	Исполнение	Цена, долл.
1	2	3	4	5	6	7	8	9	10	11	12
Acer Technology, Тайвань	AcerPower 486SX	80486SX-20	н/д	AT	-	до 14	н/д	н/д	Super VGA	н/д	4360
	AcerPower 500	80486SX-20	н/д	EISA	-	до 14	н/д	н/д	Super VGA	Н	н/д
	AcerFrame 1000	80486SX-20	н/д	EISA	-	до 14	н/д	н/д	Super VGA	Б	н/д
ACMA Computers	ACMA 486/33 Engineering	80486-33/25/20/16 12/8	Weitek 4167	AT	128	1..24	1x1,2М-133& 1x1,44М-89	80, 105, 150:18, 210, 338, 676	VGA, Super VGA	Б	от 5985
Advanced Logic Research, США	ALR Business VEISA 486SX	80486SX-20	-	EISA	н/д	н/д	н/д	н/д	Super VGA	н/д	4644
	ALR MPS 486SX	80486SX-20	-	MCA	н/д	н/д	н/д	н/д	Super VGA	н/д	н/д
	ALR PowerCache 4 M130	80486-25	-	MCA	128	н/д	1x1,44М-89	130..260	VGA [8514/A]	Н	от 9990
	ALR PowerCache 4 M150	80486-25	-	MCA	128	н/д	1x1,44М-89	150	VGA [8514/A]	С	11490
	ALR PowerCache 4 M350	80486-25	-	MCA	128	н/д	1x1,44М-89	350	VGA [8514/A]	С	н/д
	ALR PowerCache 4 M650	80486-25	-	MCA	128	н/д	1x1,44М-89	650	VGA [8514/A]	С	н/д
AGC Electronics, Тайвань	AGC Powerlay 486 EX	80486-25	-	EISA	-	4	по выбору	по выбору	VGA	Б	н/д
Apricot, Великобритания	Apricot VX FT Server	80486-25	-	MCA	64, 128	4, 8, 16..н/д	1x1,44М-89	157:16, 347:16, 647:16, 1047:16 [всего до 5 Гбайт]	VGA	С	18000-40000
Aquarius Systems Integral, СНГ	ASI-486/25	80486-25/8	н/д	AT	н/д	4..24	1x1,2М-133& 1x1,44М-89	80:19	VGA	Б	н/д
Arche Technologies	Arche Legacy ProFile 486-33	80486-33/8	Weitek 4167	EISA	256	4..64	по выбору	-, 180, 338:16	-, Super VGA	Б	от 9455

Таблица 3.8 (окончание)

1	2	3	4	5	6	7	8	9	10	11	12
AST Research, США	AST Premium II 486SX/20	80486SX-20	н/д	н/д	н/д	до 80	н/д	н/д	VGA, Super VGA	н/д	от 4390
Cheetah International, США	Cheetah Gold 425/D	80486-25	-	AT	-	16..н/д	1x1,2M-133& 1x1,44M-89	383	Super VGA	Б	9995
Club American Technologies, США	Club AT Hawk II	80486-25	-	AT	128	4..н/д	по выбору	85	VGA	Б	4995
	Club AT Hawk IIITi	80486-33	Weitek 4167	AT	64, 256	8..16	1x1,2M-133	338:16	Super VGA	Б	6100
Compaq Computer, США	Compaq DESKPRO 486/25	80486-25	Weitek 4167	EISA	128	4..100	по выбору	320, 650 [всего до 2,6 Гбайт]	VGA	Н	н/д
	Compaq DESKPRO 486/33L	80486-33	Weitek 4167	EISA	н/д	н/д	н/д	н/д	н/д	н/д	13999-19499
	Compaq DESKPRO 486/50Z	80486-50	80387-50	EISA	8+ 256	н/д	н/д	120:19, 340:18, 510:12	Enhanced VGA	н/д	11299-13499
Computer Market Place	Ultra 486-33	80486-33	-	AT	128	8..16	1x1,2M-133, 1x1,2M-133& 1x1,44M-89	211:15	VGA, Super VGA	Б	4628-6499
Everex Computer Systems, США	Everex Tempo 486SX/20	80486SX-20	н/д	AT	128	н/д	н/д	н/д	н/д	н/д	1187
	Everex Step 486SX/20	80486SX-20	н/д	н/д	н/д	н/д	н/д	н/д	н/д	н/д	н/д
Fortron/Source	Fortron NetSet 1331	80486-33	Weitek 4167	AT	64	8..32	1x1,2M-133	207	Super VGA	Н	от 5495
Gateway 2000, США	Gateway 486/25	80486-25	-	AT	-	2..н/д	1x1,2M-133& 1x1,44M-89	160	VGA	н/д	н/д
Global Computer Network, США	Global/486	80486-25	-	н/д	-	1	1x1,2M-133& 1x1,44M-89	120	Super VGA	н/д	7620
Hewlett-Packard, США	HP Vectra 486 PC	80486-25	-	EISA	-	2..32	1x1,2M-133	150:15, 330:15, 670:15	Super VGA	н/д	13999-16999
	HP Vectra 486/33	80486-33	Weitek 4167	EISA	128	4..64	по выбору	150, 440, 1000	Super VGA	Б	н/д
HM Systems, Великобритания	Minstrell 486	80486-25	-	AT	32	до 24	1x1,44M-89	40, 120, 240:15	VGA	Н	н/д
Normerel Systemes, Франция	ATP486	80486-25	-	AT	32	1..16	1x1,2M-133& 1x1,44M-89	100:19 [630]	VGA [Super VGA]	Б	н/д
	NS90	80486-25	-	MCA	64	1..16	1x1,44M-89	100:19 [630]	VGA [Super VGA]	Б	н/д
Northgate Computer Systems	Northgate Elegance 486/33	80486-33	-	AT	64	8..16	1x1,2M-133& 1x1,44M-89	200:15	Super VGA	Б	от 7195
Standard Computer	Standard 486/33	80486-33	Weitek 4167	AT	128	8..48	1x1,2M-133& 1x1,44M-89	200	Super VGA	Б	от 6315
Zenith Data Systems, США	Zenith Z-486SX/25E	80486SX-25	Weitek 4167	EISA	н/д	1..32	н/д	200	TIGA (аналог 8514/A)	н/д	от 4019

3.5.2. Переносные ПЭВМ

Переносные (portable) отличаются от стационарных ПЭВМ следующим:
— меньшими габаритами и массой;
— конструкцией, допускающей транспортировку компьютеров человеком;
— использованием наряду с дисплеями на базе электронно-лучевой трубки плазменных, жидкокристаллических и электролюминесцентных дисплеев (часто — монохромных);
— возможностью питания от аккумуляторных батарей (только для некоторых моделей);
— отсутствием (за редким исключением) в комплекте машины принтера;
— как правило, меньшим числом клавиш на клавиатуре.

Переносные ПЭВМ действительно могут переноситься (и перевозиться) человеком, но используются они аналогично стационарным (на столе). Отсюда и название «переносные».

Первый переносной компьютер, что уже отмечалось, выпущен фирмой Compaq. Сейчас пользуется популярностью как продукция этой компании, так и фирм Toshiba (Япония), Dolch American Instruments, Micronics и др.

Некоторое время переносные ПЭВМ были единственной альтернативой стационарным. Но затем появились наколенные машины почти с такими же характеристиками, в связи с чем интерес к наколенным ПЭВМ в значительной мере снизился. Грань между переносными и портативными ПЭВМ в определенной степени условна. Поэтому искушенный читатель может уличить нас в не совсем корректном разделении компьютеров на группы (автор заранее приносит свои извинения).

В группах портативных ПЭВМ мы не будем явно выделять различные классы совместимости компьютеров, считая достаточными сведения, изложенные в п. 3.5.1. Тип МП будет однозначно характеризовать класс совместимости ПЭВМ.

Основные технические характеристики пользующихся популярностью переносных ПЭВМ приведены в табл. 3.9. Они приближаются к характеристикам настольных компьютеров. Дополнительно к уже использованным введены следующие обозначения и соглашения:

1) ЭЛТ, ЖК, ЖКЗ, П и Л обозначают дисплей на базе электронно-лучевой трубки, жидкокристаллический дисплей, жидкокристаллический дисплей с задней подсветкой, плазменный дисплей и электролюминесцентный дисплей соответственно;

2) если дисплей монохромный, то этот факт никак не отображен; если же он цветной, то проставлена через тире буква «ц».

Отметим, что фирма PC BRAND предлагает переносные варианты своих моделей 286/12, 286/20, 386/SX-16, 386/20 и 386/25 как с плазменными, так и с жидкокристаллическими дисплеями с задней подсветкой. Их характеристики соответствуют стационарным моделям и поэтому данные компьютеры не представлены в табл. 3.9. Хотелось бы подчеркнуть, что эта фирма выпускает достаточно качественные ПЭВМ по сравнительно низким ценам. Так, переносная модель 386/25 с 200-Мбайт НЖМД и плазменным дисплеем стоит 4910 долл., а при наличии жидкокристаллического дисплея — 4055 долл.

Кроме того, в табл. 3.9 не отражены следующие факты:
— ПЭВМ Compaq Portable 386 имеет кэш для обмена с жестким диском;
— машины фирмы Dolch American Instruments снабжены кэш-памятью емкостью 64 Кбайт; поэтому Dolch-P.A.C. 386-20C производительнее, чем Compaq Portable 386;
— все указанные модели способны питаться только от сети;
— ПЭВМ MP400 486/25 имеет шину EISA, а Darius ProPortable — MCA;
— в последнюю из упомянутых моделей встроен контроллер кэш-памяти, которая может быть подключена факультативно;
— все модели имеют как минимум по одному адаптеру параллельного и последовательного интерфейсов;
— некоторые модели дополнительно снабжены адаптером игрового интерфейса;
— большинство моделей имеет гнезда расширения, но их число ограничено.

Цены на переносные ПЭВМ выше цен на стационарные ПЭВМ с теми же характеристиками. Обратим внимание на факт наличия переносного компьютера на базе МП 80486 (MP400 486/25).

3.5.3. Наколенные ПЭВМ

Наколенные (laptop) ПЭВМ компактнее, легче переносных и конструктивно выполнены таким образом, что возможно их использование на коленях человека. Именно это является основным признаком, отличающим их от переносных ПЭВМ. Чтобы обеспечить возможность наколенной работы, laptop-компьютеры оформлены обычно в виде дипломата: раскрыв его, можно приступать к работе. На внутренней стороне одной половинки такого дипломата находится экран, а другой — клавиатура. Есть место и для дисководов. Конечно, с учетом сказанного, не может быть и речи об использовании дисплея на базе электронно-лучевой трубки. Из-за компактности и низкого энергопотребления наибольшее применение находят жидкокристаллические дисплеи, хотя возможны и другие подходы к решению проблемы отображения информации. Большинство наколенных ПЭВМ могут питаться от свинцовых или никелькадмиевых аккумуляторных батарей. Последние используются чаще из-за меньшей массы. Клавиатуры, как правило, имеют ограниченное число клавиш.

Еще 4 — 5 лет назад наколенные машины были лишь жалкими подобиями стационарных ПЭВМ. Теперь же характеристики многих из них не уступают достаточно мощным настольным компьютерам. Начинают применяться и цветные дисплеи взамен монохромных.

Таблица 3.9

Технические характеристики переносных IBM-совместимых ПЭВМ

Производитель: фирма, страна	Модель	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Количество НГМД х емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер	Дисплей	Масса, кг	Габариты, мм	Цена, долл.
1	2	3	4	5	6	7	8	9	10	11	12
BEST Computer, США	BEST 286 LCD Portable	80286-12(0)	–	640K..4M	1x1,2M-133	20, 40:28	CGA	ЖК	10	406x229x152	1395
	BEST 286 CRT Portable	80286-12(0)	–	640K..4M	1x1,2M-133	20, 40:28	CGA	ЭЛТ	н/д	н/д	1195
	BEST 286 CRT EGA Mono	80286-12(0)	–	640K..4M	1x1,2M-133	20, 40:28	EGA	ЭЛТ	н/д	н/д	1295
	BEST 286 gas plasma mini Portable	80286-12(0)	–	640K	1x1,44M-89	40:28	EGA 720x400	П	7,26	406x229x140	1945
	BEST 286 Super Portable	80286-12(0)	–	640K..4M	1x1,44M-89	20, 40:28	EGA	ЖК	н/д	н/д	н/д
Compaq Computer, США	Compaq Portable	8088-4,77	н/д	н/д	н/д	н/д	н/д	н/д	12,7	н/д	н/д
	Compaq Portable II	80286-8	н/д	256K..1,5M	2x360K-133	[10]	CGA	ЭЛТ	10,7	188x442x348	н/д
	Compaq Portable 286	80286-12	н/д	до 2,6M	1x1,2M-133	[20]	н/д	н/д	н/д	н/д	н/д
	Compaq Portable III	80286-12	н/д	640K..6,6M	1x1,2M-133	[20, 40]	CGA	П	9,07	245x442x195	4999
	Compaq Portable 386	80386-20(1)	80387	1M..10M	1x1,2M-133	40, 100	EGA	П	9,93	406x249x198	7999
Dolch American Instruments, США	Dolch-P.A.C. 386-20C	80386-20(0)	80387	1M..2M	1x1,2M-133	20[40]	CGA	Л	н/д	н/д	н/д
	Dolch-P.A.C. 386-25	80386-25	80387	4M..8M	1x1,2M-133	40, 80, 100	CGA[EGA]	Л	9,34	413x260x216	9495
Kiss Computer, США	PX386	80386-25/20/16	80387	2M..8M	1x1,2M-133, 1x1,44M-89	20..383	CGA[VGA]	Л	9,07	н/д	от 3995
	Lyte-Byte 5400	80386SX-16	н/д	1M..4M	н/д	40	VGA	П	н/д	н/д	от 2999
Micro Express, США	ME Roadrunner Plus	80386-20	80387	1M..10M	1x1,2M-133	40[80]	EGA	ЖК	н/д	н/д	3399
	ME Regal II	80386-20	80387	1M..10M	1x1,2M-133	40[80]	EGA	П	10,34	413x260x216	3999

Таблица 3.9 (окончание)

1	2	3	4	5	6	7	8	9	10	11	12
Micronics Computers, США	MP400 386SX MP400 386/20	80386SX-16 80386-20	н/д н/д	1M..8M 2M..8M	н/д по выбору	н/д н/д	н/д VGA	н/д ЭЛТ	н/д н/д	н/д 114x 381x 406	3800 4500
	MP400 486/25	80486-25	-	2M..16M	по выбору	40[100]	VGA	ЖК, ЖК-ц	н/д	н/д	7500-10000
Mission Cyrus Group, США	Darius ProPortable	80386-25	80387, Weitek 3167	1M..16M	по выбору	40, 100, 200	VGA	ЭЛТ	9,07	н/д	10000
NEC Information Systems, США (филиал NEC, Япония)	NEC Powermate Portable SX	80386SX-16	н/д	2M..10M	1x1,44M-89	40, 100	VGA	П	9,93	394x 287x 197	6595
Sharp Electronics, Япония	Sharp PC-7200	80286-10	н/д	640K	1x1,44M-89	20	н/д	н/д	н/д	н/д	н/д
Systems Manufacturing Technology, США	Wedge Microstation	80386	н/д	4M	1x1,2M-133, 2x1,2M-133	20, 40	EGA	ЭЛТ-ц	н/д	430x 210x 400	от 2595
Toshiba America, США (филиал Toshiba, Япония)	Toshiba T5100	80386-16	80387	2M..4M	1x1,44M-89	40:29, 100:25	EGA	П	6,62	310x 361x 89	7199
	Toshiba T5200	80386-20	80387	2M..8M	1x1,44M-89	40:29, 100:25	VGA	П	8,48	422x 396x 99	9499
	Toshiba T5200C	80386-20	80387	2M..8M	1x1,44M-89	200:16	VGA	ЖК-ц	н/д	н/д	9499

В общем, портативные ПЭВМ сегодня совершенствуются опережающими темпами. Они уже стали основными компьютерами для тех пользователей, которым по роду своей деятельности приходится общаться с ПЭВМ в дороге.

Лидерство в производстве машин данной группы принадлежит фирмам Compaq, NEC, Zenith и Toshiba. Есть и ряд компаний-новичков, предлагающих заслуживающие внимания компьютеры. Фирма же IBM лишь недавно представила образец наколенной машины.

Наиболее передовой технологией производства laptop-ПЭВМ по праву считается японская.

Основные технические характеристики известных моделей наколенных ПЭВМ сведены в табл. 3.10. В ней МП 80C86/88 и 80C286 обозначают КМОП-варианты приборов 8086/88 и 80286 соответственно, характеризующиеся пониженным энергопотреблением. Символ «*» в колонке «Время автономной работы» использован тогда, когда известно, что автономная работа возможна, но нет данных о ее продолжительности; сокращение же «н/д» в той же колонке говорит о том, что возможность автономной работы нам неизвестна; символ «—» указывает на отсутствие такой возможности. Значения массы ПЭВМ весьма относительны, так как нам не всегда было известно, включается ли в нее вес НЖМД и аккумуляторных батарей.

Сделаем несколько замечаний по моделям, представленным в табл. 3.10. ПЭВМ Copam 286LT снабжена малогабаритным НЖМД с диаметром диска, равным 63 мм. Ряд компьютеров, в частности, GRiDCase 1520, работают под управлением ОС MS-DOS, записанной в ПЗУ. GRidCase 1520 содержит в ПЗУ и систему OS/2. ПЭВМ MYODA LT5200CD имеет 32-Кбайт кэш-память, что обеспечивает отсутствие тактов ожидания МП. Компьютер Sharp PC-4521 совместим с IBM PC XT. Базовый комплект Techpower T-2600 включает сопроцессор. Компьютер Epson Equity LT-286e снабжен НЖМД со съемным диском. Компания Zenith первой приступила к выпуску laptop'ов с МП 80486(SX).

Наколенные ПЭВМ имеют, как правило, по одному адаптеру параллельного и последовательного интерфейсов, а также порт для подключения внешнего дисплея соответствующего стандарта. Не редкость также наличие порта для подключения внешнего НГМД. Число гнезд расширения — минимально необходимое.

Стоимость наколенных ПЭВМ существенно выше настольных моделей с теми же характеристиками.

Установить принадлежность ПЭВМ к группе наколенных машин без явного указания на это можно по ее толщине (размер по одному из измерений — около 10 см), сравнительно небольшой массе, а часто и по наименованию модели, если в нем указано LT (от LapTop).

3.5.4. Блокнотные ПЭВМ

ПЭВМ-блокноты (notebook или hand-held) недавно выделились из класса портативных ПЭВМ в силу их дальнейшей миниатюризации. Размеры блокнотной ПЭВМ действительно соответствуют блокноту или, по крайней мере, книге. На лицевой панели такого компьютера расположены небольшой дисплей (жидкокристаллический) и усеченная клавиатура, напоминающая клавиатуру калькулятора. Однако возможно (и часто встречается) исполнение в виде дипломата. Поэтому-то граница между наколенными и блокнотными ПЭВМ размыта, о чем мы уже говорили. НМД в ПЭВМ-блокнотах поначалу не использовались из-за неудовлетворительных габаритов дисководов. Роль сменной внешней памяти при этом выполняли карты ЭСППЗУ или запоминающих устройств, основанных на новых физических принципах. Сейчас же многие модели блокнотных ПЭВМ оборудуются не только НГМД, но и НЖМД. Возможно подключение и внешних НГМД. Другие ПУ могут подсоединяться через имеющиеся адаптеры интерфейсов. Возможности современных ПЭВМ-блокнотов в настоящее время вполне приемлемые. При работе такие компьютеры держат в руке (отсюда hand-held).

Развитие сектора блокнотных ПЭВМ в настоящее время идет наиболее динамично.

Первой ПЭВМ блокнотного типа с НГМД была модель Produce фирмы Sony (Япония), какая-либо другая информация о которой у нас отсутствует.

Характерной чертой ПЭВМ-блокнотов является малая масса, составляющая около 2 — 4 кг.

Основные технические характеристики некоторых моделей ПЭВМ-блокнотов сведены в табл. 3.11. Запись «CGA 640x400» в столбце «Дисплейный адаптер» обозначает адаптер стандарта CGA с двойным разрешением.

Наиболее совершенны модели фирм NEC, Sharp, Toshiba и Zenith.

ОС для управления работой блокнотных ПЭВМ (MS-DOS) обычно записывается в ПЗУ. Машины NEC UltraLite, RHS-88 и Psion MC-600 совместимы с IBM PC XT, а PC 286 (Seiko Epson) — с IBM PC AT. Совместимость остальных моделей определяется МП. Известно, что компьютеры фирмы Sharp, а также ПЭВМ SHERRY NB-12 компании Pet Computers Service могут снабжаться сопроцессором плавающей точки. Обратите внимание на то, что машина Sharp PC-6881 оборудована цветным монитором. Эта модель, а также ПЭВМ Sharp PC-6741 и PC-6781 имеют 16-Кбайт кэш-память.

Стоимость блокнотных ПЭВМ выше, чем цена аналогичных им по техническим характеристикам наколенных компьютеров.

3.5.5. Карманные ПЭВМ

Карманные компьютеры (pocket или palmtop) конструктивно оформляются как нераскрывающиеся блокнотные ПЭВМ, но имеют меньшие габариты и массу, что обеспечивает их хранение в кармане и возможность работы на ладони. Вычислительная мощность карманных ПЭВМ пока ограниченна.

Таблица 3.10

Технические характеристики наколенных IBM-совместимых ПЭВМ

Производитель: фирма, страна	Модель	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Количество НГМД x емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт :среднее время доступа, мс	Дисплейный адаптер	Дисплей	Масса, кг	Габариты, мм	Время автономной работы, час.мин	Цена, долл.
1	2	3	4	5	6	7	8	9	10	11	12	13
Acer Technologies, Тайвань	Acer 1100LX	80386SX-16	80387SX	до 5М	1x1,44М-89	40	VGA	ЖКЗ	7,25	н/д	3.00	н/д
Aquarius Systems Integral, СНГ	ASI-Laptop 168	80286-16/8	н/д	1М..5М	1x1,44М-89	20:25 [40: 25]	VGA	ЖКЗ	н/д	н/д	н/д	3500
Bondwell Industrial, США	Bondwell B2000	80C88-8	-	256К..640К	1x720К-89	-	CGA	ЖК	3,63	н/д	*	995
	Bondwell BW8 Turbo	80C88-н/д	-	1М	н/д	-	CGA	ЖКЗ	5,99	н/д	н/д	1595
	Bondwell B300	80C286-10	-	1М..1,5М	1x1,44М-89	[20]	CGA	ЖК	6,80	н/д	0.59	2995
Compaq Computer, США	Compaq LTE	80C86-9,54	н/д	640К..1М	1x1,44М-89	[20]	CGA	ЖКЗ	н/д	н/д	н/д	2399-2999
	Compaq LTE/286	80C286-12/8	80C287-12	640К..2,6М	1x1,44М-89	[20, 40]	CGA	ЖКЗ	3,50-5,00	203x279x51	3.30	3899-4999
	Compaq SLT/286	80C286-12	80C287	640К..3,6М	1x1,44М-89	20, 40	VGA	ЖКЗ	6,35	н/д	2.25	5399-5999
	Compaq LTE 386s/20	80386SX-20	80387SX	н/д	1x1,44М-89	60	н/д	н/д	н/д	н/д	н/д	6999
Copam USA, США	Copam 286LT	80286-12	80287	1М..4М	1x1,44М-89	20:28 [40, 100]	VGA	ЖКЗ	5,44	н/д	2.00	от 2000
	Copam 386SXLT	80386SX-16	80387SX	1М..4М	1x1,44М-89	20:28 [40, 100]	VGA	ЖКЗ	5,44	н/д	2.00	от 3000
Data General, США	Data General Model 2T	80C88-8/4,77	н/д	512К	1x720К-89	10	CGA	ЖК	н/д	н/д	н/д	1695
Dell Computer, США	Dell System 316LT	80386SX-16	80387SX	1М..8М	1x1,44М-89	20, 40	VGA	ЖК	6,80	н/д	2.00	3499-3999
Dauphin	DAUPHIN LapPro-286	80C286-10 при работе на 12/8	80287	до 4М	1x1,44М-89	40	CGA [HGA]	ЖК	8,80	н/д	2.00	3753
	DAUPHIN LapPro 386SX	80386SX-16	80387SX	2М..4М	1x1,44М-89	40	EGA	ЖКЗ	7,50	н/д	*	от 4995

Таблица 3.10 (продолжение)

1	2	3	4	5	6	7	8	9	10	11	12	13
GRiD Systems, США	Tempest GRiDCase 1307	80C86	н/д	640K	н/д	–	CGA	ЖКЗ	6,80	н/д	*	995
	GRiDLite XL	80C86	н/д	128K..1M	н/д	20	CGA	ЖК	4,31	н/д	*	1950
	GRiDCase 140XT	V20-8 (аналог 8088)	8087	768K	н/д	20	CGA	ЖКЗ	5,44	н/д	3.00	2595-3050
	GRiDCase 1520	80286-10	80287	1M..4M	1x1,44M-89	20 [40, 100]	CGA	ЖКЗ [П]	5,44	н/д	1.08	3495
	GRiDCase 1530	80386-12,5	н/д	1M..8M	1x1,44M-89	20 [40, 100]	CGA	ЖКЗ [П]	5,44	292x 301x 58	*	от 4695
	GRiDCase 1535 EXP	80386-12,5	н/д	1M..8M	1x1,44M-89	40 [100]	CGA	ЖКЗ [П]	6,94	292x 384x 64	*	6995
	GRiD 1450SX	80386-16	н/д	1M..5M	1x1,44M-89	20, 40	VGA	ЖК	3,60	н/д	– [2, 4]	от 4995
Hewlett-Packard, США	Portable Plus	80C86	н/д	512K..2,5M	н/д	–	CGA	ЖК	4,08	н/д	*	2304
	Portable LS/12	80286-12	80287	640K..1,6M	1x1,44M-89	20 [40]	CGA	ЖКЗ	4,76	310x 310x 79	3.17	4879
Ing. C. Olivetti & Co, Италия	Olivetti M-22	80C88-4,77	н/д	256K..1M	1x360K-133	[н/д]	CGA	ЖК	7,30	н/д	н/д	н/д
	Olivetti M-316	80386SX-16	80387SX	1M..4M	1x1,44M-89	20, 40	VGA	ЖК	6,70	99x 230x 360	2.50	н/д
Mitsubishi Electronic America, США (филиал Mitsubishi, Япония)	Mitsubishi MP-286L	80286-12/8	80287	640K..2,6M	1x1,44M-89 [2x1,44M-89]	20, 40	CGA [EGA]	ЖК	н/д	н/д	–	от 3995
NEC, Япония	NEC N5200	80386SX-16	н/д	2M	1x1,44M-89	40, 110	VGA	ЖК-ц	н/д	н/д	н/д	5300-10400
NEC Home Electronics, США (филиал NEC, Япония)	NEC ProSpeed 286	80286-16	80287	1M..5M	1x1,44M-89	20 [100]	EGA [VGA]	ЖКЗ	10,20	н/д	2.30	4999
	NEC ProSpeed 386SX	80386SX-16	н/д	1M..16M	1x1,44M-89	40, 100	VGA	ЖКЗ	6,80	323x 87x 373	3.00	н/д
	NEC ProSpeed CSX	80386SX-16	н/д	н/д	1x1,44M-89	н/д	VGA	ЖК-ц	7,20	н/д	н/д	н/д
	NEC ProSpeed 306	80386-16	80387	2M..10M	1x1,44M-89	40, 100	VGA	ЖКЗ	н/д	403x 99x 394	*	7699
	NEC ProSpeed 486SX/C	80486SX-20	н/д	2M..20M	1x1,44M-89	120	Super VGA	н/д	7,60	н/д	н/д	8999
Ogivar Technologies, Канада	Ogivar 286 Laptop	80286-12,5	80287	640K..4,6M	1x720K-89	40 [20]	EGA	П	6,35	330x 358x 104	–	4995
Packard Bell	PB286LP	80C286-12/8	80287	до 5M	1x1,44M-89	40	CGA	ЖК	7,50	н/д	4.30	4827

Таблица 3.10 (окончание)

1	2	3	4	5	6	7	8	9	10	11	12	13
PAO-KU International, США	MYODA LT3200 MYODA LT3500 MYODA LT5200MV MYODA LT5200CD	80286-12(1) 80286-12(0) 80286-12(0) 80386-25	80287 80287 80287 80387	640K..2,6M 1M..4M 1M..4M 1M..8M	1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89	40:28 40:28 40:28 40:28	CGA EGA VGA VGA	П П П П	н/д н/д н/д н/д	н/д н/д н/д н/д	н/д н/д н/д н/д	1899 2299 2599 3599
Seiko Epson, Япония	Epson Equity LT-286e	80C286-12/8	80C287	до 2M	1x1,44M-89	40	EGA	ЖК	11,10	н/д	3.10	5397
Selsys, США	QUEST 386/20LS	80386SX-16 с возможностью замены на 80306SX-20	н/д	н/д	н/д	н/д	VGA	ЖКЗ	2,20	324x 260x 32	"	н/д
Sharp Electronics, Япония	Sharp PC-4521 Sharp PC-5541 Sharp Multicolor 386 Sharp PC-8501	н/д-8 80C286-12/8/6 80386-20 80386-20/8	н/д 80287 н/д 80387-20	640K до 3,6M 2M..8M 2M..10M	2x720K-89 1x1,44M-89 1x1,44M-89 1x1,44M-89	20 40 40:20 100	н/д VGA VGA VGA	П ЖК ЖК-ц ЖК-ц	н/д 9,00 11,60 6,90	н/д н/д н/д 318x 399x 94	н/д 3.30 н/д н/д	н/д 6143 н/д н/д
Tandy, США	Tandy 1400LT	8088-7,16	-	768K	2x720K-89	[20]	MDA	ЖК	н/д	н/д	н/д	н/д
Techpower, Тайвань	Techpower T-2600	80286-16	80287-10	н/д	1x1,44M-89	20, 40	CGA	П	н/д	н/д	н/д	н/д
Top-Link Computer, Тайвань	Top-Link EGA Laptop	80286-16/10	80287	н/д	1x1,44M-89	н/д	EGA	П	н/д	н/д	н/д	н/д
Toshiba, Япония	J3100 SS	80C86-8	н/д	640K..3,5M	2 шт. 89 мм по выбору	н/д	н/д	н/д	2,70	310x 254x 44	2.30	1394
Toshiba America, США (филиал Toshiba, Япония)	Toshiba T1200F Toshiba T1200FB Toshiba T1200H Toshiba T1200HB Toshiba T1200XE Toshiba T3100 Toshiba T3100e Toshiba T1600 Toshiba T3200 Toshiba T3100SX Toshiba T3200SX Toshiba T3200SXC	80C86-9,54 80C86-9,54 80C86-9,54/4,77 80C86-9,54/4,77 80286-12 80286-8/4 80286-12 80286-12/6 80286-12 80306SX-16/8 80386-16 80386-20	н/д н/д н/д 8087 80287 80287 80287 80C287-12 80287 80387SX 80387SX 80387SX	1M..2M 1M..2M 1M..2M 1M..2M 1M..5M 1M..5M 1M..5M 1M..5M 1M..4M 1M..н/д 1M..13M 1M..13M	1x720K-899 2x720K-899 1x720K-899 1x720K-899 1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89 1x1,44M-89	20 20 20 20 40 20 20 20:27 40 40 40:25 120: 19 120: 19	CGA CGA CGA CGA 640x 100 EGA EGA EGA EGA VGA VGA VGA	ЖК ЖКЗ ЖК ЖКЗ ЖКЗ П П ЖКЗ П ЖК П ЖК-ц	4,49 н/д 4,32 н/д 3,64 6,80 5,99 5,44 8,62 7,26 н/д 7,80	н/д н/д н/д н/д н/д н/д н/д 315x 325x 84 н/д н/д н/д 112x 371x 396	" " " " " - - 3.00 - н/д - -	2099 н/д 2799 3699 н/д 5000 4199 4999 5299 н/д 5199 7249
Zenith Data Systems, США	Zenith SupersPort Zenith SupersPort 286 Zenith SupersPort SX Zenith TurbosPort 386 Zenith SupersPort 486SX Zenith SupersPort 486	80C88 80286-12/6 80306SX-16/8 80386-16/6 (0) 80486SX-20 80486-25	н/д 80287 н/д 80387 н/д -	640K 1M..2M 1M..н/д 2M..3M 4M 4M	н/д 1x1,44M-89 1x1,44M-89 1x1,44M-89 н/д н/д	[20, 40] 20, 40 40:25 40 н/д н/д	CGA CGA VGA VGA н/д н/д	ЖКЗ ЖКЗ ЖК ЖКЗ н/д н/д	4,76 4,76 5,44 8,48 н/д н/д	н/д 310x 310x 81 н/д 337x 121x 375 н/д н/д	" 4.40 3.00 2.00 н/д н/д	от 2399 3599 н/д 7999 7499 8899

Таблица 3.11

Технические характеристики IBM-совместимых блокнотных ПЭВМ

Производитель: фирма, страна	Модель	МП	Емкость ОЗУ, байт	Внешняя сменная память	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер	Дисплей	Масса, кг	Габариты, мм	Время автономной работы, час.мин	Цена, долл.
1	2	3	4	5	6	7	8	9	10	11	12
Advanced Logic Research, США	Venture 16SX Notebook PC	80386SX-16	1M..5M	НГМД 1x1,44M-89	н/д	н/д	ЖКЗ	3,15	300x218x50	3.00	2795
Agilis, США	Agilis System	80386-20	4M	н/д	20	н/д	н/д	н/д	книга	н/д	12000
AST Research, США	AST Premium Exec 286/12	80286-12 [80386SX-16]	1M..8M	НГМД 1x1,44M-89	до 40	VGA	ЖКЗ	2,95	н/д	*	от 2495
Dynabook Technologies, США	Dynabook 286	80C286-16	н/д	н/д	20, 40	н/д	ЖКЗ	3,95	толщина - 36 мм	4.00	5195-5795
GRiD Systems, США	GRiDPad	80C86-10	1M	2 карты по 256 или 512 Кбайт	[20:29]	CGA 640x400	ЖКЗ	2,04	318x235x38	*	от 2370
	GRiD 1720	80C286-16	1M..5M	н/д	20	VGA	ЖК	2,95	н/д	3.00	3495
Husky Computers, Великобритания	Hauk	8088	640K	н/д	н/д	н/д	ЖК	н/д	н/д	*	1695
Ing. C. Olivetti & Co, Италия	Olivetti Notebook A12	80286-12	1M..5M	НГМД 1x1,44M-89	20	н/д	ЖК	2,90	н/д	*	н/д
	Olivetti Notebook A16	80286-16	2M..6M	НГМД 1x1,44M-89	40	н/д	ЖК	2,90	н/д	*	н/д
	Olivetti Notebook S20	80386SX-20	2M..6M	НГМД 1x1,44M-89	60	н/д	ЖК	2,90	н/д	*	н/д
Leading Edge Products	N3/SX	80386SX-16	1M	НГМД 1x1,44M-89	20	н/д	ЖК	н/д	н/д	*	2595
	N3/SX	80386SX-20	1M	НГМД 1x1,44M-89	30, 60	н/д	ЖК	н/д	н/д	*	2895-3195
NEC Home Electronics, США (филиал NEC, Япония)	NEC UltraLite	V30-9,83	640K..896K	1 или 2 Мбайт	н/д	CGA	ЖКЗ	2,00	317x212x36	*	2999-4526
	NEC UltraLite 286F	80286	1M..5M	НГМД 1x1,44M-89	20	н/д	ЖКЗ	3,08	н/д	*	2999
Packard Bell	PB286NB	80C286-12/6	н/д	НГМД 1x1,44M-89	20	VGA	ЖК	н/д	н/д	*	н/д
Paravant Computer Systems, США	RHC-88	V40	512K..1,5M	н/д	-	256x128	ЖК	2,00	235x160x65	*	3995

Таблица 3.11 (окончание)

1	2	3	4	5	6	7	8	9	10	11	12
Pet Computers Service, Сингапур	SHERRY NB-12	80C286-12/6(0)	1M..4M	НГМД 1x1,44M-89	20:25,6	VGA	ЖКЭ	3,10	308x272x48	1.40	1850
Psion, США	Psion MC-600	н/д	768K	1 Мбайт	–	н/д	н/д	2,00	стопка бумаги	3.00	2499
Seiko Epson, Япония	Epson PC 286	N30-10	640K..1,6M	2 карты по 128 или 640 Кбайт	н/д	н/д	н/д	2,20	315x235x35	3.00	3225
Sharp Electronics, Япония	Sharp PC-6220	80C286-12/7,16/6 (0&1)	1M..3M	НГМД 1x1,44M-89	20:23	VGA	ЖКЭ	1,95	279x216x35	1.34	3086
	Sharp PC-6240	80C286-12/7,16/6 (0&1)	1M..3M	НГМД 1x1,44M-89	40:23	VGA	ЖКЭ	1,95	279x216x35	1.14	3424
	Sharp PC-6521	80286-12	1M..4M	НГМД 1x1,44M-89	20:23	VGA	ЖКЭ	2,90	н/д	н/д	н/д
	Sharp PC-6541	80286-12	1M..4M	НГМД 1x1,44M-89	40:23	VGA	ЖКЭ	2,90	н/д	н/д	н/д
	Sharp PC-6741	80386SL-20	2M..8M	НГМД 1x1,44M-89	40:18,5	VGA	ЖКЭ	2,20	279x216x40	н/д	н/д
	Sharp PC-6781	80386SL-20	2M..8M	НГМД 1x1,44M-89	80:16,5	VGA	ЖКЭ	2,20	279x216x40	н/д	н/д
	Sharp PC-6881	80386SL-20	2M..8M	НГМД 1x1,44M-89	80:16,5	VGA	ЖКЭ-ц	3,30	279x216x48	н/д	н/д
Texas Instruments, США	TravelMate 2000	80C286-12	1M..3M	–	20:23	VGA	ЖКЭ	2,00	210x280x35	1.34	н/д
	TravelMate 3000	80386SX-20	2M	НГМД 1x1,44M-89	20	VGA	ЖК	2,50	220x280x50	*	5500
Toshiba America, США (филиал Toshiba, Япония)	Toshiba T1000	80C88-4,77	512K..1.5M	НГМД 1x720K-89	–	CGA	ЖК	2,90	н/д	*	1249
	Toshiba T1000SE	80C86-9,54	1M..3M	НГМД 1x1,44M-89	–	CGA 640x400	ЖК	2,63	315x259x46	2.30	от 1699
	Toshiba T1000XE	80C86-10	1M	–	20	CGA	ЖКЭ	н/д	н/д	*	н/д
	Toshiba T1000LE	80C86-10	1M	НГМД 1x1,44M-89	20	CGA	ЖКЭ	2,90	н/д	*	н/д
	Toshiba T2000 SX	80386SX-16	1M..9M	НГМД 1x1,44M-89	20, 40	VGA	ЖК	3,13	н/д	3.00	4999-5499
Zenith Data Systems, США	Zenith MinisPort	80C88-8/4,77	640K	НГМД 1 x н/д-59	–	CGA	ЖК	3,13	н/д	*	1999
	Zenith MastersPort 386SLe	80386SL-25	2M	н/д	85	VGA	ЖК	3,15	н/д	8.00	4999

Первая IBM-совместимая карманная ПЭВМ выпущена известной американской фирмой Atari Computer в 1989 г. Она получила название Portfolio и построена на базе 4,92-МГц МП 80C88. Емкость ОЗУ может варьироваться от 128 до 640 Кбайт. Емкость ПЗУ составляет 256 Кбайт. Сменными носителями информации являются 32-, 64- или 128-Кбайт карты памяти. Жидкокристаллический дисплей имеет разрешение 240x64 точки или 40x8 символов. Машина может снабжаться одним адаптером параллельного или последовательного интерфейса, весит 450 г, имеет габариты 180x100x27 мм и стоит 399 долл. Батарейное питание рассчитано на 48 часов автономной работы.

С таким же МП фирмой Poget Computer выпускается карманный компьютер PC Expo стоимостью 1999 долл.

Информацией о других IBM-совместимых карманных ПЭВМ автор не располагает.

3.5.6. Оценки производительности

Из приведенного обзора при сравнении двух моделей ПЭВМ отнюдь не всегда ясно, какая из них является более производительной. В подразделе 3.1 мы рассматривали методику, принятую журналом BYTE для определения производительности ПЭВМ.

Оценки производительности IBM-совместимых ПЭВМ, выполненные по этой методике, представлены в табл. 3.12 и 3.13. Первая из них составлена для ПЭВМ, в которых установлен математический сопроцессор, а вторая — для компьютеров без него. Конечно, в эти таблицы включены не все выпускаемые модели ПЭВМ, в частности, вовсе не отражены компьютеры на базе МП 80486. Дело в том, что в таблицах фигурируют только те модели из рассмотренных в данном разделе, выпуск которых освоен не позднее 1989 г.

Таблица 3.12

Оценки производительности IBM-совместимых ПЭВМ с математическим сопроцессором

Модель ПЭВМ	Обобщенный индекс производительности
SIA 386/33	32.64
ME 386-33	27.53
National Microsystems Flash 386-33	26.58
FIVESTAR 386/33	26.47
Compaq DESKPRO 386/33	24.61
Dyna Cache 386/33	24.13
ALR FlexCache 33/386	24.02
Blackship 386/33	23.77
AST Premium 386/33	22.69
Everex STEP 386/33	22.62
PC Link 386/33	22.01
ALR FlexCache 25/386	21.24
Tangent 333	21.17
SIA 386/32	20.67
Zenith Z-386/33	20.19
Matrix MDP 386-33	20.11
AST Premium 386/25	19.20
Dell System 320	18.24
Proteus 386-25	17.95
ALR FlexCache 20/386	17.94
Toshiba T5200	17.86
ALR MicroFlex 7000	17.61
Compaq DESKPRO 386/20e	17.26
IBM PS/2 Model 70-A21	16.64
Dolch-P.A.C. 386-25	16.45
AST Premium/386C	16.14
ME Regal II	15.76
AST Premium/306	14.85
FIVESTAR 306/20	14.67
Tandy 5000MC	14.27
Everex STEP 306/20	13.98
Dolch-P.A.C. 386-20C	13.58
Compaq Portable 386	13.33
IBM PS/2 Model 80-111	13.16
IBM PS/2 Model P70	13.02
IBM PS/2 Model 70-121	12.72
CompuStar 286	12.49
NEC ProSpeed 386	12.29
Compaq DESKPRO 386s	11.51
Dell System 220	11.44
Toshiba T5100	11.04
Zenith TurbosPort 386	10.84
IBM PS/2 Model 70-E61	10.52
Compaq DESKPRO 386	10.38
GRiDCase 1530	9.58
IBM PS/2 Model 55SX	9.53
GRiDCase 1535 EXP	9.44
Dell System 210	8.34
IBM PS/2 Model 55Z	8.12
AST Bravo/Z86	6.89
NEC PowerMate Portable SX	6.77
IBM PC AT (8 МГц)	5.00
IBM PC XT (4,77 МГц)	1.47

Таблица 3.13

Оценки производительности IBM-совместимых ПЭВМ без математического сопроцессора

Модель ПЭВМ	Обобщенный индекс производительности
ME 386	11.54
Gateway 386/20	11.21
ZEOS 386/16	10.64
Micro 1 Power 386-20	10.03
Club 386	9.76
Compaq SLT/286	7.70
Ogivar 286 LapTop	6.79
Zenith SupersPort 286	6.43
Mitsubishi MP-286L	5.64
NEC UltraLite	н/д

3.6. ПЭВМ фирмы Apple Computer

Как уже подчеркивалось в подразделе 1.2, фирма Apple стояла у истоков создания ПЭВМ. Она и сейчас вносит существенный вклад в структуру рынка ПЭВМ, уступая по объему сбыта компьютеров этого класса только корпорации IBM. Если учесть, что Apple производит несовместимые с IBM изделия, то их качество должно быть превосходным, чтобы занимать одно из ведущих мест на рынке.

Более того, компания Apple оставляет IBM позади себя в таких секторах рынка, как учебные и бытовые ПЭВМ, а также настольные издательские системы.

Apple всегда превосходила и превосходит IBM по графическим возможностям (в том числе цветным) своих изделий.

Компьютеры фирмы Apple делятся на 2 семейства — собственно Apple и Macintosh (Mac).

Семейство Apple включает три *серии* ПЭВМ: Apple I, Apple II и Apple III. Наибольшей популярностью пользовались модели серии Apple II — такие, как собственно Apple II, Apple IIC, Apple IIE и Apple IIGS. Они предназначены главным образом для бытовых приложений. Заметным недостатком семейства Apple является замкнутость архитектуры. Основная ОС для него — Apple DOS.

ПЭВМ *семейства Mac* обладают существенно большей вычислительной мощностью и построены на 32-разрядных МП фирмы Motorola. Основные технические характеристики моделей этого семейства представлены в табл. 3.14.

Таблица 3.14

Технические характеристики моделей ПЭВМ семейства Macintosh

Модель	МП	Сопроцессор	Емкость ОЗУ, байт	Количество НГМД х емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт: среднее время доступа, мс	Дисплейный адаптер: разрешение, точек-количество цветов (градаций яркости)	Цена, долл.
Macintosh	68000-7,83	-	128K..512K	1x400K-89	н/д	512x342	н/д
Macintosh Portable	68000-15,67	-	1M..9M	1x1,44M-89	40:25	640x400	5799-6499
Mac Plus	68000-7,83	-	1M..4M	1x400K-89, 1x800K-89	20..80	н/д	н/д
Mac SE	68000-7,83	-	1M..4M	1x400K-89, 1x800K-89	20..80	н/д	н/д
Mac SE/30	68030-15,68	68882	1M..8M	1x1,44M-89	40, 80	н/д	4369-6569
Mac II	68020-15,68	68881	1M..8M	1x1,44M-89	40, 80	640x480-256	н/д
Mac IIx	68030-15,68	68882	1M..8M	1x1,44M-89	40, 80	640x480-256	4669-9000
Mac IIcx	68030-15,68	68882	1M..8M	1x1,44M-89	40, 80	640x480-256	н/д
Mac IIci	68030-25	68882	1M..8M	1x1,44M-89	40, 80	640x480-256	н/д

Семейство Mac несовместимо с Apple и также включает 2 *серии* — собственно Mac и Mac II. Серия Mac, как и семейство Apple, характеризуется замкнутостью архитектуры, что ограничивает возможности расширения ресурсов. Серия Mac II объявлена в 1987 г. и предполагает открытую архитектуру (как в IBM), несовместимую с серией Mac. Представители серии Mac предназначены для конторских приложений, использования в качестве настольных издательских систем, а также для производства научных и инженерных расчетов. Таким образом, это ПТЭВМ. На машинах серии Mac используется ОС Macintosh OS с графическим интерфейсом, а на изделиях серии Mac II применяется UNIX-подобная система A/UX фирмы Apple.

Кратко охарактеризуем перечисленные в табл. 3.14 модели. ПЭВМ Macintosh была первой в семействе Mac. Macintosh Portable — это единственная переносная модель в данном семействе. Она имеет монохромный жидкокристаллический дисплей, весит 6,3 кг и способна работать в автономном режиме (с питанием от аккумуляторных батарей) в течение 10 час. Ее габариты составляют 387x354x101 мм. Компьютеры Mac Plus и Mac SE являются развитием модели Macintosh. Существенных различий между Mac Plus и Mac SE нет, за исключением объемов ПЗУ (128 Кбайт в Mac Plus и 256 Кбайт в Mac SE). Модель же Mac SE/30 внешне выглядит, как и другие представители семейства Mac, но представляет собой по сути аналог ПЭВМ Mac IIx. Машина Mac IIcx является более компактным вариантом изделия Mac IIx. Новинка в серии Mac II — компьютер Mac IIci.

Оценки производительности большинства моделей семейства Mac представлены в табл. 3.15.

Таблица 3.15

Оценки производительности ПЭВМ семейства Macintosh

Модель ПЭВМ	Обобщенный индекс производительности
Mac IIci	н/д
Mac IIcx	17.97
Mac SE/30	17.04
Mac IIx	16.81
Mac II	13.66
Mac SE	5.00
Mac Plus	4.36

В настоящее время фирма Apple предлагает для выпускаемых ею ПЭВМ платы, обеспечивающие выполнение большинства программ, разработанных для функционирования в среде ОС MS-DOS. Таким образом, с определенной натяжкой можно говорить, что изделия фирм Apple и IBM совместимы по исполнению.

Насколько нам известно, в настоящее время аналоги машин фирмы Apple отсутствуют.

3.7. ПЭВМ независимых производителей

К этому классу ПЭВМ будем относить изделия, несовместимые с продукцией фирм IBM и Apple Computer.

Класс ПЭВМ независимых производителей сейчас непредставителен. Действительно, на рынке ПТЭВМ фигурируют, в основном, IBM-совместимые изделия и, конечно же, продукция самой корпорации IBM. Лишь только компания Apple пытается отвоевать его часть, поставляя машины серии Mac II, и эти попытки небезуспешны. На рынке же бытовых и учебных ПЭВМ выделяется, главным образом, продукция фирмы Apple.

Чтобы изменить сложившуюся структуру рынка, независимым производителям отнюдь не достаточно предложить более хорошую модель ПЭВМ, чем имеющиеся: им нужно еще и убедить покупателей в том, что фирма в обозримом будущем не уступит лидерства и ее продукция будет играть роль стандарта, на который будут ориентироваться разработчики аппаратных и программных средств. Это связано с большим объемом накопленного ПО, консерватизмом пользователей, сложностью их адаптации к новым машинам и ОС, а также хорошей репутацией фирмы IBM. Реализовать сказанное в настоящее время представляется весьма проблематичным.

Тем не менее на рынках бытовых и учебных ПЭВМ заметны и другие производители.

В качестве *бытовых* определенной популярностью пользуются настольные ПЭВМ, выпускаемые фирмами Commodore International и Atari. Первая предлагает модели семейства Amiga (Amiga, Amiga 1000, Amiga 2000), а также Commodore 64, а вторая — машины Atari 520ST, Atari 1040ST, Mega ST2 и Mega ST4. Несмотря на то, что некоторые из них имеют значительную вычислительную мощность (в том числе МП 68000, ОЗУ до 4 Мбайт и хорошие графические возможности), в качестве профессиональных машин они практически не используются.

В начале 1988 г. фирма Atari объявила о своей новой модели Atari ABAQ. Эта ПЭВМ, по своим возможностям соответствующая хорошей графической рабочей станции, построена на базе транспьютера T800 (Inmos), работающего на частоте 20 МГц. Емкость ОЗУ составляет 5 Мбайт, из которых 1 Мбайт используется в качестве видеопамяти. Этот компьютер оснащен 89-мм НГМД и 80-Мбайт НЖМД. Дисплейный адаптер обеспечивает разрешение 1280x960 точек при 16 цветах в рабочей палитре (базовая палитра включает 4096 цветов), либо 1024x768 точек при 256 цветах в рабочей палитре (базовую палитру

в этом случае составляют 16 млн. цветов), либо 512x480 точек при 2 цветах. Предусмотрена возможность установки дополнительно до 12 транспьютеров и стыковки с четырьмя такими же ПЭВМ. Используется ОС Helios, имеющая черты системы UNIX.

Особую популярность снискал дешевый бытовой компьютер ZX Spectrum+ фирмы Sinclair Research (Великобритания), выполненный на базе 8-разрядного МП Z80A фирмы Zilog, работающего на тактовой частоте 1МГц. Емкость ОЗУ составляет всего 48 Кбайт, а в качестве внешней памяти используется НМЛ. Дисплейный адаптер обеспечивает разрешение 256x176 точек при восьми цветах, а роль дисплея может играть бытовой телевизор. В среде отечественных радиолюбителей и пользователей компьютеров вокруг этой ПЭВМ сейчас наблюдается настоящий бум: ZX Spectrum+ производится в больших количествах кустарным способом и пользуется громадным спросом.

В 1988 г. С.Джобс (один из разработчиков первых моделей семейства Apple), руководя созданной им компанией, разработал настольную ПЭВМ NeXT, которая сразу привлекла внимание специалистов благодаря известности автора и ряду новых технических решений, в ней воплощенных. Эта машина построена на основе 25-МГц МП 68030 и 25-МГц математического сопроцессора 68882 (Motorola). Емкость ОЗУ может составлять от 8 до 16 Мбайт; ПЗУ же невелико — всего 64 Кбайт. ПЭВМ может комплектоваться НЖМД емкостью 256 — 926 Мбайт. Вместо НГМД используются НОД с дисками на 256 Мбайт, обеспечивающие чтение, запись и стирание. В их основе лежит магнитооптический принцип записи. Это наиболее новаторское решение, воплощенное в ПЭВМ. Диаметр ОД составляет всего 133 мм; среднее время доступа равно 92 мс. Дисплейный адаптер обеспечивает разрешение 1120x832 точки при 4 градациях яркости. Эту модель отличает также наличие средств высококачественного воспроизведения звука. Данная ПЭВМ имеет быстродействие около 4 млн. команда/с, великолепный дизайн и базовую цену 6500 долл. За дополнительную плату (2000 долл.) поставляется лазерный принтер. В качестве ОС используется UNIX-подобная система Mach, разработанная в университете Карнеги-Мелона. По мнению автора компьютера, его детище предназначено пока только для *системы высшего образования*.

Интерес также представляет карманная ПЭВМ Agenda английской фирмы Microwriter Systems. Она появилась на рынке в 1989 г. и построена на базе МП 6303 фирмы Hitachi. Емкость ОЗУ составляет 320 Кбайт, а ПЗУ — 32 Кбайт, причем первое снабжено батарейным питанием, а потому может играть роль долговременной памяти. Габариты компьютера составляют всего 175x85x18 мм. Клавиатура занимает площадь 89x102 мм и отличается новизной: она насчитывает пять расположенных по дуге клавиш — по одной на каждый палец руки; код символа при этом формируется нажатием определенной комбинации клавиш. Имеется также и обычная, но очень миниатюрная клавиатура. Жидкокристаллический экран позволяет воспроизводить 20x4 символов. Машина может автономно питаться от никелькадмиевых аккумуляторов. Она также снабжена адаптерами последовательного и параллельного интерфейсов для связи со стационарной ПЭВМ и принтером. Цена ее составляет 250 долл.

В Японии фирма NEC прилагает все усилия к внедрению ПЭВМ собственной оригинальной разработки и добилась на этом поприще заметных успехов. Самым популярным в Японии ее изделием является 26-разрядный компьютер PC-9801, несовместимый с продукцией IBM.

3.8. ПЭВМ, выпускаемые в восточноевропейских странах

В восточноевропейских странах (в том числе СНГ) выпускаются главным образом настольные ПЭВМ различного функционального назначения: бытовые, учебные и профессиональные. Мы ограничимся рассмотрением только последних, так как другие пока имеют весьма ограниченные возможности, представляя собой в подавляющем большинстве лишь занимательные игрушки.

Выпускаемые в настоящее время в Восточной Европе ППЭВМ обладают отнюдь не выдающимися техническими характеристиками, хотя имеются и интересные перспективные разработки. В рассматриваемом регионе делается ставка на производство IBM-совместимых ППЭВМ. Это позволяет, с одной стороны, использовать уже разработанное ПО, а с другой, — создавать конкурентоспособные на западном рынке новые программные продукты.

Вместе с тем не отсекаются и другие линии ПЭВМ. Так, в СНГ с 1986 г. выпускается ППЭВМ «Электроника 85» (МС 0585), программно совместимая с миниЭВМ СМ, прототипом которых послужили компьютеры фирмы DEC. Она базируется на 16-разрядном МП, имеет ОЗУ емкостью до 4 Мбайт, два 133-мм НГМД на 360 Кбайт и НЖМД емкостью 5 — 10 Мбайт. Дисплейный адаптер обеспечивает разрешение 960x420 точек. Возможно подключение как монохромного, так и цветного дисплея.

Основные технические характеристики IBM-совместимых стационарных (точнее — настольных) ПЭВМ, производимых в странах Восточной Европы, представлены в табл. 3.16. К сожалению, за неимением информации в ней не нашли отражение сведения о ПЭВМ, выпускаемых в Югославии. А сожаление вызывает тот факт, что они производятся по лицензиям западных фирм и имеют хорошие характеристики.

Более подробно охарактеризуем отечественные ПЭВМ, в частности, изделия ЕС. Все выпускаемые в настоящее время модели, за исключением ЕС1842, совместимы с IBM PC XT. ЕС1842 построена на базе МП К1810ВМ86М, который способен программно эмулировать прибор 80286 (но при этом снижается производительность ПЭВМ). Следовательно, модель ЕС1842 совместима с IBM PC AT. Наибольшее распространение получила пока только модель ЕС1840, ресурсы которой по нынешним меркам весьма ограниченны. В частности, в ней не

Таблица 3.16

Технические характеристики IBM-совместимых настольных ПЭВМ, выпускаемых в восточноевропейских странах

Страна-производитель	Модель	Класс	МП	Факультативный сопроцессор	Емкость ОЗУ, байт	Количество НГМД x емкость, байт-диаметр, мм	Емкость НЖМД, Мбайт	Дисплейный адаптер-дисплей
Болгария	ЕС1832 (ИЗОТ 1037.С)	XT	8088-4,77	н/д	640К	2x360К-133	5, 10 [2x10]	CGA-м, CGA-ц
	ЕС1839 (Правец 16А)	XT	К1810ВМ86-4,77	н/д	640К	2x360К-133	20	CGA-м, CGA-ц
	ЕС1838	AT	80286-8/6	80287	640К..8М	2x360К-133, 2x1,2М-133	10..40	CGA-м, CGA-ц
Венгрия	ЕС1830	XT	8088-4,77	н/д	640К	2x360К-133	н/д	CGA
	VT110	XT	8088-8/4,77	н/д	640К	2x360К-133	2x20	MDA, CGA, EGA
	ЕС1861	AT-386	80386-16	н/д	до 8 Мбайт	2x1,2М-133	40	н/д
Восточная Германия	СМ1910	XT	К1810ВМ86-4,77	н/д	768К	2x360К-133	40	CGA-м, EGA-м
	ЕС1834	XT	К1810ВМ86-4,92	К1810ВМ87	640К	2x720К-89	20[42]	CGA-м
Польша	ELWRO 801 AT	AT	80286-8/6	н/д	1М	1x360К-133, 1x1,2М-133	20	н/д
	MAZOVIA 1016	XT	К1810ВМ86-8/4,77	н/д	640К	2x360К-133	20	н/д
	MAZOVIA 2016	AT	80286-10/6	н/д	1М	1x360К-133, 1x1,2М-133	40	н/д
	MAZOVIA 2032	AT-386	80386-12/8	н/д	до 16М	1x360К-133, 1x1,2М-133	40	н/д
	VARSOVIA 2032	AT-386	80386-12/8	н/д	до 16М	1x360К-133, 1x1,2М-133	40	н/д
	SELESIA 286/AT	AT	80286-10	н/д	до 4М	1x360К-133, 1x1,2М-133	40	н/д
СНГ	ЕС1840	XT	К1810ВМ86-4,77	-	256К..640К	2x720К-133	-	MDA, CGA-м, CGA-ц
	ЕС1841	XT	К1810ВМ86-4,77	К1810ВМ87	512К..2М	2x720К-133	[10]	CGA-м, CGA-ц
	ЕС1845	XT	К1810ВМ86-4,77	н/д	до 1М	2x720К-133	[20]	CGA-ц
	ЕС1842	XT	К1810ВМ86М-8	К1810ВМ87	1М..3М	2x720К-133	[20]	EGA-м, EGA-ц
	Искра 1030	XT	К1810ВМ86-4,77	н/д	256К..640К	2x360К-133	[5, 16]	CGA-м, EGA-ц
	Искра 1030 ТУРБО	XT	К1810ВМ86-8	н/д	2М	2x360К-133	до 40	CGA, HGA, EGA
	Истра 4816	XT	К1810ВМ86а 2хК580ИК80	К1810ВМ87а Форт-процессор	до 4М	по выбору	5, 10, 20, 40	CGA, EGA
	Нейрон И9.66	XT	К1810ВМ86-4,77	н/д	256М..1М	2x360К-133	[5]	CGA-м
	СМ1810	XT	К1810ВМ86-4,77	н/д	256М..1М	1x360К-133	35 [165]	CGA-м

предусмотрена возможность подключения математического сопроцессора и манипулятора типа «мышь». Отсутствует также контроллер НЖМД. ЕС1840 снабжена одним адаптером параллельного и двумя адаптерами последовательного интерфейсов. Базовая цена этой модели составляет 6 тыс. руб. (по ценам 1990 г.). ПЭВМ ЕС1841 обладает более богатым набором аппаратных средств. Так, обеспечивается возможность расширения ОЗУ до 2 Мбайт (о чем мы еще скажем особо), подключения математического сопроцессора и «мыши», а также имеется контроллер НЖМД (дополнительными сведениями о модели ЕС1845 мы не располагаем). В настоящее время разрабатывается ПЭВМ ЕС1851 на базе МП 80386 с тактовой частотой 16 — 20МГц.

ПЭВМ «Искра» и «Нейрон» по сравнению с ЕС имеют некоторые отличия в периферии. Так, в первой используются односторонние, а не двухсторонние, НГМД.

Число моделей ПЭВМ СМ в будущем будет увеличено с введением машины СМ1820, выполненной на 32-разрядном МП типа 80386.

Новинкой отечественного парка ПЭВМ является машина «Истра 4816». Она имеет многопроцессорную архитектуру и включает следующие МП:

1) системный процессор К1810ВМ86, ориентированный на выполнение программ и взаимодействие с ПУ через второй процессор;

2) периферийный 8-разрядный процессор К580ИК80, обслуживающий ПУ;

3) системный процессор К580ИК80, позволяющий выполнять программы, составленные для 8-разрядных машин.

Имеются гнезда для подключения математического сопроцессора и специализированного Форт-процессора, обеспечивающего эффективное выполнение программ, составленных на этом языке. Поддерживается возможность расширения ОЗУ до 4 Мбайт.

Для специалистов полезной будет информация о степени совместимости ПЭВМ ЕС1840 и ЕС1841 с их прототипом — IBM PC XT.

Аппаратная совместимость ЕС1840/41 с IBM PC XT не обеспечивается из-за того, что платы ЕС1840/41 имеют другие размеры и разъемы. Поэтому ПЭВМ ЕС не могут быть расширены дополнительными платами, выпускаемыми для IBM PC во всем мире и в большом количестве. Правда, в ЕС1842 имеется один разъем, совместимый с разъемами IBM PC XT, но этого явно недостаточно.

Вместе с тем обеспечивается почти полная *программная совместимость* ЕС1840/41 с IBM PC XT. Различия же имеются в способах управления последовательным интерфейсом (стык С2, или RS232C), в некоторых тонкостях управления НГМД, в организации дополнительной памяти (только ЕС1841) и в способах управления манипулятором типа «мышь» (тоже только ЕС1841).

Адаптер последовательного интерфейса ЕС1840/41 реализован на микросхемах, отличных от применяемых в IBM PC XT, и программируется по-другому. В BIOS ЕС1840 вообще отсутствуют программы обеспечения работы последовательного интерфейса, из-за чего не работают многие программы, рассчитанные на использование RS232C в IBM PC XT. В BIOS ЕС1841 такие программы имеются, и аппаратные различия «упрятаны» почти полностью (в данном случае мы говорим о совместимости на уровне BIOS). Большинство программ, использующих последовательный интерфейс, не замечают подмены аппаратуры.

Отличия в управлении НГМД связаны с тем, что в ЕС1840/41 дисководы позволяют использовать на диске 80 дорожек (720 Кбайт) в отличие от 40 дорожек (360 Кбайт), поддерживаемых НГМД IBM PC XT. BIOS нейтрализует это различие, передвигая головки НГМД через дорожку при использовании формата на 40 дорожек (360 Кбайт) для обеспечения совместимости по информации с IBM PC XT. Однако при этом возникают две проблемы.

Первая состоит в том, что во многих программных комплексах используются средства защиты от копирования, основанные на специальном формате дискеты. Если проверка этого формата при запуске программы осуществляется в обход BIOS, то санкционированность копии программы подтверждена не будет. В ЕС1842 и ЕС1845 моделируются 80-дорожечные НГМД IBM PC AT и указанной проблемы не возникает, так как «нормальные» программы работают с ними хорошо.

Вторая проблема заключается в том, что записанная в 80-дорожечном дисководе дорожка в два раза уже стандартной. Поэтому нет гарантии возможности считывания информации в 40-дорожечном НГМД после ее записи в 80-дорожечном.

Поддержка дополнительной памяти (сверх 640 Кбайт) в ЕС1841 полностью отлична от идеологии, принятой в моделях фирмы IBM. Вспомним, что МП К1810ВМ86 имеет 1-Мбайт адресное пространство. Адресация же 1-Мбайт избытка памяти реализуется специальными средствами ЕС1841. В дополнительной памяти можно хранить какую-либо информацию, используемую выполняемыми программами. Выполнять же программы непосредственно в дополнительной памяти не допускается. Можно организовать в этой области памяти и виртуальный диск, обладающий существенно бо́льшим быстродействием, чем реальный диск, однако он не будет способен сохранять информацию после отключения питания. Реализация поддержки дополнительной памяти в машинах IBM будет рассматриваться ниже.

Отличия в способах управления «мышью» практически незаметны.

По сравнению с IBM PC XT в ЕС1840/41 имеется ряд расширений. Среди них возможность загрузки знакогенератора дисплея, что позволяет программировать выводимые на экран символы (в IBM PC XT знакогенератор прошит в ПЗУ), а также большее количество клавиш на клавиатуре, целесообразность чего диктуется необходимостью работы с двумя алфавитами.

На уровне же BIOS различий между ЕС1840/41 еще меньше, чем на самом низком уровне. Вместе с тем в BIOS ЕС1840/41 имеются одна ошибка, одна недоработка и один существенный недостаток.

Ошибка заключается в том, что при загрузке ОС не инициализируется вектор прерывания 15Н, используемый для доступа к специальным функциям общего управления компьютером. В ЕС1840 они не реализованы. Однако в связи с тем, что некоторые программы вызывают прерывание 15Н, ПЭВМ может «зависать».

Недоработка просматривается в реализации одной из функций вывода на дисплей — последовательного вывода (функция 0ЕН прерывания 10Н). В ранних вариантах IBM PC при вызове этой функции необходимо было явно указывать страницу видеопамяти, в которую должен происходить вывод. В более поздних моделях вывод по умолчанию производится всегда в нулевую страницу. В ЕС1840 реализован старый вариант, из-за чего последние версии ОС MS-DOS функционируют на них некорректно.

Недостатком же является отсутствие драйвера последовательного интерфейса, что уже описывалось.

В BIOS ЕС1841 указанные ошибка и недоработка устранены; кроме того, доработаны и другие драйверы.

Таким образом, говорить о программной совместимости между ЕС1840/41 и IBM PC XT без специальных оговорок можно только лишь на уровне ОС, если она запускается на тех и других ПЭВМ.

В конце 1990 г. минским НПО «Интеграл» создан первый в СНГ наколенный компьютер собственной разработки. Он получил название ПК-300 и базируется на 7,16/4,77-МГц МП 8086. Его габариты составляют 68х306х312 мм, а масса — 4,5 кг. ПК-300 содержит 640-Кбайт ОЗУ, два 89-мм НГМД и монохромный жидкокристаллический CGA-совместимый дисплей. Время автономной работы составляет 4 часа, а стоимость (по ценам 1990 г.) — 15 — 20 тыс. руб. Планируется установка жидкокристаллического монитора с задней подсветкой и НЖМД.

В момент написания данной книги минский завод «Электроника» объявил о завершении разработки карманной ПЭВМ «ЭЛЕКТРОНИКА МК-90» стоимостью 1500 руб. (по ценам 1990 г.). Она весит 700 г и может работать на языке Basic, используя прошитый в ПЗУ интерпретатор. Емкость ОЗУ составляет 16 Кбайт. В качестве внешней памяти используются сменные модули емкостью 10 Кбайт. Имеется миниатюрный графический жидкокристаллический дисплей и адаптер последовательного интерфейса для сопряжения компьютера с другими устройствами.

3.9. Тенденции развития ПЭВМ

В настоящее время действуют и в будущем сохранятся три основные тенденции, определяющие облик перспективных ПЭВМ:

1) улучшение технических характеристик;
2) дальнейшая миниатюризация ПЭВМ;
3) удешевление персональных компьютеров.

Улучшение технических характеристик персональных машин основывается на:

— прогрессе в технологии производства электронных компонентов;
— использовании новых архитектурных решений (в том числе многопроцессорных структур);
— разработке более совершенных универсальных МП;
— расширении спектра и развитии специализированных процессоров для реализации тех или иных функций;
— использовании более совершенного периферийного оборудования.

Задачи же дальнейшей миниатюризации ПЭВМ решаются, в основном, благодаря новым технологиям производства их устройств и узлов. Однако при этом нельзя сбрасывать со счетов и использование новых физических принципов, в частности, хранения информации и ее визуализации. Качественных сдвигов можно достичь именно на этом пути.

Что касается использования ПЭВМ, то ярко выражена тенденция их объединения в ЛВС, а это имеет неоспоримые преимущества перед автономным их использованием.

В ближайшие несколько лет на рынке будут доминировать ПЭВМ с МП 80386SX. В 1994 г. ожидается появление компьютеров с МП 80586.

4. ОБЩИЕ СВЕДЕНИЯ О ПРОГРАММНОМ ОБЕСПЕЧЕНИИ ПЕРСОНАЛЬНЫХ ЭВМ

ПЭВМ, как и любая другая вычислительная машина, является не чем иным, как «слепым» *исполнителем программ*, которые и придают компьютеру всю привлекательность.

Под **программой** понимают описание, воспринимаемое ЭВМ и достаточное для решения на ней определенной задачи. Для составления программ используют искусственные языки, называемые **языками программирования**. ЭВМ, как правило, непосредственно воспринимает и выполняет программы, написанные только на одном из языков программирования, который при этом является **машинным языком** данной ЭВМ. Однако при помощи специальных программ можно обеспечить опосредованное «понимание» вычислительной машиной других языков программирования, например, за счет перевода текстов, составленных на этих языках, в тексты на машинном языке. Следовательно, программы можно составлять как на машинных языках, так и на других языках программирования, если имеются средства их реализации на ЭВМ (т.е. средства, обеспечивающие их восприятие ЭВМ).

Под **ПО** в узком смысле понимается просто совокупность программ. В широком смысле в **ПО** (наряду с программами) включают различные языки, процедуры, правила и документацию, необходимые для использования и эксплуатации программных продуктов.

Материал, содержащийся в данном разделе, вводит читателя в область ПО ПЭВМ. Он дается в обзорном плане и содержит классификацию ПО ПЭВМ, основные определения, описание функций основных его компонентов, а также характеристику ряда конкретных программных продуктов. Основное внимание при этом уделяется системному ПО (СПО).

Хотя рассмотрение и ведется применительно к ПЭВМ, базовые положения данного раздела не зависят от типа вычислительной машины и поэтому имеют универсальный характер.

4.1. Структура ПО ПЭВМ

ПО ПЭВМ по функциональному признаку традиционно делится на системное и прикладное.

Системным называется ПО, используемое для разработки и выполнения программных продуктов, а также для предоставления пользователю ЭВМ определенных услуг. Оно является необходимым дополнением к техническим средствам ПЭВМ. Без СПО машина по сути безжизненна.

Прикладным называют ПО, предназначенное для решения определенной целевой задачи или класса таких задач. К этим задачам относятся производство вычислений по заданному алгоритму, подготовка того или иного текстового документа и т.п.

Структура СПО ПЭВМ, отражающая его классификацию по функциональному признаку, приведена на рис. 4.1.

Операционные системы являются неотъемлемым обязательным дополнением ПЭВМ, организуя выполнение программ и взаимодействие пользователя с компьютером.

Другие компоненты СПО являются факультативными. Их состав определяется потребностями и желаниями пользователя.

Сервисные системы расширяют возможности ОС, предоставляя пользователю, а также выполняемым программам набор дополнительных услуг. Некоторые сервисные системы таковы, что изменяют облик ОС до неузнаваемости, а поэтому иногда называются операционными системами. Сказанное имеет отношение в особенности к *интерфейсным системам*.

Гораздо менее однородной группой системных программных средств являются *инструментальные системы*. Объединяет их то, что все они предназначены для разработки ПО, хотя часть из них может применяться и для решения прикладных задач. Использование большинства инструментальных систем связано с составлением программ, поэтому они могут считаться *системами программирования*. Однако собственно к системам программирования традиционно относят такие системы, с помощью которых можно запрограммировать и решить любую задачу, допускающую алгоритмическое решение. Иными словами, системы программирования обладают универсальностью. Другие же типы инструментальных систем являются специализированными в том смысле, что они служат для создания ПО определенного функционального назначения. При этом эффективность разработки ПО по сравнению с использованием для этой же цели универсальных инструментальных средств возрастает.

Системы технического обслуживания предназначены для облегчения тестирования оборудования и поиска неисправностей. Они являются инструментом специалистов по эксплуатации аппаратной части компьютеров и поэтому в данной книге не рассматриваются.

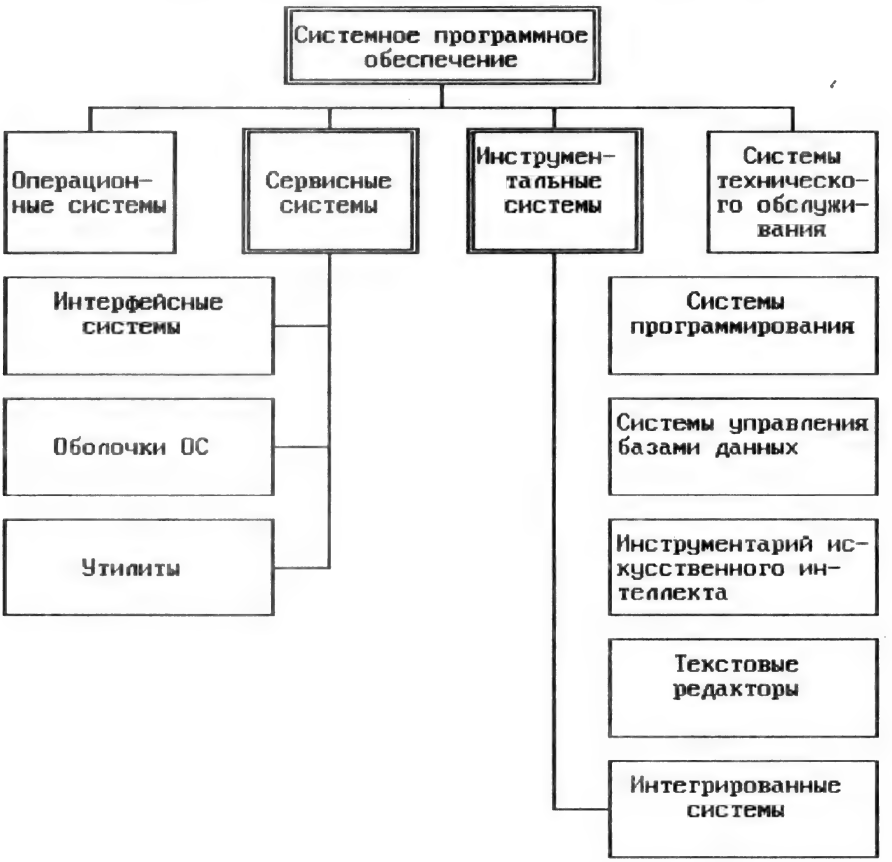


Рис. 4.1. Структура ПО ПЭВМ

В последующих подразделах мы охарактеризуем основные компоненты СПО, а также конспективно рассмотрим ряд типов прикладного ПО.

4.2. Операционные системы

Операционной системой называют комплекс программ, обеспечивающий *управление ресурсами* ЭВМ (вычислительной системы) и *процессами,* использующими эти ресурсы при вычислениях.

Под **ресурсом** понимают любой логический или физический компонент ЭВМ и предоставляемые им возможности. Основными ресурсами являются *процессор (процессорное время), ОП и ПУ.*

Управление ресурсами сводится к выполнению следующих функций:

1) *упрощению доступа к ресурсам*;

2) *распределению ресурсов между конкурирующими за них процессами*.

Реализация первой функции позволяет «спрятать» аппаратные особенности ЭВМ и тем самым предоставить в распоряжение пользователей и программистов *виртуальную машину* с существенно облегченным управлением. Иными словами, ОС поддерживает два следующих *интерфейса*, уровень которых заметно выше аппаратного:

1) **пользовательский интерфейс** (командный язык для управления функционированием компьютера и набор сервисных услуг, освобождающих пользователя от *выполнения* рутинных операций);

2) **программный интерфейс** (набор услуг, освобождающий программиста от *кодирования* рутинных операций).

Схематично интерфейсы ОС изображены на рис. 4.2.

Уровень программного интерфейса может быть дополнительно повышен путем использования систем программирования для языков высокого уровня.

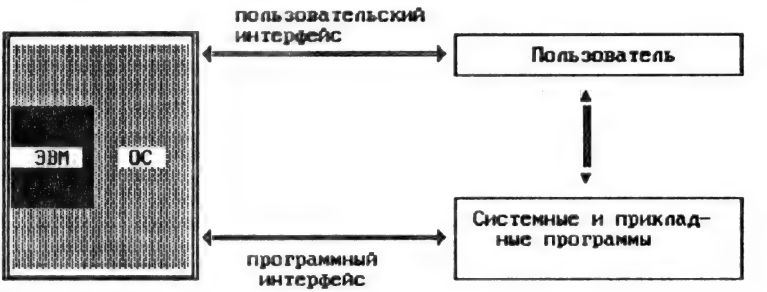


Рис. 4.2. Интерфейсы ОС

Пользовательский интерфейс может быть *символьным (текстовым) или графическим*. Последнее считается более перспективным, однако существующие в настоящее время ОС поддерживают только символьный интерфейс.

Функция распределения ресурсов присуща не всем ОС, а только тем, которые обеспечивают одновременное выполнение (или по крайней мере одновременное хранение в ОЗУ) нескольких программ. Если ПЭВМ имеет один процессор, то одновременное выполнение нескольких программ сводится по сути к поочередной реализации их фрагментов. Это не имеет ничего общего с параллелизмом, но в ряде случаев позволяет повысить производительность ПЭВМ в результате *совмещения* выполнения операций в различных устройствах во времени. Например, пока одна программа ожидает завершения обмена информацией с ПУ, другая может использовать МП. Конечно, для такой организации работы необходимы соответствующие аппаратные средства. Если такой режим работы компьютера недоступен, то одновременное выполнение нескольких программ сводится к их простому *переключению* без какого-либо выигрыша по производительности. В действительности будет наблюдаться даже проигрыш по этому показателю, однако возможность переключения программ повышает гибкость вычислительной системы и позволяет им взаимодействовать между собой, например, обмениваться информацией, в частности, обрабатывать один и тот же файл.

Процессом (задачей) же называется *последовательность действий*, предписанных программой или ее логически законченной частью, а также *данные*, используемые при вычислениях. Процесс (задача) является минимальной единицей работы, для которой выделяются ресурсы.

В настоящее время существует большое разнообразие ОС. Они классифицируются по следующим признакам:

1) по *количеству пользователей*, одновременно обслуживаемых системой;
2) по *числу процессов*, которые могут одновременно выполняться под управлением ОС;
3) по *типу доступа* пользователя к ЭВМ;
4) по *типу СВТ*, для управления ресурсами которых система предназначена.

В соответствии с *первым* признаком различают *однопользовательские* и *многопользовательские* ОС. Многопользовательские системы поддерживают одновременную работу на ЭВМ нескольких пользователей (конечно, за различными терминалами).

Второй признак делит ОС на *однозадачные* и *многозадачные*. Заметим, что если система многопользовательская, то обычно она и многозадачная, но не наоборот.

В соответствии с *третьим* признаком ОС делятся на:

— *системы с пакетной обработкой*, когда из программ, подлежащих выполнению, формируется пакет, который предъявляется ЭВМ. В этом случае пользователи непосредственно с ОС не взаимодействуют. Данный тип ОС предназначен для наиболее эффективного использования ресурсов ЭВМ;

— *системы разделения времени*, обеспечивающие одновременный диалоговый (интерактивный) доступ к ЭВМ нескольких пользователей через терминалы. Ресурсы ЭВМ выделяются при этом каждому пользователю «по очереди» в соответствии с той или иной дисциплиной обслуживания. Этот тип ОС предназначен для обеспечения удобства работы группы пользователей;

— *системы реального времени*, которые должны обеспечивать гарантированное время ответа на внешние события. Такие ОС служат для управления внешними по отношению к ЭВМ процессами и объектами.

По *четвертому* признаку ОС делятся на *однопроцессорные, многопроцессорные, сетевые* и *распределенные*.

ОС не могут, как правило, предоставить пользователям возможности, которыми не обладает ЭВМ. Они в состоянии только эффективно использовать аппаратные средства компьютера. Поэтому мы сначала перечислим возможные режимы работы ПЭВМ, чтобы понять, какими типами ОС они могут комплектоваться.

В настоящее время ПЭВМ поддерживают широкий спектр *режимов работы*, среди которых:

1) *однопрограммный* режим;

2) *однопользовательский многопрограммный,* или просто *многопрограммный* режим;

3) *многопользовательский многопрограммный,* или просто *многопрограммный* режим;

4) *система виртуальных машин* (дальнейшее развитие мультипрограммирования, основным признаком которого является возможность одновременной работы нескольких ОС, что уже отмечалось).

С точки зрения МП режимы 2 и 3 близки друг к другу, но для обеспечения последнего необходимо наличие нескольких терминалов (дисплеев и клавиатур). Многопрограммные режимы могут реализовываться как на одно-, так и на многопроцессорных ПЭВМ.

Для поддержки перечисленных режимов работы ПЭВМ существуют следующие *типы* ОС:

1) *однопользовательские однозадачные,* или просто *однозадачные*;

2) *однопользовательские многозадачные,* или просто *многозадачные*;

3) *многопользовательские многозадачные*, или просто *многопользовательские*.

Для обеспечения работы ПЭВМ в режиме системы виртуальных машин необходим *монитор виртуальных машин.*

При рассмотрении режимов работы ПЭВМ и ОС не случайно использовались различные термины — соответственно «программа» и «задача». Без дополнительных пояснений здесь не обойтись, что мы сейчас и сделаем.

На аппаратном уровне случаи одновременного выполнения последовательностей команд нескольких программ или одной программы неразличимы. Понятие же «задача» вообще не вводится, а посему можно использовать лишь термин «программа», понимая под многопрограммностью способность одновременного (при наличии одного процессора — только попеременного) выполнения нескольких последовательностей команд.

На уровне же ОС дело обстоит несколько иначе: считается, что система организует выполнение задачи, формируемой из целой программы или из логически законченного фрагмента программы. Поэтому в данном случае правомерно говорить об одно- или многозадачности. Однако следует иметь в виду, что многозадачность бывает разная. Простейшим случаем многозадачности является поддержка одновременного выполнения нескольких программ без возможности разбиения программы на несколько задач. «Чистая» же многозадачность предполагает обеспечение такой возможности. Это дополнительно требует наличия в составе ОС средств для *взаимодействия* и *синхронизации* процессов. В связи с различными видами многозадачности применительно к ОС иногда употребляют термины *«многопрограммность»* для обозначения простейшего случая многозадачности и собственно *«многозадачность»* для обозначения полностью реализованного многозадачного режима. Мы же будем употреблять только термин *«многозадачность»*, понимая его в широком смысле. В целях конкретизации при этом будет использоваться понятие *«гранула параллелизма»*, которой может являться программа целиком, процесс (задача) как часть программы или даже цепочка команд в рамках процесса.

Дополнительно заметим, что многопользовательская ОС должна быть многозадачной (иначе нельзя будет обслуживать нескольких пользователей одновременно), хотя последняя возможность в отдельности каждому конечному пользователю может и не предоставляться.

Для многопользовательских и многозадачных ОС важным показателем является *дисциплина обслуживания.* В соответствии с этим различают вытесняющий и согласующий режимы многозадачной работы.

При *вытесняющей* организации выделением задачам процессорного времени занимается исключительно ОС. Примерами такого режима являются *квантование*, когда каждой задаче процессор выделяется по очереди, причем на фиксированный промежуток времени, и *приоритетное обслуживание*. Вытеснение поддерживают ОС OS/2 и UNIX, а также интерфейсная система DESQview.

В случае *согласующей* организации каждая задача, получившая управление, сама определяет, когда ей отдать процессор другой задаче. Иначе говоря, здесь инициатива исходит не от ОС, а главным образом от самой задачи. Согласование применяется в сетевой ОС фирмы Novell, а также в интерфейсной системе MS Windows.

В общем случае согласование эффективнее и надежнее вытеснения, так как позволяет программе самой выбирать удобный и безопасный момент своего прерывания. Однако при этом ни одна из программ не должна узурпировать процессор, добровольно отказываясь от монопольного его использования.

Очевидно, однозадачная ОС может быть поставлена на поддерживающую любой режим работы ПЭВМ, что и делается многими пользователями. Однако, на что уже обращалось внимание читателя, современные мощные ПЭВМ имеют такие ресурсы, которые не могут быть эффективно использованы одним пользователем даже в многопрограммном режиме. На таких машинах целесообразнее применять многопользовательские ОС.

Для IBM-совместимых ПЭВМ разработаны и используются следующие классы ОС:

1) ОС семейства CP/M;

2) ОС семейства DOS;

3) ОС семейства OS/2;

4) ОС семейства UNIX;

5) различные ОС и управляющие программы, предназначенные для ПЭВМ на базе МП 80386 (и, естественно, 80486).

Наибольшее распространение в настоящее время имеют представители семейства DOS, за ними — UNIX, и в меньшей степени — OS/2. Данное соотношение в ближайшие годы сохранится. Основные характеристики перечисленных семейств ОС, в соответствии с которыми можно

выбрать наиболее подходящую для Вашей ПЭВМ и Вас систему, сведены в табл. 4.1. Сделаем относительно нее следующие замечания:

— для OS/2 в настоящее время имеется серьезный конкурент — интерфейсная система MS Windows (см. п. 4.3.1);

— некоторые UNIX-подобные системы являются менее требовательными к ресурсам ПЭВМ и способны функционировать на ПЭВМ менее мощных классов;

— можно использовать и меньший, чем указано, объем ОЗУ, однако при этом некоторые программы могут оказаться неработоспособными, а эффективность ОС снизится.

Таблица 4.1

Основные характеристики ОС ПЭВМ

Характеристика	DOS	OS/2	UNIX
Тип ОС	однозадачная с элементами многозадачности	многозадачная	многопользовательская
Рекомендуемая минимальная конфигурация ПЭВМ	МП 8088/86(XT), ОЗУ - 640Кбайт, два НГМД	МП 80286 (AT), ОЗУ- 2..6Мбайт, НЖМД - 20 Мбайт	МП 80386 (AT-386, PS/2), ОЗУ- 4..8Мбайт, НЖМД - 40..80 Мбайт
Максимальный объем ОЗУ для пользователя	640 Кбайт с возможностью расширения	16 Мбайт на процесс	2 Гбайт на процесс
Организация памяти	физическая, с сегментацией	виртуальная, с сегментацией и подкачкой по требованию	виртуальная, страничная, с подкачкой по требованию
Системная защита	нет	защищенный режим работы на уровне ОС	пароли пользователя, разрешения группового уровня
Режим работы МП	реальный	защищенный 16-разрядный	защищенный 32-разрядный
Величина гранулы параллелизма	программа	цепочка команд в рамках процесса	процесс
Средства взаимодействия между процессами	нет	разделяемая память, очереди сообщений, семафоры, программные каналы	разделяемая память, очереди сообщений, семафоры, программные каналы
Управление вводом-выводом	простое, часто с последовательным опросом	по прерываниям	по прерываниям
Сетевые средства	PC Net, MS Net, PC NFS, фирмы Novel и др.	менеджер локальной сети (LAN)	TCP/IP, NFS, RFS

В последующих пунктах мы охарактеризуем каждый из пяти упомянутых классов ОС, а также кратко остановимся на сетевых ОС и ОС для ПЭВМ фирмы Apple Computer.

4.2.1. ОС семейства CP/M

Система CP/M (от англ. Control Program for Microcomputers — управляющая программа для микрокомпьютеров) была разработана в 1974 г. фирмой Digital Research и исторически является одной из первых ОС для ПЭВМ. Она предназначена для управления ресурсами 8-разрядных персональных компьютеров на основе МП 8080 и в настоящее время фактически служит стандартом для данного класса ПЭВМ. Это *однозадачная* ОС, состоящая из нескольких компонентов, что позволяет достаточно легко адаптировать ее к архитектурным особенностям той или иной машины путем перекодирования только одного компонента, а именно, BIOS.

Развитием CP/M явилась система CP/M-86 (незаконно русифицированный и распространяемый ее вариант — М-86), предназначенная для ПЭВМ класса XT. Дальнейшее совершенствование ОС CP/M привело к появлению многозадачной системы CCP/M-86 (Concurrent CP/M-86), а затем и многопользовательской ОС MP/M-86.

Достоинство систем данного класса состоит в малой потребной емкости ОЗУ для работы — в пределах 20 Кбайт, однако сейчас это уже не так актуально. Представители семейства CP/M довольно примитивны и имеют слабый командный язык наряду с простейшей файловой системой. Поэтому на 16-разрядных ПЭВМ они находят весьма ограниченное применение, хотя в принципе их можно использовать на любом IBM-совместимом персональном компьютере. В свете сказанного по совсем непонятным для нас причинам ЕС 1840 первоначально поставлялись только с ОС М-86.

4.2.2. ОС семейства DOS

Системы данного семейства в настоящее время являются наиболее популярными ОС для ПЭВМ. Первый представитель этого семейства — система MS-DOS (Microsoft Disk Operating System — дисковая операционная система фирмы Microsoft) была выпущена в 1981 г. в связи с появлением ПЭВМ IBM PC. Среди этих систем наряду с упомянутой широко известна ОС PC DOS (Personal Computer DOS — DOS для персональных компьютеров), распространяемая корпорацией IBM, а в последнее время стали популярными также программные продукты DR DOS уже известной нам фирмы Digital Research. Есть и системы других производителей, учитывающие архитектурные особенности того или иного типа ПЭВМ, либо обладающие некоторыми дополнительными возможностями. ОС различных производителей, но одинаковой версии, отличаются лишь незначительными деталями. Например, драйвер виртуального диска ОС MS-DOS записан в файле RAMDRIVE.SYS, а PC DOS — в файле VDISK.SYS. Различаются и имена двух основных системных файлов.

В настоящее время главным образом применяются системы DOS 3.3 (1987 г.), а также DOS 4.0 (1988 г.) и появившаяся вслед за ней DOS 4.01. В 1991 г. представлена ОС DOS 5.0, которая, вероятно, получит широкое распространение, вытеснив предыдущие версии.

DOS 3.3 — это добротная, надежная система, использующая все возможности ПЭВМ класса XT. Пока предпочтение отдается именно данной версии DOS, в особенности, в СНГ, где преобладают компьютеры указанного класса.

По сравнению с DOS 3.3 система DOS 4.0:

— имеет дополнительную (MEM — показать распределение ОЗУ) и усовершенствованные команды;

— поддерживает жесткие диски емкостью свыше 32 Мбайт без необходимости их разбиения на ряд логических дисков (в предыдущих версиях системы это не допускалось);

— располагает упрощенной процедурой установки системы;

— обладает улучшенными видеографическими средствами (благодаря расширению команд MODE и GRAPHICS), реализующими все режимы работы графических адаптеров EGA и VGA;

— имеет более широкий набор сообщений об ошибках.

Следовательно, DOS 4.0 использует ряд специфических возможностей ПЭВМ класса AT, а поэтому ее целесообразнее применять именно на таких компьютерах вместо DOS 3.3.

Однако DOS 4.0 имеет тенденцию зависать, обладает меньшей эффективностью, требует больше памяти и в свое время не получила благословения фирмы IBM. Она распространена, главным образом, в Европе.

DOS 4.01 снабжена собственной оболочкой MS-DOS Shell графического типа (см. п. 4.3.2), чем и отличается от DOS 4.0.

Первоначально предполагалось, что DOS 5.0 будет полностью использовать ресурсы ПЭВМ с МП 80286. Однако этого не случилось. Тем не менее DOS 5.0 — высококачественная система, содержащая ряд дополнительных команд. Но основное ее отличие в другом, а именно в том, что данная система включает усовершенствованные средства управления оперативной памятью, которые позволяют несколько увеличить размер доступного для работы ОЗУ. Для этого необходима ПЭВМ класса AT, а еще лучше — компьютер с МП 80386/486(SX). В DOS 5.0 имеется также ряд других новшеств и усовершенствований.

ОС семейства DOS являются *однозадачными,* но имеют и некоторые *элементы многозадачности*. В частности, можно организовать фоновую печать на принтере, а также разместить в ОЗУ несколько резидентных программ и активизировать их при необходимости.

Все версии DOS совместимы снизу вверх (т.е. программа, разработанная для младшей версии, в подавляющем большинстве случаев будет работать и под управлением более старшей версии ОС).

ОС семейства DOS могут работать на всех классах IBM-совместимых ПЭВМ. Однако эти системы позволяют полностью использовать ресурсы только компьютеров класса XT.

Системы DOS содержат четыре основных компонента, два из которых являются машинонезависимыми.

На DOS заметное влияние оказали концепции, заложенные в систему UNIX. ОС семейства DOS обладают нижеприведенными характерными чертами и достоинствами:

1) возможностью задания в качестве имен файлов *образцов*, что позволяет специфицировать при выполнении тех или иных действий множества файлов вместо одного;

2) удобным и простым пользовательским интерфейсом, а также поддержкой *командных файлов*, что обеспечивает возможность создания пользовательских *макрокоманд*;

3) поддержкой *иерархической (древовидной) файловой структуры*;

4) возможностью не только *последовательного*, но и *прямого доступа* к содержимому файлов;

5) трактовкой на *логическом* уровне устройств ввода-вывода как файлов, что унифицирует средства обмена информацией с любыми ПУ и файлами;

6) наличием *конвейеров* (средств передачи вывода одной программы или команды на вход другой) и возможностью *перенаправления ввода-вывода* на уровне командного языка;

7) некоторыми средствами поддержки сетей ЭВМ;

8) модульностью структуры, упрощающей перенос системы на другие типы ПЭВМ;

9) небольшим потребным объемом оперативной памяти для работы (около 60 Кбайт) и внешней памяти для хранения системных файлов;

10) возможностью создания в памяти *виртуальных дисков*, что ускоряет обмен информацией;

11) возможностью запуска *фоновых задач*;

12) поддержкой ряда национальных алфавитов и соглашений.

Наряду с неоспоримыми достоинствами ОС семейства DOS имеют и ряд недостатков. Наиболее существенные из них — полное отсутствие средств защиты от несанкционированного доступа к ресурсам ПЭВМ и к самой ОС, а также жесткое ограничение на размер доступного для работы ОЗУ. Скорее, это недостатки того МП, для которого DOS первоначально разработана, чем самой системы. Вспомним, что МП 8088/86 не имеет средств защиты, а его адресное пространство составляет всего 1 Мбайт. Одна часть в нем отводится под ПЗУ, другая — под видеопамять, а третья — под ОЗУ максимальной емкостью 640 Кбайт. При установке DOS на модели ПЭВМ старше XT, последние эмулируют машину на базе МП 8088/86, в том числе и ее схему адресации. Таким образом, ни DOS, ни ПЭВМ, на которую она установлена, не позволяют непосредственно использовать, возможно, имеющуюся в наличии *дополнительную* память.

Существует несколько методов увеличения доступной для работы памяти (особенно в среде DOS 5.0), рассмотрение которых мы отложим до п. 5.2.6. Здесь лишь отметим, что для всеобъемлющего разрешения проблемы памяти необходима другая ОС.

ОС семейства DOS свойственна также некоторая непоследовательность командного языка.

Для отечественных ПЭВМ существуют пиратские (нарушающие авторское право) русскоязычные версии DOS, такие, как «Альфа-ДОС» (ДОС-16), АДОС и НЕЙРОН-ДОС1. Эти «русские» системы уже давно морально устарели, перевод их нельзя признать удовлетворительным, использование кириллицы (букв русского и аналогичных алфавитов) ограниченно, а качество документации вообще не выдерживает никакой критики.

Однако пользователям ПЭВМ, не владеющим английским языком, должны внушить оптимизм следующие важные события:

1) в начале декабря 1989 г. фирма IBM зарегистрировала для СНГ *кодовую страницу* (имеется в виду *кодовая таблица*) с номером 866, разработанную СП «Диалог» и фирмой Microsoft при участии фирм IBM и HP, а также зарезервировала номера 867 и 868 для других языков Содружества;

2) 5 апреля 1990 г. в Москве состоялась презентация русифицированной (*локализованной* для СНГ) совместными усилиями специалистов фирмы Microsoft и СП «Диалог» версии MS-DOS 4.01, использующей упомянутую кодовую страницу.

Русификация системы выразилась:

1) в возможности использования букв русского, а также белорусского и украинского алфавитов наравне с латинскими, в частности, для именования файлов;

2) в выдаче ОС сообщений на русском языке;

3) в переводе документации на русский язык.

За основу зарегистрированной русскоязычной кодовой таблицы была принята так называемая *«альтернативная»* таблица, получившая наибольшее распространение в СНГ потому, что она позволяет использовать зарубежные программы, выдающие псевдографику, без переделки. Стандартизовано также расположение символов кириллицы на клавиатуре ПЭВМ.

Теперь обладающие валютой организации и пользователи могут приобрести русскоязычную версию MS-DOS.

Уже завершена локализация для СНГ и новой версии системы — MS-DOS 5.0.

4.2.3. ОС семейства OS/2

В связи с созданием в 1987 г. нового семейства ПЭВМ PS/2 фирмой IBM совместно с Microsoft была разработана *многозадачная* ОС второго поколения, названная OS/2 (Operating System/2 — операционная система/2). Она считается преемником DOS и имеет схожий с последней пользовательский интерфейс. OS/2 управляет МП 80286 в защищенном режиме, а поэтому может применяться только на ПЭВМ с МП 80286 — 486. Эта система позволяет программам использовать *физическую* память размером до 16 Мбайт и *виртуальную* — до 0,5 Гбайт на каждую задачу. С целью выполнения программ, разработанных для DOS, в рамках OS/2 может быть запущена эта ОС в качестве подзадачи. OS/2 обеспечивает одновременную работу до 12 программ, но только одну программу DOS. Имеются и средства поддержки ЛВС.

В настоящее время существуют три основные *версии* OS/2:

1) MS OS/2 Standard Edition — стандартная версия для ПЭВМ типа PC AT, AT-386, AT-486 и PS/2 с такими же МП; как раз эта версия разработана IBM совместно с Microsoft;

2) IBM OS/2 Standard Edition, предназначенная только для моделей семейства PS/2 и IBM-совместимых ПЭВМ с шиной MCA, снабженных МП 80286 — 486; она разработана корпорацией IBM и ориентирована именно на шину MCA (в отличие от предыдущей версии);

3) OS/2 Extended Edition — расширенная версия, применяемая также только на MCA-ПЭВМ и созданная фирмой IBM; она содержит систему управления базой данных DB2, а также другие программные продукты, поддерживаемые IBM.

Каждая версия имеет две *редакции*:

1.0 — стандартный продукт;

1.1 — ОС, включающая графическую интерфейсную систему PM (см. подраздел 4.3), поддерживающая файлы и логические диски на винчестере размером более 32 Мбайт, а также содержащая текстовый редактор и ряд систем программирования.

В редакциях 1.2 (IBM) и 1.21 (Microsoft) реализована концепция загружаемой (устанавливаемой) файловой системы (Installable File System — IFS) и несколько изменен «внешний вид» PM.

Реализация концепции IFS обеспечивает возможность подключения во время загрузки ОС дополнительной *файловой системы*. В настоящее время предлагаются следующие файловые системы:

— *файловая система высокой производительности* (High Performance File System — HPFS), рассчитанная на быстрый доступ к сверхбольшим файлам, базам данных, а также к большому числу файлов в каталогах;

— *UNIX-совместимая файловая система*;

— *файловая система для доступа к данным на оптических дисках* (типа CD-ROM).

Стандартная же файловая система OS/2 совместима с файловой системой DOS.

Новинкой фирмы IBM является OS/2 Standard Edition 1.3, имеющая более высокое быстродействие и требующая для работы «всего» 2-Мбайт ОЗУ.

Собственный пользовательский интерфейс OS/2 похож на DOS, однако интерфейсная система PM превращает его в графический (как, впрочем, и Windows для DOS).

К дополнительным достоинствам OS/2 следует отнести:

— поддержку *динамической компоновки* программных модулей в единую *исполняемую программу*, т.е. формирование программы во время ее выполнения, а не предварительно. Динамическая компоновка позволяет включить в программу только требуемые для данного конкретного выполнения модули;

— поддержку *виртуальной памяти*.

Несмотря на широкую и даже навязчивую рекламу OS/2 ее разработчиками — фирмами IBM и Microsoft, пользователи относятся к ней с большой осторожностью. Это определяется нижеприведенными обстоятельствами:

1) консерватизмом пользователей, имеющим объективные причины, связанные с необходимостью приобретения новой ОС и ее освоения;

2) невысокой *реактивностью* (быстродействием) OS/2;

3) большим потребным объемом ОЗУ для ее работы (сама OS/2 занимает около 500 Кбайт; при этом ПЭВМ должна иметь память объемом не менее 2 Мбайт), а также невозможностью хранения системы на гибком диске, а следовательно, и загрузки с него;

4) недоработками в самой ОС;

5) небольшим количеством имеющихся в настоящее время системных и прикладных программных средств для этой ОС (если же использовать ПО, разработанное для DOS, то его эффективность заметно упадет вследствие функционирования одновременно двух ОС).

В связи с неудачами OS/2 в недрах IBM и Microsoft сформировались определенные пути их преодоления. Они оказались различными, что сделало отношения между двумя бывшими союзниками весьма натянутыми.

IBM продолжает делать ставку на OS/2 и уже в 1991 г. пообещала выпустить OS/2 2.0. Она должна использовать все возможности 32-разрядного МП (80386/486), в том числе обеспечивать работу ПЭВМ в режиме системы виртуальных машин. Кроме того, IBM заключила союз с компанией Apple для создания *объектно-ориентированной* ОС нового поколения.

Политика фирмы Microsoft состоит в поддержке двух систем одновременно: DOS в совокупности с интерфейсной системой Windows 3.0 (она пользуется сейчас лавинообразно нарастающим спросом) и OS/2, но с новым содержанием. Свежий подход к OS/2 связан с созданием нового *технологического ядра* системы, на котором будут покоиться компоненты, обеспечивающие совместимость с DOS, Windows, Posix (один из стандартов ОС UNIX) и собственно с OS/2. Таким образом, Microsoft пытается слить воедино четыре системы. Это будет OS/2 3.0, выпуск которой планируется на 1992 г.

Microsoft считает, что DOS плюс Windows — это многоцелевая система для широкого круга пользователей, в то время как OS/2 — ОС для высокопрофессионального применения.

В нашей стране аналоги OS/2 пока не выпускаются.

4.2.4. ОС семейства UNIX

По мере увеличения доли мощных ПЭВМ растет и популярность ОС семейства UNIX.

Первый представитель этого семейства был разработан в 1969 г. для миниЭВМ К.Томпсоном при участии Д.М.Ритчи, М.Д.МакИлроя и Дж.Ф.Осанна — сотрудниками американской фирмы Bell Laboratories, являющейся филиалом американской корпорации AT&T. Интересен тот факт, что работы по ОС UNIX начались вопреки желаниям администрации этой фирмы. Созданная система получилась настолько удачной, что впоследствии стала стандартом de facto для промышленных и научных организаций. По данным журнала Electronics (1987, V.60, N.21) ОС UNIX уже завоевала лидирующее положение почти на всех классах ЭВМ, включая инженерные АРМ, минисуперЭВМ, параллельные процессоры и суперЭВМ. Можно ожидать, что в будущем такое же положение сложится и в области ПЭВМ.

Основные концепции, заложенные в UNIX, воплощаются во многие новые ОС.

Большинство UNIX-подобных систем являются *многопользовательскими* и имеют:

— поддержку *иерархической* файловой структуры с *монтируемыми* дисковыми томами;

— *конвейеры* и средства *перенаправления ввода-вывода*;

— средства *коммуникации* в локальных и других вычислительных сетях;

— поддержку широкого разнообразия ПУ одинаковым с файловой трактовкой образом;

— множество полезных стандартных и дополнительных *утилит*;

— встроенные *инструментальные системы*;

— средства *парольной* защиты;

— высокую *мобильность* вследствие модульности ОС и использования для ее разработки языка программирования С;

— *открытость* для модификаций и расширений;

— эффективные средства *электронной почты* и *передачи данных*;

— поддержку *виртуальной памяти* со *страничным* запросом.

Системы семейства UNIX можно использовать на МП 80286 (есть специальные версии для этого прибора), однако наиболее полно их достоинства раскрываются в случае МП 80386 (конечно, если применяются версии, разработанные именно для него).

Для ПЭВМ создано множество вариантов ОС UNIX, среди которых наиболее известны следующие:

— Xenix System V/286 фирмы Microsoft, реализующая все возможности МП 80286 и способная функционировать на МП 80386;

— Xenix System V/386 той же фирмы, появившаяся в 1987 г. и являющаяся первой ОС, которая полностью использует все ресурсы МП 80386, в том числе виртуальную память, увеличенное адресное пространство и страничную организацию памяти;

— SCO Xenix 386 версии 2.3.2 фирмы The Santa Cruz Operation, разработанная на основе предыдущей системы и позволяющая выполнять программы DOS; имеется версия системы и для прибора 80286;

— многозадачная (но однопользовательская) PC-IX фирмы Interactive Systems, способная работать на моделях ПЭВМ типа XT и старше;

— 386/ix версии 2.0.1 той же фирмы для ПЭВМ на базе МП 80386;

— AIX корпорации IBM, разработанная для старших моделей PS/2 и интегрирующая системы UNIX и DOS (первоначально эта ОС использовалась на АРМ IBM PC RT);

— VP/ix фирм Interactive Systems и Phoenix Technologies, также интегрирующая системы DOS и UNIX;

— VENIX/386 для прибора 80386, поддерживающая работу в режиме реального времени, что противоречит широко распространенному мнению о невозможности создания на базе UNIX высокореактивной системы;

— QNX фирмы Quantum Software для ПЭВМ класса AT, имеющая высокую реактивность и также обеспечивающая режим реального времени;

— UNIX System V корпорации AT&T (последняя версия способна функционировать даже на многопроцессорных ПЭВМ);

— V/386 версии 3.2 фирмы Intel для МП 80386, аналогичная одноименной версии предыдущей системы;

— Esix V/386 версии 3.2 фирмы Esix для прибора 80386, которая также аналогична одноименной версии ОС UNIX System V фирмы AT&T;

— многопользовательская SCO UNIX System V/386 версии 3.2, использующая все возможности МП 80386 и 80486;

— BSD 4.3 Калифорнийского университета в Беркли;

— QUADRIX совместного предприятия ИНТЕРКВАДРО.

Заметим, что ОС Xenix и PC-IX не обеспечивают выполнения программ DOS. они лишь имеют средства обмена файлами с последней. Большинство же UNIX-подобных систем для ПЭВМ (в отличие от OS/2) способно одновременно выполнять *несколько* программ, ориентированных на DOS.

На отечественных ПЭВМ может использоваться относящаяся к семейству UNIX система ДЕМОС.

Широкое распространение ОС UNIX и наличие множества несовместимых ее версий привели к необходимости стандартизации этой системы. На роль стандарта для ПЭВМ претендует изделие UNIX System V версии 4.0 (AT&T), интегрирующее несколько UNIX-подобных систем и совместимая с другим кандидатом на стандарт — Posix.

4.2.5. Системы для микропроцессора 80386/486

Внедрение в ПЭВМ прибора 80386 повлекло за собой разработку ОС и управляющих программ (наряду с адаптацией системы UNIX), наиболее полно использующих его ресурсы. Нельзя сказать, что эти программные продукты пользуются большой популярностью, однако все-таки оказывают определенное влияние на состояние рынка ОС. Среди таких систем можно отметить следующие:

— VM/386 фирмы Softguard Systems, созданную по образцу VM/370 фирмы IBM для поддержки режима системы виртуальных машин;

— MVM/386 — многопользовательский вариант изделия VM/386, обеспечивающий работу до 8 пользователей;

— VM/386 MultiUser американской компании Intelligent Graphic, полностью совместимую с DOS, но поддерживающую многопользовательскую работу в сети, ядром которой является ПЭВМ на базе МП 80386;

— PC-MOS фирмы The Software Link — многопользовательскую ОС, способную работать также на МП 80286 и совместимую с DOS;

— New DOS фирмы Microsoft, реализующую возможность МП 80386 работать в режиме системы виртуальных машин и способную функционировать на приборе 80286;

— управляющую программу Merge 386, обеспечивающую одновременное выполнение нескольких ориентированных на DOS программ в среде UNIX;

— CTOS/VM фирмы Convergent Technologies, являющуюся многопользовательской системой и имеющую интерфейс с DOS;

— многозадачную систему Concurrent DOS фирмы Digital Research, совместимую с DOS (сейчас вместо нее предлагается ОС DRMultiuser DOS 5.0).

Для управления ресурсами сетей ПЭВМ служат специально разработанные *сетевые* ОС, которые предоставляют администраторам сети средства обеспечения безопасности данных путем контроля прав доступа пользователей к рабочим программам, массивам данных и ресурсам сети (печатающим устройствам, модемам и подкаталогам других рабочих мест), а также поддерживают выполнение прикладных программ нескольких пользователей. Среди таких систем наиболее популярны следующие:

— 3+Share фирмы 3Com;
— 3+Open той же фирмы;
— NetWare фирмы Novell;
— Vines/386 фирмы Banyan;
— LAN Manager корпорации Microsoft.

Лучшей среди лучших считается ОС NetWare 386 3.1.

Следует иметь в виду, что сетевая система *не заменяет* ОС на рабочих станциях, а только *дополняет* их. К тому же каждая рабочая станция должна иметь свой экземпляр сетевой ОС.

Что касается машин фирмы Apple, то для них, как уже отмечалось, разработаны и широко используются системы Apple DOS, Macintosh OS и UNIX-подобная ОС A/UX, а в 1989 г. выпущена ОС нового поколения — System 7.0 — для семейства Mac, реализующая многие возможности OS/2.

4.3. Сервисные системы

Сервисными будем называть системы, дополняющие и расширяющие *пользовательский,* а возможно, и *программный* интерфейсы ОС.

По функциональному назначению сервисные системы делят на:

1) **интерфейсные системы** (interface), в основном, графического типа, модифицирующие как пользовательский, так и программный интерфейс ОС, а также иногда реализующие и дополнительные возможности по распределению ресурсов ЭВМ; вследствие этого они считаются естественным «продолжением» ОС;

2) **оболочки** (shell) **ОС,** модифицирующие только пользовательский интерфейс, повышая его уровень (главным образом за счет «меню» и использования функциональных клавиш), а также предоставляя дополнительные возможности;

3) **утилиты** (utility) — обслуживающие программы, которые предоставляют пользователю сервисные услуги, т.е. обогащают пользовательский интерфейс ОС.

Разница между оболочками и развитыми утилитами зачастую состоит лишь в универсальном характере первых и специализации вторых.

Место сервисных систем различных типов в программно-аппаратном комплексе и их функциональные различия представлены на рис. 4.3.

Еще один вид услуг, связанный с модификацией только программного интерфейса ОС, реализуют так называемые *драйверы,* которые относятся собственно к ОС и здесь не рассматриваются. Конечно, это не означает, что они поставляются только совместно с ней. Драйверы не принадлежат к классу сервисных систем потому, что не влияют на пользовательский интерфейс ОС.

Заметим, что ПО, разработанное для функционирования под управлением какой-либо интерфейсной системы, вовсе не обязательно будет работать под управлением базовой ОС.

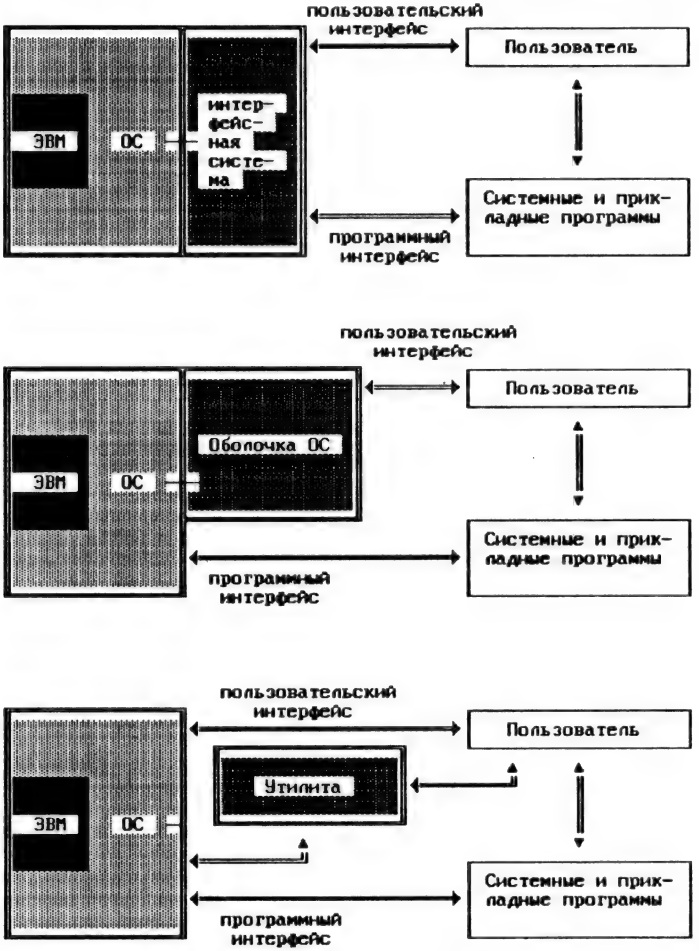


Рис. 4.3. Роль сервисных систем

В соответствии с режимом использования различают *резидентные* и *нерезидентные* сервисные системы. *Первые* постоянно находятся в ОЗУ и активизируются по мере необходимости, на что требуются минимальные усилия и время. *Вторые* находятся в памяти только до тех пор, пока не возникнет необходимость вернуться на уровень пользовательского интерфейса, предоставляемого ОС. Если же нерезидентная система наряду со своим не исключает пользовательский интерфейс ОС (т.е. является «прозрачной»), то отличия ее от резидентной системы заключаются только в деталях реализации.

4.3.1. Интерфейсные системы

Из имеющихся интерфейсных систем следует назвать:

1) Windows фирмы Microsoft, DESQview компании Quarterdeck Office Systems и Ensemble фирмы Geos Work, используемые совместно с DOS;

2) Presentation Manager (PM) фирмы Microsoft и корпорации IBM для OS/2;

3) Motif и Ten/Plus, предназначенные для работы совместно с UNIX.

Все перечисленные системы, за исключением DESQview, поддерживают *графический* пользовательский интерфейс в том смысле, что пользователь при работе с ними манипулирует *образами,* а не символами, причем с использованием «мыши».

Кроме того, все интерфейсные системы являются *многооконными.* Многооконность состоит в том, что экран дисплея динамически делится на несколько графических и/или текстовых *окон.* При этом программа (или несколько одновременно работающих программ) может (могут) выводить информацию в одно или несколько окон. Это — существенное отличие от стандартных средств ОС, которое, имея ряд преимуществ, предоставляет пользователю дополнительные воз-

можности. В частности, можно переключать ввод с клавиатуры на любую из программ (на любое окно), не обновляя в данном случае весь экран. Благодаря многооконности поддерживаются также дополнительные средства по ведению диалога с ОС и программами, в том числе посредством меню. В общем, интерфейсные системы существенно повышают уровень общения пользователей с ПЭВМ. Что же касается программных интерфейсов, то сервисные системы данного типа расширяют и дополняют услуги, предоставляемые программам (точнее — программистам).

Более того, все перечисленные системы являются *многозадачными*, даже если базовая ОС (DOS) обладает практически только однозадачными возможностями.

Система Windows выпускается с 1983 г., но первые ее версии были неудачными. 22 мая 1990 г. фирма Microsoft представила новую версию — Windows 3.0, которая благодаря своим возможностям мгновенно завоевала симпатии пользователей ПЭВМ. Более того, Windows 3.0 признана лучшим программным продуктом для персональных компьютеров 1990 г.

Windows 3.0 привлекательна прежде всего удобным *графическим пользовательским интерфейсом, многозадачными возможностями* и более высокой по сравнению с предыдущими версиями *реактивностью*. Программный интерфейс DOS этой системой также существенно модифицирован.

Windows 3.0 устанавливается поверх DOS и требует наличия по крайней мере МП 80286, ОЗУ емкостью не менее 640 Кбайт, жесткого диска, а также графического дисплея. Чтобы Ваша работа была эффективной и чтобы Вы смогли использовать все возможности Windows, следует иметь ПЭВМ с МП 80386SX или старше, а также не менее 2 Мбайт ОЗУ.

Windows 3.0 может работать в одном из следующих режимов: в реальном, стандартном и улучшенном.

Реальный режим обеспечивает использование только *стандартной* 640-Кбайт памяти и поддержку *отображаемой* (expanded-) памяти. Многозадачные возможности отсутствуют.

Для работы в *стандартном* режиме требуется как минимум 1-Мбайт ОЗУ. В этом режиме обеспечивается использование не только стандартной и отображаемой, но и *расширенной* (extended-) памяти. При этом можно выполнять *одновременно несколько* задач.

Улучшенный режим допустим только на МП 80386SX и более старших. Данный режим, как и предыдущий, обеспечивает богатые возможности по управлению памятью и *многозадачную* работу. Дополнительно к этому поддерживается *виртуальная память* и режим *системы виртуальных машин*. Поэтому можно выполнять не только одну, но и несколько программ, разработанных для DOS. Два предыдущих режима способны обеспечить выполнение лишь единственной DOS-программы. Программы же, разработанные в среде или для среды Windows, под управлением DOS функционировать не будут.

«Свои» программы Windows выполняет в режиме согласования, а «чужие» (DOS-овские) — в режиме вытеснения, так как последние не способны самостоятельно освободить процессор.

Современная версия системы DESQview, а именно, DESQview 386 2.2, является серьезным конкурентом для Windows. Она обладает *многозадачными* возможностями, поддерживает развитые средства управления *дополнительной (расширенной, отображаемой* и *верхней)* памятью и до появления Windows 3.0 была бесспорным лидером среди интерфейсных систем для DOS. DESQview может функционировать на ПЭВМ класса XT и старше, но наиболее полно проявляет себя только при наличии МП 80386(SX) и 1-Мбайт ОЗУ. Если Вам не нужен графический интерфейс, то можете отдать предпочтение данной системе, а не Windows 3.0. Система DESQview, как и Windows, способна выполнять программы, разработанные в среде DOS, в том числе одновременно несколько программ в режиме системы виртуальных машин. Кроме того, она способна выполнять и саму Windows, что обеспечивает переносимость Windows-программ в среду DESQview. Многозадачные возможности реализуются путем вытеснения.

Известно, что уже появилась версия 2.3 системы DESQview.

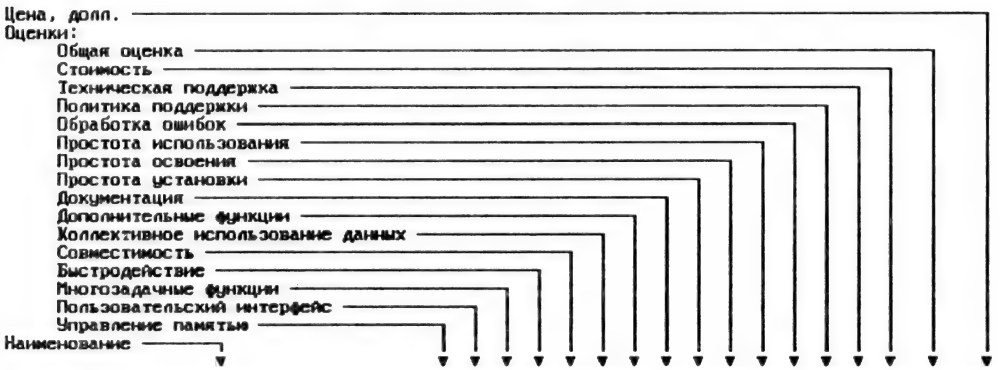
В табл. 4.2 на качественном уровне представлены основные характеристики интерфейсных систем Windows и DESQview, а также функционально аналогичной им системы OS/2, заимствованные из журнала «Мир ПК». Здесь использована и в дальнейшем будет применяться следующая оценочная шкала: О — отлично, ОХ — очень хорошо, Х — хорошо, У — удовлетворительно, П — плохо, Н — неудовлетворительно. Приведенные оценки (но в числовом выражении) умножены на веса соответствующих характеристик и все полученные числа для каждой системы просуммированы. Таким образом определена общая оценка, которая свидетельствует о превосходстве Windows 3.0. Вместе с тем OS/2 выгодно отличается средствами управления памятью, так как она управляет МП не в реальном, а в защищенном режиме, имеющем более емкое адресное пространство.

Для маломощных ПЭВМ класса XT оптимальной является многозадачная система Ensemble, поддерживающая графический пользовательский интерфейс. Она полностью написана на языке Ассемблера и обладает высокой реактивностью. Для ее функционирования достаточно иметь 512-Кбайт ОЗУ и свободных 3 Мбайт на жестком диске. Сама Ensemble занимает в памяти только 100 Кбайт. Конечно, ее разработчикам для уменьшения потребностей в ресурсах ПЭВМ пришлось пойти на определенные жертвы.

Другие интерфейсные системы рассматривать не будем. Только отметим, что в конце 1989 г. небольшая голландская компания объявила о создании программных средств, обеспечивающих функционирование PM в среде DOS. Их поставка была запланирована на начало 1990 г. американской фирмой Cyco International.

Таблица 4.2

Характеристики многозадачных систем



Наименование																	
DESQview 386, версия 2.2	ОХ	У	ОХ	ОХ	ОХ	У	Х	ОХ	ОХ	ОХ	ОХ	ОХ	У	Х	Х	6,8	219,90
OS/2, версия 1.2	О	Х	ОХ	Х	У	ОХ	Х	У	У	Х	Х	ОХ	П	Н	П	6,1	340
Microsoft Windows, версия 3.0	ОХ	ОХ	Х	ОХ	ОХ	ОХ	О	ОХ	ОХ	ОХ	ОХ	Х	Х	ОХ	ОХ	7,4	149

Альтернативными названиями интерфейсных систем являются *интерфейс, (операционная) среда* и даже *ОС*.

4.3.2. Оболочки ОС

Оболочки ОС предоставляют пользователю качественно новый, по сравнению с реализуемым операционной системой, интерфейс и тем самым освобождают пользователя-непрофессионала от детального знания последнего. Эти сервисные системы существенно упрощают задание общеупотребимых действий и предлагают пользователю ряд дополнительных услуг. В общем, оболочки заметно повышают уровень пользовательского интерфейса, наиболее полно удовлетворяя потребности пользователя. Вместе с тем пользователь-профессионал не может считать себя свободным от кропотливого изучения соответствующего интерфейса ОС, так как существующие оболочки не могут полностью его заменить.

Большинство распространенных оболочек, характеризующихся универсальностью предоставляемого интерфейса, обеспечивают:

1) *работу с файлами и каталогами,* в том числе:

— манипулирование файлами, а именно: создание, копирование, пересылку, переименование, удаление и быстрый поиск файлов по образцу составного имени файла (имени и расширения);

— выдачу и смену характеристик файлов (времени и даты создания, размера, прав доступа, т.е. атрибутов, и т.п.);

— выдачу содержимого каталогов в естественном порядке, а также в отсортированном по определенному критерию виде (например, по имени файла, расширению, дате и времени создания или размеру);

— выдачу части (фильтрацию) содержимого каталогов в соответствии с образцом составного имени файла;

— сравнение содержимого каталогов;

— выдачу файловой структуры в виде дерева;

— манипулирование каталогами, а именно: создание, удаление, переименование, а иногда — копирование и пересылку каталогов;

2) *просмотр* как текстовых *файлов* (в формате ASCII), так и файлов, подготовленных в специальных форматах различными популярными системными и прикладными программными продуктами, для чего используются соответствующие программы просмотра (визуализаторы);

3) *редактирование текстовых файлов* встроенным или внешним текстовым редактором;

4) *создание пользовательских меню* для упрощения запуска часто используемых системных и прикладных программ;

5) *выдачу сведений* о размещении информации на дисках (например, о степени его занятости), а также об ОЗУ;

6) *доступ к пользовательскому интерфейсу ОС*, в частности, для запуска на выполнение системных и прикладных программ;

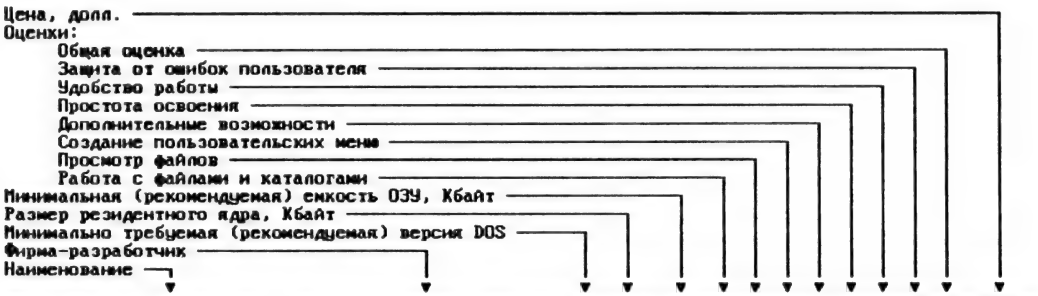
7) *освобождение* большей части *занимаемой памяти* при запуске внешней программы (в ОЗУ остается лишь небольшое резидентное ядро) и автоматическое *восстановление состояния* оболочки после завершения выполнения этой программы.

Возможна реализация и других дополнительных функций. Всем оболочкам присуща та или иная степень защиты от ошибок пользователя, что, например, может уменьшить вероятность непреднамеренного (случайного) удаления файлов.

Результаты сравнительной оценки известных оболочек *универсального типа* для DOS, а также их требования к ПЭВМ представлены в табл. 4.3 (данные взяты из журнала «Мир ПК»).

Таблица 4.3

Характеристики оболочек для DOS



Наименование	Фирма-разработчик	Минимально требуемая (рекомендуемая) версия DOS	Размер резидентного ядра, Кбайт	Минимальная (рекомендуемая) емкость ОЗУ, Кбайт	Работа с файлами и каталогами	Просмотр файлов	Создание пользовательских меню	Дополнительные возможности	Простота освоения	Удобство работы	Защита от ошибок пользователя	Общая оценка	Цена, долл.
Disk Director, версия 1.06	Athena Software	2.0	6	384 (640)	У	У	У	У	Х	У	Х	У	79
Magellan, версия 1.0	Lotus Development	2.1	10	512	У	ОХ	Х	Х	Х	ОХ	У	Х	195
MS-DOS Shell, версия 4.01	Microsoft	4.01	3	256 (512)	У	У	ОХ	У	ОХ	ОХ	Х	Х	150
Norton Commander,версия 3.0	Peter Norton Computing	2.0	13	256	Х	ОХ	ОХ	ОХ	ОХ	ОХ	Х	ОХ	149
1 Dir Plus, версия 3.03E	Bourbaki	2.0	7	256	ОХ	П	ОХ	ОХ	У	У	У	Х	95
PC Shell (PC Tools Deluxe), версия 6.0	Central Point Software	3.0 (3.2)	10	512 (640)	ОХ	ОХ	Х	О	ОХ	ОХ	ОХ	ОХ	149
Q-DOS II, версия 2.0	Gazelle Systems	2.0	40	256	У	П	Н	У	ОХ	У	Х	У	79,95
Xtree Pro Gold, версия 1.3	Xtree	3.1	7	384	О	Х	У	Х	ОХ	ОХ	ОХ	ОХ	129

Все рассматриваемые оболочки реализуют группы функций, перечисленные выше, но имеют различия в качестве их воплощения.

Из табл. 4.3 видно, что наилучшими показателями в совокупности обладают оболочки PC Shell и Norton Commander. Наибольшую же популярность приобрела последняя, так как она:

— лучше удовлетворяет потребности пользователей-непрофессионалов, а их значительно больше;

— обладает бо́льшим удобством в работе, требуя меньшего количества ответов пользователя, правда, иногда в ущерб степени защиты от ошибок;

— предъявляет менее жесткие требования к оборудованию ПЭВМ и ОС;

— может (с некоторыми ограничениями) удовлетворительно функционировать на ПЭВМ без НЖМД.

PC Shell же имеет оверлейную структуру, вследствие чего требует постоянного наличия «своего» диска в дисководе, а на оставшемся втором дисководе выполнить многие действия не представляется возможным. Поэтому PC Shell следует размещать только на жестком диске.

Заметим, что уже объявлено о создании комплекта утилит PC Tools 7.0.

Развитием Norton Commander'а является оболочка Pie Commander.

Для работы с файлами и каталогами наилучшей является система XTree Pro Gold (последняя версия — 2.0), специально разработанная как раз с этой целью.

Отметим, что оболочка Disc Director имеет и сетевую версию — LAN Director. Система Magellan требует обязательного наличия НЖМД. Оболочка 1Dir Plus может быть использована для работы с сетевыми системами Banyan Vines, фирмы Novell и IBM PC LAN, а PC Shell — с сетями Token Ring и фирмы Novell.

Оболочка MS-DOS Shell поставляется совместно с DOS. Пользовательский интерфейс оболочки из DOS 5.0 напоминает Windows.

Имеются и (частично) *специализированные* оболочки, которые обладают ярко выраженными специфическими функциями и вместе с тем слабой реализацией общих функций, чтобы удовлетворить лишь минимум требований. Примером такой оболочки является система IDCshell американской фирмы Infinity Design Consepts, ориентированная на *архивацию* (сжатие) файлов

и имеющая уникальные средства по отображению файлов с Escape-последовательностями. Кроме того, она обеспечивает печать файлов в различных режимах. Однако гораздо более мощной, чем IDCshell, является оболочка SHEZ, обеспечивающая как создание архивов, так и извлечение файлов из них, причем поддерживающая практически все используемые в настоящее время типы архивов. Как IDCshell, так и SHEZ функционируют в среде DOS.

В СНГ разработаны заслуживающие внимания оболочки CP (Command Processor) и Виктория.

В качестве альтернативных названий оболочки ОС применяются термины *«командная оболочка»* и *«операционная оболочка»*.

4.3.3. Утилиты

Утилиты предоставляют пользователям часто необходимые им услуги, реализация которых иначе потребовала бы разработки специальных программ.

Многие из утилит обладают развитым диалоговым интерфейсом с пользователем и приближаются по уровню общения к оболочкам. Остальные же используются путем их запуска с определенными аргументами.

Существующие в настоящее время утилиты обеспечивают реализацию таких (но не всех сразу) основных функций, как:

1) *обслуживание МД*, в том числе:

— форматирование дисков в различных режимах, причем часто с возможностью последующего восстановления информации, если форматирование выполнено по оплошности;

— обеспечение сохранности системной информации на МД и возможности ее восстановления в случае разрушения;

— восстановление ошибочно удаленных файлов и каталогов, а также содержимого файлов и каталогов в случае его разрушения;

— низкоуровневое редактирование информации на дисках;

— дефрагментация файлов на МД, вследствие чего время доступа к файлам сокращается на величину до 30% и облегчается восстановление информации в случае ее разрушения;

— надежное затирание на диске конфиденциальной информации, после чего ее прочтение становится невозможным ни при каких условиях;

2) *обслуживание файлов и каталогов* (аналогично оболочкам, но зачастую предоставляемые утилитами возможности изощреннее);

3) *создание и обновление архивов* как со сжатием, так и без сжатия (т.е. просто с дублированием) информации, а также извлечение файлов из них;

4) *предоставление пользователю информации* о:

— персональном компьютере (его ресурсах);

— распределении памяти на МД (размещении файлов, фрагментации, свободном пространстве);

— распределении ОЗУ между программами;

5) *шифрование информации*;

6) *печать содержимого* текстовых и других *файлов* в различных режимах и форматах;

7) *защита от компьютерных вирусов*;

8) *выполнение вычислительных работ* (по принципу калькулятора).

Этот список можно было бы продолжить в определенном смысле до бесконечности, но мы не будем этим увлекаться, так как в любом случае все функции утилит в полном объеме не охватить.

Некоторые из перечисленных функций требуют пояснений, что мы сейчас и сделаем.

Утилиты архивации позволяют создать *резервные копии* файлов путем помещения их в *архив* (часто в сжатом виде, в результате чего экономится память). Сжатие обеспечивается путем перекодирования с тем, чтобы заменить более коротким кодом наиболее часто используемые последовательности битов и/или байтов. Возможна даже переменная длина кодов символов. Вся информация о перекодировании хранится в специально создаваемой таблице. Лучше всего сжатию поддаются текстовые файлы, хотя эффект достигается и на файлах с другим содержимым. Так, текстовый файл можно уменьшить в 3 раза, а исполняемый — на треть.

Архив может содержать несколько логически связанных файлов, что создает дополнительные преимущества по сравнению с обычным дублированием, поскольку в этом случае не нужно привлекать вспомогательные средства и прикладывать какие-либо усилия для объединения файлов в единое целое.

Современные средства архивации-разархивации, как правило, обеспечивают:

— создание архива;

— обслуживание архива (добавление файлов в архив, удаление файлов из архива, замену файлов в архиве, выдачу оглавления архива и т.п.);

— извлечение файлов из архива;

— автоматическую архивацию и разархивацию поддерева файловой структуры;

— защиту от несанкционированного доступа к архиву по паролю;

— создание саморазархивирующихся исполняемых файлов;

— тестирование целостности архивов;

— работу с частично разрушенными архивами.

Саморазархивирующийся исполняемый файл представляет собой сжатый файл с расширением EXE, объединеный с компактным разархиватором. В случае запуска такого файла осуществляется разархивация его содержимого.

Каждый архиватор обычно реализует свой собственный уникальный алгоритм сжатия.

Среди архиваторов-разархиваторов для DOS широко известны следующие:

— комплект утилит PKZIP, PKUNZIP и др. (версии 1.2) американской фирмы PKWARE, поддерживающий ZIP-формат;

— архиватор-разархиватор LHarc 1.13 фирмы Yoshi, хорошо уплотняющий даже исполняемые файлы и создающий при желании саморазархивирующиеся файлы (используется LZH-формат);

— архиватор-разархиватор LHice 1.14, разработанный Haruyasu Yoshizaki и обслуживающий ICE-архивы; он совместим с утилитой LHarc;

— архиватор-разархиватор LHA 2.05, являющийся развитием утилиты LHarc и совместимый с ней снизу вверх;

— архиватор-разархиватор ARJ 2.20, разработанный Р.Янгом и поддерживающий одноименный формат;

— утилиту разархивации NARC версии 3.1 (1989 г.) фирмы Infinity Design Concepts, обладающую хорошим интерактивным интерфейсом и способную работать с ZIP-, а также с устаревшими ARC-архивами.

Высокий рейтинг имеет комплект PKZIP — PKUNZIP, в основном, благодаря высокому быстродействию при приемлемой степени сжатия, а также обеспечению высокой надежности хранения архива и возможности его восстановления после повреждений. Хорошими характеристиками обладает утилита LHA. Максимальная степень сжатия и наилучшие возможности присущи архиватору ARJ. Он в состоянии даже создавать архив на нескольких дискетах, что позволяет использовать его для резервирования содержимого жесткого диска вместо специально предназначенных для этого средств. ARJ обладает и возможностью восстановления частично разрушенных архивов.

Представляют интерес еще два типа архиваторов.

Первые из них сжимают исполняемые файлы, оставляя их по-прежнему исполняемыми. Распаковка осуществляется только при запуске программы на выполнение, причем это практически не снижает производительность ПЭВМ. Пользователь даже может не знать о том, что исполняемые файлы сжаты и их размеры в результате этого уменьшены примерно на 30%. Лучшим в данном классе является архиватор LZEXE, созданный во Франции.

Архиваторы *другого* типа резидентно размещаются в памяти и фильтруют всю информацию, записываемую на жесткий диск и считываемую с него. При записи осуществляется сжатие, а при чтении — распаковка информации. Их использование ведет к повышенным накладным расходам по времени и оперативной памяти, но зато реально увеличивает емкость жесткого диска примерно вдвое, так как сжимаются все, а не только исполняемые файлы.

Стандартные же архиваторы служат только для резервирования файлов, затрудняя доступ к ним, так как перед использованием файл следует извлечь из архива явным образом.

Утилиты, обеспечивающие *защиту от компьютерных вирусов,* выполняют целый ряд функций по предотвращению «заражения» вирусом, поиску вирусов и их удалению. Они классифицируются и рассматриваются в соответствующем разделе.

Как правило, разработчики предлагают многофункциональные или специализированные *комплекты* утилит. Отдельные утилиты распространяются реже.

Среди многофункциональных наиболее совершенным и поэтому признанным лучшим за 1988 г. для DOS является комплект утилит Norton Utilities фирмы Peter Norton Computing, последняя версия 6.0 которого выпущена уже под маркой компании Symantec в 1991 г. Он по прежнему лидирует в своем классе. Разноплановые утилиты содержит и высококачественный комплект PC Tools Deluxe 7.0, в состав которого входит упоминавшаяся оболочка PC Shell.

Специализированные комплекты утилит обеспечивают реализацию одной из перечисленных выше групп функций.

4.4. Инструментальные системы

Инструментальной системой будем называть совокупность программного продукта, обеспечивающего разработку информационно-программного обеспечения, и формальных языков, поддерживаемых этим продуктом.

В данном подразделе мы сделаем обзор основных *типов* инструментальных систем, используемых на ПЭВМ.

4.4.1. Системы программирования

Под **системой программирования** понимают совокупность *языка программирования* и *виртуальной машины*, обеспечивающей выполнение на реальной машине программ, составленных на этом языке.

Языком программирования называют систему обозначений, служащую в целях точного описания алгоритмов для ЭВМ или по крайней мере достаточную для автоматического нахождения такого алгоритма. Эти языки являются *искусственными языками* со строго определенным *син-*

таксисом и *семантикой*, поэтому они не допускают свободного толкования конструкций, характерного для естественного языка (языка общения между людьми).

Виртуальная машина — это программный комплекс, *эмулирующий* работу реальной машины с определенным *входным языком* на ЭВМ с другим, *машинным языком*, а иными словами, *реализующий* входной язык программирования. Такая техника реализации языка программирования позволяет сделать последний удобным для использования человеком. Виртуальная машина содержит *транслятор* и/или *интерпретатор* и может включать *библиотеки стандартных подпрограмм, отладчик, компоновщик* и другие *сервисные средства*.

Транслятор представляет собой программу, осуществляющую перевод текстов с одного языка на другой. В системе программирования транслятор переводит программу с входного языка этой системы на *машинный язык* реальной ЭВМ (на которой функционирует данная система программирования или будет функционировать разрабатываемая программа) либо на *промежуточный язык* программирования, уже реализованный или подлежащий реализации. Одной из разновидностей транслятора является **компилятор**, обеспечивающий перевод программ с языка *высокого уровня* (приближенного к человеку) на язык более *низкого уровня* (близкий к ЭВМ), или *машинозависимый* язык. Другая разновидность транслятора — **ассемблер**, осуществляющий перевод программ с языка *низкого уровня* (языка Ассемблера) на *машинный* язык, имеющий примерно тот же уровень. Некоторые трансляторы служат для переноса программ с одной машины на другую. Программа, подающаяся на вход транслятора, называется **исходной**, а результат трансляции — **объектной программой**. Использование трансляторов обеспечивает высокую скорость выполнения полученных с их помощью программ, однако затрудняет и удлиняет процесс отладки программных продуктов. Последнее обстоятельство объясняется относительно большим временем трансляции и сложностью создания отладчиков, которые должны отображать машинную программу в программу на исходном языке.

Диаметрально противоположными характеристиками обладает альтернативное средство реализации языка — *интерпретатор*. **Интерпретатор** представляет собой программный продукт, выполняющий предъявленную программу путем одновременного ее анализа и реализации предписанных ею действий. При использовании интерпретатора отсутствует разделение на две стадии (*перевод* и *выполнение*) и, более того, отсутствует явный перевод программы даже по частям перед очередным этапом выполнения. В действительности же распознается очередная конструкция программы и интерпретатором выполняются определяемые ею действия. После этого процессы анализа и реализации предписанных действий циклически повторяются. Таким образом, при интерпретации реально выполняется только программа-интерпретатор, управляемая исходной программой и, естественно, исходными данными для последней.

Возможны и смешанные стратегии реализации языков программирования, например, трансляция на промежуточный язык с последующей интерпретацией промежуточной программы.

Вырожденной является система программирования, поддерживающая машинный язык. В этом случае в качестве основного средства его реализации выступает сама ЭВМ, которую в отличие от *программного* интерпретатора можно рассматривать как *аппаратный* интерпретатор.

Главным классифицирующим признаком языков и, следовательно, систем программирования, является принадлежность к одному из оформившихся к настоящему времени *стилей программирования*, основные среди которых — *процедурное, функциональное, логическое и объектно-ориентированное*.

Стилю (парадигме) программирования соответствует своя собственная (*уникальная*) *модель вычислений*.

На ПЭВМ реализованы представители всех стилей программирования. И вообще следует заметить, что быстрее всего вновь созданные языки воплощаются в системы программирования именно для ПЭВМ, с обеспечением при этом сервиса высочайшего уровня.

Существуют два подхода к конструированию систем программирования:

1) подход, при котором целью является создание *комплекса автономных средств*, в совокупности выполняющих роль системы программирования;

2) подход, при котором создается *интегрированная среда программирования*, поддерживающая развитый пользовательский интерфейс и объединяющая в единое целое все средства разработки и выполнения программ (текстовый редактор, компилятор, компоновщик, отладчик и библиотеки стандартных подпрограмм).

Автономные средства позволяют разрабатывать программы большего размера, так как различные этапы разработки программы разнесены во времени, а поэтому сами автономные средства не обязаны одновременно находиться в ОЗУ. Кроме того, можно интенсивно использовать внешнюю память для хранения промежуточных и окончательного результатов ее обработки. Однако применять такие средства неудобно. *Интегрированная* же *среда* предоставляет лучший сервис в работе, но предъявляет и более жесткие требования к памяти (или к размеру разрабатываемой программы, что эквивалентно).

Ведущими разработчиками систем программирования в настоящее время являются американские фирмы Borland International и Microsoft.

Первая из них предлагает системы, содержащие как автономные средства, так и интегрированные среды. Торговой маркой продуктов этой фирмы стала приставка Turbo (а для последнего изделия — BORLAND) к названию системы программирования, совпадающему с наименованием

входного языка, например: Turbo Pascal. Дословно же «Turbo» означает «быстрый», что соответствует действительности.

Фирма Microsoft раньше предлагала только мощные автономные средства, а сравнительно недавно вступила в конкурентную борьбу с фирмой Borland, выпустив интегрированную среду Quick Pascal 1.0 и ей подобные с другими входными языками. Между прочим, Quick тоже означает «быстрый».

Системы программирования предлагаются и многими другими фирмами-разработчиками ПО.

Ниже мы кратко охарактеризуем основные стили программирования и рассмотрим относящиеся к ним системы программирования, доступные на ПЭВМ.

Процедурное программирование

Процедурное (императивное) программирование является отражением архитектуры традиционных ЭВМ, которая была предложена фон Нейманом в 40-х гг. Теоретической моделью процедурного программирования служит алгоритмическая система под названием «машина Тьюринга».

Программа на процедурном языке программирования состоит из последовательности операторов (инструкций), задающих те или иные действия. Основным является оператор присваивания, служащий для изменения содержимого областей памяти. Вообще концепция памяти как хранилища значений, содержимое которого может обновляться операторами программы, является фундаментальной в императивном программировании.

Выполнение программы сводится к *последовательному* выполнению операторов с целью преобразования *исходного состояния* памяти (т.е. значений переменных) в *заключительное*. Таким образом, с точки зрения программиста имеется программа и память, причем первая последовательно обновляет содержимое последней.

Процедурные языки характеризуются:

— значительной сложностью;

— отсутствием строгой математической основы;

— необходимостью явного управления памятью, в частности необходимостью описания переменных;

— малой пригодностью для символьных вычислений;

— высокой эффективностью реализации на традиционных ЭВМ.

Из-за наличия *побочных эффектов* (т.е. взаимного влияния различных программных модулей через общую память) программы на таких языках трудно читаемы, плохо модифицируемы и трудно проверяемы, а следовательно, ненадежны. По этой же причине они предполагают лишь последовательное выполнение.

Символьные вычисления состоят в преобразовании *динамических структур данных*, т.е. структурированных объектов, конфигурация которых меняется во времени (в отличие от **числовых вычислений**, предполагающих обработку данных *статической структуры*). К символьным вычислениям относятся задачи искусственного интеллекта, сортировки, трансляции, интерпретации, управления базами данных, символической алгебры и др. Символьные вычисления имеют ряд *особенностей:*

— последовательность исполняемых инструкций сильно зависит от данных;

— как правило, не требуется векторных и матричных операций;

— наиболее частыми операциями являются *вызовы процедур, агрегирование* и *декомпозиция* структур данных, *поиск* по дереву и т. д.;

— основной управляющей конструкцией служит рекурсия на структурах данных, таких, как *деревья, списки, множества*.

Одним из важнейших классификационных признаков процедурных языков является их уровень. **Уровень** языка программирования определяется семантической (смысловой) емкостью его конструкций и его ориентацией на программиста-человека. Язык программирования (частично) ликвидирует семантический разрыв между методами решения задач человеком и машиной. Чем более язык ориентирован на программиста, тем выше его уровень. Распределение основных реализованных на ПЭВМ императивных языков по уровням приведено на рис. 4.4. Ряд языков программирования на нем не представлен. Среди них FORTH, который трудно классифицировать по уровню, а также PL/1, Cobol, RPG, Logo, Snobol и GPSS, в настоящее время не пользующиеся большой популярностью.

Двоичный язык является не чем иным, как непосредственно машинным языком. В настоящее время такие языки программистами не применяются.

Шестнадцатеричный язык обеспечивает некоторое упрощение записи программы на машинном языке путем представления четырех двоичных цифр одной шестнадцатеричной. Этот язык используется в качестве дополнения к языкам высокого уровня, таким, как Pascal, для программирования критичных к времени выполнения фрагментов алгоритмов.

Язык Ассемблера — это язык, предназначенный для представления в удобочитаемой символической форме программ, записанных на машинном языке. Он позволяет программисту пользоваться мнемоническими кодами операций, по своему усмотрению присваивать символические имена регистрам ЭВМ и ячейкам памяти, а также задавать наиболее удобные в том или ином контексте схемы адресации. Кроме того, язык Ассемблера обеспечивает представление констант в различных системах счисления (например, в десятичной или шестнадцатеричной).

Язык детализированных схем программ — это не язык программирования, а язык представления алгоритмов при разработке программ, некогда широко используемый. В связи с достаточно

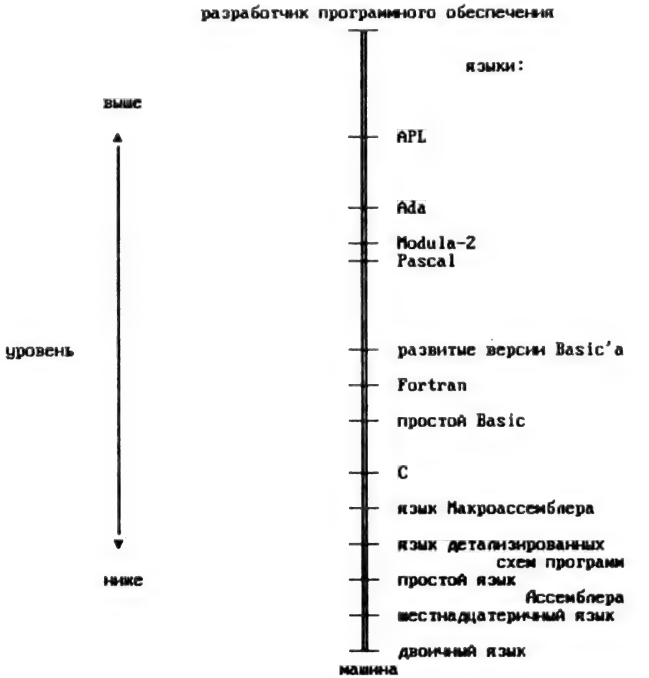


Рис. 4.4. Уровни процедурных языков программирования

низким уровнем этот язык в настоящее время на практике почти не применяется, а предпочтение отдается более развитым языкам проектирования программных комплексов. Мы показали язык детализированных схем программ на рисунке лишь затем, чтобы стало понятным его место в иерархии процедурных языков программирования.

Язык Макроассемблера является расширением языка Ассемблера за счет включения макросредств. С их помощью предоставляется возможность описывать в программе последовательности инструкций с параметрами *(макроопределения)* и использовать снабженные аргументами *макрокоманды*, которые автоматически замещаются в процессе ассемблирования *макрорасширениями*, представляющими собой макроопределения с подставленными вместо параметров аргументами. Иными словами, язык Макроассемблера предоставляет средства определения и использования новых, более мощных команд как последовательностей базовых инструкций, что несколько повышает его уровень.

Языки Ассемблера и Макроассемблера применяются системными программистами-профессионалами с целью использования всех возможностей оборудования ЭВМ и получения эффективной, как по времени выполнения, так и по потребному объему памяти, программы.

Язык программирования C первоначально разработан для реализации ОС UNIX в начале 70-х гг. ее создателями и в последующем приобрел высокую популярность среди системных, а также (частично) среди прикладных программистов. В настоящее время этот язык реализован в рамках большинства ОС ЭВМ и особенно ПЭВМ. В C сочетаются достоинства современных высокоуровневых языков (в части управляющих структур, а также структур данных) и возможность доступа к аппаратным средствам машины на уровне, который обычно ассоциируется с языком Ассемблера. C имеет синтаксис, обеспечивающий чрезвычайную краткость программ, а компиляторы вследствие особенностей языка способны генерировать весьма эффективный объектный код. Этот язык разработан на базе языка BCPL; промежуточной версией, которая просуществовала недолго, был язык B. В языке C воплощены основные концепции языка Pascal (но не только они). Одна из наиболее существенных особенностей языка C сотоит в нивелировании различий между *выражениями* и *операторами*, что приближает его к функциональным языкам. В частности, выражение может обладать побочным эффектом (эффектом присваивания), а также может использоваться в качестве оператора. Нет также четкой границы между *процедурами* и *функциями*. Более того, понятие процедуры вообще не вводится. Вместе с тем синтаксис языка таков, что затрудняет программирование и понимание составленных программ. Отсутствует также строгая типизация, что не способствует получению надежных программ.

Basic (Beginner's All-purpose Symbolic Instruction Code — многоцелевой язык символических инструкций для начинающих) представляет собой простой язык программирования, разработанный

в 1964 г. для использования новичками. История Basic'а такова. Еще в 1962 г. профессор математики Дортмутского колледжа Т.Куртц предложил директору, чтобы студенты изучали вычислительную технику в течение всех четырех лет обучения. Но ЭВМ тех лет не позволяли реализовать эту идею, так как компиляция программ осуществлялась непомерно долго, а пользователи при этом не присутствовали. Для получения (как правило) ошибочного листинга требовалось несколько дней. Поэтому было решено разработать простейший язык для непосредственного общения человека с машиной. Вот почему работа в среде Basic'а первоначально велась только в режиме *интерактивной (диалоговой) интерпретации*. В настоящее время имеются и компиляторы с этого языка. На ранних этапах своего развития Basic применялся только для обработки числовых величин; позднее он был расширен средствами обработки строк. Basic по концепциям, заложенным при его разработке, в смысле строгости и стройности является антиподом языка Pascal. В частности, в нем широко используются разного рода умолчания, что считается плохим тоном в большинстве современных языков. Несмотря на это, Basic очень популярен, в особенности на ПЭВМ. Существует множество его диалектов, несовместимых между собой. Не без оснований этот язык иногда сравнивают с питоном, заглатывающим и переваривающим все новое, что появляется в других языках программирования. Поэтому Basic является одним из наиболее динамичных, а его уровень нельзя определить однозначно. Современные диалекты Basic'а весьма развиты и мало чем напоминают своего предка.

Язык Fortran был разработан в 1956 г. сотрудником фирмы IBM Дж.Бэкусом. В 1958 г. появилась версия Fortran-II, которую затем сменил язык Fortran-IV, стандартизованный в 1966 г. Американским национальным институтом стандартов (ANSI) и поэтому называемый также Fortran 66. Он стал поистине «рабочей лошадью» научных работников и широко используется в настоящее время, несмотря на его ограниченность и «корявость». Со временем появилась следующая версия языка Fortran-77, а в 1988 г. — новые версии — Fortran 8x, содержащая ряд черт функциональных языков программирования, и Fortran 88. Отметим, что в версии Fortran-II впервые была реализована идея *раздельной компиляции модулей*, что дало возможность создавать, в частности, библиотеки научных подпрограмм. И наконец, подчеркнем, что вследствие своей простоты язык Fortran позволяет сгенерировать достаточно эффективный машинный код.

Одним из наиболее популярных, среди прикладных программистов, процедурных языков программирования вообще и тем более для ПЭВМ является **язык Pascal**. Он разработан в 1970 г. швейцарским специалистом в области вычислительной техники профессором Н.Виртом, назван в честь французского математика и по замыслу автора предназначался для обучения программированию. Однако язык получился настолько удачным, что стал одним из основных инструментов прикладных, а зачастую и системных программистов при решении задач как вычислительного, так и информационно-логического характера. Эволюция Pascal'я привела к появлению множества несовместимых его диалектов, в той или иной мере дополняющих концепции Вирта, что потребовало стандартизации языка. В настоящее время существуют три Pascal-стандарта: британский стандарт BS6192:1982 г., международный стандарт ISO 7185:1983 г., идентичный предыдущему, и ANSI-стандарт. В Pascal'е реализован ряд концепций, в настоящее время рассматриваемых как основа для «дисциплинированного» программирования и заимствованных разработчиками многих других языков. Одним из наиболее существенных новшеств в Pascal'е является последовательная и в определенном смысле полная реализация концепций *структурного программирования* не только путем упорядочения связей между фрагментами программы по управлению, но и за счет структуризации данных. Так, в Pascal'е, по всей видимости, впервые воплощена концепция определения новых *типов данных*. Этот язык в отличие от C является *строго типизированным*. Pascal характеризуется:

— достаточно высоким уровнем;
— богатыми выразительными возможностями;
— стройностью и простотой;
— строгостью, способствующей написанию надежных программ;
— достаточно высокой *эффективностью реализации* (т.е. возможностью компиляции программ в быстро выполняющийся машинный код приемлемого размера).

Следующим созданным Н.Виртом в 1979 г. языком был **язык Modula-2**, первоначально предназначенный для ПЭВМ Lilith. Modula-2 — это, по существу, развитие Pascal'я. Его особенности состоят в высокой *модульности программ*, наличии средств *описания параллельных процессов*, а также механизмов *синхронизации*. *Модули* предполагают наличие секции *интерфейса* и *реализации*, а также поддерживают *механизм экспорта/импорта*, что в совокупности и обеспечивает высокую степень независимости модулей. Подобно C в нем присутствуют низкоуровневые средства. Поддерживаются также развитые средства абстракции данных.

В настоящее время Modula-2 реализован на большинстве ПЭВМ и для различных ОС. Наиболее полно возможности языка могут проявиться только при использовании многозадачной ОС (например, OS/2 или UNIX) на ПЭВМ, допускающей многопрограммную работу.

В 1988 г. Н.Вирт предложил новый язык под названием **Oberon**, который уже реализован на АРМ с МП NS32032 в виде интегрированной среды программирования. Он является прямым потомком языка Modula, но проще и мощнее последнего, однако не имеет средств асинхронного программирования. Дополнительная его особенность состоит во введении концепции *расширяемых типов*.

Язык Ada разработан в 1979 г. ведущими специалистами в области программирования по заказу Министерства обороны США для использования во встроенных системах с управляющими ЭВМ, что требует поддержки режима *реального времени*. ANSI-стандарт этого языка принят в 1983 г., а с 1986 г. он стал обязательным для многих военных приложений. Язык назван в честь Августы Ады Лайвлейс,

которая была ассистентом Ч.Бэббиджа и по праву считается первым в мире программистом. Несмотря на то, что основное назначение языка Ada — написание больших программных систем реального времени, не исключается, конечно, его использование и при решении задач вычислительного характера, системного программирования, параллельной обработки и т.п. Поэтому он вполне может рассматриваться как *универсальный* язык программирования. Язык Ada — это современный язык программирования, содержащий такие возможности паскалеподобных языков, как определение типов, развитые управляющие структуры и подпрограммы. Более того, в нем собраны многие теоретические достижения, полученные после 1970 г. Язык поддерживает *логическую модульность*, для которой данные, типы и подпрограммы могут быть пакетами. *Физическая* же *модульность* достигается раздельной компиляцией. Программирование в реальном масштабе времени обеспечивается за счет механизма распараллеливания и обработки исключений. Системное программирование поддерживается путем организации доступа к системно-зависимым параметрам и управления точностью при представлении данных. Следует отметить также, что данный язык вводит достаточно строгую дисциплину программирования, чем препятствует написанию «плохих» программ. Несмотря на достоинства, программистов отталкивает его громоздкость, и этот язык прокладывает себе дорогу с большим трудом. Некоторые же средства языка принципиально неэффективны в смысле потребностей в ресурсах при их реализации. В сравнении с языком Modula-2 язык Ada имеет несколько более богатые выразительные возможности, но существенно сложнее его.

Язык программирования APL (A Programming Language) был создан Иверсоном в 1969 г. и сразу получил широкое распространение. К числу его основных преимуществ относятся богатый набор мощных операторов, позволяющих работать с многомерными массивами как с единым целым, а также предоставление пользователю возможности определять собственные операторы. Для записи встроенных операторов используются в основном одиночные символы из набора специальных знаков. Поэтому программы на языке APL очень компактны и вместе с тем зачастую малопонятны. Основное его назначение — обработка массивов.

Многими разработчиками ПО для ПЭВМ используется гибкий и достаточно простой **язык FORTH** (четвертый), разработанный Ч.Муром в 1971 г. Программа на этом языке имеет вид строк в обратной польской записи (сначала записываются операнды, а затем — операция). Основной структурой данных при организации вычислений является *стек (магазин)*. Важная особенность FORTH'а — его *открытость (расширяемость)*. Программист может легко добавлять новые операции, типы данных и операторы. Последнее достигается путем связывания любой строки программы с заданным программистом словом, которое затем может использоваться наравне со стандартными операторами. Однако расширение словаря ведет к снижению эффективности программы. FORTH позволяет программисту получать полный доступ ко всем средствам ЭВМ. В этом смысле он сочетает в себе (как и C) достоинства языков высокого и низкого уровней. Язык FORTH используется для решения, в основном, задач информационно-логического, а не вычислительного характера, и хорошо зарекомендовал себя при работе в режиме реального времени. Сама система FORTH очень компактна.

Некогда популярный на больших ЭВМ мощный, но неоднородный конгломерат языков Fortran, Algol и Cobol под названием **PL/1** в настоящее время, видимо, безвозвратно сдал свои позиции и на ПЭВМ практически не используется, хотя и имеется ряд ограниченных его реализаций. Этот язык достаточно сложен и имеет такие свойства, которые не стимулируют написание корректных, надежных и наглядных программ. Кроме того, его реализация ресурсоемка и поэтому неэффективна.

Ориентированным на обработку *коммерческой* информации является **язык Cobol**, разработанный в 1960 г. и за период своего существования, как многие языки, претерпевший ряд изменений. В настоящее время используются стандарты Cobol-74, Cobol-68 и Cobol-85.

Применяется также **язык SNOBOL**, предназначенный, главным образом, для обработки *текстовых* данных. Он включает в себя мощные операторы для сравнения фрагментов текста и поиска заданных цепочек символов. Существующей на сегодняшний день версией SNOBOL IV предусматриваются средства для работы и с другими типами данных. Фактически это в достаточной степени универсальный язык с ярко выраженными средствами обработки текстов.

С целью *обучения* детей в 1967 г. разработан и используется по настоящее время **язык LOGO**. Он отличается простотой, но весьма богатыми возможностями, среди которых процедуры, графические средства и др.

Специально для *генерации отчетов* служит **язык программирования RPG** (Report Program Generator). Вход в систему RPG состоит из описания структуры файла, спецификации нужной информации и расположения ее на странице. На основе этой информации система RPG строит программу для считывания файла, извлечения из него нужной информации и переформатирования ее требуемым образом.

Интересные возможности предоставляет **система GPSS**, ориентированная на моделирование систем массового обслуживания.

Представляет значительный интерес и *параллельный* **язык OCCAM**, используемый для программирования транспьютеров английской фирмы Inmos, которые, как уже отмечалось, могут выполнять роль акселераторов для ПЭВМ.

По не вполне ясным причинам **языки семейства Algol** на ПЭВМ не представлены. Algol-60 был разработан коллективом известных в то время в области программирования специалистов

и составил непосредственную конкуренцию Fortran'y. Он отличался от Fortran'а в лучшую сторону, однако марка фирмы IBM сделала в данном случае свое «черное» дело и последний стал доминирующим. Версия Algol-68 обладала богатым набором средств, высокой строгостью и стройностью, но и большой сложностью. Ее реализация затянулась на долгие годы, в результате чего время было упущено и данный язык в некотором смысле пережил самого себя (проще сказать — устарел), практически прекратив свое существование.

Сведения об известных авторам системах процедурного программирования для ПЭВМ в алфавитном порядке названий языков приведены в табл. 4.4. В графе «Способ реализации языка» использованы сокращения К, А и И для *компиляции, ассемблирования* и *интерпретации* соответственно. В графе же «Тип системы программирования» буквы А и И соответственно обозначают *автономные средства* и *интегрированную среду программирования*. Сведения о системах программирования для UNIX-подобных систем неполны.

Подавляющее большинство систем процедурного программирования основано на трансляции, что является естественным следствием отражения ими архитектуры современных ЭВМ, в том числе ПЭВМ. Это обстоятельство как раз и обеспечивает возможность перевода программ на машинный язык.

Расставим некоторые акценты и сделаем ряд замечаний, не нашедших отражения в приведенной таблице.

Система Artek Ada включает практически весь стандарт языка. Остальные же Ada-системы ограничиваются лишь его подмножеством.

Продукт APL PLUS (STSC) совершеннее системы APL (IBM).

Для реализации языка Ассемблера используются утилиты DEBUG и LINK, включенные в состав DOS. Первая из них позволяет ассемблировать исходные программы с получением объектного модуля и отлаживать исполняемые (загрузочные) модули, а вторая — компоновать объектные модули в единый загрузочный модуль.

Системы BASIC, BASICA, GW-BASIC и QBASIC также являются составными частями DOS и называются утилитами, хотя это и несправедливо. Достоинства продукта BASICA — предоставляемые программисту графические возможности и средства работы с дисками. А в принципе системы BASICA и GW-BASIC аналогичны, но разработаны различными фирмами. QBASIC входит в состав MS-DOS 5.0 и представляет собой заметный шаг вперед в сравнении с GW-BASIC. Система True BASIC реализует ANSI-стандарт языка 1984 г.

Среди С-систем наиболее мощной является BORLAND C++. Эта система, являясь развитием системы Turbo C(++), поддерживает языки С и С++ и предназначена для разработки программ, работающих в среде DOS или Windows. Сам же язык С++ — это усовершенствованный язык С, в который включены объектно-ориентированные средства. В С++ имеются также другие особенности. Вообще фирма Borland International прекратила работы по всем языкам программирования, за исключением С(++), Pascal и языка Макроассемблера. Зато оставшиеся системы — лучшие в своем классе.

Богатыми возможностями обладает также Microsoft C Compiler 6.0.

Многие системы, такие, как Turbo C и WATCOM C, имеют хорошую совместимость с ANSI-стандартом языка 1989 г. Дополнительной особенностью продукта WATCOM C 6.5 является то, что он генерирует машинные коды с наилучшими показателями по скорости их работы. Компилятор PCC отличается от CC для UNIX своей мобильностью. Особенности назначения компиляторов для ОС AIX состоят в следующем:

CC — основной компилятор;
FCC — для использования сопроцессора плавающей точки;
VCC — для получения объектных файлов в формате VRM;
SCC — для получения автономных программ.

Среди Макроассемблеров снова лучшим является изделие фирмы Borland International — Turbo Assembler. Он поддерживает работу как в *режиме MASM*, обеспечивая совместимость с системой Macro Assembler фирмы Microsoft, так и в *режиме IDEAL*, т.е. в собственном улучшенном режиме.

Отличается хорошими характеристиками система TopSpeed Modula-2 для DOS и OS/2, как и другие изделия фирмы Jensen & Partners International. Она содержит высокоскоростной оптимизирующий компилятор, интерфейс с пользователем типа «меню», многооконный многофайловый текстовый редактор и быстрый «интеллектуальный» компоновщик.

Недавно фирма Jensen & Partners International предложила систему TopSpeed 3.0, имеющую модульную структуру и позволяющую работать на четырех языках (Modula-2, Pascal, С и С++), просто меняя компиляторы. В основе системы лежат модули TopSpeed Environment (среда программирования для DOS и OS/2) и TopSpeed TechKit. С помощью последнего можно создавать программы для Windows и PM.

Среди Pascal-систем лидером является Turbo Pascal 6.0. А конкурирует с ней Quick Pascal 1.0.

Интегрированный вариант Turbo Pascal 6.0 содержит:

— многооконный *текстовый редактор* для подготовки программ и данных;

— быстрый *компилятор*;

— «интеллектуальный» *компоновщик*, обеспечивающий удаление из программы при формировании загрузочного модуля неиспользуемых подпрограмм, что важно для воплощения концепции модулей (см. ниже);

— превосходный интерактивный *отладчик,* позволяющий выполнять программу по частям и после каждого этапа выполнения просматривать, а также изменять значения переменных, причем точки останова могут быть условными;
— развитый *контекстно-зависимый интерактивный справочник*;
— богатые *библиотеки стандартных подпрограмм (стандартные модули).*

Таблица 4.4

Системы процедурного программирования для ПЭВМ

Тип системы программирования
Требуемая ОС
Способ реализации языка
Фирма-разработчик
Наименование системы программирования
Язык

1	2	3	4	5	6
Ada	ALSYS PC-AT ADA	Alsys	К	DOS	н/д
	Artek Ada	Artek	К	DOS	н/д
	Janus	RR Software	К	CP/M	н/д
	Janus/ADA	RR Software	К	DOS	н/д
	Supersoft Ada	Supersoft	К	CP/M	н/д
APL	APL	IBM	К	DOS	н/д
	APL PLUS	STSC	К	DOS	н/д
Язык Ассемблера	Assembler	Microsoft, IBM	А	DOS	А
Basic	Turbo Basic 1.1	Borland International	К	DOS	И
	Thoroughbred BASIC	Concept Omega	И	DOS	И
	Pro-Basic	DEC	н/д	н/д	н/д
	CBASIC-86	Digital Research	И	CP/M	н/д
	CBASIC-86 Compiler	Digital Research	К	CP/M	н/д
	Utah BASIC	Ellis Computing	И	DOS	н/д
	PopBASIC 3.0	Hedge Systems	И&К	DOS	И
	BASICA	IBM	И	DOS	А
	BASIC-80	Microsoft	И	CP/M	н/д
	BASIC Compiler 6.0	Microsoft	К	DOS, OS/2	А
	GW-BASIC	Microsoft	И	DOS	А
	QBASIC	Microsoft	И	DOS	А
	Quick BASIC 4.5	Microsoft	И-К	DOS	И
	BASIC	Microsoft, IBM	И	DOS	А
	387BASIC	Microway	К	DOS	н/д
	BASIC Professional	Morgan	н/д	н/д	н/д
	Better BASIC	Summit Software Technology	К	DOS	н/д
	True BASIC 2.1	True Basic	К	DOS, Apple DOS, Macintosh OS, Amiga	И
	Watcom BASIC	Watcom Products	И&К	DOS	н/д
	ZBASIC	Zedcor	К	DOS, CP/M	н/д
C	Turbo C 2.0	Borland International	К	DOS	А&И
	Turbo C++ 2.0	Borland International	К	DOS	А&И
	BORLAND C++ 3.0	Borland International	К	DOS(+Windows)	А&И
	C 86 PLUS 1.20d	Computer Innovations	К	DOS	А
	DeSmet DC88 3.1	C-Ware	К	DOS	А
	DR C	Digital Research	К	CP/M	н/д
	Eco-C88 4.15	Ecosoft	К	DOS	А
	CC	IBM	К	AIX	н/д
	FCC	IBM	К	AIX	н/д
	VCC	IBM	К	AIX	н/д
	SCC	IBM	К	AIX	н/д
	TopSpeed C	Jensen&Partners International	К	DOS	И
	TopSpeed C OS/2	Jensen&Partners International	К	OS/2	И
	TopSpeed C++	Jensen&Partners International	К	DOS	И
	TopSpeed C++ OS/2	Jensen&Partners International	К	OS/2	И
	Lattice C 6.0	Lattice	К	DOS, OS/2	А
	Power C 1.2	MIX Software	К	DOS, CP/M	А

Таблица 4.4 (продолжение)

1	2	3	4	5	6
	Aztec C86 4.1c	Manx Software Systems	К	DOS, CP/M, Apple DOS, Macintosh OS	А
	Let's C	Mark Williams Company	К	DOS	н/д
	High C 1.61	MetaWare	К	DOS, OS/2	А
	C Compiler 6.0a	Microsoft	К	DOS, OS/2	А
	Quick C 2.0	Microsoft	К	DOS	И
	Quick C Compiler 2.0 with Quick Assembler	Microsoft	К	DOS	И
	C	The Santa Cruz Operation	К	Xenix System V	
	WATCOM C 8.0	WATCOM Product	К	DOS	А&И
	Zortech C 1.06	Zortech	К	DOS	А
	Zortech C++ 2.18	Zortech	К	DOS	А&И
	CC	н/д	К	UNIX 2.9BSD, UNIX 4.2BSD, ДЕМОС-2, ДЕМОС-32	н/д
	PCC	н/д	К	то же	н/д
Cobol	Level II Cobol	Digital Research	К	CP/M	н/д
	Utah COBOL 5.0	Ellis Computing	К	DOS	н/д
	Cobol Compiler	IBM	К	DOS	н/д
	Cobol/2	MicroFocus	К	DOS, OS/2+PM	н/д
	Cobol 3.0	Microsoft	К	DOS, Xenix, OS/2	н/д
	Cobol 80	Microsoft	К	CP/M	н/д
FORTH	80x8x Forth 2.21	H.Laxen&M.Perry	К	DOS	И
	PC/Forth	Laboratory Microsystem	К	DOS	н/д
	Neon	Kriya Systems	К	Apple DOS, Macintosh OS	н/д
	Forth	MPI	К	CP/M	н/д
	Forth	Supersoft	К	CP/M, DOS	н/д
Fortran	AC/FORTRAN-77	Absoft	К	Macintosh OS	н/д
	FORTRAN-77	DEC	К	н/д	н/д
	Utah FORTRAN	Ellis Computing	К	DOS	н/д
	Lahey FORTRAN F77L	Lahey Computer Systems	К	DOS	н/д
	FORTRAN Compiler 5.0	Microsoft	К	DOS, OS/2	А
	FORTRAN-77	Microsoft	К	DOS	н/д
	FORTRAN-77	PECAN	К	DOS, Amiga	н/д
	Pro-FORTRAN	Prospero	К	DOS	н/д
	FORTRAN-66	Supersoft	К	CP/M	н/д
GPSS	GPSS/PC	Minuteman Software	И	DOS	н/д
LOGO	Logo	Apple Computer	И	Apple DOS	н/д
	Object LOGO	Coral Software	И	Apple DOS	н/д
	DR Logo	Digital Research	н/д	CP/M	н/д
	Logo	IBM	н/д	DOS	н/д
Язык Макроас-семблера	Object Assembler	Apple Computer	А	Apple DOS	н/д
	AVMAC	Avocet Systems	А	DOS	н/д
	Turbo Assembler 2.5	Borland International	А	DOS	А
	Macro Assembler 5.1	Microsoft	А	DOS, OS/2	А
Modula-2	TopSpeed Modula-2	Jensen&Partners International	К	DOS	И
	TopSpeed Modula-2 OS/2 1.20	Jensen&Partners International	К	OS/2	И
	Modula-2/86	Logitech	К	DOS, Xenix	н/д
	Modula-2 OS/2 1.00	Logitech	К	OS/2	н/д
	Modula-2	PECAN	К	DOS,Amiga, Macintosh OS	н/д
	Modula-2	Volition	К	DOS	н/д

Таблица 4.4 (окончание)

1	2	3	4	5	6
	Professional Modula-2	Stony Brooks Software	К	OS/2	н/д
Pascal	Object Pascal	Apple Computer	К	Apple DOS	н/д
	Turbo Pascal 6.0	Borland International	К	DOS	А&И
	Turbo Pascal for Windows	Borland International	К	DOS+Windows	А&И
	Pascal/MT+86	Digital Research	К	CP/M	н/д
	Utah PASCAL	Ellis Computing	К	DOS	н/д
	Pascal	IBM	К	DOS	н/д
	TopSpeed Pascal	Jensen&Partners International	К	DOS	И
	TopSpeed Pascal OS/2	Jensen&Partners International	К	OS/2	И
	Pascal Compiler 4.0	Microsoft	К	DOS, OS/2	А
	Quick Pascal 1.0	Microsoft	К	DOS	И
	UCSD PASCAL	PECAN	К	DOS,Amiga, Macintosh OS	н/д
	Pro-Pascal	Prospero	К	DOS	н/д
PL/1	PL/1-80	Digital Research	К	CP/M	н/д
	PL/M	Intel	К	ISIS-11	н/д
	PL/M-86	Intel	К	DOS	н/д
	PL/Z	Zilog	К	CP/M	н/д
RPG	RPG II Compiler	Lattice	К	DOS	н/д
SNOBOL	SNOBOL 4	Betstis International	К	DOS	н/д

Все это объединено единым удобным многооконным интерфейсом типа «меню». Сам же язык Turbo Pascal 6.0, базируясь на ANSI-стандарте, развивает его следующим образом:

— поддерживает средства объектно-ориентированного программирования;
— имеет дополнительные стандартные типы;
— реализует аппарат типизированных констант;
— поддерживает новые операции;
— имеет средства приведения типов;
— включает ряд усовершенствованных операторов управления;
— имеет усовершенствованный механизм передачи аргументов подпрограммам;
— имеет непосредственный доступ к средствам ОС, портам ввода-вывода, ОЗУ и видеопамяти;
— предоставляет возможность включения в программу машинных кодов и инструкций языка Ассемблера, а также комплексирования программ с ассемблерными модулями;
— имеет дополнительные средства управления динамической памятью, что позволяет, в частности, создавать и обрабатывать динамические массивы;
— допускает разработку многомодульных и оверлейных программ с раздельной компиляцией модулей (оверлейная программа загружается в ОЗУ по частям, что снимает проблему ограниченности его объема);
— трактует модуль как библиотеку описаний, а не просто как подпрограмму, что перекликается с концепцией модуля в языке Modula-2, обеспечивая высокую степень независимости фрагментов программы;
— благодаря наличию семейства стандартных программных модулей допускает использование дополнительных предопределенных констант, типов, переменных и подпрограмм;
— имеет объектно-ориентированную среду разработки прикладных программ, обеспечивающую создание оконных пользовательских интерфейсов;
— имеет мощную графическую подсистему, позволяющую формировать и выводить на экран дисплея изображения;
— имеет средства условной компиляции, позволяющие автоматически формировать текст программы;
— предоставляет возможность управлять генерацией объектного кода с помощью директив.

В сравнении с охарактеризованной системой среда Quick Pascal обладает следующими особенностями:

— не имеет средств создания оверлейных программ;
— не поддерживает возможность размещения загрузочного модуля в ОЗУ (а не в файле на диске), что не позволяет сократить время выполнения вновь разработанной или откорректированной программы;

— объектно-ориентированные расширения языка не обеспечивают *раннее* (на этапе компиляции) «связывание» объектов со своими методами, что также снижает время выполнения программ (Turbo Pascal же обеспечивает как *раннее*, так и *позднее*, в ходе выполнения программы, «связывание»).

Подчеркнем, что обе охарактеризованные системы поддерживают язык Pascal, дополненный объектно-ориентированными средствами, и поэтому в равной степени могут быть отнесены к системам объектно-ориентированного программирования. Мы их рассмотрели здесь лишь потому, что интегрированные среды Pascal'я без таких расширений фирмами Borland и Microsoft в настоящее время не предлагаются.

Отметим, что система TopSpeed Pascal также поддерживает объектно-ориентированные средства.

Что касается языка PL/1, то все представленные системы реализуют лишь ограниченные подмножества языка.

Несмотря на наличие отладчиков в системах программирования, фирмы Borland и Microsoft предлагают также автономные отладчики Turbo Debugger 2.5 и Codeview 2.2 соответственно, обладающие дополнительными возможностями. Первый отладчик во всех отношениях лучше второго. Turbo Debugger 2.5 позволяет осуществлять интерактивную отладку программ на языках Pascal, C(++) и Ассемблера как в терминах исходных языков, так и в терминах машинных кодов. Данным отладчиком обеспечивается даже обратное выполнение программ (откат назад). Имеется великолепный пользовательский интерфейс.

Фирма Borland International дополнительно предлагает новый продукт — Turbo Profiler, позволяющий оценить эффективность программы и отдельных ее фрагментов.

Функциональное программирование

Кратко и точно сущность **функционального (аппликативного)** программирования определена А.П.Ершовым как «... способ составления программ, в которых единственным действием является вызов функции, единственным способом расчленения программы на части является введение имени для функции, а единственным правилом композиции — оператор суперпозиции функции. Никаких ячеек памяти, ни операторов присваивания, ни циклов, ни, тем более, блок-схем, ни передачи управления».

Аппликативное программирование *базируется* на одной из первых алгоритмических систем — λ-исчислении А.Черча, а также родственном ему исчислении комбинаторов, предложенном советским математиком М.И.Шейнфинкелем еще в 1924 г. и развитом Х.Карри. Именно исчисление комбинаторов является исторически первой алгоритмической системой, хотя многие об этом забывают.

Роль *основной конструкции* в функциональных языках играет *выражение*. К выражениям относятся скалярные константы, структурированные объекты, функции, тела функций и вызовы функций. Функция трактуется как однозначное отображение из X^n в X, где X — множество выражений, что полностью соответствует понятию функции в математике.

Любой аппликативный язык программирования включает следующие *элементы:*

1) *классы констант*, которыми могут манипулировать функции;

2) *набор базовых функций,* которые программист может использовать без предварительного определения;

3) *правила построения новых функций из базовых*;

4) *правила формирования выражений* на основе вызовов функций.

Программа представляет собой совокупность описаний функций (возможно, вложенных) и выражения, которое необходимо вычислить. Оно *вычисляется* посредством *редукции* (т.е. серии упрощений) до тех пор, пока это возможно, по следующим правилам:

1) вызовы базовых функций заменяются соответствующими значениями;

2) вызовы небазовых функций заменяются их телами, в которых параметры замещены аргументами.

Так как в выражении одновременно могут присутствовать несколько вызовов функций, то операционная семантика должна определять и *стратегию вычисления*. В рамках теории установлено, что любая стратегия вычисления ведет к получению одного и того же результата, если вычислительный процесс обрывается. Поэтому вызовы функций можно вычислять *одновременно,* что открывает широкие перспективы по автоматическому распараллеливанию программ.

Следовательно, функциональное программирование не использует концепцию памяти как хранилища значений переменных. Операторы присваивания отсутствуют, вследствие чего переменные обозначают не области памяти, а объекты программы, что полностью соответствует понятию переменной в математике. (В действительности можно составлять программы вообще без переменных.) Кроме того, нет существенных различий между функциями и константами, т.е. между программами и данными. В результате этого функция может быть значением вызова другой функции (функции порядка выше первого), и может быть элементом структурированного объекта. Число аргументов в вызове функции не обязательно должно совпадать с числом параметров при ее описании. При недостатке аргументов значением вызова будет функция, а при избытке — либо функция, либо константа, если такой вызов имеет смысл.

Перечисленные свойства характеризуют аппликативные языки как языки программирования *сверхвысокого уровня.*

Аппликативное программирование можно считать дальнейшим развитием идей *структурного программирования* за счет структуризации не только *управляющих связей* и *данных*, но и *информационных связей*. Действительно, процедурным языкам неотъемлемо присуща неупорядоченная связь по данным между различными участками программы. В аппликативной же программе любой программной функции аргументы могут быть переданы только явно, посредством вызова последней, а вычисление значения функции не способно привести к изменению значений каких-либо переменных. Это обеспечивает ясную иерархическую структуру программ (хорошую читабельность, проверяемость и совместимость), а следовательно, более высокую надежность. Функциональные языки отличаются своей простотой, легкостью реализации, компактностью представления алгоритмов, полностью автоматическим распределением памяти и пригодностью для символьных вычислений. Последнее объясняется тем, что по соображениям общности и эффективности реализации основными структурированными объектами в аппликативных языках являются *списки* (упорядоченные последовательности объектов, в том числе списков), удобные для символьной обработки; кроме того, в аппликативном программировании легко организуется рекурсивная обработка структурированных объектов. Числовые же вычисления — не та область, где ярко проявляются достоинства функционального программирования.

Наличие стройной математической основы обеспечивает возможность широкого использования алгебраических методов композиции, преобразования, исследования и верификации программ.

Эффективность реализации функциональных языков на традиционных ЭВМ в области символьных вычислений, во многом благодаря работам Т.Джонссона и Л.Аугустссона из Гетеборга, приближается к эффективности реализации процедурных языков, несмотря на то, что модель вычислений по функциональной программе и семантика машинного языка кардинально различны.

Кроме преимуществ аппликативных языков как языков программирования, они также привлекательны в качестве средств описания семантики других языков, а также в роли средств проектирования программных комплексов.

Установлено, что императивные языки по своим возможностям несколько уже аппликативных.

Самым первым функциональным языком явился LISP (List Processing — обработка списков), разработанный и реализованный группой авторов под руководством пионера в области искусственного интеллекта Дж.Маккарти в Массачусетском технологическом институте в 1959 г. Цель его создания состояла в удобстве обработки символьной информации. Существенная черта LISP'а — предельная унификация программных структур и структур данных (выражения записываются в виде списков). Этот язык является вторым по возрасту (после Fortran'а) языком программирования, широко используемым в настоящее время, и рассматривается специалистами как основной язык программирования систем искусственного интеллекта.

LISP непрерывно совершенствовался и сейчас существует множество развитых его версий с превосходной средой программирования, среди которых особо следует отметить INTERLISP и Common LISP.

В настоящее время имеется целый спектр машин для аппаратной поддержки языка LISP. Среди них — LISP-микропроцессор для военных приложений.

LISP послужил стартовой площадкой для разработки языков PLANNER и CONNIVER, которые используются для реализации процедурных моделей знаний.

Сейчас известно множество функциональных языков, значительно более мощных, чем LISP. Наиболее интересны и проработаны среди них ML (Milner Language) Милнера и Miranda Д.Тернера. Нам больше импонирует последний из-за его стройности и простоты. Перечислим особенности этих языков в сравнении с LISP'ом:

1) простота использования *функций высших порядков*;

2) подстановка аргументов функций в них производится не по значению, а *по необходимости* (последняя интегрирует достоинства подстановки по значению и по наименованию, что позволяет повысить быстродействие программ и манипулировать с потенциально бесконечными объектами);

3) описание функции представляется не единственным равенством (правилом, уравнением), а совокупностью *условных равенств*, накладывающих различные ограничения на структуру аргументов, что ведет к повышению компактности и наглядности программ. При вызове функции автоматически осуществляется выбор того или иного правила в соответствии с аргументами и декомпозиция последних;

4) реализация *функциональной трактовки ввода-вывода* и работы с файлами (в том числе с базами данных), что обеспечивается «ленивой» семантикой (подстановкой по необходимости);

5) поддержка *абстрактных типов данных*;

6) хороший *синтаксис*;

7) *непротиворечивость* в математическом смысле (противоречивость LISP'а хорошо известна, что затрудняет его параллельную реализацию).

С аппликативными тесно связаны *языки машин потока данных*. Среди последних известность приобрели VALID, VAL, ID и LUCID, причем по LUCID опубликовано полномасштабное руководство.

Разработка аппаратных и программных средств функционального программирования вошла составной частью в проекты ЭВМ пятого поколения.

Таким образом, функциональное программирование по сравнению с императивным, по крайней мере, в области символьных вычислений, имеет преимущества как с точки зрения пользователя, так и с точки зрения реализации (благодаря параллелизму).

В 1987 г. фирма IBM объявила о своей поддержке языка Common LISP и закупила на него все права. В связи с этим можно считать, что он уже стал стандартом языка LISP.

На ПЭВМ в настоящее время из функциональных языков широко используется только LISP и доступен Scheme («чистое» подмножество LISP'а, т.е. язык без императивных включений). Известны следующие их реализации для DOS:

1) система PC Scheme фирмы Texas Instruments, базирующаяся на компиляции;
2) система LISP компании Microsoft, также основанная на компиляции;
3) система LISP фирмы Norell, базирующаяся на интерпретации;
4) система Mulisp компании Software Ware House, поддерживающая ограниченный диалект LISP'а и основанная на интерпретации;
5) система Mulisp 85 фирмы Microsoft, основанная на интерпретации;
6) система IQ LISP компании Integral Quality, также основанная на интерпретации;
7) среда Byso Lisp фирмы Levien Instrument, включающая:
— интерпретатор Byso Lisp 1.17, совместимый с LISP 1.5, MACLISP и Common LISP;
— компилятор Byso Lisp 2.17;
— текстовый редактор;
— структурный редактор, обеспечивающий графическое отображение программ;
— средства печати программы в различных режимах, в частности, в режиме структуризации текста;
8) система Golden Common LISP американской фирмы Gold Hill Computers, поддерживающая несколько усеченный вариант языка Common LISP.

Для Macintosh OS существует система программирования Macsheme, удовлетворяющая требованиям стандарта Scheme.

Развитые версии LISP'а, в том числе Common LISP, поддерживаются в среде ОС UNIX.

Для эффективной работы с системой функционального программирования необходима мощная ПЭВМ, так как ресурсоемкость таких систем значительна.

Логическое программирование

За исключением нескольких результатов, логика и программирование долгое время были непересекающимися областями исследований. Еще в 1936 г. Тьюринг показал, что любая *вычислимая функция* может быть вычислена посредством *дедукции (вывода)* в *исчислении предикатов первого порядка*, существующий вариант которого построен Г.Фреге в 1879 г. Однако этот результат не нашел практического применения, так как не был известен алгоритм установления *общезначимости* (истинности во всех *интерпретациях) логической формулы*. К тому же было показано, что эта проблема *алгоритмически неразрешима*. В 1930 г. Френч и Эрбран независимо доказали фундаментальную теорему, предоставляющую алгоритм, который способен установить общезначимость логической формулы, если она является таковой. В противном случае этот алгоритм зацикливается. Поэтому проблему установления общезначимости в исчислении предикатов первого порядка стали считать *полуразрешимой*. Разработанный алгоритм также не нашел практического применения вследствие большой его трудоемкости.

В 1962 г. Робинсон предложил *метод резолюции*, который обладает существенно меньшей трудоемкостью, чем алгоритм Эрбрана, однако все еще не решает проблему комбинаторного взрыва. В этом методе используется единственное мощное *правило вывода* вместо множества правил, предложенных логиками. Метод резолюции стал применяться в автоматическом доказательстве теорем. Одновременно предпринимались усиленные попытки поиска стратегий вывода, сокращающих *пространство поиска*. На этом пути были достигнуты заметные успехи.

Основываясь на полученных к тому времени результатах, французский ученый А.Кольмероэ создал язык программирования PROLOG (PROgramming in LOGic — программирование в терминах логики), первоначально предназначенный для работы с естественными языками, и опубликовал его описание в 1973 г. Значительного сужения пространства поиска удалось достичь за счет использования логической системы Хорна, являющейся частью исчисления предикатов, и *линейной по входу* стратегии резолюции. Но даже при таких ограничениях оказалось возможным представлять любую вычислимую функцию. (Заметим, что логика Хорна шире теории реляционных баз данных.) Большое значение в становлении, развитии и осмыслении PROLOG'а имели работы Р.Ковальского.

Появление PROLOG'а открыло новую область исследований — **логическое, или реляционное,** программирование, где практические результаты зачастую предшествуют их теоретическому осмыслению и обоснованию.

Центральным понятием в логическом программировании является *отношение*. *Программа* представляет собой совокупность определений отношений между объектами (в терминах условий, или ограничений) и цели (запроса). *Процесс выполнения программы* трактуется как процесс установления общезначимости логической формулы, построенной из программы по правилам, установленным семантикой того или иного языка. *Результат вычисления* является побочным продуктом этого процесса. В реляционном программировании нужно только специфицировать факты, на которых алгоритм основывается, а не определять последовательность шагов, которые требуется выполнить. Это свидетельствует о *декларативности* языков логического программирования. Она метко выражена в формуле Р.Ковальского: «алгоритм = логика + управление».

В настоящее время известны и другие языки логического программирования.

Языки логического программирования характеризуются:
— *сверхвысоким уровнем*;
— жесткой ориентацией на *символьные вычисления* (числовая обработка затруднена);
— возможностью *инверсных вычислений* (переменные в вызовах «процедур» не делятся на входные и выходные), что является уникальной чертой реляционного программирования, хотя недавно разработан аналогичный подход и для функциональных языков;
— зачастую *логической неполнотой* в двух аспектах: невозможностью выразить в программе определенные логические конструкции, а также невозможностью получить из программы все правильные выводы.

К сожалению, инверсность вычислений больше декларируется, чем обеспечивается, что вызвано различными усовершенствованиями, делающими языки эффективнее, но разрушающими логическую семантику программ.

Логические программы отличаются принципиально низким быстродействием, так как вычисления осуществляются методом проб и ошибок (посредством поиска с возвратами), а также высокой степенью параллелизма. Однако организация параллельного исполнения затруднительна; сам же параллелизм требует чрезвычайно много ресурсов ЭВМ вследствие недетерминизма и поэтому считается «необузданным». Кроме того, один из основных механизмов реляционного программирования — *унификация* — имеет последовательную природу.

Таким образом, языки логического программирования являются достаточно мощными, но неэффективными, с точки зрения реализации, языками.

Тем не менее языки логического программирования играют центральную роль в проектах ЭВМ пятого поколения. Уже разработан и разрабатывается ряд архитектур с целью аппаратной поддержки реляционного программирования.

В настоящее время для ПЭВМ существует около пятнадцати реализаций PROLOG'а. Компиляция, хотя она и затруднена из-за большого семантического разрыва языка и ПЭВМ, предусматривается в следующих трех, функционирующих в среде DOS, системах: Arity/Prolog 5.0 фирмы Arity, Turbo Prolog 2.0 фирмы Borland International и Prolog-2 английской компании Expert Systems. Остальные же системы обеспечивают только интерпретацию PROLOG-программ. Prolog-2 уступает системе Arity/Prolog по ряду показателей, ни в чем его не превосходя. Однако есть вариант системы Prolog-2 для Windows 3.0.

Наиболее удачными системами считаются Arity/Prolog 5.0 и Turbo Prolog 2.0, оформленные в виде интегрированных сред. Небольшие разработки, видимо, лучше осуществлять при помощи Turbo Prolog'а, в то время как при работе с большими программными проектами лучше использовать Arity/Prolog, поскольку он включает более эффективный отладчик. Однако при необходимости осуществить компиляцию, а не интерпретацию, придется выйти из интегрированной среды и вызвать автономный компилятор. Входной же язык компилятора является всего лишь подмножеством входного языка интерпретатора. Эта система позволяет компилировать программы непосредственно в среде. Эта система содержит графические средства, а Arity/Prolog их не имеет. Остальные характеристики рассматриваемых двух систем примерно одинаковы.

К сожалению, работы по Turbo Prolog'у фирмой Borland International заморожены.

Интересен также модульный вариант Prolog'а — MPROLOG, разработанный в Венгрии. Его реализация ресурсоемка и поэтому он воплощен в систему программирования только для ПЭВМ на МП 80386(SX) и старше.

Объектно-ориентированное программирование

Прототипом **объектно-ориентированного** программирования послужил ряд средств, содержащихся в языке SIMULA-67. Но оформилось оно в самостоятельный стиль программирования в связи с появлением языка SMALLTALK, первоначально предназначенного для реализации функций машинной графики и разработанного А.Кеем. Первая версия этого языка создана в 1972 г. С тех пор построено несколько его версий.

Корни объектно-ориентированного программирования уходят в одну из ветвей логики, в которой первичным считается не отношение (как для логического программирования), а *объект*. По сравнению с исчислением предикатов объектно-ориентированные логические системы обладают более сложными синтаксисом и правилами вывода. С объектно-ориентированным программированием тесно связана *теория акторов*.

Основными особенностями объектно-ориентированных языков являются:
1) наличие активных *объектов (акторов)*;
2) формирование объектов путем *наследования свойств*;
3) посылка *сообщений* от объекта к объекту как механизм организации вычислительного процесса.

Суть данного стиля программирования выражается формулой «объект = данные + процедуры».

Итак, *объект* интегрирует некоторое *состояние* (или структуру данных) и доступные только ему *механизмы изменения этого состояния*. Для того чтобы модифицировать состояние некоторого объекта, необходимо послать ему соответствующее сообщение. Действие (или *метод*), выполняемое (выполняемый) адресатом сообщения, касается только его самого: другие объекты не должны знать, каким образом данный объект реализует ту или иную функцию. Объединение данных и процедур в объекте называется **инкапсуляцией,** и это свойство неотъемлемо присуще объектно-ориентиро-

ванному программированию. Многие развитые объектно-ориентированные языки программирования (Turbo Pascal, C++) обладают наряду с этим **полиморфизмом,** т.е. возможностью использования методов с одинаковыми именами для работы с данными различных типов.

Концепция объекта опирается на методы структурного программирования и методы разработки программ, основанные на абстракции данных.

Структурное программирование связано с функциональной декомпозицией и предполагает проектирование программного продукта «сверху вниз». Однако такой метод не позволяет учесть зависимость архитектуры программы от структур данных, которые ей придется обрабатывать.

Использование подхода, основанного на *абстракции данных*, ведет к противоположному эффекту: разработка программы осуществляется «от данных», а упор делается на выборе способа их представления. В этом случае, естественно, образуется разрыв между структурами данных и процедурами их обработки.

Объектно-ориентированное программирование позволяет ликвидировать противопоставление процедур данным и их неравноправность, свойственные двум описанным подходам, и одновременно с этим интегрирует достоинства рассмотренных методов разработки программ. Таким образом, объектно-ориентированное программирование поддерживает качественно новый уровень совместной структуризации данных и процедур их обработки.

Многие объектно-ориентированные языки имеют средства для объединения объектов в *классы*. Объекты, принадлежащие одному классу, называются его *примерами*. Это позволяет хранить процедуры (в терминах объектно-ориентированного программирования — методы), применимые ко всем примерам класса, в единственном экземпляре, только в соответствующем классе. Данный подход имеет очевидные преимущества: код размещается только в одном месте, чем экономится память и обеспечивается модифицируемость программы.

Ряд языков, включая SMALLTALK-80, C++, Turbo Pascal и Objective-C, развивают это понятие за счет того, что классы размещаются в узлах *дерева наследования* (свойств). Таким образом строится *иерархия* классов. При этом пример наследует свойства (процедуры-методы) своего класса, а также всех суперклассов этого класса. Методы, определенные в классе-корне дерева, наследуются всеми классами и их примерами. **Наследование** наряду с инкапсуляцией и полиморфизмом является важнейшим свойством объектно-ориентированного программирования.

Механизм наследования с древовидной структурой в определенной степени ограничен, так как во многих случаях разделение свойств в соответствии со строгой иерархией не может быть достигнуто. Поэтому некоторые языки допускают классы, которые наследуют свойства более чем одного непосредственного суперкласса, в результате чего иерархия классов становится направленным ациклическим графом, а не деревом. Такой механизм добавлен в качестве расширения в SMALLTALK. Подобные механизмы доступны также в основанных на LISP'е объектных системах Flavours, CommonLoops и CLOS.

Описанная модель дискретных объектов, взаимодействующих друг с другом, является хорошей основой для параллелизма. В зависимости от конкретизации объектно-ориентированной модели вычислений существуют три различных подхода к распараллеливанию. Наиболее перспективен из них следующий. Объект (называемый в данном случае *актором*), посылающий сообщение, не ждет ответа от своего адресата, а продолжает функционирование. Поэтому сообщения должны быть способны становиться в очередь к адресату. Одновременно может быть активизировано много акторов, причем все вычисления выражаются, главным образом, в терминах их коммуникаций. В этом случае накладные расходы на каждый процесс будут малы, вследствие чего даже небольшие объекты могут быть совместно исполняемыми, что и обеспечит наилучшую степень параллелизма. Языками данного класса являются система Astra, а также Act 1 и Act 3. В простейшем случае посылка сообщения эмулируется вызовом метода в том или ином объекте как процедуры.

SMALLTALK-системы обладают высокой мобильностью, так как в них первоначально осуществляется компиляция программы на промежуточный язык, а затем — интерпретация кода на этом языке.

Разработан язык Concurrent SMALLTALK, добавляющий асинхронные посылки сообщений к уже имеющимся в SMALLTALK'е синхронным, что обеспечивает лучший параллелизм.

Объектно-ориентированные языки находят применение, главным образом, при построении моделей, в том числе при создании языков представления знаний и реализации протоколов вычислительных сетей. Потенциальные же возможности объектно-ориентированных языков гораздо шире. Разрабатываются архитектуры ЭВМ для данной парадигмы.

Этот стиль программирования характеризуется богатыми графическими возможностями и средой программирования, развитой модульной структурой программ, но порою невысокой эффективностью реализации языков на непараллельных ЭВМ.

На модульности следует заострить особое внимание: именно она упрощает разработку сложных программных продуктов, а также облегчает их комплексирование, модификацию и сопровождение.

Интерес к объектно-ориентированному программированию в настоящее время быстро растет, что объясняется и что отчасти явилось причиной появления систем программирования Turbo Pascal 5.5 и 6.0, Turbo C++, а также BORLAND C++.

На ПЭВМ доступны следующие SMALLTALK-системы:

1) Smalltalk/V фирмы Digital для DOS;
2) Smalltalk-80 компании Apple Computer для Aplle DOS;

3) Smalltalk-80 фирмы Softsmarts для DOS;
4) Smalltalk PC компании Software Systems для DOS;
5) Smalltalk /V PM для OS/2 PM;
6) Smalltalk-AT с графическим интерфейсом и поддержкой «мыши» (для DOS);
7) Smalltalk/Object фирмы ParkPlace для OS/2 PM.

Для Windows 3.0 фирмой The Whitewater Group предлагается объектно-ориентированная система программирования Actor.

Известна также система C_talk для DOS, интегрирующая C и Smalltalk. Другие варианты интеграции процедурного и объектно-ориентированного программирования уже рассматривались в подпункте «Процедурное программирование» данного пункта.

В последнее время большой интерес проявляется к языку C++, созданному в 1983 г. Строуструпом. Этот язык совместим снизу вверх с ANSI-стандартом C, обладает в отличие от C жестким контролем типов и является достаточно сложным языком.

В данном пункте мы практически не затронули вопросы интеграции различных стилей программирования. Однако это не означает отсутствия таких работ и многообещающих их результатов. Дело в том, что нам практически неизвестны коммерческие программные продукты такого типа для ПЭВМ, за исключением уже упомянутых. Тем не менее имеется интегрированная среда программирования XSimp 2.0, поддерживающая все четыре рассмотренные парадигмы программирования, обладая при этом функциональным акцентом. Данная система открыта для расширений и предлагается за рубли на отечественном рынке. Имеются ее реализации для DOS и Macintosh.

4.4.2. Системы управления базами данных

Прежде чем ввести понятие системы управления базой данных (СУБД), дадим общее представление о банке данных (БД), для создания которого она используется и основным компонентом которого она является.

Неформально БД представляет собой хранилище информации для различных приложений, которая в настоящее время рассматриваемой как один из важнейших ресурсов.

Информация — это любые *сведения* о каком-либо событии, сущности, процессе и т.п., являющиеся объектом некоторых операций: восприятия, передачи, преобразования, хранения или использования.

Информация, зафиксированная в *определенной форме*, пригодной для последующей обработки, хранения и передачи представляет собой **данные**.

Очевидно, что понятия информации и данных формально различны. Тем не менее вне данного пункта мы не проводили и не будем проводить между ними четкой грани.

Банком данных называют программную систему, предоставляющую услуги по хранению, а также поиску данных определенной группе пользователей и по определенной тематике.

БД используются в АСУ, информационно-поисковых и справочных системах, системах автоматизации конторской деятельности, а также в системах искусственного интеллекта.

К БД предъявляются следующие *требования*:

1) удовлетворение информационных потребностей пользователей;
2) обеспечение возможности работы с большими объемами различной информации;
3) поддержка заданного уровня достоверности хранимой информации и ее непротиворечивости;
4) осуществление доступа к данным только пользователей, имеющих на это полномочия;
5) обеспечение возможности поиска информации по любой требуемой группе признаков;
6) соответствие заданным требованиям по производительности;
7) возможность реорганизации и расширения при изменении границ предметной области;
8) обеспечение выдачи информации в форме, удобной для восприятия;
9) простота использования;
10) возможность обслуживания нескольких (не обязательно одновременно) пользователей.

С БД взаимодействуют следующие *категории лиц*:

1) *конечные пользователи* (вводят и извлекают данные);
2) *программисты* (пишут и отлаживают программы обработки данных);
3) *администраторы* БД (отвечают за проектирование, реализацию, эксплуатацию и сопровождение БД).

Структура БД показана на рис. 4.5.

База данных (БзД) представляет собой совокупность специальным образом организованных наборов данных (*файлов*), хранимых во внешней памяти ЭВМ (обычно на МД).

СУБД — это программный продукт, обеспечивающий *централизованное управление данными* в БзД. Названную систему можно рассматривать как надстройку над средствами управления данными (файловой системой) ОС.

Преимущества *централизованного управления данными* состоят в следующем:

1) в сокращении избыточности хранимых данных;
2) в поддержании целостности и непротиворечивости данных;
3) в обеспечении многоаспектного использования данных, т.е. в поддержке новых приложений на основе уже имеющихся данных;

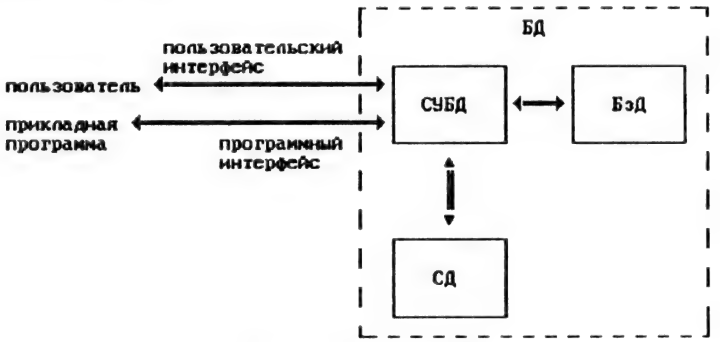


Рис. 4.5. Структура БД

4) в обеспечении комплексной оптимизации хранения и использования данных;
5) в стандартизации и унификации способов представления данных;
6) в разграничении доступа к данным различных групп пользователей.

Под *управлением данными* понимается:

1) *задание и коррекция схемы* БзД, т.е. ее логической и/или физической структуры;
2) *манипулирование данными,* т.е. добавление, обновление, удаление и выборку (извлечение) данных из БзД.

Для решения первой задачи служит **язык описания данных** (обычно диалоговый), а второй — **язык манипулирования данными.** Эти языки зачастую реализуются СУБД посредством интерпретации и реже — компиляции.

Язык манипулирования данными может быть *самостоятельным (автономным)* языком или *подязыком,* погруженным в один из широко распространенных языков программирования. Последний при этом называется **включающим,** а сам подязык содержит соответствующие подпрограммы и/или специальные операторы. Интерактивный язык манипулирования данными часто называют **языком запросов.**

Читатель должен уяснить, что в зависимости от стадии жизненного цикла БД СУБД выполняет одну из следующих двух функций:

1) *формирование* БзД, включая ее модификацию (на стадии создания БД);
2) *обработку данных* из БзД, включая ее модификацию (на стадии эксплуатации БзД).

В соответствии с *первой* функцией СУБД можно рассматривать как инструментальную систему. Коммерческие СУБД, предлагаемые различными фирмами, как раз и являются таковыми. Словом, Вы можете приобрести СУБД, создать с ее помощью БзД и получить тем самым БД, в который, естественно, будет входить и сама СУБД.

Часто фирмами-производителями предлагаются два варианта одной и той же СУБД:

1) полный вариант, обеспечивающий выполнение *обеих* вышеперечисленных функций;
2) усеченный вариант, называемый СУБД *времени выполнения* (Run-Time), который поддерживает только *вторую* функцию.

Такая стратегия сулит пользователям очевидные выгоды: СУБД времени выполнения существенно компактнее полной, да и работает зачастую быстрее последней.

Иногда (в частности, в системе Clipper) можно обойтись вообще без СУБД времени выполнения, если программа на языке манипулирования данными компилируется, а интерактивный режим работы с БД не обеспечивается.

СУБД может поддерживать один из следующих интерфейсов или одновременно оба:

1) *пользовательский интерфейс* для взаимодействия пользователя с БД в интерактивном режиме;
2) *программный интерфейс* для выполнения запросов по управлению данными из прикладных программ.

В действительности понятие пользовательского интерфейса существенно шире: в него, конечно же, входят и средства для управления самой СУБД со стороны пользователя.

Словарь данных (СД) представляет собой специальную информационную структуру, содержащую общие сведения о ресурсах БД. СД включает:

— *описание схемы и подсхем* БзД, т.е. сведения об общей организации БзД, а также о возможных (допустимых) значениях и форматах представления данных;
— *сведения о полномочиях пользователей* по управлению данными;
— *сведения об источниках данных*;
— другие *справочные сведения.*

Развитые СУБД обеспечивают независимость прикладных программ, к ним обращающихся, от конкретной организации данных. Это позволяет реорганизовать данные без модернизации соответствующих программ. Указанная независимость реализуется, как правило, в результате того, что СУБД поддерживает следующие три *уровня представления данных:*

1) физический уровень;

2) логический уровень;
3) концептуальный уровень.

Физический уровень определяет форматы размещения данных (логических записей) на внешних носителях информации.

Логический уровень является промежуточным и описывает взаимосвязи между логическими записями.

Концептуальный уровень определяет структуру БзД в терминах объектов предметной области и отношений между ними.

В результате разделения на уровни БзД можно сравнить с многослойным пирогом, причем концептуальный уровень является *внешним,* а два других — *внутренними.*

Каждый уровень описывается соответствующей схемой БзД. СУБД обеспечивает *преобразование (отображение)* данных одного уровня в данные другого уровня. Изменение одного из внутренних уровней не влияет на прикладные программы. Два же внутренних уровня введены с целью облегчения модификации самого БД.

Для представления данных на концептуальном уровне применяются различные *модели данных.* Под **моделью данных** понимают *формализм* (*формальную систему*), используемый для представления данных.

Известны такие модели данных:
1) иерархическая модель;
2) сетевая модель;
3) реляционная модель.

Первоначально наибольшее распространение получила *иерархическая* модель данных. В иерархической модели объекты предметной области представляются узлами (вершинами), а отношения между ними — дугами, связывающими вершины, причем получившийся в результате такого отображения граф должен удовлетворять ограничениям, налагаемым на деревья. Иерархическая модель характеризуется:
— недостаточной простотой;
— неоднородностью, затрудняющей манипулирование данными;
— отсутствием строгой математической основы;
— высокой эффективностью, ведущей к увеличению скорости манипулирования данными;
— неполнотой, так как не любая предметная область может быть представлена в этой модели;
— неравноправием между данными вследствие того, что одни из них должны быть жестко подчинены другим (т.е. асимметрией отношений);
— сложностью обновления БзД.

Часть недостатков рассмотренной модели устраняются в *сетевой* модели данных. Последняя является обобщением иерархической модели за счет того, что предметная область может представляться графом произвольного вида. Сетевой модели присущи:
— сложность;
— неоднородность;
— отсутствие строгой математической основы;
— невысокая эффективность;
— полнота;
— равноправие между данными;
— сложность обновления БзД.

В основе *реляционной* модели данных (от англ. relation — отношение) лежит понятие *отношения* между объектами предметной области, а сами отношения представляются *таблицами.* БзД при этом являет собой совокупность таблиц. Реляционная модель предложена Э.Ф.Коддом и должна была удовлетворять 15-ти требованиям, чтобы считаться таковой. В апреле 1990 г. Кодд усилил понятие реляционной модели, предъявив к ней уже 300 (!) требований. Реляционной модели свойственны:
— простота;
— однородность (единообразие) представления данных, что облегчает их обработку;
— наличие строгой и стройной математической основы, что позволяет создавать реляционно полные как *процедурные* (базирующиеся на реляционной алгебре), так и *декларативные* (опирающиеся на реляционное исчисление) языки манипулирования данными;
— низкая эффективность вследствие разобщенности семантически взаимосвязанных данных;
— полнота;
— равноправие между данными;
— легкость обновления данных.

Отметим, что реляционная алгебра формально *не обладает полнотой.* Одним из известных контрпримеров для подтверждения этого является невозможность выражения в ней транзитивного замыкания бинарного отношения.

Последнее десятилетие ознаменовало собой триумф реляцинной модели данных. На фоне этого сетевая модель применяется лишь изредка, а иерархическая модель практически «канула в лету». В течение ближайших пяти-десяти лет здесь, видимо, радикально ничего не изменится. Но уже сейчас поговаривают о преемнике реляционной модели, каковым считается *объектно-ориентированная* модель данных, не имеющая пока даже точного определения.

Перспективы развития архитектур БД связаны с созданием *распределенных* (размещенных фрагментарно по узлам вычислительной сети) БзД и управлением ими. Первым шагом в этом направлении явилась разработка архитектуры «клиент-сервер», где под *клиентом* понимается фронтальная программа, обеспечивающая взаимодействие с пользователем, а под *сервером* — тыловая программа, обеспечивающая централизованную обработку разобщенных данных. В действительности же БзД может считаться распределенной только тогда, когда пользователю не нужно будет знать, что она является таковой. На настоящее время ни одна из фирм не поставляет СУБД, обеспечивающих создание БД с указанным ограничением, хотя многие к этому стремятся.

Статусом стандартного реляционного языка манипулирования данными сейчас обладает язык SQL (Structured Query Language — структурированный язык запросов). В особенности это справедливо для сетевых приложений.

Среди СУБД для ПЭВМ, способных функционировать в среде DOS, в настоящее время наиболее известны следующие:

— dBASE IV 1.1 компании Ashton-Tate;
— Paradox 3.5 фирмы Borland International;
— R:base 2.11 компании Microrim;
— FoxPro фирмы Fox Software;
— Clipper 5.0 компании Nantucket;
— db_VISTA III корпорации Raima.

Все перечисленные СУБД, кроме db_VISTA III, основаны на реляционной модели данных, хотя реляционность dBASE весьма слабая, потому что поддерживаемый ею язык манипулирования данными далек как от реляционной алгебры, так и от реляционного исчисления. Система же db_VISTA базируется на сетевой модели.

Ранние версии **СУБД dBASE** были очень популярны. Сейчас же она утрачивает свое лидерство, хотя общее количество проданных ее экземпляров пока остается наибольшим. Здесь уместно отметить, что в 1991 г. компания Ashton-Tate куплена фирмой Borland International. Последствия этого очевидны. dBASE IV 1.1 характеризуется:

— наличием как командного пользовательского интерфейса, так и пользовательского интерфейса типа «меню» для управления СУБД;
— наилучшими средствами создания приложений (разработки программ на языке манипулирования данными), обеспечивающими порой даже их автоматическую генерацию;
— прекрасными средствами генерации отчетов;
— хорошим сервисом при вводе данных в таблицы (наличием соответствующих экранных форм);
— поддержкой интерактивного языка запросов типа QBE (Query By Example — запрос по примеру), посредством которого запросы к БзД специфицируются в экранных формах, являющихся заголовками таблиц, что облегчает формулирование запросов;
— поддержкой языка SQL;
— посредственным быстродействием при выполнении запросов к БзД;
— необходимостью наличия в составе ПЭВМ жесткого диска (в отличие от dBASE III Plus);
— наличием интерфейса с системами программирования C, Pascal и Ассемблер корпорации Microsoft.

Таким образом, dBASE использует три класса языков манипулирования данными:

1) *интерактивный язык* типа QBE;
2) *самостоятельный* (неинтерактивный) *язык* с поддержкой SQL;
3) *включающие языки* C, Pascal и Ассемблера.

В последнее время резко возросла популярность **системы Paradox** и особенно ее последней версии — 3.5. В 1990 г. она по объему продаж превзошла все аналогичные программные продукты и в том же году признана (после Windows 3.0) лучшим программным продуктом. Этой СУБД свойственны:

— широкие функциональные возможности;
— наличие пользовательского интерфейса типа «меню» для управления СУБД, но он уже морально несколько устарел и выглядит старомодно;
— относительно слабые средства создания приложений (язык PAL — Paradox Application Language), которые целесообразно применять для автоматизации только несложных предметных областей;
— интерпретация, а не компиляция PAL-программ, что отрицательно влияет на их быстродействие;
— хорошие средства генерации отчетов;
— хороший сервис при вводе данных в таблицы;
— поддержка великолепного интерактивного языка запросов типа QBE;
— поддержка языка SQL наряду с возможностью доступа к распределенной БзД посредством модуля Paradox SQL Link;
— высокое быстродействие при выполнении запросов к БзД, что достигается не за счет изощренного программирования самой СУБД, а путем использования новаторских архитектурных решений с задействованием дополнительной памяти ПЭВМ;
— наличие средств деловой графики, что является уникальным для программных продуктов данного класса;

— совершенство генератора отчетов;
— неудобный интерактивный справочник;
— необходимость наличия в составе ПЭВМ жесткого диска;
— низкие требования к другим ресурсам ПЭВМ;
— наличие интерфейса (модуль Paradox Engine 2.0) с системами программирования Microsoft C 5.1, Turbo C, Turbo C++ и Turbo Pascal, который используется для разработки сложных приложений.

Следовательно, аналогично dBASE системой Paradox поддерживаются три класса языков манипулирования данными.

Система R:base 2.11 является мощным инструментом при программировании, но она не в состоянии удовлетворить потребности интерактивных пользователей. Ей присущи:
— неудобный пользовательский интерфейс;
— поддержка языка манипулирования данными (с SQL-подобным синтаксисом) только через программный интерфейс;
— хорошее быстродействие (но по этому показателю R:base уступает Paradox'у).

Система FoxPro является преемником СУБД FoxBase Plus. Все изделия фирмы Fox Software — это аналоги СУБД dBASE, некоторые из показателей качества которых улучшены. Так, FoxPro, например, опережает dBASE IV 1.1 по быстродействию. Вообще же FoxPro считается одним из лучших вариантов выбора для профессионалов. Последняя версия системы — FoxPro 2.0 обладает великолепным быстродействием.

Система Clipper 5.0 представляет собой компилятор с языка манипулирования данными, очень близкого к собственному языку dBASE. По причине компиляции вместо интерпретации результирующий программный продукт обладает высоким быстродействием, а сама СУБД для его функционирования не требуется. Интерактивный язык запросов системой Clipper не поддерживается. Эта СУБД пользуется заслуженной популярностью среди высококвалифицированных программистов.

Система db_VISTA III имеет высокий рейтинг среди профессиональных С-программистов и поддерживает интерфейс только с этим языком (интерактивный язык запросов также отсутствует). Естественно, С при этом является включающим языком. Данную СУБД отличают высокое быстродействие и базирование не на реляционной, а на сетевой модели данных, что уже было отмечено. Для менее квалифицированных пользователей предлагается интерфейс типа «меню». Дополнительная к db_VISTA утилита db_QUERY позволяет эмулировать реляционную модель данных и поддерживать интерактивный язык запросов.

4.4.3. Инструментарий искусственного интеллекта

Искусственный интеллект (ИИ) — это молодая, бурно развивающаяся научная дисциплина, возникшая в 50-х гг. на стыке *кибернетики, лингвистики, психологии* и *программирования*. **Теорию ИИ** можно определить как науку о знаниях, о том, как их добывать, представлять в искусственных системах, перерабатывать внутри системы и использовать для решения практических задач.

ИИ восходит к работам Винера, Маккалока и Розенблата по *нейронным сетям*.

В настоящее время исследования в области ИИ проводятся по следующим направлениям:
— обработка естественного языка и моделирование диалога;
— экспертные системы (ЭС);
— автоматическое доказательство теорем;
— робототехника;
— интеллектуальные вопросно-ответные системы;
— автоматическое программирование;
— распознавание образов;
— решение комбинаторных задач (головоломки, игры).

Наибольший прогресс связан с созданием ЭС, которые уже получили достаточно широкое распространение и используются при решении практических задач.

Под **ЭС** будем понимать программу, в которую включены *знания* специалистов о некоторой предметной области и которая в пределах этой области способна принимать *экспертные решения*, т.е. заменить эксперта-человека. Таким образом, *главное отличие ЭС от обычных программ, также способных принимать экспертные решения, состоит в отделении декларативных знаний от процедурного компонента*, манипулирующего ими.

Так, с помощью ЭС PROSPECTOR открыты ранее не известные запасы молибдена; система же EURISCO произвела переворот в области создания сверхбольших интегральных схем, изобретя трехмерный узел типа И/ИЛИ.

Структура ЭС представлена на рис. 4.6. *Модуль вывода* (решатель) по запросу от пользователя, используя имеющиеся знания, осуществляет поиск ответа, причем этот поиск, как правило, сопровождается диалогом между пользователем и ЭС. Если решение задачи (ответ) у пользователя вызывает сомнения, то он может потребовать объяснений. Формально эту работу выполняет *подсистема объяснений*, на рисунке не показанная. Функционирование *модуля усвоения знаний*, если он представлен в ЭС, основывается на принципах индуктивного вывода, который находится в начальной стадии своего развития. Поэтому данный модуль в большинстве существующих ЭС отсутствует.

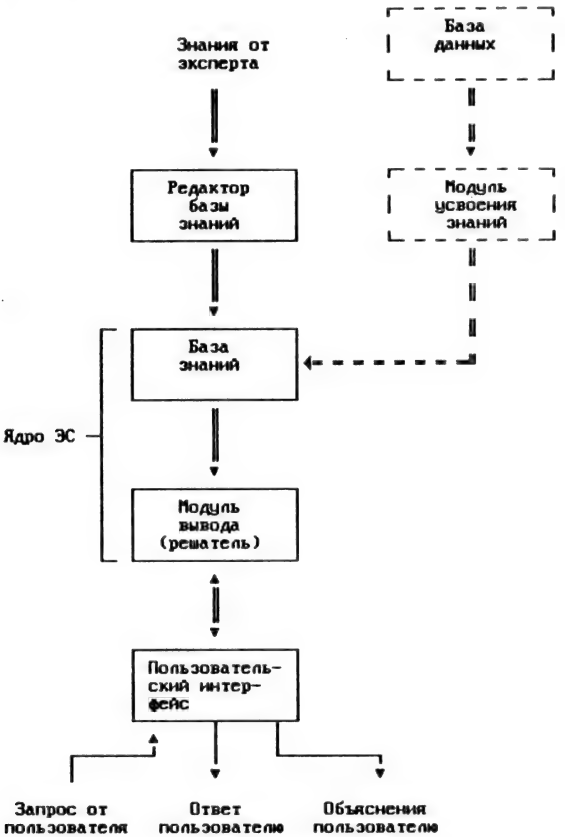


Рис. 4.6. Структура ЭС

ЭС применимы в достаточно широком спектре предметных областей, которые классифицируют по следующим *типам:*

— *интерпретация* (вывод описаний ситуаций из наблюдаемых данных);

— *прогнозирование* (вывод вероятных следствий из заданных ситуаций);

— *диагностика* (суждение о нарушениях в работе технической системы или организма человека по наблюдениям);

— *проектирование* (построение конфигурации объектов, которая удовлетворяет заданным ограничениям);

— *планирование* (построение плана действий для достижения заданной цели);

— *контроль* (предупреждение об «опасности», или нештатной ситуации);

— *отладка* (выдача рекомендаций по ликвидации плохого функционирования);

— *ремонт* (выполнение плана устранения некоторого обнаруженного дефекта);

— *обучение*;

— *управление* (адаптивное управление поведением некоторой системы).

Последнее совпадает с понятием *ситуационного управления*, которое продолжительное время развивается отечественными учеными.

По состоянию на конец 1984 г. только в США уже насчитывалось более 170 ЭС, предназначенных для работы в различных областях — в медицине, финансовой деятельности, юриспруденции, геологии, химии, электронике, вычислительной технике, математике, машиностроении, метеорологии, космических исследованиях, производстве, управлении, сельском хозяйстве, физике и военном деле. В 1987 г. лишь одна фирма IBM использовала 70 ЭС.

Применение ЭС, в частности, позволяет:

1) при проектировании ИМС повысить (по данным фирмы NEC) производительность труда в 3 — 6 раз, причем время выполнения некоторых операций сокращается в 10 — 15 раз;

2) ускорить поиск неисправностей в устройствах в 5 — 10 раз;

3) повысить (по данным фирмы Toshiba) производительность труда программистов в 5 раз;

4) в сфере профессиональной подготовки сократить без потери качества затраты на индивидуальную работу с обучаемыми.

Итак, важнейшая особенность систем ИИ состоит в том, что в рамках этих систем представляются *знания*, которыми они *манипулируют*. Вот почему системы ИИ часто называют *системами, основанными на знаниях*. Знания обладают рядом свойств, отличающих их от данных: это *внутренняя интерпретируемость, структурированность, связность* и *активность*. Особая роль знаний объясняется следующими причинами:

— отсутствием для большинства трудных и интересных задач алгоритмов решений, поскольку многие важные задачи возникают в сложных контекстах социальных и физических явлений, не поддающихся точному и строгому описанию;

— достижением людьми исключительно высоких результатов благодаря своим познаниям;

— признанием собственной ценности знаний как дорогого, но тиражируемого ресурса.

В настоящее время проработаны и используются на практике следующие *модели знаний*:

1) *логические исчисления,* в том числе исчисление предикатов первого порядка, псевдофизические, временная, пространственная, каузальная и нечеткая логики;

2) *семантические сети* (ориентированные графы, вершины которых представляют понятия и объекты, а дуги — отношения между ними, в какой-то мере аналогично сетевой модели данных);

3) *фреймы* (регулярные, или однородные разновидности семантических сетей);

4) *системы продукций,* т.е. правил вида «условие-действие», «если-то» и т.п.

Для каждой модели знаний имеются соответствующие *методы вывода решений* с их использованием. Возможны и смешанные модели знаний. В настоящее время наблюдается смещение акцентов в сторону систем продукций. Это объясняется тем, что вывод в них адекватен рассуждениям экспертов, а сами методы вывода хорошо проработаны. Есть, конечно, у систем продукций и другие достоинства, среди которых простота, однородность структуры продукций, локальность внесенных изменений в базу знаний (чем объясняется простота ее модификации) и т.п.

Интеллектуальные системы, в том числе ЭС, представляют собой сложные комплексы программ, осуществляющие, в основном, символьные вычисления. Поэтому для их создания предложены различные инструментальные средства, учитывающие как специфику переработки информации в системах ИИ, так и их структуру, чтобы облегчить их разработку.

Итак, **инструментарием ИИ** будем называть программные продукты и поддерживаемые ими языки, предназначенные для создания интеллектуальных систем.

Имеющиеся в настоящее время инструментальные средства для разработки систем ИИ делятся на нижеперечисленные *типы*:

1) скелетные системы (оболочки систем ИИ);

2) средства автоматизированного проектирования интеллектуальных систем;

3) системы представления знаний;

4) системы программирования.

Скелетная система представляет собой полномасштабную систему ИИ с пустой базой знаний, ориентированную на определенный круг приложений. Задача разработчика в этом случае сводится только к подготовке базы знаний. Каждая скелетная система характеризуется жестко зафиксированными способом представления знаний (моделью знаний), методом вывода решений и организацией взаимодействия своих компонентов.

Средства автоматизированного проектирования напоминают скелетные системы, но обладают большей гибкостью, предоставляя разработчику несколько вариантов организации базы знаний и функционирования системы. Поэтому данные средства называют «настраиваемыми оболочками».

Системы представления знаний поддерживают специальные языки для формального выражения знаний в той или иной модели, называемые **языками представления знаний.** В инструментальное средство данного типа входит также модуль вывода, причем разработчику предоставляются определенные возможности по управлению его работой.

При использовании *систем программирования* разработчику необходимо создавать полную инфраструктуру для интересующего приложения, т.е. разрабатывать собственный язык представления знаний и программировать все компоненты системы. Наиболее широко для этих целей используются непроцедурные языки программирования LISP, PROLOG и SMALLTALK. Благодаря эффективности своей реализации находят применение также и процедурные языки.

Типы инструментальных средств рассмотрены в порядке повышения универсальности и эффективности работы будущего продукта, а также увеличения затрат на разработку с их помощью системы ИИ.

Для ПЭВМ создано множество инструментальных средств различных типов. Информация о таких средствах для разработки ЭС, способных функционировать под управлением DOS, в алфавитном порядке сведена в табл. 4.5. Все перечисленные средства являются коммерческими, т.е. пригодными не только для использования их создателями, но и для продажи различным потребителям. Конечно, системы программирования не представлены, так как они уже рассматривались. В табл. 4.5 приняты следующие сокращения:

СС — скелетная система;
САП — средство автоматизированного проектирования;
СПЗ — система представления знаний.

Таблица 4.5

Инструментальные средства разработки ЭС для DOS

Наименование	Тип	Модель знаний
ACLS	СС	система продукций, примеры
Analyser Plus	СИП	система продукций
APES	СИП	система продукций
Autologie	СИП	система продукций
Cash Value	СПЗ	система продукций
CODES	СПЗ	фреймы
Cristal	СС	система продукций
ES/P Advisor	СС	система продукций
ESP Frame Engine	СС	фреймы
Exper-teach-II	СИП	система продукций
Expert-4	СС	система продукций, нечеткая логика
EXPERT-EASE	СС	система продукций, примеры
Expertkit	СС	система продукций
Exsys	СС	система продукций
Ex-TRAN-7 (Expert Translator)	СС	система продукций, примеры
First-Class	СС	система продукций
GURU	СИП	система продукций
Gold Hill Expert System Toolkit (Acorn Shell)	СИП	система продукций, фреймы
Insight-1	СС	система продукций
Insight-2	СС	система продукций
Intelligence Service (ISExpert Consultant)	СС	система продукций
KES	СИП	схемы,ассоциации, система продукций
KISS	СИП	система продукций
Micro-IN-ATE	СС	система продукций
LEONARDO	СС	система продукций
NEXUS	СС	система продукций
PC/Beogle	СС	система продукций
Personal Consultant Plus	СС	система продукций, фреймы
Pro MD	СС	система продукций
RULE MASTER	СИП	система продукций
SAVOIR	СС	система продукций
Super Expert	СИП	система продукций
TESS	СС	система продукций
TIMM	СИП	система продукций
VP-Expert	СИП	система продукций
WIZDOW	СС	семантические сети
WPS	СС	система продукций
Xi	СС	система продукций
Xi Plus	СС	система продукций
XSYS	СС	система продукций
ДИЭС	СС	система продукций
ЭКО	СС	система продукций
НЭКС-2	СС	система продукций, объекты
МикроПРИЗ & МикроЭксперт	СИП	логика, система продукций, сети
ЭКСПЕРТ	СС	система продукций
ЭКСПЕРТИЗА	СС	система продукций, фреймы

Большой популярностью пользуется также продукционный язык OPS-5 фирмы DEC и его развитие — OPS-83, находящиеся на исследовательской стадии существования, но уже реально применяющиеся.

Важно понять, что система PROLOG может выступать как в роли собственно *системы программирования* для разработки систем ИИ, так и в роли *системы представления знаний*. В последнем случае программа на PROLOG'е рассматривается как база знаний, а встроенный в систему механизм вывода — как решатель. Однако PROLOG при таком использовании обладает существенными недостатками, среди которых:

— реализация гипотезы о замкнутости мира;

— недопустимость противоречий;

— ограничение логикой первого порядка (а точнее — только ее подмножеством);

— необходимость уяснения и постоянного учета программистом встроенного механизма возврата при поиске (выводе) решений.

Гипотеза о замкнутости мира ведет к предположению о полноте базы знаний. Поэтому утверждение, которое не следует из базы знаний, считается ложным, что может не соответствовать действительности.

Любая система аксиом, выражающая знания о мире, неизбежно оказывается *противоречивой*. Логические же системы, в том числе PROLOG, противоречия не преемлют: согласно известной теореме Геделя, противоречивые системы аксиом бессмысленны, так как из них можно вывести любое абсурдное утверждение.

Ограничение логикой первого порядка не позволяет адекватно описывать некоторые реальные задачи.

Весьма интересен новый продукт фирмы Borland International, названный Object Vision. Он предназначен для автоматизации программирования. Все, что нужно сделать пользователю, — это корректно описать задачу, в результате чего при помощи Object Vision он получит программу ее решения, которая будет функционировать в среде Windows. Для тех же целей используется отечественная система ПРИЗ.

4.4.4. Текстовые редакторы

Редактором называется программный продукт, служащий для создания и изменения целевого документа. При этом целевой документ записывается в файл на диске для последующего использования (обновления, вывода на печать и т.п.). Использование редактора по сравнению, например, с пишущей машинкой имеет неспоримые преимущества, так как документы можно легко изменять (редактировать), печатать в любом количестве экземпляров и форматировать (применяя различные шрифты, осуществляя выравнивание и т.п.). Современные редакторы обладают удобным пользовательским интерфейсом и позволяют отображать требуемую часть обрабатываемого документа на экране дисплея.

В понятие *документа* входят такие объекты, как исходные программы (на каком-либо языке программирования) и данные для них, тексты, формулы, таблицы, диаграммы, чертежи и различные графические изображения — словом, все то, что может содержаться в типографских изданиях.

В зависимости от *вида* обрабатываемых документов редакторы подразделяют на следующие *типы*:

1) *текстовые редакторы*, способные работать с текстовыми документами (программами, данными, собственно текстами, таблицами и частично с диаграммами, формулами, а также чертежами);

2) *графические редакторы*, специализирующиеся на работе со всевозможными графическими документами, включая диаграммы, иллюстрации и чертежи;

3) *системы верстки*, обладающие развитыми возможностями по форматированию текстов и графических материалов с последующим выводом их на печать.

Очевидно, только текстовые редакторы в определенном смысле могут рассматриваться как инструментальные системы в нашем понимании. Действительно, если текстовый редактор используется для подготовки исходной программы с целью последующей ее трансляции (интерпретации) или для подготовки данных, которые программа будет обрабатывать, то его можно считать инструментальной системой. В противном случае он рассматривается как прикладная программа. Другие типы редакторов однозначно относятся к прикладному ПО. Таким образом, если редактор обеспечивает подготовку документа для последующего использования системными программами, то он считается инструментальной системой. Иначе (если документ готовится только с целью вывода на печать) редактор считается прикладной программой.

В данном пункте мы рассмотрим только текстовые редакторы как (частично) инструментальные системы. К остальным же типам редакторов вернемся в подразделе 4.5.

В дальнейшем под **текстом** будем понимать любой документ, представленный в символьном виде.

Развитые текстовые редакторы выполняют следующие группы функций:

1) *редактирование текста*, а именно:

— запись текста в файл;

— удаление, вставку и пересылку символов, последовательностей символов, строк и целых фрагментов текста;

— поиск и замену цепочек символов;

— одновременную обработку различных фрагментов одного или нескольких файлов в различных окнах;

— создание и использование собственных макрокоманд для обработки текста;

— проверку орфографии;

— поиск синонимов;

2) *форматирование текста*, в том числе:

— оформление текста с использованием различных шрифтов;

— управление делением текста на абзацы;

— автоматический перенос слов;

— выравнивание текста по правой границе;

— структуризацию текстов (главным образом, при подготовке программ);

— многоколонковый набор;

3) *слияние файлов*, т.е. импорт файлов в различных форматах, подготовленных другими программными продуктами;

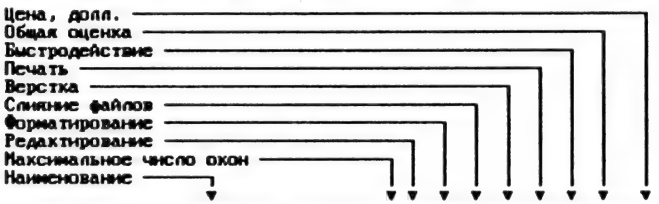
4) *экспорт файлов*, т.е. перезапись их в другом формате с целью использования в иных программных продуктах;

5) *подготовку текстов к печати* (функции верстки в усеченном виде), среди которых:
— включение в текст иллюстраций (импорт графики);
— предварительный просмотр сверстанных страниц;
— вычерчивание линий;
— подготовку оглавлений, сносок и индексов;
6) *печать текстов* на различных типах принтеров в различных режимах.

Для ПЭВМ разработано большое множество текстовых редакторов. Результаты сравнительной оценки получивших широкое распространение текстовых редакторов, работающих под управлением DOS, приведены в табл. 4.6. В ней использованы те же соглашения и сокращения, что и в табл. 4.2 (см. п. 4.3.1).

Таблица 4.6

Характеристики текстовых редакторов для DOS



DisplayWrite 4 версия 2	1	У	У	Х	П	Х	Х	У	495
Microsoft Word 5.0	8	О	ОХ	ОХ	ОХ	ОХ	Х	ОХ	н/д
Multimate Advantage II 1.0	1	Х	У	О	П	Х	Х	У	565
Samna Word IV 2.0	2	У	У	У	П	Х	У	У	595
WordPerfect 5.0	2	ОХ	Х	ОХ	О	ОХ	Х	ОХ	495
WordStar 2000 Plus 3.5	2	Х	Х	ОХ	Х	Х	Х	Х	495
WordStar Professional 6.0	2	Х	Х	ОХ	Х	Х	Х	Х	495
XyWrite III Plus 3.54	9	Х	Х	П	П	ОХ	О	Х	445

Лучшим среди существующих текстовых редакторов является Word 5.0 фирмы Microsoft. Он обладает великолепными средствами редактирования файлов и наилучшим образом использует возможности «мыши». Обеспечивается возможность импорта как текстов, так и иллюстраций, подготовленных в различных форматах. Поддерживается широкий набор шрифтов (как по форме, так и по размеру) и стилей оформления документов. Имеется орфографический словарь на 130000 английских слов для проверки правописания, а также словарь синонимов на 220000 слов. Однако пользовательский интерфейс требовал улучшения, что и сделано, по последним сведениям, в Word 5.5. Для использования этого редактора достаточно иметь ПЭВМ IBM PC, ОС DOS 2.0 или выше, 384 Кбайт ОЗУ и два НГМД, но все же желателен жесткий диск. Многие возможности Word 5.0 (в частности, автоматический перенос, проверка правописания и поиск синонимов) ориентированы только на англоязычные тексты, что не позволяет их использовать при подготовке текстов на русском языке. Однако сотрудники СП «Диалог» перевели на русский язык пользовательский интерфейс и обеспечили работу с русскими буквами; сейчас это предприятие адаптирует Word 5.0 к работе с русскоязычными текстами в полном объеме. Конкурентом «Диалога» в этой области является СП «ПараГраф», уже предлагающее программный продукт «Русское слово», содержащий редактор Word 5.0 и его русификатор.

В связи с упомянутыми сложностями обработки русскоязычных текстов в СНГ в настоящее время наибольшее распространение получил отечественный текстовый редактор ЛЕКСИКОН. Он обладает весьма ограниченными возможностями, но учитывает специфику текстов на русском языке и позволяет их печатать машинописным шрифтом, а также другими шрифтами в графическом режиме принтера. Наиболее существенный недостаток этого редактора состоит в том, что он не поддерживает редактирование файлов, которые не могут быть размещены в ОЗУ целиком. Входящий в ЛЕКСИКОН модуль ОРТОДОК обеспечивает проверку орфографии.

Заострим наше внимание еще на двух текстовых редакторах — MULTI-EDIT 5.0 и ChiWriter.

Первый из них отличается возможностью одновременной работы со 100 окнами, великолепным пользовательским интерфейсом, наличием удобных средств вычерчивания линий и мощнейших средств контекстного поиска и замены цепочек символов с использованием регулярных выражений, а также удобством подготовки текстов программ благодаря наличию средств их структуризации и шаблонов программных структур.

Редактор ChiWriter обеспечивает подготовку текстов, содержащих формулы и простые рисунки.

4.4.5. Интегрированные системы

Интегрированной системой называют программный продукт, представляющий собой совокупность функционально различных компонентов, способных взаимодействовать между собой путем передачи информации и объединенных единым унифицированным пользовательским интерфейсом. Такие системы обеспечивают различные информационные и вычислительные потребности пользователя и служат, главным образом, для автоматизации *учрежденческой деятельности.* В частности, интегрированная система может быть помощником руководителя любого ранга и просто делового человека в его повседневной деятельности. Интегрированные системы в идеале претендуют на решение всех задач определенной группы пользователей с тем, чтобы им не нужно было обращаться к другим программным продуктам.

Современные интегрированные системы, как правило, содержат следующие пять *функциональных компонентов*:

1) электронную таблицу;
2) текстовый редактор;
3) СУБД;
4) графический редактор;
5) коммуникационный модуль.

Так как часть компонентов относится к системному, а другая часть — к прикладному ПО, интегрированные системы (как и текстовые редакторы) занимают промежуточное положение. Мы предпочли описать их в рамках инструментальных систем, поскольку они содержат текстовые редакторы и СУБД.

Некоторые из перечисленных компонентов требуют особого пояснения, так как программные продукты такого типа мы еще не рассматривали.

Электронная таблица — это программа, служащая для обработки прямоугольных таблиц, ячейки которых могут содержать произвольные объекты (среди них числа, строки и формулы, задающие зависимость значения ячейки от содержимого других ячеек). Изменение пользователем содержимого ячейки приводит к изменению значений в зависящих от нее ячейках. Эти изменения осуществляются автоматически или по специальному запросу.

Что касается *графических редакторов*, то подчеркнем, что большинство интегрированных систем поддерживают только *деловую* графику (подготовку графиков, диаграмм, таблиц и т.п.), в том числе на основе содержимого электронных таблиц и БзД. *Иллюстративная* же графика (подготовка произвольных изображений), обеспечивается лишь некоторыми системами.

Коммуникационный модуль служит для связи пользователя через оборудование ПЭВМ с внешней средой, т.е. с другими компьютерами, телефонными линиями и т.п. При этом часто обеспечивается возможность автоматического набора заданного номера телефона и эмуляции различных терминалов.

Интегрированные системы могут содержать и дополнительные средства, среди которых:

— системы программирования;
— системы экспорта/импорта файлов;
— средства подготовки стандартных писем;
— средства сортировки и фильтрации записей в файлах;
— калькуляторы;
— календари.

Достоинства интеграции разнородных программных компонентов в единое целое состоят в следующем:

1) в обеспечении возможности информационной связи между компонентами системы;
2) в организации единого унифицированного пользовательского интерфейса, что снижает затраты на освоение системы и облегчает работу с ней.

Информационная связь между компонентами интегрированной системы обеспечивается путем унификации форматов представления различных документов. Такая связь может осуществляться через файлы или непосредственно в ходе работы системы, например, через общую рабочую область ОЗУ. При связи через файлы часто используется метод «отрезания-приклеивания».

В этом случае пользователь «отрезает» данные от файла в одном приложении и «приклеивает» их к файлу в другом приложении. Иногда обеспечивается также активная связь между файлами: копия файла отражает все изменения, выполненные в оригинале после ее создания.

Вместе с тем интеграция обладает и *недостатками,* среди которых:

1) повышенные требования к ОЗУ (для работы системы обычно необходимо 200 — 400 Кбайт);
2) неизбежность реализации компонентов с некоторыми ограничениями функциональных возможностей по сравнению с лучшими образцами в своем классе неинтегрированных продуктов.

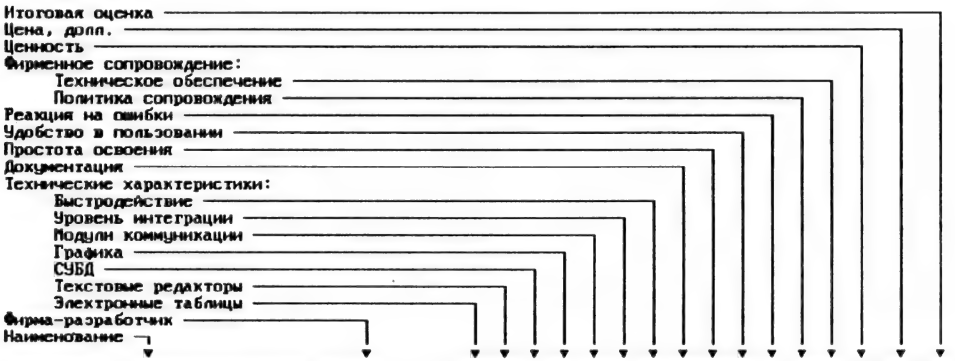
Сравнительные характеристики распространенных интегрированных систем для DOS, заимствованные из журнала «Мир ПК», приведены в табл. 4.7. В ней также использованы соглашения и сокращения из табл. 4.2. Отметим, что под *ценностью* системы понимается соотношение между ее достоинствами и стоимостью.

Интегрированные системы по сфере применения условно можно разделить на две группы.

К *первой* группе относятся системы с богатыми функциональными возможностями обработки данных. Они предназначены для профессиональных пользователей, работающих со сложными документами, оперирующих большими объемами информации и выполняющих значительный

Таблица 4.7

Характеристики интегрированных систем для DOS



Alphaworks 1.0	Alpha Software	У	Х	ОХ	У	У	Х	ОХ	Х	Х	Х	У	ОХ	Х	Х	195	6.1
Enable/OA 3.0	Enable Software	ОХ	ОХ	ОХ	О	У	ОХ	Х	П	П	П	О	У	Х	У	695	5.8
Framework III 1.1	Ashton-Tate	ОХ	ОХ	П	У	О	О	У	ОХ	ОХ	ОХ	Х	У	У	ОХ	695	7.0
Microsoft Works 1.05	Microsoft	Х	Х	Х	ОХ	У	ОХ	ОХ	Х	ОХ	ОХ	ОХ	О	Х	ОХ	149	7.0
PFS:First Choise 3.0	Software Publishing	У	У	Х	Х	У	П	Х	Х	Х	Х	У	Х	Х	У	159	5.4
Smartware II 1.0	Informix Software	Х	ОХ	ОХ	ОХ	Х	ОХ	Х	Х	Х	ОХ	ОХ	У	Х	ОХ	699	6.9
Symphony 2.0	Lotus Development	ОХ	Х	П	Х	У	ОХ	У	ОХ	У	Х	П	О	Х	П	695	5.5

объем вычислительных работ. Enable/OA, Framework III, Smartware II и Symphony как раз и являются такими системами. Среди них по функциональным характеристикам предпочтение отдается системе Smartware II; за ней следует Framework III. Enable/OA имеет большие потенциальные возможности, но не все они пока реализованы. Дополнительное достоинство этого продукта заключается в том, что существует его версия для OS/2.

Вторую группу составляют системы индивидуального пользования, предназначенные для начинающих и для тех, кто работает с относительно несложными текстами, электронными таблицами и базами данных. К таким продуктам относятся Alphawork, PFS:First Choice и Microsoft Works. Явным лидером среди них является последняя система. В 1990 г. фирма Microsoft стала предлагать уже Works 2.0 с более богатыми возможностями. В настоящее время продается также фирменный русифицированный вариант этого продукта.

В СНГ Е.Н.Веселовым разработана интегрированная система МАСТЕР, включающая:
- электронную таблицу;
- текстовый редактор ЛЕКСИКОН;
- СУБД;
- графический редактор, поддерживающий как деловую, так и иллюстративную графику;
- систему процедурного программирования.

В рамках СП ИНТЕРКВАДРО создан инструментальный комплекс ИРИС для разработки документо-ориентированных информационно-поисковых систем, необходимых для автоматизации различного рода конторской деятельности.

4.5. Прикладное ПО

Мы не ставили перед собой цели в рамках настоящей книги рассмотреть прикладное ПО (пакеты прикладных программ) ПЭВМ. Тем не менее автор все-таки решил бегло ознакомить читателя с состоянием дел в данной области.

В настоящее время для ПЭВМ предлагается широкое разнообразие прикладных программных продуктов, автоматизирующих различные виды (сферы) человеческой деятельности. Выделим среди них такие, как:

1) редакторы;
2) интеллектуальные системы, в том числе ЭС;
3) банки данных;
4) информационно-поисковые системы;
5) электронные таблицы;
6) обучающие системы;
7) математические программы;

8) программы для моделирования;
9) системы автоматизированного проектирования.

Одна из разновидностей **редакторов**, а именно, *текстовые редакторы*, уже обсуждалась в п. 4.4.4. Дополнительно к этому отметим еще два момента.

Во-первых, наряду с мощными текстовыми редакторами существуют дешевые редакторы с максимально упрощенным пользовательским интерфейсом, но и с менее широкими возможностями. Они служат для удовлетворения потребностей деловых людей в подготовке небольших текстовых документов. Лидером среди редакторов данного класса является продукт Beyond Word Writer 1.0 фирмы Timeworks, удачно сочетающий достаточно мощные средства редактирования с удобством их использования, а также обладающий высоким быстродействием и имеющий низкую стоимость.

Во-вторых, хотя существующие текстовые редакторы и обладают богатыми возможностями, дополнительно к ним предлагаются различные *сервисные программы*. Такое положение дел является следствием того, что специализация, как правило, означает повышение эффективности. Среди программ этого типа выделяют следующие:

1) *программы форматирования*, обеспечивающие приведение документа к требуемой форме;
2) различные *словари* (словари синонимов, разноязыковые словари, толковые словари и т.п.), многие из которых могут работать в резидентном режиме, что облегчает их использование;
3) *программы орфографического, грамматического* и/или *стилистического контроля* текстовых документов, которые зачастую также способны работать в резидентном режиме;
4) *программы групповой записи текстов*, обеспечивающие работу нескольких пользователей с одним и тем же документом, причем с фиксацией авторства фрагментов текста;
5) *программы преобразования файлов*, содержащих тексты, из одного формата в другой;
6) *программы компоновки* для оформления стандартных документов.

Существующие *графические редакторы* поддерживают деловую и/или иллюстративную графику.

Деловая графика предполагает построение графиков различных типов (функциональных зависимостей, секторных диаграмм, диаграмм Ганта, гистограмм и т.п.). Наиболее мощным редактором этого класса является изделие Boieng Graph фирмы Boieng.

Иллюстративная графика служит для создания всевозможных рисунков и мультипликационных изображений на экране дисплея.

Изображения строятся с использованием клавиатуры или «мыши» из имеющихся заготовок различных фигур, линий, заполнителей и шрифтов. Допускается управление размером фигур и шрифтов, а также перемещение фигур и букв в рамках документа. При создании иллюстраций можно формировать любые изображения. Все, что Вы делаете, отображается на экране дисплея.

Наиболее популярным в настоящее время графическим редактором, поддерживающим иллюстративную графику, является продукт PC Paintbrush IV Plus фирмы Zsoft. Для его работы необходима ПЭВМ семейства PC IBM с НЖМД или двумя НГМД, манипулятор «мышь» или световое перо, графический дисплейный адаптер Hercules, CGA, EGA, MCGA или VGA, ОЗУ емкостью 640 Кбайт, ОС MS-DOS версии 3.0 или выше, а также драйвер «мыши». Стоимость редактора составляет 99.95 долл.

В качестве примера системы для создания мультфильмов можно привести FantaVision.

Разработано и большое количество пакетов прикладных программ, расширяющих графические возможности систем программирования.

Среди *систем верстки*, называемых также *настольными издательскими системами*, лидируют продукты PageMaker фирмы Aldus и Ventura Publisher фирмы Xerox, цены которых составляют 795 и 895 долл. соответственно.

PageMaker требует IBM PC AT, 640-Кбайт ОЗУ, DOS версии 3.1 или более поздней, НЖМД, графический дисплейный адаптер и манипулятор «мышь» с драйвером. Эта система имитирует рабочий стол, используемый при верстке; в процессе верстки текст и иллюстрации можно «монтировать» на странице, перемещая их произвольным образом; лучше подходит для коротких документов разнообразных форматов; удобна в использовании.

Ventura Publisher требует ПЭВМ класса XT или старше, 640-Кбайт ОЗУ, DOS версии не ниже 2.1, НЖМД, графический дисплейный адаптер и манипулятор «мышь» с драйвером. В этом продукте основное внимание уделяется автоматизации верстки. Он лучше подходит для подготовки больших документов со строго определенной структурой и сложнее в использовании, чем PageMaker.

Для эффективной работы систем верстки требуются лазерные принтеры или их функциональные аналоги.

Интеллектуальные системы и **БД** уже нами рассматривались в подразделе 4.4 в связи с инструментарием для их разработки. **Информационно-поисковая система** может основываться на БД и содержать дополнительные, как правило, специализированные средства для хранения и извлечения информации.

Понятие **электронной таблицы** введено в п. 4.4.5. Первым продуктом такого типа была программа VisiCalc, разработанная Д.Бриклином и Б.Фрэкстоном в 1979 г. для ПЭВМ Apple II, а впоследствии перенесенная на IBM PC. В 1983 г. фирмой Lotus Development выпущена электронная таблица Lotus 1-2-3, успех которой превзошел все ожидания. С тех пор и до недавнего времени это изделие было бесспорным лидером в классе электронных таблиц. Lotus 1-2-3 дополнительно к электронной таблице содержит СУБД и графический модуль. Из нее как раз и была создана интегрированная

система Symphony. С Lotus 1-2-3 в острую конкурентную борьбу вступили продукты Excel корпорации Microsoft и Quattro Pro фирмы Borland International. Изделием Quattro Pro 2.0 фирме Borland International удалось забрать «пальму первенства» у компании Lotus Development. Quattro Pro 2.0 стояла на третьей позиции в списке лучших программных продуктов для ПЭВМ в 1990 г.

Функции **обучающей системы** непосредственно следуют из ее названия. Такие системы служат для обучения людей в той или иной предметной области. Работа с ними имеет ярко выраженный интерактивный характер. Наиболее развитые обучающие системы базируются на методах ИИ. Технология обучения может быть различной.

Математические программы используются математиками, физиками, инженерами и др. специалистами при решении разнообразных вычислительных задач. Различают *программы символьной математики,* где решение задачи (например, взятие производной) осуществляется в аналитическом виде, и *программы численной математики,* решение задачи которыми предлагается в численном виде. Среди программ, интегрирующих оба рассмотренных способа решения математических задач, наиболее широкими возможностями обладает продукт Mathematica 1.2 фирмы Wolfram Research, требующий 4-Мбайт ОЗУ, НЖМД и МП 80386.

Для различных систем программирования предлагаются также пакеты прикладных программ, усиливающие их вычислительные возможности.

Среди пестрого многообразия **программных продуктов для моделирования** существуют пакеты программ для логического моделирования работы цифровых электронных схем, для моделирования процессов в аналоговых электронных схемах, систем автоматического управления, механических свойств конструкций, тепловых режимов работы конструкций, для анализа механики газов и жидкостей, а также для имитационного моделирования.

Системы автоматизированного проектирования служат для оказания помощи в проектировании технических изделий. Они содержат мощные модули инженерной графики, обеспечивающие изготовление высококачественных чертежей.

Путем подбора соответствующего прикладного ПО на базе ПЭВМ можно создать различные АРМ.

5. ОПЕРАЦИОННАЯ СИСТЕМА DOS

В настоящее время, как было сказано выше, наиболее распространенными ОС на ПЭВМ являются системы семейства DOS. Поэтому сделанный нами для рассмотрения выбор закономерен.

В данном разделе изучаются:

— принципы построения и функционирования DOS;
— программный интерфейс DOS (но только на понятийном уровне);
— пользовательский интерфейс DOS (с максимальной степенью детализации);
— основы функционирования и методы управления часто используемыми ПУ;
— размещение информации на магнитных дисках.

Изложение построено таким образом, чтобы читатели получили исчерпывающие сведения как по DOS 3.3, так и по DOS 4.0. Целесообразность отдельного рассмотрения устаревшей версии системы объясняется тем, что DOS 3.3, оставаясь наиболее распространенной в настоящее время системой, лучше всего подходит для ПЭВМ класса XT, а все отечественные персональные компьютеры пока являются, как правило, таковыми. Кроме того, DOS 3.3 надежнее своей преемницы.

DOS 5.0 появилась в тот момент, когда работа над рукописью настоящей книги была, в основном, завершена. Поэтому автор счел целесообразным описать особенности новой версии DOS в Приложении (см. Приложение 2). Однако такое решение вовсе не означает, что в рамках данного раздела DOS 5.0 вообще не будет упоминаться.

5.1. Версии DOS

Первый представитель семейства DOS появился одновременно с ПЭВМ IBM PC в 1981 г. и сильно напоминал систему CP/M. С тех пор развитие DOS идет в сторону ОС UNIX. Полного их слияния, видимо, не произойдет, так как в противном случае DOS потеряет свою индивидуальность.

Материал этого подраздела требует предварительного знакомства с данной ОС, в связи с чем неподготовленный читатель сначала может его только просмотреть, чтобы впоследствии вернуться к нему при необходимости. И еще одно замечание: автор ограничился рассмотрением продуктов только фирм Microsoft и IBM, не называя их явно.

Каждая новая версия DOS появлялась, как правило, в связи с созданием новых аппаратных средств. *Номер версии* состоит из двух чисел, разделенных точкой. Первое число обозначает *основную редакцию*, второе — ее *модификацию*. Так, DOS 2.1 сильно отличается от DOS 1.1, но очень похожа на DOS 2.0.

Теперь перейдем к рассмотрению наиболее существенных особенностей различных версий DOS.

DOS 1.00 Появилась в связи с созданием IBM PC. Подобна CP/M, но предназначена для МП 8088. Поддерживает только односторонние 133-мм 8-секторные 160-Кбайт НГМД.

DOS 1.05 Устраняет ряд ошибок, обнаруженных в DOS 1.00.

DOS 1.10 Была стандартом более года. Дополнительно к предыдущей версии поддерживает двухсторонние 133-мм 8-секторные 320-Кбайт НГМД.

DOS 2.00 Появилась в связи с созданием IBM PC XT. Поддерживает НЖМД емкостью до 10 Мбайт. Дополнительно к предыдущей версии ОС обслуживает 133-мм 9-секторные односторонние (180-Кбайт) и двухсторонние (360-Кбайт) НГМД. Поддерживает древовидную файловую структуру. Реализует концепции стандартного ввода-вывода, перенаправления ввода-вывода и фильтров. Обрабатывает следующие новые команды: FC (только MS-DOS), BACKUP, RESTORE, TREE, CD, MD, RD, PATH и др. Имеет расширенный язык командных файлов за счет новых команд GOTO, IF, ECHO и др. Реализует возможность подключения (установки) внешних драйверов устройств. Обеспечивает фоновую печать по команде PRINT. Поддерживает видеосистему CGA.

DOS 2.10 Создана для IBM PCjr. Основана на DOS 2.00 и устраняет обнаруженные в ней ошибки.

DOS 3.00 Появилась в связи с созданием IBM PC AT. За счет указания маршрута поиска позволяет выполнять программы из файлов, которые находятся не в рабочем каталоге. Поддерживает НЖМД емкостью до 20 Мбайт. Дополнительно к предыдущим версиям обслуживает двухсторонние 133-мм 15-секторные 1,2-Мбайт НГМД. Обрабатывает новые команды ATTRIB, LABEL, SELECT, KEYBxx, SHARE, GRAFTABL, COUNTRY=. Поддерживает виртуальный диск в ОЗУ.

DOS 3.10 Имеет некоторые сетевые средства. Поддерживает новые команды JOIN и SUBST.

DOS 3.20 Создана для IBM PC Convertible. Дополнительно поддерживает 89-мм 720-Кбайт НГМД. Обрабатывает новые команды REPLACE и XCOPY. Поддерживает усовершенствованные команды ATTRIB, COMMAND, FORMAT, SELECT, GRAPHICS, SHELL=. Препятствует непреднамеренному форматированию жесткого диска. Поддерживает драйвер DRIVER.SYS для создания фиктивных дисководов.

DOS 3.30 Появилась в связи с созданием семейства PS/2 и способна функционировать на моделях семейства PC. Поддерживает концепцию разбиения жестких дисков любого объема на логические диски размером до 32 Мбайт каждый, которые можно использовать одновременно (все они доступны DOS). Дополнительно обслуживает 89-мм 1,44-Мбайт НГМД. Содержит усовершенствованные средства для поддержки национальных языков (введено понятие кодовой страницы). Имеет несколько выявленных ошибок (например, команда FORMAT может забраковать хороший диск). Поддерживает новые команды APPEND, CALL, CHCP, FASTOPEN и NLSFUNC, а также усовершенствованные команды DATE, TIME, ATTRIB, BACKUP, FDISK, RESTORE и XCOPY. Реализует усовершенствованный язык командных файлов.

DOS 4.00 Поддерживает логические диски на винчестере размером свыше 32 Мбайт. Использует отображаемую память для буферов ОС и структур данных команды FASTOPEN (требуется EMS 4.0). Позволяет задействовать для размещения резидентных программ первые 64 Кбайт расширенной памяти (HMA-память). Обеспечивает расширенную поддержку национальных языков. Отличается значительным числом ошибок. Обрабатывает новую команду MEM, а также усовершенствованные команды APPEND, ATTRIB, BACKUP, COUNTRY, MODE, FASTOPEN, FDISK, GRAPHICS, GRAFTABL, NLSFUNC, REPLACE, SELECT, TREE, DEL и др. Имеет новые команды конфигурирования системы. Содержит усовершенствованные драйверы устройств ANSI.SYS, DISPLAY.SYS, DRIVER.SYS и PRINTER.SYS. Наконец-то полностью поддерживает все режимы работы видеосистем EGA и VGA.

DOS 4.01 Содержит графическую оболочку MS-DOS Shell, поддерживающую манипулятор «мышь».

DOS 5.00 Обеспечивает размещение своего ядра, а также драйверов и резидентных программ в верхней памяти. Содержит усовершенствованную оболочку, внешне напоминающую пользовательский интерфейс Windows и обеспечивающую переключение задач. Способна работать с 89-мм 2,88-Мбайт НГМД, которые вскоре начнут применяться. Непосредственно (без загрузки SHARE) поддерживает логические диски на винчестере размером свыше 32 Мбайт. Обрабатывает новые команды DELOLDOS, DOSKEY, EXPAND, LOADHIGH, MIRROR, SETVER, UNDELETE и UNFORMAT, а также усовершенствованные команды DIR, FORMAT, SYS и др. Поддерживает новые команды конфигурирования системы. Имеет встроенную справочную систему. Содержит улучшенную систему программирования Basic. Отличается высокой надежностью в работе.

Каждая последующая версия DOS содержит все возможности предыдущей и дополнительно обладает новыми. Кроме указанных отличий, от версии к версии развивается и программный интерфейс DOS, но рассмотрение этих вопросов выходит за рамки данной книги. Наиболее существенные изменения в этом направлении связаны с новыми редакциями DOS. Из приведенных сведений видно, что облик современной DOS главным образом определяется концепциями, заложенными в DOS 2.0.

С переносом программ в среду новой версии DOS, как правило, проблем не возникает, за исключением случая, когда Вы отформатировали дискету в среде более поздней версии, а используете ее с более ранней версией DOS. Если на экране дисплея появится сообщение «Wrong DOS version» (Ошибочная версия DOS), Вам придется загрузить ту версию, в среде которой подготовлена дискета. В противоположном случае (когда дискета подготовлена в среде старой версии, а подлежит использованию с новой версией DOS) целесообразно придерживаться следующих рекомендаций:

1) если дискета содержит только файлы с данными, то ее можно использовать без ограничений;

2) в противном случае надо скопировать, по крайней мере, исполняемые файлы на новую дискету (отформатированную в среде новой версии DOS) и использовать последнюю, возможно, совместно (для чтения данных) с первой.

5.2. Принципы построения и функционирования DOS

В этом подразделе рассматривается организация DOS, ее возможности и порядок работы. Излагаемый материал ориентирован на специалиста, знакомого с основами построения ОС ЭВМ. Обсуждение ведется на понятийном уровне: читатель получит ответы на вопросы «ЧТО DOS делает?», «КАК в принципе она это делает?», но не «КАК ЗАСТАВИТЬ ее что-то сделать?». Для ответа на третий вопрос, касающийся программного интерфейса системы, следует обратиться к справочнику программиста по DOS. Чтобы получить ответ на третий вопрос применительно к пользовательскому интерфейсу DOS, прочитайте подразделы 5.5 — 5.10.

Управление ресурсами ПЭВМ (за исключением памяти), а также процессами в подзаголовках не фигурирует, однако все эти вопросы с той или иной степенью подробности находят здесь свое отражение.

Предлагаемый материал необходимо прочитать и пользователю, не интересующемуся организацией и функционированием DOS, так как в нем вводятся базовые понятия и основные процедуры работы, которые требуется знать каждому, кто собирается работать на ПЭВМ (для таких пользователей, конечно, приводимые сведения окажутся избыточными).

5.2.1. Структура DOS

В состав DOS входят следующие *структурные компоненты:*

1) базовая система ввода-вывода (BIOS);
2) системный загрузчик (SB — System Bootstrap);
3) модуль расширения (МР) BIOS;
4) внешние (устанавливаемые) драйверы устройств;
5) базовый модуль (БМ) DOS;
6) командный процессор (КП), или интерпретатор команд;
7) утилиты DOS.

Не надо пугаться сокращений на разных языках. Мы понимаем, что это не очень хорошо, но тем не менее устоявшиеся сокращения целесообразнее приводить на языке оригинала.

BIOS хранится в ПЗУ и поэтому выполняет троякую роль:

1) является частью ПЭВМ;
2) является компонентом DOS;
3) является компонентом любой ОС, запускаемой на данной ПЭВМ.

Остальные компоненты DOS, за исключением внешних драйверов и утилит, должны размещаться на системном диске (гибком или жестком) в специальных областях и файлах, структура которого рассматривается в п. 5.2.3. Внешние драйверы и утилиты DOS могут располагаться в файлах как на системном, так и на любом другом диске. Однако для их хранения обычно используется именно системный диск, так как это удобнее для работы. В принципе КП также можно разместить на несистемном диске, но при этом придется сконфигурировать DOS специальным образом. Внешние драйверы и утилиты только дополняют DOS, принципиально не влияя на ее работоспособность. Конкретный их набор определяется пользователем в зависимости от его потребностей, а также конфигурации оборудования.

Для того чтобы на ПЭВМ можно было работать, необходимо предварительно выполнить *загрузку* DOS с *системного диска* в ОЗУ и передать на нее управление. Эта процедура описывается в п. 5.2.4.

В данном пункте мы рассмотрим только *назначение*, *состав* и *порядок доступа* к компонентам DOS во время ее работы, а также приведем их краткую характеристику. *Функции* же этих компонентов, а также их *взаимодействие* в ходе загрузки DOS и выполнения программ, будут обсуждаться в последующих пунктах данного подраздела.

BIOS, «скрывая» архитектурные особенности конкретной модели ПЭВМ, реализует наиболее простые и универсальные услуги DOS по управлению основными (стандартными) ПУ, в частности, по организации ввода-вывода. Поэтому BIOS освобождает обращающиеся к ней программы и другие компоненты DOS от «знания» и учета особенностей оборудования, а также деталей управления тем или иным ПУ, что обеспечивает независимость ПО от ПУ. Отметим, что BIOS выполняет и другие функции, не связанные с ее основным назначением.

BIOS содержит:

1) драйверы стандартных ПУ;
2) тестовые программы для контроля работоспособности оборудования;
3) программу начальной загрузки;
4) интерпретатор BASIC'a.

Драйверы как раз и решают основную задачу BIOS. **Драйвером** называется программа, обслуживающая те или иные ПУ. Драйвер выполняет следующие функции:

— принимает запросы на обращение к ПУ;

— преобразует запросы в команды управления устройством с учетом всех деталей его конструкции и особенностей работы в реальном времени;

— обрабатывает прерывания от обслуживаемого ПУ.

Следовательно, драйвер является промежуточным звеном (посредником) между обращающимися к ПУ программами и самим ПУ.

Программа начальной загрузки не привязана к конкретной ОС и обеспечивает загрузку в ОЗУ, а также запуск SB из определенной области системного диска. По этой причине она называется *первичным загрузчиком.*

Доступ к средствам BIOS осуществляется, главным образом, через аппарат прерываний. Она совместно с MP BIOS обрабатывает прерывания 00H — 1FH, являющиеся прерываниями *нижнего уровня* (потому что услуги BIOS считаются низкоуровневыми). Среди этих прерываний есть как программные, так и аппаратные.

SB является *вторичным загрузчиком*, участвующим в загрузке DOS. Он зависит от той ОС, которую должен загружать. Двухуровневая организация загрузки придает гибкость системе, упрощая ее модификацию. Более того, иная организация загрузки просто невозможна, так как первичный загрузчик прошит в ПЗУ и не в состоянии учесть особенности загрузки всех существующих, а тем более еще не созданных ОС.

MP BIOS является надстройкой над BIOS. В его задачи входят:

1) организация интерфейса с BIOS;
2) логическая замена драйверов, хранящихся в BIOS;
3) подключение (если требуется) новых драйверов.

Необходимость замены существующих и подключения к системе новых драйверов возникает при изменении состава ПУ и при потребности в использовании имеющихся ПУ нестандартным образом. Это делается при помощи механизма подмены адресов обработчиков прерываний в векторах прерываний. Кроме того, драйверы связываются в *цепочку (список)*, которая при поиске драйвера просматривается с той стороны, с которой драйверы добавляются, строго последовательно, чем имитируется стек. Использованию подлежит первый встретившийся из драйверов для данного ПУ. Очевидно, описанного механизма достаточно как для замены существующих, так и подключения новых драйверов.

Драйверы могут находиться как внутри MP BIOS, так и вне его, т.е. храниться в отдельных файлах. В первом случае они называются **внутренними (основными)**, а во втором — **внешними (устанавливаемыми).** Указания на подключение внешних драйверов должны содержаться в **файле конфигурации** CONFIG.SYS. В этом же файле специальными командами указываются параметры системы, используемые при загрузке для ее *конфигурирования* (настройки).

Способность подключения внешних драйверов существенно облегчает расширение возможностей ОС по управлению ПУ, так как при этом не требуется модифицировать основные ее компоненты. С другой стороны, некоторые драйверы нежелательно помещать в BIOS или в его расширение по той причине, что они используются не на каждой модели ПЭВМ и не каждым пользователем. В этом случае драйверы оформляются как внешние и подключаются только при необходимости, что повышает эффективность DOS.

MP BIOS содержит внутренние драйверы, код инициализации (который выполняется при загрузке), а также ряд управляющих блоков и таблиц.

Доступ программ к средствам MP BIOS осуществляется через аппарат прерываний, причем за ним и самой BIOS, как уже указывалось, закрепляются прерывания 00H — 1FH.

Вообще BIOS, MP BIOS и подключенные внешние драйверы составляют единую подсистему, которую мы назовем **системой ввода-вывода**. Такое название полностью соответствует выполняемым ею функциям, так как основной задачей этой системы является организация обмена информацией с ПУ.

Система ввода-вывода в общем случае содержит следующие драйверы, часть из которых является обязательной, а другая — факультативной:

— драйверы накопителей на гибких и жестких дисках;
— драйверы (обычный и расширенный) дисплея и клавиатуры;
— драйвер принтера;
— драйверы адаптеров интерфейсов;
— драйвер фиктивного устройства (вывод в это устройство воспринимается, но данные отбрасываются, а при попытке ввода немедленно опознается конец файла);
— драйвер часов;
— драйвер виртуального диска;
— драйвер манипулятора «мышь»;
— драйвер отображаемой памяти;
— драйвер расширенной памяти;
— драйверы верхней памяти.

Система ввода-вывода является *машинозависимой* частью DOS.

БМ DOS — это центральный ее компонент, реализующий основные функции по управлению всеми ресурсами ПЭВМ и выполняемыми программами. Управление ПУ осуществляется на более высоком, чем посредством драйверов, уровне путем организации обращения к последним. Именно здесь обеспечивается функционирование *файловой системы.*

БМ DOS включает код инициализации, выполняемый при загрузке системы, и совокупность обработчиков прерываний *верхнего уровня*. Доступ к средствам БМ DOS осуществляется по прерываниям 20H — 3FH, за исключением 22H — 24H. Особую роль играет прерывание 21H,

по которому доступны 57H сервисных функций DOS. Именно прерывания верхнего уровня выдает большинство программ, работающих под управлением DOS. Обработчики этих прерываний, в свою очередь, могут выдавать прерывания нижнего уровня, причем неоднократно.

Система ввода-вывода и БМ DOS в процессе работы системы находятся в ОЗУ резидентно (постоянно).

В то время как BIOS, МР BIOS и БМ DOS управляют ресурсами ПЭВМ, КП отвечает за поддержку пользовательского интерфейса DOS. Он решает следующие задачи:

— воспринимает команды DOS, вводимые пользователем с клавиатуры;
— выполняет часть из них, называемых внутренними;
— обрабатывает командные файлы;
— загружает программы в ОЗУ для выполнения;
— обрабатывает прерывания 22H (завершение задачи), 23H (реакция на одновременное нажатие клавиш Ctrl и Break) и 24H (реакция на *критическую* ошибку), дополняя БМ DOS.

Если на системном диске имеется командный файл AUTOEXEC.BAT, называемый **файлом автозапуска,** то при загрузке DOS КП организует его интерпретацию. В данный файл включают команды DOS, которые пользователю нужно регулярно выдавать после *запуска* системы (например, для установки даты и времени), чтобы освободить его от выполнения рутинных операций. Этот файл может включать и команды на запуск резидентных программ, подменяющих ряд обработчиков прерываний и выполняющих функции драйверов. По этой причине их часто называют драйверами, хотя они организуются и подключаются к системе несколько иначе.

Очевидно, что КП выполняет в основном роль оболочки. DOS предоставляет возможность замены КП при своей загрузке на любую другую оболочку, что ведет к экономии оперативной памяти. Действительно, если Вы пользуетесь какой-то оболочкой, то Вам КП может и не потребоваться. Тогда ему незачем занимать и память. Однако такая организация системы используется редко.

КП состоит из следующих модулей:

1) *резидентного модуля,* постоянно хранящегося после запуска DOS в ОЗУ и включающего обработчики прерываний 22H — 24H, а также код подзагрузки транзитной части КП;

2) *модуля инициализации,* выполняемого при загрузке DOS и затем затираемого выполняемыми программами;

3) *транзитного* (нерезидентного) *модуля,* который в ОЗУ может перекрываться выполняемыми программами, а затем восстанавливаться путем считывания с диска; этот модуль содержит интерпретатор (исполнитель) *внутренних команд* DOS и загрузчик программ в ОЗУ для выполнения.

В качестве внутренних выбраны наиболее употребимые команды DOS, чтобы быть резидентными и тем самым быстрее выполняться.

Доступ к КП осуществляется по прерываниям от клавиатуры и по прерываниям 22H — 24H.

Утилиты DOS обеспечивают выполнение *внешних команд* DOS. Они названы внешними потому, что реализуются отдельными программами, а не КП. Внешние команды дополняют пользовательский интерфейс DOS.

БМ DOS, КП и утилиты DOS составляют ее *машинонезависимую* часть.

Наряду с рассмотренными, есть и еще одна возможность управления ПУ. Она состоит в непосредственном обращении из программ к ПВВ (через так называемый *программный интерфейс ПЭВМ*). В этом случае средства ОС не используются, управление ПУ существенно усложняется, программа получается немобильной, но зато появляется возможность задействовать абсолютно все специфические возможности ПУ, причем наиболее эффективным способом.

На рис 5.1 схематично представлена структура DOS с учетом изложенного в данном пункте. SB не показан, так как он при работе ОС не используется. Взаимодействующие между собой модули на данном рисунке соприкасаются. Таким образом, DOS имеет *модульную структуру*, что облегчает ее модификацию, она открыта для наращивания возможностей. Основным механизмом функционирования DOS является система прерываний.

5.2.2. Файловая система DOS

Рассматриваемый в этом пункте материал затрагивает вопросы работы с внешней памятью.

Мы уже знаем, что в качестве *первичной* внешней памяти ППЭВМ выступают гибкие и жесткие МД. Поэтому здесь и в дальнейшем ограничимся рассмотрением методов работы только с дисковой памятью. В нашем поле зрения останутся и основные УВВ (их отношение к файловой системе вскоре станет понятным).

Для обеспечения удобства работы с записанными на диск сведениями их размещают в *файлах*. **Файлом** называется поименованная целостная совокупность данных на внешнем носителе информации. В этом контексте под **данными** понимается любая информация, включающая *программы* и *исходные данные* для их выполнения, *результаты выполнения программ, тексты, иллюстрации* и т.п.

Под **файловой системой** понимают функциональную часть ОС, обеспечивающую выполнение операций над файлами. В зависимости от файловой системы набор таких операций может меняться, но при этом всегда обеспечиваются возможности *создания* и *удаления* файлов, а также *считывание* их содержимого и *запись* информации в них.

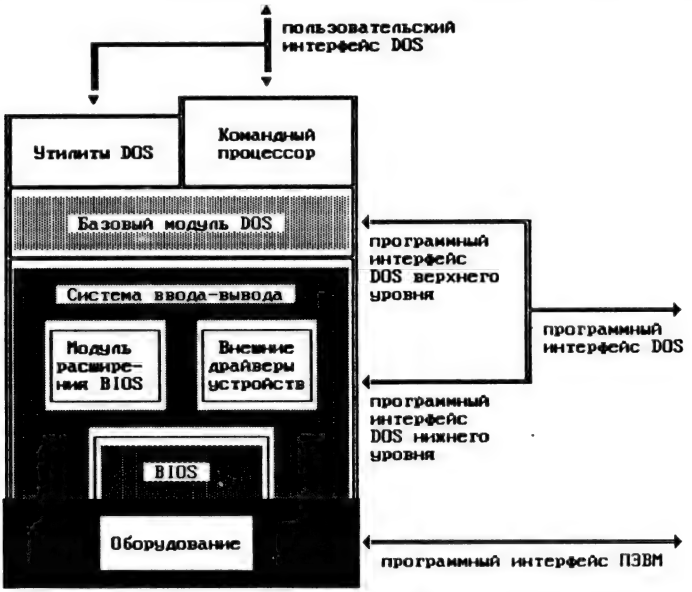


Рис. 5.1. Структура DOS

Файловая система выполняет следующие функции:
1) задает возможные *способы организации файлов*;
2) реализует *методы доступа к содержимому файлов*;
3) определяет *способы организации файловой структуры*;
4) предоставляет *средства манипулирования файловой структурой,* в том числе файлами.

Под **способом организации файла** понимают логическую структуру файла в терминах его компонентов и взаимосвязей между ними. Некоторые ОС обеспечивают работу с *последовательными, индексно-последовательными* и *библиотечными* файлами, а также с файлами *прямого доступа*. В настоящее время отсутствует четкая и общепризнанная классификация способов организации файлов. В большинстве предложенных вариантов в той или иной степени смешиваются понятия «организация» и «методы доступа», так как последние классифицируются проще. Поэтому на практике организация файла зачастую определяется через совокупность допустимых для нее методов доступа.

Методом доступа называют алгоритм запоминания и поиска записей (компонентов) в файле. Известны *последовательный, индексно-последовательный, библиотечный* и *прямой* методы доступа. Метод доступа определяет требования к организации файлов, чтобы он был применим. Так, последовательный метод доступа может быть применен к файлам с любой организацией, а прямой — только к файлам прямого доступа, имеющим специфическую логическую структуру с целью обеспечения адресации каждой записи. ВЗУ налагают свои ограничения на возможные методы доступа к хранимой на них информации. НМД может обеспечить любой метод доступа в связи с возможностью позиционирования головок непосредственно на заданный участок диска. НМЛ же поддерживают только последовательный метод доступа.

Под **файловой структурой** будем понимать совокупность файлов и взаимосвязей между ними. Файловая система может поддерживать тот или иной вид *(способ организации)* файловой структуры. В простейшем случае на диске создается каталог всех содержащихся на нем файлов для обеспечения доступа к ним (иначе при поиске файла пришлось бы просматривать весь диск). Более развитые файловые системы поддерживают *древовидную (иерархическую)* файловую структуру. Можно ожидать появления и *сетевых* файловых структур.

Средства манипулирования файловой структурой обеспечивают изменение конфигурации файловой структуры, в частности, создание файлов, удаление файлов и изменение взаимосвязей между ними, а также изменение содержимого файлов.

Файловые системы, как правило, имеют только программный интерфейс. Однако это не означает, что их услугами невозможно воспользоваться через пользовательский интерфейс ОС. Последняя может транслировать команды, вводимые пользователем, в программные запросы к файловой системе.

Определив понятие файловой системы как функциональной части ОС и обсудив ее функции, перейдем к рассмотрению конкретной файловой системы, содержащейся в DOS.

Файловая система DOS является наиболее развитой ее частью. Другие функциональные компоненты DOS (например, средства управления задачами и памятью) достаточно примитивны,

что объясняется однозадачностью данной системы. Понятие о файловой системе и особенно файловой структуре DOS необходимо иметь даже пользователям, не интересующимся принципами построения и функционирования этой ОС, так как через ее пользовательский интерфейс доступны многие средства файловой системы. Любому пользователю придется создавать и обслуживать файлы на дисках самостоятельно.

Структурно файловая система DOS распределена по БМ DOS и системе ввода-вывода.

Мы рассмотрим *устройства*, имеющие непосредственное отношение к файловой структуре, ее элементы, а затем кратко остановимся на ее функционировании. Основное внимание при этом будет уделяться возможностям, доступным через пользовательский интерфейс DOS.

Устройства

DOS различает два типа устройств: *посимвольные* и *поблочные*.

Обмен информацией между ОЗУ и **посимвольными устройствами** осуществляется побайтно и строго последовательно (байт за байтом). К устройствам этого типа относятся все УВВ и некоторые другие ПУ ПЭВМ.

В DOS зарезервированы следующие имена посимвольных устройств:

LPT1, или PRN — первый адаптер параллельного интерфейса, а точнее — то ПУ, которое к нему подключено (обычно принтер);

LPT2 — второй адаптер параллельного интерфейса;

LPT3 — третий адаптер параллельного интерфейса;

COM1, или AUX — первый адаптер последовательного интерфейса (дополнительная консоль, модем, принтер и т.п.);

COM2 — второй адаптер последовательного интерфейса;

COM3 — третий адаптер последовательного интерфейса;

COM4 — четвертый адаптер последовательного интерфейса;

NUL — фиктивное устройство;

CON — консоль (стандартно клавиатура при вводе и дисплей при выводе);

CLOCK$ — часы (для ПЭВМ класса AT и старше).

Фиктивное устройство может использоваться при отладке программ, а также для проверки читаемости какого-либо файла путем копирования его содержимого в это устройство, чтобы не создавать копию файла и тем самым ускорить процесс тестирования.

Для каждого из перечисленных устройств в DOS имеются соответствующие драйверы. Путем подключения внешних драйверов можно расширить список обслуживаемых посимвольных устройств.

Имена устройств можно, в частности, указывать пользователем в командах DOS для обмена информацией с ними.

Отметим, что строчные и прописные буквы DOS не различаются практически во всех контекстах. Поэтому вместо, например, CON можно указать Con, con и coN. Дело в том, что КП обычно автоматически переводит все строчные буквы из своего ввода в прописные.

Рассматривать одно *физическое* посимвольное устройство как несколько *логических* устройств не допускается, так как подключение нового драйвера для уже зарегистрированного в системе устройства (т.е. устройства с тем же именем) приводит к невозможности использования имеющегося драйвера. Это объясняется тем, что из-за совпадения имен устройств из цепочки драйверов всегда будет выбираться именно новый драйвер вместо используемого ранее. Кроме того, один драйвер может обслуживать только одно посимвольное устройство.

Посимвольные устройства здесь рассмотрены потому, что в DOS они трактуются как файлы (обычно как *текстовые*). Иными словами, в большинстве контекстов вместо имени файла можно задать имя посимвольного устройства. Если, например, вместо имени файла, размещенного на диске, в команде DOS указать имя такого устройства, то обмен будет осуществляться с последним, причем никаких дополнительных действий от пользователя не потребуется. Такая трактовка посимвольных устройств унифицирует средства обмена информацией с ПУ и обеспечивает гибкость как программ, так и команд DOS.

Обмен информацией между ОЗУ и **поблочными устройствами** на физическом уровне осуществляется *секторами* (блоками) по 512 Кбайт, которые можно считать *физическими записями*. Однако файловой системой может создаваться иллюзия обмена *логическими записями* меньшего или большего размера. К поблочным устройствам относятся ВЗУ как с прямым доступом (НМД, НОД), так и с последовательным доступом (в частности, НМЛ).

Поблочные устройства, каковыми обычно являются НМД, именуются буквами латинского алфавита (A, B, ...), а если их не хватает, то могут использоваться и другие символы, например, «<» и «/». Жесткое закрепление имен за НМД отсутствует. Имена назначаются в зависимости от порядка помещения драйверов в цепочку (список) во время загрузки DOS, т.е. от порядка их подключения, в частности, порядка указания внешних драйверов в файле CONFIG.SYS. Именование

начинается с A и продолжается в порядке английского алфавита. Если подключается драйвер НМД, для которого драйвер в цепочке уже имеется, то формируется новое имя устройства, а не подменяется старое. Поэтому в цепочке оказываются доступными по имени (под различными именами) все драйверы для данного НМД, что эквивалентно наличию нескольких *логических* устройств на одном *физическом*. Это удобно, в частности, для работы в одном дисководе с разными форматами дискет. Есть и другие достоинства рассмотренного механизма. Один драйвер может управлять несколькими поблочными устройствами одновременно.

Например, если ALPHA — первый драйвер в списке, и он определяет два поблочных устройства, то им будут присвоены имена A и B. Если за ALPHA подключается и помещается в список другой драйвер BETA для этих же устройств, то им будут назначены имена C и D соответственно. В этом случае один НМД доступен под именами A и C, а другой — под B и D.

Ограничения на количество логических НМД очень слабые: общее их число может достигать 63.

DOS организует свою работу таким образом, что физически первый НГМД получает имя A, а второй — имя B. Физические номера накопителей отмечаются на корпусе ПЭВМ, например, соответствующим числом точек. Если ПЭВМ имеет только один НГМД, то для него назначается два имени — A и B, благодаря чему создаются два логических привода. Это позволяет, в частности, копировать содержимое дискет без промежуточного его переноса на жесткий диск. DOS всегда «помнит», какой из двух логических дисководов в настоящее время *активен* (это не одно и то же, что *текущий* дисковод). В случае, когда активен дисковод A, при обращении к приводу B DOS предложит установить дискету в последний, что пользователь должен и осуществить. Для этого следует изъять из дисковода дискету, установить новую и нажать любую клавишу. Система теперь будет работать с приводом B, считая его активным, а потом снова может запросить дисковод A и т.д. НЖМД всегда получает имя C, а возможно, и ряд других имен.

При загрузке DOS на ЕС1840 обычно подключают внешний драйвер, позволяющий рассматривать не только каждый дисковод как два логических устройства, но и один диск как два логических диска емкостью по 360 Кбайт каждый. Дело в том, что в ПЗУ ЕС1840 прошит драйвер для 360-Кбайт НГМД, а реально (если читатель помнит) установлены приводы на 720 Кбайт. Для наиболее полного использования их возможностей и осуществляется подключение внешнего драйвера. Дополнительно можно подключить и внешний драйвер, обеспечивающий работу с одним диском емкостью 720 Кбайт. Это, конечно, более удобно. Но отказываться полностью от предыдущего драйвера нецелесообразно, так как загрузка DOS с 720-Кбайт дискеты не идет, а пренебрегать 360-Кбайт памятью на системном диске крайне нежелательно. Разбиение одного диска на два логических осуществляется в результате того, что первому логическому диску отводится 40 цилиндров (из 80-ти) через цилиндр, а второму — оставшиеся 40 цилиндров. Информация, записанная на первом логическом диске, может быть прочитана в 360-Кбайт НГМД и наоборот, информацию, записанную на 360-Кбайт приводе, можно прочитать в НГМД ЕС1840, если считать, что эта информация находится на первом логическом диске (A или B в зависимости от того, в какой привод дискета установлена).

В командах DOS за именем привода всегда ставится двоеточие, например: A, b:.

Файловая система DOS в отличие от UNIX не позволяет на уровне пользовательского интерфейса рассматривать содержимое диска как единый файл и осуществлять доступ к нему путем указания имени привода. Однако в ней, аналогично UNIX'у, обеспечивается возможность создания на диске иерархической файловой структуры. Иными словами, файловой системой DOS поддерживается размещение на диске иерархически организованной совокупности файлов и доступ к любому файлу (но не к содержимому всего диска как к файлу). Тем не менее, можно обратиться к любому заданному сектору диска (так называемое прямое чтение и прямая запись) без привязки к файловой структуре, но, конечно, не через пользовательский интерфейс DOS.

Таким образом, файловая система DOS поддерживает две разновидности доступа к содержимому дисков:

1) *прямое чтение* и *прямая запись*, что позволяет с определенными допущениями рассматривать содержимое диска как единственный файл, но только на уровне программного интерфейса;

2) *доступ к файлам* (как на уровне программного, так и пользовательского интерфейса).

В DOS всегда имеется один **текущий дисковод**. Первоначально (при загрузке DOS) текущим является привод, на котором установлен системный диск (обычно A или C). Затем он может быть изменен предназначенной для этого командой DOS. Смысл текущего дисковода в том, что при обращении к нему явное указание в командах DOS его имени не требуется. Следовательно, работа пользователя упрощается. *Диск*, установленный в текущий дисковод, также будем называть *текущим*.

В дальнейшем под диском будем понимать как жесткий диск (точнее — логический диск на нем), так и дискету. Аналогично, если речь будет вестись просто о дисководе, приводе или накопителе, то будет иметься в виду как НГМД, так и НЖМД. При необходимости разновидность диска (привода) будет конкретизироваться.

В подразделе 5.6 нам понадобится еще одно не введенное пока понятие, а именно, «том». Вообще под **томом** понимают съемный носитель информации, но в рамках DOS этот термин считается синонимом диска.

Файлы

DOS не предоставляет развитых *средств организации* файлов на дисках, однако имеющихся средств достаточно для программирования обработки файлов любой структуры, причем с использованием необходимых методов доступа. На *уровне файловой системы* файл трактуется как последовательность логических записей. Такая организация называется **последовательной.**

На логическом уровне дисковая память рассматривается как непрерывная последовательность секторов, каждый из которых имеет свой *номер.* Память создаваемому файлу выделяется *динамически* (по мере необходимости) в начале свободной области логического дискового пространства, причем не секторами, а **кластерами** (обычно 2 или более смежных сектора). Дискретность в терминах кластеров, а не секторов, объясняется необходимостью минимизации объема системной информации. На основе этой информации обеспечивается возможность работы с *фрагментированными* (занимающими несмежные области логического дискового пространства) файлами, что весьма важно для используемого механизма динамического выделения памяти. Фрагментация файлов возникает при их интенсивном создании, удалении и пополнении. Допустим например, что сначала был создан файл X и в него записана определенная информация, а затем — файл Y. Если теперь понадобится дописать данные в файл X, то они будут размещены в кластерах за файлом Y. Эта ситуация иллюстрируется на рис. 5.2.

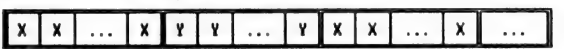


Рис. 5.2. Фрагментация файла X

Фрагментация файлов увеличивает время доступа к их содержимому и существенно затрудняет восстановление файлов при разрушении (случайном по вине пользователя или из-за аппаратных сбоев) файловой структуры. Поэтому фрагментации файлов следует избегать, регулярно используя для упорядочения размещения информации на диске специальные *утилиты-дефрагментаторы.*

Недостаток выделения дисковой памяти кластерами вместо секторов состоит в большом расходовании дискового пространства при размещении множества маленьких файлов (кластеры окажутся полупустыми).

Более подробно порядок размещения информации на дисках мы рассмотрим позже, в подразделе 5.12.

Полностью *статическое* выделение дисковой памяти для файла с указанием требуемого размера при его создании неприемлемо, так как:

1) даже приблизительный размер файла заранее неизвестен, а задание максимально возможной длины приводит к неэффективному использованию внешней памяти;

2) в случае необходимости увеличения длины файла сверх указанной при его создании (в случае пополнения файла) придется создавать новый файл большего размера и записывать в него всю старую, а также новую информацию, что приводит к большим временным затратам;

3) неэффективно используется дисковая память, так как под файл приходится выделять, как правило, бóльшую область, чем ему реально требуется.

В некоторых ОС принята *смешанная схема* выделения памяти под файлы: первоначально статически выделяется заданная область, а при ее заполнении — распределяется дополнительная область определенной длины и т.д. Это в некоторой степени уменьшает фрагментацию.

Несмотря на последовательную организацию файлов и возможность их фрагментации, файловая система DOS поддерживает не только последовательный, но и прямой методы доступа к их содержимому.

При *последовательном доступе* записи из файла считываются только в порядке их расположения в файле. Поэтому, чтобы обратиться к определенной записи, необходимо считать все предыдущие.

При *прямом доступе* обеспечивается непосредственное обращение к записи по ее номеру в файле.

В связи с поддержкой прямого доступа есть возможность реализовать любые другие методы доступа. Эта возможность может быть использована создателями систем программирования или разработчиками программного обеспечения. Например, логически организовав файл определенным образом и разработав соответствующие подпрограммы, можно предоставить возможность применения библиотечного метода доступа. Как мы уже знаем, свои методы доступа к данным предлагают СУБД. Возможно также использование и других *систем управления данными.*

Подводя итог сказанного, обратим Ваше внимание на рис. 5.3, где представлена полная иерархия *средств управления данными,* которые основываются на файловой системе DOS. Соприкосновение тех или иных средств по горизонтали означает использование средством, которое расположено выше, возможностей, предоставляемых находящимся под ним средством (или средствами).

DOS различает файлы в двух форматах, а именно, двоичные и текстовые файлы. Другие программные продукты могут поддерживать файлы в своих, специфических форматах.

Двоичный файл — это файл общего вида, на содержимое которого не накладывается никаких ограничений. Считается, что он состоит из последовательности байтов, возможно, сгруппированных в логические записи *фиксированной длины.* Последняя запись может быть неполной. В двоичных файлах хранятся исполняемые программы и «снимки» основной памяти, т.е. какие-либо данные во внутреннем представлении. Несмотря на то, что ограничения на содержимое двоичных файлов отсутствуют, файлы с исполняемыми программами при их запуске на выполнение должны иметь

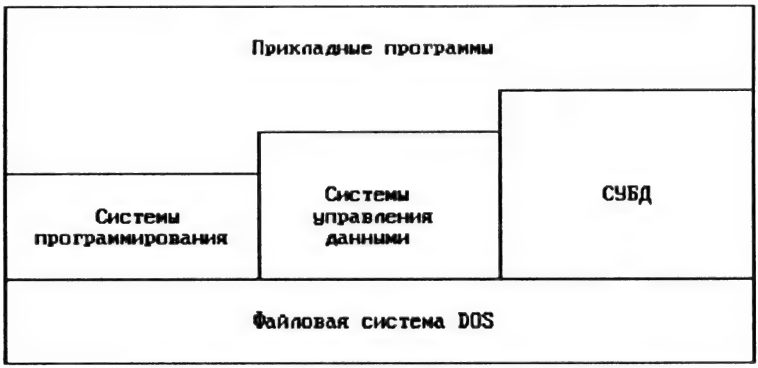


Рис. 5.3. Иерархия средств управления данными

определенную структуру, что DOS обязательно анализирует. В других же случаях (при записи в двоичный файл или чтении из такого файла) его содержимое абсолютно никакой роли не играет и ни на что не влияет.

Текстовым (ASCII-) файлом называется файл, содержимое которого без преобразования может быть выдано на экран дисплея или принтер и непосредственно воспринято человеком. Такой файл состоит из последовательности *строк переменной длины*, которые можно считать логическими записями, так как файловая система DOS обеспечивает ввод-вывод строк. Каждая строка содержит только *текстовые символы (символы пишущей машинки)* и завершается *маркером конца строки* (EOL — End Of Line). Роль маркера EOL играет *символ возврата каретки* (CR), за которым, возможно, следует *символ перевода строки* (LF). В качестве исключения внутри строк текстового файла допустим *символ горизонтальной табуляции* (HT), заменяющий несколько расположенных подряд пробелов и обеспечивающий сдвиг остатка строки вправо, к очередной позиции табуляции, при выдаче файла на устройство вывода. Текстовый файл обычно завершается *маркером конца файла* (EOF — End Of File), роль которого играет *символ подстановки* (SUB). Однако иногда маркер EOF может отсутствовать. Часто между некоторыми строками текстового файла размещают еще один управляющий символ — *символ перевода формата* (FF), обеспечивающий прогон бумаги на принтере до начала следующей страницы. На экране же дисплея он отображается как таковой и ни к каким действиям не приводит. Чтобы узнать коды упомянутых здесь символов, следует обратиться к подразделу 5.3. В текстовых файлах хранят различные текстовые документы, в том числе исходные программы, данные к ним, а также окончательные результаты их выполнения. Конечно, для хранения данных экономнее использовать двоичные файлы, однако такие файлы не могут быть проинтерпретированы человеком. Поэтому данные записывают одной программой в двоичный файл только тогда, когда они в последующем понадобятся другой или этой же программе.

Многие программные продукты, в частности, текстовые редакторы, создают файлы, *близкие к текстовым,* но содержащие дополнительные управляющие символы, например, для задания шрифтов, интервалов, разбивки текста на абзацы или строки и т.п. При копировании таких файлов на экран дисплея или принтер средствами DOS отнюдь не всегда удается получить нормально читаемый документ. Несмотря на наличие управляющих символов, их иногда называют текстовыми (но не ASCII-) файлами.

Однако данные файлы не удовлетворяют приведенному нами определению, а поэтому мы будем их считать **базами данных** для соответствующих программных продуктов.

Важно подчеркнуть, что УВВ в DOS трактуются именно как текстовые файлы. Однако текстовый файл на диске, в отличие от файла-УВВ, может быть проинтерпретирован и как последовательность записей определенной (в частности, единичной) длины. При копировании текстового файла на экран дисплея или принтер маркер конца строки реально приводит к продолжению вывода последующих символов с новой строки. Конечно, можно скопировать на устройство вывода и любой другой файл, но проинтерпретировать его содержимое человеком будет практически невозможно (произвольные 8-разрядные коды переводятся в символы-«иероглифы», звуковые сигналы и т.п. или вообще не воспринимаются, если данный код не имеет графического представления и никак на устройство не действует).

Дополнительно к сказанному, Вы должны уяснить, что при вводе с клавиатуры DOS моделирует чтение из текстового файла. Строка при этом завершается нажатием клавиши Enter, а весь файл — нажатием комбинации клавиш Ctrl-Z.

С каждым файлом в DOS связываются:

1) составное имя файла;
2) атрибуты файла;
3) дата создания файла;
4) время создания файла;
5) длина файла.

Составное (полное) имя файла представляет собой совокупность *имени файла* и *расширения имени файла*.

Имя файла — это последовательность из не более восьми символов, в качестве которых допускается использовать буквы латинского алфавита от A (a) до Z (z), цифры от 0 до 9, а также символы !, @, #, $, %, &, (,), -, _, ~, ', ‘, ^, { и }. Можно также использовать буквы национального алфавита, поддерживаемого активизированной кодовой страницей. Никаких ограничений на порядок следования символов в имени файла не налагается, а одноименные строчные и прописные буквы (как и практически везде в DOS) не различаются. Возможно, легче запомнить недопустимые в имени файла символы, а именно: символ пробела, /, запятая, ;, +, [,], ", =, *, ?, :, ., \, |, < и >. В качестве имен файлов по вполне понятным причинам нельзя использовать имена посимвольных устройств.

Например, последовательности символов PROG, 123, PROG_1, $IOBUF и fileinfo являются допустимыми именами файлов, а MAS[5], 24/11/90, #invoices и con не являются таковыми.

Имя файла — это обязательный элемент и поэтому при доступе к файлу оно должно указываться всегда.

Расширение имени файла характеризует тип (содержимое) файла и представляется последовательностью из не более трех символов, допустимых для имени файла. При этом ограничения на использование имен устройств в качестве расширения отсутствуют.

Например, расширение PAS обычно обозначает исходную программу на языке Pascal.

Расширение имени файла является необязательным элементом и поэтому при создании файла может не задаваться. Однако это снижает информативность составного именования файла.

В командах DOS и программах на языках программирования составное имя файла записывается как имя файла, за которым следует расширение, отделенное от имени *точкой*.

Например, PROG1.PAS может именовать файл, содержащий исходную программу под номером 1 на языке Pascal. Однако con.bas будет интерпретироваться как устройство CON, так как в случае использования в качестве имени файла имени посимвольного устройства расширение отбрасывается.

Если файл не имеет расширения, то оно, естественно, при обращении к нему не указывается. В этом случае точка тоже может быть опущена (например, PROG1. или PROG). Но если требуется подчеркнуть отсутствие расширения в том контексте, где оно подразумевается по умолчанию, то точка необходима.

Умолчания расширений файлов поддерживают многие распространяемые программные продукты. При работе с таким ПО крайне желательно придерживаться принятых в нем соглашений. В этом случае явно указывать расширения существующих и создаваемых файлов необязательно, что упрощает диалог. Но сказанное не означает, что расширений имен файлов реально не существует и что они реально не будут созданы.

Таким образом, расширение имени файла не указывается в двух случаях:

1) когда оно отсутствует;

2) когда оно подразумевается по умолчанию и правила умолчания не противоречат тому, что требуется выполнить.

Если в данном контексте умолчания имеются, а Вам нужно указать файл без расширения, то за именем файла должна следовать точка.

С учетом сказанного пользователь может в остальном использовать произвольные расширения, но желательно, чтобы они, как и имена файлов, несли какую-либо смысловую нагрузку.

DOS при запуске программ в случае указания только имени существующего файла предполагает одно из следующих расширений: COM, EXE или BAT.

Расширение COM (от COMmand) зарезервировано для файлов, содержащих готовые к выполнению машинные программы, не требующие перемещения при загрузке их в ОЗУ для выполнения независимо от адреса загрузки. Иными словами, программа инвариантна к адресу загрузки. Она может быть загружена в ОЗУ и выполнена без настройки содержащихся в ней адресов по месту загрузки. Размер такой программы не может превышать одного сегмента (64 Кбайт). COM-программу назовем **позиционно независимой.**

Расширение EXE (от EXEcutable) зарезервировано для файлов, содержащих готовые к выполнению машинные программы, которые при загрузке на выполнение в ОЗУ требуют настройки адресов, что увеличивает общее время выполнения программ. Такой программе предшествует заголовок, содержащий необходимую для перемещения информацию, а ее размер может быть больше 64 Кбайт. EXE-программу назовем **позиционно зависимой.**

Как позиционно независимые, так и позиционно зависимые программы являются *перемещаемыми* в том смысле, что могут быть размещены для выполнения в любом месте ОЗУ. В отличие от перемещаемой *абсолютная программа* должна загружаться в память по вполне определенному адресу. Выполнение абсолютных программ DOS непосредственно не поддерживает.

Любой файл, содержащий исполняемую программу, будем называть **программным**.

Расширение BAT (от BATch) зарезервировано для **командных файлов**, т.е. текстовых файлов, содержащих программы на командном языке DOS.

Файлы с расширениями COM, EXE и BAT (если их содержимое соответствует тому, что описано выше) называются **исполняемыми**. Исполняемыми являются также файлы и с другими расширениями, если они содержат готовые к выполнению машинные программы или программы

на командном языке. Однако такие файлы не всегда могут быть выполнены без предварительного явного изменения расширения в соответствии с их содержимым.

DOS распознает формат COM- и EXE-файлов (но только их) по содержимому, а не по расширению. Поэтому файлы с перемещаемыми программами могут иметь любое расширение, но тогда при запуске такого файла на выполнение расширение придется указать явно. Изменение расширений программных файлов можно иногда применять для защиты от компьютерных вирусов. Вместе с тем командный файл обязательно должен иметь расширение BAT, чтобы его можно было выполнить.

Далеко не полный перечень умолчаний расширений имен файлов, используемых существующими программными продуктами, а также часто используемых расширений, ставших в определенном смысле стандартными, в алфавитном порядке приведен в табл. 5.1. Этот список поможет пользователю уяснить назначение и содержимое неизвестных файлов.

Таблица 5.1

Соглашения по расширениям имен файлов

Расширение	Назначение
ARC	Архив
ARJ	Архив
ASM	Программа на языке Ассемблера
BAK	Предыдущая (резервная) версия файла
BAS	Программа на языке Basic
BAT	Командный файл DOS
BIN	Двоичный файл, в том числе машинная программа или драйвер
C	Программа на языке C
CFG	Файл, описывающий конфигурацию (параметры) программы
CHI	Документ текстового редактора ChiWriter
COB	Программа на языке Cobol
COM	Позиционно-независимая машинная программа
CPI	Файл кодовых страниц ПУ
DAT	Файл данных
DB	База данных
DBF	Файл данных с развитой организацией, в том числе база данных
DOC	Документ, подготовленный одним из текстовых редакторов, в частности, Microsoft Word
DRV	Драйвер
EXE	Позиционно-зависимая машинная программа
EXT	Файл расширений
FNT	Файл со шрифтами для принтера
FOR	Программа на языке Fortran
HLP	Файл, содержащий экранный интерактивный справочник
ICE	Архив
INI	Файл, описывающий конфигурацию (параметры) программы
LET	Письмо
LIB	Библиотека подпрограмм
LSP	Программа на языке LISP
LST	Листинг программы
LZH	Архив
MAC	Макрокоманда на языке Ассемблера
MAP	Листинг компоновщика
MNU	Файл меню
MSG	Файл сообщений программы
OBJ	Объектная программа
OVG	Оверлейный файл
OVL	Оверлейный файл
OVR	Оверлейный файл
PAK	Архив
PAS	Программа на языке Pascal
PCX	Иллюстрация графического редактора PC Paintbrush
PIC	Иллюстрация
PIF	Оверлейный файл
PRN	Файл, подготовленный для вывода на принтер
PRO	Программа на языке Prolog
REF	Таблица перекрестных ссылок
SEQ	Программа на языке Forth
SYS	Драйвер, если имя файла отлично от IO, MSDOS и CONFIG
TMP	Временный файл
TPL	Библиотека объектных модулей Turbo Pascal'я
TPU	Объектный модуль Turbo Pascal'я
TXT	Текстовый файл
$$$	Временный файл
ZIP	Архив
ZOO	Архив

Атрибуты файла определяют *способы* его *использования* и *права доступа* к нему. DOS допускает задание следующих атрибутов:

R (Read-only) — файл предназначен *только для чтения* и не может быть ни удален, ни изменен (однако можно скопировать файл и изменить или удалить его копию);

A (Archive) — *архивный файл* (лучше говорить *«неархивированный»*); этот атрибут приписывается каждому создаваемому файлу и сбрасывается средствами архивирования и резервирования файлов; может использоваться этими средствами для определения файлов, подлежащих архивации или резервированию;

H (Hidden) — *скрытый файл*, который игнорируется многими командами DOS;

S (System) — *системный файл*.

В существующих версиях DOS атрибуты H и S интерпретируются одинаково, но в будущем они будут, видимо, различаться.

Файлу могут быть присвоены одновременно любые из перечисленных атрибутов или ни один из них. В последнем случае файл называется *обычным* и к нему применимы все возможные операции.

Дата создания и время создания приписываются файлу по показаниям *системных часов*. Если они установлены неправильно, то это отразится и на соответствующей информации, связываемой с файлом, что может спутать карты. Для установки даты и времени в ПЭВМ класса XT нужно использовать DOS-команды DATE и TIME. ПЭВМ класса AT и старше имеют встроенные часы-календарь с автономным питанием. Поэтому установка даты и времени после загрузки DOS необязательна. Вы должны только своевременно корректировать показания системных часов по мере необходимости. При обновлении файла дата и время его создания корректируются в соответствии с текущими показаниями системных часов.

Длина файла указывается в байтах и связывается с ним после его создания или обновления. Для текстовых файлов длина не имеет особого значения, так как файл обычно ограничивается в конце маркером EOF. Для двоичных же файлов длина существенна в связи с тем, что они считываются с учетом ее значения.

В файловой системе DOS воплощена идея **стандартного ввода-вывода**. В качестве стандартных (текстовых) файлов ввода и вывода выступает устройство CON. Программы, использующие средства стандартного ввода-вывода вместо явного указания устройств, обладают более высокой универсальностью. Действительно, стандартное УВВ может быть переопределено средствами DOS, а следовательно, модификация программ для перенаправления ввода-вывода не потребуется.

Файловая система DOS обеспечивает, наряду с другими, выполнение следующих основных операций над файлами:

1) создание и удаление файлов;
2) переименование и пересылку файла в другой каталог;
3) позиционирование магнитных головок на заданную запись в файле;
4) чтение, запись, а также обновление (чтение и запись) файлов;
5) поиск файлов;
6) считывание и смену атрибутов файлов;
7) считывание и изменение даты и времени создания, а также длины файлов;
8) перенаправление стандартного ввода-вывода.

К *обновлению* относятся операции замены записей в файле, добавления новых записей в конец и отсечение конца файла, которые производятся над существующим файлом без его реорганизации, требующей перезаписи.

Перед выполнением над ним операций файл должен быть соответствующим образом *открыт* (для чтения, записи или обновления), а после использования — *закрыт* (автоматически при завершении выполнения программы файл закрывается некорректно!).

Все операции над файлами, а также открытие и закрытие файлов доступны через прерывания верхнего уровня.

Часть перечисленных операций реализована также через команды DOS. Есть и другие команды, более мощные, чем базовые средства файловой системы DOS. Здесь они не характеризуются, так как командный язык DOS в этой книге рассматривается позже и весьма подробно (см. подраздел 5.6).

Системами программирования, как правило, предлагаются дополнительные средства по работе с файлами, которые реализованы через базовые операции файловой системы и последней непосредственно не поддерживаются.

DOS предоставляет средства для указания не одного, а сразу группы существующих в том или ином каталоге файлов путем задания так называемых шаблонов. Наиболее последовательно эти средства реализованы в командном языке DOS, а на уровне программного интерфейса файловой системы — только частично.

Шаблоном (образцом) является составное имя файла, в полях имени и/или расширения которого используются *символы-заменители (глобальные символы)*. Шаблон обозначает не единственный файл, а *группу существующих файлов,* составные имена которых *сопоставляются* с

данным шаблоном, т.е. подходят под образец. *Область действия образца* ограничивается содержимым определенного каталога (см. следующий подпункт).

DOS использует символы-заменители ? и *.

Заменитель ? обычно указывает на любой (но единственный) символ в данной позиции. Однако если заменитель ? записан последним в поле имени файла (расширения) или если за ним записаны только такие же заменители до конца поля, то он обозначает любой символ или его отсутствие. В полях имени файла и расширения допускается использовать любое количество заменителей ?.

Примеры шаблонов с заменителем ?:

- MEMO?R.EXE подходит к любому файлу с расширением EXE, имя которого начинается с MEMO, завершается символом R, а между ними находится ровно один произвольный символ;
- PROG.??M сопоставляется со всеми файлами, имеющими имя PROG и *трехсимвольное расширение*, если последним символом в расширении является M;
- GL???.DAT подходит ко всем файлам с расширением DAT, имеющим имя, начинающееся с символов GL, за которыми следует *не более* трех символов.

Заменитель * обозначает произвольную последовательность символов (возможно, нулевой длины) от данной позиции до конца поля имени файла или расширения (в соответствии с тем, в каком поле заменитель использован). В каждом поле (имени или расширения) допускается только по одному заменителю *, а все символы, которые указаны в поле за ним, игнорируются.

Примеры шаблонов с заменителем *:

- *.BAS сопоставляется со всеми файлами с расширением BAS;
- P*.PAS подходит ко всем файлам с расширением PAS, имя которых начинается с символа P;
- *.* сопоставляется со всеми файлами, в том числе без расширений;
- *.*S — то же, что в предыдущем примере;
- *. подходит ко всем файлам, не имеющим расширения;
- * — то же, но только на уровне интерфейса DOS;
- DIR*1.MEM сопоставляется со всеми файлами с расширением MEM и именем, начинающимся с DIR, независимо от наличия символа 1 в конце имени файла.

Сделаем два замечания:

1) к примеру, шаблоны * и *.* не эквивалентны, так как заменитель не может замещать разделитель имени файла и расширения (.);

2) при использовании заменителей ограничения на длину имени файла и расширения в образце не снимаются.

Полезность шаблонов состоит не только в возможности задать множество файлов, но и в возможности упростить задание составного имени единственного файла. Например, вместо существующего файла PROGRAM.PAS часто достаточно указать P*.*, если при этом не возникает неоднозначности.

Каталоги

Файловая система DOS позволяет объединять файлы в каталоги. **Каталогом** называется специальный файл, в котором *регистрируются* другие файлы. Если файл зарегистрирован в каталоге, то говорят, что файл *входит* в каталог или *содержится* в каталоге. Вхождение файла в каталог означает, что в последнем содержится вся характеризующая файл информация и сведения о том, в каком месте диска файл расположен. Сам же файл хранится как последовательность байтов без каких-либо дополнительных справочных сведений. Каталог, в свою очередь, может входить в другой каталог, благодаря чему на диске может быть организована разветвленная файловая структура.

К каталогам, хотя они и являются файлами, неприменимы стандартные операции, рассмотренные выше. Правила именования каталогов (за исключением корневого) совпадают с правилами именования файлов, однако расширения, как правило, не используются и точка при этом не ставится.

В дальнейшем для удобства изложения мы будем проводить между *файлом* и *каталогом* четкую границу.

На каждом диске всегда имеется единственный **корневой каталог**, именуемый символом \, в который могут входить другие каталоги и файлы. Корневой каталог создается при форматировании (разметке) диска, хранится во вполне определенной области дисковой памяти, имеет ограниченный размер и не может быть удален никакими средствами. По сути пользователь не имеет возможности что-либо сделать с корневым каталогом, за исключением помещения в него файлов и других каталогов, а также удаления файлов и каталогов из него.

Каждый диск хранит свою *файловую структуру*, которая формируется по следующим правилам:

1) каталог или файл может входить только в один каталог;

2) допускается вхождение в различные каталоги каталогов и файлов с одинаковыми именами (но, конечно, не в один каталог);

3) на порядок следования файлов и каталогов в каталоге никаких ограничений (за исключением корневого каталога системного диска) не накладывается;

4) глубина вложенности каталогов не ограничивается.

Следовательно, файловая система DOS обеспечивает формирование *иерархической многоуровневой,* а короче — *древовидной* файловой структуры, в корне которой находится корневой каталог, а листьями являются файлы и, возможно, пустые каталоги.

Пример файловой структуры приведен на рис. 5.4. Эта файловая структура состоит из каталогов \ (корневой каталог), IVANOV, PETROV, EXE, двух каталогов PROGS и двух каталогов DATA. В частности, каталог IVANOV содержит два каталога — PROGS и DATA. Различные файлы prog1.pas входят в различные каталоги PROGS, которые содержатся в каталогах IVANOV и PETROV, зарегистрированных в корневом каталоге диска.

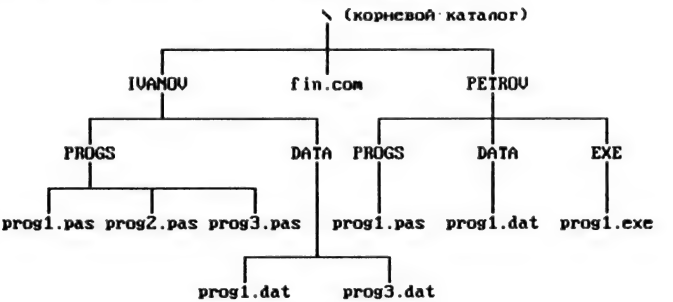


Рис. 5.4. Пример файловой структуры

Если один каталог входит в другой, то первый называется **дочерним каталогом (подкаталогом)** второго, а второй — **родительским каталогом (надкаталогом)** первого. Если в дереве файловой структуры существует путь от одного каталога к другому, направленный от корневого каталога к листьям, то второй каталог является **подчиненным каталогом** первого. Например (см. рис. 5.4) каталоги PROGS и DATA одновременно выступают в роли дочерних и подчиненных каталогов IVANOV, а также в роли только подчиненных каталогов корневого каталога; каталог IVANOV является родительским каталогом каталогов PROGS и DATA.

Порядок вхождения файлов и подкаталогов в каталог в принципе не существенен, однако он влияет на последовательность их обработки командами DOS при групповых операциях и на время доступа к ним.

В процессе работы с жестким диском желательно обеспечить сбалансированность файловой структуры (чтобы не было очень длинных ветвей наряду с очень короткими). На дискетах обычно каталоги (дополнительно к корневому) не создают, так как на них может быть размещено сравнительно мало файлов и к тому же каталоги требуют дополнительной памяти.

В один каталог обычно объединяют группу файлов (каталогов), связанных между собой по какому-либо признаку, например, файлы (каталоги) одного владельца, функционально подобные файлы (каталоги), файлы, имеющие однотипное содержимое (тексты, исходные программы и т.п.).

В отличие от ОС UNIX файловая система DOS предлагает весьма ограниченные возможности сцепления отдельных файловый структур на установленных в данный момент дисках в единую структуру, но таковые все же имеются.

Достоинства организации древовидной файловой структуры в DOS (т.е. структуризации множества файлов) состоят в следующем:

1) в хорошей *визуализации* структурированного множества файлов (пользователю не требуется работать с очень большим списком файлов одновременно);

2) в возможности *разграничения доступа* к файлам (пользователь, работающий со своим фрагментом файловой структуры, не сможет по оплошности разрушить информацию в оставшейся части этой структуры);

3) в *локализации имен* (область действия имен ограничивается каталогами, где они зарегистрированы, что уменьшает вероятность коллизии имен и вызванного ею разрушения информации);

4) в возможности *манипулирования группой файлов,* входящих в один каталог (содержимым каталога), как единым целым;

5) в *ограничении множества выбираемых по шаблону файлов* заданным каталогом;

6) в *ускорении доступа* к требуемому файлу в результате продвижения по цепочке каталогов, ведущей к нему, вместо его поиска на всем множестве файлов.

Недостатки же заключаются в некотором усложнении файловой системы и незначительном увеличении объема дисковой памяти для размещения файловой структуры.

Чтобы ускорение доступа к файлу в древовидной файловой структуре стало возможным, вместе с составным именем файла файловой системе необходимо передавать *маршрут* его поиска по каталогам. Это же требуется и при создании файла, так как иначе файловая система не сможет определить, где его размещать. В принципе при доступе к существующему файлу без задания маршрута можно было бы обойтись, но тогда файловой системе пришлось бы просматривать все дерево каталогов, а область действия имен не могла бы быть локализована.

Полным маршрутом (путем) к файлу называется последовательность каталогов, ведущая от корневого каталога к этому файлу. Полный маршрут представляется перечислением имен каталогов, разделенных символом \, причем корневой каталог от его дочернего каталога символом \ не отделяется. Совпадение разделителя каталогов в маршруте с именем корневого каталога ни к каким неприятностям не приводит.

Примеры полных маршрутов (см. рис. 5.4):

- строка \IVANOV\PROGS является полным маршрутом к файлам prog1.pas, prog2.pas и prog3.pas;
- символ \ является полным маршрутом к файлу fin.com.

Очевидно, указание полных маршрутов в разветвленной файловой структуре на жестком диске для доступа к файлам утомительно. DOS предоставляет следующие три возможности, использование которых позволит избежать задания полных маршрутов в большинстве случаев.

Во-первых, DOS хранит информацию о текущем каталоге для каждого дисковода ПЭВМ. Первоначально после загрузки системы текущими каталогами для каждого привода являются корневые каталоги дисков. Затем их можно изменить одной из команд DOS. **Текущим** называется такой каталог, которым заканчивается полный маршрут к файлу в случае, когда маршрут явно не задан. Например (см. рис. 5.4), если текущим является каталог EXE, то для доступа к файлу prog1.exe или создания в этом каталоге новых файлов никакой маршрут указывать не нужно. Текущий каталог может быть использован и для доступа к содержимому всех подчиненных ему каталогов. Для этого маршрут нужно задать без корневого каталога и указать в нем путь по подчиненным каталогам. Так (см. тот же рисунок), если текущим является каталог IVANOV, то для доступа к файлам prog1.dat или prog3.dat, а также для создания в каталоге DATA новых файлов достаточно задать маршрут DATA вместо \IVANOV\DATA.

Очевидно, рассмотренный способ не является полным, так как не позволяет подняться по файловой структуре вверх. Его дополняет следующая предоставляемая DOS возможность.

Прежде чем ее описать, для удобства дальнейшего рассмотрения определим **рабочий каталог** как *текущий каталог текущего диска.*

Во-вторых, каждый каталог, за исключением корневого, наряду с элементами, описывающими содержащиеся в нем файлы и дочерние каталоги, имеет *два специальных элемента,* обозначаемых через . и .. . Элемент . (точка) является ссылкой каталога на самого себя, т.е. интерпретируется как «этот каталог». Элемент .. (две точки) указывает на родительский каталог, т.е. интерпретируется как «родительский каталог данного каталога». Первый элемент используется для указания в командах DOS рабочего каталога, если задание каталога обязательно. Применение второго элемента позволяет подняться по файловой структуре вверх и тем самым облегчить задание маршрутов к содержимому вышестоящих каталогов текущего каталога, а также к содержимому подчиненных им каталогов. Например (см. рис. 5.4), если текущим является каталог EXE, то маршрут ..\PROGS позволит обратиться к файлу prog1.pas, а маршрут ..\.. — к файлу fin.com.

Таким образом, маршрут к файлу может быть задан абсолютно или относительно. *Абсолютным* является полный маршрут. Для указания *относительного* маршрута, т.е. маршрута относительно текущего каталога, используется первый из описанных способов, возможно, дополненный вторым способом (..). Относительное задание маршрута зачастую оказывается более простым и повышает гибкость программных продуктов, так как появляется возможность настраивать последние путем смены текущего каталога.

В-третьих, DOS хранит определенный пользователем список полных маршрутов, которые используются при поиске существующего файла, если окажется, что он по заданному маршруту не обнаружен или этот маршрут не указан (при отсутствии маршрута подразумевается текущий каталог, так что оба случая эквивалентны). Более детально речь об этом пойдет в п. 5.6.5. Последовательность каталогов, ведущую к файлу, и не начинающуюся корневым каталогом, будем называть **неполным маршрутом** к файлу. Понятия полного и неполного маршрута объединим общим термином **«маршрут»**. Каталог, указанный последним в маршруте, или текущий каталог, если маршрут не определен, назовем **выделенным каталогом**. В нем будет осуществляться первоначальный поиск существующего или создание нового файла.

Файловая система DOS реализует операции создания и удаления каталога, которые доступны и через командный язык DOS.

Завершив рассмотрение организации DOS, выделим следующие принципы, положенные в ее основу:

1) *модульность* и *многослойность структуры,* что облегчает ее развитие и адаптацию к новому оборудованию;

2) *открытость архитектуры,* допускающей наращивание возможностей DOS, в частности, подключение новых драйверов и нового командного процессора;

3) *унификацию средств и методов обмена информацией,* что выразилось в трактовке посимвольных устройств как файлов;

4) *взаимодействие* компонентов DOS между собой и с другими программами *через механизм прерываний*;

5) *простоту* как самой системы, так и ее интерфейсов.

При совместной записи маршрута и составного имени файла первый элемент, если он не состоит из единственного корневого каталога, отделяется от второго элемента символом \. Пример: \IVANOV\PROGS\prog2.pas.

Спецификации файла и каталога

В предыдущем материале мы рассмотрели все элементы, которые необходимо указать, чтобы файловая система нашла существующий файл или создала новый файл в нужном месте файловой структуры. Тем не менее очевидна необходимость перечисления всех правил, касающихся указания файла, в одном месте, что мы и сделаем в этом подпункте.

Для обеспечения доступа к существующему файлу или определения места в файловой структуре, где нужно разместить новый файл, в общем случае требуется задать:

1) *имя привода,* на котором установлен диск, содержащий искомый файл или предназначенный для размещения нового файла;

2) *маршрут* к файлу по файловой структуре этого диска;

3) *составное имя файла (имя файла* и *расширение имени файла).*

Данные сведения указываются в **спецификации файла**, которая имеет следующий *синтаксис (представление, форму, структуру)*:

[*привод*:][*маршрут*\]*имя_файла*.[*расширение*]

Здесь необязательные элементы заключены в квадратные скобки. В случае, когда те или иные элементы отсутствуют, они восстанавливаются по нижеприведенным правилам:

1) если привод не задан, то выбирается *текущий привод*;

2) если маршрут начинается с символа \ (указан полный маршрут), то поиск каталога, где должен содержаться файл, осуществляется, начиная с *корневого каталога* диска на выбранном дисководе;

3) если условие в п. 2 не выполняется, то поиск каталога, где должен содержаться файл, осуществляется, начиная с *текущего каталога* диска на выбранном дисководе;

4) если маршрут не задан, то считается, что файл содержится в *текущем каталоге* диска на выбранном дисководе;

5) если расширение не задано, то считается, что его нет.

В случае, когда не заданы расширение и предшествующая ему точка, то оно либо отсутствует, либо принимается по умолчанию в зависимости от контекста, в котором спецификация файла указана.

На длину спецификации файла накладывается ограничение: она, не включая имя привода и завершающее его двоеточие, не должна превышать 63 символов (вместе — 65 символов).

Примеры:

- спецификация C:\IVANOV\PROGS\prog3.pas полностью определяет местонахождение файла в файловой структуре диска, установленного в привод C;
- спецификация \prog.exe задает файл prog.exe в корневом каталоге текущего диска;
- спецификация prog.exe задает файл в рабочем каталоге;
- спецификация C:DIR1\prog.exe задает файл в дочернем каталоге DIR1 текущего каталога диска в приводе C;
- спецификация C:prog.exe задает файл в текущем каталоге диска в приводе C.

Введем понятие **спецификации шаблона файла** как спецификации файла, в которой вместо составного имени файла используется его шаблон. Она определяет совокупность файлов в выделенном каталоге диска на заданном приводе, составные имена которых сопоставляются с шаблоном.

Например (см. рис. 5.4), спецификация шаблона файла \IVANOV\PROGS\p*.* задает три файла prog1.pas, prog2.pas и prog3.pas, т.е. в данном случае — полностью содержимое каталога PROGS.

Можно утверждать, что спецификация файла является *частным случаем* спецификации шаблона файла.

Задать поиск файлов по шаблону оказывается возможным только в определенном каталоге, а не во всей файловой структуре диска.

Нам понадобится в дальнейшем понятие **спецификации каталога,** которая имеет следующий синтаксис:

[*привод*:][*маршрут*]

Она однозначно определяет каталог, указанный последним в маршруте (или текущий каталог) и полностью согласуется с правилами умолчания для спецификаций файлов, описанными выше. Отличие состоит только в том, что не задается файл и предшествующий его имени разделитель \.

Если в спецификации (любой) указаны все допустимые для нее элементы и полный маршрут, то будем называть ее **полной**, в противном случае — **неполной**.

Выполнение запросов к файловой системе

Чтобы воспользоваться услугами файловой системы по выполнению операций над элементами файловой структуры (файлами и каталогами), в программе следует предусмотреть загрузку необходимых для этого аргументов в регистры МП и выдачу соответствующего прерывания верхнего уровня (как правило, вызов функции DOS).

Получив таким образом управление, файловая система идентифицирует запрос на выполнение операции, выбирает требуемый для этого драйвер и транслирует запрос, возможно, в последовательность обращений к драйверу.

Драйвер обеспечивает непосредственное управление адаптером устройства по выполнению требуемой операции через ПВВ, выдавая последовательность команд, а также принимая и анализируя результаты их выполнения. После завершения операции драйвер возвращает управление и результаты своей работы ядру файловой системы, которое, возможно, после предварительной их обработки передает результаты программе, выдавшей запрос на операцию, через регистры МП и/или область ОЗУ. Затем выполнение программы продолжается.

Отметим, что файловая система DOS реализует операции ввода-вывода только *синхронно*, т.е. выполнение программы приостанавливается до завершения операции.

5.2.3. Структура системного диска

Теперь читатель готов к уяснению логической структуры *системного диска* (диска, с которого загружается DOS). Она представлена на рис. 5.5. Начало логического дискового пространства показано в верхней части этого рисунка, а конец — в нижней.

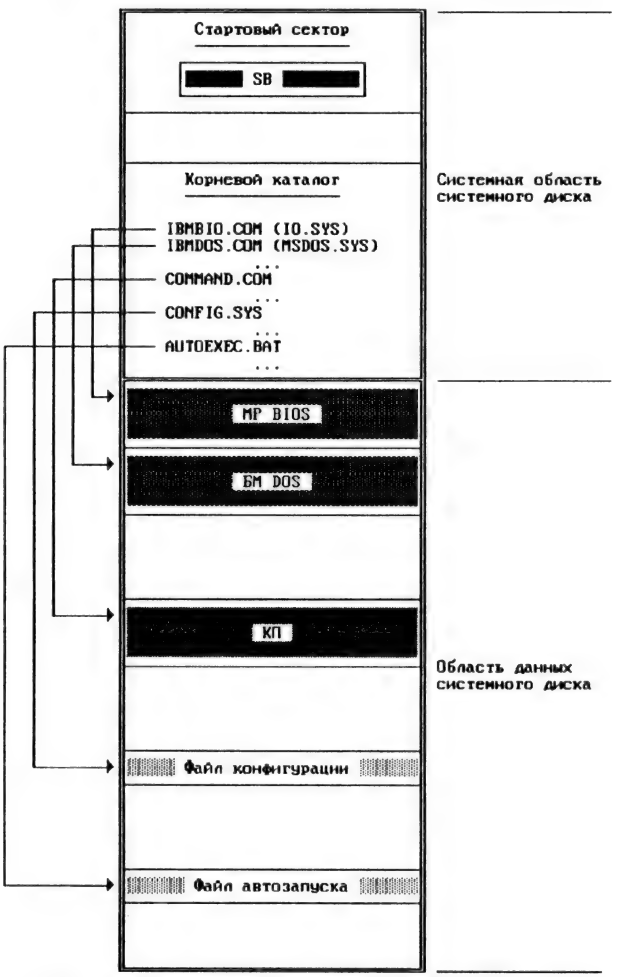


Рис. 5.5. Логическая структура системного диска

Компоненты DOS хранятся на системном диске (за исключением SB) в файлах с зарезервированными составными именами. На рисунке представлены только те файлы, на размещение которых имеются достаточно жесткие ограничения (в последних версиях DOS они существенно ослаблены, о чем мы скажем особо). Тем не менее не все из этих файлов являются принципиально необходимыми для функционирования DOS.

SB содержится в **стартовом секторе** системного диска, т.е. в самом первом его секторе.

MP BIOS хранится в файле IBMBIO.COM (PC DOS) или IO.SYS (MS-DOS), который должен размещаться на системном диске первым и первым же входить в корневой каталог.

БМ DOS хранится в файле IBMDOS.COM (PC DOS) или MSDOS.SYS (MS-DOS), который должен быть размещен сразу вслед за файлом с MP BIOS и входить в корневой каталог вторым.

Фрагментация двух упомянутых файлов раньше не допускалась. Множество ограничений на их размещение связано с необходимостью упрощения SB, который должен поместиться в 512-Кбайт сектор. Файлы с MP BIOS и БМ DOS имеют атрибуты R, H и S. В DOS 3.0 ослаблены требования по размещению файла с БМ DOS, а в версии 3.3 — и по размещению файла с MP BIOS. Таким образом, теперь два упомянутых системных файла могут находиться на системном диске где угодно и быть фрагментированными, но все же должны регистрироваться на первых двух позициях корневого каталога. Тем не менее полезно соблюсти все перечисленные выше ограничения, чтобы загрузка DOS выполнялась быстрее.

КП находится в файле COMMAND.COM, который должен содержаться в корневом каталоге системного диска на любой позиции и может занимать любую область логического дискового пространства. Если предпринять специальные меры (посредством команды конфигурирования системы SHELL=), то файл с КП можно «спрятать» в любом каталоге, в том числе и на другом диске.

Все перечисленные выше компоненты являются обязательными (без них загрузка невозможна).

Файлы конфигурации системы и *автозапуска* имеют имена CONFIG.SYS и AUTOEXEC.BAT соответственно и могут быть размещены только в корневом каталоге системного диска, но без каких-либо дополнительных ограничений.

Для хранения трех системных файлов (IBMBIO.COM, IBMDOS.COM и COMMAND.COM) DOS 3.3 требуется 77566 байт дисковой памяти. Суммарный размер этих файлов в других версиях DOS отличается от указанного незначительно (в DOS 4.0 он несколько больше, а в DOS 5.0 — несколько меньше).

Рассмотренную структуру системного диска формируют соответствующие команды DOS, которые будут описаны ниже.

5.2.4. Загрузка DOS

Вообще под **загрузкой программы** понимается размещение ее в ОЗУ для исполнения. Функцию загрузки выполняет специальная программа, называемая **загрузчиком**. В случае загрузки ОС дело обстоит несколько иначе, и вот почему: ОС является первичным программным продуктом в том смысле, что ее загрузкой и запуском на выполнение никакая другая программа не управляет. Поэтому ОС должна *сама себя загружать*, возможно, после небольшого «толчка» извне, а процедура загрузки должна включать *запуск ОС в работу*, т.е. ее активизацию для управления ресурсами ЭВМ.

Последовательность основных этапов загрузки DOS в случае, когда не возникает никаких нештатных ситуаций, представлена на рис. 5.6.

Опишем эту процедуру более подробно и обратим внимание читателя на ситуации, при возникновении которых нормальный ход загрузки может быть нарушен, а от пользователя требуются определенные действия. Загрузка DOS начинается автоматически после *включения питания* ПЭВМ. Включать машину нужно в последовательности, указанной в документации на нее. При этом обычно действует следующее правило: сначала включаются требуемые для данного сеанса работы ПУ в любом порядке и только после этого — системный блок ПЭВМ.

Включение питания системного блока приводит к аппаратной передаче управления на программу *тестирования* (проверки работоспособности) оборудования, находящуюся в BIOS. Тестированию подлежат все устройства ПЭВМ, на которые подано электропитание. Оно сопровождается миганием индикаторов на ПУ и подачей звуковых сигналов. На экране дисплея наблюдается возрастающая последовательность чисел, быстро сменяющих друг друга. Таким образом отражаются результаты тестирования ОЗУ. Если ОЗУ функционирует нормально, то после каждого появившегося на экране числа, определяющего объем проверенной области ОЗУ в Кбайт, выводится соответствующее сообщение (ОК — сокращение от английского O'Key, Passed и т.п.).

Когда в работе оборудования ПЭВМ выявлены нарушения, на экран дисплея выдается идентифицирующее ошибку сообщение, включающее *код ошибки*. Если ошибка *некритическая* (причиной может быть сбой, а не отказ оборудования), то пользователю предоставляется возможность возобновить процесс загрузки, начиная с тестирования, нажав клавишу F1 на клавиатуре, о чем информирует сообщение на экране дисплея. В случае *критической* неисправности (отказ, не дающий возможности продолжить работу) процесс загрузки прекращается с выдачей на экран соответствующего сообщения. В любом из двух случаев о возникшей ситуации и коде ошибки следует сообщить лицу, ответственному за техническое обслуживание ПЭВМ.

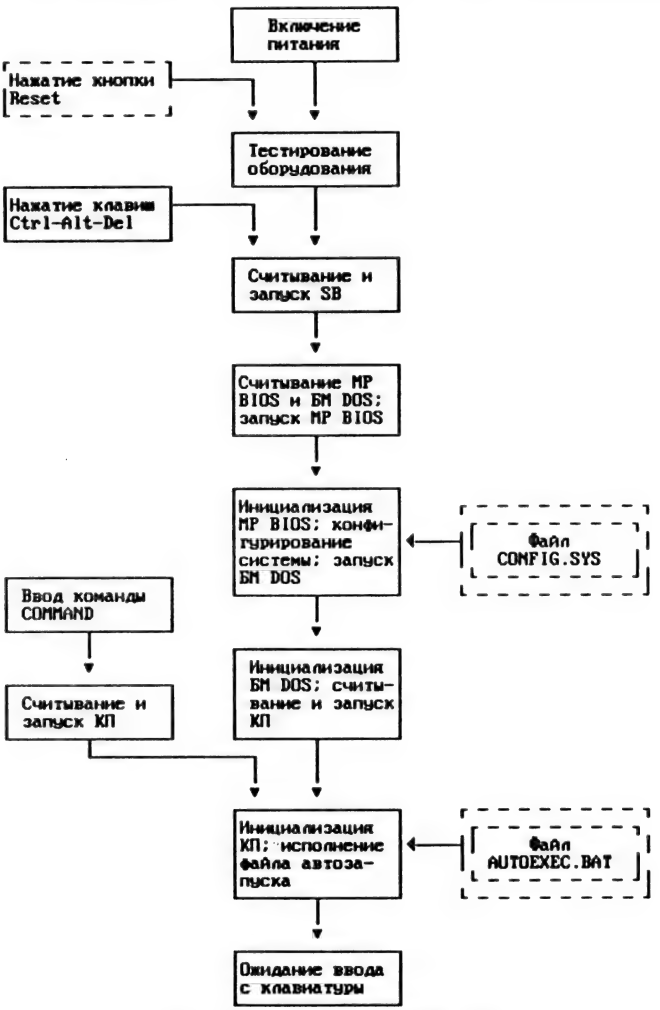


Рис. 5.6. Процесс загрузки DOS

После включения питания ПЭВМ, но до окончания тестирования оборудования, пользователь, желающий осуществить загрузку DOS с дискеты или не имеющий другой возможности, должен установить системную дискету в привод с именем A. Загрузка DOS с ГД осуществляется в следующих случаях:

— когда ПЭВМ не укомплектована НЖМД;

— когда НЖМД имеется, но требуется разместить на жестком диске новую версию DOS, восстановить разрушенную по какой-либо причине систему или удалить компьютерные вирусы.

Дискета устанавливается в НГМД обычно горизонтально таким образом, чтобы прорезь разрешения записи и отверстие в пластиковом конверте дискеты, индицирующее начало дорожки, были слева, а прорези для подвода магнитных головок к дискете — впереди. Помещенная в привод дискета закрывается специальной защелкой (шторкой, дверцей). Устанавливать дискеты в отключенную от сети ПЭВМ и оставлять их в дисководах после выключения питания не рекомендуется, так как при этом возможна порча магнитного покрытия.

После успешного завершения тестирования оборудования ПЭВМ инициализируются векторы прерываний нижнего уровня таким образом, что при их возникновении будут выбираться обработчики из BIOS, вслед за чем управление передается на программу начальной загрузки в BIOS. Программа начальной загрузки обращается к дисководу A и, если в него установлена дискета, *считывает в ОЗУ SB,* хранящийся в ее стартовом секторе (заметим, что SB имеется в стартовом секторе каждого диска, отформатированного средствами DOS, с тем, чтобы при необходимости выдать сообщение об отсутствии системы на диске, как описано ниже). Если же дискета в

приводе A отсутствует, то осуществляется попытка загрузить SB с системного логического диска в активном разделе жесткого диска (конечно, при наличии последнего). Детали доступа к SB на винчестере мы пока рассматривать не будем. В случае, когда загрузить DOS и с жесткого диска не удается, запускается интерпретатор Basic'a, прошитый в ПЗУ.

Предположим, что SB считан в ОЗУ. Дальнейшая загрузка DOS продолжается с того диска, из стартового сектора которого SB прочитан (дисковод A для гибкого диска и дисковод C — для жесткого).

После занесения SB в ОЗУ программа начальной загрузки *передает на него управление,* прекращая свою работу.

SB проверяет наличие на диске файлов с MP BIOS и БМ DOS. Если они находятся на своем месте и правильно помещены в корневой каталог системного диска (см. п. 5.2.3), то *эти модули загружаются в ОЗУ* и *управление получает первый из них.* В противном случае SB выдает на экран следующее сообщение об ошибке (в скобках под ним приведен перевод на русский язык):

Non-system disk or disk error
Replace and strike any key when ready
(Несистемный диск или ошибка на диске.
Замените диск и затем нажмите любую клавишу)

Появление этого сообщения свидетельствует о разрушении информации на диске или случайной попытке загрузки с действительно несистемного диска. При получении такого сообщения для помещенной в привод A дискеты пользователю следует:

— установить в привод A действительно системную дискету, если требовалось загрузить DOS с гибкого диска;

— открыть защелку НГМД, если требовалось загрузить DOS с жесткого диска.

После этого достаточно нажать на клавиатуре любую алфавитно-цифровую клавишу, клавишу пробела или клавишу Enter (Ввод) для продолжения процесса загрузки DOS.

Если же сообщение о несистемном диске появилось для винчестера, то пользователю ничего не остается, как перезагрузить DOS с дискеты и выяснить причину отказа жесткого диска.

Получив управление, MP BIOS выполняет нижеприведенные действия:

1) определяет состояние оборудования и *инициализирует* (устанавливает в исходное состояние) *включенные ПУ*;

2) обрабатывает файл конфигурации CONFIG.SYS (если он, конечно, имеется) и осуществляет *конфигурирование DOS,* загружая в ОЗУ и подключая к системе указанные внешние драйверы, а также устанавливая параметры системы; если файл CONFIG.SYS отсутствует, то никакие внешние драйверы не подключаются, а параметры DOS устанавливаются по умолчанию;

3) *инициализирует* (устанавливает) и *переустанавливает некоторые векторы прерываний нижнего уровня*;

4) *передает управление на БМ DOS.*

БМ DOS продолжает загрузку системы, реализуя следующие функции:

1) *инициализирует свои внутренние таблицы*;

2) *инициализирует векторы* обрабатываемых им *прерываний верхнего уровня*;

3) *загружает в ОЗУ КП и передает на него управление.*

Получив управление, КП:

1) *инициализирует три вектора прерываний,* которые он обрабатывает;

2) *считывает, обрабатывает и организует выполнение файла автозапуска* AUTOEXEC.BAT;

если этот файл отсутствует, то КП последовательно выдает запросы на установку даты и текущего времени, на которые допускается ответить просто нажатиями клавиши Enter.

С помощью файла AUTOEXEC.BAT можно автоматически выполнять команды DOS и программы для создания необходимой Вам операционной среды.

Загрузка системы завершается выдачей на экран дисплея **приглашения (подсказки) DOS** в виде

A>_ или C>_

Подчеркиванием здесь обозначен курсор.

Первое приглашение появляется в случае загрузки с дискеты, а второе — при загрузке с жесткого диска. Буква в приглашении информирует пользователя об имени текущего дисковода. Приглашение DOS может иметь и другой вид (если предпринимаются определенные действия в файле AUTOEXEC.BAT), а может и вообще не появиться (если из файла AUTOEXEC.BAT запускается интерактивный программный продукт). В последнем случае Вы сразу попадете в его среду.

В ответ на приглашение DOS можно вводить с клавиатуры команды и запускать на выполнение различные программы.

Сделаем два замечания:

1) последовательность выдаваемых на экран дисплея сообщений во время загрузки DOS зависит от ее версии, фирмы-разработчика, а также от содержимого файлов CONFIG.SYS и AUTOEXEC.BAT;

2) во время обработки файла AUTOEXEC.BAT модулем инициализации КП и выполнения этого файла возможно переназначение имен поблочным логическим устройствам вследствие того, что резидентные программы-драйверы, указанные в AUTOEXEC.BAT, могут обладать жесткой привязкой к именам, уже выделенным другим логическим устройствам. Переназначение имен устраняет их коллизии.

Перезагрузку (повторную загрузку) DOS, например, в случае аппаратного или программного сбоя, можно осуществить одним из следующих способов:

1) путем выключения и последующего включения через некоторое время системного блока ПЭВМ (так называемый *холодный перезапуск*);

2) путем нажатия кнопки Reset на корпусе ПЭВМ, если она имеется;

3) путем одновременного нажатия клавиш Ctrl (УПР), Alt (ДОП) и Del (УДЛ) на клавиатуре ПЭВМ, что обозначается как Ctrl-Alt-Del (так называемый *горячий*, или *теплый перезапуск*);

4) путем ввода команды COMMAND с клавиатуры ПЭВМ.

Первый способ используется для полной перезагрузки DOS, начиная с тестирования оборудования. Отключение и включение питания плохо сказывается на работоспособности аппаратуры. Поэтому ряд моделей ПЭВМ имеют специальную кнопку Reset, при нажатии которой (второй способ) осуществляется аппаратная передача управления на программу тестирования оборудования без отключения питания. Если необходимость в тестировании ПЭВМ отсутствует, то ускорить процесс перезагрузки можно, используя третий способ. Четвертый способ наиболее быстрый, но путем ввода команды COMMAND осуществляется только *частичная перезагрузка DOS*, а именно, считывание в ОЗУ и запуск КП.

Распределение основной памяти ПЭВМ после загрузки DOS представлено на рис. 5.7. Для размещения резидентной части DOS требуется относительно небольшая область ОЗУ: например, DOS 3.3 занимает около 58 Кбайт ОЗУ в «усредненной» конфигурации, а DOS 4.0 и DOS 5.0 (без задействования верхней памяти) — 75 и 65 Кбайт соответственно.

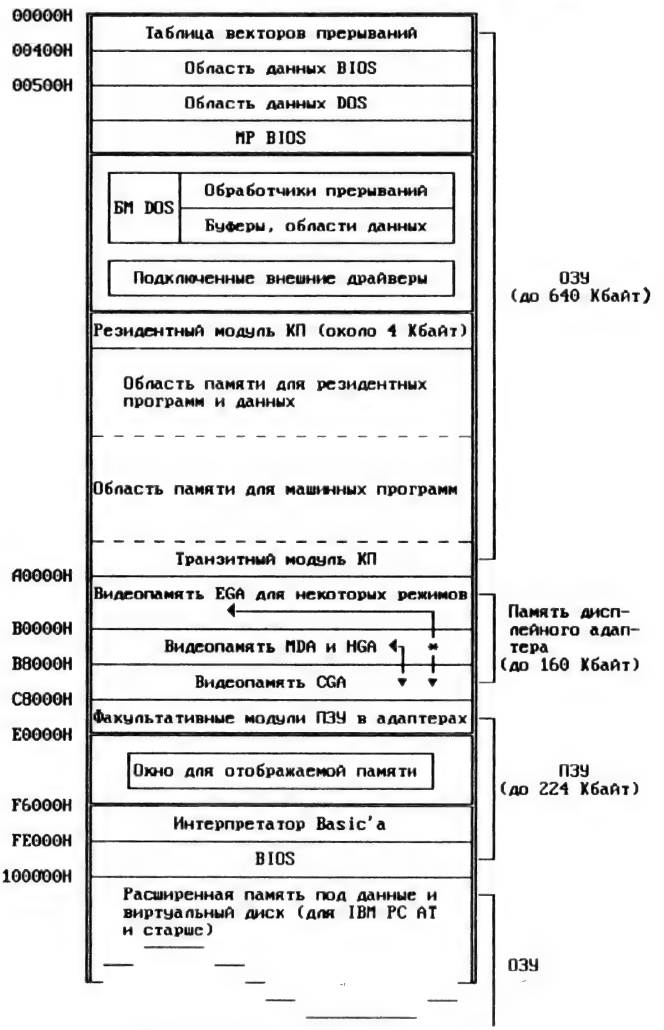


Рис. 5.7. Карта основной памяти

В DOS нет специальной команды выхода (окончания работы). Для *завершения работы* необходимо:

1) убедиться в том, что последняя введенная команда или запущенная программа завершила свою работу, о чем свидетельствует приглашение DOS в последней выведенной на экран строке;

2) извлечь дискеты из приводов и поместить их в защитные конверты;

3) выключить питание ПЭВМ.

При наличии в ПЭВМ жесткого диска непосредственно перед выключением питания целесообразно осуществить парковку его головок, что выполняется специальной утилитой.

Выключение ПЭВМ производится в соответствии с инструкцией по эксплуатации. Обычно оно осуществляется в обратном по отношению к включению питания порядке: сначала отключается системный блок, а затем ПУ.

На наш взгляд, здесь уместно также привести правила обращения с ГД, чтобы обеспечить возможность продолжительной работы с ним:

1) не разрешается сгибать дискету;

2) нельзя прикасаться руками к магнитному покрытию диска;

3) недопустимо подвергать дискету воздействию магнитных полей;

4) нужно хранить дискету в бумажном конверте при положительной температуре окружающего воздуха;

6) надписи на приклеенной к дискете этикетке следует делать без нажима, мягким карандашом или перьевой ручкой;

7) брать дискету рукой можно только за один из углов пластмассового конверта;

8) нельзя мыть дискету.

5.2.5. Выполнение команд и программ

Если Вы садитесь за ПЭВМ для производства определенных работ, то Вам потребуется выполнять команды DOS и программы под управлением системы. Запросы на их выполнение вводятся с клавиатуры в ответ на приглашение DOS и называются **командными строками**. Длина командной строки не должна превышать 127 символов. В дальнейшем мы увидим, что последовательность командных строк может быть размещена в командном файле и выполнена из него. Командная строка состоит из *имени команды* или *спецификации исполняемого файла* и, при необходимости, *аргументов,* разделенных по крайней мере одним пробелом. Возможно также задание *переключателей*, начинающихся символом /. Командная строка набирается на клавиатуре, как на обычной пишущей машинке, и отображается на экране дисплея. Сигнал от каждой нажатой пользователем клавиши обрабатывается DOS, но КП никаких действий не предпринимает до обнаружения окончания командной строки. Сигналом об окончании командной строки служит нажатие клавиши Enter — только тогда командная строка считается *введенной.* До этого момента командную строку можно *редактировать* и *аннулировать.*

В общем случае под **вводом** символа или последовательности символов с клавиатуры будем понимать их печатание и последующее нажатие клавиши Enter.

Пример командной строки: DIR A:, что означает запрос на выдачу содержимого текущего каталога диска в приводе A.

Введенная командная строка анализируется КП, и дальнейшие его действия зависят от того, что требуется выполнить: внутреннюю команду, либо внешнюю команду или программу, либо командный файл.

Если в командной строке задана *внутренняя команда,* то она просто выполняется транзитным модулем КП и на экране дисплея вновь появляется приглашение DOS, возможно, после различных информационных сообщений и запросов к пользователю, требующих ответа.

Если командная строка не распознана как запрос на выполнение внутренней команды (а это означает, что она содержит спецификацию исполняемого файла, имя внешней команды, которое также является по сути спецификацией программного файла, или ошибочна), то транзитный модуль КП осуществляет попытку найти в файловой структуре заданного диска указанный исполняемый файл. Когда эта попытка не приводит к успеху (ошибочно задана спецификация файла или в выделенном каталоге файл отсутствует), на экран дисплея выдается соответствующее сообщение и КП переходит в состояние ожидания ввода командной строки с клавиатуры. Приведем пример диалога с КП в этом случае:

```
C>ERUNDA ◄Enter►
Bad command or file name
(Ошибочная команда или ошибочное имя файла)
C>
```

В скобки ◄ и ► заключено наименование клавиши, которую следует нажать.

Если в спецификации подлежащего выполнению файла расширение в командной строке не задано, а в выделенном каталоге содержится несколько исполняемых файлов с указанными именами, но с различными расширениями, то возникший конфликт разрешается на основании *приоритетов расширений.* В DOS принято, что COM-файлы пользуются преимуществом перед другими (EXE- и BAT-файлами), а EXE-файлы — только перед BAT-файлами. Поэтому выбор

файла для использования осуществляется однозначно в любом случае. При явном указании расширения коллизии не возникает.

Предположим, что по спецификации файла найден и выделен для выполнения *COM-* или *EXE-файл* (обработку BAT-файлов рассмотрим позже). В этом случае управление получает загрузчик программ, содержащийся в транзитном модуле КП. Он выполняет следующие действия:

1) создает дубликат (копию) *окружения DOS* для использования в дальнейшем программой, подлежащей выполнению (см. ниже);

2) помещает полную спецификацию исполняемого файла (даже если его поиск проводился в разных каталогах) за дубликатом окружения DOS;

3) выделяет в ОЗУ область памяти для подлежащей выполнению программы, еще не занятую резидентной частью DOS и резидентными программами (отводится вся свободная память);

4) в начале этой области (с минимальными адресами) резервирует память под *префикс программного сегмента* (PSP — Program Segment Prefix), который служит для хранения важной для выполняемой программы информации;

5) заполняет поля PSP следующими сведениями:

— объемом памяти, доступным для программы;

— адресом дубликата окружения DOS;

— аргументами из командной строки в символьном виде, которые благодаря этому становятся доступными программе;

— текущими векторами прерываний 22H — 24H для их восстановления при завершении выполнения программы и возврата в DOS;

6) загружает программу в ОЗУ за PSP.

Затем, если в память переписана EXE-программа, то загрузчик по информации из ее заголовка осуществляет настройку этой программы в соответствии с точкой загрузки, т.е. *перемещение программы.*

Для COM-программы никакой настройки не требуется. Любая программа длиной менее 64 Кбайт и не содержащая никаких других сегментов, кроме программного, может быть таковой. Однако COM-программа способна иметь и бóльшую длину, если в ней не используются явные ссылки к другим сегментам. При этом допустимы относительные ссылки с использованием точки загрузки как базового адреса. COM-программы загружаются быстрее и требуют меньшего объема внешней памяти для своего хранения.

После загрузки программы и возможного ее перемещения загрузчик передает управление (посредством дальнего JMP'а) на начало программы и она начинает выполняться.

КП имеет специальную область, называемую **окружением DOS** (точнее — *корневым окружением,* так как при порождении процесса оно дублируется и, возможно, в измененном виде передается новому процессу; при его окончании дубликат окружения уничтожается, а оригинал окружения активизируется). Окружение хранит имена **глобальных переменных** и их значения в символьном виде, которые могут использоваться для получения определенной информации любыми выполняемыми под управлением DOS программами. Например, по значению определенной глобальной переменной выполняемая программа может установить, в каком каталоге содержатся необходимые для ее работы файлы, либо как «вести себя» в том или ином случае. Пользователь может, используя средства командного языка, прочитать окружение, добавить в него новую запись вида *«имя глобальной переменной = значение»*, обновить или удалить любую такую запись. Эти операции доступны выполняющимся программам. Как уже отмечалось, дочерняя программа получает от родительской программы статическую копию ее окружения, снабженную полной спецификацией файла, содержащего дочернюю программу. Этот факт используют «умные» программы для вычленения имени привода и выделенного маршрута при своем запуске для доступа к другим файлам, задействуемым этими программами и находящимся в одних с ними каталогах или каким-либо образом связанных с последними. Даже если исполняемый файл при запуске программы ищется в нескольких каталогах по маршрутам, определенным в системе, то все равно к дубликату окружения дописывается имя того привода и маршрут, где файл найден, а также его составное имя. Любые изменения, сделанные в окружении выполняемой программой, отменяются при ее завершении (так как любая программа работает с дубликатом окружения).

Для обмена информацией между ОЗУ и ПУ выполняемая программа может использовать средства файловой системы DOS (см. п. 5.2.2) или непосредственно обращаться к ПВВ. При возникновении какой-либо ошибки в ПУ система пытается повторить операцию обмена 3 — 5 раз. Когда ошибку устранить не удалось, возникает прерывание 24H, и DOS выдает на экран идентифицирующее ошибку сообщение, предлагает пользователю решить, что делать в этом случае, и ждет от него ответа. Например, если программа обращается к дисководу A для чтения информации, а он не готов к обмену (открыта защелка), то появятся следующие сообщения:

```
Not ready error reading drive a:
(Ошибка чтения из-за неготовности привода A:)
Abort, Retry, Fail ? _
(Прервать, Повторить, Пропустить?)
```

На запрос пользователь может дать один из трех вариантов ответа, нажав клавишу A (для Abort), R (для Retry) или F (для Fail). В зависимости от варианта ответа, DOS реагирует на возникшую ошибку по-разному:

A — *аномально завершает выполнение программы,* выдавшей запрос на ввод-вывод;

R — *повторяет операцию ввода-вывода*;

F — *завершает выполнение операции* и *возвращает программе* соответствующий *код ошибки*; программа при этом продолжает выполняться.

Использовать ответ A нужно с осторожностью: Вы можете испортить большую часть выполненной Вами работы по диалогу с интерактивной программой. Ответ R имеет смысл вводить, когда Вы устранили причину ошибки. Так, для рассмотренного примера достаточно запереть дисковод A: и нажать клавишу R. Ответ F может использоваться программой для анализа кода ошибки и выполнения определенных действий по ее нейтрализации, например, путем организации ввода с другого дисковода, о чем пользователь должен быть извещен, чтобы переставить диск в новый привод (такая стратегия удобна, когда задействуемый накопитель неисправен).

Часто DOS выдает расширенный запрос

Abort, Ignore, Retry, Fail ?_ ,

где ответ I (для Ignore — *проигнорировать*) означает продолжение операции ввода-вывода, как будто бы ничего не случилось. Такой ответ может привести к потере данных.

В зависимости от версии DOS пользователю может быть выдан и другой вариант запроса, но всегда он является подмножеством только что приведенного запроса.

В программе можно предусмотреть свою реакцию на критическую ошибку, подменив вектор прерывания 24H.

Полный перечень сообщений MS-DOS 4.0 (как чисто информационных, так и требующих обязательного ответа) и их интерпретация содержатся в Приложении 1.

Любая программа в ходе своего выполнения может *породить дочерний процесс,* код которого находится в EXE- или COM-файле. Для этого она должна вызвать функцию 4BH (EXEC) по прерыванию 21H. В качестве аргументов порожденному процессу передается командная строка его запуска, текущее или модифицированное текущее окружение DOS (дубликат!), а также другая информация. Можно таким образом инициировать и вторичную копию КП для выполнения внутренней команды DOS. Порожденный процесс запускается и выполняется, как описано выше, а породившая его программа ждет его окончания. После завершения процесса родительская программа возобновляет активность.

Используя ту же функцию DOS 4BH, можно не только породить процесс, но и просто *загрузить программу в ОЗУ,* не передавая ей управления. Это полезно при разработке *оверлейных* (загружающихся по частям с целью экономии памяти) программ. Больше никакой поддержки оверлейных программ DOS не обеспечивает. Все заботы по управлению оверлеями ложатся на плечи программиста или перекладываются на систему программирования.

Любая программа возвращает управление родительской программе по функции 4CH прерывания 21H. Если родительской программой является КП, то дополнительно инициируется прерывание 22H, при обработке которого выполняются следующие действия:

1) аварийно закрываются все незакрытые в программе файлы, причем информация о файлах в каталогах не обновляется, так что созданные и обновленные программой файлы будут закрыты некорректно, а это приведет к потере информации и образованию на диске «мусора»; поэтому файлы всегда необходимо закрывать в программе явно;

2) освобождается вся память, использованная программой, в том числе и динамически;

3) из соответствующих полей PSP восстанавливаются векторы 22H — 24H, которые могли быть подменены программой.

Пользовательская программа не должна выдавать прерывание 22H (это прерогатива системы). Вместе с тем выполняемая программа может подменить вектор прерывания 22H, тем самым задав другую реакцию на завершение программы, в том числе после возникновения ошибки.

Каждая выполненная программа может возвратить родительской программе (в частности, DOS) **код возврата**, который может использоваться в дальнейшем для анализа результата работы завершившейся программы.

Преждевременно завершить выполнение программы пользователь может путем нажатия клавиш Ctrl-Break (УПР-ФСД СТОП). Однако сказанное справедливо только, если программа не подменила вектор прерывания 23H. В противном случае реакция на нажатие этих клавиш будет определяться новым обработчиком данного прерывания. Сама возможность реакции на прерывание 23H определяется в файле CONFIG.SYS.

Если обработка прерывания по Ctrl-Break разрешена, то оно действует всегда, кроме тех моментов, когда реализуются функции DOS 06H и 07H (по прерыванию 21H). В противном случае (если обработка Ctrl-Break запрещена) Ctrl-Break действует только во время обмена информацией с посимвольными устройствами.

Важно отметить, что:

1) нажатие Ctrl-Break приводит к установке по прерыванию 1BH соответствующего флага в компьютере, но немедленная реакция на это отсутствует;

2) состояние указанного флага может проверяться (если обработка Ctrl-Break разрешена) только при вызове функции DOS;

3) прерывание 23H возникает только в указанной в п. 2 ситуации.

Аналогом Ctrl-Break в DOS является комбинация клавиш Ctrl-C. Однако Ctrl-Break сильнее Ctrl-C, так как приводит практически к немедленному возникновению аппаратного прерывания 1BH для установки флага. В отличие от этого код Ctrl-C вводится обычными средствами через буфер клавиатуры и обрабатывается DOS с тем, чтобы проимитировать нажатие Ctrl-Break. Если же чтение из буфера клавиатуры системой в данной ситуации не осуществляется, то нажатие Ctrl-C останется незамеченным и поэтому ни к каким изменениям в работе программы не приведет.

После окончания выполнения программы она может оставить себя **резидентной** в ОЗУ, вызвав функцию 31H по прерыванию 21H. При этом управление возвращается родительскому процессу (в частности DOS), но память, занятая программой, не освобождается. Вызываться на выполнение такая программа может впоследствии при возникновении прерываний, векторы которых она подменила. Такая техника используется в резидентных сервисных системах, а также при разработке программ, выполняющих роль драйверов устройств. Остающиеся резидентными программы обозначаются через TSR (Terminate but Stay Resident — завершиться, но остаться резидентной).

После окончания выполнения программы любым из перечисленных способов (если она была запущена КП, а не другой программой) на экране дисплея появляется подсказка DOS. Это и является внешним признаком завершения выполнения всех программ и ожидания DOS новой работы.

Выполняемая программа, особенно большого размера, может перекрывать в ОЗУ транзитный модуль КП. Резидентный модуль последнего, вслед за выходом из выполненной программы, распознает эту ситуацию путем подсчета контрольной суммы содержимого области памяти, в которой должен находиться транзитный модуль КП. Если результат не совпадает с эталоном, то резидентный модуль КП осуществляет загрузку с системного диска своей транзитной части.

Такая схема приводит к трудностям работы на ПЭВМ, не оборудованной жестким диском. Если пользователь во время выполнения какой-либо программы заменил системный диск на другой, требуемый для этой программы, то после завершения ее выполнения он скорее всего получит на экране сообщение

```
Insert disk with COMMAND.COM in drive a:
Press any key to continue ...
(Установите диск с COMMAND.COM'ом в привод A.
Нажмите любую клавишу для продолжения)
```

До тех пор, пока пользователь не выполнит это предписание, работа на ПЭВМ не может быть продолжена. Облегчить работу путем исключения необходимости установки системного диска в привод A после выполнения программ можно одним из двух способов:

1) в корневой каталог каждой используемой дискеты записать файл COMMAND.COM;

2) при загрузке DOS создать виртуальный диск, скопировать на него файл COMMAND.COM и определить полную спецификацию копии этого файла в качестве значения глобальной переменной COMSPEC в окружении DOS (по умолчанию это значение указывает на системный диск).

Применение первого способа ведет к неэкономному использованию дисковой памяти, а второго — ОЗУ.

В случае, когда по спецификации файла в командной строке выбран для выполнения *командный файл*, транзитный модуль КП считывает из него первую строку, анализирует ее и организует выполнение указанной в ней команды или программы, как описано выше. Затем считывается вторая строка командного файла, и процесс его интерпретации повторяется. Каждая строка командного файла содержит командную строку или команду, допустимую только в файлах этого типа. Командные файлы выполняются медленно, так как за каждой их строкой производится обращение к диску, а затем, скорее всего, запуск другой программы. Достоинством же построчного чтения является возможность создания *самомодифицирующихся* командных файлов.

5.2.6. Управление памятью

В среде DOS вся оперативная память ПЭВМ логически делится на *стандартную* и *дополнительную*.

Стандартной (обычной) памятью считается память в диапазоне адресов от 0 до 640K — 1 в *стандартном* 1-Мбайт *адресном пространстве*. Очевидно, ее размер не может превышать 640 Кбайт, что для ряда приложений явно недостаточно.

Дополнительной является вся оперативная память, не удовлетворяющая указанному для стандартной памяти ограничению.

Данный пункт мы посвятим рассмотрению особенностей управления этими двумя разновидностями памяти.

Стандартная память

Стандартная память может использоваться DOS для хранения как выполняемых программ, так и данных любой природы без каких-либо ограничений. Вы уже знаете, в частности, что сама DOS размещается тоже в этой памяти.

Стандартная память распределяется *блоками*.

Блоком памяти называется ее *непрерывный фрагмент*, выделяемый для хранения загружаемой программы или данных во время выполнения программы. Первые 16 байт каждого блока памяти отводятся под **блок управления памятью** (Memory Control Block — MCB). MCB описывает размер блока памяти и его владельца, а также содержит ссылку на начало следующего блока памяти. Таким образом, все MCB, а следовательно, и блоки памяти, связаны в цепочку, что обеспечивает возможность управления памятью (в частности, ее выделения и освобождения).

Когда DOS запускает программу, она сначала выделяет блок памяти для копии своего окружения, а затем — весь остаток памяти для самой программы, о чем мы уже вскользь упоминали. Эти блоки связываются в цепочку так, как показано на рис. 5.8, и не обязательно должны быть смежными. Блоки памяти всегда выделяются в начале свободной области ОЗУ, поэтому, если память не фрагментирована, блок с окружением и блок с кодом программы будут соприкасаться. Вспомним, что вместе с окружением выполняемой программе передается полная спецификация программного файла, которая подстыковывается в конец окружения. Поэтому размер копии окружения не совпадает с размером оригинала. Программа в ходе выполнения может уменьшить размер выделенного ей блока, а резидентная программа просто обязана это сделать (иначе не смогут выполняться другие программы). После завершения нерезидентной программы занимаемые ею блоки памяти освобождаются. Однако если программа завершается с тем, чтобы остаться резидентной, блоки с ее окружением и кодом остаются в памяти. Они будут освобождены только в результате явного удаления такой программы из ОЗУ.

Рис. 5.8. Цепочка блоков памяти

Удаление из ОЗУ резидентной программы, которая загружена не последней, ведет к *фрагментации памяти*. Следствием фрагментации памяти фактически является уменьшение свободной области для размещения новых программ, так как в образовавшуюся «дыру» удастся лишь скопировать окружение. Поэтому установку резидентов нужно тщательно планировать заранее, чтобы, по крайней мере, уменьшить (если не удастся полностью исключить) фрагментацию памяти.

Фрагментация памяти может возникнуть также и при выполнении такой последовательности действий, как:

1) временный выход на уровень DOS из выполняемой программы (в результате этого запускается вторичная копия КП);
2) установка резидентной программы;
3) возврат в выполняемую программу по команде EXIT;
4) завершение выполнения этой программы.

DOS обеспечивает возможность *динамического управления памятью* для размещения информационных объектов выполняемой программы. Программа может запросить:

— выделение блока памяти заданного размера или определение размера наибольшего незанятого блока (по функции 48H прерывания 21H);
— освобождение блока памяти (по функции 49H прерывания 21H);
— изменение размера уже выделенного блока памяти как в сторону увеличения, конечно при возможности этого, так и в сторону уменьшения (по функции 4AH прерывания 21H).

Дополнительная память

Последние версии DOS в совокупности с соответствующими внешними драйверами обеспечивают поддержку следующих видов дополнительной памяти (если не принимать во внимание виртуальные диски):

— *отображаемой* (expanded) *памяти*;
— *расширенной* (extended) *памяти*;
— *верхней* (high) *памяти*.

Независимо от вида памяти требуемые области в ней выделяются блоками (аналогично стандартной памяти), но каждый блок памяти управляется своим **обработчиком**.

Для того чтобы доступ к дополнительной памяти оказался возможным, необходимо иметь в ПЭВМ избыток памяти и, возможно, специальное оборудование.

Концепцию **отображаемой памяти** предложили и реализовали совместно три фирмы — Lotus, Intel и Microsoft. Ее спецификация обозначается как L/I/M EMS (Lotus/Intel/Microsoft Expanded Memory Specification — спецификация отображаемой памяти упомянутых фирм) или просто как LIM EMS. Средства EMS позволяют предоставлять дополнительную память для хранения данных с целью их использования выполняемыми программами и осуществлять доступ к этим данным путем задания адреса. Хранение же выполняемых программ в отображаемой памяти невозможно. L/I/M EMS допустима, начиная с DOS 3.3, и поддерживает дополнительную память до 8 Мбайт (EMS 3.2) или даже до 32 Мбайт (EMS 4.0) путем обеспечения видимости ее фрагментов через созданное в стандартном (1-Мбайт) адресном пространстве *окно*.

Принцип действия отображаемой памяти основан на техническом приеме, известном как *замещение страниц*, или как *переключение блоков памяти:* внутри адресного пространства, отведенного под ПЗУ и видеопамять, но DOS используемого не полностью, выделяется 64-Кбайт *окно,* в которое отображаются четыре произвольных 16-Кбайт *страницы* дополнительной памяти, необходимые в данный момент. Тем самым МП «вводится в заблуждение», поскольку с помощью виртуальной адресации он обращается к хранимым в окне данным, хотя на самом деле их физические адреса могут быть смещены в дополнительной памяти относительно окна на несколько Мбайт.

EMS может использоваться на любых IBM-совместимых ПЭВМ, включая XT. Для этого необходимо иметь в компьютере класса XT или AT специальную *плату памяти*, удовлетворяющую требованиям EMS, и программный драйвер, который управляет отображаемой памятью. Платы для XT и AT различаются. При выборе платы необходимо также учитывать соотношение ее быстродействия и тактовой частоты МП ПЭВМ. Некоторые платы могут быть сконфигурированы и таким образом, что размещенная на них память будет непосредственно рассматриваться как оперативная память с адресами свыше 1 Мбайт. ПЭВМ с МП 80386(SX) и 80486(SX) не требуют специальных плат, так как обладают встроенными возможностями отображения *логических* (*виртуальных*) адресов в *физические*. Поэтому при наличии в такой ПЭВМ ОЗУ большой емкости и подключении к DOS специального драйвера можно осуществить *эмуляцию отображаемой памяти*.

Читателю известно, что ПЭВМ класса AT и старше могут адресовать ОЗУ емкостью, существенно превышающей 640 Кбайт, и зачастую комплектуются памятью значительного объема. До недавнего времени вся дополнительная память в таких компьютерах могла задействоваться только для создания в ней виртуальных дисков. (Исключение составляют ПЭВМ с МП 80386(SX) и 80486(SX), где возможна эмуляция отображаемой памяти.)

Для более полного использования ОЗУ большой емкости на компьютерах класса AT и старше совсем недавно была предложена спецификация **расширенной памяти** (XMS — eXtended Memory Specification). Программный драйвер, ее реализующий, позволяет пересылать данные из ОЗУ в расширенную память и обратно. Одним из наиболее известных таких драйверов является HIMEM.SYS, входящий в комплект поставки MS-DOS, начиная с версии 4.0. Способ использования дополнительной памяти, принятый в XMS, отличается как от EMS, так и от подхода, воплощенного в концепции виртуального диска.

Различия спецификаций EMS и XMS проиллюстрированы на рис. 5.9.

Технически доступ к расширенной памяти осуществляется путем переключения МП из реального в защищенный режим работы, где вся память ПЭВМ может адресоваться без каких-либо трудностей. Основная проблема заключается в другом, а именно, в возврате в реальный режим работы. Это сделать немного сложнее, чем начальный переход. Реализованный в XMS-драйвере способ трудоемок и не отличается высокой надежностью, в связи с чем загрузка выполняемых программ в расширенную память не допускается.

Доступ к отображаемой и расширенной памяти осуществляется через аппарат прерываний DOS.

Верхней называется память в диапазоне адресов от 640K до 1M — 1. Эта область стандартного адресного пространства, как известно, зарезервирована для видеопамяти и ПЗУ, однако обычно в ней остаются «дыры». Одна из 64-Кбайт «дыр» может задействоваться под окно отображаемой памяти. Наряду с этим «дыры» можно использовать как обычную память, если обеспечить отображение их адресов на какие-либо модули оперативной памяти.

В настоящее время существуют две разновидности верхней памяти, каждая из которых обеспечивает хранение не только данных, но и выполняемых программ.

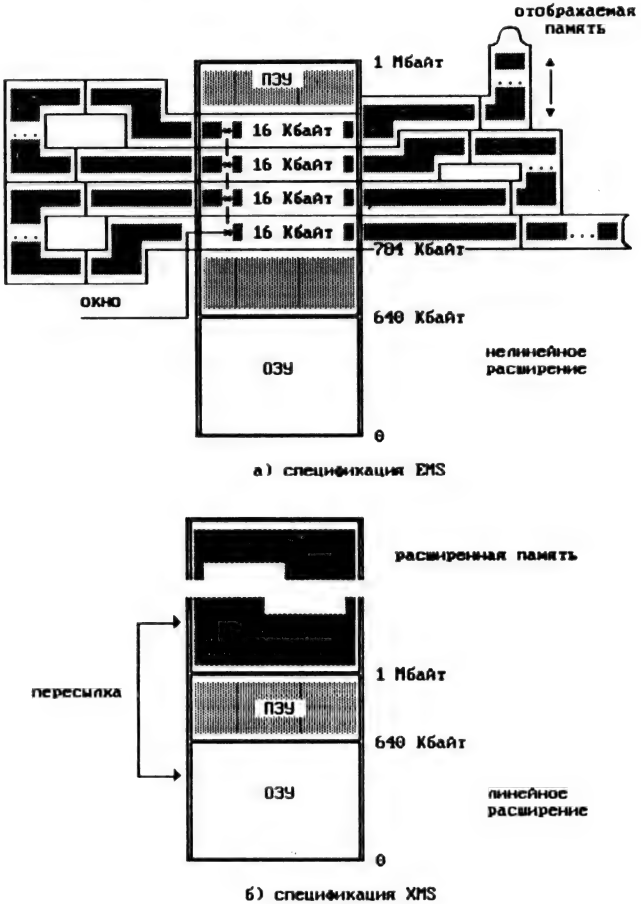


Рис. 5.9. Распределение памяти ПЭВМ в соответствии со спецификациями EMS и XMS

Первая разновидность верхней памяти поддерживается благодаря использованию различий МП 80286 и старше, с одной стороны, и прибора 8086/88, с другой стороны, при работе в реальном режиме. В МП 8086/88 адресация 64-Кбайт сегмента, начинающегося с последнего параграфа стандартного 1-Мбайт адресного пространства, невозможна, так как старший разряд адреса отбрасывается. При проектировании же прибора 80286 была допущена ошибка, в результате которой адресация в данной ситуации производилась к первым 64 Кбайт памяти, находящейся за 1-Мбайт границей (точнее — к первым 64 Кбайт — 16 байт, так как первый параграф остается в стандартном адресном пространстве). Для исключения описанного несоответствия в ПЭВМ класса AT и старше устанавливается специальный узел. Имеются драйверы, блокирующие его и тем самым обеспечивающие свободный доступ к указанной области памяти. В качестве примеров таких драйверов назовем уже упомянутый драйвер HIMEM.SYS, а также HIDOS.SYS, входящий в комплект поставки DR DOS 5.0. Данная разновидность верхней памяти может использоваться только специально разработанными программами. В нее помещаются, в частности, резидентные модули DOS 5.0, высвобождая тем самым стандартную память для прикладных программ. Таким образом, для увеличения стандартной памяти на 64 Кбайт за счет верхней памяти необходима ПЭВМ класса AT и старше с повышенной емкостью ОЗУ, а также соответствующий драйвер. Рассмотренную разновидность верхней памяти обычно называют **HMA-памятью** (HMA — High Memory Area — область верхней памяти).

Вторая разновидность верхней памяти поддерживается преимущественно на ПЭВМ с МП 80386(SX) и 80486(SX) путем отображения логических адресов памяти в физические, находящиеся за 1-Мбайт границей. В данном случае «дыры» в области видеопамяти и ПЗУ «заполняются» избыточной оперативной памятью. Эта техника аналогична реализации отображаемой памяти, но имеет два отличия:

— невозможно использовать память, превышающую размеры «дыр» (т.е. механизм переключения страниц не поддерживается);

— в верхнюю память можно загружать программы.

Обычно в верхнюю память второй разновидности помещают драйверы и резидентные программы, частично освобождая тем самым стандартную память. Поддержку такой загрузки обеспечивает только DOS 5.0.

В качестве драйвера для управления загрузкой программ в верхнюю память второй разновидности можно выбрать:

— драйвер 386MAX или Blue Max фирмы Qualites;

— драйвер QEMM версии 5.0 или более поздней, выпускаемый компанией Quarterdeck;

— драйвер HIDOS.SYS фирмы Digital Research.

Подчеркнем, что драйвер HIMEM.SYS не обеспечивает выполнение рассматриваемой функции. Тем не менее в комплект MS-DOS 5.0 входит драйвер EMM386.EXE, решающий эту задачу. Новейшие драйверы способны обеспечить отображение адресов даже на ПЭВМ класса AT. Таким образом, если Вы имеете ПЭВМ класса AT, снабженную платой EMS 4.0 и избыточной памятью, то все же можете обеспечить возможность загрузки драйверов и резидентных программ в верхнюю память, подключив к системе одну из следующих пар драйверов:

— драйверы MOVE-EM версии 1.02 или более поздней, предлагаемый фирмой Qualitas, и HIMEM.SYS;

— драйверы QRAM и QEXT компании Quarterdeck.

Вторую разновидность верхней памяти часто обозначают как **UMB** (Upper Memory Blocks — блоки верхней памяти).

5.3. Кодирование символов в ПЭВМ

Под **кодированием** понимают запись данных с использованием некоторого кода.

Для представления любой информации в современных ЭВМ используются *двоичные числа (двоичные коды)*. При вводе информации в ЭВМ специальными аппаратными и программными средствами каждый *символ* «преобразуется» в соответствующий двоичный код и записывается в ОЗУ или внешнюю память для последующей обработки. При выводе информации из ЭВМ осуществляется обратное преобразование: двоичный код каждого символа «переводится» во внешнее представление этого символа для его визуального отображения на устройстве вывода (например, принтер по коду символа печатает на бумаге его графический образ).

Числа, в отличие от других символов, при вводе-выводе обрабатываются несколько иначе. После ввода закодированное, но все еще десятичное число программным способом обычно переводится в двоичную систему счисления, чтобы обеспечить эффективное выполнение арифметических операций. Перед выводом осуществляется обратное преобразование.

За основу кодирования символов в ПЭВМ, как и во многих других ЭВМ, взят **код ASCII** (American Standard Code for Information Interchange — *американский стандартный код для обмена информацией*). Он введен в 1963 г. и ставит в соответствие каждому символу *семиразрядный* двоичный код. Легко определить, что в коде ASCII можно представить 128 символов.

ASCII-кодировка символов приведена в табл. 5.2. Элементами изображенной матрицы являются *обозначения символов*, а индексами — шестнадцатеричные цифры кодов символов. Для получения шестнадцатеричного кода символа необходимо к цифре-индексу, записанной в соответствующем столбце, приписать справа цифру-индекс, размещенную в соответствующей строке матрицы. Например, символ G имеет код 47H. Из шестнадцатеричного кода легко получить двоичный код путем записи каждой шестнадцатеричной цифры в двоичной системе счисления (для нашего примера получим 1000111). Если требуется по двоичному коду определить представленный им символ, то следует свернуть этот код в шестнадцатеричное число, разбить его на цифры и найти обозначение символа на пересечении столбца и строки, индексами которых являются старшая и младшая шестнадцатеричные цифры кода соответственно.

ASCII содержит две группы символов:

1) прописные и строчные латинские буквы, цифры, а также специальные знаки, т.е. *символы пишущей машинки*;

2) *управляющие символы,* используемые в коммуникационных протоколах, в частности, для передачи команд ПУ.

Эти группы символов разделены в табл. 5.2 двойной линией. Символы пишущей машинки имеют коды 20H — 7EH, а управляющие символы — 00H — 1FH и 7FH.

Относительно символов пишущей машинки специальные пояснения не требуются. Отметим только, что символ с кодом 2DH — это знак *«минус»*, а с кодом 5FH — *знак подчеркивания*. Символы с кодами 27H и 60H также различаются между собой: первый является *апострофом*, а второй — *знаком тупого (обратного) ударения*. Символ " (один символ!) называется *кавычками*, & — *амперсандом*, # — *знаком номера*, ^ — *стрелкой вверх*, | — *вертикальной чертой*, ~ — *знаком надчеркивания (тильдой)*, а @ — *коммерческим «эт»*. Символ SP обозначает пробел (Space).

Управляющие символы стандартно интерпретируются следующим образом:

NUL (NULl) — *пустой символ*, не имеющий значения (может использоваться для задержек);
SOH (Start Of Heading) — *начало заголовка* (с него начинается передача блока данных);

Таблица 5.2

Таблица символов кода ASCII

Старшая цифра	0	1	2	3	4	5	6	7
Младшая цифра								
0	NUL	DLE	SP	0	@	P	`	p
1	SOH	DC1	!	1	A	Q	a	q
2	STX	DC2	"	2	B	R	b	r
3	ETX	DC3	#	3	C	S	c	s
4	EOT	DC4	$	4	D	T	d	t
5	ENQ	NAK	%	5	E	U	e	u
6	ACK	SYN	&	6	F	V	f	v
7	BEL	ETB	'	7	G	W	g	w
8	BS	CAN	(	8	H	X	h	x
9	HT	EM	)	9	I	Y	i	y
A	LF	SUB	*	:	J	Z	j	z
B	VT	ESC	+	;	K	[	k	{
C	FF	FS	,	<	L	\	l	|
D	CR	GS	-	=	M	]	m	}
E	SO	RS	.	>	N	^	n	~
F	SI	US	/	?	O	_	o	DEL

STX (Start Of Text) — *начало текста* (отмечает начало текста, следующего за заголовком);
ETX (End of TeXt) — *конец текста* (отмечает конец текста и начало контрольного кода);
EOT (End Of Transmission) — *конец передачи*;
ENQ (ENQuiry) — *запрос подтверждения* (запрашивает информацию о статусе удаленной станции);
ACK (ACKnowledge) — *подтверждение* (подтверждает успешный обмен между станциями);
BEL (BELl) — *звонок* (инициирует выдачу звукового сигнала);
BS (BackSpace) — *возврат на одну позицию* (для отмены предыдущего символа);
HT (Horisontal Tab), или TAB — *горизонтальная табуляция*;
LF (Line Feed) — *перевод строки*;
VT (Vertical Tab) — *вертикальная табуляция*;
FF (Form Feed) — *переход к новой странице (перевод формата)*;
CR (Carriage Return) — *возврат каретки*;
SO (Shift Out) — *переход на нижний регистр*;
SI (Shift In) — *переход на верхний регистр*;
DLE (Data Link Escape) — *завершение сеанса связи*;
DC*i* (Device Control *i*) — *управление устройством i* ($i = 1,...,4$);
NAK (Negative AcKnowledge) — *ошибка передачи*;
SYN (SYNchronous idle) — *холостые данные синхронной передачи*;
ETB (End Transmit Block) — *конец передачи блока*;
CAN (CANcel) — *отмена* (сигнализирует об ошибке передачи);
EM (End of Medium) — *конец носителя данных* (сигнализирует о физическом конце источника данных);
SUB (SUBstitute) — *подстановка* (для замены символов);
ESC (ESCape) — *переход* (изменение интерпретации последующих кодов);
FS (File Separator) — *разделитель файлов*;
GS (Group Separator) — *разделитель групп*;
RS (Record Separator) — *разделитель записей*;
US (Unit Separator) — *разделитель элементов*;
DEL (DELete) — *удаление* (символа).

В той или иной ЭВМ могут использоваться не все управляющие символы, а задействованные могут интерпретироваться несколько иначе. Использование и интепретацию управляющих символов в IBM-совместимых ПЭВМ мы рассмотрим в следующем подразделе, когда станет ясно, как их вводить с клавиатуры.

С учетом легкости формирования символов из-за графического характера ПУ ПЭВМ (т.е. возможности формировать различные символы по точкам) ASCII совершенно недостаточен для представления многих полезных символов по причине того, что он содержит всего 128 кодов. Потребность в большем числе кодов особенно ощущается в случае необходимости работать на языках, алфавиты которых отличны от английского. С другой стороны, 7-разрядный двоичный код все равно занимает байт (8 разрядов). При коммуникациях между удаленными станциями дополнительный бит используется для контроля по четности (нечетности). В большинстве же ПЭВМ это не требуется и один бит в коде каждого символа остался бы незадействованным. По указанным причинам фирма IBM *расширила* ASCII таким образом, что каждый символ стал представляться 8-разрядным двоичным кодом. Это позволило закодировать уже 256 символов.

Зарегистрированная фирмой IBM *кодировка* (**кодовая страница 437**, называемая *американской кодировкой*) используется на всех IBM-совместимых ПЭВМ во всем мире и представлена в табл. 5.3. Вместо наименований символов в ячейках матрицы даны их *графические изображения* на экране дисплея. Следует иметь в виду, что для получения графических изображений всех символов на экране дисплея необходимо осуществить запись их кодов непосредственно в видеопамять или использовать одно из прерываний нижнего уровня. Попытка вывести изображение символа при помощи высокоуровневых средств файловой системы DOS не приведет к достижению желаемого результата, если символ оказывает управляющее воздействие на дисплей или трактуется DOS (либо другим используемым в настоящий момент программным продуктом) особым образом. Например, если попытаться вывести символ с кодом 07H (BEL), то вместо появления на экране дисплея крупной точки будет выдан звуковой сигнал. Аналогично при попытке вывести символ с кодом 09H (HT, TAB) вместо его изображения реакция дисплея будет иной: произойдет перемещение курсора вправо до очередной позиции табуляции. На вводе символов с клавиатуры мы остановимся особо в следующем подразделе (в этом случае также имеются определенные тонкости).

Расширенный код ASCII полностью включает в себя стандарт ASCII (левая половина матрицы) и дополнительно содержит 128 кодов с единицей в старшем бите (правая половина матрицы). Среди дополнительных символов часто используемые (но не все) *буквы ряда европейских алфавитов* (немецкого, французского, финского и др.), *буквы греческого алфавита, математические символы и символы псевдографики*. Последние можно использовать при вычерчивании простейших схем и таблиц (см. столбцы B, C и D). В качестве псевдографических допускается применять и некоторые (незадействованные в ПЭВМ и DOS) управляющие символы, которые будут перечислены в следующем подразделе.

Отметим, что символ с кодом FFH (как и 00H) не задействован и является «пустым».

В настоящее время под кодом ASCII применительно к ПЭВМ часто понимают описанное расширение ASCII.

Для использования в Европе больше подходит **кодовая страница 850** *(многоязыковая кодировка)*, содержащая как *латиницу*, так и *большинство букв европейских*, а также *северо-* и *южно-американских алфавитов*. В жертву же принесены буквы греческого алфавита и некоторые символы псевдографики, замена которых наименее ощутима. К таким псевдографическим символам относятся так называемые *полуторные символы* (соединения одинарных и двойных линий), изменение представления которых может исказить лишь углы диалоговых окон на экране.

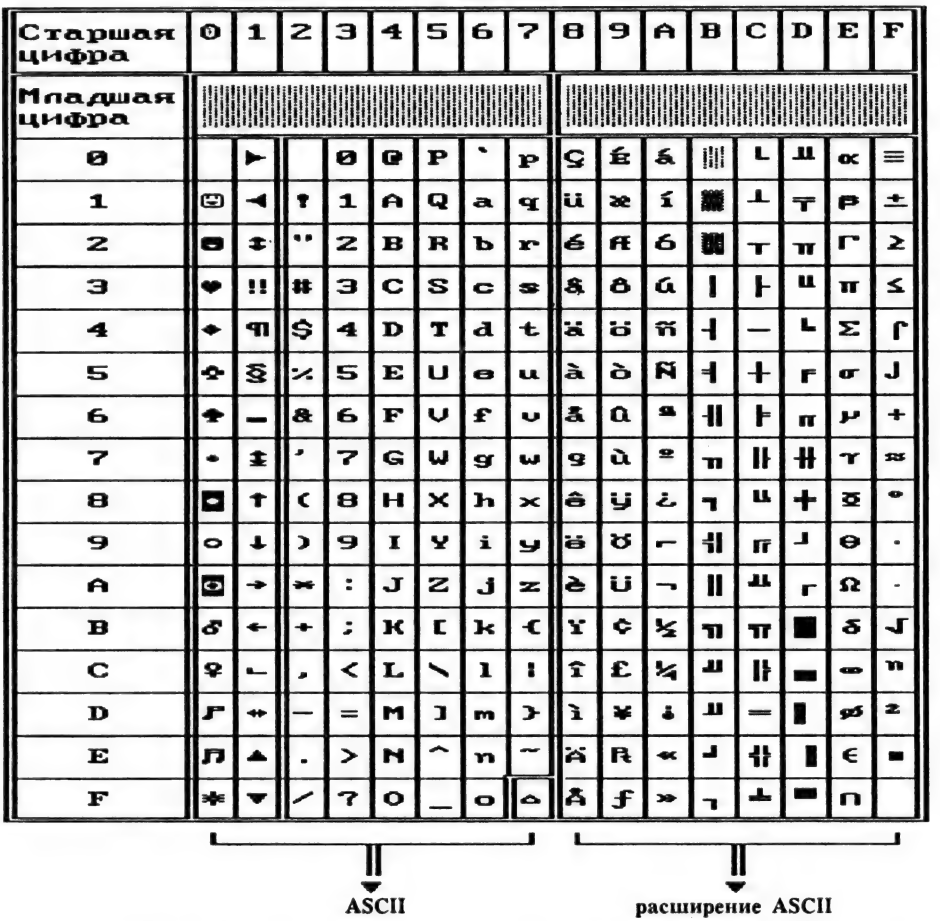
Нередко для практических целей требуется иметь информацию о кодах символов в десятичной или двоичной системах счисления. Поэтому в табл. 5.4 мы представили соответствие между кодами символов в шестнадцатеричной, двоичной и десятичной системах счисления. Эта таблица аналогична табл. 5.3, но вместо графических изображений символов проставлены их десятичные коды, а индексирование элементов матрицы может вестись не только в шестнадцатеричной, но и в двоичной системе счисления. Например, по табл. 5.3 можно установить, что шестнадцатеричным кодом символа M будет 4DH. Тогда на основании табл. 5.4 легко определить, что код этого символа в двоичной системе счисления равен 01001101, а в десятичной — 77.

Для представления символов *кириллицы* (букв русского алфавита) было предложено много модификаций кодовой таблицы IBM. Все они основываются на подмене некоторых символов символами кириллицы и иногда на изменении кодировки ряда символов. ГОСТом закреплена так называемая *основная кодировка*, в которой символы кириллицы размещены в столбцах B, C, D, E и частично F, а символы псевдографики перенесены в столбцы 8, 9 и A. Однако эта таблица распространения не получила, так как при ее использовании не могут нормально функционировать англоязычные программные продукты, выводящие на экран дисплея псевдографические символы (вместо рамок, окон и таблиц получаются неприятные изображения, состоящие из русских букв).

Для устранения этого недостатка Брябрин В.М. и Чижов А.А. предложили *альтернативную кодировку*, в которой символы псевдографики имеют те же коды, что и в таблице IBM, а русские буквы занимают столбцы 8, 9, A, E и частично F (см. табл. 5.5), замещая буквы европейских алфавитов, буквы греческого алфавита и математические символы. В альтернативной кодировке столбец F содержит некоторые дополнительные символы псевдографики и другие символы.

Таблица 5.3

Таблица символов расширенного кода ASCII



Рассматриваемый вариант кодировки не изменяет расположения в таблице символов псевдографики (столбцы B, C и D). Поэтому англоязычные программные продукты функционируют нормально. Однако русские буквы размещены несколько неудобно: их нормальная последовательность прерывается псевдографическими символами, вследствие чего в общем случае нельзя получить код следующей по алфавиту буквы путем прибавления единицы к коду очередной буквы. Это несколько ограничивает возможности манипулирования русскими буквами в развитых языках программирования. Тем не менее *лексикографический* (алфавитный) порядок следования букв все же сохранен, что в определенной степени сглаживает отмеченный недостаток альтернативной кодировки, позволяя, по крайней мере, сортировать слова в алфавитном порядке. До недавнего времени альтернативная кодировка была в СССР самой распространенной, да и сейчас широко применяется на отечественных ПЭВМ.

На машинах класса AT и старше альтернативную кодировку сменил ее *модифицированный вариант* (см. табл. 5.6). Отличие этой кодировки от прототипа состоит в том, что столбец F кодовой таблицы полностью заимствован из кодовой таблицы IBM. Целью такого «отката» назад является обеспечение большей совместимости данной таблицы с IBM-таблицей. Плата за это заключается в том, что из кодировки исключены русская буква Ё и ряд других полезных символов, включая стрелки.

Мы уже говорили о том, что фирма IBM недавно зарегистрировала *кодовую страницу для СССР* (теперь — СНГ). Кодировка символов в ней основывается на альтернативном варианте. Отличия состоят лишь в том, что в столбце F находятся уникальные *буквы украинского* и *белорусского алфавитов*, а также некоторые символы из кодовой таблицы IBM. Эта кодировка представлена в табл. 5.7.

Заметим, что буквы латиницы и кириллицы одинакового начертания во всех рассмотренных кодировках имеют разное представление. Это связано с необходимостью обеспечить лексиког-

Таблица 5.14

Соответствие шестнадцатеричных, двоичных и десятичных чисел

Старшие цифры	16-я	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
	2-е	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
Младшие цифры																	
16-я	2-е																
0	0000	0	16	32	48	64	80	96	112	128	144	160	176	192	208	224	240
1	0001	1	17	33	49	65	81	97	113	129	145	161	177	193	209	225	241
2	0010	2	18	34	50	66	82	98	114	130	146	162	178	194	210	226	242
3	0011	3	19	35	51	67	83	99	115	131	147	163	179	195	211	227	243
4	0100	4	20	36	52	68	84	100	116	132	148	164	180	196	212	228	244
5	0101	5	21	37	53	69	85	101	117	133	149	165	181	197	213	229	245
6	0110	6	22	38	54	70	86	102	118	134	150	166	182	198	214	230	246
7	0111	7	23	39	55	71	87	103	119	135	151	167	183	199	215	231	247
8	1000	8	24	40	56	72	88	104	120	136	152	168	184	200	216	232	248
9	1001	9	25	41	57	73	89	105	121	137	153	169	185	201	217	233	249
A	1010	10	26	42	58	74	90	106	122	138	154	170	186	202	218	234	250
B	1011	11	27	43	59	75	91	107	123	139	155	171	187	203	219	235	251
C	1100	12	28	44	60	76	92	108	124	140	156	172	188	204	220	236	252
D	1101	13	29	45	61	77	93	109	125	141	157	173	189	205	221	237	253
E	1110	14	30	46	62	78	93	110	126	142	158	174	190	206	222	238	254
F	1111	15	31	47	62	79	95	111	127	143	159	175	191	207	223	239	255

Таблица 5.5

Таблица символов альтернативной кодировки

Старшая цифра	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Младшая цифра																
0		▶		0	@	P	`	p	А	Р	а	░	└	╨	р	Ё
1	☺	◀	!	1	A	Q	a	q	Б	С	б	▒	┴	╤	с	ё
2	☻	↕	"	2	B	R	b	r	В	Т	в	▓	┬	╥	т	ђ
3	♥	‼	#	3	C	S	c	s	Г	У	г	│	├	╙	у	ѓ
4	♦	¶	$	4	D	T	d	t	Д	Ф	д	┤	─	╘	ф	є
5	♣	§	%	5	E	U	e	u	Е	Х	е	╡	┼	╒	х	ѕ
6	♠	▬	&	6	F	V	f	v	Ж	Ц	ж	╢	╞	╓	ц	→
7	•	↨	'	7	G	W	g	w	З	Ч	з	╖	╟	╫	ч	←
8	◘	↑	(	8	H	X	h	x	И	Ш	и	╕	╚	╪	ш	↓
9	○	↓	)	9	I	Y	i	y	Й	Щ	й	╣	╔	┘	щ	↑
A	◙	→	*	:	J	Z	j	z	К	Ъ	к	║	╩	┌	ъ	÷
B	♂	←	+	;	K	[	k	{	Л	Ы	л	╗	╦	■	ы	±
C	♀	∟	,	<	L	\	l	!	М	Ь	м	╝	╠	▄	ь	№
D	♪	↔	-	=	M	]	m	}	Н	Э	н	╜	═	▌	э	¤
E	♫	▲	.	>	N	^	n	~	О	Ю	о	╛	╬	▐	ю	■
F	☼	▼	/	?	O	_	o	⌂	П	Я	п	┐	╧	▀	я	

рафический порядок следования букв. Факт различия кодов одноименных букв разных алфавитов нужно обязательно учитывать при работе на ПЭВМ.

Все представленные кодировки с кириллицей имеют следующий недостаток: программные продукты, выдающие сообщения и принимающие информацию от пользователя на европейских языках, не могут функционировать нормально. Поэтому такие программы неудобно использовать в СНГ, а кодовые страницы с кириллицей не могут быть приняты в европейских странах. Усилиями ряда ведущих советских программистов и руководителей совместных предприятий найдено компромиссное решение, при котором в кодовой таблице остаются буквы европейских алфавитов, добавляются символы кириллицы, а в жертву приносится часть псевдографических символов. Эта кодировка получила название *европейской* и также зарегистрирована фирмой IBM. Она включает *латиницу, кириллицу,* те же *буквы европейских алфавитов*, что и кодовая страница 437, а также только те *символы псевдографики,* которые содержатся в кодовой странице 850. Заметным недостатком данной таблицы является нарушение лексикографического порядка следования букв кириллицы.

На смену базирующимся на ASCII кодировкам символов в ПЭВМ в скором времени, видимо, придет недавно созданная 16-битная **кодировка Unicode**. Она содержит 65536 символов, а именно:

— практически все *символы большинства языков народов мира*;

— *элементы* китайских, корейских и японских *иероглифов*, позволяющие строить любые иероглифы;

— *специальные символы* (знаки пунктуации, транскрипции, математические и технические символы, а также псевдографические символы);

Таблица 5.6

Таблица символов модифицированной альтернативной кодировки

Старшая цифра	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Младшая цифра																
0		▶		0	@	P	`	p	А	Р	а	▒	└	ш	р	≡
1	☺	◀	!	1	A	Q	a	q	Б	С	б	▓	┴	т	с	±
2	☻	↕	"	2	B	R	b	r	В	Т	в	▓	т	п	т	≥
3	♥	!!	#	3	C	S	c	s	Г	У	г	│	├	ц	у	≤
4	♦	¶	$	4	D	T	d	t	Д	Ф	д	┤	—	┗	ф	⌠
5	♣	§	%	5	E	U	e	u	Е	Х	е	┤	┼	┏	х	J
6	♠	▬	&	6	F	V	f	v	Ж	Ц	ж	╣	╞	п	ц	÷
7	•	↨	'	7	G	W	g	w	З	Ч	з	┐	╢	╫	ч	≈
8	◘	↑	(	8	H	X	h	x	И	Ш	и	┐	╙	╪	ш	°
9	○	↓	)	9	I	Y	i	y	Й	Щ	й	╡	╓	┘	щ	·
A	◙	→	*	:	J	Z	j	z	К	Ъ	к	╢	╨	┌	ъ	·
B	♂	←	+	;	K	[	k	{	Л	Ы	л	╗	╤	■	ы	√
C	♀	∟	,	<	L	\	l	|	М	Ь	м	╝	╟	▄	ь	©
D	♪	↔	—	=	M	]	m	}	Н	Э	н	╜	═	▌	э	²
E	♫	▲	.	>	N	^	n	~	О	Ю	о	┘	╬	▐	ю	■
F	☼	▼	/	?	O	_	o	⌂	П	Я	п	┐	╧	▀	я	

— *5000 мест для частного использования*;

— *резерв* из 30000 мест.

В Unicode не входят управляющие символы, однако возможность их использования предусмотрена.

С помощью данной кодировки разрешится множество проблем, связанных с ограниченностью набора доступных символов.

Для удобства вычерчивания линий и таблиц, а также заполнения фона в текстовых редакторах мы выделили соответствующие символы псевдографики из рассмотренных кодировок и представили их (вместе с кодами) в табл. 5.8. Эти символы инвариантны ко всем четырем явно приведенным в данном подразделе кодировкам. Псевдографические символы разбиты на группы по функциональному признаку и изображены в центральном поле таблицы. В левом поле с той же расстановкой записаны их коды в десятичной, в правом — в шестнадцатеричной системе счисления. Порядок их ввода с клавиатуры будет обсуждаться в следующем подразделе. Заметим, что в эту таблицу затесались и четыре управляющих символа, которые не оказывают управляющих воздействий ни на принтер, ни на дисплей. Однако они могут не печататься на некоторых типах принтеров, в особенности, в сочетании с драйверами кириллицы. Эти вопросы можно уточнить по документации на Ваш принтер, где содержатся все поддерживаемые им кодовые таблицы.

5.4. Организация ввода информации с клавиатуры

В п. 2.5.1 уже кратко охарактеризована клавиатура как основное устройство ввода информации в ПЭВМ. Здесь же мы подробно рассмотрим различные варианты клавиатур персональных

Таблица 5.7

Кодировка символов для СССР (кодовая страница 866)

Старшая цифра	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
Младшая цифра																
0		▶		0	@	P	`	p	А	Р	а	▒	└	ш	р	Ё
1	☺	◀	!	1	A	Q	a	q	Б	С	б	▓	┴	т	с	ё
2	☻	↕	"	2	B	R	b	r	В	Т	в	▓	┬	п	т	Є
3	♥	!!	#	3	C	S	c	s	Г	У	г	│	├	и	у	є
4	♦	¶	$	4	D	T	d	t	Д	Ф	д	┤	─	ь	ф	Ї
5	♣	§	%	5	E	U	e	u	Е	Х	е	╡	┼	г	х	ї
6	♠	▬	&	6	F	V	f	v	Ж	Ц	ж	╢	╞	п	ц	Ў
7	•	↨	'	7	G	W	g	w	З	Ч	з	╖	╫	╫	ч	ў
8	◘	↑	(	8	H	X	h	x	И	Ш	и	╕	╪	╪	ш	°
9	○	↓	)	9	I	Y	i	y	Й	Щ	й	╣	╥	┘	щ	·
A	◙	→	*	:	J	Z	j	z	К	Ъ	к	║	╙	г	ъ	·
B	♂	←	+	;	K	[	k	{	Л	Ы	л	╗	╘	▌	ы	√
C	♀	∟	,	<	L	\	l	|	М	Ь	м	╝	╒	▄	ь	№
D	♪	↔	-	=	M	]	m	}	Н	Э	н	╜	═	▌	э	¤
E	♫	▲	.	>	N	^	n	~	О	Ю	о	╛	╬	▐	ю	■
F	☼	▼	/	?	O	_	o	⌂	П	Я	п	┐	╧	▀	я	

машин фирмы IBM и совместимых с ними компьютеров, опишем принципы работы этих клавиатур, организацию ввода символов и командных строк. Наконец, приведем перечень специальных клавиш DOS и дадим их назначение.

Необходимость детального рассмотрения перечисленных вопросов диктуется тем, что любой пользователь обязательно будет использовать клавиатуру и поэтому должен хорошо представлять себе ее возможности, а программисту необходимо знать и организацию ввода информации с клавиатуры на достаточно низком уровне.

5.4.1. Клавиатуры IBM-совместимых ПЭВМ

Первой клавиатурой фирмы IBM для ПЭВМ была **83-клавишная клавиатура**, которой комплектовались машины IBM PC и IBM PC XT. Общий вид ее показан на рис. 5.10.

Клавиши пишущей машинки расположены в центральной части клавиатуры и служат для ввода букв (прописных и строчных), цифр и различных специальных знаков. Нижняя длинная, никак не помеченная клавиша, называется Space (пробел) и обеспечивает ввод одноименного символа.

К *служебным* относятся клавиши Esc, ⇦, ⇆, ↵, ⇧, PrtSc, Ctrl, Alt, Caps Lock, Num Lock и Scroll Lock (Break). Они расположены в разных частях клавиатуры и в общем случае имеют следующее назначение:

Esc — (от Escape — *«переход»*) — служит для отмены каких-либо действий и/или выхода из программы, подменю и т.п.;

⇦ — *клавиша возврата*; при ее нажатии курсор перемещается по экрану влево на одну позицию — тем самым удаляется предыдущий символ;

Таблица 5.8

Кодировка псевдографических символов

Назначение	Десятичные коды			Символы			Шестнадцатеричные коды		
Линии	196		205	─		═	C4		CD
	179		186	│		‖	B3		BA
Таблицы	218	194	191	┌	┬	╖	DA	C2	BF
	195	197	180	├	┼	┤	C3	C5	B4
	192	193	217	└	┴	┘	C0	C1	D9
	201	203	187	╔	╦	╗	C9	CB	BB
	204	206	185	╠	╬	╣	CC	CE	B9
	200	202	188	╚	╩	╝	C8	CA	BC
	214	210	183	╓	╥	╖	D6	D2	B7
	199	215	182	╟	╫	╢	C7	D7	B6
	211	208	189	╙	╨	╜	D3	D0	BD
	213	209	184	╒	╤	╕	D5	D1	B8
	198	216	181	╞	╪	╡	C6	D8	B5
	212	207	190	╘	╧	╛	D4	CF	BE
Заполнители	176	177	178	▒	▓	▓	B0	B1	B2
Заполнители/линии		223			▀			DF	
	222	219	221	▌	▓	▐	DE	DB	DD
		220			▄			DC	
Стрелки		30			▲			1E	
	17		16	◄		►	11		10
		31			▼			1F	

↹ — *клавиша табуляции*; действует только на нижнем регистре и обеспечивает перемещение курсора вправо до очередной позиции табуляции, интервал между которыми равен восьми символам; эту клавишу удобно использовать, например, при формировании таблиц и набора текста с отступами; на верхнем регистре возможно перемещение курсора до очередной позиции табуляции влево;

↵ — *клавиша ввода (возврата каретки)*; служит для завершения ввода очередной строки информации; курсор при нажатии клавиши перемещается в крайнее левое положение следующей строки;

Ctrl (от Control — *управляющая*) — самостоятельного значения не имеет, но при нажатии совместно с другой клавишей изменяет ее действие;

⇧ — *клавиша смены регистра*; если клавиатура находится на нижнем регистре, то при нажатии этой клавиши осуществляется переход на верхний регистр (можно будет вводить прописные буквы и специальные знаки, изображенные в верхних частях клавиш); на нижнем регистре возможен ввод строчных букв, цифр и специальных знаков, изображения которых нанесены в нижних частях клавиш; если клавиатура находится на верхнем регистре, то нажатие клавиши ⇧ переводит ее на нижний регистр; эта клавиша логически не фиксируется, в результате чего ее требуется удерживать; может изменять действие других клавиш;

PrtSc (от Print Screen — *печать экрана*); при индивидуальном нажатии клавиша эквивалентна клавише *, в то время как ее нажатие на фоне клавиши ⇧ приводит к распечатке на принтере информации, видимой на экране;

Alt (от Alternate — *изменяющая*) — так же, как и Ctrl, самостоятельного значения не имеет, но при нажатии совместно с другой клавишей изменяет действие последней;

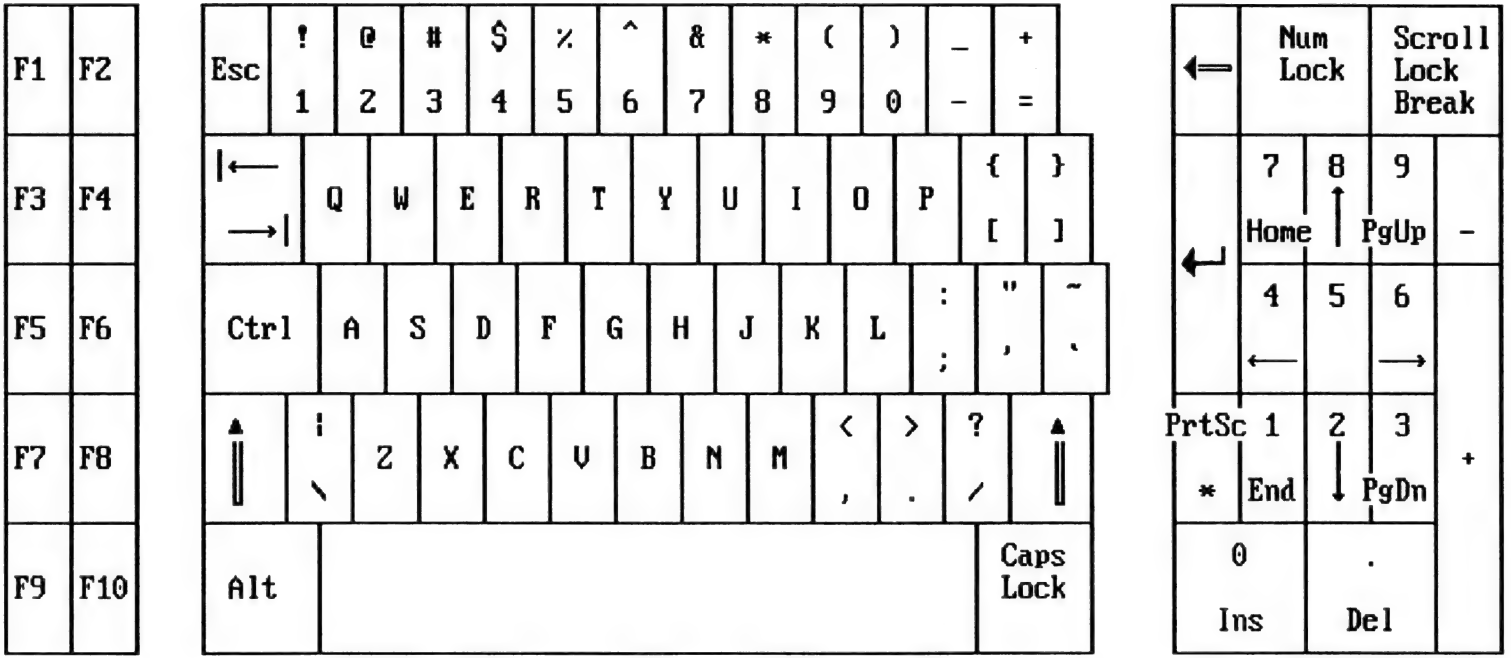


Рис. 5.10. Стандартная клавиатура IBM PC XT

Caps Lock (от Capitals Lock — *фиксация прописных букв*) — служит для фиксации верхнего регистра клавиатуры; при повторном нажатии фиксируется нижний регистр и т.д.; ее удерживать не надо;

Num Lock (от Number Lock — *фиксация цифр*) — обеспечивает переключение (с фиксацией) режимов работы малой цифровой клавиатуры (см. ниже);

Scroll Lock — *клавиша блокировки прокрутки*; самостоятельно используется для переключения режима вывода на экран дисплея, если при нажатии клавиш управления курсором сдвигается не курсор, а экран; может применяться аналогично клавишам Ctrl, ⇧, и Alt, но пока для этих целей не задействована;

Break — *клавиша прерывания*; самостоятельного значения не имеет, но на фоне клавиши Ctrl может привести к принудительному завершению выполнения текущей программы или команды.

Клавиши Sroll Lock и Break на клавиатуре IBM PC XT совмещены.

Заметим, что клавиша ⇦ на клавиатуре обозначена как ←. Мы умышленно изменили обозначение для того, чтобы отличить ее от соответствующей клавиши управления курсором.

При описании назначения служебных клавиш употреблялся термин «курсор». **Курсором** называется значок, указывающий знакоместо на экране дисплея, в котором будет отображаться очередной выведенный на экран символ.

Сделаем еще одно замечание. Назначение клавиш рассмотрено нами в увязке с экраном дисплея. Может возникнуть следующий вопрос: какое отношение дисплей имеет к клавиатуре? Ответ на него прост: обычно все символы, набираемые на клавиатуре, немедленно показываются на экране. Вот поэтому удобно давать назначение клавиш в терминах их воздействия на информацию, отображаемую на экране дисплея.

Для обеспечения ввода русских букв в той или иной кодировке необходим внешний драйвер, который, будучи подключенным к DOS, при нажатии определенной комбинации клавиш переключает клавиатуру в режим ввода символов кириллицы и наоборот. Часто такой комбинацией являются обе клавиши ⇧. Возможны и другие варианты, в том числе явно задаваемые пользователем при установке драйвера.

Функциональные клавиши F1 — F10 размещены в левой части клавиатуры. Они обычно программируются и для каждого программного продукта имеют свое назначение. Тем не менее уже стало традицией задействовать клавишу F1 для получения подсказки.

Малая цифровая клавиатура находится в правой части клавиатуры и содержит следующие клавиши: 7 (Home), 8 (↑), 9 (PgUp), —, 4 (←), 5, 6 (→), +, 1 (End), 2 (↓), 3 (PgDn), 0 (Ins) и . (Del). Малая цифровая клавиатура может работать в двух режимах:

1) в режиме ввода чисел;
2) в режиме управления курсором.

Переключение режимов (с логической фиксацией) осуществляется клавишей Num Lock, а без фиксации — клавишей ⇧. «Состояние» клавиши Caps Lock здесь значения не имеет. В *режиме ввода чисел* эта часть клавиатуры обеспечивает более удобный ввод чисел и знаков арифметических операций. В *режиме управления курсором* клавиши малой цифровой клавиатуры служат для перемещения курсора, перелистывания страниц и переключения режимов работы основной клавиатуры.

Рассмотрим традиционное назначение клавиш во *втором режиме*, хотя оно в принципе зависит от используемого программного продукта (т.е. эти клавиши обычно программируются):

←,↑,→,↓ — служат для перемещения курсора соответственно влево, вверх, вправо и вниз на одну позицию (строку) и называются *стрелками*;

Home (*исходное положение, начало*) — обеспечивает перемещение курсора в первую позицию строки;

End (*конец*) — служит для перемещения курсора в последнюю позицию строки;

PgUp (от Page Up — *страница вверх*) — обеспечивает перемещение по тексту в направлении его начала на одну страницу (обычно на 25 строк), т.е. возврат на одну страницу;

PgDn (от Page Down — *страница вниз*) — служит для перемещения по тексту в направлении его конца на одну страницу, т.е. продвижение вперед на одну страницу; иными словами клавиши PgUp и PgDn обеспечивают листание назад и вперед соответственно;

Ins (от Insert — *вставить*) — служит для переключения клавиатуры из режима замены в режим вставки и обратно; в *режиме замены* каждый вновь введенный символ заменяет на экране символ, указываемый курсором; *в режиме вставки* вводимый символ помещается перед символом, на который указывает курсор; часть же строки, расположенная правее курсора, сдвигается на одну позицию вправо;

Del (от Delete — *удалить*) — обеспечивает удаление на экране указанного курсором символа; при этом часть строки, расположенная правее курсора, сдвигается на одну позицию влево, исключая разрыв строки; «состояние» клавиши Ins на действие клавиши Del влияния не оказывает.

ПЭВМ IBM PC AT первоначально поставлялась с **84-клавишной клавиатурой**, которая отличается от рассмотренной лишь наличием одной дополнительной клавиши Sys Req, относящейся к малой цифровой клавиатуре. Клавиша Sys Req (от System Request — *запрос к системе*) может применяться в многопользовательских системах для входа в главное меню системы. В DOS эта клавиша не задействуется.

В 1986 г. фирма IBM, отреагировав на критику пользователей, разработала **усовершенствованную 101-клавишную клавиатуру** и стала поставлять ее с IBM PC AT, а по специальному заказу — и с XT. Общий вид этой клавиатуры с зарегистрированным недавно фирмой IBM размещением символов кириллицы и специальных символов для набора на «русском» регистре показан на рис. 5.11. Заметим, что это размещение в части, касающейся букв и цифр, ничем не отличается от принятого на пишущих машинках и клавиатурах отечественных ПЭВМ.

Чтобы понять, как действуют клавиши пишущей машинки на различных регистрах, условно разделите каждую клавишу на 4 части, проведя линию по горизонтали и по вертикали. Обозначение на левой верхней четвертинке соответствует верхнему латинскому регистру, на левой нижней четвертинке — нижнему латинскому регистру, на верхней правой четвертинке — верхнему русскому регистру, а на нижней правой четвертинке — нижнему русскому регистру. Если нижняя правая четвертинка пуста, то возьмите для нее обозначение из нижней левой четвертинки; когда пустой является верхняя правая четвертинка, то воспользуйтесь обозначением из нижней правой четвертинки; если же последняя тоже пуста, то для верхней правой четвертинки возьмите обозначение с верхней левой четвертинки. Имейте при этом в виду, что на нижнем регистре всегда вводятся строчные буквы, а на верхнем — прописные. Действие клавиш в конечном счете определяется драйвером клавиатуры, а поэтому обозначения клавиш на русском регистре могут не совпадать с вводимыми путем их нажатия символами.

Усовершенствованная клавиатура имеет нижеперечисленные особенности:

— несколько изменено расположение клавиш пишущей машинки;
— добавлены следующие служебные клавиши: правая Alt, правая Ctrl и Pause;
— изменено расположение и наименование некоторых служебных клавиш;
— клавиша Break совмещена с клавишей Pause;
— клавиша SysRq (Sys Req) совмещена с клавишей PrtSc;
— в правой верхней части клавиатуры имеется световая индикация положения ("состояния") логически фиксирующих служебных клавиш;
— добавлены две функциональные клавиши — F11 и F12;
— продублированы клавиши малой цифровой клавиатуры для режима управления курсором;
— к малой цифровой клавиатуре для режима ввода чисел добавлены клавиши Enter, / и *.

Полностью новой является клавиша Pause (*пауза*), при нажатии которой осуществляется приостановка (до нажатия какой-либо другой клавиши) выполнения ПЭВМ любых действий (в частности, вывода информации на экран дисплея).

Клавиатура моделей ПЭВМ семейства PS/2 по сравнению с рассмотренной обладает рядом отличий.

Производители IBM-совместимых ПЭВМ либо копируют рассмотренные клавиатуры, либо несущественно модифицируют их.

Клавиатуры отечественных ПЭВМ базируются на клавиатуре IBM PC XT. В качестве примера на рис. 5.12 показан общий вид клавиатуры ПЭВМ ЕС1840/41. Отличия данной клавиатуры от прототипа состоят в следующем:

— изменено расположение некоторых клавиш пишущей машинки и служебных клавиш;
— обозначения служебных и функциональных клавиш, а также клавиш малой цифровой клавиатуры выполнены на русском языке или в виде мнемонических рисунков;
— добавлены клавиши ЛАТ, РУС, Р/Л и ИНФ.

Клавиши ЛАТ и РУС являются логически фиксирующими и служат для переключения клавиатуры на латинский или русский регистры соответственно. Клавиша Р/Л предназначена для кратковременной смены этих регистров. Она не фиксируется, т.е. при ее нажатии клавиатура переходит на другой регистр, а при отпускании — возвращается в исходное состояние. Использовать эту клавишу нужно совместно с одной из клавиш пишущей машинки. Клавиша ИНФ задействована в ОС М86 для получения подсказок пользователем.

Легко видеть, что одни и те же клавиши на различных клавиатурах обозначены по-разному. Чтобы в этих условиях можно было легко сориентироваться, мы свели альтернативные обозначения одинаковых клавиш в табл. 5.9. В дальнейшем будем использовать обозначения, указанные в правом столбце этой таблицы. Клавиша пробела будет обозначаться через Space.

Общепринятое назначение клавиш мы уже рассмотрели. Теперь перечислим соглашения об использовании некоторых *комбинаций клавиш*, не оказывающих управляющего воздействия:

Ctrl-←/→ — сдвиг курсора на слово влево/вправо;

Ctrl-End — удаление текста от позиции курсора до конца строки;

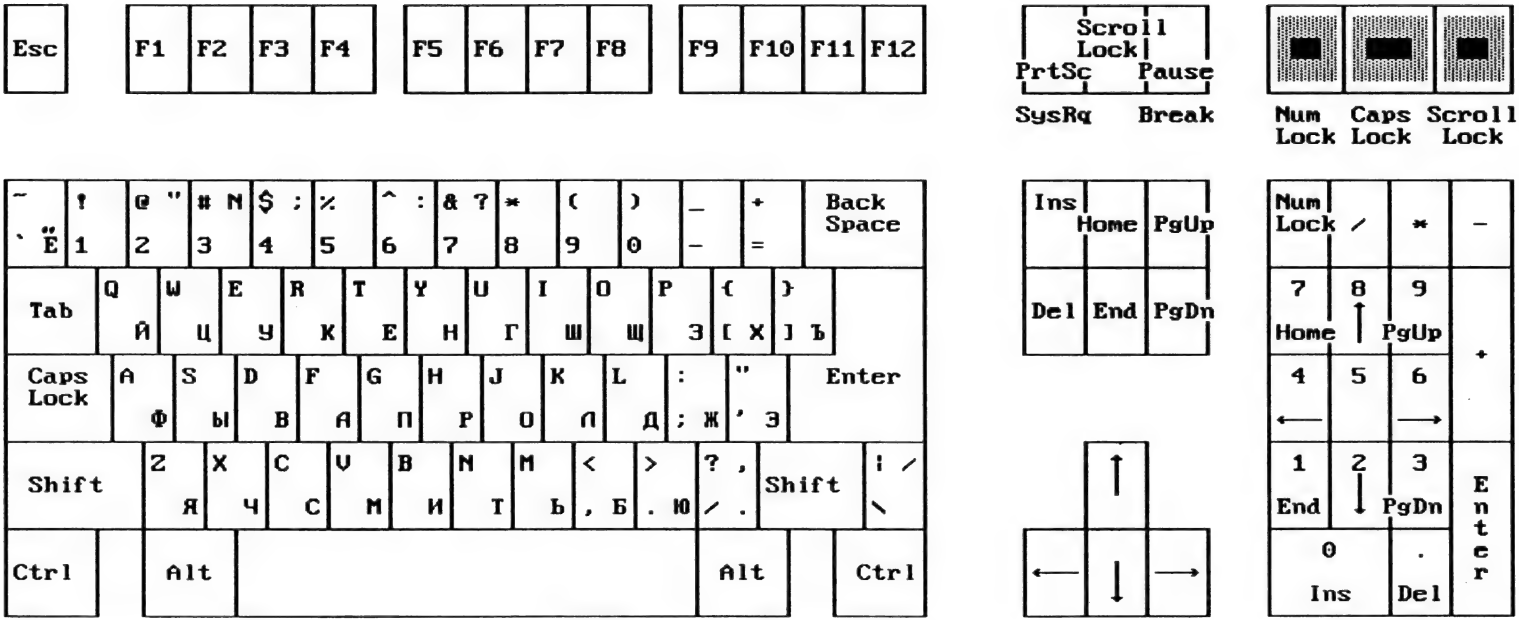


Рис. 5.11. Усовершенствованная клавиатура IBM PC AT

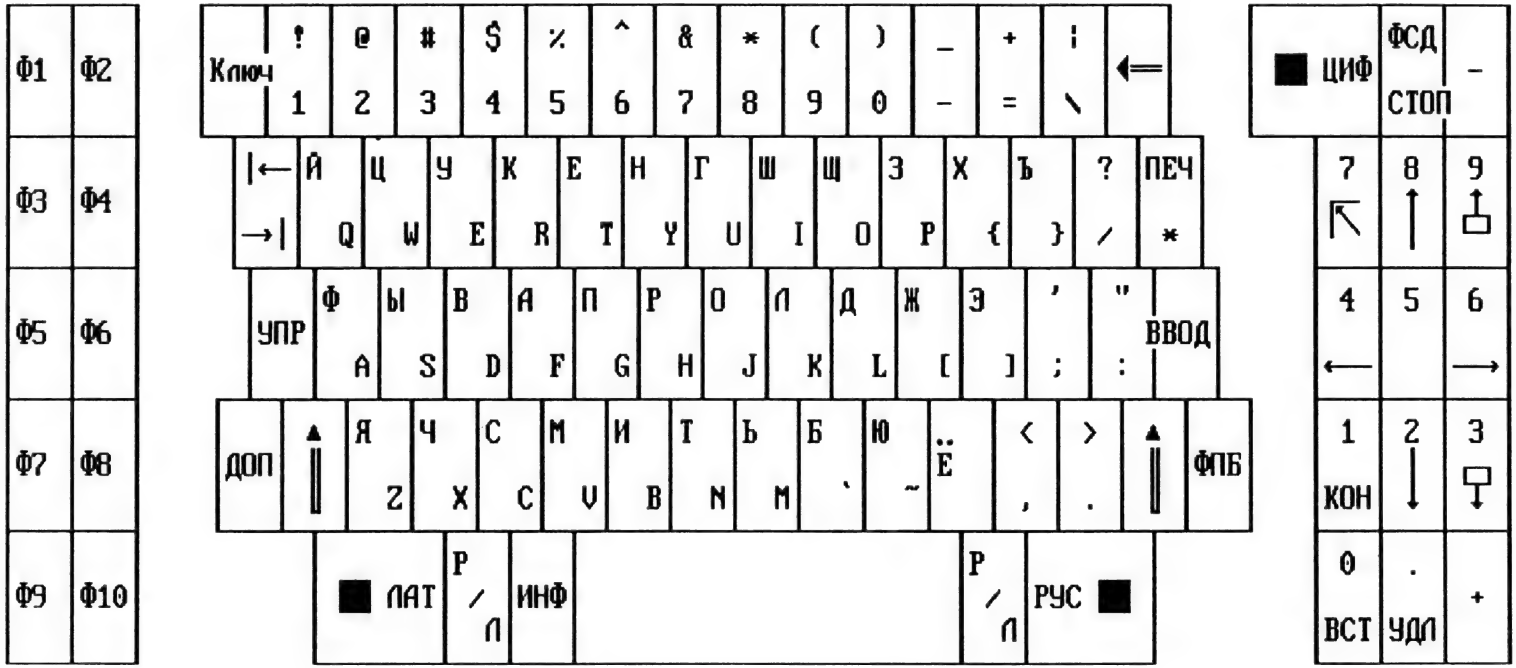


Рис. 5.12. Клавиатура ЕС1840/41

Таблица 5.9

Соответствие обозначений клавиш на различных клавиатурах

Клавиатура IBM PC XT	Альтернативные обозначения	Используемые обозначения
Esc	КЛЮЧ, Спец	Esc
←	Back Space, BS	BS
↹	Tab, Таб	Tab
↵	Enter, CR, Return, ВВОД	Enter
Ctrl	УПР, CTR	Ctrl
⇧	Shift, Shft, Верх	Shift
PrtSc	Print Screen, PrS, ПЕЧ, Печ Экр	PrtSc
Alt	ДОП, Альт	Alt
Caps Lock	ФПБ, Фикс Верх	Caps Lock
Num Lock	ЦИФ, NLK, БлкЦифр	Num Lock
Scroll Lock	ФСД, S, Блк прокр	Scroll Lock
Break (*)	Scroll Lock (на XT), Pause (на AT), СТОП, B	Break
Sys Req (*)	SysRq	Sys Req
Pause (*)		Pause
F1 - F10	Ф1 - Ф10	F1 - F10
↑,→,↓,←		↑,←,↓,→
Home	↖, ⤒ , НАЧ, Начало	Home
End	КОН, →, Кнц	End
PgUp	Page Up, ⇞ , ± , СтрВв, СтрВверх	PgUp
PgDn	Page Down, ⇟ , ∓ , СтрВнз, СтрВниз	PgDn
Ins	Insert, ВСТ, ⟷	Ins
Del	Delete, УДЛ, ⇥ , Удал	Del

(*) клавиатуры других ПЭВМ

Ctrl-PgDn — удаление текста от позиции курсора до конца экрана;

Ctrl-Home — очистка экрана и перемещение курсора в левый верхний угол экрана;

Ctrl-PgUp — возврат на одну страницу и установка курсора в левый верхний угол экрана (либо переход к началу текста, либо переход к началу экрана).

Очевидно, устоявшееся соглашение по применению комбинации клавиш Ctrl-PgUp отсутствует.

Использование программистами всех перечисленных здесь и выше соглашений при разработке программного продукта упрощает освоение его пользовательского интерфейса. Однако, к сожалению, не все разработчики ПО этим руководствуются.

5.4.2. Принцип действия клавиатуры

После рассмотрения кодирования символов в ПЭВМ и клавиатур становится очевидным следующее:

1) каждому расширенному коду ASCII не может быть поставлена в соответствие определенная клавиша на клавиатуре, так как количество клавиш намного меньше;

2) не каждая комбинация клавиш может быть представлена в расширенном коде ASCII в связи с тем, что этих комбинаций может быть достаточно много (существенно больше 256).

Более того, многим одиночным клавишам, не входящим в группу клавиш пишущей машинки, в большинстве случаев не ставится в соответствие какой-либо расширенный код ASCII. Тем не менее эти клавиши и комбинации клавиш полезно задействовать в создаваемых программных продуктах, чтобы облегчить работу пользователя за клавиатурой путем уменьшения объема вводимой информации. Ведь различным комбинациям клавиш можно поставить в соответствие множество различных вариантов действий DOS, программы и ПЭВМ.

Исходя из сказанного возникают вопросы о том, *как обеспечить ввод с клавиатуры любого расширенного кода ASCII*, и о том, *как в общем случае фиксируются в ПЭВМ результаты нажатий клавиш и их комбинаций*. Ответ на первый вопрос будет дан в следующем пункте, а в рамках данного пункта мы получим ответ на второй вопрос, рассмотрев принцип действия клавиатуры.

Вначале остановимся на клавиатуре ПЭВМ IBM PC XT, а затем разберем особенности функционирования клавиатур компьютера IBM PC AT.

Каждой клавише на **клавиатуре XT** соответствует семиразрядный **код сканирования**, или **опроса клавиши** (scan-код). Перечень этих кодов приведен в табл. 5.10. Буква М в ней предваряет наименования клавиш на малой цифровой клавиатуре, если это не следует из контекста. Подчеркнем, что приведенные коды не совпадают с ASCII-кодами.

При нажатии клавиши аппаратура клавиатуры генерирует однобайтный **код нажатия**, а при отпускании — также однобайтный **код отпускания**. *Код нажатия* совпадает с кодом сканирования. *Код* же *отпускания* отличается от кода сканирования наличием единицы в старшем разряде байта

Таблица 5.10

Коды сканирования клавиш клавиатуры IBM PC XT

Клавиша	Код сканирования		Клавиша	Код сканирования	
	16-я	10-я		16-я	10-я
Esc	01	1	\ |	2B	43
1 !	02	2	Z	2C	44
2 @	03	3	X	2D	45
3 #	04	4	C	2E	46
4 $	05	5	V	2F	47
5 %	06	6	B	30	48
6 ^	07	7	N	31	49
7 &	08	8	M	32	50
8 *	09	9	, <	33	51
9 (	0A	10	. >	34	52
0)	0B	11	/ ?	35	53
- _	0C	12	Shift (правая)	36	54
= +	0D	13	* PrtSc	37	55
BS	0E	14	Alt	38	56
Tab	0F	15	Space	39	57
Q	10	16	Caps Lock	3A	58
W	11	17	F1	3B	59
E	12	18	F2	3C	60
R	13	19	F3	3D	61
T	14	20	F4	3E	62
Y	15	21	F5	3F	63
U	16	22	F6	40	64
I	17	23	F7	41	65
O	18	24	F8	42	66
P	19	25	F9	43	67
[{	1A	26	F10	44	68
] }	1B	27	Num Lock	45	69
Enter	1C	28	Scroll Lock	46	70
Ctrl	1D	29	Home 7	47	71
A	1E	30	↑ 8	48	72
S	1F	31	PgUp 9	49	73
D	20	32	(M)-	4A	74
F	21	33	← 4	4B	75
G	22	34	(M)5	4C	76
H	25	35	→ 6	4D	77
J	24	36	(M)+	4E	78
K	25	37	End 1	4F	79
L	26	38	↓ 2	50	80
; :	27	39	PgDn 3	51	81
' "	28	40	Ins 0	52	82
` ~	29	41	Del .	53	83
Shift (левая)	2A	42			

(т.е. он больше кода сканирования на 128, или 80H). Если клавиша остается нажатой дольше, чем 0,5 с, то автоматически начинают генерироваться ее коды нажатия с частотой 10 раз в секунду, что имитирует серию быстрых нажатий на эту клавишу. Такой принцип очень удобен при необходимости многократного ввода одного и того же символа. Автоматическая *регенерация* кода нажатия прекращается при отпускании клавиши или нажатии другой клавиши. Поэтому в случае, когда клавиша «залипла», для устранения нежелательных последствий этого достаточно нажать любую другую клавишу. Код нажатия/отпускания выдается в порт 60H.

Одновременно с записью кода нажатия/отпускания в порт 60H оборудование клавиатуры выдает аппаратное прерывание 09H. При этом МП немедленно приостанавливает свою текущую работу и переходит к обработке возникшего прерывания по программе-драйверу клавиатуры. Драйвер клавиатуры считывает из порта 60H код нажатия/отпускания, а дальнейшие его действия зависят от типа кода (нажатие или отпускание) и от типа клавиши, соответствующей этому коду.

Драйвер клавиатуры различает следующие типы клавиш:

1) *клавиши-переключатели*;

2) *кодовые клавиши.*

Клавиши-переключатели служат для изменения состояния клавиатуры (переключения регистров). На каждом регистре нажатие кодовых клавиш интерпретируется по-разному. К клавишам-переключателям (на клавиатуре IBM PC XT) относятся клавиши Ins, Caps Lock, Num Lock, Scroll Lock, Alt, Ctrl, левая и правая клавиши Shift, а также комбинация клавиш Ctrl-Num Lock. Для хранения статуса (состояния) клавиш-переключателей в области данных BIOS, находящейся в ОЗУ, зарезервировано два байта.

Если драйвер клавиатуры получил код нажатия (отпускания) клавиши-переключателя, то соответствующий бит статуса устанавливается в единицу (нуль). В результате появляется возможность распознать одновременное нажатие клавиши-переключателя и какой-либо другой кла-

виши. Дополнительно к этому за каждой логически фиксирующейся клавишей-переключателем (Ins, Caps Lock и Num Lock) закреплен еще один бит в статусе, содержимое которого определяет состояние соответствующего режима (вставки, ввода прописных букв и фиксации цифр). При включении режима указанный бит устанавливается драйвером клавиатуры в единицу, а при выключении — сбрасывается.

Нажатие комбинации клавиш Ctrl-Num Lock обрабатывается несколько иначе, однако конечный результат будет совпадать с описанным в предыдущем абзаце.

Кодовые клавиши (к которым относятся все остальные клавиши) служат для непосредственного ввода символов в ПЭВМ, т.е. при нажатии любой из них в ОЗУ компьютера передается соответствующий ей код, который может использоваться выполняемыми программами.

При считывании драйвером клавиатуры кода нажатия такой клавиши осуществляется анализ байтов статуса и с учетом их содержимого код нажатия преобразуется (транслируется) в **двухбайтовый код символа**. Структура этого кода определяется тем, есть или нет в расширенном коде ASCII соответствие нажатой клавише (комбинации клавиш).

Если была нажата клавиша на клавиатуре пишущей машинки и клавиатура находится только на нижнем или верхнем регистре, а также если была нажата клавиша на малой цифровой клавиатуре (только в режиме фиксации цифр), то первым (младшим) байтом двухбайтового кода символа будет расширенный код ASCII соответствующего символа, а вторым (старшим) — код сканирования соответствующей клавиши. Код сканирования не является принципиально необходимым, однако он все же фиксируется для того, чтобы в программе можно было определить, какая же все-таки из одноименных клавиш нажата (например, клавиша + на малой цифровой клавиатуре или на клавиатуре пишущей машинки).

Допустим, что клавиатура находится на нижнем регистре и нажата клавиша A. При этом драйвер клавиатуры считывает код нажатия 1EH (так как код сканирования клавиши A равен 1EH), а выдает двухбайтовый код 1E61H (в связи с тем, что расширенный код ASCII буквы «а» равен 61H).

Нажатие некоторых служебных клавиш, не являющихся переключателями, также приводит к формированию двухбайтового кода описанной структуры. Это клавиши Esc, BS, Tab и Enter. Они определяют расширенные коды ASCII 27, 8, 9 и 13 соответственно. Символы с этими кодами являются управляющими в ASCII и обозначаются соответственно ESC, BS, HT (TAB) и CR (см. табл. 5.2).

В противном случае, т.е. когда нажатой клавише или комбинации клавиш не соответствует никакой расширенный код ASCII (с учетом состояния клавиатуры — регистра), будет сформирован двухбайтовый код, называемый **расширенным кодом клавиши**. Расширенный код клавиши в первом (младшем) байте содержит нулевой код (ASCII-код символа NUL), а во втором (старшем) — двоичное число, однозначно определяющее нажатую клавишу (или комбинацию клавиш). Последнее число часто совпадает с кодом сканирования клавиши. К клавишам, после обработки нажатий которых выдается расширенный код клавиши, относятся функциональные клавиши, клавиши малой цифровой клавиатуры в режиме управления курсором, а также комбинации различных клавиш с Alt, Shift и Ctrl (однако многие комбинации с Ctrl используются для ввода управляющих символов кода ASCII, а с Shift — символов пишущей машинки на верхнем регистре). Кодировка клавиш и их комбинаций в расширенных кодах клавиш сведена в табл. 5.11, где представлены только вторые байты кодов.

Отметим следующее:

— клавиша-переключатель Scroll Lock драйвером клавиатуры пока не используется. Чтобы применить ее для ввода новых расширенных кодов клавиш, необходимо заменить обработчик прерывания 09H или программно анализировать соответствующий бит байтов статуса и осуществлять дополнительное перекодирование;

— клавиша Del не соответствует управляющему символу DEL из ASCII;

— клавиша Ins на самом деле выполняет роль не только клавиши-переключателя, но и роль кодовой клавиши (при обработке ее нажатия драйвер выдает расширенный код клавиши);

— расширенные коды клавиш жестко привязаны к конкретным клавишам, а не к их обозначениям;

— комбинации Alt-*цифра* и Alt-– — обозначают клавиши на основной (но не на малой) клавиатуре;

— состояние логически фиксирующих клавиш Caps Lock, Num Lock и Scroll Lock на генерацию расширенных кодов клавиш не влияет;

— состояние логически фиксирующей клавиши Num Lock влияет только на интерпретацию нажатий одиночных клавиш на малой цифровой клавиатуре (генерируются либо расширенные коды ASCII, либо расширенные коды клавиш).

Например, двухбайтовый код 1E00H представляет комбинацию клавиш Alt-A (1EH — код сканирования клавиши A).

Наличие нулевого кода в младшем байте двухбайтового кода символа позволяет программным способом отличать расширенный код ASCII от расширенного кода клавиши.

Считывание драйвером клавиатуры кода отпускания кодовой клавиши ни к каким действиям не приводит.

Двухуровневая кодировка символов при вводе с клавиатуры (в кодах сканирования и двухбайтовых кодах) обеспечивает возможность перепрограммирования клавиш.

Таблица 5.11

Расширенные коды клавиш клавиатуры IBM PC XT

Клавиши	Код		Клавиши	Код	
	16-я	10-я		16-я	10-я
F1	3B	59	Alt-A	1E	30
F2	3C	60	Alt-B	30	48
F3	3D	61	Alt-C	2E	46
F4	3E	62	Alt-D	20	32
F5	3F	63	Alt-E	12	18
F6	40	64	Alt-F	21	33
F7	41	65	Alt-G	22	34
F8	42	66	Alt-H	23	35
F9	43	67	Alt-I	17	23
F10	44	68	Alt-J	24	36
			Alt-K	25	37
Shift-F1	54	84	Alt-L	26	38
Shift-F2	55	85	Alt-M	32	50
Shift-F3	56	86	Alt-N	31	49
Shift-F4	57	87	Alt-O	18	24
Shift-F5	58	88	Alt-P	19	25
Shift-F6	59	89	Alt-Q	10	16
Shift-F7	5A	90	Alt-R	13	19
Shift-F8	5B	91	Alt-S	1F	31
Shift-F9	5C	92	Alt-T	14	20
Shift-F10	5D	93	Alt-U	16	22
			Alt-V	2F	47
Ctrl-F1	5E	94	Alt-W	11	17
Ctrl-F2	5F	95	Alt-X	2D	45
Ctrl-F3	60	96	Alt-Y	15	21
Ctrl-F4	61	97	Alt-Z	2C	44
Ctrl-F5	62	98			
Ctrl-F6	63	99	Alt-0	81	129
Ctrl-F7	64	100	Alt-1	78	120
Ctrl-F8	65	101	Alt-2	79	121
Ctrl-F9	66	102	Alt-3	7A	122
Ctrl-F10	67	103	Alt-4	7B	123
			Alt-5	7C	124
Alt-F1	68	104	Alt-6	7D	125
Alt-F2	69	105	Alt-7	7E	126
Alt-F3	6A	106	Alt-8	7F	127
Alt-F4	6B	107	Alt-9	80	128
Alt-F5	6C	108	Alt--	82	130
Alt-F6	6D	109	Alt-=	83	131
Alt-F7	6E	110			
Alt-F8	6F	111	Ctrl-@	03	3
Alt-F9	70	112	Shift-Tab	0F	15
Alt-F10	71	113	Ins	52	82
			Del	53	83
↑	48	80	Ctrl-PrtSc	72	114
←	4B	75			
→	4D	77	Ctrl-←	73	115
↓	48	72	Ctrl-→	74	116
End	4F	79	Ctrl-End	75	117
Home	47	71	Ctrl-Home	77	119
PgDn	51	81	Ctrl-PgDn	76	118
PgUp	49	73	Ctrl-PgUp	84	132

Сформированный двухбайтовый код символа помещается драйвером клавиатуры в **буфер клавиатуры**, расположенный в области данных BIOS в ОЗУ. На этом обработка нажатия клавиши завершается, и МП продолжает выполнять отложенную работу. Буфер рассчитан на хранение 15 двухбайтовых кодов клавиш и логически организован в виде циклического списка-очереди. Двухбайтовый код только что нажатой клавиши (комбинации клавиш) размещается в конце списка, если он не пуст. Асинхронно по запросу из программы на ввод символа (конечно, по прерыванию) осуществляется выдача из буфера первого двухбайтового кода списка с одновременным удалением этого кода из буфера. Если на момент запроса буфер пуст, то программа переходит в состояние ожидания реального ввода символа с клавиатуры. Наличие буфера и описанная техника организации ввода позволяют нажимать клавиши на клавиатуре с упреждением (заранее), что несколько ускоряет работу на ПЭВМ. В случае, когда буфер в результате этого оказывается заполненным, нажатие следующей клавиши сопровождается звуковым сигналом, а двухбайтовый код отвергается (его просто некуда помещать).

Непосредственный доступ к клавиатуре за двухбайтовым кодом клавиши осуществляется прикладной программой или системой по прерыванию 16H, обработчик которого составляет вторую важную часть драйвера клавиатуры.

DOS содержит и высокоуровневые средства для считывания из буфера целой строки. Но все они обращаются к обработчику прерывания 16H.

Любая функция DOS, осуществляющая ввод (а реально — считывание из буфера) одного символа, возвращает либо расширенный код ASCII, либо нулевой код. В последнем случае необходимо повторить программный запрос для выдачи другого (старшего) байта расширенного кода клавиши. Программисты на языке Ассемблера могут в первом случае получить и код сканирования клавиши.

Языки программирования для ПЭВМ зачастую также располагают аналогичными низкоуровневыми средствами.

Заметим, что один лишь ввод каких-либо (управляющих) символов еще не обеспечивает автоматического выполнения ПЭВМ определенных действий. Последние нужно программировать, либо они уже запрограммированы в DOS или другой выполняемой в данный момент программе. Исключение составляют следующие специальные комбинации клавиш, при нажатии которых никакой код в буфер клавиатуры не записывается, но выполняются предопределенные действия после их распознавания обработчиком прерывания 09H:

Ctrl-Alt-Del — рестарт ОС;

Ctrl-Break — принудительное завершение выполнения программы;

Ctrl-Num Lock — приостановка выполнения программы до нажатия какой-либо клавиши на клавиатуре, за исключением клавиши Num Lock;

Shift-PrtSc — печать экрана.

Все эти функции реализуются путем выдачи драйвером клавиатуры соответствующих прерываний.

Коды нажатия и отпускания клавиш на **84-клавишной клавиатуре** ПЭВМ IBM PC AT совпадают, но последние предваряются однобайтовым кодом F0H.

В остальном принципы работы 84-клавишной клавиатуры и ввода символов аналогичны описанным выше. Единственное отличие связано с наличием клавиши Sys Req, код сканирования которой равен 84 (54H). Клавиша Sys Req уникальна по своей функции. При обнаружении ее нажатия или отпускания драйвер клавиатуры генерирует прерывание 15H, обработчику которого через регистр AX МП передается информация о том, что прерывание инициировано от клавиши Sys Req, а также о том, нажата эта клавиша или отпущена. Обработчик прерывания 15H пока не выполняет никаких действий, а просто осуществляет возврат. Клавишу Sys Req можно задействовать в программе, подменив вектор прерывания 15H. Нажатие или отпускание этой клавиши не приводит к записи какого-либо кода в буфер клавиатуры.

101-клавишная клавиатура обладает большей гибкостью, допуская, в частности, программирование интервала времени, в течение которого необходимо удерживать клавишу до начала периодической регенерации ее кода нажатия, а также частоты регенерации. Принципы работы этой клавиатуры существенно отличаются от рассмотренных. Так, 101-клавишная клавиатура имеет три различных варианта соответствия кодов сканирования клавишам. Требуемый вариант выбирается путем выдачи соответствующей информации в порт 64H. Если вариант явно не задавался, то действует вариант с номером 2. Этот вариант сильно отличается от кодов сканирования 84-клавишной клавиатуры. Но аппаратные средства клавиатуры транслируют коды сканирования в такие, которые совместимы с кодами сканирования 84-клавишной клавиатуры, прежде чем они становятся доступными выполняемым программам.

Благодаря этому все коды сканирования 84-клавишной клавиатуры совпадают с кодами сканирования соответствующих клавиш 101-клавишной клавиатуры. Дополнительные коды сканирования для новых клавиш приведены в табл. 5.12. Отметим, что 101-клавишная клавиатура генерирует не только коды сканирования одиночных клавиш, но и некоторых комбинаций клавиш, а также не только одиночные коды, но и их последовательности. Подчеркнем, что все указанные в таблице клавиши при наличии альтернативы уникальны именно для этой клавиатуры. Например, Ins означает не клавишу на малой цифровой клавиатуре, а дополнительную клавишу вне малой цифровой клавиатуры (см. рис. 5.11). Единственное расхождение по кодам сканирования между 84- и 101-клавишными клавиатурами состоит в интерпретации клавиши PrtSc. Оно связано с тем, что на рассматриваемой клавиатуре эта клавиша не предназначена для ввода символа *, а совмещена с клавишей Sys Req. Код сканирования клавиши * на малой цифровой клавиатуре 101-клавишной клавиатуры совпадает с кодом сканирования клавиши PrtSc на 84-клавишной клавиатуре. Если у Вас 101-клавишная клавиатура, то для печати содержимого экрана достаточно нажать единственную клавишу PrtSc. В тех случаях, когда дополнительная клавиша на 101-клавишной клавиатуре имеет то же наименование, что и заимствованная из 84-клавишной клавиатуры, второй байт кода сканирования совпадает со старым кодом.

В байтах статуса клавиатуры отведено дополнительно по одному биту для левых клавиш Ctrl и Alt, а уже задействованные биты взводятся как при нажатии левых, так и при нажатии правых клавиш Ctrl и Alt. Таким образом, например, если бит для Ctrl установлен в единицу, а бит для левой Ctrl сброшен, то, значит, нажата правая Ctrl. Такой, на первый взгляд странный, подход тем не менее позволяет обеспечить совместимость с 84-клавишной клавиатурой и распознать

Таблица 5.12

Дополнительные коды сканирования клавиш 101-клавиатуры IBM PC AT

Клавиши	Код сканирования	
	16-я	10-я
F11	57	87
F12	58	88
Alt (правая)	E0 38	224 56
Ctrl (правая)	E0 1D	224 29
PrtSc	E0 2A E0 37	224 42 224 55
Shift-PrtSc (Sys Req)	E0 37	224 55
Ctrl-PrtSc (Sys Req)	E0 37	224 55
Alt-PrtSc	54	84
Pause	E1 1D 45 E1 9D C5	225 29 69 225 157 197
Ctrl-Pause (Break)	E0 46 E0 C6	224 70 224 198
Ins	E0 52	224 82
Shift-Ins	E0 AA E0 52	224 170 224 82
Del	E0 53	224 83
Shift-Del	E0 AA E0 53	224 170 224 83
←	E0 4B	224 75
Shift-←	E0 AA E0 4B	224 170 224 75
Home	E0 47	224 71
Shift-Home	E0 AA E0 47	224 170 224 71
End	E0 4F	224 79
Shift-End	E0 AA E0 4F	224 170 224 79
↑	E0 48	224 72
Shift-↑	E0 AA E0 48	224 170 224 72
↓	E0 50	224 80
Shift-↓	E0 AA E0 50	224 170 224 80
PgUp	E0 49	224 73
Shift-PgUp	E0 AA E0 49	224 170 224 73
PgDn	E0 51	224 81
Shift-PgDn	E0 AA E0 51	224 170 224 81
→	E0 4D	224 77
Shift-→	E0 AA E0 4D	224 170 224 77
(M)Enter	E0 1C	224 28
(M)/	E0 35	224 53
Shift-(M)/	E0 AA E0 35	224 170 224 53

нажатие левой (правой) клавиши Ctrl (Alt). Еще один бит в статусе зарезервирован для клавиши Sys Req, вследствие чего она может использоваться в качестве переключателя, но такая возможность стандартно не реализована. Дополнительно к комбинации клавиш Ctrl-Num Lock соответствующий бит статуса переключает и клавиша Pause, в связи с чем она действует так же, вызывая паузу в обработке.

Драйвер 101-клавишной клавиатуры обеспечивает выдачу дополнительных расширенных кодов клавиш, вторые байты которых сведены в табл. 5.13. Отметим, что дополнения связаны не только с наличием новых клавиш, но и с задействованием других комбинаций уже имеющихся клавиш. Как и раньше, для устранения неоднозначности клавиши малой цифровой клавиатуры помечены буквой М в скобках. Но здесь явно выделены все такие клавиши, так как они на клавиатуре продублированы. Если признак (М) не указан, то клавиша находится вне малой цифровой клавиатуры. Как и в случае кодов сканирования, от этого зависит значение расширенного кода клавиши.

К сожалению, не всегда в BIOS прошит драйвер 101-клавишной клавиатуры: зачастую имеющийся драйвер эмулирует работу только 84-клавишного пульта.

Обобщим все рассмотренные в данном пункте сведения, необходимые пользователю при работе на клавиатуре, а также программисту, не интересующемуся вводом символов из порта, в табл. 5.14. Такая необходимость вызвана тем, что эта информация была разрозненной, а просто запомнить ее невозможно. В таблице имеются и дополнительные сведения, касающиеся комбинаций с клавишей Ctrl, обеспечивающих ввод управляющих символов ASCII.

Табл. 5.14 позволяет определить, какой код вводится при нажатии клавиши или допустимой комбинации клавиш на клавиатуре. Возможно и диаметрально противоположное ее использование: определение того, как ввести требуемый код. Коды комбинаций клавиш делятся на две группы:

— *расширенные коды ASCII,* которые представлены их обозначениями;

— *расширенные коды клавиш,* которые обозначены через n;m, где n -первый, а m — второй байт в десятичной системе счисления.

Если клавиша или комбинация клавиш не обеспечивает ввод какого-либо кода, то соответствующее поле содержит запись н_д, что означает «не действует». Запись «спец» указана для комбинаций, которые входят в состав специальных комбинаций клавиш или непосредственно являются таковыми.

Таблица применима к любой из рассмотренных клавиатур. Пометка (—) указывает на то, что данная запись (клавиша или код) соответствует только 83- и/или 84-клавишным клавиатурам;

Таблица 5.13

Дополнительные расширенные коды клавиш 101-клавишной клавиатуры IBM PC AT

Клавиши	Код		Клавиши	Код	
	16-А	10-А		16-А	10-А
F11	85	133	Alt-↑	98	152
F12	86	134	Alt-↓	A0	160
			Alt-←	9B	155
Shift-F11	87	135	Alt-→	9D	157
Shift-F12	88	136			
			Alt-Del	A3	163
Ctrl-F11	89	137	Alt-End	9F	159
Ctrl-F12	8A	138	Alt-Home	97	151
			Alt-Ins	A2	162
Alt-F11	8B	139	Alt-PgUp	99	153
Alt-F12	8C	140	Alt-PgDn	A1	161
Alt-[	1A	26	Alt-(M)/	A4	164
Alt-]	1B	27	Alt-(M)*	37	55
Alt-;	27	39	Alt-(M)-	4A	74
Alt-'	28	40	Alt-(M)+	4E	78
Alt-`	29	41	Alt-(M)Enter	A6	166
Alt-\	2B	43			
Alt-,	33	51	Ctrl-(M)/	95	149
Alt-.	34	52	Ctrl-(M)*	96	150
Alt-/	35	53	Ctrl-(M)-	8E	142
			Ctrl-(M)+	90	144
Alt-BS	0E	14	Ctrl-(M)↑	8D	141
Alt-Enter	1C	28	Ctrl-(M)5	8F	143
Alt-Esc	01	1	Ctrl-(M)↓	91	145
Alt-Tab	A5	165	Ctrl-(M)Ins	92	146
Ctrl-Tab	94	148	Ctrl-(M)Del	93	147

пометка же (+) выделяет записи, относящиеся только к 101-клавишной клавиатуре. Записи, ничем не помеченные, относятся к любой клавиатуре.

В первом столбце перечислены клавиши всех клавиатур. Содержимое таблицы соответствует выключенным режимам Caps Lock и Num Lock. Это ни в коей мере не ограничивает ее значимость, так как фиксация данных клавиш имитируется удержанием клавиши Shift в нажатом состоянии (имеются в виду только буквенные клавиши и клавиши малой цифровой клавиатуры; интерпретация нажатия других клавиш от состояния режимов Caps Lock и Num Lock не зависит). Второй столбец таблицы определяет коды клавиш на основном регистре, т.е. коды одиночных клавиш. Остальные столбцы задают коды комбинаций клавиш с Shift, Ctrl и Alt.

На некоторых клавиатурах имеется клавиша Null, обеспечивающая ввод расширенного кода клавиши 0;3.

В зависимости от производимого эффекта при нажатии клавиши и их комбинации можно подразделить на следующие группы:

1) клавиши и их комбинации, обеспечивающие ввод символов расширенного кода ASCII;

2) клавиши и их комбинации, при нажатии которых вводятся расширенные коды клавиш;

3) специальные комбинации клавиш, при нажатии которых ПЭВМ выполняет предопределенные действия;

4) клавиши-переключатели, нажатие которых не приводит к вводу какого-либо кода, но обеспечивает изменение статуса клавиатуры;

5) комбинации клавиш, не приводящие к какому-либо эффекту;

6) комбинации клавиш, сводимые к другим комбинациям на основании приоритетов клавиш-переключателей.

Дело в том, что из одновременно нажатых клавиш-переключателей Shift, Ctrl и Alt только одна остается активной, а именно та, которая имеет наиболее высокий приоритет. В порядке убывания приоритета эти клавиши располагаются следующим образом: Alt, Ctrl, Shift.

Детальный анализ табл. 5.14 показывает следующее:

1) с клавиатуры можно ввести практически все символы ASCII (но не расширенного), за исключением символа NUL, что достигается путем нажатия одиночной клавиши или комбинации клавиши Ctrl с одной из кодовых клавиш;

2) имеется 5 символов ASCII, которые можно ввести двумя способами:

BS — нажатием клавиши BS или комбинации Ctrl-H;
HT (TAB) — нажатием клавиши Tab или комбинации Ctrl-I;
CR — нажатием клавиши Enter или комбинации Ctrl-M;
ESC — нажатием клавиши Esc или комбинации Ctrl-{;
LF — нажатием комбинаций Ctrl-Enter или Ctrl-J.

Символ NUL без ухищрений ввести не удастся, так как его код используется в качестве идентификатора расширенных кодов клавиш. Решить проблему ввода символа NUL можно следующими способами:

1) использовать в программе специальное представление этого символа *последовательностью* других символов;

2) использовать для ввода NUL какую-нибудь *комбинацию* клавиш, генерирующую расширенный код клавиши, или *множество* таких комбинаций.

В последнем случае программа должна после считывания первого (нулевого) байта расширенного кода клавиши (т.е. символа NUL) дополнительно ввести второй байт и проанализировать его. По результатам анализа можно отбросить второй байт, и тогда останется код символа NUL. Обычно для этого используется комбинация Ctrl-@.

Оба эти способа имеют следующие недостатки: исключается возможность использования зарезервированных для представления символа NUL символов (комбинаций клавиш) по своему прямому назначению; необходимо осуществлять определенный анализ в программе.

Представлять же символ NUL в программе в качестве константы, как правило, труда не составляет.

Смысл использования клавиши Ctrl для ввода управляющих символов ASCII, а также техника ввода символов, образующих расширение ASCII, будут описаны в следующем пункте.

Таблица 5.14

Коды клавиш

Клавиши	Основной регистр	Shift-	Ctrl-	Alt-
1	2	3	4	5
Пишущей машинки:				
1 !	1	!	н_д	0;120
2 @	2	@	0;3	0;121
3 #	3	#	н_д	0;122
4 $	4	$	н_д	0;123
5 %	5	%	н_д	0;124
6 ^	6	^	RS	0;125
7 &	7	&	н_д	0;126
8 *	8	*	н_д	0;127
9 (	9	(	н_д	0;128
0)	0	)	н_д	0;129
- _	-	_	US	0;130
= +	=	+	н_д	0;131
Q	q	Q	DC1	0;16
W	w	W	ETB	0;17
E	e	E	ENQ	0;18
R	r	R	DC2	0;19
T	t	T	DC4	0;20
Y	y	Y	EM	0;21
U	u	U	NAK	0;22
I	i	I	HT (TAB)	0;23
O	o	O	SI	0;24
P	p	P	DLE	0;25
[{	[	{	ESC	(+)0;26
] }	]	}	GS	(+)0;27
A	a	A	SOH	0;30
S	s	S	DC3	0;31
D	d	D	EOT	0;32
F	f	F	ACK	0;33
G	g	G	BEL	0;34
H	h	H	BS	0;35
J	j	J	LF	0;36
K	k	K	VT	0;37
L	l	L	FF	0;38
; :	;	:	н_д	(+)0;39
' "	'	"	н_д	(+)0;40
` ~	`	~	н_д	(+)0;41
\ |	\	|	FS	(+)0;43
Z	z	Z	SUB	0;44
X	x	X	CAN	0;45
C	c	C	ETX	0;46
V	v	V	SYN	0;47
B	b	B	STX	0;48
N	n	N	SO	0;49
M	m	M	CR	0;50
, <	,	<	н_д	(+)0;51
. >	.	>	н_д	(+)0;52
/ ?	/	?	н_д	(+)0;53
Space	SP	SP	SP	SP

Таблица 5.14 (окончание)

1	2	3	4	5
Служебные:				
Esc	ESC	ESC	ESC	(+)0;1
BS	BS	BS	DEL	(+)0;14
Tab	HT (TAB)	0;15	(+)0;148	(+)0;165
Enter	CR	CR	LF	(+)0;28
Ctrl	н_д	н_д	н_д	н_д
Shift	н_д	н_д	н_д	н_д
(-)* PrtSc	(-)*	спец	0;144	н_д
(+)PrtSc	(+)спец	спец (как без Shift)	0;144	н_д
Alt	н_д	н_д	н_д	н_д
Caps Lock	н_д	н_д	н_д	н_д
Num Lock	н_д	н_д	спец	н_д
Scroll Lock	н_д	н_д	(-)спец	н_д
(+)Pause	(+)спец	(+)спец (как без Shift)	(+)спец	(+)спец (как без Alt)
Функциональные:				
F1	0;59	0;84	0;94	0;104
F2	0;60	0;85	0;95	0;105
F3	0;61	0;86	0;96	0;106
F4	0;62	0;87	0;97	0;107
F5	0;63	0;88	0;98	0;108
F6	0;64	0;89	0;99	0;109
F7	0;65	0;90	0;100	0;110
F8	0;66	0;91	0;101	0;111
F9	0;67	0;92	0;102	0;112
F10	0;68	0;93	0;103	0;113
(+)F11	(+)0;133	(+)0;135	(+)0;137	(+)0;139
(+)F12	(+)0;134	(+)0;136	(+)0;138	(+)0;140
Малой цифровой клавиатуры:				
Home 7	0;71	7	0;119	н_д
↑ 8	0;72	8	(+)0;141	н_д
PgUp 9	0;73	9	0;132	н_д
-	-	-	(+)0;142	(+)0;74
← 4	0;75	4	0;115	н_д
5	н_д	5	(+)0;143	н_д
→ 6	0;77	6	0;116	н_д
+	+	+	(+)0;144	(+)0;78
End 1	0;79	1	0;117	н_д
↓ 2	0;80	2	(+)0;145	н_д
PgDn 3	0;81	3	0;118	н_д
Ins 0	0;82	0	(+)0;146	н_д
Del .	0;83	.	(+)0;147	спец
Sys Req	н_д	н_д	н_д	н_д
(+)*	(+)*	(+)*	(+)0;150	(+)0;55
(+)/	(+)/	(+)/	(+)0;149	(+)0;164
(+)Enter	(+)CR	(+)CR	(+)LF	(+)0;28
Управления курсором:				
(+)Ins	(+)0;82	(+)0;82	(+)0;146	(+)0;162
(+)Del	(+)0;83	(+)0;83	(+)0;147	(+)0;163
(+)Home	(+)0;71	(+)0;71	(+)0;119	(+)0;151
(+)End	(+)0;79	(+)0;79	(+)0;117	(+)0;159
(+)PgUp	(+)0;73	(+)0;73	(+)0;132	(+)0;153
(+)PgDn	(+)0;81	(+)0;81	(+)0;118	(+)0;161
(+)↑	(+)0;72	(+)0;72	(+)0;141	(+)0;152
(+)←	(+)0;75	(+)0;75	(+)0;115	(+)0;155
(+)→	(+)0;77	(+)0;77	(+)0;116	(+)0;157
(+)↓	(+)0;80	(+)0;80	(+)0;145	(+)0;160

5.4.3. Ввод символов и командной строки с клавиатуры

Ввести с клавиатуры символ расширенного кода ASCII в выполняемую в данный момент программу можно одним из следующих способов:

1) путем нажатия соответствующей клавиши пишущей машинки на клавиатуре, находящейся на нижнем или верхнем регистре, либо клавиши на малой цифровой клавиатуре, когда она установлена в режим ввода чисел (Caps Lock включен), либо служебной клавиши (см. табл. 5.14);

2) путем нажатия комбинации Ctrl-*кл*, соответствующей символу, где *кл* является клавишей пишущей машинки или определенной служебной клавишей (см. табл. 5.14);

3) путем нажатия клавиши Alt, набора расширенного кода ASCII в десятичной системе счисления на малой цифровой клавиатуре и последующего отпускания клавиши Alt.

Способы ввода символов перечислены в порядке увеличения универсальности и одновременно усложнения работы пользователя. Исключение из этого правила состоит в том, что второй способ нельзя считать универсальнее первого.

Первым способом могут быть введены все символы пишущей машинки и четыре управляющих символа, а именно: ESC, BS, HT (TAB) и CR. Если к DOS подключен соответствующий драйвер клавиатуры, то можно ввести и другие символы из расширенного набора ASCII, в частности, символы кириллицы или псевдографические символы. Клавиатуры и драйверы клавиатур отечественных ПЭВМ обеспечивают ввод букв кириллицы без каких-либо усилий со стороны пользователя.

Вторым способом можно ввести любой управляющий символ ASCII, за исключением NUL. Комбинация Ctrl-*кл*, где *кл* — одна из клавиш A — Z, [, \,], ^ и —, обеспечивает ввод символа, код ASCII которого меньше кода ASCII символа *кл* на 40H. Например, код ASCII символа G равен 47H. Тогда путем нажатия комбинации Ctrl-G будет введен символ с кодом 47H — 40H = 07H, т.е. символ BEL (звонок). Комбинация Ctrl-BS вводит управляющий символ ASCII DEL, а Ctrl-Enter — символ ASCII LF, хотя последний можно задать и как Ctrl-J. Эти комбинации (Ctrl-BS и Ctrl-Enter), а также комбинация Ctrl-@ не подчиняются приведенному правилу. Так, Ctrl-@ выдает расширенный код клавиши 0;3.

Третий способ обеспечивает ввод любого ASCII-символа, за исключением NUL, путем явного задания его кода в десятичной системе счисления. Чтобы драйвер распознал такой способ ввода символа, код последнего набирается на фоне нажатой клавиши Alt. Например, если требуется ввести символ ‖, то нужно нажать клавишу Alt и, удерживая ее, на малой цифровой клавиатуре набрать число 186 (т.е. BAH), после чего отпустить клавишу Alt. Вслед за этим на экране дисплея появится требуемый символ, если отображение пользовательского ввода *(эхо-отображение)* обеспечивается. Такой способ ввода символов мы будем обозначать через Alt-{*n*}, где *n* — десятичный код символа.

Код введенного одним из перечисленных способов символа заносится в буфер клавиатуры, из которого выполняемая программа, используя средства DOS, в состоянии его прочитать. Поэтому, чтобы ввести символ, нет необходимости завершать это действие нажатием клавиши Enter. Однако часто программы, в том числе DOS, требуют ввода строки символов, т.е. не продолжают свою работу, пока в буфере клавиатуры не появится маркер конца строки — символ CR. В таких случаях ввод символа или последовательности символов должен завершаться нажатием клавиши Enter.

Каждый введенный с клавиатуры символ, как правило, немедленно отображается на экране дисплея в позиции курсора. Однако имеется возможность специфицировать в программе ввод символов без эхо-отображения на экране.

Пользователю и программисту часто приходится иметь дело с задействованными в ПЭВМ управляющими символами ASCII, которые представлены в табл. 5.15, где приведены обозначение каждого символа, его графическое изображение, код, назначение и различные варианты ввода.

Многие программные продукты, в том числе и DOS, реагируют на приведенные управляющие символы по-особому. Поэтому отнюдь не всегда и не все символы можно увидеть в виде изображения на экране после их ввода с клавиатуры. Чаще эти символы наблюдаются при копировании текстового файла на экран дисплея. Сделанные замечания относительно особой роли управляющих символов распространяются и на принтер. Вместе с тем некоторыми управляющими символами обычно можно пользоваться в качестве псевдографических (см. табл. 5.16). Тем не менее при записи таких символов в текстовый файл можно столкнуться с определенными трудностями. Если таковые возникнут, то воспользуйтесь встроенным редактором оболочки Norton Commander или даже шестнадцатеричным редактором утилиты Disk Editor. Следует иметь в виду, что внешние драйверы принтеров, поддерживающие кириллицу, зачастую ликвидируют возможность псевдографического использования некоторых управляющих символов. При возникновении таких ситуаций проблему можно решить путем преобразования текстового файла в графический формат (например, PCX) и последующей его печати на принтере в графическом режиме. В данном случае символы печатаются по точкам и поэтому никакого управляющего воздействия на принтер не оказывают.

Сведения, достаточные для ввода псевдографических символов расширенного кода ASCII, были уже представлены в табл. 5.8. При этом (за исключением стрелок — см. дополнительно табл. 5.16) возможно использование только комбинации Alt-{*n*}, где *n* — код символа.

Ввод командной строки в ответ на приглашение DOS осуществляется следующим образом. Пользователь на клавиатуре последовательно набирает символы командной строки. Обычно каждый напечатанный символ отображается на экране дисплея в позиции курсора, а сам курсор после этого смещается на одну позицию вправо. Поэтому на экране можно видеть уже набранную часть командной строки. Целесообразно поглядывать на экран с целью своевременного обнаружения ошибок. Если выяснилось, что набранную часть командной строки требуется отредактировать, то это можно сделать одним из следующих способов:

1) нажать клавишу Esc, в результате чего ввод всей командной строки будет отменен и можно будет начать набор командной строки заново;

Таблица 5.15

Задействованные в ПЭВМ управляющие символы ASCII

Обозначение	Графическое изображение	Код		Назначение	Варианты ввода с клавиатуры
		16-я	10-я		
NUL (пустой символ)	нет	00	0	признак конца символьной строки	нет
BEL (звонок)	•	07	7	выдача звукового сигнала при выводе на экран дисплея или принтер	Ctrl-G, Alt-{7}
BS (возврат на одну позицию)	◘	08	8	стирание введенного последним символа	BS, Ctrl-H, Alt-{8}
HT (TAB) (горизонтальная табуляция)	○	09	9	перемещение курсора или печатающей головки вправо до очередной позиции табуляции (при выводе на экран дисплея или принтер)	Tab, Ctrl-I, Alt-{9}
LF (перевод строки)	◙	0A	10	совместно с предшествующим CR - признак конца строки текстового файла; переход к текущей позиции следующей строки при выводе на экран или принтер	Ctrl--Enter, Ctrl-J, Alt-{10}
FF (перевод формата)	♀	0C	12	признак конца страницы текстового файла; при выводе на принтер - прогон бумаги до начала следующей страницы	Ctrl-L, Alt-{12}
CR (возврат каретки)	♪	0D	13	признак конца строки текстового файла; переход к первой позиции следующей строки при выводе на экран или текущей строки при выводе на принтер	Enter, Ctrl-M, Alt-{13}
SUB (подстановка)	→	1A	26	признак конца текстового файла; завершение ввода текстового файла с клавиатуры	Ctrl-Z, Alt-{26}
ESC (переход)	←	1B	27	выход из подменю или программы; отмена каких-либо действий; признак начала управляющей последовательности (Escape-последовательности)	Esc, Ctrl-[, Alt-{27}

2) нажать клавишу BS, вследствие чего последний набранный символ командной строки будет стерт, курсор сместится на одну позицию влево, а ввод строки можно будет продолжить.

Второй способ можно применять неоднократно до тех пор, пока все набранные символы командной строки не будут удалены. Пользователь выбирает тот или иной способ редактирования по своему усмотрению в зависимости от того, в какой части строки обнаружена ошибка (в начале или в конце), а также от того, нужно ли вообще вводить данную командную строку (возможно, пользователь допустил ошибку на этапе планирования своих действий).

Для разделения элементов командной строки обычно используется символ SP (пробел), вводимый путем нажатия клавиши Space. Однако можно использовать и символ TAB (клавиша Tab).

Ввод командной строки завершается нажатием клавиши Enter. После этого она начинает обрабатываться КП DOS. Если введена ошибочная командная строка, то пользователь ничего не сможет сделать до появления на экране сообщения

Bad command or file name
(Ошибочная команда или ошибочное имя файла)

При получении такого сообщения можно набрать и ввести корректную командную строку заново. Если она слишком длинная, то работа пользователя замедляется. Для устранения этого недостатка DOS сохраняет введенную последней командную строку в специальном буфере, содержимое или часть содержимого которого может быть извлечено, отредактировано и повторно введено. Для выполнения этих действий используются функциональные клавиши. Более подробно на этом мы останавливаться не будем, так как имеются специальные внешние драйверы клавиатуры (например, DOSEDIT.COM), которые обладают более богатыми возможностями. Такой драйвер хранит в стеке (в обратном порядке) несколько введенных последними командных строк, каждую из которых можно извлечь на экран дисплея путем последовательных нажатий клавиши ↑, после этого, возможно, отредактировать с использованием клавиш управления курсором на малой

Таблица 5.16

Управляющие символы ASCII, допустимые в качестве псевдографики

Обозначение	Графическое изображение	Код 16-й	Код 10-й	Отображение при копировании на CON	Печать на принтере	Варианты ввода с клавиатуры
SOH	☺	01	1	есть	есть	Ctrl-A, Alt-(1)
STX	☻	02	2	есть	есть	Ctrl-B, Alt-(2)
ETX	♥	03	3	есть	есть	Ctrl-C, Alt-(3)
EOT	♦	04	4	есть	есть	Ctrl-D, Alt-(4)
ENQ	♣	05	5	есть	есть	Ctrl-E, Alt-(5)
ACK	♠	06	6	есть	есть	Ctrl-F, Alt-(6)
SO	•	0E	14	есть	нет	Ctrl-N, Alt-(14)
SI	☼	0F	15	есть	нет	Ctrl-O, Alt-(15)
DLE	►	10	16	есть	есть	Ctrl-P, Alt-(16)
DC1	◄	11	17	есть	есть	Ctrl-Q, Alt-(17)
DC2	↕	12	18	есть	нет	Ctrl-R, Alt-(18)
DC3	‼	13	19	есть	нет	Ctrl-S, Alt-(19)
DC4	¶	14	20	есть	нет	Ctrl-T, Alt-(20)
NAK	§	15	21	есть	есть	Ctrl-U, Alt-(21)
SYN	▬	16	22	есть	есть	Ctrl-V, Alt-(22)
ETB	↨	17	23	есть	есть	Ctrl-W, Alt-(23)
CAN	↑	18	24	есть	нет	Ctrl-X, Alt-(24)
EM	↓	19	25	есть	есть	Ctrl-Y, Alt-(25)
FS	∟	1C	28	есть	есть	Ctrl-\, Alt-(28)
GS	↔	1D	29	есть	есть	Ctrl-], Alt-(29)
RS	▲	1E	30	есть	есть	Ctrl-^, Alt-(30)
US	▼	1F	31	есть	есть	Ctrl-_, Alt-(31)

цифровой клавиатуре и клавиши BS, а затем повторно ввести нажатием клавиши Enter. Аналогичные возможности предоставляют и оболочки DOS, в частности Norton Commander.

Отменить выполнение ошибочно введенной, но корректной командной строки можно путем нажатия комбинации клавиш Ctrl-Break.

Обычно командная строка не должна занимать более одной строки на экране дисплея. Зачастую пользователю это ограничение не мешает. Если же все-таки требуется ввести слишком длинную командную строку, то для перехода на следующую физическую строку без выдачи кода CR следует ввести код LF путем нажатия Ctrl-Enter или Ctrl-J (но при этом не следует забывать о том, что символ LF не является разделителем полей командной строки).

5.4.4. Специальные клавиши DOS

DOS использует некоторые клавиши и комбинации клавиш для реализации определенных функций. Это достигается путем анализа кодов, записываемых в буфер клавиатуры, и выполнения той или иной последовательности действий в зависимости от кода. Реакция на ряд клавиш и их комбинаций определена в драйвере клавиатуры, и никакие коды при их нажатии в буфер клавиатуры не заносятся (об этом мы уже говорили).

Специальной клавишей DOS будем называть клавишу или комбинацию клавиш, нажатие которой (которых) приводит к выполнению управляющих действий (отличных от обычного ввода соответствующего кода), независимо от того, каким образом эти действия реализованы.

Многие из специальных клавиш DOS уже упоминались раньше, причем неоднократно. В этом пункте мы приведем исчерпывающие сведения обо всех таких клавишах, за исключением клавиш редактирования буфера командной строки.

Специальные клавиши делятся на две группы:

1) общие специальные клавиши;
2) специальные клавиши редактирования командной строки.

К *общим (управляющим) специальным клавишам* относятся следующие:

Ctrl-Alt-Del— вызывает рестарт (перезагрузку) DOS без тестирования оборудования ПЭВМ. Ее желательно вводить только в ответ на приглашение DOS;

Pause — приводит к приостановке выполнения программы, в частности, вывода информации на экран дисплея, до нажатия произвольной клавиши. Для достижения этого же эффекта можно использовать комбинации клавиш Ctrl-Num Lock или Ctrl-S, но последняя действует не всегда;

Ctrl-Break — обеспечивает принудительное завершение выполнения программы или команды DOS. На 83- и 84-клавишных клавиатурах следует использовать Ctrl-Scroll Lock, а на 101-клавишной — Ctrl-Pause. В большинстве случаев на любой клавиатуре срабатывает и Ctrl-C;

Shift-PrtSc — посылает информацию, изображенную на экране дисплея, на принтер для печати. На 101-клавишной клавиатуре клавишу Shift задействовать не обязательно;

Ctrl-PrtSc — включает/выключает дублирование содержимого экрана на принтере (используется для получения твердой копии протокола диалога: при включении дублирования вся выводимая на экран информация будет одновременно печататься на принтере). Можно использовать Ctrl-P;

Ctrl-Alt-F1 — переключает клавиатуру на американский регистр;

Ctrl-Alt-F2 — переключает клавиатуру на национальный регистр;

F6 — вводит символ SUB (маркер конца текстового файла) для завершения набора файла на клавиатуре. Мы не говорили об этом раньше, так как клавиша F6 некоторыми программными продуктами перепрограммируется, а в DOS используется именно для достижения указанной цели.

Переключение клавиатуры на американский или национальный регистр поддерживается средствами DOS после выполнения команды KEYB.

К *специальным клавишам редактирования командной строки* относятся следующие:

Esc — аннулирует текущую (вводимую) командную строку, но набранная ее часть остается в буфере командной строки. Взамен можно использовать Ctrl-[;

BS — удаляет из командной строки последний введенный символ. Можно использовать Ctrl-H;

Enter — завершает ввод командной строки;

Ctrl-Enter — осуществляет перевод курсора в начало следующей строки экрана для продолжения набора длинной командной строки. Можно использовать Ctrl-J.

5.5. Общие сведения о командах DOS

Материал данного подраздела является вводным для последующего текста. Здесь рассматривается классификация команд DOS, перечисляются используемые в дальнейшем соглашения и обозначения, описывается общий формат команды, а также приводится структура ее описания.

5.5.1. Классификация команд DOS

Команды DOS обеспечивают взаимодействие пользователя с системой. Совокупность этих команд с учетом правил их записи и выполняемых ими функций составляет **командный язык DOS**, на котором основан пользовательский интерфейс системы.

По функциональному назначению и использованию команды DOS делятся на пять групп (см. рис. 5.13):

1) общие команды;
2) инструментальные команды;
3) фильтры;
4) команды для командных файлов;
5) команды конфигурирования системы.

Общие и *инструментальные команды*, а также *команды-фильтры* вводятся командной строкой с клавиатуры в ответ на приглашение DOS или выполняются из командных файлов.

Первые из них обеспечивают выполнение DOS различных действий, реконфигурирование (перенастройку) системы в процессе работы и выдачу пользователю информационных сообщений. Эти команды делятся на семь пересекающихся подгрупп:

1) *команды манипулирования дисками*, объектом действий которых выступает МД в целом;
2) *команды манипулирования каталогами*, обеспечивающие выполнение различных действий с каталогами файловой структуры;
3) *команды манипулирования файлами*, в качестве объектов действий которых выступают файлы;
4) *команды управления посимвольными устройствами*, воздействующие на клавиатуру, дисплей и принтер;
5) *команды реконфигурирования системы*, обеспечивающие перенастройку DOS в процессе работы;
6) *команды управления системой*, объектом действий которых является сама DOS;
7) *информационные команды*, выдающие пользователю сведения о состоянии ресурсов DOS, конфигурации и режимах работы оборудования, о файловой структуре и т.п.

В каждой из этих подгрупп на рис. 5.13 имя команды подчеркнуто, если она в дальнейшем будет рассматриваться в рамках данной подгруппы.

Инструментальные команды выполняют функции инструментальных систем (см. подраздел 4.4).

Фильтры обеспечивают преобразование входного потока в выходной по определенному алгоритму.

Многие внешние команды DOS генерируют код возврата, который можно анализировать в командных файлах или родительских программах.

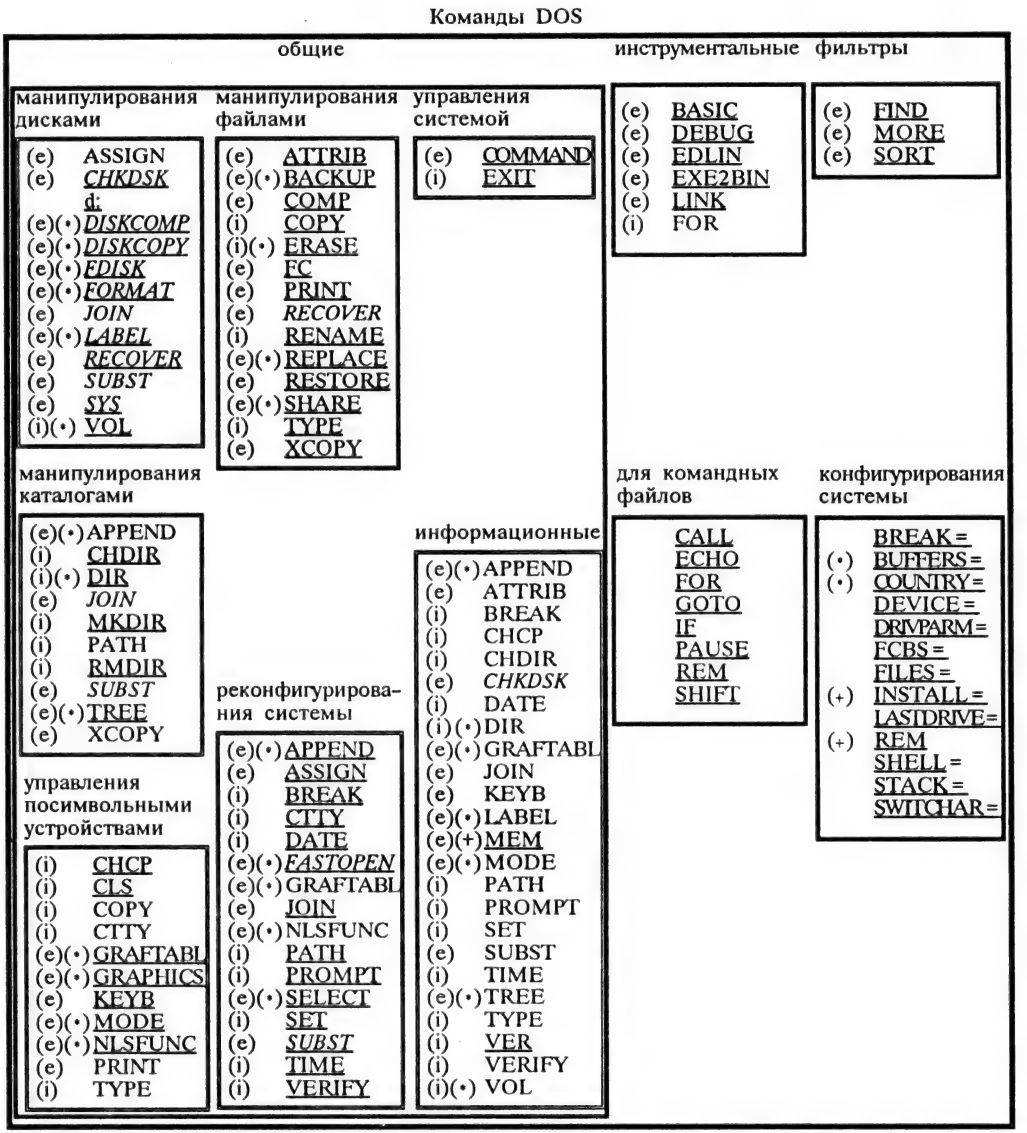


Рис. 5.13. Классификация команд DOS

Команды для командных файлов могут использоваться главным образом только в одноименных файлах с целью управления их выполнением. Наряду с этими командами в командных файлах допустимы все команды, которые можно ввести и с клавиатуры, что уже отмечалось.

Команды конфигурирования системы используются только в файле CONFIG.SYS для настройки DOS во время загрузки, если принимаемые по умолчанию значения пользователя не устраивают.

Новые для DOS 4.0 команды отмечены признаком (+), а усовершенствованные — (*). Внешние команды снабжены пометкой (e), а внутренние — (i). Курсивом же набраны команды, которые не могут использоваться в сети ПЭВМ для воздействия на другие ее узлы (среди них в основном команды, приводящие к изменению информации на внешних носителях, что для совместно используемых ресурсов недопустимо). Остальные команды могут применяться в сети без каких-либо ограничений.

5.5.2. Соглашения, используемые при описании команд

При описании синтаксиса команд прописными буквами будут представляться элементы командной строки, которые должны использоваться точно в том виде, как они записаны. Понятия,

которые должны замещаться их экземплярами (конкретизациями), будут выделяться курсивом. Дополнительно к этому используются следующие соглашения и обозначения:

[a] — указывает на необязательность элемента *a*;

[a1|a2|...aN] — свидетельствует о том, что должен быть только один из элементов *ai*, но может и не быть ни одного элемента;

{a1|a2|...aN} — указывает на то, что обязательно должен присутствовать один из элементов *ai* (но не более одного);

◁a1 a2 ... aN ▷ — группирует (но только при описании) элементы *a1, a2, ..., aN* [символы ◁ и ▷ играют роль скобок аналогично символам (и) в арифметических выражениях];

... — свидетельствует о том, что предыдущий элемент может неоднократно повторяться; для однозначной идентификации группового элемента он должен быть (но только при описании) заключен в одну из пар скобок;

◄k ► — обозначает нажатие клавиши *k*.

С использованием принятых соглашений типовая структура команды DOS представляется следующим образом:

имя_команды [*аргумент*]... [*переключатель*]...

Первым элементом командной строки является **имя команды**, полный перечень которых уже приведен на рис. 5.13. За именем команды могут следовать **аргументы** (спецификации файлов, каталогов и т.п.), обычно отделяемые от него и друг от друга, по крайней мере, одним пробелом и используемые для указания объектов, над которыми требуется выполнить те или иные действия. Командную строку могут завершать **переключатели (флаги)**, которые уточняют или модифицируют действие команды. Каждый переключатель начинается со слэша (/) и поэтому не обязательно должен отделяться от предыдущего элемента пробелами.

Аргументы команд, как правило, являются позиционными и поэтому должны следовать строго в указанном при описании синтаксиса команды порядке. Переключатели распознаются не по месту, а по виду (ключу). Поэтому порядок их следования значения не имеет.

Некоторые команды, в особенности команды для командных файлов и команды конфигурирования системы, могут иметь отличающуюся от описанной структуру.

В дальнейшем без каких-либо дополнительных пояснений будут использоваться следующие обозначения:

d — для имени логического привода (накопителя, дисковода);

pattern — для спецификации шаблона файла (полной или неполной);

file — для спецификации файла (полной или неполной), которая рассматривается как частный случай спецификации шаблона файла;

dir — для спецификации каталога (полной или неполной);

name — для образца имени файла (не составного!);

ext — для образца расширения файла;

compname — для образца составного имени файла, т.е. для *name*[.*ext*] или *name*.;

arglist — для списка аргументов (последовательность разделенных пробелами аргументов);

process — для строки, обеспечивающей вызов команды или исполняемого файла на выполнение (имя команды или спецификация исполняемого файла с аргументами и переключателями);

string — для символьной строки, включающей буквы, цифры, пробелы и другие неуправляющие символы;

k, l, m, n и т.п. — для числа.

Большинство упомянутых понятий введены нами в п. 5.2.2. Если в командной строке используется несколько одинаковых понятий, то их имена при описании команд могут нумероваться (например, *d1, d2*), чтобы на них было удобно ссылаться в тексте.

Синтаксис командного языка DOS не безупречен. Так, в частности, *d*: в различных контекстах обозначает дисковод или текущий каталог на этом дисководе. Поэтому при описании команд возникают определенные трудности. Мы будем употреблять всегда для привода обозначение *d*, а для каталога — *dir*. В случае необходимости указания текущего каталога на дисководе *d* всегда будет предполагаться *d*:. . Поэтому запись *dir**compname* всегда оказывается корректной (в противном случае в качестве ее конкретизации допускалась бы строка *d*:*compname*, которая означала бы ссылку на файл *compname* в корневом, а не в текущем каталоге диска в приводе *d*). Но всегда нужно иметь в виду, что запись *d*: при указании каталога является сокращенной формой строки *d*:. .

В командном языке DOS наблюдается также некоторая непоследовательность в именовании переключателей (одни и те же переключатели в разных командах могут обозначаться по-разному, а разные переключатели — одинаково).

И наконец, синтаксис некоторых команд DOS (например, MODE) не укладывается в рамки общего формата.

5.5.3. Структура описания команды

Команды DOS будут описываться нами в следующей последовательности:

1) **назначение**, определяющее функции команды;

2) **тип**, где будет указываться, является ли команда внешней или внутренней, а также является ли она несетевой (не работающей в сети ПЭВМ);

3) **синтаксис** (формат команды);

4) **комментарии**, где будут описываться действия команды *(семантика)*, в том числе аргументы и допустимые переключатели, а также обсуждаться порядок ее использования;

5) **замечания**, в которых будут отмечаться важные дополнительные моменты, связанные с использованием команды;

6) **примеры** для иллюстрации задания и использования команды;

7) **DOS 4.0**, где будут указываться особенности задания и использования команды в DOS 4.0 по сравнению с DOS 3.3, если таковые имеются.

5.6. Общие команды DOS

В этом подразделе описываются следующие команды DOS (в скобках приводятся их синонимы):

APPEND — устанавливает и отображает маршруты поиска файлов с данными;

ASSIGN — подменяет один дисковод другим;

ATTRIB — изменяет и отображает атрибуты файлов;

BACKUI — осуществляет резервное копирование файлов с одного диска на другой;

BREAK — устанавливает и отображает режим контроля нажатия комбинации клавиш Ctrl-Break;

CHCP — изменяет и отображает текущую (активную) кодовую страницу для КП DOS;

CHDIR (CD) — изменяет и отображает текущий каталог;

CHKDSK — проверяет диск на предмет целостности файловой структуры и корректирует ошибки, а также отображает статус диска и памяти;

CLS — очищает экран дисплея;

COMMAND — запускает КП DOS;

COMP — сравнивает содержимое файлов;

COPY — копирует файлы;

CTTY — изменяет стандартное УВВ DOS;

d: — изменяет текущий привод;

DATE — устанавливает и отображает дату;

DIR — отображает содержимое каталога или его подмножество;

DISKCOMP — сравнивает содержимое двух дискет;

DISKCOPY — копирует содержимое дискеты;

ERASE (DEL) — удаляет файлы;

EXIT — осуществляет выход из КП и возврат на предыдущий уровень;

FASTOPEN — ускоряет открытие файлов и каталогов;

FC — сравнивает содержимое файлов и отображает различия между ними;

FDISK — конфигурирует жесткий диск для DOS;

FORMAT — форматирует диск (подготавливает его к использованию);

GRAFTABL — обеспечивает отображение расширения ASCII в графических режимах адаптера CGA;

GRAPHICS — подготавливает принтер для печати в графическом режиме;

JOIN	— логически подсоединяет дисковод к каталогу диска на другом накопителе и отображает текущие подсоединения;
KEYB	— настраивает клавиатуру на национальный алфавит и отображает двухбуквенный код клавиатуры;
LABEL	— устанавливает и отображает метку диска;
MEM	— отображает информацию о распределении ОЗУ;
MKDIR (MD)	— создает каталог;
MODE	— отображает статус и устанавливает режимы работы посимвольных устройств;
NLSFUNC	— загружает зависящую от страны информацию;
PATH	— устанавливает и отображает маршруты поиска исполняемых файлов;
PRINT	— печатает файлы на принтере;
PROMPT	— изменяет приглашение DOS;
RECOVER	— восстанавливает информацию на дефектном диске;
RENAME (REN)	— переименовывает файлы;
REPLACE	— заменяет файлы их новыми версиями;
RESTORE	— восстанавливает зарезервированные посредством команды BACKUP файлы;
RMDIR (RD)	— удаляет каталог;
SELECT	— устанавливает DOS на новый диск и конфигурирует ее;
SET	— устанавливает значение глобальной переменной в окружении DOS или отображает его;
SHARE	— устанавливает многопользовательский режим использования файлов в сети;
SUBST	— обозначает маршрут именем дисковода и отображает введенные обозначения;
SYS	— переносит системные файлы DOS на заданный диск;
TIME	— устанавливает и отображает время;
TREE	— отображает файловую структуру диска;
TYPE	— отображает содержимое файла;
VER	— отображает номер версии DOS;
VERIFY	— устанавливает и отображает режим контроля правильности записи информации на диски;
VOL	— отображает метку диска;
XCOPY	— копирует файлы и подкаталоги.

В приведенном списке команд кратко указано и их назначение. Под *отображением* здесь понимается вывод на экран дисплея, а маршрут трактуется в несколько расширенном смысле и включает привод.

Сообщения, выводимые командами, могут несколько различаться в зависимости от версии DOS и ее производителя.

5.6.1. Команды манипулирования дисками

Команда d:

Назначение: изменение (выбор) текущего привода.
Тип: внутренняя.
Синтаксис:

d:

Комментарии. Действие команды состоит в смене текущего дисковода. После ее выполнения текущим становится указанный накопитель *d*, что обычно фиксируется в приглашении DOS.

Пример фрагмента диалога с DOS:

```
C>A:◄Enter►
A>_
```

Здесь показана вся информация, отображаемая на экране дисплея, включая приглашение DOS.

Команда FORMAT

Назначение: форматирование диска.
Тип: внешняя, несетевая.
Синтаксис:

FORMAT *d*: [/1] [/4] [/8] [/N:*n*] [/T:*t*] [/V[:*метка*]] [/S|/B]

Комментарии. Команда FORMAT подготавливает диск в указанном накопителе к использованию в среде DOS. Она может применяться как для гибких, так и для логических дисков на винчестере. В последнем случае необходимо предварительно выполнить команду FDISK.

При форматировании (инициализации) диска производятся следующие действия:

1) разбиение дорожек диска на секторы (только для дискет);

2) проверка всех секторов на предмет возможности записи и считывания информации (проверка записи — только для дискет); дефектные секторы при этом соответствующим образом отмечаются и впоследствии использоваться для размещения информации не будут;

3) формирование на диске системной области, а именно:

— запись стартового сектора с SB;

— создание двух копий таблицы размещения файлов (FAT — File Allocation Table) и запись в них необходимой информации;

— создание корневого каталога.

Дополнительно к этому (в зависимости от заданных переключателей) на диск могут переноситься системные файлы DOS, резервироваться место для них, копироваться любые другие заданные файлы, а также записываться метка.

Допустимые форматы (число сторон, цилиндров и секторов на дорожке) инициализируемой дискеты зависят от типа логического привода, который определяется возможностями реального накопителя и обслуживающего его драйвера. Например, если НГМД обеспечивает размещение на диске 80 цилиндров, а драйвер служит для обеспечения работы с 40-цилиндровыми дискетами, то указание в команде FORMAT имени логического привода, связанного с этим драйвером, приведет к инициализации 40 цилиндров. К DOS может быть одновременно подключен и драйвер на 80 цилиндров. В этом случае указание в команде FORMAT соответствующего имени логического привода обеспечит инициализацию 80 цилиндров. Заметим, что для достижения указанных целей используется один и тот же физический накопитель, но под разными именами. Изменить формат инициализируемой дискеты (но только *в сторону уменьшения* результирующей емкости) можно путем задания соответствующих переключателей. Однако возможность инициализации в соответствии с каким-либо форматом автоматически не означает возможности использования (считывания и записи) в системе дискет этого формата. Последнее зависит от «способностей» драйвера.

Некоторые внешние драйверы требуют использования специальных средств и методов форматирования.

Ряд типов ПЭВМ, не полностью совместимых с изделиями фирмы IBM на уровне BIOS, могут накладывать другие ограничения на возможности форматирования дискет с меньшей, чем стандартная, емкостью.

Жесткие же диски имеют свой формат, который не может быть изменен переключателями команды FORMAT.

Команда FORMAT допускает следующие переключатели:

/1 — форматировать одну сторону дискеты (для ее последующего использования в одностороннем дисководе);

/4 — форматировать двухсторонний 133-мм 360-Кбайт гибкий диск в дисководе на 1,2 Мбайт. Важно подчеркнуть, что некоторые 360-Кбайт накопители не могут надежно читать проинициализированные таким образом дискеты из-за различной ширины дорожки;

/8 — форматировать 8 секторов на дорожке дискеты (стандартный формат дискеты для DOS до 2.0 и устаревших накопителей);

/S — скопировать после форматирования файлы DOS с MP BIOS, БМ DOS и КП с системного диска, установленного в текущий дисковод, на проинициализированный диск. Если системный диск в текущем приводе отсутствует, пользователь получит сообщение с требованием установить в него системный диск (или в привод A в случае, когда текущим является дисковод с несъемным носителем, т.е. НЖМД). В использовании этого переключателя состоит один из способов создания нового системного диска;

/B — зарезервировать после форматирования диска пространство для размещения файлов с MP BIOS и БМ DOS, которые могут быть впоследствии перенесены командой SYS (создаются требуемые элементы корневого каталога и выделяется область дисковой памяти необходимого размера, но сами системные файлы не переносятся);

/T:*t* — создать на дискете *t* (40 или 80) дорожек на одной стороне (цилиндров);

/N:*n* — создать на каждой дорожке дискеты *n* (8, 9, 15 или 18) секторов;

/V[:*метка*] — записать на диск после его форматирования указанную метку. Если метка не задана, то пользователю будет выдан запрос на ее ввод с клавиатуры.

Варианты задания поддерживаемых форматов дисков для различных типов дисководов приведены в табл. 5.17. Форматы на емкость меньше 360 Кбайт устарели и в настоящее время практически не используются. Формат, обеспечиваемый 133-мм 80-дорожечными дисководами (на 720 Кбайт), непосредственно, т.е. без дополнительных программных средств, не поддерживается.

Таблица 5.17

Задание форматов дисков

Формат диска				Варианты задания переключателей
Емкость, байт	Число сторон	Диаметр, мм	Тип привода	
160K	1	133	DS/DD	/1 /8
			DS/HD	/1 /4 /8
180K	1	133	DS/DD	/1
			DS/HD	/1 /4
320K	2	133	DS/DD	/8 или /N:8
			DS/QD	/N:8 /T:40
			DS/HD	/N:8 /T:40 или /8
360K	2	133	DS/DD	не требуются
			DS/QD	/T:40 /N:9 или /T:40
			DS/HD	/T:40 /N:9 или /4
720K	2	133	DS/QD	не требуются
1,2M	2	133	DS/HD	не требуются
720K	2	89	DS/QD	не требуются
			DS/HD	/T:80 /N:9 или /N:9
1,44M	2	89	DS/HD	не требуются
жесткий диск			HXMД	не требуются

Переключатели /B, /V и /S могут использоваться для любых дисков и приводов, а остальные — только для гибких.

После ввода команды FORMAT для гибкого диска на экран дисплея выдается сообщение

Insert new diskette for drive *d*:
and strike ENTER when ready
(Установите новую дискету в привод *d*:
и затем нажмите клавишу Enter)

Пользователь должен выполнить эти указания, если он хочет начать инициализацию дискеты, либо нажать комбинацию клавиш Ctrl-Break для отмены форматирования.

При инициализации логического диска на винчестере DOS предусматривает определенную защиту для исключения непреднамеренного уничтожения информации на нем, чем сопровождается форматирование. После выдачи команды FORMAT для логического диска на винчестере на экране дисплея появляется сообщение

WARNING, ALL DATA ON NON-REMOVABLE DISK
DRIVE *d*: WILL BE LOST!
Proceed with Format (Y/N)?
(Предупреждение: все данные на несъемном
дисководе *d*: будут потеряны!
Продолжить форматирование (Y—да/N—нет)?)

Пользователю следует нажать клавишу Y или N в соответствии с выбранным вариантом ответа, а затем — клавишу Enter.

В ходе инициализации диска утилита FORMAT выдает на экран дисплея информационное сообщение, из которого видно, какая часть диска уже отформатирована. Под заголовками Head (головка, или сторона диска) и Cylinder (цилиндр) выводятся сменяющие друг друга числа, определяющие инициализируемый в данный момент участок диска.

По окончании форматирования появляется сообщение

Format complete
(Форматирование завершено)

и указываются общий объем дискового пространства, объем дефектных секторов, размер DOS (если использовался переключатель /S), а также размер свободного, т.е. доступного для размещения файлов дискового пространства. Эта информация дается в байтах, и поэтому некоторые пользователи

делают неправильные выводы о емкости дискеты в Кбайт, забывая о том, что 1 К = 1024, а не 1000. Последним сообщением утилиты FORMAT для дискеты является

Format another (Y/N)?
(Форматировать другой (Y—да/N—нет)?)

Введя в ответ на него Y, Вы можете проинициализировать следующую дискету или повторить форматирование только что размеченной, но формат изменить в любом случае нельзя.

DOS может работать с дискетами, имеющими дефектные секторы, за исключением тех случаев, когда они размещены на дорожке 0, т.е. в самом начале логического дискового пространства. Последнее связано с тем, что на этой дорожке должна находиться системная область диска, где дефектные секторы недопустимы. При обнаружении этой ситуации утилита выдает сообщение

Track 0 bad — disk unusable
(Дорожка 0 дефектна — диск использованию не подлежит)

В этом случае пользователь может несколько раз повторить форматирование, в том числе и другими имеющимися в его распоряжении утилитами. Если эти попытки положительного результата не дали, то можно прибегнуть еще к одной мере: вскрыть защитный конверт дискеты, аккуратно изъять ее из конверта, перевернуть дискету и снова поместить в конверт, который затем заклеить. Если Вы проделали эти операции осторожно и не повредили дискету, то она нормально отформатируется, так как трек с дефектным сектором теперь будет иметь номер 1. В случае, когда ни одна из описанных мер к успеху не привела, дискету можно использовать разве что только в качестве сувенира. Заметим, что последнее сообщение может появиться и при недопустимости для дискеты того формата, в соответствии с которым Вы хотите ее проинициализировать. Если сообщение о дефектности дорожки 0 выдано для логического диска на винчестере и попытки повторного форматирования к успеху не привели, то измените разбивку жесткого диска на разделы и/или логические диски с тем, чтобы начало форматируемого логического диска не приходилось на дефектную дорожку.

Невозможность форматирования диска по какой-либо причине всегда сопровождается сообщением

Format failure
(Неудача форматирования)

Утилита FORMAT генерирует следующие коды возврата:

0 — успешное завершение операции;
3 — форматирование принудительно прекращено пользователем (по Ctrl-Break);
4 — критическая ошибка;
5 — на сообщение «Proceed with format (Y/N)?» пользователь ответил N.

Замечания:

— команда FORMAT разрушает информацию на дискете, причем эта информация не может быть восстановлена никакими средствами (перезапись обычно осуществляется кодом F6H). Поэтому использовать данную команду нужно в высшей степени внимательно и осторожно;

— при форматировании логического диска на винчестере содержимое файлов физически ничем не замещается, в результате чего в принципе возможно восстановление информации с нечаянно проинициализированного диска (см. п. 8.4.4);

— некоторые версии DOS допускают форматирование текущего диска, если опущен аргумент *d*:; несмотря на это, желательно при форматировании всегда указывать привод явно, чтобы застраховаться от форматирования не «того» диска;

— команда FORMAT перед первым использованием нового диска должна выполняться всегда. Переформатирование диска целесообразно осуществлять только для удаления всей содержащейся на нем информации и его тестирования на наличие дефектных секторов, чтобы запретить их использование (хотя для решения этих задач имеются и более совершенные системные программные продукты). DOS не содержит других средств (если не считать опасную команду RECOVER), позволяющих исключить возможность использования дефектных секторов под файловую структуру;

— при задании переключателя /S перенос файлов DOS на проинициализированный диск осуществляется не обязательно с системного диска, с которого производилась загрузка системы, а с любого системного диска в текущем приводе;

— чтобы создать системный диск, наиболее полно удовлетворяющий Вашим потребностям, после форматирования с переключателем /S нужно будет дополнительно скопировать, а возможно и создать, необходимые файлы DOS;

— утилита FORMAT DOS 3.3 содержит ошибку, в результате чего качественная дискета может быть забракована. Чтобы справиться с этой проблемой, выдайте команду DIR A: перед форматированием;

— пользователь не должен выдавать команду FORMAT для приводов, задействованных в команде ASSIGN, JOIN или SUBST, чтобы избежать катастрофических побочных эффектов, вызванных переименованием дисководов. Отметим, что команда FORMAT игнорирует присваивание имен, выполненное командой ASSIGN;

— дополнительная информация о метках дисковых томов содержится в описании команды LABEL в данном подразделе;

— если после форматирования диска число дефектных секторов оказалось большим, попытайтесь выполнить форматирование несколько раз и прекратите этот процесс только тогда, когда при серии последовательных инициализаций будут выдаваться одинаковые сообщения о емкости дефектной области. Иными словами, требуется добиться устойчивой работы диска, чтобы избежать неприятностей, поджидающих Вас на этапе его использования, когда сектор вдруг перестает читаться;

— несмотря на то, что перед командой DISKCOPY форматирование целевой дискеты принципиально не требуется, лучше его все-таки осуществить для своевременного обнаружения дефектного диска. Это отнюдь не лишняя предосторожность, так как большинство коммерческих программных продуктов, защищенных от копирования, дают возможность выполнить одну-единственную перезапись. Неудачная же попытка копирования не позволит Вам использовать и эту возможность.

Примеры:

- FORMAT A: /S — отформатировать дискету в приводе A и скопировать на нее DOS;
- FORMAT A: /V:VOL1 — отформатировать дискету в приводе A и записать на нее метку VOL1;
- FORMAT A:/4 — отформатировать двухстороннюю дискету на 360 Кбайт в приводе DS/HD;
- FORMAT A: /N:9 /T:40 — то же.

DOS 4.0. В командной строке допустим еще один переключатель — /F:*формат*, упрощающий задание требуемого формата дискеты. Возможны следующие спецификации формата:

— для односторонней 133-мм 160-Кбайт дискеты — 160, 160K или 160KB;
— для односторонней 133-мм 180-Кбайт дискеты — 180, 180K или 180KB;
— для двухсторонней 133-мм 320-Кбайт дискеты — 320, 320K или 320KB;
— для двухсторонней 133-мм 360-Кбайт дискеты — 360, 360K или 360KB;
— для 133-мм 1,2-Мбайт дискеты — 1200, 1200K, 1200KB, 1.2, 1.2M или 1.2MB;
— для 89-мм 720-Кбайт дискеты — 720, 720K или 720KB;
— для 89-мм 1,44-Мбайт дискеты — 1440, 1440K, 1440KB, 1.44, 1.44M или 1.44MB.

Действие переключателя /S дополнено следующим. Если системный диск в текущем приводе содержит в корневом каталоге текстовый файл FORMATS.TBL и новый диск форматируется на емкость, не меньшую чем 1,2 Мбайт, то на проинициализированный диск будут дополнительно скопированы все файлы DOS, перечисленные в FORMATS.TBL. В случае, когда первое условие выполнено, а второе — нет, перенос файлов, зарегистрированных в FORMATS.TBL, не производится. Автоматическое копирование требуемых файлов облегчит создание системного диска.

При форматировании диска дополнительно осуществляется запись на него уникального *серийного номера тома* (не метка тома!), который однозначно идентифицирует диск. Этот серийный номер (Serial Number) выдается утилитой FORMAT по завершении форматирования диска, а также может быть отображен командой LABEL, VOL или DIR.

При попытке инициализации логического диска на винчестере предусмотрены дополнительные меры по предотвращению непреднамеренного его форматирования. С этой целью на экран сначала выдается сообщение

Enter current Volume Label for drive
(Введите текущую метку тома для привода *d*:)

В ответ на это сообщение пользователь должен ввести правильную метку логического диска на винчестере или просто нажать клавишу Enter, если он никак не помечен. В случае же, когда жесткий диск еще ни разу не форматировался или имеет ошибочный стартовый сектор, данное сообщение не выдается.

Если пользователь задал метку логического диска неверно (возможно, и преднамеренно), то на экране дисплея появляется сообщение

Involid Volume ID Format failure
(Неверный идентификатор тома. Неудача форматирования)

и утилита FORMAT завершает свою работу. В противном случае выдается еще одно предупреждающее сообщение «WARNING, ALL DATA ...», приведенное раньше.

Пример:

- FORMAT A:/F:360 — отформатировать двухстороннюю дискету на 360 Кбайт в приводе DS/HD.

Команда SYS

Назначение: перенос системных файлов DOS на заданный диск.
Тип: внешняя, несетевая.

Синтаксис:

SYS *d*:

Комментарии. Команда SYS переносит системные файлы с MP BIOS и БМ DOS на диск в приводе *d*, делая этот диск *загружаемым* (системным). Перед выдачей команды SYS системный диск должен располагаться на текущем дисководе (точнее — требуется сделать его текущим), что совпадает с требованиями команды FORMAT /S.

Файлы DOS помещаются в корневой каталог целевого диска на первые две позиции, чтобы загрузка DOS в последующем была возможной. Поэтому для целевого диска должно выполняться одно из нижеперечисленных условий:

1) диск уже является системным;

2) диск был отформатирован с переключателем /B;

3) диск проинициализирован без переключателей /S и /B, но пока является пустым и не содержит метки, а следовательно, первые позиции корневого каталога свободны (заметим, что метка записывается в корневой каталог диска аналогично составному имени файла; вот почему ее быть не должно).

В первом случае команда SYS замещает системные файлы с целью обновления, во втором — копирует эти файлы в зарезервированное при инициализации пространство, а в третьем — также копирует их требуемым образом (диск еще пуст, и нет никаких препятствий для этого).

Замечания:

— указание привода в команде SYS обязательно;

— команда SYS не переносит на целевой диск файл с КП (COMMAND.COM). Для этого можно использовать команду COPY;

— начиная с DOS 3.3, размещение файлов с MP BIOS и БМ DOS на диске не фиксируется, вследствие чего команда SYS может применяться не только для создания нового системного диска или обновления системных файлов, но и для установки на системный диск новой версии DOS (последнее обычно приводит к фрагментации файла с БМ DOS, так как по мере развития DOS размеры системных файлов, как правило, увеличиваются и поэтому их не удается полностью разместить на месте старых файлов; однако сказанное не относится к DOS 5.0);

— файлы с MP BIOS и БМ DOS можно было бы перенести на целевой диск не командой SYS, а командой COPY, но они имеют атрибуты H и S и для команды COPY поэтому недоступны. Однако в некоторых оболочках DOS (в частности, в Norton Commander'е) можно реализовать команду SYS посредством функции копирования файлов, которая обрабатывает файлы с любыми атрибутами. Нужно только соблюсти порядок копирования двух системных файлов (первым должен копироваться файл с MP BIOS). Если же системные файлы на целевом диске уже существуют, то они просто замещаются;

— команда SYS не работает с приводами, задействованными в командах SUBST и JOIN.

Пример:

- SYS A: — скопировать файлы DOS с текущего дисковода на диск в дисководе A.

Команда LABEL

Назначение: создание, изменение, уничтожение и отображение метки тома на диске.
Тип: внешняя, несетевая.
Синтаксис:

LABEL [*d*:][*метка*]

Комментарии. *Метка тома* — это имя, которое пользователь назначает диску и которое записывается на него. Обычно метка играет только информационную роль, позволяя быстро идентифицировать диск, если он не подписан. Однако некоторые программы проверяют метку, чтобы убедиться в установке требуемого диска.

d: в команде определяет привод, содержащий обрабатываемый диск. Если этот аргумент опущен, то помечается текущий диск.

Метка тома может содержать не более 11 любых символов пишущей машинки, за исключением &, ", ^, *, ?, /, \, |, ., запятая, ;, :, +, =, (,), <, >, [и]. Допустим также и символ SP, но не TAB. Очевидно, множества символов, допустимых для именования файлов (каталогов) и спецификации меток, лишь пересекаются, не совпадая и не поглощая друг друга. Если в командной строке метка не задана, то на экран дисплея выдается следующая последовательность сообщений:

Volume in drive *d*: is *xxxxxxxxxxx*
Volume label (11 characters, ENTER for none)?
(В привод *d* установлен том *xxxxxxxxxxx*.
Метка тома (11 символов, Enter для никакой)?)

В первом сообщении указывается имеющаяся на диске метка тома. В ответ на второе сообщение следует ввести новую метку тома или просто нажать клавишу Enter, если новую метку задавать не требуется. При втором варианте ответа выдается сообщение

Delete current volume label (Y/N)?
(Удалить текущую метку тома (Y—да/N—нет)?)

Если пользователь нажмет клавишу Y, то метка будет удалена и диск окажется непомеченным. В противном случае метка тома останется неизмененной. Следовательно, если в командной строке не задать метку, на первый вопрос ответить ◄Enter►, а на второй — N, то никаких действий командой LABEL выполнено не будет, но пользователь сможет узнать текущую метку тома.

Замечания:

— создать метку тома можно командой FORMAT с переключателем /V;
— для отображения метки тома удобнее пользоваться командами DIR и VOL;
— команда LABEL не работает с приводами, задействованными в командах SUBST и JOIN.

Пример:

- LABEL A:VOL_5 — создать метку VOL_5 на диске в приводе A.

DOS 4.0. По команде LABEL дополнительно отображается серийный номер тома (Volume Serial Number), записываемый командой FORMAT.

Команда VOL

Назначение: отображение метки тома.
Тип: внутренняя.
Синтаксис:

VOL [*d*:]

Комментарии. При выполнении команды на экран дисплея выдается метка тома в приводе *d* (если он задан) или в текущем приводе (если дисковод не задан) посредством сообщения

Volume in drive *d*: is *xxxxxxxxxx*
(В привод *d* установлен том *xxxxxxxxxx*)

Если диск не имеет метки, то появляется сообщение

Volume in drive *d*: has no label
(Том в приводе *d* метки не имеет)

Замечания:

— метка тома создается командой FORMAT /V или командой LABEL, а модифицируется только последней командой;
— отобразить метку тома можно также командами LABEL и DIR;
— дополнительная информация о метке тома содержится в описаниях команд FORMAT и LABEL.

Пример:

- VOL A: — отобразить метку диска в приводе A.

DOS 4.0. Команда VOL отображает не только метку, но и серийный номер тома, формируемый командой FORMAT. Пример сообщения:

Volume serial number is 2224-18C8

Команда DISKCOPY

Назначение: копирование содержимого гибкого диска в исходном приводе на гибкий диск в целевом приводе.
Тип: внешняя, несетевая.
Синтаксис:

DISKCOPY [*d1*:] [*d2*:] [/1]

Комментарии. *d1* и *d2* являются *исходным* и *целевым* дисководами соответственно. Наличие переключателя /1 означает, что требуется скопировать одностороннюю дискету или первую сторону (с номером 0) двухсторонней дискеты в двухстороннем дисководе. Иначе будет происходить копирование в соответствии с типом исходного привода и форматом дискеты в нем. Если дискета в целевом дисководе не проинициализирована, то автоматически осуществляется ее форматирование в точном соответствии с форматом дискеты в исходном приводе. При несовпадении форматов исходной и целевой дискет последняя приводится в соответствие с первой. Команда DISKCOPY осуществляет копирование с сохранением физического размещения информации, которое было на исходной дискете.

Предположим, что в команде явно заданы различные *d1* и *d2*. В этом случае после ввода командной строки выдаются сообщения

Insert SOURSE diskette in drive *d1*:
Insert TARGET diskette in drive *d2*:
Press any key when ready...
(Установите исходную дискету в привод *d1*:
Установите целевую дискету в привод *d2*:
Затем нажмите любую клавишу...)

Пользователь может выполнить эти предписания и нажать для копирования любую клавишу пишущей машинки либо нажать комбинацию клавиш Ctrl-Break для отмены выполнения команды.

После завершения копирования DISKCOPY спрашивает:

Copy another diskette (Y/N)?
(Копировать другую дискету (Y—да/N—нет)?)

Если Вы решили продолжить копирование дискет, нажав клавишу Y, то система предложит установить исходный и целевой диски. В противном случае выполнение команды завершается.

Возможны следующие варианты задания дисководов в командной строке:

1) указание единственного привода;
2) спецификация двух одинаковых приводов (т. е. одного и того же привода дважды);
3) отсутствие *d1* и *d2*.

В первом случае будет осуществляться копирование с указанного дисковода на текущий, во втором — копирование с использованием единственного заданного привода, а в третьем — только на текущем накопителе. Если Вы определили копирование с задействованием только одного дисковода, то на экран будут выдаваться сообщения о необходимости замены исходной дискеты на целевую, и наоборот. Следуя этим указаниям, Вы обеспечите выполнение стоящей задачи.

Команда DISKCOPY выдает следующие коды возврата:

0 — копирование завершено успешно;
1 — некритическая ошибка чтения-записи (произошла невосстанавливаемая, но не фатальная ошибка чтения-записи);
2 — копирование принудительно завершено пользователем путем нажатия Ctrl-Break;
3 — устойчивая критическая ошибка (DISKCOPY не смогла прочитать исходный ГД или отформатировать целевую дискету);
4 — ошибка инициализации команды (недостаточно памяти, ошибочно указаны приводы или допущена синтаксическая ошибка в командной строке).

Замечания:

— несмотря на возможность автоматического форматирования целевой дискеты, лучше эту операцию выполнить явно посредством команды FORMAT;

— команда DISKCOPY неприменима для копирования с жесткого диска или на жесткий диск;

— копирование содержимого дискет можно осуществить командами COPY и XCOPY. Их применение позволяет одновременно дефрагментировать файлы. DISKCOPY это не обеспечивает, однако она имеет следующие преимущества: более проста в использовании, так как копирует целиком файловую структуру дискеты; копирует файлы с атрибутами H и S; копирует метку тома; не требует обязательного предварительного форматирования гибкого диска командой FORMAT; имеет наивысшую скорость работы, особенно при большом объеме ОЗУ;

— команда DISKCOPY безвозвратно уничтожает информацию на целевой дискете, и поэтому использовать данную команду следует в высшей степени внимательно и аккуратно;

— копировать двухстороннюю дискету с переключателем /1 не имеет никакого смысла, так как будут продублированы только части файлов, а системная информация на дискете останется без каких-либо изменений;

— DISKCOPY игнорирует переименование дисководов, выполненное командой ASSIGN;

— используемые при копировании дисководы могут поддерживать большую емкость, чем та, которой обладает исходная дискета, но целевая дискета должна иметь те же тип и диаметр, что и исходная. При несовместимости дискет копирование не выполняется и выдается соответствующее сообщение;

— дискеты, скопированные на 1,2-Мбайт приводах, но имеющие меньшую емкость (в частности, 360 Кбайт), могут не считываться на некоторых соответствующих им дисководах, что объясняется различной шириной дорожки на носителе информации;

— если ПЭВМ не снабжена НЖМД, то скопировать дискеты в двух имеющихся НГМД можно следующим образом: запустить DISKCOPY с дискеты, а затем после появления сообщений об установке копируемых дискет извлечь смонтированную дискету и поместить в приводы обрабатываемые дискеты. Если бы выдача такого сообщения не была предусмотрена, пришлось бы довольствоваться одним свободным дисководом, что замедлило бы выполнение копирования.

Примеры:

- DISKCOPY A: B: — скопировать дискету в приводе A на дискету в накопителе B;
- DISKCOPY A: — скопировать дискету в приводе A на дискету в текущем накопителе;
- DISKCOPY A: A: — скопировать дискеты с использованием единственного привода A;
- DISKCOPY — скопировать дискеты с использованием текущего привода.

DOS 4.0. Если исходная дискета имеет серийный номер тома, то DISKCOPY создает новый уникальный серийный номер, записывает его на целевую дискету и отображает последний на экране дисплея по завершении копирования. В использовании этого факта состоит простейший способ защиты информации на дисках от несанкционированного копирования.

Команда DISKCOMP

Назначение: сравнение содержимого двух дискет.
Тип: внешняя, несетевая.
Синтаксис:

DISKCOMP [*d1*:] [*d2*:] [/1] [/8]

Комментарии. *d1* и d2 являются *исходным* и *целевым* приводами соответственно. Команда осуществляет подорожечное сравнение содержимого дискет, а не только проверяет идентичность файловой структуры на логическом уровне и совпадение содержимого файлов. DISKCOMP автоматически распознает формат исходной дискеты и использует его при сравнении.

Допускаются следующие переключатели:

/1 — сравнить односторонние дискеты или первые стороны (с номером 0) двухсторонних дискет в двухсторонних приводах;

/8 — сравнить 8-секторные дискеты или первые 8 секторов дискет на каждой дорожке в 9- или 15-секторных дисководах.

Предположим, что в команде явно заданы различные *d1* и *d2*. В этом случае после ввода командной строки выдаются сообщения

Insert FIRST diskette in drive *d1*:
Insert SECOND diskette in drive *d2*:
Press any key when ready...
(Установите первую (исходную) дискету в привод *d1*:
Установите вторую (целевую) дискету в привод *d2*:
Затем нажмите любую клавишу...)

Если сравнивать дискеты не требуется, то следует нажать комбинацию клавиш Ctrl-Break. В противном случае необходимо выполнить предписанные действия, и сравнение будет начато.

При несоответствии формата целевой дискеты формату исходной дискеты на экране появится сообщение

Drive types or diskette types not compatible
(Типы приводов или дискет несовместимы)

и на этом сравнение завершится.

Если содержимое дискет совпало, то будет выдано сообщение

Compare OK
(Сравнение прошло успешно)

В противном случае появится сообщение

Compare error on side *s*, track *t*
(Ошибка сравнения на стороне *s*, дорожка *t*)

и сравнение будет продолжено.

После завершения сравнения DISKCOMP выдает сообщение

Compare another diskette (Y/N)?
(Сравнить другую дискету (Y—да/N—нет)?)

При желании сравнить еще пару дискет следует нажать клавишу Y, в противном случае — N.

Если в командной строке задан единственный дисковод, то он считается исходным, а текущий — целевым. Если же *d1* и *d2* одинаковы, то для сравнения будет использоваться единственный привод *d1 (d2)*. В случае, когда ни один накопитель не указан, в качестве такового будет рассматриваться текущий дисковод. В двух последних случаях DOS будет указывать, когда надо осуществить замену исходной дискеты на целевую и обратно.

Команда DISKCOMP выдает следующие коды возврата:

0 — сравнение прошло успешно (содержимое дискет идентично);
1 — содержимое дискет различно;
2 — сравнение принудительно завершено пользователем путем нажатия Ctrl-Break;
3 — устойчивая ошибка ввода-вывода (сравнение не выполнялось);
4 — ошибка инициализации команды (недостаточно памяти, ошибочно указаны приводы или синтаксическая ошибка в командной строке).

Замечания:

— команду DISKCOMP нельзя использовать с приводами, назначенными по командам ASSIGN, JOIN и SUBST;

— сравнить содержимое файлов на дискетах можно при помощи команд COPY и FC. В отличие от них DISKCOMP учитывает физическую структуру дискет, работает быстрее и более проста в использовании;

— использование DISKCOMP для жестких дисков недопустимо;
— в дисководах большей емкости можно сравнивать дискеты меньшей емкости;
— если ПЭВМ не снабжена НЖМД, то сравнить дискеты в двух имеющихся НГМД можно следующим образом: запустить DISKCOMP с дискеты, а затем, после появления сообщения об установке сравниваемых дискет, извлечь смонтированную дискету и поместить в приводы требуемые дискеты. Если бы выдача такого сообщения не была предусмотрена, пришлось бы довольствоваться одним свободным приводом.

Примеры:

- DISKCOMP A: B: — сравнить содержимое дискет в приводах A и B;
- DISKCOMP A: — сравнить содержимое дискет в приводе A и текущем дисководе;
- DISKCOMP A: A: — сравнить дискеты в единственном накопителе A;
- DISKCOMP — сравнить дискеты в текущем приводе.

DOS 4.0. Команда DISKCOMP игнорирует различия в серийных номерах дискет.

Команда CHKDSK

Назначение: проверка целостности файловой структуры на диске, коррекция ошибок, а также отображение статуса диска и ОЗУ.
Тип: внешняя, несетевая.
Синтаксис:

CHKDSK [*d*:|*pattern*] [/F] [/V]

Комментарии. В процессе эксплуатации дисков периодически возникают различные дефекты, которые можно подразделить на следующие группы:
1) *физические дефекты,* связанные с механическим повреждением или старением магнитного покрытия (появление дефектных секторов);
2) *логические дефекты,* вызванные повреждением файловой структуры.
К последним относятся:
— *появление потерянных кластеров* (lost clusters), т.е. таких кластеров, которые зафиксированы в FAT как используемые файлом, но доступ к ним ни через один каталог диска невозможен;
— *появление пустых файлов* и *полностью пустых каталогов* (даже без элементов . и ..);
— *появление пересекающихся* (cross-linked) *файлов*, т.е. файлов, имеющих общие кластеры;
— *разрушение информации в каталогах,* FAT и стартовом секторе;
— *неидентичность копий FAT.*
Логические дефекты возникают из-за сбоев оборудования, внезапного отключения питания ПЭВМ, выполнения некорректных программ и действия компьютерных вирусов. Например, потерянные кластеры появляются тогда, когда оформление файла на диске оказалось незавершенным по причине внезапного отключения питания ПЭВМ или окончания выполнения программы без явного закрытия файла, в который производилась запись. По этим же причинам на диске образуются пустые файлы и каталоги. Причинами двух других логических дефектов являются сбои и вирусы. Последний дефект может вызываться многими причинами. Возникновение потерянных кластеров — наиболее часто встречающаяся ошибка.
Основными функциями команды CHKDSK являются *тестирование* диска в приводе *d* на предмет наличия логических дефектов (т.е. проверка целостности файловой структуры) и, попутно, выявление еще не учтенных в системной области диска физических дефектов, а также выдача отчета о проделанной работе. Дополнительно к этому команда может *скорректировать* (исправить) ряд обнаруженных логических ошибок; другие же в состоянии исправить только более мощная утилита или пользователь. CHKDSK также выдает на экран дисплея распределение дискового пространства и ОЗУ, может проверять файлы на нефрагментированность (непрерывность — contiguous) и выдавать полный перечень обрабатываемых на диске файлов.
Если в командной строке в качестве аргумента задан только привод или аргумент вообще отсутствует, то проверка непрерывности размещения файлов не производится. В случае задания *pattern* осуществляется как тестирование диска, так и проверка непрерывности размещения файлов, составные имена которых сопоставляются с шаблоном.
Допустимы два переключателя, расширяющие возможности команды:

/F — корректировать обнаруженные ошибки (если этот переключатель не задан, то ошибки только обнаруживаются);

/V — выводить на экран дисплея имена всех файлов во всех каталогах в процессе тестирования диска.

Если указан переключатель /F, то CHKDSK работает в интерактивном режиме, выдавая сообщения об обнаруженных ошибках и требуя подтверждения пользователя на их коррекцию, например:

```
10 lost clusters found in 3 chains.
Convert lost chains to files (Y/N)?
```

(Найдено 10 потерянных кластеров в 3 цепочках.
Преобразовать потерянные цепочки в файлы (Y—да/N—нет)?)

Здесь указывается, что обнаружены три непрерывные цепочки потерянных кластеров. Каждая цепочка обязательно относится к единственному файлу, и поэтому пользователю предлагается создать три файла, каждый из которых будет содержать потерянную цепочку. Получив ответ N, CHKDSK продолжит свою работу без коррекции этой ошибки. В противном случае будет создано три файла FILE*nnnn*.CHK, где *nnnn* — последовательные числа, начиная с 0001, после чего тестирование диска будет продолжено. Пользователь может впоследствии просмотреть содержимое этих файлов, чтобы определить причину появления потерянных кластеров. Если в них обнаружится текстовая информация, то файлы можно использовать по своему усмотрению. Иначе их придется просто удалить.

CHKDSK может выдавать множество различных сообщений, которые вместе с реакцией на них приведены в Приложении 1.

Если переключатель /F не задан, то при обнаружении ошибок выдаются только информационные сообщения.

Работа утилиты CHKDSK завершается отображением отчета о статусе. Приведем пример типичного отчета (для DOS 4.0):

```
Volume VOL1 created 10-18-1991 9:54a
Volume Serial Number is 0D41-10FA

  362496   bytes total disk space
  347136   bytes in 38 user files
   15360   bytes available on disk

    1024   bytes in each allocation unit
     354   total allocation units on disk
      15   available allocation units on disk

  655360   total bytes memory
  538688   bytes free
```

В нем указаны:
— метка тома (VOL1), дата (18.10.1991 г.) и время (9.54) форматирования диска;
— серийный номер тома (0D41-10FA);
— общий объем дискового пространства (362496 байт);
— объем дискового пространства, занятого пользовательскими файлами (347136 байт);
— объем свободного дискового пространства (15360 байт);
— число байт в кластере (1024);
— общее число кластеров на диске (354);
— число свободных кластеров (15);
— общая емкость ОЗУ (655360 байт);
— размер свободной области ОЗУ (538688 байт).

Дополнительно может сообщаться объем дисковой памяти, занимаемый скрытыми (hidden) файлами, а также объем дискового пространства с дефектными (bad) секторами.

Замечания:
— CHKDSK использует стандартные умолчания для текущего привода и каталога;
— CHKDSK не работает с приводами, использованными в командах SUBST и JOIN;
— команда CHKDSK с переключателем /F будет выдавать ошибку при обнаружении на диске открытых резидентными программами файлов; если переключатель /F не задан, то кластеры, принадлежащие открытым файлам, считаются потерянными;
— никогда не выполняйте CHKDSK с переключателем /F, не запустив ее предварительно без переключателя;
— команды вида CHKDSK *d*: целесообразно включать в файл AUTOEXEC.BAT для быстрой проверки целостности файловых структур при загрузке DOS;
— выполняйте CHKDSK сразу после внезапного сбоя программы или зависания системы;
— при обнаружении серьезных ошибок в файловой структуре лучшим выходом является копирование файлов (если это возможно) на другой диск и переформатирование исходного диска. Квалифицированный пользователь, однако, может попытаться восстановить информацию командой RECOVER, каким-либо редактором диска или скорректировать информацию на диске более мощной, чем средства DOS, утилитой. Великолепные в этом отношении утилиты содержатся в комплекте Norton Utilities;
— при обнаружении физических дефектов следует выполнить команду RECOVER или воспользоваться более совершенной утилитой, например, NDD из пакета Norton Utilities. Если Вы имеете утилиту NDD и умеете ею пользоваться, то она окажет Вам гораздо больше услуг, чем CHKDSK и RECOVER вместе взятые;
— CHKDSK с переключателем /V выдает список всех файлов на диске, в том числе имеющих атрибуты H и S;

— полностью понять порядок работы команды CHKDSK можно только после изучения размещения информации на магнитных дисках в подразделе 5.12.

Примеры:

- CHKDSK C: — протестировать жесткий диск C;
- CHKDSK — протестировать диск в текущем дисководе;
- CHKDSK A:REF*.TXT — протестировать диск в дисководе A, а также проверить на непрерывность файлы, содержащиеся в текущем его каталоге, имеющие имена, начинающиеся с REF, и расширение TXT;
- CHKDSK /F — протестировать текущий диск и исправить обнаруженные на нем логические ошибки;
- CHKDSK /F /V — то же, но дополнительно вывести список всех файлов.

В любом случае выдается отчет о статусе соответствующего диска и памяти.

Команда RECOVER

Назначение: восстановление информации на дефектном диске, а именно, читаемых фрагментов файлов, разрушенных из-за физических и некоторых логических дефектов.

Тип: внешняя, несетевая.

Синтаксис:

RECOVER {*file*|*d*:}

Комментарии. Команда RECOVER работает в двух режимах в зависимости от заданного аргумента:

1) если указана спецификация файла (*file*), то включается *режим восстановления* этого файла;

2) если задан только привод (или аргумент вообще не задан), то включается *режим восстановления информации на всем диске* в указанном (или текущем) дисководе.

Действия команды в этих двух режимах существенно различны.

Если на диске в области размещения одного из файлов образовался физический дефект (один или несколько дефектных секторов), то при последовательном доступе средствами DOS к содержимому такого файла удастся прочитать только информацию, размещенную до первого дефектного сектора. К остальным читаемым участкам файла последовательный доступ не обеспечивается, если не использовать определенных ухищрений. Другие методы доступа в принципе допустимы, но могут возникнуть трудности при попадании на дефектные секторы. Физические дефекты обнаруживаются при чтении дефектных файлов, а также командой CHKDSK. Обычно ясно, какие файлы размещены на дефектных секторах.

В случае выявления файлов с такими физическими дефектами следует выполнить команду RECOVER *в режиме восстановления файла* для каждого из них. Действие команды при этом состоит в посекторном чтении указанного файла и пропуске дефектных секторов. FAT обновляется таким образом, чтобы файл не занимал дефектные секторы и чтобы последние больше никогда не выделялись для размещения файлов. Модифицированный таким способом файл сохраняется на диске под своим же именем и в том же каталоге, где он первоначально находился.

Описанные действия не обеспечивают полного восстановления содержимого файлов. Поэтому обработанный командой RECOVER файл можно использовать только для просмотра его содержимого и редактирования. Это важно для текстовых файлов, так как небольшую утраченную порцию информации легко восстановить. Что же касается двоичных файлов, то полностью восстановить их содержимое не представляется возможным. В частности, не удается выполнить обработанные командой RECOVER EXE- и COM-файлы, имевшие дефектные секторы. Тем не менее даже для двоичных файлов следует выполнять эту команду для того, чтобы новые дефектные секторы (а точнее — кластеры, их содержащие) были зарегистрированы в FAT диска и поэтому больше никогда не выделялись вновь размещаемым файлам. После (частичного) восстановления двоичные файлы следует удалить и впоследствии использовать их корректные копии. Другой путь регистрации дефектных секторов командами DOS состоит только в переформатировании диска.

Режим восстановления информации на диске может привести к непоправимым последствиям в смысле невосполнимой потери содержимого диска и поэтому используется в крайних случаях. Его целесообразно применять только тогда, когда доступ к корневому каталогу диска невозможен из-за физического или логического дефекта, а FAT предположительно является неповрежденной. В данном режиме осуществляется (если это возможно) создание нового корневого каталога на месте старого, просматривается FAT и на основании ее содержимого выделяются все (теперь уже безымянные) файлы и каталоги на диске, причем последние в данном случае уже ничем не отличаются от файлов. Далее эти файлы и каталоги обрабатываются полностью аналогично предыдущему режиму и помещаются в новый корневой каталог диска под именами FILE*nnnn*.REC, где *nnnn* — последовательные числа, начиная с 0001. Конечно, таким образом будут обработаны и все потерянные кластеры (см. описание команды CHKDSK). В связи с тем что размер корневого каталога ограничен, при наличии большого числа файлов содержимое части из них будет безвозвратно утрачено. Таким образом, в режиме восстановления информации на диске переименовываются все файлы и каталоги диска и вместе с тем полностью разрушается файловая структура.

Созданные файлы можно просмотреть и отредактировать, переименовать, а ненужные — удалить. Если Вам удастся распознать двоичные файлы и эти файлы не занимали дефектные секторы, то после соответствующего переименования они могут быть использованы по своему назначению. Но все это, как правило, можно определить только опытным путем.

После своего запуска утилита RECOVER независимо от заданного режима работы всегда потребует подтверждения от пользователя на проведение восстановления сообщением вида

Press any key to begin recovery of the
file(s) on drive *d*:
(Нажмите любую клавишу, чтобы начать восстановление
файла (файлов) на приводе *d*:)

Замечания:

— в целях предосторожности аргумент для команды RECOVER требуется всегда (по умолчанию ничего не принимается);

— команда RECOVER в режиме восстановления информации является чрезвычайно опасной. Поэтому никогда не следует применять ее таким образом к жестким дискам. Что же касается дискет, то советуем работать на их копиях (по DISKCOPY), сохраняя на всякий случай оригиналы;

— команда RECOVER не работает с приводами, использованными в командах SUBST и JOIN, а также с дискетами, записанными командой BACKUP;

— в случае физических дефектов на диске после выполнения команды RECOVER целесообразно осуществить копирование всех файлов на другой диск, а исходный отформатировать несколько раз, так как возможно появление новых дефектных секторов в связи со старением и износом магнитного покрытия. Таким образом Вы получите ценнейшую информацию о его качестве. Если результаты форматирования различаются и нет уверенности в том, что в ближайшее время вновь не появятся дефектные секторы, то лучше этот диск (конечно, если он гибкий) больше не использовать, по крайней мере, для размещения уникальных для Вас файлов. Этим Вы обезопасите себя от возможных неприятностей;

— более богатыми возможностями и удобством в работе по сравнению с RECOVER обладает утилита NDD из комплекта Norton Utilities;

— квалифицированный пользователь может попытаться восстановить информацию на диске вручную с использованием какого-либо редактора диска, например, из того же комплекта;

— если никакие попытки восстановления информации успеха не имели и Вам не помогли более квалифицированные специалисты, то единственный выход состоит в переформатировании диска;

— вместо аргумента *file* в командной строке допустим аргумент *pattern*. Однако при задании последнего утилита будет восстанавливать только первый встретившийся в каталоге файл, сопоставимый с образцом, а не все такие файлы;

— полностью уяснить порядок работы команды RECOVER можно лишь после изучения размещения информации на магнитных дисках в конце данного раздела.

Примеры:

- RECOVER C:CHAPTER1.TXT — восстановить читаемую часть файла CHAPTER1.TXT из текущего каталога диска в приводе C;
- RECOVER A: — восстановить информацию, хранящуюся на диске в приводе A.

Команда FDISK

Назначение: конфигурирование жесткого диска для использования в среде DOS.
Тип: внешняя, несетевая.
Синтаксис:

FDISK

Комментарии. Жесткий диск может быть разделен на одну или несколько независимых частей, называемых **разделами**. Максимальное число разделов равно четырем. Каждый раздел может быть выделен для использования какой-либо ОС (например, DOS, Xenix, CP/M-86). DOS в принципе в состоянии работать с одним или одновременно с несколькими (пока только с двумя) разделами жесткого диска, если они созданы соответствующим образом и имеют подходящие типы.

Различают три типа разделов:

— *первичный раздел DOS*;
— *расширенный раздел DOS*;
— *раздел не-DOS*.

Жесткий диск может иметь один первичный раздел DOS, один расширенный раздел DOS и несколько разделов не-DOS, но общее число разделов, как мы уже говорили, не должно превышать 4.

Если Вы собираетесь использовать ОС семейства DOS, наличие **первичного раздела DOS** является обязательным. В нем создается единственный *системный логический диск* (обычно с именем привода C, и это от пользователя не зависит), с которого производится загрузка системы.

Расширенный раздел DOS является факультативным (необязательным). Такой раздел может быть разбит на один или несколько *логических дисков,* которым даются различные имена как приводам и которые допускается использовать под управлением DOS.

Таким образом, с точки зрения пользователя жесткий диск представляет собой совокупность логических дисков, постоянно установленных в свои *логические приводы.*

Разделы не-DOS также факультативны и создаются при необходимости работать на ПЭВМ не только с DOS, но и с другими ОС. Эти разделы для DOS недоступны.

Исходя из сказанного логическую структуру жесткого диска можно представить так, как показано на рис. 5.14.

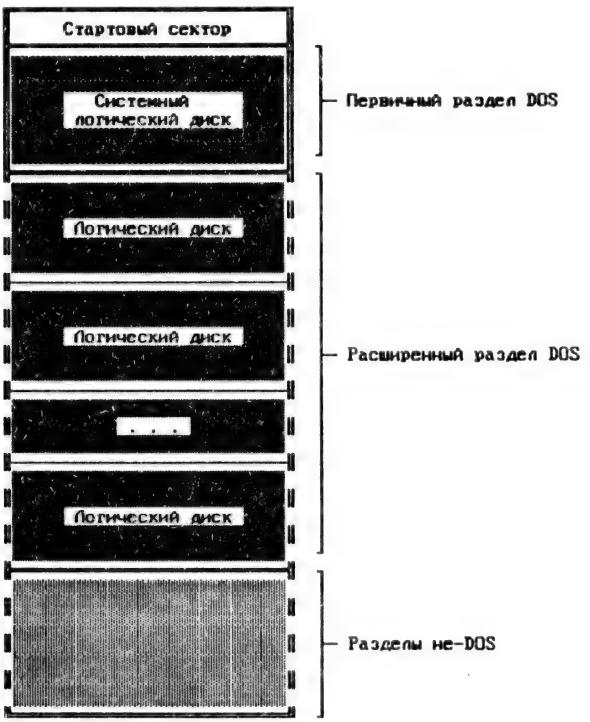


Рис. 5.14. Логическая структура жесткого диска

Каждый логический диск, созданный в одном из разделов DOS, не может иметь емкость более 32 Мбайт. Это ограничение явилось главной причиной воплощения в DOS концепции нескольких логических дисков на одном физическом, так как современные НЖМД, как правило, имеют гораздо бóльшую емкость.

Другое основание разделения физического диска на несколько логических состоит в обеспечении удобства работы с разными приложениями, а также в разграничении доступа пользователей к жесткому диску при условии коллективного использования ПЭВМ.

Заметим, что имеются внешние драйверы, поддерживающие работу с логическими дисками размером свыше 32 Мбайт. Это реализуется за счет адресации не каждого, а сразу нескольких секторов (кластера) на диске как единого целого и, возможно, соответствующего увеличения размера буферов ввода-вывода.

Команда FDISK — именно та команда, по которой осуществляется описанное выше конфигурирование жесткого диска. Утилита FDISK является интерактивной программой и выполняет следующие функции:

1) создание первичного раздела DOS с логическим диском (приводом) в нем;
2) создание расширенного раздела DOS;
3) создание логических дисков (приводов) в расширенных разделах DOS;
4) установку или смену *активного раздела*, т.е. раздела, с логического диска в котором будет осуществляться загрузка ОС после включения питания ПЭВМ;
5) удаление логических дисков и разделов DOS;
6) отображение информации о конфигурации (разбивке на разделы и логические диски) жесткого диска;
7) конфигурирование другого жесткого диска, если он имеется в ПЭВМ.

Формирование единственного логического диска в первичном разделе DOS осуществляется автоматически при создании последнего. Разделы не-DOS утилитой FDISK не строятся и не обрабатываются. Она лишь может оставить для них некоторую часть жесткого диска. Остальные же действия должны выполняться средствами тех ОС, с которыми разделы не-DOS будут использоваться.

Часто ПЭВМ поставляются с уже сконфигурированным жестким диском (один первичный раздел DOS) и записанной на него DOS. Если Вас это устраивает, то нет никакой необходимости выполнять команду FDISK. В противном случае Вам придется ее выдать.

Чтобы определить, в каком состоянии находится жесткий диск приобретенной ПЭВМ и сформировать на нем при необходимости логические диски, следует:

1) попытаться загрузить DOS с жесткого диска, для чего включить компьютер и не устанавливать никакую дискету ни в один из приводов. Если DOS в результате этого загрузилась, то можно не прибегать к помощи FDISK'а. Если загрузка не проходит, то нужно выполнить п. 2;

2) проверить, сконфигурирован ли жесткий диск утилитой FDISK. Для этого нужно загрузить DOS с дискеты, ввести команду FDISK (конечно, на системной дискете должна присутствовать утилита FDISK) и выбрать опцию «Display Partition Data» («Отобразить данные о разделах») в главном меню. Если в результате будет сообщено, что разделы отсутствуют, то конфигурирование жесткого диска обязательно. В противном случае конфигурирование факультативно. Если же утилита FDISK не запускается, нужно перейти к п. 3;

3) выполнить *низкоуровневое форматирование* жесткого диска специальной утилитой, поставляемой вместе с ПЭВМ и соответствующей установленному жесткому диску. Низкоуровневое форматирование винчестера заключается в формировании секторов (с учетом чередования) и в проверке их доброкачественности (возможности записи в них информации). После низкоуровневого форматирования следует выполнить команду FDISK, предварительно загрузив DOS с дискеты.

Невозможность низкоуровневого форматирования жесткого диска означает, что оборудование ПЭВМ неисправно или плохо подстыкованы кабели.

После конфигурирования командой FDISK жесткого диска нужно осуществить *высокоуровневое форматирование* всех созданных логических дисков, что производится утилитой FORMAT, причем диска в первичном разделе DOS — с переключателем /S.

Таким образом, для подготовки жесткого диска к работе «с нуля» требуется выполнить следующие три этапа:

1) низкоуровневое форматирование жесткого диска;
2) конфигурирование жесткого диска;
3) высокоуровневое форматирование всех созданных на нем логических дисков.

При необходимости переподготовки жесткого диска можно начать работу с любого из перечисленных этапов.

После запуска утилиты FDISK на экране появляется ее главное меню (как обычно, в скобках дается перевод на русский язык):

```
                FDISK Options
(Опции FDISK'а)

                Current fixed disk drive: 1
(Текущий привод жесткого диска:)

                Choose one of the following:
(Выберите одну (опцию) из следующих:)

                1. Create DOS Partition or Logical DOS drive
                   (Создать раздел DOS или логический привод DOS)

                2. Set Active Partition
                   (Установить активный раздел)

                3. Delete DOS Partition or Logical DOS drive
                   (Удалить раздел DOS или логический привод DOS)

                4. Display Partition Data
                   (Отобразить данные о разделах)

                5. Select Next Fixed Disk Drive
                   (Выбрать следующий привод жесткого диска)

Enter choice: [1]
(Введите выбор:)

Press ESC to return to DOS
(Нажмите Esc для возврата в DOS)
```

Если ПЭВМ имеет единственный НЖМД, то опция *5* не выдается.

Выбор практически любой опции приводит к появлению на экране меню следующего, более низкого, уровня и т.д. В каждом меню обычно в квадратных скобках предлагается ответ по умолчанию. Если он Вас устраивает, то достаточно нажать клавишу Enter и выбор будет сделан.

В противном случае следует напечатать номер опции или какое-либо значение (в зависимости от контекста) и нажать клавишу Enter. Возврат из любого подменю в главное меню, а также из последнего в DOS осуществляется путем нажатия клавиши Esc. Первая строка меню всегда информирует о том, на каком уровне Вы находитесь, а вторая — о номере конфигурируемого жесткого диска.

Рассмотрим все пункты (опции) главного меню, их назначение и использование.

Первая опция главного меню служит для создания разделов DOS и логических приводов в расширенных разделах DOS. При выборе этой опции в случае, когда конфигурирование жесткого диска еще не завершено, на экран выдается нижеприведенное меню:

```
Create DOS Partition
(Создать раздел DOS)

Current fixed disk drive: 1
(Текущий привод жесткого диска:)

                1. Create Primary DOS Partition
                   (Создать первичный раздел DOS)

                2. Create Extended DOS Partition
                   (Создать расширенный раздел DOS)

                3. Create Logical DOS Drive(s) in
                   the Extended DOS Partition
                   (Создать логический (логические) привод(ы) в
                   расширенном разделе DOS)

Enter choice: [1]
(Введите выбор)

Press ESC to return to FDISK Options
(Нажмите Esc для возврата в «Опции FDISK'а»)
```

Если расширенный раздел DOS на жестком диске еще не создан, то третья опция не появляется.

Первичный раздел DOS должен быть создан до расширенного раздела DOS. Для этого вводится вариант 1 (как раз он предлагается по умолчанию) и на экран выдается:

```
Cteate Primary DOS Partition
(Создать первичный раздел DOS)

Current fixed disk drive: 1
(Текущий привод жесткого диска:)

Do you wish to use the maximum size
for a DOS partition and make the DOS
partition active (Y/N).....? [Y]
(Намереваетесь ли Вы использовать максимальный размер
для раздела DOS и сделать
раздел DOS активным (Y—да/N—нет)?
```

Если Вы ответили Y (для «да») и емкость жесткого диска не превышает 32 Мбайт, то FDISK выдает следующее сообщение:

```
System will now restart
(Сейчас будет произведен рестарт системы)

Insert DOS diskette in drive A:
Press any key when ready...
(Установите системную дискету в привод A:.
Затем нажмите любую клавишу)
```

Выполните указанные действия, и произойдет перезагрузка DOS. Затем отформатируйте единственный созданный в первичном разделе DOS логический диск командой FORMAT C: /S с переносом на него системы.

Если Вы ответили N (для «нет») на вопрос о выделении под первичный раздел DOS всего жесткого диска или емкость последнего превышает 32 Мбайт, то на экране появится второе меню опции «Создать первичный раздел DOS», подобное следующему:

```
Create Primary DOS Partition
(Создать первичный раздел DOS)

Current fixed disk drive: 1
(Текущий привод жесткого диска:)
```

Total disk space is 42 Mbytes
(1 Mbyte = 1048576 bytes).
(Общее дисковое пространство составляет 42 Мбайт
(1 Мбайт = 1048576 байт).)

Maximum space available for partition is 32 Mbytes (76%).
(Максимальное пространство, доступное для раздела, составляет 32 Мбайт (76%).)

Enter partition size in Mbytes or percent
of disk space (%) to create a Primary
DOS partition . . . . . [32]
(Введите размер раздела в Мбайт или процентах
дискового пространства (%) для создания первичного
раздела DOS)

Press ESC to return to Fdisk Options
(Нажмите Esc для возврата в опции FDISK'a)

Вы можете принять значение по умолчанию либо ввести другое значение в Мбайт или в процентах. В последнем случае за числом ставится символ %.

Любая часть жесткого диска, не задействованная первичным разделом DOS, может быть использована для расширенного раздела DOS и/или разделов не-DOS.

Для создания *расширенного раздела DOS* Вам следует выбрать вторую опцию в меню «Создать раздел DOS». FDISK выдает меню, подобное нижеприведенному (если предыдущий ответ был, скажем, 21):

Create Extended DOS Partition
(Создать расширенный раздел DOS)

Current fixed disk drive: 1
(Текущий привод жесткого диска:)

Partition (Раздел)	Status (Статус)	Type (Тип)	Size in Mbytes (Размер в Мбайт)	Persentage of Disk Used (Доля использованного дискового пространства в процентах)
C:1		PRI DOS	21	50

Total disk space is 42 Mbytes (1 Mbyte = 1048576 bytes).
(Общее дисковое пространство составляет 42 Мбайт (1 Мбайт = 1048576 байт).)

Maximum space available for partition is 21 Mbytes (50%).
(Максимальное пространство, доступное для раздела, составляет 21 Мбайт (50%).)

Enter partition size in Mbytes or persent of disk space
to create an Extended DOS partition . . . . . [21]
(Введите размер раздела в Мбайт или в процентах дискового пространства
для создания расширенного раздела DOS)

Press ESC to return to Fdisk Options
(Нажмите Esc для возврата к опциям FDISK'a)

В графе «Раздел» отображенной таблицы указаны имя всего НЖМД (C) и порядковый номер существующего раздела на нем (1). Графа «Статус» сообщает, является ли раздел активным (в данном случае пока нет). В графе «Тип» указывается тип раздела (например, PRI DOS — первичный раздел DOS). По умолчанию для создаваемого расширенного раздела предлагается весь остаток жесткого диска. Следует сделать свой выбор или воспользоваться умолчанием.

После формирования расширенного раздела DOS следует создать в нем один или несколько *логических приводов* (логических дисков), выбрав третью опцию в меню «Создать раздел DOS». В этом случае на экран выдается:

Create Logical DOS Drive(s) in the Extended
DOS partition
(Создать логический (логические) привод(ы) DOS в
расширенном разделе DOS)

Total partition space is 21 Mbytes (1 Mbyte = 1048576 bytes).
(Общее пространство раздела составляет 21 Мбайт (1 Мбайт = 1048576 байт).)

Maximum space available for logical
drive is 21 Mbytes (100%).
(Максимальное пространство для логического
привода составляет 21 Мбайт (100%))

Enter logical drive size . . . . . [21]
(Введите размер логического привода)

Press ESC to return to Fdisk Options
(Нажмите любую клавишу для возврата в опции FDISK'a)

Выбор размера логического диска осуществляется аналогично описанному выше. При этом можно (часто целесообразно) в расширенном разделе DOS создать несколько логических дисков, которые именуются, начиная с D. FDISK продолжает выдавать такие же меню до тех пор, пока весь расширенный раздел не будет распределен. Разбиение раздела на логические диски завершается выдачей сообщения

All available space in Extended DOS
partition is assigned to logical drives
(Все доступное пространство расширенного
раздела DOS распределено между логическими приводами)

Вам следует нажать клавишу Esc для возврата в главное меню и продолжить конфигурирование жесткого диска либо выйти из утилиты FDISK.

Для установки активного раздела (если разделов для различных ОС несколько) требуется выбрать **вторую опцию в главном меню**. Активный раздел имеет статус A и содержит на своем логическом диске ОС, которая загружается при включении питания ПЭВМ или перезагрузке (перезапуске, рестарте) системы. Если Вы создали раздел с другой ОС, эта опция позволяет переключать статус активности между данным разделом и разделом DOS. В любой момент на жестком диске может быть активным только один раздел. Среди разделов DOS активным может быть только первичный раздел.

Например, если на жестком диске имеются разделы как ОС Xenix, так и DOS, на экране появится меню следующего вида (некоторые сообщения не переведены потому, что уже неоднократно встречались ранее):

Set Active Partition
(Установить активный раздел)

Current fixed disk drive: 1

Partition	Status	Type	Size in Mbytes	Persentage of Disk Used
C: 1		non DOS	21	50
2	A	PRI DOS	10	24
3		EXT DOS	11	26

Total disk space is 42 Mbytes.

Enter the number of partition you
want to make active . . . . . .[2]
(Введите номер раздела, который Вы
хотите сделать активным)

Press ESC to return to Fdisk Options

Здесь non DOS обозначает раздел не-DOS, а EXT DOS — расширенный раздел DOS.

Введите номер раздела, который Вы хотите сделать активным (по умолчанию всегда предлагается текущий активный раздел). При попытке активизации расширенного раздела DOS появляются такие сообщения:

Partition selected (3) is not startable,
active partition not changed.
(Выбранный раздел (3) не является стартовым,
активный раздел не изменен)

Press ESC to return to Fdisk Options.

Если активным сделан раздел другой ОС, то для возврата из такого состояния необходимо использовать средства именно этой системы или запустить FDISK после загрузки DOS с дискеты. Неактивные (пассивные) разделы в процессе работы недоступны.

Для удаления раздела DOS следует выбрать **третью опцию в главном меню**. Необходимость в этом может возникнуть при полном или частичном реконфигурировании жесткого диска. Удалять разделы допускается только в следующем порядке:

1) удалить все логические диски из расширенного раздела DOS;
2) удалить «пустой» расширенный раздел DOS;
3) удалить первичный раздел DOS.

Пользователь может прервать этот процесс на любом из этапов и осуществить затем частичную реконфигурацию в пределах освободившегося дискового пространства.

В режиме удаления раздела на экран выдается меню

```
Delete DOS Partition
(Удалить раздел DOS)

Current fixed disk drive: 1

                1. Delete Primary DOS Partition
                   (Удалить первичный раздел DOS)

                2. Delete Extended DOS Partition
                   (Удалить расширенный раздел DOS)

                3. Delete Logical DOS Drive(s) in
                   the Extended DOS Partition
                   (Удалить логический (логические) привод(ы) DOS в
                   расширенном разделе DOS)

Enter choice: [ ]

Press ESC to return to Fdisk Options
```

Выберите требуемый вариант дальнейших действий путем ввода его номера (по умолчанию здесь ничего не предлагается).

Если Вы *удаляете первичный раздел DOS*, то появится меню вида:

```
Delete Primary Dos Partition

Current fixed disk drive: 1

Partition    Status    Type        Size in Mbytes    Persentage
                                                     of Disk Used
C: 1         A         PRI DOS          21              50

Total disk space is 42 Mbytes (1 Mbyte = 1048576 bytes).

Warning! Data in Primary DOS
Partition will be lost. Do you wish
to continue . . . . . ? [N]
(Предупреждение! Данные в первичном разделе DOS
будут потеряны. Намереваетесь ли Вы
продолжить?)

Press ESC to return to Fdisk Options
```

Если ответить Y, то раздел будет удален; иначе — сохранится.

В случае, когда Вы решили *удалить расширенный раздел DOS,* действия будут аналогичны только что описанным.

Если Вы *удаляете логический привод* в расширенном разделе DOS, выдается меню следующего вида:

```
Delete Logical DOS Drive(s)
(Удалить логический (логические) привод(ы) DOS)

Drv          Volume Label    Mbytes      System        Usage
(Привод)     (Метка тома)    (Мбайт)     (Система)     (Использование)
  D:           DOS1            19        FAT16          90%
  E:           DOS2             2        FAT12          10%

Total extended partition space is 21 Mbytes (1 Mbyte = 1048576 bytes).
(Общее пространство расширенного раздела составляет 21 Мбайт)

Warning! Data in the logical DOS drive
will be lost. What drive do you wish
to delete . . . . . ? [   ]
(Предупреждение! Данные в логическом приводе DOS
```

будут потеряны. Какой привод Вы хотите
удалить?)

Press ESC to return to Fdisk Options

Здесь в колонке «Mbytes» указан размер логических дисков в Мбайт.
Введите имя логического привода, и Вы получите нижеприведенные сообщения:

Enter Volume Label [ ]
(Введите метку тома)

Are you sure . . . . . ? [N]
(Вы уверены?)

Если Вы зададите правильную метку логического диска и на второй вопрос ответите Y, то привод будет удален; в противном случае никаких действий предпринято не будет.
Чтобы вывести информацию о разбиении жесткого диска на разделы, следует выбрать **четвертую опцию главного меню**. При этом на экране отобразится меню следующего вида:

Display Partition Information
(Отобразить информацию о разделах)

Current fixed disk drive: 1

Partition	Status	Type	Size in Mbytes	Persentage of Disk Used
C: 1	A	PRI DOS	21	50
2		EXT DOS	21	50

Total disk space in 42 Mbytes
(1 Mbyte = 1048576 bytes).

The Extended DOS Partition contains
logical DOS drives. Do you want to
display logical drive information? [Y]
(Расширенный раздел DOS содержит
логические приводы DOS. Хотите ли Вы
выдать информацию о логических приводах?)

Если ответить Y, то появятся сообщения вида:

Display Logical DOS Drive(s) Information
(Отобразить информацию о логическом (логических) привод(е) DOS)

Drv	Volume Label	Mbytes	System)	Usage
D:	DOS1	19	FAT16	90%
E:	DOS2	2	FAT12	10%

Total Extended DOS Partition size is 21 Mbytes
(1 Mbyte = 1048576 bytes)

Press ESC to return to Fdisk Options

Если в ПЭВМ установлен второй НЖМД, то выберите **последнюю опцию главного меню** и сконфигурируйте и этот диск, как описано выше. Во второй строке каждого меню будет присутствовать сообщение

Current fixed disk drive: 2

Замечания:
— технически при создании разделов (логических дисков) утилита FDISK лишь формирует и обновляет содержимое стартового сектора жесткого диска, а также его аналогов в расширенном разделе DOS, что позволяет использовать ее для восстановления данных областей без разрушения содержимого жесткого диска;
— функцию удаления разделов (логических дисков) нужно использовать с особой осторожностью;
— если FDISK находит дефектную дорожку в начале создаваемого раздела (логического диска), то граница раздела (логического диска) автоматически смещается на доброкачественный участок поверхности;
— при удалении первичного раздела DOS предварительно необходимо сделать активным один из разделов не-DOS (если таковые имеются), чтобы загрузка какой-либо ОС впоследствии была возможной;
— FDISK не работает с приводами, использованными в командах SUBST и JOIN;

— размещение информации (в том числе системной) на жестких дисках будет рассматриваться в подразделе 5.12.

DOS 4.0. Утилита усовершенствована таким образом, что максимальный размер логического диска в любом из разделов DOS составляет теперь 2 Гбайт (2048 Мбайт).

5.6.2. Команды манипулирования каталогами

Команда CHDIR (CD)

Назначение: изменение (выбор) и отображение текущего каталога.
Тип: внутренняя.
Синтаксис:

CHDIR [*dir*|*d*:]

или

CD [*dir*|*d*:]

Комментарии. Поддержка в DOS текущих каталогов избавляет пользователя от явного указания длинных маршрутов к файлам. С каждым приводом в ПЭВМ связывается свой *текущий каталог*. При загрузке DOS в качестве текущих принимаются корневые каталоги. После установки в накопитель диска для данного накопителя можно сделать текущим любой из существующих на диске каталогов. Реализация этого как раз и является основной функцией команды CHDIR (сокращенный вариант — CD).

Если в качестве аргумента в командной строке присутствует спецификация *dir*, то действие команды состоит в том, что указанный каталог становится текущим для дисковода, на диске в котором *dir* находится. Таким дисководом является либо явно фигурирующий в *dir* накопитель, либо текущий накопитель, если ссылка на него в *dir* отсутствует.

С определенными оговорками можно увязывать понятие текущего каталога не с приводами, а с дисками (при условии, когда последние не переустанавливаются в другие накопители).

Если в качестве аргумента в командной строке указан привод (или аргумент вообще отсутствует), то по команде CHDIR на экране дисплея отображается спецификация текущего каталога диска в заданном дисководе (или в текущем дисководе). Это позволяет оперативно получить справку о текущем каталоге для каждого установленного в ПЭВМ диска.

Замечания:

— команда CHDIR не изменяет текущий привод;

— специфицировать в команде CHDIR текущий каталог какого-либо диска не имеет никакого смысла, причем *d*: в командной строке обозначает накопитель, а не текущий каталог диска в приводе *d*;

— при установке в НГМД другой дискеты текущим каталогом автоматически станет корневой каталог, если соответствующие фрагменты файловой структуры на старой и новой дискетах не совпадают;

— для автоматического отображения в каждом приглашении DOS спецификации рабочего каталога можно использовать команду PROMPT PG;

— удобным средством введения аббревиатуры маршрута является команда SUBST, вследствие чего ее в определенном смысле можно рассматривать как альтернативу для CHDIR.

Примеры:

- CHDIR C:\WORDPROC\LTRS — установить текущим каталог \WORDPROC\LTRS для диска в приводе C;
- CD .. — установить в качестве нового рабочего каталога родительский каталог старого рабочего каталога;
- CD B:..\DATA — установить в качестве нового текущего каталога диска в приводе B дочерний каталог DATA родительского каталога старого текущего каталога;
- CD C:DATA — установить в качестве нового текущего каталога диска в приводе C каталог DATA, являющийся дочерним каталогом старого текущего каталога;
- CD B:\ — сменить текущий каталог диска в приводе B на \;
- CD A: — отобразить спецификацию текущего каталога диска в приводе A;
- CD — отобразить спецификацию рабочего каталога.

Команда MKDIR (MD)

Назначение: создание нового каталога.
Тип: внутренняя.
Синтаксис:

MKDIR *dir*

или

MD *dir*

Комментарии. Команда MKDIR (сокращенно — MD) является единственным средством DOS для создания многоуровневой древовидной файловой структуры. Аргумент *dir* в команде — это спецификация нового каталога, который будет создан. Длина маршрута, включающего имя создаваемого каталога и разделители \, не должна превышать 63 символов.

Замечание: если Вы никогда не задаете расширения имен каталогов, то сможете легко отличать каталоги от файлов.

Примеры:

- MD C:\DBASE — создать каталог DBASE в корневом каталоге диска в приводе C;
- MD A:DBASE — создать каталог DBASE в текущем каталоге диска в приводе A;
- MD B:..\DBASE — создать каталог DBASE в том же каталоге диска в приводе B, в котором содержится и текущий каталог;
- MD DBASE — создать каталог DBASE в рабочем каталоге.

Команда RMDIR (RD)

Назначение: удаление существующего каталога.
Тип: внутренняя.
Синтаксис:

RMDIR *dir*

или

RD *dir*

Комментарии. Спецификация удаляемого каталога задается посредством *dir*. Данный каталог должен быть пустым (за исключением элементов . и ..). Поэтому предварительно необходимо удалить все содержащиеся в нем файлы и подкаталоги, применяя эту процедуру рекурсивно.

Замечания:
— нельзя удалить каталог, использованный в команде SUBST;
— невозможно удалить текущий каталог;
— средствами DOS не удастся удалить каталог, содержащий файлы с атрибутами H и/или S.

Пример:

- RD C:\DBASE\CONTACTS — удалить каталог CONTACTS.

Команда DIR

Назначение: отображение содержимого каталога (информации о файлах и подкаталогах в каталоге) или его подмножества.
Тип: внутренняя.
Синтаксис:

DIR *pattern* [/P] [/W]

Формат команды будет уточнен ниже.
Комментарии. По команде DIR на экран дисплея выводится следующая информация:
1) метка тома, на котором находится выделенный по *pattern* каталог *dir*, если она имеется, либо запись «... no label» («... метка отсутствует»);
2) полная спецификация выделенного по *pattern* каталога;
3) подмножество элементов этого каталога, выбранное по образцу составного имени файла в *pattern*, причем для каждого элемента выдаются:
— составное имя файла (каталога);
— размер файла в байтах (запись <DIR> для каталога);
— дата создания или последней модификации файла (дата создания каталога);
— время создания или последней модификации файла (время создания каталога);
4) общее число сопоставленных с образцом файлов и каталогов в выделенном каталоге;
5) размер свободного пространства на диске в байтах.
Отметим, что при обновлении содержимого каталога ни дата, ни время его создания не изменяются.
Результаты выполнения команды DIR могут быть, например, следующими:

```
Volume in drive C is PRI DOS
Directory of C:\

.                    <DIR>      6-09-86     7:45p
..                   <DIR>      6-09-86     7:45p
COMMAND   COM        45696      2-16-85    12:01a
AUTOEXEC  BAT          213      6-09-86    12:35p
UTILS                <DIR>      4-07-86     8:01p
        3 Files(s) 9152224 bytes free
```

Команда DIR допускает следующие переключатели:

/P — установить *постраничный режим*, при котором после заполнения экрана выдача автоматически приостанавливается до тех пор, пока на клавиатуре не будет нажата какая-либо клавиша. Иначе прокрутка изображения будет осуществляться непрерывно, что затрудняет восприятие большого списка файлов. Нажатие клавиши Pause или эквивалентной по действию комбинации клавиш позволит Вам и без переключателя /P вручную реализовать постраничный режим, хотя это менее удобно;

/W — для элементов каталога выводить только составные имена файлов по пять штук в каждой строке. Используется для ограничения объема выдачи.

Команда DIR допускает следующие сокращения аргумента:

[*dir*] — для [*dir*\]*.*

[*dir*\]*name* — для [*dir*\]*name.**

[*dir*\].*ext* — для [*dir*\]**.ext*

Замечания:
— команда DIR не выдает информацию о файлах с атрибутами H и/или S. Для этого можно использовать команду CHKDSK;
— форматы выводимых для файла (каталога) даты и времени создания зависят от страны, указанной в команде COUNTRY= файла CONFIG.SYS;
— DIR является единственной командой, в которой шаблон действует не только на файлы, но и на каталоги.

Примеры:

- DIR — выдать информацию о всех файлах и каталогах, содержащихся в рабочем каталоге;
- DIR A: — выдать информацию о всех файлах и каталогах, содержащихся в текущем каталоге диска в приводе A;
- DIR C:\DBASE /P — выдать информацию о всех файлах и каталогах, содержащихся в каталоге C:\DBASE, используя постраничный режим;
- DIR B:\.EXE — выдать информацию о всех EXE-файлах, содержащихся в корневом каталоге диска, установленного в привод B;
- DIR C:\CONFIG.SYS — выдать информацию о файле CONFIG.SYS, содержащемся в каталоге C:\;
- DIR PROG* — выдать информацию о всех файлах и каталогах из текущего каталога текущего диска, имена которых начинаются с PROG, независимо от расширений имен файлов (каталогов);
- DIR *. — выдать информацию о всех каталогах, содержащихся в рабочем каталоге (при условии, что они не имеют расширений). Наряду с каталогами будут отображены и сведения о файлах без расширений.

DOS 4.0. По команде DIR дополнительно отображается серийный номер тома.

Команда TREE

Назначение: отображение файловой структуры диска.
Тип: внешняя.
Синтаксис:

TREE [*d*:] [/F]

Комментарии. Команда TREE позволяет вывести на экран дисплея перечень каталогов на диске в приводе *d* (или на текущем диске, если *d* не задан).
Если указан переключатель /F, то дополнительно к каталогам в файловой структуре выводятся и имена всех файлов.

Замечание: имеется множество утилит, которые решают эту задачу гораздо лучше, чем команда TREE.

Примеры:

- TREE — отобразить перечень каталогов текущего диска без указания файлов;
- TREE C: /F — отобразить перечень каталогов диска в накопителе C и имена всех содержащихся на нем файлов.

DOS 4.0. Вывод команды усовершенствован (перечень каталогов представляется деревом с использованием псевдографики), а ее формат следующий:

TREE [*dir*] [/F] [/A]

Указание *dir* ограничивает обработку командой поддерева файловой структуры, в корне которого находится выделенный по *dir* каталог. Если аргумент не задан, то подразумевается рабочий каталог. Когда в качестве аргумента специфицирован *d*:, предполагается *d*:. (но не привод *d*). Поэтому для отображения всей файловой структуры диска следует задать *d*:\ (или просто \ для текущего диска). Переключатель /A обеспечивает использование командой символов пишущей машинки вместо псевдографики, что ускоряет ее выполнение. Замена псевдографики может оказаться полезной также при печати дерева каталогов на принтерах, которые ее не поддерживают.

Пример:

- TREE /A — отобразить поддерево файловой структуры текущего диска, начиная с рабочего каталога, с использованием только символов пишущей машинки.

5.6.3. Команды манипулирования файлами

Команда ERASE (DEL)

Назначение: удаление файлов.
Тип: внутренняя.
Синтаксис:

ERASE {*pattern*|*dir*}

или

DEL {*pattern*|*dir*}

Комментарии. Если в командной строке задан *pattern*, то удаляются все сопоставимые с образцом файлы. В случае указания *dir* удаляются все содержащиеся в каталоге файлы, но не подкаталоги и не содержимое последних. Следовательно, *dir* здесь считается сокращением (синонимом) *dir**.*.

При указании на удаление всех файлов в каталоге (независимо от способа) на экране появится сообщение

All files in directory will be deleted!
Are you sure (Y/N)?
(Все файлы в каталоге будут удалены!
Вы уверены (Y—да/N—нет)?)

При утвердительном ответе (Y) выполнение команды продолжается, а при отрицательном (N) — отменяется.

Замечания:

— DOS не содержит средств восстановления удаленных файлов. Поэтому команду DEL следует использовать аккуратно. В частности, если в качестве ее аргумента предполагается задать действительно образец, а не файл, то целесообразно предварительно выдать команду DIR с тем же аргументом, чтобы просмотреть список файлов, которые будут удалены. Это позволит предотвратить попутное удаление файлов, которые должны быть сохранены;

— для удаления всех файлов в текущем каталоге диска в приводе *d* необходимо в качестве аргумента команды задать *d*:. (просто *d*: здесь не проходит);

— файлы с атрибутами R, H и/или S командой ERASE не удаляются;

— технически при удалении файла изменяются только первый символ его имени в соответствующем элементе каталога и связанные с файлом элементы FAT, что обеспечивает возможность выделения элемента каталога и дискового пространства, прежде задействованных файлом, для размещения других файлов. Информация же, содержащаяся в файле, физически не стирается. Поэтому если файл занимал непрерывную область дискового пространства и после его удаления задействованная им ранее дисковая память еще не была выделена другому файлу, то имеется принципиальная возможность восстановления удаленного файла, которая и реализуется многими утилитами.

Примеры:

- DEL A:*.PAS — удалить все файлы с расширением PAS из текущего каталога диска в приводе A;
- DEL C:\DBASE — если C:\DBASE является каталогом, то удалить все содержащиеся в нем файлы (и оставить необработанными подкаталоги); иначе — удалить только файл C:\DBASE;
- DEL . — удалить все файлы (но не каталоги) в рабочем каталоге.

DOS 4.0. Для повышения безопасности команды ERASE можно использовать переключатель /P. При этом перед удалением каждого файла из заданного множества на экране дисплея появляется сообщение

file, Delete (Y/N)?
(Удалить *file* (Y—да/N—нет)?)

При ответе Y указанный файл будет удален; иначе же он останется в целости и сохранности. Выполнение команды продолжается при любом ответе, если еще есть необработанные файлы.

Команда RENAME (REN)

Назначение: переименование файлов.
Тип: внутренняя.
Синтаксис:

RENAME *pattern compname*

или

REN *pattern compname*

Комментарии. Множество переименовываемых файлов выделяется по образцу в *pattern*. Затем каждый файл из этого множества переименовывается таким образом, чтобы его новое составное имя было сопоставимо с образцом *compname* и одновременно являлось наиболее близким к старому имени. Другими словами, вместо символов-заменителей в *compname* подставляются соответствующие фрагменты старого имени файла. При этом если символ-заменитель ? находится на i-й позиции поля имени (расширения) в *compname*, то вместо него подставляется i-й символ имени (расширения) переименовываемого файла. Когда поле имени (расширения) переименовываемого файла содержит менее i символов, символ-заменитель ? ничем не заменяется. Только что сказанное справедливо для всех команд манипулирования файлами.

Замечания:
— команда RENAME не изменяет размещения файлов, поэтому любая спецификация *dir* во втором аргументе игнорируется;
— присваиваемое файлу новое имя не должно совпадать с именем уже существующего в том же каталоге файла, иначе команда не выполняется;
— файлы с атрибутами H и/или S переименовать не удастся. Остальные атрибуты командой RENAME сохраняются в неизменном виде;
— командой RENAME невозможно переименовать каталог;
— символ-заменитель * удобно использовать во втором аргументе команды не только для задания множества составных имен файлов, но и для упрощения указания единственного составного имени (см. примеры).

Примеры:

- REN *.TXT *.DOC — изменить расширение TXT на DOC у всех файлов в корневом каталоге текущего диска;
- REN CURRENT.DAT *.OLD — изменить на OLD расширение файла CURRENT.DAT, содержащегося в рабочем каталоге;
- REN CURRENT.DAT CURRENT.OLD — то же;
- REN CURRENT.DAT 1986.* — изменить на 1986 имя файла CURRENT.DAT, оставив то же расширение;
- REN 02.TXT 1?.* — переименовать файл 02.TXT в файл 12.TXT;
- REN 2.TXT 1?.* — переименовать файл 2.TXT в файл 1.TXT.

Команда ATTRIB

Назначение: изменение и отображение атрибутов файлов.
Тип: внешняя.
Синтаксис:

ATTRIB [+R|—R] [+A|—A] *pattern* [/S]

Комментарии. Если в командной строке задан аргумент $+x$, то для всех файлов, сопоставленных с образцом в *pattern*, устанавливается атрибут x. В случае указания аргумента $-x$ для выделенных по образцу в *pattern* файлов атрибут x сбрасывается. Если файл имеет атрибут x (не имеет атрибута x), то задание $+x$ ($-x$) для этого файла никакого влияния на него не оказывает.

Указание в команде только *pattern* обеспечивает выдачу на экран текущих атрибутов выделенных файлов без их изменения.

Переключатель /S распространяет действие команды на подходящие под шаблон файлы в каталогах, подчиненных выделенному по *pattern* (конечно, в выделенном каталоге файлы также обрабатываются). Следовательно, /S задает обработку файлов в целом поддереве файловой структуры по одному и тому же критерию.

Замечания:
— атрибут A позволяет управлять командами BACKUP и XCOPY, обеспечивающими резервное копирование файлов;

— установка атрибута R затрудняет удаление и модификацию файла, что удобно делать для файлов, совместно используемых (разделенных) в сети, а также для файлов с ценной информацией;
— дополнительные сведения об атрибутах файлов содержатся в п. 5.2.2;
— команда ATTRIB не позволяет устанавливать и сбрасывать атрибуты H и S, а также просматривать атрибуты таких файлов;
— в качестве аргумента команды ATTRIB нельзя указывать спецификацию каталога (вместо этого используйте *dir**.*);
— последовательностью команд ATTRIB (устанавливая и сбрасывая определенный атрибут у разных подмножеств файлов с использованием образцов) можно быстро установить требуемые атрибуты тех или иных файлов;
— команду ATTRIB без атрибутов, но с переключателем /S, можно использовать для выдачи на экран всех файлов в файловой структуре или ее части, сопоставимых с заданным образцом.

Примеры:

- ATTRIB +R C:\USER\XYZ.DAT — установить для указанного файла атрибут R;
- ATTRIB -R +A *.TXT — установить атрибут A и сбросить атрибут R у всех файлов с расширением TXT в рабочем каталоге;
- ATTRIB +A C:*.* /S — установить атрибут A у всех файлов на диске в приводе C;
- ATTRIB *.* — отобразить атрибуты всех файлов, содержащихся в рабочем каталоге;
- ATTRIB +A C:*.TXT /S — установить атрибут A у всех файлов с расширением TXT на диске в приводе C;
- ATTRIB A:P1.PAS — отобразить атрибуты файла P1.PAS, содержащегося в текущем каталоге диска в накопителе A;
- ATTRIB +R C:\DOC*.DOC /S — установить атрибут R у всех файлов с расширением DOC, содержащихся в поддереве файловой структуры диска в приводе C, корнем которого является каталог \DOC;
- ATTRIB +A A:*.*
  ATTRIB —A A:*.BAK — установить атрибут A у всех файлов в корневом каталоге диска в приводе A, за исключением тех файлов, которые снабжены расширением BAK, а у последних атрибут A сбросить;
- ATTRIB C:*.UNI /S — отобразить атрибуты всех существующих на диске в приводе C файлов с расширением UNI (одновременно с этим выводятся и их спецификации, что позволяет найти все требуемые файлы).

Команда COMP

Назначение: сравнение содержимого файлов.
Тип: внешняя.
Синтаксис:

COMP [*dir1*|*pattern1*] [*dir2*|*pattern2*]

Комментарии. Аргументы в командной строке задают два множества файлов, содержимое которых будет попарно сравниваться.

После ввода команды по первому аргументу выделяется *исходное множество* файлов, а по второму — *целевое*. Затем содержимое каждого файла из исходного множества сравнивается с содержимым того файла из целевого множества, составное имя которого получается подстановкой в *pattern2* вместо имеющихся в нем символов-заменителей соответствующих фрагментов составного имени файла из исходного множества. Файлы из исходного множества выбираются в том порядке, в каком они зарегистрированы в каталоге. Команда COMP сообщает о том, содержимое каких файлов она в данный момент сравнивает. Результаты сравнения содержимого пар файлов также выводятся на экран дисплея. Сравнение файлов завершается по исчерпании исходного множества. Поэтому число элементов в целевом множестве может превышать количество элементов в исходном множестве, но не наоборот.

Если в командной строке задан только один аргумент, то считается, что он является первым. Когда в командной строке указано менее двух аргументов (один или не одного), то команда выдаст запросы на ввод с клавиатуры недостающих спецификаций. Запрос на ввод первого аргумента выглядит следующим образом:

Enter primary filename
(Введите первичное имя файла)

Второй аргумент должен быть задан в ответ на запрос

Enter 2nd filename or drive id
(Введите второе имя файла или идентификатор привода)

Задание в качестве аргумента спецификации *pattern* приводит к тому, что для сравнения выбираются те файлы, составные имена которых сопоставимы с шаблоном. Аргумент *dir* считается сокращением для *dir**.*.

При сравнении содержимого каждой пары файлов команда COMP сначала проверяет равенство их длин. Если размеры файлов различны, то появляется сообщение

Files are different sizes
(Файлы различаются по размеру)

и сравнение не производится.

В ходе работы COMP содержимое файлов сравнивается побайтно, а при несовпадении байтов в файлах на экран выводится сообщение нижеприведенного вида:

Compare error at OFFSET *xxxxxxxx*
(Ошибка сравнения при смещении *xxxxxxxx*)
FILE1 = *yy*
FILE2 = *zz*

Здесь *x*, *y* и *z* — шестнадцатеричные цифры, FILE1 — ссылка на исходный, а FILE2 — на целевой файл. Смещение указывает номер байтов от начала файлов, на которых обнаружено несовпадение (нумерация начинается с нуля). Для каждого файла сообщается содержимое несовпавшего байта (числа *xx* и *yy*).

Сравнение двух файлов на этом не завершается, но после десяти несовпадений оно все же прекращается с выдачей сообщения

10 Mismatches — ending compare
(10 несовпадений — завершение сравнения)

При полном совпадении содержимого двух файлов на экран выводится сообщение

Files compare OK
(Файлы сравнились успешно)

Если маркеры EOF в сравниваемых файлах не найдены, то командой в процессе работы будет выдано сообщение

EOF mark not found
(Маркер EOF не найден),

но это не ошибка, а возможная информация к размышлению, так как двоичные файлы и не должны иметь этих маркеров.

После успешного или неуспешного завершения сравнения очередной пары файлов из двух множеств команда COMP переходит к сравнению содержимого следующей пары.

По исчерпании исходного множества файлов появляется сообщение

Compare more files (Y/N)?
(Еще сравнивать файлы (Y—да/N—нет)?)

При ответе Y команда COMP предлагает Вам задать с клавиатуры два новых множества для сравнения, и ее выполнение после ответов будет продолжено. В противном случае выполнение команды завершается.

Замечания:

— команда COMP сравнивает только содержимое файлов и не учитывает их физическое размещение на дисках;

— сравниваемые множества файлов могут содержаться в различных каталогах и на различных дисках;

— если по второму аргументу выделено пустое множество файлов, то выдается соответствующее сообщение. В случае, когда пустым является исходное множество, команда COMP просто запрашивает указание двух следующих пар множеств для сравнения;

— содержимое каталогов командой COMP не сравнивается;

— использование символа-заменителя * во втором аргументе позволяет не только задать множество файлов, но и упростить задание единственного файла;

— сравнение файла с самим собой обеспечивает проверку его считываемости;

— для спецификации в качестве аргумента команды рабочего каталога удобно использовать точку;

— если требуется сравнить файлы на различных дискетах, а ПЭВМ не оборудована НЖМД, то следует запустить COMP с дискеты без указания, по крайней мере, второго аргумента. Затем после выдачи сообщения на ввод аргумента нужно заменить в приводах дискеты на требуемые и ответить на запрос.

Примеры:

- COMP C:\DATA\P1.DAT A:\DATA\P2.DAT — сравнить содержимое указанных файлов;
- COMP C:*.ASM B:*.BAK — сравнить файлы *.ASM, содержащиеся в текущем каталоге диска в приводе C, с одноименными, но с расширением BAK, файлами из текущего каталога диска в приводе B;

- COMP C:*.ASM B:\PROGRAMS — сравнить файлы *.ASM, содержащиеся в текущем каталоге диска в приводе C, с одноименными файлами из каталога \PROGRAMS диска в приводе B;
- COMP C:*.ASM *.BAK — сравнить файлы *.ASM из текущего каталога диска в приводе C с одноименными, но с расширением BAK, файлами из рабочего каталога;
- COMP — сравнить файлы, которые будут заданы в ответ на запросы команды;
- COMP C:*.ASM — сравнить файлы *.ASM из текущего каталога диска в приводе C с файлами, которые будут заданы в ответ на запрос команды;
- COMP C:\ A:\ — сравнить все файлы из корневого каталога диска в приводе C с одноименными файлами, содержащимися в корневом каталоге диска в приводе A;
- COMP C:\DATA A: — сравнить все файлы из каталога \DATA диска в приводе C с одноименными файлами, содержащимися в текущем каталоге диска в приводе A;
- COMP A:CURRENT.DAT B:*.BAK — сравнить файл CURRENT.DAT из текущего каталога диска в приводе A с фалом CURRENT.BAK, содержащимся в текущем каталоге диска в приводе B;
- COMP A:CURRENT.DAT B: — сравнить тот же файл с одноименным файлом, содержащимся в текущем каталоге диска в приводе B;
- COMP . A: — сравнить все файлы из рабочего каталога с одноименными файлами, содержащимися в текущем каталоге диска в приводе A;
- COMP *.* A: — то же.

Команда FC

Назначение: сравнение содержимого файлов и отображение различий между ними.
Тип: внешняя.
Синтаксис.
Для *двоичного сравнения*:

FC [/B] *file1 file2*

Для *текстового сравнения*:

FC [/A] [/C] [/L] [/LB*m*] [/N] [/T] [/W] [/*n*] *file1 file2*

Комментарии. FC является усовершенствованной командой COMP. Последняя обеспечивает только побайтное, но не построчное сравнение содержимого файлов, причем сопоставление прекращается после десяти несовпадений. Команда же FC поддерживает режимы как *двоичного (побайтного)*, так и *текстового (построчного) сравнения*.

При *двоичном сравнении* последовательно сопоставляется содержимое соответствующих байтов двух файлов и выдается список всех различий между ними, если они имеются (проверка файлов производится до конца, независимо от числа несовпадений). Сравнивать таким способом можно как двоичные, так и текстовые файлы.

При *текстовом сравнении* содержимое двух файлов сравнивается построчно, а внутри соответствующих строк — побайтно (посимвольно). В случае нахождения первого различия в соответствующих строках их сравнение прекращается. Затем утилита FC постарается найти в файлах такие места (точнее — строки), начиная с которых, они снова становятся одинаковыми. В результате этого определяются и выдаются на экран дисплея различающиеся в двух файлах фрагменты текста (последовательности строк). При удачном совмещении содержимого файлов после таких фрагментов сравнение продолжается. В противном случае выдается соответствующее этой ситуации сообщение и сравнение файлов завершается. Режим текстового сравнения целесообразно применять только для сравнения текстовых файлов.

В командной строке *file1* и *file2* задают два файла, содержимое которых подлежит сравнению. Результаты выполнения команды FC выводятся в форме, несколько отличающейся от COMP.

Переключатели в командной строке имеют следующий смысл:

/A — сократить (Abbreviate) объем выдачи при текстовом сравнении: вместо вывода несовпадающих фрагментов целиком выдавать только их первые и последние строки;

/B — установить режим двоичного (Binary) сравнения файлов (принимается по умолчанию для файлов с расширениями EXE, COM, SYS, OBJ, LIB и BIN);

/C — игнорировать при текстовом сравнении различие в кодировке одноименных строчных и прописных букв (Case), рассматривая все буквы как прописные;

/L — установить режим текстового (Line — построчного) сравнения содержимого файлов (принимается по умолчанию для файлов, расширением имен которых не является EXE, COM, SYS, OBJ, LIB или BIN);

/LB*m* — установить размер внутреннего буфера (Line Buffer), достаточный для хранения заданного посредством *m* числа строк (по умолчанию *m* = 100);

/N — отображать при текстовом сравнении не только сами различающиеся строки, но и их номера (Number);

/T — не расширять символы табуляции (TAB) при текстовом сравнении в соответствующее число пробелов, которые приводят к размещению следующего символа в очередной позиции табуляции. По умолчанию такое преобразование файлов производится, что обеспечивает совпадение содержимого файлов, различающихся только по данному параметру (в одном файле все пробелы присутствуют явно, а в другом — заменены символами TAB);

/W — при текстовом сравнении сжимать «чистые» пробелы (White spaces — последовательности символов TAB и SP) в единственный пробел внутри каждой строки (между словами) и игнорировать их в начале, а также в конце строки. Указание этого переключателя (как и предыдущего) позволяет не принимать во внимание несущественные различия в сравниваемых текстовых файлах;

/*n* — задать посредством *n* число подряд совпавших строк при текстовом сравнении, чтобы считалось, что содержимое файлов снова согласовано (различающиеся фрагменты выделены и пройдены). Если совпало подряд после обнаружения расхождений меньшее количество строк, то они помещаются в различающиеся фрагменты текста и согласование содержимого файлов продолжается. По умолчанию в качестве *n* принимается число 2.

В случае *двоичного* сравнения отчет о различиях содержимого двух файлов выдается на экран в виде последовательности строк с тремя шестнадцатеричными числами следующего вида:

xxxxxxxx: yy zz ,

где *xxxxxxxx* — порядковый номер несовпавших байтов (начиная с нуля, т.е. смещение относительно начала файла);
yy — содержимое несовпавшего байта первого файла;
zz — содержимое несовпавшего байта второго файла.

В режиме *текстового сравнения* несовпадающие фрагменты файлов выводятся на экран в следующем виде:

```
***** file1
Последняя совпадающая строка
Отличающийся фрагмент первого файла
Первая вновь совпадающая строка
***** file2
Последняя совпадающая строка
Отличающийся фрагмент второго файла
Первая вновь совпадающая строка
```

Здесь *file1* и *file2* — спецификации сравниваемых файлов. Отметим, что первая вновь совпадающая строка соответствует началу согласованных участков файлов.

Как уже подчеркивалось, при двоичном сравнении выводится информация о всех различиях. На это никак не влияют размеры файлов.

В режиме же текстового сравнения при обнаружении несоответствий необходимо осуществить просмотр файлов вперед с сохранением различающихся фрагментов. На возможность этого непосредственное влияние оказывает размер внутреннего буфера, задаваемый переключателем /LB*m*. В случае, когда буфер уже заполнен, а файлы еще не согласованы, выдается сообщение

Resync failed. Files are too different
(Повторная синхронизация не удалась. Файлы слишком различны)

и сравнение содержимого пары файлов прекращается. В этом случае можно увеличить размер буфера и повторить сравнение.

Если сравниваемые файлы имеют различные длины, то при достижении конца одного из них на экране появляется сообщение вида

fc: *file1* longer than *file2*
(*file1* длиннее, чем *file2*)

В случае, когда содержимое сравниваемых файлов (с учетом установленных переключателями послаблений) полностью совпадает, выдается сообщение

fc: no differences encountered
(различий не обнаружено)

Замечания:
— команда FC в PC DOS 3.3 отсутствует;
— указание в командной строке шаблонов и каталогов недопустимо;

— при задании двух файлов с различными расширениями режим сравнения в общем случае следует специфицировать явно;

— допускается осуществлять сравнение содержимого файла с тем, что будет введено пользователем с клавиатуры, что достигается указанием в качестве одного из аргументов устройства CON;

— переключатель /C не обеспечивает игнорирование различий одноименных прописных и строчных букв кириллицы;

— переключатели не обязательно должны присутствовать только в начале командной строки;

— справедливы многие замечания, сделанные для команды COMP;

— команда FC может полностью заменить команду COMP, так как не обеспечивает сравнение содержимого множеств файлов.

Примеры:

- FC /A MONTHLY.RPT SALES.RPT — сравнить содержимое заданных файлов в текстовом режиме с выдачей сокращенного отчета;
- FC /LB500 /4 /L /W MYFILE.TXT YOURFILE.TXT — сравнить содержимое заданных файлов в текстовом режиме, причем размер буфера установить размером в 500 строк и считать файлы вновь согласованными при идентичности четырех строк подряд, а различия в количестве промежутков между словами во внимание не принимать.

Команда COPY

Назначение: копирование и сцепление (конкатенация) файлов.
Тип: внутренняя.
Синтаксис.
Для *копирования* файлов:

COPY {*dir1*|*pattern1*} [/V][/A|/B] [*dir2*|*pattern2*] [/V][/A|/B]

Для *конкатенации* файлов:

```
COPY      {pattern1|dir1} [/V]   [/A|/B]
[+ {pattern2|dir2   [/V]   [/A|/B]]...
  [patternN|dirN] [/V]   [/A|/B]
```

Комментарии. Очевидно, формат команды для копирования файлов является частным случаем формата для конкатенации файлов. Тем не менее мы умышленно (с целью упрощения изложения) не воспользовались имеющейся возможностью объединения двух форматов в один. Сначала рассмотрим порядок копирования, а затем — организацию конкатенации файлов. В заключение обсудим действие переключателей.

В синтаксисе для **копирования файлов** первый аргумент определяет, какие *файлы-оригиналы* подлежат дублированию, и поэтому называется *исходным*, а второй — место размещения и составные имена будущих *файлов-дубликатов*, в связи с чем называется *целевым*.

Как и в некоторых других командах, задание в качестве аргумента спецификации *dir* эквивалентно указанию *dir**.*.

Копированию подлежат те файлы из выделенного по исходному аргументу каталога, составные имена которых сопоставляются с шаблоном. Порядок копирования файлов определяется последовательностью их вхождения в исходный каталог. Каждому дубликату присваивается составное имя, полученное путем замещения символов-заменителей в целевом аргументе соответствующими фрагментами составного имени файла-оригинала. Если целевой аргумент не задан, то копирование осуществляется в рабочий каталог.

Копирование файла «на себя» не допускается. При задании такого режима выполнения команды COPY на экран выводится сообщение

```
File cannot be copied onto itself
0 File(s) copied
(Файл не может быть скопирован на себя.
 Скопировано 0 файлов)
```

и копирование не производится.

При дублировании файла дата и время его создания (последней модификации) не изменяются.

Режим **конкатенации файлов** используется для объединения их содержимого в единый файл (или в несколько файлов). Это полезно главным образом для текстовых файлов, когда из их последовательности формируется единый документ. Данный режим можно использовать и для добавления фрагментов текста в конец существующего файла.

В режиме конкатенации множества сцепляемых файлов задаются в командной строке спецификациями образцов файлов *(pattern)* и/или спецификациями каталогов *(dir)*, между которыми ставится символ +, возможно, ограниченный слева и справа последовательностями пробелов. Последний аргумент, которому не предшествует +, определяет местоположение и составные имена файлов, которые будут содержать результаты конкатенации. Таким образом, в режиме конкатенации может быть несколько *исходных* аргументов (но не менее одного) и только один —

целевой. В случае задания только одного исходного аргумента один из возможных режимов (копирования или конкатенации) выбирается в соответствии с видом аргументов и это неоднозначности не вызывает. Как и в режиме копирования, в режиме конкатенации *dir* является сокращением для *dir**.*. Файлы, выделенные по исходным аргументам, будем называть *исходными*, а сформированные по целевому аргументу — *результирующими*.

Существует несколько вариантов реализации конкатенации в зависимости от того, какие аргументы и каким образом определены. В связи с этим рассмотрим три случая:

1) все исходные аргументы и целевой аргумент заданы шаблонами (или шаблоны подразумеваются);

2) по крайней мере, одним из исходных аргументов задан единственный файл, а целевой аргумент является шаблоном (или шаблон подразумевается);

3) в качестве целевого аргумента указан единственный файл.

В *первом* случае выполнение команды COPY сводится в определенном смысле к реализации последовательности «подкоманд» COPY. Аргументы каждой такой «подкоманды» аналогичны аргументам исходной команды, но конкретизированы вплоть до указания на единственные файлы. Конкретизация аргументов в командной строке команды COPY производится по элементам множества составных имен файлов, определяемого первым исходным аргументом. По составному имени каждого файла из этого множества осуществляется уточнение других аргументов путем подстановки вместо символов-заменителей в них соответствующих фрагментов этого составного имени. Описанный процесс, а следовательно, порядок «подкоманд» COPY, определяется порядком вхождения в каталог файлов, выделенных по первому исходному аргументу. Но это в данном случае на конечный результат выполнения команды COPY влияния не оказывает (не имеет значения, в какой последовательности формируются результирующие файлы).

Во *втором* случае результатом конкатенации всегда будет единственный файл, но порядок вхождения в него исходных файлов и сами принципы вхождения, а также имя результирующего файла определяются комбинацией исходных аргументов и могут быть различными. Выделить из возможного набора вариантов общее правило затруднительно, а фирменная документация по этому поводу ничего не говорит. Вот почему проиллюстрируем данный случай лишь двумя примерами (см. ниже), чтобы заострить внимание читателя на данном вопросе.

В *третьем* случае содержимое исходных файлов последовательно сцепляется, а результат помещается в результирующий файл. Порядок конкатенации файлов определяется порядком задания исходных аргументов и вхождения исходных файлов в каталог. Сначала объединяются файлы, выделенные по первому исходному аргументу в порядке их вхождения в соответствующий каталог, затем — файлы, выделенные по второму исходному аргументу в том же порядке и т.д.

Когда целевой аргумент отсутствует, вместо него используется первый исходный аргумент. Никаких ошибок в этом случае не возникает, а выполнение команды COPY сводится к присоединению к выделенным по первому исходному аргументу (уже существующим) файлам содержимого файлов, заданных остальными исходными аргументами. С учетом сделанных замечаний для данного варианта применим один из трех рассмотренных выше случаев. Такой способ задания командной строки является аббревиатурой (сокращением) для указания в качестве целевого первого исходного аргумента (т.е. когда первый исходный и целевой аргументы совпадают). Так,

COPY *file1*+*file2 file1*

эквивалентно

COPY *file1*+*file2*

Однако если спецификации некоторых результирующих файлов (заданных явно или неявно первым аргументом) совпадают со спецификациями исходных файлов, выделенных не по первому аргументу, то для каждой такой пары выдается сообщение

Content of destination lost before copy
(Содержимое адресата (результирующего файла) потеряно до копирования)

Оно говорит о том, что файл открыт для записи (создан заново) и поэтому нет возможности прочитать его старое содержимое для повторной записи. Выполнение команды COPY в таких случаях не прекращается, но содержимое каждого такого исходного файла в результирующий файл не переносится (тем самым теряется его первоначальное содержимое). Следовательно, обработка исходного и результирующего файлов с одинаковыми именами в «подкоманде» COPY осуществляется не всегда корректно.

Если один и тот же файл выступает в роли первого и целевого (пусть даже неявно заданного) аргумента, а также сопоставим с каким-либо еще аргументом, то приведенное выше сообщение о потере содержимого адресата выдается, но реально содержимое адресата не теряется. В этом случае просто игнорируется использование данного файла в роли неперового аргумента. Объяснение указанного факта состоит в том, что файл здесь открывается не для записи, а для добавления в конец. Описанная ситуация потенциально опасна, и ее лучше избегать.

Для файлов, создаваемых или обновляемых в режиме конкатенации, устанавливаются текущие дата и время создания (в то время как в режиме копирования эти характеристики файлов не изменяются).

При определении режима работы команды COPY в случае, когда задан единственный исходный аргумент, действует следующее правило: если в качестве целевого аргумента специфицирован

единственный файл, а исходный аргумент указывает на множество файлов, то включается режим конкатенации; в противном случае выбирается режим копирования, который, очевидно, является частным случаем первого.

Выполняемые командой действия отображаются на экране дисплея путем выдачи составных имен обрабатываемых файлов.

После завершения копирования или конкатенации файлов на экране дисплея отображается сообщение

n File(s) copied
(Скопировано *n* файлов)

где *n* — число созданных файлов.

В командной строке можно использовать следующие переключатели, определяющие дополнительные режимы выполнения команды COPY и уточняющие ее действие:

/V — контролировать правильность записи информации на диск путем проверки считываемости;
/A — осуществлять копирование (конкатенацию) в *текстовом* режиме;
/B — осуществлять копирование (конкатенацию) в *двоичном* режиме.

Переключатель /V независимо от того места, где он указан в команде, действует на всю командную строку (поэтому достаточно задать его один раз). Переключатель /A (/B) действует на предшествующий и все последующие аргументы до начала области действия переключателя /B (/A). Эти два переключателя несовместимы (конечно, для одного и того же, а не для разных аргументов команды).

Длина текстовых и двоичных файлов современными версиями DOS фиксируется по-разному. Для первых длина, указанная в соответствующем элементе каталога, определяющего значения не имеет (фактический размер файла может быть выяснен только путем последовательного чтения его содержимого до маркера EOF). Этого оказывается достаточным, так как прямой доступ к текстовым файлам неприменим. Обычно (но не всегда) реальная длина текстового файла (включая маркер EOF) все же совпадает со значением, указанным в элементе каталога. Однако проверить это в каждом конкретном случае можно только путем считывания содержимого файла. Несовпадение *зарегистрированной* и *реальной* длин возникает, в частности, при пользовании текстовыми редакторами, которые для ускорения работы осуществляют не посимвольную, а поблочную запись документа в файл. В результате реальная длина может оказаться меньше зарегистрированной, если последний блок неполон. Длина же двоичного файла определяется только по значению, указанному в элементе каталога. Необходимость такого подхода объясняется возможностью прямого доступа к содержимому двоичного файла, вследствие чего до начала чтения файла должна быть известна его длина. Наличие маркера EOF в конце двоичного файла не является обязательным.

Переключатели /A и /B как раз и задают условие окончания считывания файлов-оригиналов (исходных файлов) и правила оформления файлов-дубликатов (результирующих файлов).

При применении к файлам-оригиналам (исходным файлам) переключатели имеют следующий смысл:

/A — данные из файла копируются (считываются) вплоть до первого маркера EOF, но не включая его. Остаток файла не копируется (не считывается). При отсутствии маркера EOF или если файл имеет бóльшую длину, чем указано в каталоге, команда COPY руководствуется сведениями о длине из соответствующего файлу элемента каталога;

/B — файл копируется (считывается) полностью в соответствии со значением в поле длины файла из элемента каталога.

Для файлов-дубликатов (результирующих файлов) эти переключатели интерпретируются так:

/A — после записи на диск дубликата (результирующего файла) в качестве последнего добавить маркер EOF;

/B — после записи файла на диск маркер EOF не добавлять.

Таким образом, переключатель /A используется главным образом для обработки текстовых, а /B — двоичных файлов.

Независимо от сочетания переключателей в командной строке и установленного режима работы команды COPY в элементы каталогов, описывающие файлы-дубликаты (результирующие файлы), заносятся реальные длины созданных файлов, т.е. количество записанных байтов, включая маркер EOF, если он в файл помещается.

По умолчанию в режиме копирования действует переключатель /B, а в режиме конкатенации файлов — переключатель /A. Последнее объясняется тем, что сцеплять имеет смысл только текстовые файлы.

Замечания:

— команда COPY является одной из наиболее часто используемых и наименее познанных команд DOS;

— как это ни парадоксально, но тем не менее сокращение *d*: для *d*:. в качестве исходного (но не целевого) аргумента в команде COPY не допускается;

— наряду с копированием файлов она поддерживает и их одновременное переименование;

— все заданные в режиме копирования исходными аргументами файлы реально дублируются, а не повторно регистрируются в других каталогах (пусть даже на том же диске);

— при копировании (конкатенации) файлов оригиналы (исходные файлы) сохраняются без каких-либо изменений, если не используется вариант конкатенации файлов с добавлением данных в конец существующего файла;

— команда COPY является потенциально опасной командой: если при копировании файлов или их конкатенации (за исключением добавления данных в конец файла) в качестве дубликата (результирующего файла) указан уже существующий файл, то он перезаписывается без всякого предупреждающего сообщения; однако файлы с атрибутом R не замещаются;

— файлы с атрибутами H и/или S командой COPY не обрабатываются; атрибут R в файлы-дубликаты (результирующие файлы) из файлов-оригиналов (исходных файлов) не переносится; для файлов, созданных по целевому аргументу, всегда устанавливается атрибут A;

— первый аргумент в команде COPY должен быть всегда задан (по крайней мере, точкой);

— символ-заменитель * может использоваться не только для задания множества файлов, но и для упрощения задания единственного файла в качестве целевого аргумента в режиме копирования;

— нецелесообразно пользоваться переключателем /V часто, так как информация записывается на диск, как правило, без ошибок. Однако его следует указывать при обработке файлов с важной информацией, а также при записи на «сбойные» дискеты. Задание переключателя /V увеличивает время выполнения команды COPY. Альтернативным средством верификации записи информации на диски является команда VERIFY, действующая в отличие от переключателя /V на все последующие операции записи на диск до отмены режима контроля этой же командой (переключатель же /V действует только в рамках выполнения данной команды);

— переключатель /A (/B) может указываться и перед первым аргументом команды; тогда его действие распространяется на все последующие исходные аргументы до начала области действия переключателя /B (/A) и на целевой аргумент, причем его воздействие на последний ничем не отменяется;

— копирование текстовых файлов в текстовом режиме обеспечит приведение зарегистрированных длин файлов в соответствие с реальными;

— копирование файлов на диск, содержимое которого не фрагментировано (в частности, на полностью пустой диск) позволяет реализовать их дефрагментацию;

— при попытке дублирования файлов с защищенных от перезаписи дискет операция копирования может внешне завершиться успешно, но на самом деле некоторые файлы могут быть скопированы не полностью;

— в команде COPY в качестве спецификаций файлов допускается и даже часто целесообразно задавать имена посимвольных устройств, чтобы записать текстовый файл с устройства ввода или вывести его на устройство вывода;

— переключатель /V при копировании на посимвольное устройство отвергается;

— в режиме копирования символы TAB в последовательности пробелов не разворачиваются. По этой причине команду COPY не всегда целесообразно использовать для вывода текстов на посимвольные устройства. Лучше для этих целей применять команды TYPE и PRINT;

— команда COPY не обеспечивает копирования поддерева файловой структуры. Для реализации этого служит команда XCOPY.

Примеры:

- COPY A:\MYFILE.DAT B:MYFILE.BAK — скопировать файл MYFILE.DAT из корневого каталога диска в приводе A в текущий каталог диска в приводе B и присвоить дубликату имя MYFILE.BAK;
- COPY A:\MYFILE.DAT B:*.BAK — то же;
- COPY A:\MYFILE.DAT — скопировать указанный файл в рабочий каталог (без переименования);
- COPY MYFILE.DAT B: — скопировать указанный файл из рабочего каталога в текущий каталог диска в накопителе B;
- COPY MYFILE.DAT B:\DATA — скопировать указанный файл из рабочего каталога в каталог \DATA диска в приводе B;
- COPY A:*.BAS C: — скопировать все файлы с расширением BAS из текущего каталога диска в приводе A в текущий каталог диска в приводе C;
- COPY A:*.BAS C:*.BAK — то же, но у дубликатов меняется расширение с BAS на BAK;
- COPY A: . B: — скопировать все файлы из текущего каталога диска в приводе A в текущий каталог диска в приводе B;
- COPY A:. B:*.BAK — то же, но расширение имен файлов заменить на BAK;
- COPY . B:UTILS — скопировать все файлы из рабочего каталога в каталог UTILS, содержащийся в текущем каталоге диска в приводе B;

- COPY MAIN.PAS+P1.PAS + P2.PAS MYPROG.PAS — сцепить в указанном порядке содержимое файлов MAIN.PAS, P1.PAS и P2.PAS с записью результата в файл MYPROG.PAS (все файлы обрабатываются в рабочем каталоге);
- COPY *.TXT+ *.REF *.DOC — сцепить каждый файл с расширением TXT с одноименным файлом, но имеющим расширение REF, и записать результат в файл с тем же именем и расширением DOC (все файлы обрабатываются в рабочем каталоге);
- COPY *.TXT +R.REF *.DOC — найти в рабочем каталоге первый TXT-файл, присоединить к нему содержимое файла R.REF и поместить результат в файл с тем же именем, что и TXT-файл, но с расширением DOC;
- COPY R.TXT+*.REF *.DOC — присоединить к файлу R.TXT содержимое всех REF-файлов из рабочего каталога и поместить результат в файл R.DOC;
- COPY *.TXT COMBIN.DOC — объединить все файлы с расширением TXT в единый файл COMBIN.DOC; очередность обработки исходных файлов зависит от порядка их вхождения в рабочий каталог;
- COPY *.TXT+R.REF COMBIN.DOC — то же, но в конец файла COMBIN.DOC дополнительно записывается содержимое файла R.REF;
- COPY *.TXT+*.REF COMBIN.DOC — сцепить последовательно все файлы с расширением TXT, присоединить к ним содержимое всех файлов с расширением REF и поместить результат в файл COMBIN.DOC; порядок выбора TXT- и REF-файлов зависит от последовательности их вхождения в рабочий каталог;
- COPY A:.+B:. C: — сцепить попарно все файлы из текущего каталога диска в приводе A с одноименными файлами из текущего каталога диска в приводе B и поместить результаты в текущий каталог диска в приводе C под теми же именами;
- COPY *.TXT ALL.TXT — сцепить содержимое всех файлов с расширением TXT и записать результат в файл ALL.TXT; если файл ALL.TXT существовал, то его первоначальное содержимое будет потеряно и не скопируется в новую версию файла;
- COPY ALL.TXT+*.TXT — то же, но результат конкатенации будет содержать в префиксе первоначальное содержимое файла ALL.TXT, хотя и выдается сообщение о потере содержимого результирующего файла;
- COPY ALL.TXT+*.TXT ALL.TXT — то же;
- COPY MEMO.DOC /A LETTER.DOC — скопировать содержимое файла MEMO.DOC в файл LETTER.DOC в текстовом режиме (/A действует на оба аргумента);
- COPY F1.TXT /A F2.TXT /B — скопировать содержимое файла F1.TXT в файл F2.TXT с отсечением маркера EOF, если он в первом файле имеется;
- COPY F1.TXT /A F2.TXT — скопировать содержимое файла F1.TXT в файл F2.TXT с тем, чтобы зарегистрированную длину файла привести в соответствие с реальной;
- COPY MYFILE.TXT CON — отобразить содержимое файла MYFILE.TXT на экране дисплея;
- COPY MYFILE.TXT PRN — распечатать содержимое файла MYFILE.TXT на принтере;
- COPY CON AUTOEXEC.BAT — создать файл AUTOEXEC.BAT и записать в него текст, заданный с клавиатуры (ввод текста завершается нажатием клавиши F6, а вслед за ней — Enter);
- COPY AUTOEXEC.BAT+CON — дополнить файл AUTOEXEC.BAT информацией, введенной с клавиатуры;
- COPY MYFILE.TXT/B+NUL/A MYFILE.TXT/B — изменить дату и время создания файла MYPROG.TXT на текущие;
- COPY /B MYFILE.TXT+NUL/A MYFILE.TXT — то же;
- COPY MYFILE.TXT /B+„ — то же;
- COPY CON PRN — скопировать ввод с клавиатуры непосредственно на принтер (печать будет осуществлена вслед за вводом с клавиатуры маркера EOF, а в остальном будет полностью имитироваться пишущая машинка).

Команда XCOPY

Назначение: копирование файлов и подкаталогов.
Тип: внешняя.
Синтаксис:

XCOPY {*dir1*|*pattern1*} [*dir2*|*pattern2*] [/A] [/D:*date*] [/E[/M][/P][/S][/V][/W]

Комментарии. Данная команда (ее имя — аббревиатура от eXtended COPY) является расширенным вариантом команды COPY в режиме копирования и дополнительно обеспечивает:

1) выделение подлежащих копированию файлов не только по образцу составного имени файла, но и по другим критериям;

2) копирование поддеревьев файловой структуры.

Кроме того, XCOPY использует для своей работы всю свободную оперативную память и поэтому работает гораздо быстрее.

Если в командной строке задан один аргумент, то он является *исходным* и определяет подлежащие копированию файлы *(файлы-оригиналы)*, а *дубликаты* в этом случае будут размещаться в рабочем каталоге.

В случае задания двух аргументов последний из них *(целевой)* указывает, где и под какими именами разместить дубликаты, а первый (исходный) специфицирует подлежащие копированию файлы.

Если в качестве аргумента указан каталог *dir*, а не *pattern*, то предполагается спецификация *dir**.* (все файлы в каталоге).

Копированию подлежат те файлы в выделенном по исходному аргументу каталоге, составные имена которых сопоставляются с заданным образцом. Каждый дубликат при этом размещается в выделенном по целевому аргументу каталоге и ему присваивается составное имя, полученное путем подстановки вместо символов-заменителей целевого аргумента соответствующих фрагментов составного имени файла-оригинала. В частном случае (если в целевом аргументе указано или предполагается *.*) имена файлов сохраняются.

Таким образом, правила задания аргументов здесь аналогичны команде COPY.

В связи с тем, что XCOPY (как мы увидим ниже) в отличие от команды COPY может создавать каталоги, она не всегда в состоянии однозначно интерпретировать свои аргументы. В таких случаях пользователю выдается запрос. Например, при задании в качестве целевого аргумента спецификации без символов-заменителей не всегда можно определить, что же в конечном счете требуется пользователю: поместить дубликат в указанный файл (естественно, создав его) или рассматривать эту спецификацию как каталог, в который требуется скопировать файл. Команда XCOPY принимает в данной ситуации однозначное решение, если целевой аргумент указывает на существующий каталог (файл копируется в него с сохранением имени). В противном же случае выдается сообщение

```
Does string specify a file name
or directory name on the target
      (F = file  D = directory)?
(Задает ли string в (целевом аргументе) имя файла
или имя каталога
      (F = файл  D = каталог)?)
```

Пользователю следует ответить F или D по своему усмотрению. При ответе D создается новый каталог, и в нем будет размещен дубликат. В случае ответа F осуществляется копирование файла с переименованием.

Факультативные переключатели в команде XCOPY имеют следующий смысл:

/A — копировать из выделенных по шаблону только те файлы, которые имеют атрибут A. При этом атрибуты файлов-оригиналов не модифицируются. Переключатель /A несовместим с /M;

/M — то же, но после копирования атрибут A у оригиналов сбрасывается. Этот переключатель несовместим с /A;

/D:*date* — копировать из выделенных по шаблону только созданные не раньше указанной даты *(date)* файлы. Для получения дополнительной информации см. описание команды DATE в этом же пункте;

/S — копировать не только заданные исходным аргументом файлы, но и файлы во всех подчиненных выделенному по *pattern* каталогах с учетом ограничений, накладываемых шаблоном и переключателями. При этом структура поддеревьев воспроизводится полностью, за исключением пустых каталогов (последние отбрасываются);

/E — копировать и пустые подкаталоги. Этот переключатель используется только совместно с /S;

/P — запрашивать у пользователя подтверждение на копирование каждого выделенного файла (Y/N);

/V — контролировать правильность записи информации на диск путем проверки считываемости;

/W — сделать в начале выполнения команды паузу с выдачей сообщения «Press any key when ready to start copying files» («Нажмите любую клавишу при готовности начать копирование файлов»). Пользователь может поступить в соответствии с предписанием для продолжения выполнения команды или нажать комбинацию клавиш Ctrl-Break для ее завершения.

Переключатель /D совместим с переключателем /A (/M), и оба они действуют в качестве дополнительного ограничителя множества файлов-оригиналов.

Сообщения, выдаваемые на экран в процессе выполнения XCOPY, аналогичны сообщениям, отображаемым командой COPY.

Команда XCOPY генерирует следующие коды возврата:

0 — копирование прошло без ошибок;
1 — для копирования не найдено ни одного файла;
2 — выполнение команды принудительно завершено пользователем путем нажатия комбинации клавиш Ctrl-Break;
4 — ошибка инициализации (недостаточно внешней памяти для выполнения команды, неправильный синтаксис команды или заданы аргументы, не соответствующие реальной файловой структуре и конфигурации оборудования);
5 — критическая ошибка ввода-вывода, на которую пользователь прореагировал ответом A.

Замечания. Было бы неправильным считать, что при наличии утилиты XCOPY использование команды COPY нецелесообразно. Дело в том, что последняя все же имеет следующие преимущества:

— является внутренней, а поэтому запускается быстрее и пользоваться ею более удобно;

— команда COPY допускает использование имен последовательных устройств вместо спецификаций файлов, а XCOPY — нет;

— команда COPY обеспечивает конкатенацию файлов, а XCOPY — нет;

— командой COPY в отличие от XCOPY обеспечивается возможность явного задания текстового или двоичного режима копирования файлов (XCOPY поддерживает только двоичное копирование).

Преимущества команды XCOPY были перечислены выше.

В отношении XCOPY можно сделать следующие замечания, аналогичные, но не полностью совпадающие с указанными для команды COPY:

— файлы-оригиналы и каталоги-оригиналы сохраняются без каких-либо изменений;

— при копировании в существующие файлы, не имеющие атрибута R, они заменяются заданными в команде оригиналами (т.е. уничтожаются и создаются заново); если же атрибут R установлен, то выдается сообщение «Access denied» («Доступ отвергнут») и копирование не производится. Копирование же поддерева файловой структуры сопровождается не заменой, а пополнением содержимого каталогов, возможно, уже существующего поддерева, на которое указывает целевой аргумент. В данном случае никакой подкаталог не заменяет даже одинаково размещенный и так же поименованный каталог. Вместо этого копируются только файлы и добавляются соответствующие им элементы подкаталогов. Отсутствующие каталоги, конечно, создаются. Такая техника позволяет сохранить уже созданное поддерево каталогов и содержащиеся в нем файлы, если при дублировании не возникает коллизий составных имен файлов;

— файлы с атрибутами H и/или S командой XCOPY не обрабатываются, а при копировании файлов с атрибутом R последний не переносится. Для файлов-дубликатов всегда устанавливается атрибут A;

— дата и время создания файлов при копировании не изменяются;

— рабочий каталог в исходном аргументе удобно задавать точкой;

— символ-заменитель * может использоваться не только для задания множества файлов, но и для упрощения задания единственного файла в качестве целевого аргумента;

— задание переключателя /V замедляет выполнение команды, но придает уверенность при дублировании ценной информации. Альтернативой переключателя /V является команда VERIFY, которая включает и выключает контроль записи на диск для всех последующих операций;

— команда XCOPY в случае подходящего (нефрагментированного) размещения информации на целевом диске обеспечивает дефрагментацию файлов;

— дублирование файлов с защищенных от копирования дискет к успеху не приводит (как, впрочем, и выполнение команды DISKCOPY, а также COPY);

— символы TAB в последовательности пробелов не разворачиваются;

— все заданные исходными аргументами файлы реально дублируются, а не повторно регистрируются в других каталогах (пусть даже на том же диске).

Кроме того:

— следует четко представлять себе, что в качестве исходного аргумента команды XCOPY семантически указывается не каталог, находящийся в корне копируемого поддерева файловой структуры, а содержащиеся в каталоге элементы и компоненты файловой структуры, подлежащие копированию;

— использование переключателя /W облегчит выполнение копирования информации с дискеты на дискету при наличии в ПЭВМ только двух НГМД и отсутствии НЖМД. Для этого надо запустить утилиту XCOPY с дискеты, а перед ответом на сообщение по /W заменить дискеты на требуемые для дублирования;

— циклическое копирование выполнять запрещается (каталог, в который осуществляется копирование, не должен входить в копируемое поддерево файловой структуры);

— команду XCOPY следует использовать вместо DISKCOPY при копировании всего содержимого одной дискеты на другую, имеющую иной формат;

— для устранения неоднозначности целесообразно в качестве аргументов команды XCOPY по возможности не использовать *dir*, а задавать только *pattern*, в частности, *dir**.*;

— сокращение *d*: для *d*:. допускается в качестве любого аргумента в командной строке;

— команда XCOPY с переключателями /S и /M является функциональным аналогом команды резервирования BACKUP и имеет следующие особенности: работает быстрее, чем BACKUP; файлы могут быть восстановлены этой же командой или командой COPY, но не RESTORE; не позволяет резервировать файлы, размер которых превышает емкость дискеты; не разрушает файловую структуру целевого диска; при необходимости зарезервировать все файлы требует предварительного выполнения команды ATTRIB с переключателем /S, чтобы затем использовать в командной строке переключатель /M (без него XCOPY не позволит продублировать файлы, суммарный размер которых превышает емкость дискеты, без явного удаления уже зарезервированных).

Примеры:

- XCOPY *.DAT B: — скопировать все файлы с расширением DAT из рабочего каталога в текущий каталог диска в приводе B;
- XCOPY A: — скопировать все файлы из текущего каталога диска в накопителе A в рабочий каталог,
- XCOPY *.BAS *.BAK — скопировать все файлы с расширением BAS из рабочего каталога в тот же каталог с одновременным изменением расширений на BAK;
- XCOPY *.* A:\ — скопировать все файлы из рабочего каталога в корневой каталог диска в приводе A;
- XCOPY A:\ B:\ /S/E — скопировать содержимое диска в приводе A на диск в приводе B. Если последний пуст, а первый не содержит файлов с атрибутами H и S, то диск в приводе B станет логической копией диска в приводе A. В противном случае файловая структура целевого диска пополнится файловой структурой исходного, если не принимать во внимание скрытые и системные файлы;
- XCOPY C:*.DAT A:\ /S — скопировать все файлы с расширением DAT с диска в приводе C на диск в приводе A с сохранением их размещения в файловой структуре;
- XCOPY C:\ A:\ /S/M — скопировать все файлы, которые были модифицированы или созданы после последнего выполнения такой же команды, сохраняя их размещение в файловой структуре, с диска в приводе C на диск в приводе A;
- ATTRIB C:\ +A /S
  XCOPY C:\ A:\ /S/M — скопировать (зарезервировать) файловую структуру диска в приводе C на последовательность дискет в приводе A (за исключением пустых каталогов). Как только очередная дискета заполнится, XCOPY выдаст сообщение «disk full», после чего следует установить чистую дискету и вновь ввести команду XCOPY. Если повторить описанные действия до тех пор, пока все файлы и каталоги не будут продублированы, то на дискетах получится резервная копия жесткого диска. Для восстановления информации на нем в случае ее разрушения можно воспользоваться командой COPY или XCOPY;
- XCOPY C:\LOTUS /D:10-22-91 — скопировать файлы из каталога \LOTUS диска в приводе C, созданные не раньше 22.10.91 г., в рабочий каталог;
- XCOPY C:\LOTUS /A — то же, но копируются только файлы, имеющие атрибут A, независимо от даты их создания;
- XCOPY C:\LOTUS /A /D:10-22-91 — скопировать из каталога \LOTUS диска в приводе C файлы, созданные не раньше 22.10.91 г. и имеющие атрибут A, в рабочий каталог;
- XCOPY C:\LOTUS /A /D:10-22-91 /S — то же, но дополнительно копируются поддеревья файловой структуры, корни которых содержатся в каталоге \LOTUS, включая только файлы, удовлетворяющие тем же ограничениям. Пустые каталоги не дублируются;
- XCOPY A:*.EXE B:\ /S/E — скопировать файловую структуру диска в приводе A с учетом только файлов с расширением EXE и всех (в том числе пустых) каталогов на диск в приводе B. Корнем дубликата будет корневой каталог этого диска.

Команда REPLACE

Назначение: копирование файлов с заменой или без замены существующих файлов.
Тип: внешняя.
Синтаксис:

REPLACE *pattern* [*dir*] [/A] [/P] [/R] [/S] [/W]

Комментарии. REPLACE предоставляет пользователю дополнительные по сравнению с командами COPY и XCOPY возможности по копированию файлов.

Спецификация *pattern* задает множество файлов, подлежащих копированию, а *dir* — каталог, в который требуется поместить дубликаты. При этом составные имена дубликатов всегда будут совпадать с составными именами файлов-оригиналов. Первый аргумент *(pattern)*, как обычно, назовем *исходным*, а второй *(dir)* — *целевым*. Если целевой аргумент не задан, то подразумевается рабочий каталог.

Без переключателей команда REPLACE осуществляет замену существующих в целевом каталоге файлов одноименными файлами из множества, выделенного по исходному аргументу. Таким образом, копируются не все выделенные по *pattern* файлы, а только те из них, которые существуют в целевом каталоге.

В командной строке допустимы следующие переключатели:

/S — осуществить поиск одноименных выделенным по *pattern* файлов во всем поддереве файловой структуры с корнем *dir* и найденные файлы заменить соответствующими файлами-оригиналами путем копирования последних. Ненайденные по целевому аргументу файлы не дублируются. Поддеревья, файловой структуры, доступные по исходному аргументу, в глубину при поиске исходных файлов не просматриваются. Этот переключатель несовместим с /A;

/A — дополнить содержимое *dir* теми файлами из выделенных по *pattern*, составные имена которых не зарегистрированы в *dir* (т.е. проинвертировать действие команды). В результате этого копируются только новые для *dir* файлы. Переключатель /A несовместим с /S и отменяет действие /R;

/R — обеспечить возможность замены не только файлов без атрибута R, но и с таким атрибутом. Этот переключатель не имеет смысла указывать совместно с /A;

/P — запрашивать у пользователя подтверждение на запись каждого файла на диск по целевому аргументу (для замены или добавления) путем выдачи сообщения «Replace file (Y/N)?»;

/W — в начале выполнения команды сделать паузу с выдачей на экран дисплея сообщения «Press any key when ready» («Затем нажмите любую клавишу») или «Press any key to continue» («Нажмите любую клавишу для продолжения»).

По завершении выполнения команды на экран дисплея выводится сообщение

n file(s) added/replaced
(*n* файл(ов) добавлено/заменено)

или

No files added/replaced
(Никакие файлы не добавлены/не заменены)

Команда REPLACE вырабатывает следующие коды возврата:

0 — успешное завершение;
2 — файл не найден;
3 — маршрут не найден;
5 — доступ запрещен (попытка замены файла с атрибутом R без переключателя /R);
8 — недостаточно памяти для выполнения команды;
11 — ошибка в командной строке;
15 — ошибочно задан привод.

Замечания:

— команду REPLACE целесообразно использовать для замены старой версии программного продукта новой версией, а также для ускорения добавления отсутствующих файлов;
— спецификация каталога в качестве исходного аргумента не допускается;
— файлы с атрибутами H и/или S командой REPLACE не обрабатываются;
— атрибут /R в дубликат не переносится;
— у каждого дубликата всегда взводится атрибут /A;
— никакие каталоги командой REPLACE не создаются;
— переключатель /W обеспечивает удобство работы на ПЭВМ, не оборудованной жестким диском.

Примеры:

- REPLACE *.* D: — заменить существующие в текущем каталоге диска в приводе D файлы одноименными файлами из рабочего каталога;
- REPLACE A:*.* C:\ /S — заменить каждый файл в любом каталоге диска в приводе C, совпадающий с одним из файлов в корневом каталоге диска в приводе A, последним;
- REPLACE A:*.PRD C:\MSTOOLS /A — дополнить каталог C:\MSTOOLS PRD-файлами из корневого каталога диска в приводе A.

DOS 4.0. Допустим дополнительный переключатель /U, разрешающий замену только в том случае, когда файл-оригинал создан позже (по дате и времени) подлежащего замене одноименного файла.

Команда TYPE

Назначение: отображение содержимого файла.
Тип: внутренняя.

Синтаксис:

TYPE *file*

Комментарии. Эта команда обеспечивает беглый просмотр на экране дисплея содержимого текстового файла, спецификация которого задана в качестве аргумента.

Замечания:

— задание шаблона в аргументе команды недопустимо. Для вывода на экран содержимого множества файлов можно использовать команду COPY с целевым аргументом CON;

— не следует указывать в команде TYPE двоичный файл (Вы увидите только странные символы, услышите звуковые сигналы, и Вам, возможно, придется неоднократно нажимать какую-либо клавишу для продолжения выдачи);

— файлы с атрибутами H и/или S команде TYPE недоступны;

— в отличие от COPY команда TYPE при выводе на экран всегда расширяет каждый символ TAB в файле до такого количества пробелов, чтобы последующий символ появился в очередной позиции табуляции (в ближайшей справа позиции строки, номер которой кратен восьми, т.е. 8, 16, 24 и т.д.);

— если размер файла велик, вследствие чего все его содержимое не помещается на экране, то данные выдаются сплошным потоком без остановки. Для приостановки прокрутки изображения можно использовать клавишу Pause или фильтр MORE, а для продолжения — любую клавишу;

— для печати командой TYPE содержимого файла на принтере можно задать перенаправление стандартного вывода или нажать перед завершением ввода команды комбинацию клавиш Ctrl-PrtSc, но лучше применить команду PRINT;

— действие комбинации клавиш Ctrl-PrtSc в оболочке Norton Commander блокируется;

— перенаправление вывода команды TYPE в файл не обеспечивает создания дубликата исходного файла с необходимым количеством пробелов вместо символов TAB (осуществляется копирование байт в байт).

Примеры:

- TYPE MYFILE.TXT — вывести содержимое файла MYFILE.TXT из рабочего каталога на экран дисплея;
- TYPE MYFILE.TXT ◄Ctrl-PrtSc► ◄Enter► — то же, но файл дополнительно печатается на принтере (Ctrl-PrtSc целесообразно нажать перед завершением ввода команды, а не до ее набора, чтобы сама команда не была отпечатана).

Команда PRINT

Назначение: печать файлов на принтере.
Тип: внешняя.
Синтаксис.
При *первом вызове:*

PRINT [/D:*устройство*] [/B: *b*] [/Q: *q*] [/S: *s*] [/M: *m*] [/U: *u*][/T| ◃*pattern* [/P] [/C] ▹]...

При *последующих вызовах:*

PRINT [/T| ◃*pattern* [/P] [/C] ▹]...

Комментарии. Данная команда обеспечивает вывод содержимого выделенных по *pattern* (обычно текстовых) файлов на принтер в фоновом режиме. Это означает, что в ходе печати на ПЭВМ может выполняться и другая (основная) задача. Разделение ресурсов компьютера между основной и фоновой задачами осуществляется DOS в соответствии со значениями, заданными в переключателях /S, /M и /U.

Команду PRINT можно использовать только в том случае, когда устройство вывода (принтер или графопостроитель) подключено к одному из адаптеров параллельного или последовательного интерфейса.

При *первом* после загрузки DOS выполнении команды PRINT можно установить *параметры печати* для оптимизации этого процесса в зависимости от конфигурации оборудования и потребностей пользователя. *Последующие* вызовы данной команды обеспечивают только *управление очередью печати.*

Для **установки параметров печати** используются следующие факультативные переключатели (все они, за исключением /D, имеют значения по умолчанию):

/D:*устройство* — задает имя посимвольного устройства вывода. Допустимыми значениями являются имена адаптеров параллельного интерфейса LPT1 (PRN), LPT2 и LPT3, а также имена адаптеров последовательного интерфейса COM1, COM2, COM3 и COM4. Если Вы специфицировали /D, то он должен быть первым переключателем в командной строке;

/B: *b* — устанавливает равным *b* (в байт) размер внутреннего буфера, в котором хранится фрагмент печатаемого в данный момент файла. Целесообразно задавать число, кратное 512 (размеру сектора). Допустимые значения лежат в диапазоне от 512 до 1634. По умолчанию принимается 512. Увеличение размера буфера ведет к ускорению печати, но к уменьшению объема доступной основной задаче памяти;

/Q: *q* — обеспечивает формирование очереди печати из *q* элементов (максимальное число файлов, которые могут одновременно находиться в очереди печати). Допустимы значения от 4 до 32, а по умолчанию принимается 10. Если Вы указываете переключатель /Q, то в командной строке никакие аргументы *pattern* недопустимы. Увеличение размера очереди печати повышает удобство работы, но ведет к некоторому увеличению задействованного объема ОЗУ;

/S: *s* — задает числом *s* интервал времени в тиках внутренних часов ПЭВМ, в течение которого основная задача будет использовать МП и по истечении которого будет происходить переключение МП с нее на фоновую задачу (PRINT). 1 тик равен 55 мс (1/18 доля секунды). Допустимы значения от 1 до 255; по умолчанию — 8;

/M: *m* — задает непрерывный интервал времени в тиках (посредством числа *m*), в течение которого команде PRINT разрешается использовать ресурсы ПЭВМ до переключения на основную задачу. Допустимы значения от 1 до 255. По умолчанию принимается 2;

/U: *u* — определяет числом *u* интервал времени в тиках, в течение которого команда PRINT будет ждать перевода принтера в готовность до активизации интервала, заданного переключателем /S. Допустимыми являются значения от 1 до 255, а по умолчанию принимается 1.

Если переключатель /D не задан, то пользователю выдается запрос

Name of list device [PRN]:
(Имя печатающего устройства)

Для указания PRN достаточно в ответ на него нажать клавишу Enter. В противном случае следует ввести имя адаптера интерфейса.

Для **управления очередью печати** используются следующие переключатели:

/T — удалить все файлы из очереди печати и отменить печать текущего файла;

/P — поставить файлы, выделенные по *pattern*, в очередь печати. Область действия этого переключателя распространяется на предыдущий и все последующие аргументы до начала области действия переключателя /C;

/C — удалить файлы, выделенные по *pattern*, из очереди печати. Область действия этого переключателя распространяется на предыдущий и все последующие аргументы до начала области действия переключателя /P.

Очевидно, переключатели /P и /C, приписанные к одному аргументу, несовместимы. По умолчанию действует переключатель /P, так что его использование имеет смысл, если ранее в командной строке указан один из переключателей /T или /C.

Файлы выводятся на принтер в порядке постановки их в очередь для печати. Если одновременно в очередь помещается несколько файлов (по шаблону в *pattern*), то они выбираются в порядке их вхождения в каталог на диске.

Чтобы отобразить на экране содержимое очереди печати, следует выдать команду PRINT без каких-либо аргументов и переключателей.

Замечания:

— команда PRINT не обеспечивает форматирования выводимого текста, хотя перед началом вывода каждого файла из очереди печати осуществляется перевод страницы. Разбивка содержимого файла на страницы может быть реализована только самим принтером или по управляющим символам в файле;

— не следует указывать в команде PRINT двоичные файлы;

— аналогично TYPE команда PRINT расширяет символы TAB в несколько пробелов;

— PRINT позволяет повысить производительность ПЭВМ за счет организации фоновой печати;

— команда PRINT увеличивает размер резидентной части DOS;

— каждый элемент очереди печати может содержать не более 64 символов (включая имя логического привода), так что пользователю иногда может потребоваться изменить местоположение печатаемых файлов в файловой структуре;

— принтер командой PRINT используется монопольно;

— чем меньше значение в переключателе /M и чем большее значение задано переключателем /S, тем медленнее будет осуществляться печать;

— если Вы используете команду PRINT часто, то следует опытным путем подобрать оптимальные для Вас и конфигурации оборудования значения в переключателях. Затем команду PRINT в формате первоначального использования (с целью установки параметров печати) можно поместить в командный файл, в частности в AUTOEXEC.BAT;

— некоторые программные продукты снабжены собственным аналогом команды PRINT, который и следует использовать при работе с ними;

— существует много утилит, имеющих более богатые возможности, чем PRINT, но последняя уникальна тем, что обеспечивает фоновую печать;

— если Ваша ПЭВМ снабжена принтером с низким уровнем автоматизации процесса печати (в частности, без автоматической подачи бумажных листов), то Вам во время вывода на него некогда будет выполнять другие работы (тогда возможности фоновой печати не для Вас).

Примеры:

- PRINT /D:AUX /U:2 /S:4 — установить типичные параметры для оптимизации производительности последовательного LQ-принтера (или другого относительно медленного печатающего устройства), но пока ничего не печатать;
- PRINT C:MEMO.TXT — начать печатать (точнее — поместить в очередь печати) файл MEMO.TXT;
- PRINT /T CHAP*.TXT /P INTR.TXT CHAPTER4.TXT /C — немедленно завершить печать, очистить очередь печати и поместить в нее файлы CHAP*.TXT, а также INTR.TXT; затем удалить из очереди файл CHAPTER4.TXT;
- PRINT CHAP1.TXT /C CHAP10.TXT — удалить из очереди печати указанные файлы.

Команда BACKUP

Назначение: резервное копирование (резервирование) файлов с одного диска на другой.
Тип: внешняя.
Синтаксис:

BACKUP {dir|*pattern*} [*d*:] [/S] [/M] [/A] [/F] [/D:*date*] [/T:*time*] [/L:[*file*]]

Комментарии. Эта команда осуществляет резервное копирование файлов на другой диск для того, чтобы в последующем можно было восстановить оригиналы в случае разрушения информации на диске или случайного их удаления. Дубликаты файлов создаются в специальном формате, что делает невозможной их обработку традиционными средствами DOS. Обычно BACKUP применяется для дублирования файлов, размещенных на жестком диске. Резервные копии при этом помещаются на дискеты, совокупность которых называется **резервным набором**.

Первый аргумент в командной строке является *исходным* и определяет (по шаблону) подлежащее резервированию множество файлов. Это множество может расширяться или ограничиваться переключателями. Если указана спецификация *dir*, то предполагается *dir**.*.

Второй аргумент является *целевым* аргументом (адресатом) и задает имя логического привода, на диск в котором будет осуществляться резервное копирование файлов.

В командной строке допустимы следующие переключатели:

/S — резервировать не только заданные по *pattern* файлы, но и файлы во всех подчиненных выделенному каталогах, сопоставимые с шаблоном в *pattern*. Дубликаты при этом будут размещаться таким образом, чтобы обеспечивалось последующее восстановление не только содержимого этих файлов, но и всей файловой подструктуры, в которой находились оригиналы. Выделенное описанным способом множество файлов может быть ограничено переключателями /M, /D и /E;

/M — резервировать из выделенного множества только те файлы, которые были созданы или модифицированы после последнего резервирования. При задании /M выбираются лишь файлы с атрибутом A, и после дублирования этот атрибут у оригиналов сбрасывается;

/A — добавить резервируемые файлы к тем, которые уже находятся на целевом диске. Если /A опущен, то имеющаяся на целевом диске информация удаляется;

/F — перед резервированием отформатировать целевой диск, если он еще не размечен. Форматирование осуществляется путем запуска утилиты FORMAT. Поэтому она должна находиться в рабочем каталоге или в каталоге, маршрут к которому задан в команде PATH;

/D:*date* — резервировать из выделенного множества только те файлы, которые созданы не ранее указанной посредством *date* даты. Для получения дополнительной информации см. описание команды DATE;

/T:*time* — резервировать из выделенного множества только те файлы, которые созданы не ранее заданного посредством *time* времени. Для получения дополнительной информации см. описание команды TIME ниже. Переключатель /T целесообразно использовать только совместно с /D;

/L:[*file*] — поместить протокол резервирования в указанный файл. Если спецификация файла не задана, то протокол помещается в файл BACKUP.LOG в корневом каталоге исходного диска. Файл протокола не обязательно должен существовать, но если он уже имеется, то новый протокол дописывается в его конец.

Каждый из переключателей /M, /D и /T только усиливает ограничения, налагаемые на резервируемое множество файлов (дополнительно фильтрует его по определенному критерию).

Следовательно, критерии объединяются, как и в других командах, по «И», а не по «ИЛИ». Иными словами, резервированию подлежат только файлы, удовлетворяющие одновременно всем ограничениям, заданным исходным аргументом и переключателями.

При заполнении целевого диска команда выдает сообщение, в ответ на которое следует поместить в накопитель следующий диск. Каждый целевой диск нужно соответствующим образом пометить, написав на нем его порядковый номер в данном сеансе резервирования. Это облегчит при необходимости восстановление информации.

Протокол резервирования в первой строке содержит текущую дату и время, а в каждой последующей — спецификацию продублированного файла и порядковый номер целевого диска, на который помещена копия.

Команда BACKUP выдает следующие коды возврата:

0 — нормальное завершение;
1 — для резервирования не выбрано ни одного файла;
2 — некоторые файлы не зарезервированы из-за конфликтов по совместному их использованию (разделению);
3 — выполнение команды принудительно завершено пользователем;
4 — выполнение команды аварийно завершено из-за ошибки.

Замечания:

— команда BACKUP является уникальным средством DOS для переноса файлов, длина которых превышает емкость дискеты, на жесткие диски других ПЭВМ, а также для резервирования таких файлов;

— переносимые файлы целесообразно предварительно разместить в корневом каталоге исходного диска, чтобы впоследствии перед восстановлением файлов на винчестере другой ПЭВМ не воссоздавать исходный маршрут. Дело в том, что по команде RESTORE файлы будут восстанавливаться с сохранением их местоположения в исходной файловой структуре;

— BACKUP может быть использована для резервирования файлов с любого диска на любом (не обязательно с жесткого на гибком);

— наиболее близким функциональным аналогом BACKUP является команда XCOPY, причем последняя выполняется быстрее, но не позволяет резервировать указанные в первом замечании файлы;

— восстановить зарезервированную посредством команды BACKUP информацию можно только путем выполнения команды RESTORE;

— команда BACKUP часто требует явного задания (пусть даже текущего) исходного привода;

— вся резервируемая информация записывается на целевой диск в единый файл BACKUP.*ппп* (где *ппп* — порядковый номер диска), а справочная информация помещается в файл CONTROL.*ппп* и используется командой RESTORE. Оба названных файла будут иметь атрибуты R и A;

— наличие протокола резервирования и надписей на дискетах облегчит избирательное восстановление файлов вследствие того, что пользователь сможет быстро выбрать дискеты, содержащие требуемые файлы;

— утилита BACKUP не резервирует системные файлы с MP BIOS, БМ DOS и КП;

— файлы с атрибутами H и/или S командой BACKUP не обрабатываются; если не задан переключатель /M, то атрибут A у оригиналов не сбрасывается, но в любом случае сбрасывается в дубликатах; файлы с атрибутом R резервируются, причем этот атрибут будет установлен при их восстановлении;

— не следует использовать команду BACKUP, когда привод, с которого осуществляется резервирование, задействован в команде ASSIGN, JOIN или SUBST. Если Вы все же пренебрегли этой рекомендацией, то не сможете восстановить зарезервированные файлы;

— при разделении (совместном использовании) файлов система позволит Вам зарезервировать только те файлы, доступ к которым разрешен;

— резервирование требует много времени: например, дублирование всего лишь 10-Мбайт жесткого диска займет около двух часов; при этом нужны будут около 30 360-Кбайт дискет. Тем не менее эту процедуру необходимо выполнять регулярно, если на жестком диске имеется уникальная обновляемая информация;

— для создания надежных архивов перед резервированием нужно выполнить команду VERIFY ON.

Примеры:

- BACKUP C:\ARC\LONGFILE.LZH A: — зарезервировать указанный файл на дискете (дискетах) в приводе A;
- BACKUP C:\ A: /S — зарезервировать все файлы с диска в приводе C, на дискетах в накопителе A;
- BACKUP C:\DBASE*.DBF A: /S — зарезервировать все файлы с расширением DBF из файловой подструктуры, в корне которой находится каталог C:\DBASE, на дискетах в приводе A;
- BACKUP C:*.DAT A: /S/M/A — зарезервировать все файлы с расширением DAT с диска в приводе C, которые были изменены после последнего дублирования с переключателем /M, на одной или нескольких дискетах в приводе A с сохранением имеющейся на этих дискетах информации;

- BACKUP C:\ A: /S/D:01-13-90 /T:17:00:00 /L:C:\13JAN90.BKP — зарезервировать все файлы с диска в приводе C, созданные не ранее 17.00 13.01.90 г., на дискетах в накопителе A и поместить протокол в файл C:\13JAN90.BKP;
- BACKUP C:\ A: /F — зарезервировать содержимое диска в приводе C на дискетах в накопителе A с их предварительным (при необходимости) форматированием.

DOS 4.0. Для задания формата дискеты, отличного от определяемого НГМД и его драйвером, переключатель /F можно указывать в форме /F:*формат*. Допустимые значения поля *формат* приведены в описании команды FORMAT.

Команда RESTORE

Назначение: восстановление зарезервированных посредством команды BACKUP файлов.
Тип: внешняя.
Синтаксис:

RESTORE *d*: *pattern* [/S] [/P] [/B:*date*] [/A:*date*] [/E:*time*] [/L:*time*] [/M] [/N]

Комментарии. Данная команда используется для восстановления на диске разрушенных или случайно удаленных файлов, если они предварительно были зарезервированы на другом диске (дисках) командой BACKUP. Обычно в качестве исходных для восстановления применяются гибкие диски, а информация регенерируется на жестком диске.

Первый аргумент команды является *исходным* и определяет привод, в который устанавливаются диски с резервными копиями файлов.

Второй аргумент является *целевым* и определяет восстанавливаемые файлы (а также их размещение при восстановлении). Задавать целевой аргумент нужно так, чтобы он соответствовал исходному аргументу команды BACKUP, посредством которой исходный резервный набор создавался. Под соответствием в данном случае понимается то, что множество спецификаций файлов, выделенных по целевому аргументу команды RESTORE, должно пересекаться с множеством спецификаций файлов, выделенных по исходному аргументу соответствующей команды BACKUP (при этом маршруты должны полностью совпадать, а приводы могут различаться). Если первое множество содержится во втором, то будет восстанавливаться только часть зарезервированных файлов. В противном случае могут восстанавливаться либо часть, либо все зарезервированные файлы, что определяется видом шаблона. В частности, если вы зарезервировали файлы по шаблону *.DOC, то можете восстановить их по шаблону *.* (все DOC-файлы) или CHAP*.* (только те DOC-файлы, имена которых начинаются с CHAP). Такая техника позволяет извлечь из резервного набора не все имеющиеся в нем файлы, а только некоторые из них, но восстановленные файлы обязательно должны быть размещены в тех же местах файловой структуры, где были оригиналы при резервировании. Как раз поэтому перед переносом (возможно, длинных) файлов с одной ПЭВМ на другую рекомендуется скопировать исходные файлы в корневой каталог жесткого диска.

В командной строке допустимы следующие переключатели, расширяющие или ограничивающие множество выделенных по *pattern* файлов:

/S — восстанавливать не только заданные целевым аргументом файлы, но и файлы во всех подчиненных каталогах с учетом ограничений, накладываемых посредством *pattern* и другими переключателями;

/P — выдавать запрос на подтверждение восстановления (перезаписи) каждого файла, если он на целевом диске уже существует и имеет атрибут R. Когда /P не задан, файлы с атрибутом R заменяются на целевом диске без каких-либо предупреждающих сообщений;

/B:*date* — восстановить только те файлы, которые на целевом диске в последний раз модифицированы *не позже* указанной даты *date*, тем самым заместив их зарезервированными версиями;

/A:*date* — восстановить только те файлы, которые на целевом диске в последний раз модифицированы *не раньше* указанной даты *date* (тем самым заместив их зарезервированными версиями). Пара переключателей /B и /A позволяет выбрать любой диапазон дат;

/E:*time* — восстановить только те файлы, которые на целевом диске в последний раз модифицированы *не позже* указанного времени *time*. Переключатель имеет смысл использовать только совместно с /B;

/L:*time* — восстановить только те файлы, которые на целевом диске в последний раз были модифицированы *не раньше* указанного времени *time*. Переключатель имеет смысл использовать только совместно с /A;

/M — восстановить только те файлы, которые на целевом диске модифицированы после последнего резервирования (т.е. в момент восстановления имеют атрибут A) или отсутствуют (удалены);

/N — восстановить только файлы, которые на целевом диске больше не существуют (удалены).

Восстановлению подлежат лишь файлы, удовлетворяющие одновременно всем критериям, заданным целевым аргументом (шаблоном) и переключателями.

После ввода команды Вы будете оповещены о необходимости установки дискеты 01 созданного ранее резервного набора. Можно выполнить это предписание или установить сразу именно ту дискету, которая содержит восстанавливаемый файл (файлы). Сведения о расположении файлов в резервном наборе находятся в протоколе резервирования (см. описание команды BACKUP). Если требуется поместить в дисковод следующий диск, то команда RESTORE сообщит Вам об этом.

Данная команда вырабатывает следующие коды возврата:

0 — нормальное завершение;
1 — для восстановления не найдено ни одного файла;
3 — выполнение команды принудительно завершено пользователем;
4 — выполнение команды аварийно завершено из-за ошибки.

Замечания:

— команда RESTORE восстанавливает файлы из резервных наборов, созданных командами BACKUP более ранних версий DOS;

— ни исходный, ни целевой аргументы опускать нельзя;

— целевой аргумент часто допускается задавать в виде символа . для обозначения шаблона *.* или *d*: для краткого указания *d*:.*.*, однако спецификацию *dir* в общем случае употреблять нельзя;

— команда RESTORE при восстановлении поддерева файловой структуры в случае необходимости автоматически создает требуемые каталоги;

— RESTORE аналогично команде XCOPY не разрушает, а лишь дополняет существующее дерево файловой структуры;

— после завершения выполнения команды RESTORE не мешает убедиться в результатах ее работы с помощью команд DIR и TYPE;

— зарезервированные с жесткого диска на одной ПЭВМ файлы можно восстановить на жесткий диск другой ПЭВМ, обеспечив тем самым транспортирование информации;

— переключатели /B и /E используются для замещения старых файлов их новыми версиями;

— переключатели /A и /L или переключатель /M (последний — после команды BACKUP с одноименным переключателем) применяются для замещения неверно модифицированных файлов их старыми зарезервированными версиями;

— когда Ваш жесткий диск заполнится файлами целиком, Вы можете зарезервировать какое-либо не используемое в данный момент поддерево файловой структуры и удалить его, чтобы разместить на диске новые файлы. Для восстановления удаленных файлов нужно проделать ту же процедуру с другим поддеревом файловой структуры и выполнить команду RESTORE для ранее зарезервированного поддерева.

Примеры:

- RESTORE A: C:\ARC\LONGFILE.LZH — восстановить указанный файл;
- RESTORE A: C:*.* /S — восстановить на диске в приводе C все существующие в резервном наборе файлы;
- RESTORE A: C:\DBASE*.DBF /S — восстановить поддерево файловой структуры, содержащее файлы с расширением DBF, в каталоге C:\DBASE;
- RESTORE A: C:*.* /S /A: 01-13-90 /B:01-14-90 — восстановить на диске в приводе C все файлы, модифицированные в последний раз с 13 по 14 января 1990 г.

Команда SHARE

Назначение: установка многопользовательского режима использования файлов в ЛВС.
Тип: внешняя.
Синтаксис:

SHARE [/F:*f*] [/L:*l*]

Комментарии. Эта команда используется в локальной сети при наличии сетевого ПО, осуществляющего блокировку файлов для того, чтобы в каждый момент доступ к ним мог осуществлять только один пользователь (точнее — процесс). Такой режим доступа к файлам принципиально необходим при их обновлении, так как результат одновременного выполнения этой операции разными процессами непредсказуем.

Действие команды SHARE состоит в том, что каждая операция обмена с файлом, запрошенная выполняемой программой, проверяется на допустимость в соответствии с *кодом разделения* (sharing), т.е. с состоянием *запора*, после установки которого файл открыт. Для реализации этого активизируется контроль блоков управления файлами — FCB (см. описание команды FCBS= в п. 5.10.2). FCB здесь является аббревиатурой от File Control Block.

Если во время выполнения операции чтения или записи диск был извлечен из привода, то специальным сообщением будет предложено установить его, а после выполнения этого требования будет осуществлена проверка диска по метке тома. При выявлении установки не того диска,

который требуется, будет снова выдано сообщение. И только после того, как резидентная программа, загруженная командой SHARE, убедится в корректности установки диска, операция чтения/записи будет продолжена.

В командной строке можно указывать следующие переключатели:

/F:*f*— выделить область памяти размером в *f* байт для записи информации о разделении файлов. По умолчанию принимается 2048. Отметим, что каждый открытый файл требует поля, достаточного для размещения его составного имени и еще 11 байт другой информации. Тогда получается около 20 байт на файл;

/L:*l*— установить *l* запоров на блокировку доступа к файлам (по умолчанию — 20).

Замечания:

— команда SHARE только создает условия для блокировки одновременного доступа к файлам, но не обеспечивает ее. В частности, запоры устанавливаются и снимаются сетевым ПО;

— выполнение команды SHARE приводит к увеличению размера резидентной части DOS примерно на 6 Кбайт в результате загрузки в ОЗУ дополнительных программ и таблиц;

— команда SHARE обеспечивает определенную защиту информации на дискетах в IBM PC AT и ПЭВМ семейства PS/2. Например, если извлечь дискету при выполнении операции записи, в частности после обновления каталога, но до обновления FAT, будет выдано сообщение и запись не будет продолжена до тех пор, пока Вы не установите именно требуемую дискету.

Пример: SHARE /F:1024

DOS 4.0. Команда SHARE обеспечивает работу с логическими дисками на винчестере, размер которых превышает 32 Мбайт. Если у Вас имеются такие диски, то необходимо при запуске DOS выдать SHARE из файла AUTOEXEC.BAT или с использованием команды конфигурирования INSTALL=. Если же утилита SHARE находится в корневом каталоге системного диска или в каталоге, указанном в команде конфигурирования SHELL=, то она выполняется автоматически. Иногда при автоматическом выполнении команда SHARE работает лучше.

5.6.4. Команды управления посимвольными устройствами

Описываемые здесь команды не охватывают всех возможностей, предоставляемых пользователю ПЭВМ по управлению посимвольными ПУ. Дополнительные средства и методы реализации этого будут рассмотрены в подразделе 5.11.

Команда CLS

Назначение: очистка экрана дисплея.
Тип: внутренняя.
Синтаксис:

CLS

Комментарии. При выводе информации на экран его заполнение осуществляется построчно. Когда последняя строка окажется занятой, то содержимое экрана смещается на одну строку вверх, а новая информация отображается в нижней строке. Чтобы полностью освободить экран от ненужной информации и возобновить вывод с его первой (верхней) строки, следует выполнить команду CLS. При этом экран полностью очищается, а в левом верхнем углу появляются приглашение DOS и курсор. Если Вы специальными средствами DOS (Escape-последовательностями) не меняли оцветовку экрана, то весь он заполняется черным цветом, а символы будут выводиться белым цветом. В противном случае экран окрашивается установленным фоновым цветом, а для выдачи информации также будет использоваться выбранный для этой цели цвет.

Замечание: команду CLS часто полезно использовать в командных файлах.

Пример: CLS.

Команда MODE: общие сведения

Назначение: отображение статуса и установка режимов работы посимвольных устройств.
Тип: внешняя.
Комментарии. Команда MODE выполняет следующие функции:

1) устанавливает и изменяет режимы работы принтеров, подключенных к адаптерам параллельных интерфейсов;

2) устанавливает и изменяет режим работы дисплея, а также осуществляет переключение между имеющимися в ПЭВМ дисплейными адаптерами;

3) устанавливает и изменяет режимы работы адаптеров последовательных интерфейсов;

4) перенаправляет вывод, адресованный подключенному к адаптеру параллельного интерфейса принтеру, в адаптер последовательного интерфейса;

5) устанавливает и изменяет скорость работы клавиатуры;

6) отображает статус посимвольных устройств;

7) подготавливает посимвольные устройства к смене и обеспечивает смену кодовых страниц.

Для задания каждой функции используется свой синтаксис, резко отличающийся от других. Кроме того, семантика от функции к функции также существенно различается. Следовательно, команда MODE по сути объединяет в себе несколько связанных между собой по целевому назначению команд, и поэтому мы рассмотрим каждую из них отдельно.

Замечания:
— некоторые функции, в частности, установка скорости работы клавиатуры и режима дисплея, требуют подключения драйвера ANSI.SYS (см. п. 5.10.3);
— необходимые команды MODE можно ввести с клавиатуры, однако целесообразнее задавать их в командных файлах, в частности в AUTOEXEC.BAT (см. подраздел 5.9).

DOS 4.0. Команда MODE модифицирована как синтаксически, так и семантически. Функции же 5 и 6 являются полностью новыми.

Команда MODE: управление принтером

Назначение: задание режима работы принтера, подключенного к адаптеру параллельного интерфейса.
Синтаксис:

MODE LPT*n*: [*c*] [,[*l*][, P]]

Комментарии. В командной строке числа *n*, *c* и *l* обозначают следующее:

n — номер адаптера параллельного интерфейса, к которому подключен принтер (1, 2 или 3);

c — число выводимых символов в строке (80 для обычного и 132 для сжатого шрифта при условии, что принтер узкий; для широкого принтера заданное значение будет автоматически пересчитано);

l — число строк на дюйм (6 или 8).

Умолчания для перечисленных аргументов отсутствуют. Если *c* и/или *l* не специфицированы, то действуют ранее установленные режимы.

При задании последнего аргумента (P) загружается и остается резидентным код, имитирующий ответ пользователя R (от Retry — повторить) на сообщение об ошибке ввода-вывода, связанной с тайм-аутом (неготовностью принтера принять кодовую последовательность). Тайм-аут, в частности, возникает на устройствах, не обеспечивающих автоматическую подачу листов бумаги, при окончании печати очередной страницы. Если аргумент P в команде не указан, то при возникновении этой ситуации через небольшой интервал времени на экране появится сообщение об ошибке и пользователю каждый раз после заправки очередного листа придется вводить на это сообщение ответ R, что затрудняет его работу. Задание же аргумента P освободит его от многократного выполнения описанной процедуры. Однако при возникновении настоящей ошибки на принтере необходимо отменить действие аргумента P. Для этого достаточно нажать комбинацию клавиш Ctrl-Break. Действие аргумента P отменяется также повторной выдачей команды MODE без его указания.

Замечания:
— принтер обычно подключается к адаптеру LPT1;
— синоним PRN в LPT1 для данном контексте недопустим;
— задание аргументов *c* и *l* оказывает воздействие только на IBM-совместимые принтеры;
— установка режимов работы принтера приводит к невозможности последующего перенаправления вывода, ему адресованного;
— указанный в команде MODE адаптер становится системным принтерным адаптером (PRN);
— при включении питания принтер устанавливается в режим работы, заданный его переключателями, так что команда MODE необходима только для оперативного изменения режима;
— при отключении питания принтера установленные командой MODE режимы, как правило, сбрасываются;
— повторная выдача команды MODE переустанавливает заданные ранее режимы, если соответствующие аргументы указаны.

Пример:

- MODE LPT1: 80,8,P — установить для принтера, подключенного к LPT1, режим, при котором печатаются 80 символов в строке, 8 строк на дюйм, а также задать автоматический режим повторения передачи информации в случае неготовности принтера.

DOS 4.0. Синтаксис:

MODE LPT*n*[:][*c*][,[*l*][,*r*]]

или

MODE LPT*n* [COLS=*c*] [LINES=*l*] [RETRY=*r*]

Две приведенные формы семантически полностью эквивалентны, но вторая обладает большей наглядностью из-за ключевых, а не позиционных параметров. Смысл аргументов *c* и *l* не меняется.

Аргумент же *r* обобщает последний аргумент команды MODE для DOS версии 3.3. При задании *r* в памяти резидентно остается код, перекодирующий ошибку тайм-аута на принтере в соответствии с указанным значением и возвращающий результат в процесс, выводящий информацию на принтер. Реакция на результат перекодировки должна быть указана в этом процессе.

В качестве *r* можно задать:

E (для Error — ошибка)	— возвратить ошибку (принимается по умолчанию);
B (для Busy — занятость)	— возвратить сигнал занятости;
R (для Ready — готовность)	— возвратить сигнал готовности;
NONE	— ничего не предпринимать

Значение B эквивалентно указанию P в команде для DOS версии 3.3. При работе в сети аргумент *r* указывать нельзя.

Пример:

- MODE LPT1 COLS=80 LINES=8 RETRY=B — то же, что и раньше.

Команда MODE: управление дисплеем

Назначение: активизация дисплейного адаптера, задание режима работы дисплея и его адаптера, а также центровка экрана.

Синтаксис:

MODE *режим*

или

MODE [*режим*], *направление* [,T]

Комментарии. Эта команда позволяет устанавливать один из допустимых текстовых режимов работы видеосистемы. Если ПЭВМ оборудована как MDA, так и CGA (EGA, VGA), то задание режима косвенно определяет активизацию соответствующего адаптера.

В качестве режима можно задать 40, 80, BW40, BW80, CO40, CO80 или MONO. Число в этих значениях (40 или 80) определяет количество выводимых в строке символов (40 задает отображение широких символов); BW и CO указывают на необходимость вывода в монохромном (с различными градациями яркости) и цветном режиме соответственно. Все значения, кроме MONO, применимы только к адаптерам CGA (EGA, VGA). MONO специфицирует MDA с неизменяемым числом символов в строке (80).

Если изображение на экране сдвинуто так, что не видно левых или правых символов в каждой строке, то необходимо для центровки экрана выполнить команду MODE во второй форме. При этом в качестве направления можно указать L (для сдвига изображения влево на 2 символа) или R (то же, но вправо). Если дополнительно специфицировать T, то устанавливается диалоговый режим сдвига влево (вправо). В этом случае на экран выводится текстовая строка вида

0123456789 . . . 0123456789

и задается вопрос о том, видна ли правая (левая) ее граница. Отвечая на этот вопрос соответствующим образом, можно обеспечить требуемую центровку.

Одновременно с центровкой экрана вторая форма команды MODE позволяет установить также один из допустимых текстовых режимов работы видеосистемы.

Замечания:

— команда MODE не поддерживает всех текстовых режимов работы EGA и VGA (для них только эмулируются режимы CGA);

— режимы BW используются тогда, когда ПЭВМ оборудована CGA и дешевым монохромным дисплеем. Они также полезны и в других случаях, когда текст на экране невиден. Если установка ни одного из таких режимов эффекта не дала, то можно попробовать нажать клавишу F7.

Примеры:

- MODE BW80 — установить режим BW80;
- MODE ,L — сдвинуть экран влево на две колонки.

DOS 4.0. Синтаксис:

MODE *режим, n*

или

MODE CON[:] [COLS=*m*] [LINES=*n*]

или

MODE [*режим*], *направление*[,T]

Первые две из приведенных форм задания команды семантически эквивалентны и устанавливают один из текстовых режимов работы видеосистемы, включая все текстовые режимы EGA и VGA. Собственно *режим* имеет те же значения, что и для DOS 3.3, а *n* определяет количество строк на экране (25, 43 или 50). Аргумент *m* специфицирует число символов в строке (40 или 80).

Третья форма команды идентична второй форме для DOS 3.3.

Команда MODE: управление адаптером последовательного интерфейса

Назначение: задание режима работы адаптера последовательного интерфейса.
Синтаксис:

MODE COM*n*: *b*[,[*p*][,[*l*][,[*s*][,P]]]]

Комментарии. Эта разновидность команды MODE обеспечивает конфигурирование (установку протокола) адаптера последовательного интерфейса, к которому могут быть подключены медленнодействующий принтер (возможно, обеспечивающий высококачественную печать), графопостроитель, модем (модулятор-демодулятор) или другое медленнодействующее ПУ.

В командной строке можно задать следующие аргументы:

n — номер адаптера последовательного интерфейса (1, 2, 3 или 4);

b — скорость передачи данных в бодах, т.е. бит/с (110, 150, 300, 600, 1200, 2400, 4800 или 9600);

p — режим контроля правильности передачи (N — контрольный бит отсутствует, O — контроль по нечетности, E — контроль по четности). По умолчанию — E;

l — разрядность кода, т.е. число передаваемых битов данных (7 или 8). По умолчанию — 7;

s — число стоп-битов при передаче данных (1 или 2). По умолчанию — 2 для скорости 110 и 1 в противном случае;

P — указывает, что к адаптеру подключен принтер, и обозначает то же, что и при управлении принтером, подсоединенным к адаптеру параллельного интерфейса.

Как минимум необходимо задать только скорость передачи данных, а остальные значения могут быть приняты по умолчанию.

Замечания:

— многие программы, использующие адаптер последовательного интерфейса, самостоятельно устанавливают его протокол, так что команда MODE для этого используется редко;

— команда MODE тем не менее применяется обычно перед перенаправлением принтерного вывода и для инициализации пассивного терминала, подсоединенного к адаптеру последовательного интерфейса (см. описание команды CTTY);

— конкретные значения аргументов для команды MODE следует взять из документации на устройство, связанное с адаптером последовательного интерфейса.

Примеры:

- MODE COM1:1200,N,8,1,P — установить для COM1 скорость передачи данных 1200 бод без контроля правильности передачи, 8 битов данных и 1 стоп-бит, причем адаптер используется для связи с принтером;
- MODE COM1:9600,,,,P — установить скорость передачи данных 9600 бод, считать, что к адаптеру подключен принтер, а остальные аргументы принять по умолчанию.

DOS 4.0. Синтаксис:

MODE COM*n*[:] *b*[,[*p*][,[*l*][,[*s*][,*r*]]]]

или

MODE COM*n* BAUD=*b* [PARITY=*p*] [DATA=*l*] [STOP=*s*] [RETRY=*r*]

Приведенные формы семантически эквивалентны. Аргументы *n*, *b*, *p*, *l* и *s* имеют тот же смысл, однако возможны дополнительные значения:

b — 19200, причем при любом значении указывать достаточно только первые две цифры;
p — M (контроль по маркеру) и S (контроль по пробелу);
l — 5 и 6;
s — 1.5.

Аргумент *r* имеет тот же смысл, что и в команде MODE для управления принтером, подключенным к адаптеру параллельного интерфейса.

Команда MODE: перенаправление принтерного вывода

Назначение: перенаправление принтерного вывода с адаптера параллельного интерфейса на адаптер последовательного интерфейса.
Синтаксис:

MODE LPT*n*:[=COM*m*]

Комментарии. Этот вид команды MODE перенаправляет принтерный вывод с *n*-го адаптера параллельного интерфейса на *m*-й адаптер последовательного интерфейса (*n* = 1, 2 или 3, *m* = 1, 2, 3 или 4). Если конструкция =COM*m* не задана, то отменяется сделанное ранее перенаправление принтерного вывода.

Замечания:
— перенаправление принтерного вывода используется только перед выполнением неграмотно составленных программ, не учитывающих конфигурацию оборудования;
— синоним PRN для LPT1 недопустим;
— перед перенаправлением может потребоваться установить протокол адаптера последовательного интерфейса;
— после перенаправления адаптер последовательного интерфейса становится системным принтерным адаптером (PRN).

Пример: MODE LPT1:=COM3.

DOS 4.0. Синтаксис:

MODE LPT*n*[:][=COM*m*]

Команда MODE: управление клавиатурой

Назначение: установка скорости работы клавиатуры (только в DOS 4.0).
Синтаксис:

MODE CON[:] DELAY=*l* RATE=*r*

Комментарии. Этой командой устанавливаются *задержка регенерации* (по значению *l*), а также *частота регенерации* (по значению *r*) кода удерживаемой в нажатом состоянии клавиши. Величины *l* и *r* задаются целыми числами в диапазонах 1 — 4 и 1 — 32 соответственно. Задержка регенерации и частота регенерации принимаются равными обратным значениям *l* и *r* (1/*l* и 1/*r*) в секундах.

Пример: MODE CON DELAY=1 RATE=1

Команда MODE: отображение статуса устройств

Назначение: отображение статуса всех или заданного посимвольного устройства (только в DOS 4.0).
Синтаксис:

MODE [*устройство*] [/STA[TUS]]

Комментарии. Команда MODE без аргумента выводит на экран дисплея статус всех подключенных к ПЭВМ посимвольных устройств. При задании имени устройства выдается статус только указанного устройства. Переключатель /STATUS (сокращение /STA) необходим лишь при спецификации адаптера параллельного интерфейса, чтобы исключить неоднозначность.
Пример: MODE CON

Команда MODE: поддержка кодовых страниц

Назначение: подготовка кодовых страниц, а также активизация, отображение и восстановление активной кодовой страницы для посимвольного устройства.
Синтаксис:

MODE *устройство* CODEPAGE PREPARE=((*cp*[,*cp*]...)*file*)

или

MODE *устройство* CODEPAGE SELECT=*cp*

или

MODE *устройство* CODEPAGE [/STATUS]

или

MODE *устройство* CODEPAGE REFRESH

Комментарии. Периферийное оборудование ПЭВМ и сама DOS обеспечивают возможность работы с различными национальными алфавитами. Необходимость рассмотрения вопросов, связанных с настройкой DOS для использования в той или иной стране, диктуется недавней регистрацией кодовой страницы для СССР (теперь — СНГ) и потребностью некоторых пользователей в подготовке текстов на языках, отличных от английского и русского. Вместе с тем упомянутые вопросы в отечественной литературе практически еще не затрагивались. Поэтому мы сочли необходимым остановиться на этом достаточно подробно.

Кодовая страница содержит таблицу, являющуюся расширением ASCII (см. подраздел 5.3). Она ставит в соответствие символам на УВВ определенные двоичные коды. Так, если код n представляет в таблице символ S, то это соответствие должно быть установлено как для устройств ввода (преобразование S в n), так и для устройств вывода (преобразование n в S).

Каждая кодовая страница служит для работы с одним или несколькими национальными алфавитами.

Различают два типа кодовых страниц: аппаратные и составленные. **Аппаратной** называется кодовая страница, встроенная в устройство. Устройство может иметь одну или несколько таких страниц. **Составленная** кодовая страница находится в файле, может загружаться в память и

определять функционирование ПУ (правила перекодировки в нем). Работу с такими страницами поддерживают не все УВВ.

DOS содержит следующие файлы с составленными кодовыми страницами для конкретных ПУ:

4201.CPI — для модели 4201 принтера IBM Proprinter и совместимых с ним;
5202.CPI — для модели 5202 принтера IBM Quetwriter III и совместимых с ним;
EGA.CPI — для дисплейного адаптера EGA и совместимых с ним;
LCD.CPI — для жидкокристаллического дисплея ПЭВМ IBM PC Convertible.

В DOS поддерживаются следующие составленные кодовые страницы:

437 — для США (принимается по умолчанию для всех команд, требующих задания кодовой страницы). Именно ее таблица приведена в качестве расширения ASCII в подразделе 5.3;

850 — многоязыковая кодовая страница, включающая наиболее часто встречающиеся буквы многих европейских, северо- и южноамериканских языков;

860 — для Португалии;

863 — для Канады (французский язык);

865 — для Норвегии и Дании.

Для той или иной страны специфическими являются не только алфавит языка, но и *формат даты, формат времени, формат денежной единицы, разделители в числах,* а также *алфавитный порядок букв.* Перечисленные характеристики устанавливаются в DOS с использованием **кода страны**, применяемого в международной телефонной системе.

Возможность использования национального регистра клавиатуры определяется в DOS не кодовой страницей и не кодом страны, а двухбуквенным **кодом клавиатуры**.

При выполнении команд DOS, обеспечивающих поддержку национальных языков, всегда проверяется допустимость сочетания составленной кодовой страницы, кода страны и кода клавиатуры. Все поддерживаемые DOS их комбинации приведены в табл. 5.18 вместе с соответствующими форматами даты и времени (на примере 17 час. 35 мин. 15 января 1991 г.). Соответствующими командами DOS не обязательно определять названную *триаду* полностью, но если Вы это делаете, то следует воспользоваться представленной таблицей.

Для максимально полной настройки оборудования ПЭВМ и DOS на использование в зарегистрированной стране, кроме США, требуется:

1) *установить код страны* командой конфигурирования системы COUNTRY=;

2) *задать активные аппаратные кодовые страницы* устройств и *выделить буфера для составленных кодовых страниц* командами конфигурирования системы DEVICE=;

3) *загрузить в память* командой NLSFUNC *резидентные программные средства национальной поддержки*;

4) *подготовить (загрузить в память) составленные кодовые страницы* для каждого устройства, допускающего переключение кодовых страниц, командой MODE с ключевым словом PREPARE;

5) *настроить клавиатуру для поддержки национального алфавита* командой KEYB;

6) *выбрать из подготовленных кодовых страниц активную* командой CHCP (сразу для всех устройств) или командами MODE с ключевым словом SELECT (для каждого устройства в отдельности).

При желании можно ограничиться только пунктом 1 для задания приемлемого формата даты и времени. Пункты 5 и 6 можно поменять местами. Имеется возможность подготовки сразу нескольких кодовых страниц с тем, чтобы затем оперативно осуществлять переключение активности между ними, повторяя пункт 6. Команды COUNTRY= и DEVICE= допустимы только в файле CONFIG.SYS. Остальные в принципе можно ввести с клавиатуры, но обычно их размещают в файле AUTOEXEC.BAT или в другом командном файле.

Незаданные командами DOS элементы триады устанавливаются следующим образом: код страны — 001, код клавиатуры — US, а кодовые страницы — в соответствии с переключателями в устройствах (для дисплея обычно 437).

Часто (например, при отключении питания на устройстве) возникает потребность в восстановлении активной кодовой страницы, что можно осуществить командой MODE с ключевым словом REFRESH.

Очевидно, команда MODE играет центральную роль в национальной поддержке.

В команде используются следующие аргументы:

устройство — имя одного из устройств CON, LPT1 (PRN), LPT2 или LPT3;

cp — номер кодовой страницы;

file — спецификация CPI-файла, соответствующего заданному устройству.

Команда в *первой форме* (PREPARE) обеспечивает подготовку одной или нескольких составленных кодовых страниц для указанного устройства.

Команда во *второй форме* (SELECT) служит для выбора из подготовленных кодовых страниц единственной, которая становится активной для заданного устройства.

Таблица 5.18

Соответствие кодов для различных стран

Страна или язык	Код страны	Формат даты	Формат времени	Номера кодовых страниц	Код клавиатуры
Английский (международный), Австралия	061	01-15-1991	5:35:00.00p	437, 850	-
Арабский	785	15/01/1991	5:35:00,00p	437	-
Бельгия	032	15/01/1991	17:35:00,00	437, 850	BE
Великобритания	044	15-01-1991	5:35:00.00p	437, 850	UK
Германия	049	15.01.1991	17.35.00,00	437, 850	GR
Дания	045	15-01-1991	17.35.00,00	865, 850	DK
Израиль	972	15 01 1991	17:35:00.00	437,	-
Испания	034	15/01/1991	17:35:00,00	437, 850	SP
Италия	039	15/01/1991	17:35:00,00	437, 850	IT
Канада (французский)	002	15-01-1991	17:35:00,00	863, 850	CF
Латинская Америка	003	15/01/1991	17:35:00.00	437, 850	LA
Нидерланды	031	15-01-1991	17:35:00,00	437, 850	NL
Норвегия	047	15.01.1991	17.35.00,00	865, 850	NO
Португалия	351	15/01/1991	17:35:00,00	860, 850	PO
США, Канада (английский)	001	01-15-1991	5:35:00.00p	437, 850	US
Финляндия	358	15.01.1991	17.35.00,00	437, 850	SU
Франция	033	15/01/1991	17:35:00,00	437, 850	FR
Швейцария (немецкий)	041	15.01.1991	17.35.00.00	437, 850	SG
Швейцария (французский)	041	15.01.1991	17.35.00.00	437, 850	SF
Швеция	046	1991-01-15	17.35.00,00	437, 850	SV

Третья форма команды (/STATUS) обеспечивает выдачу на экран номера активной кодовой страницы для специфицированного устройства. Необязательный переключатель /STATUS при этом абсолютно никакой роли не играет.

И наконец, *четвертая форма* команды (REFRESH) служит для восстановления (при необходимости) активной кодовой страницы указанного устройства.

Замечания:

— не следует возлагать слишком большие надежды на средства DOS для национальной поддержки, так как в любом случае сообщения будут выдаваться на английском языке (если DOS не локализована) и использование национального алфавита на уровне DOS (в частности, для именования файлов) не станет возможным;

— в СНГ для поддержки русского алфавита пока, как правило, используются специально разработанные внешние драйверы, однако уже начали поставляться версии DOS с кодовой страницей для нашего содружества;

— видеосистема CGA не предоставляет возможности переключения кодовых страниц (однако кое-что можно сделать командой GRAFTABL).

Примеры:

- MODE PRN CODEPAGE PREPARE=((863)C:\DOS33\4201.CPI) — подготовить кодовую страницу 863 для принтера IBM Proprinter;
- MODE CON CODEPAGE PREPARE=((863,850)C:\DOS33\EGA.CPI) — подготовить кодовые страницы 863 и 850 для адаптера EGA;
- MODE CON CODEPAGE SELECT=863 — активизировать для дисплея кодовую страницу 863.

DOS 4.0. Добавлен файл 4208.CPI для модели 4208 принтера IBM Proprinter. Файл EGA.CPI доработан с тем, чтобы поддерживать дисплейные адаптеры ПЭВМ семейства PS/2 и совместимые с ними.

Дополнительные кодовые страницы, коды стран и коды клавиатур, которые могут поддерживаться (но не все новые кодовые страницы обязательно поддерживаются) DOS 4.0, сведены в табл. 5.19.

Таблица 5.19

Соответствие дополнительных для DOS 4.0 кодов

Страна	Код страны	Формат даты	Формат времени	Номера кодовых страниц	Код клавиатуры
Япония	081	1991-01-15	17:35:00.00	932, 850, 437	JA
Корея	082	1991-01-15	17:35:00.00	934, 850, 437	KO
КНР	086	1991-01-15	17:35:00.00	936, 850, 437	CH
Тайвань	088	01-15-1991	17:35:00.00	938, 850, 437	TN

Можно использовать следующие сокращения:

CP — вместо CODEPAGE

/STA — вместо /STATUS

PREP — вместо PREPARE

SEL — вместо SELECT

REF — вместо REFRESH

Команда NLSFUNC

Назначение: загрузка резидентных средств национальной поддержки.
Тип: внешняя.
Синтаксис:

NLSFUNC [*file*]

Комментарии. Аргумент *file* указывает файл, содержимым которого следует воспользоваться для учета национальных соглашений. Если он не задан, то отыскивается файл, указанный в команде конфигурирования COUNTRY=. В DOS такую информацию содержит файл COUNTRY.SYS. Поэтому, как правило, специфицируют именно его, однако в принципе возможно использовать и любой другой файл, предназначенный для этой же цели. Если аргумент *file* не задан и команда COUNTRY= в файле CONFIG.SYS отсутствует, то национальная специфика использоваться не будет.

Замечания:
— команда NLSFUNC дополняет команду COUNTRY= и является обязательной, если будут использоваться команды MODE и CHCP для подготовки и выбора кодовых страниц (см. описание команды MODE: поддержка кодовых страниц);
— выполнение команды приводит к увеличению резидентной части DOS.

Пример: NLSFUNC C:\DOS33\COUNTRY.SYS

Команда CHCP

Назначение: отображение номера и выбор (смена) активной кодовой страницы для максимально возможного числа устройств (иными словами — для КП DOS).
Тип: внутренняя.
Синтаксис:

CHCP [*cp*]

Комментарии. Аргумент *cp* задает номер кодовой страницы, которую предписывается сделать активной. Если указанная кодовая страница не подготовлена командой MODE ни для одного из устройств, то на экране появится сообщение об ошибке, например:

Code page 850 not prepared for system
(Кодовая страница 850 для системы не подготовлена)
Active code page: 437
(Активная кодовая страница: 437)
Prepared system code pages: 437 865
(Подготовленные системные кодовые страницы: 437 865)

В случае, когда активизируемая кодовая страница не подготовлена для какого-либо устройства (а для других — подготовлена), на экран дисплея выдается сообщение, аналогичное следующему:

Code page 850 not prepared for device CON
(Кодовая страница 850 не подготовлена для устройства CON)

Для других же устройств заданная кодовая страница будет активизирована.

Команда CHCP без аргумента служит для получения информации о подготовленных и активной кодовых страницах для КП DOS (эти сведения не обязательно совпадают с выводом команды вида MODE *устройство*). В этом случае CHCP выдает сообщение типа

Active code page: 850
(Активная кодовая страница: 850)
Prepared system code pages: 850 437
(Подготовленные системные кодовые страницы: 850 437)

Замечания:

— перед использованием CHCP должна быть выполнена команда NLSFUNC;

— любая программа, запущенная после активизации новой кодовой страницы, будет использовать именно ее. Однако все программы, запущенные до этого, будут работать со старой кодовой страницей;

— дополнительные сведения, в том числе допустимые номера кодовых страниц, содержатся в описании команды MODE: поддержка кодовых страниц.

Пример:

- CHCP 863 — активизировать кодовую страницу 863 (для Канады, французский язык).

Команда KEYB

Назначение: настройка клавиатуры на национальный алфавит и отображение двухбуквенного кода клавиатуры.

Тип: внешняя.

Синтаксис:

KEYB [*код*[,[*cp*][,*file*]]]

Комментарии. Выполнение этой команды приводит к:

— загрузке в ОЗУ специального драйвера клавиатуры (по спецификации *file*);

— переключению клавиатуры на национальный регистр в соответствии с заданным *кодом*;

— установке раскладки клавиатуры (размещения клавиш), принятой в заданной кодом стране;

— активизации одной из кодовых страниц, допустимых для данной страны (аргумент *cp*), с тем, чтобы определять кодировку клавиш.

Если номер *cp* не задан, то активной для клавиатуры становится кодовая страница, принимаемая для каждой страны по умолчанию. В случае отсутствия аргумента *file* загружается и подключается к системе драйвер клавиатуры из файла KEYBOARD.SYS, содержащегося в корневом каталоге системного диска (конечно, именно этот файл используется в большинстве стандартных случаев). Если он находится в другом каталоге, то аргумент *file* следует указать.

При отсутствии всех аргументов команда выдает на экран действующий код клавиатуры и номер активизированной кодовой страницы, например:

Current keyboard code: FR code page: 437
(Текущий код клавиатуры: FR, кодовая страница: 437)
Current CON code page: 437
(Текущая кодовая страница для CON: 437)

Возврат к американскому регистру и принятой в США раскладке клавиатуры после выдачи команды KEYB всегда возможен и осуществляется путем нажатия комбинации клавиш Ctrl-Alt-F1. Для восстановления национального регистра клавиатуры следует нажать комбинацию клавиш Ctrl-Alt-F2.

Команда KEYB вырабатывает следующие коды возврата:

0 — успешное завершение;
1 — ошибочный синтаксис;
2 — файл с драйвером клавиатуры ошибочен или отсутствует;
3 — невозможно создать таблицу клавиатуры в резидентной памяти;
4 — ошибка на устройстве CON;
5 — запрашиваемая кодовая страница не подготовлена;
6 — таблица для выбираемой кодовой страницы в резидентной таблице клавиатуры не найдена;
7 — некорректная версия DOS (команда KEYB выдана в среде старой версии DOS).

Замечания:

— команду KEYB целесообразно использовать тогда, когда клавиши клавиатуры ПЭВМ помечены соответствующими символами. Иногда некоторые национальные символы вводятся комбинацией Ctrl-Alt-*клавиша* или последовательным нажатием двух клавиш;

— команда KEYB может выдаваться многократно без перезагрузки DOS;

— выполнение команды приводит к увеличению резидентной части DOS;
— дополнительные сведения, включая коды национальных клавиатур, содержатся в описании команды MODE: поддержка кодовых страниц.

Пример:

- KEYB UK,,C:\DOS33\KEYBOARD.SYS — переключить клавиатуру на английский регистр и установить раскладку клавиатуры для использования в Великобритании.

Команда GRAPHICS

Назначение: подготовка принтера для печати в графическом режиме.
Тип: внешняя.
Синтаксис:

GRAPHICS [*принтер*] [/R] [/B] [/LCD]

Комментарии. Обычно (после включения питания) принтер устанавливается в текстовый режим. Программные продукты, обеспечивающие печать графики, переводят его в графический режим самостоятельно. Команда же GRAPHICS необходима только в том случае, когда видеосистема находится в графическом режиме и Вам требуется отпечатать копию экрана (путем нажатия комбинации клавиш Shift-PrtSc). Эта команда должна быть выполнена до печати содержимого экрана, а ее действие распространяется на все последующие операции печати. Если команда GRAPHICS не выдана, то обеспечивается печать копии экрана, находящегося в текстовом режиме.

Аргументом *принтер* можно задать следующие типы печатающих устройств:

GRAPHICS — точечно-матричные принтеры IBM Graphics, IBM Proprinter, IBM Quietwriter или фирмы Epson, а также совместимые с ними (этот тип принимается по умолчанию);

COLOR1 — точечно-матричный принтер IBM Color с одноцветной (черной) красящей лентой или совместимый с ним;

COLOR4 — тот же принтер, но с RGB-красящей лентой (красный, зеленый, голубой и черный цвета);

COLOR8 — тот же принтер, но с CMY-красящей лентой (бирюзовый, фиолетовый, желтый и черный цвета);

COMPACT — принтер IBM Compact или совместимый с ним.

Факультативные переключатели в командной строке имеют следующий смысл:

/R — обеспечить печать позитивного изображения, как видно на экране (белое — белым, а черное — черным). Если /R не задан, то будет печататься негативное изображение (белое — черным, а черное — белым), что во многих случаях предпочтительнее;

/B — обеспечить печать на цветном принтере фонового цвета (иначе фон печататься не будет). Этот переключатель допустим только для аргументов COLOR4 и COLOR8;

/LCD — обеспечить печать изображения с учетом коэффициента сжатия (соотношения разрешающей способности по горизонтали и вертикали) жидкокристаллического дисплея, а не дисплея на базе электронно-лучевой трубки, чтобы отпечатанное изображение было пропорциональным.

Замечания:
— выполнение команды GRAPHICS приводит к увеличению размера резидентной части DOS;
— команда GRAPHICS поддерживает печать содержимого экрана, только если дисплей находится в одном из графических режимов CGA. В режиме с разрешением 320x200 точек на принтерах GRAPHICS и COLOR1 изображение будет выводиться с четырьмя градациями серого. Если установлен режим с разрешением 640x200, то двухцветное изображение будет печататься с поворотом на 90°, чтобы разместить картинку на узком принтере;
— для перевода принтера снова в текстовый режим обычно достаточно отключить и включить питание на нем. Можно также выполнить рестарт DOS;
— имеются дополнительные независимые программы-драйверы, поддерживающие современные дисплейные адаптеры, например EGAEPSON и EGAPRINT.

Пример:

- GRAPHICS COLOR4 /R — подготовить принтер типа COLOR4 для печати копии CGA-экрана в графическом режиме в позитивном виде без воспроизведения фона.

DOS 4.0. Синтаксис:

GRAPHICS [*принтер*] [*file*] [/R] [/B] [/LCD] [/PRINTBOX:*id*]

Переключатели /R, /B и /LCD здесь имеют то же назначение.

Дополнительно поддерживается тип *принтера* GRAPHICSWIDE для принтера IBM Graphics с шириной каретки в 11 дюймов (широкий принтер) или совместимого с ним. Взамен типа COMPACT используется THERMAL (термопринтер ПЭВМ IBM PC Convertible или совместимый с ним).

Аргумент *file* определяет *профильный файл*, содержащий информацию о всех поддерживаемых принтерах в текстовом виде. Если аргумент опущен, то используется файл GRAPHICS.PRO из корневого каталога системного диска (он имеется в DOS 4.0).

Новый переключатель /PRINTBOX:*id* обеспечивает повышенную гибкость управления печатью в зависимости от коэффициента сжатия дисплея. Значение *id* здесь должно совпадать с первым аргументом любого из утверждений PRINTBOX из *профиля*, относящихся к заданному типу принтера. Пока в нем зарегистрированы значения STD (стандартный) и LCD (жидкокристаллический), однако есть большие потенциальные возможности для расширений. Переключатель в виде /PRINTBOX:LCD заменяет /LCD, а в виде /PRINTBOX:STD противоречит указанию /LCD. Для ключевого слова PRINBOX допустимо сокращение PB.

Поддерживаются все графические режимы адаптеров EGA, VGA и 8514/A.

Команда GRAFTABL

Назначение: обеспечение отображения расширения ASCII в графических режимах адаптера CGA.

Тип: внешняя.

Синтаксис:

GRAFTABL [*cp* | /STATUS]

Комментарии. Адаптер CGA, находясь в графическом режиме, обычно не позволяет выводить на экран символы из расширения ASCII, т.е. символы с кодами 128 — 255. Вместо них отображается «мусор». После выполнения команды GRAFTABL все становится на свои места: любая программа сможет выдавать на экран символы с такими кодами и они будут отображаться соответствующим образом (но, конечно же, в графическом режиме).

Если в командной строке задан аргумент *cp* (номер кодовой страницы), то будет использоваться кодировка символов из указанной кодовой страницы (см. описание команды MODE: поддержка кодовых страниц).

В случае, когда в командной строке кроме имени команды ничего не задано, будет использована кодовая страница по умолчанию, т.е. 437.

Указание в командной строке переключателя /STATUS приводит к выдаче на экран дисплея номера активной кодовой страницы.

Команда GRAFTABL генерирует следующие коды возврата:

0 — команда выполнена успешно, причем шрифты (кодовая страница) загружены впервые;
1 — шрифты были уже загружены ранее и поэтому заменены новыми;
2 — обнаружена ошибка;
3 — некорректный аргумент, вследствие чего никаких действий не выполнено;
4 — некорректная версия DOS (команда введена в среде старой версии DOS).

Замечания:

— команда GRAFTABL может использоваться многократно для смены активной кодовой страницы;

— для адаптеров, отличных от CGA, выдавать команду GRAFTABL нет необходимости;

— выполнение команды GRAFTABL приводит к увеличению размера резидентной части DOS.

Пример:

- GRAFTABL 850 — загрузить шрифты из кодовой страницы 850.

DOS 4.0. Синтаксис:

GRAFTABL [*cp* | /STA[TUS] | ?]

Аргумент *cp* и переключатель /STATUS здесь имеют тот же смысл, причем для последнего допустимо сокращение /STA. Указание в качестве аргумента символа ? приводит к выдаче на экран справки-инструкции о порядке использования данной команды.

5.6.5. Команды реконфигурирования системы

Команда SET

Назначение: установка значения глобальной переменной в окружении DOS и отображение окружения.

Тип: внутренняя.

Синтаксис:

SET [*string1* = [*string2*]]

КомМентарии. В командной строке *string1* представляет собой имя глобальной переменной, а *string2* — ее значение. После ввода команды (в случае задания как *string1*, так и *string2*) конструкция *string1* = *string2* записывается в качестве строки окружения DOS и может использоваться любыми выполняемыми программами для получения *string2* по имени *string1* (см. п. 5.2.5) с целью

настройки на конкретные условия работы. Если глобальная переменная *string1* уже представлена в окружении, то соответствующая ей строка окружения заменяется новой. Поэтому существует прямая аналогия между командой SET и оператором присваивания в языках программирования (но не в плане реализации этих средств).

Когда значение глобальной переменной *(string2)* не задано, то идентифицированная посредством *string1* строка окружения удаляется и вследствие этого *string1* теряет свое значение.

Если в командной строке кроме имени команды ничего не указано, то на экран построчно выводится содержимое окружения DOS.

Имя *string1* не должно содержать пробелов и символов равенства. На значение *string2* не накладывается никаких ограничений (воспринимаются все символы вплоть до нажатия клавиши Enter).

Замечания:

— глобальные переменные обычно служат для указания маршрутов, по которым выполняемые программы должны искать необходимые им файлы, а также для настройки программ (анализ значения той или иной глобальной переменной в этих случаях предусматривается в самой программе);

— окружение имеет формат текстового файла;

— при подзагрузке транзитного модуля КП система пользуется значением глобальной переменной COMSPEC, указывающим полную спецификацию файла COMMAND.COM. После загрузки DOS COMSPEC определяет файл COMMAND.COM в корневом каталоге системного диска, если иное не задано в команде конфигурирования SHELL=. Поэтому в случае, когда Вы решили с целью повышения скорости работы DOS скопировать его на виртуальный диск и использовать созданную копию, следует соответствующим образом установить значение переменной COMSPEC;

— команды SET обычно помещают в файл AUTOEXEC.BAT или в другой командный файл;

— глобальные переменные могут использоваться не только в программах, но и в командных файлах, что существенно повышает их гибкость;

— после ввода команды SET имя глобальной переменной перекодируется так, что строчные буквы заменяются на прописные, а ее значение записывается в окружение без каких-либо изменений;

— если при выполнении очередной команды SET на экран будет выдано сообщение «Out of Environment Space» («Нет памяти в окружении»), то для выделения большего объема памяти можно воспользоваться одним из следующих методов: выполнить команду COMMAND с требуемым значением в переключателе /E; поместить в файл CONFIG.SYS команду SHELL=COMMAND.COM с требуемым значением в переключателе /E и произвести рестарт системы; выполнить все команды SET перед загрузкой первой резидентной программы, для чего можно даже установить фиктивные значения глобальных переменных, чтобы зарезервировать память в окружении. При использовании последнего метода окружение автоматически будет расширено, сообщение об ошибке не появится, а впоследствии окружение не будет перекрываться выполняемыми программами;

— в среде используемой Вами оболочки DOS, например Norton Commander'a, выдавать команду SET для изменения окружения не следует, так как любая программа (в том числе и оболочка) использует копию окружения, а не оригинал;

— глобальные переменные наряду с командой SET устанавливают команды PATH, APPEND с переключателем /E, PROMPT и COMMAND с переключателем /P.

Пример: SET HELPPATH=C:\UTILS\HELP

Команда PATH

Назначение: установка и отображение маршрутов поиска *исполняемых* файлов.
Тип: внутренняя.
Синтаксис:

PATH [*dir*[;*dir*]...]

или

PATH ;

Комментарии. Данная команда в принципе является частным случаем SET, устанавливая и отображая в окружении DOS значение глобальной переменной PATH, однако в отличие от SET команда PATH перекодирует все строчные буквы в своем вводе в прописные. Оно используется при поиске исполняемых файлов (COM-, EXE- и BAT-файлов) в случае их запуска на выполнение.

Исполняемый файл сначала ищется DOS в выделенном по его спецификации каталоге, а затем в каждом заданном командой PATH каталоге *dir* в порядке их перечисления в командной строке.

Каждая последующая команда PATH с каталогами (маршрутами поиска) *dir* отменяет действие предыдущей команды.

Если введено только имя команды, то на экран дисплея выводятся текущие (установленные последними) маршруты поиска исполняемых файлов.

Команда PATH во второй форме отменяет все маршруты поиска.

Замечания:

— длина строки, задающей маршруты поиска исполняемых файлов, не должна превышать 127 символов;

— до выдачи первой команды PATH никакие маршруты поиска не действуют;
— помещать пробелы между спецификациями *dir* в командной строке не допускается;
— полезность команды PATH состоит в том, что после ее выполнения полные спецификации исполняемых файлов можно не указывать (достаточно задать имя файла), причем для этого нет необходимости изменять текущий каталог;
— задавать маршруты поиска исполняемых файлов следует рационально, чтобы существенно не снизить производительность DOS (иными словами, не создавайте слишком большой список маршрутов и определяйте часто используемые маршруты первыми);
— утилиты DOS целесообразно поместить в один каталог и установить маршрут поиска в нем, чтобы внешние команды можно было выполнять без указания маршрута перед их именами независимо от того, какой каталог в данный момент является рабочим;
— команду PATH обычно помещают в файл AUTOEXEC.BAT;
— в среде оболочки DOS выдавать команду PATH не следует;
— если ПЭВМ снабжена дополнительной памятью, то целесообразно создать в ней виртуальный диск, скопировать на него часто используемые программные продукты и установить маршрут поиска исполняемых файлов на этом диске;
— средства программирования командных файлов позволяют не только заменять и удалять маршруты поиска, но и добавлять их к уже установленным;
— команда SET оказывается бесполезной для программ, имеющих свои *оверлеи* и другие *файлы данных*, размещенные в других каталогах. Такие программы используют специальные методы, в числе которых: извлечение маршрута запуска программы из поля, непосредственно следующего за дубликатом окружения (см. п. 5.2.5); явный просмотр значения глобальной переменной PATH; получение информации по значению какой-либо глобальной переменной, которое предварительно должно быть установлено пользователем. Вместе с тем команда APPEND предоставляет возможность поиска файлов данных без явного его программирования.

Примеры:

- PATH C:\UTILS;C:\DOS33 — установить два заданных маршрута поиска;
- SET PATH=C:\UTILS;C:\DOS33 — то же, если не использовать в каталогах строчные буквы;
- PATH D:\;C:\UTILS;C:. — установить три заданных маршрута поиска, причем последним является текущий каталог диска в приводе C (который может меняться с целью настройки этих маршрутов).

Команда APPEND

Назначение: установка и отображение маршрутов поиска файлов с *данными* для выполняемых программ.

Тип: внешняя.

Синтаксис.

Для задания режимов (при первом использовании) *можно:*

APPEND [/X] [/E]

Для установки или отображения маршрутов поиска (при первом или последующем использовании) *следует:*

APPEND [*dir*[;*dir*]...]

Для отмены (удаления) маршрутов поиска требуется:

APPEND ;

Комментарии. Данная команда является аналогом команды PATH, но в отличие от нее устанавливает не маршруты поиска исполняемых файлов для DOS, а маршруты поиска любых файлов для выполняемых программ (главным образом файлов с данными).

DOS в принципе предоставляет ряд возможностей для явного поиска программами требуемых файлов, среди которых:

— доступ в программе к маршруту запуска файла, ее содержащего (вследствие этого программа может найти требуемый файл, даже если она запускалась не из текущего каталога);
— доступ к окружению DOS для использования значений глобальных переменных, в том числе переменной PATH.

Однако если в программе поиск файла явно не запрограммирован, а просто указывается полная или неполная его спецификация и файл в выделенном по этой спецификации каталоге отсутствует, то помочь найти данный файл в другом каталоге сможет только предварительная выдача команды APPEND.

Если команда выполнена сразу в форме для установки маршрутов поиска, то DOS будет использовать их для доступа к файлу только в том случае, когда в выполняемой программе файл открывается (с использованием функций 0FH или 3DH по прерыванию 21H) или производится запрос его размера (по функции 23H прерывания 21H). Поиск файла будет осуществляться сначала в выделенном по его спецификации каталоге (в частности, в рабочем каталоге), а затем по всем

установленным спецификациями *dir* маршрутам в порядке их указания в командной строке. Этот процесс прекращается при обнаружении первого файла с заданным составным именем.

Предварительная установка переключателями режимов работы команды APPEND дополняет ее возможности. Два допустимых переключателя определяют следующее:

/X — задать расширенный режим поиска файлов. В этом случае маршруты поиска дополнительно будут использоваться тогда, когда программа выдает запросы на выполнение другой программы (функция 4BH прерывания 21H) и на поиск первого файла, сопоставимого с шаблоном (функции 11H и 4EH прерывания 21H);

/E — сохранять маршруты поиска файлов в окружении DOS, чтобы они могли отображаться командой SET.

Повторное выполнение команды APPEND с аргументами *dir* отменяет действие предыдущей команды (заменяет маршруты поиска на новые).

Если Вы ввели командную строку, содержащую только имя команды APPEND, то на экран дисплея будут выданы установленные маршруты поиска.

Для аннулирования всех маршрутов поиска следует ввести APPEND ;.

Замечания:

— выполнение команды APPEND увеличивает размер резидентной части DOS примерно на 5 Кбайт;

— при использовании команды ASSIGN команду APPEND следует выдать до нее;

— после выполнения команды APPEND в третьей форме можно ввести ее вновь, начиная при необходимости с первой формы (перезагрузка DOS не требуется);

— команда APPEND имеет ряд нежелательных побочных эффектов, например: если программа читает файл из каталога, заданного в APPEND, и обновляет его содержимое, то новая версия файла запишется в выделенный по спецификации каталог, в результате чего оригинал останется необновленным; команда DIR может выдать информацию о файлах не только в выделенном каталоге, но и в каталогах, заданных маршрутами поиска; если использован переключатель /X, то команды BACKUP и RESTORE будут также работать не в соответствии с ожиданиями. Сказанное относится и к ряду других команд DOS. В связи с нежелательными побочными эффектами команду APPEND целесообразно использовать только в случае крайней необходимости. Такая необходимость может быть вызвана потребностью размещения объемного программного продукта на нескольких дискетах при отсутствии НЖМД. Но не забывайте при этом своевременно отключать действие команды APPEND ;

— ни один из переключателей /X и /E по умолчанию не устанавливается;

— команда APPEND в состоянии контролировать не все возможности доступа к файлам из программ, так что иногда оказывается бесполезной;

— команду APPEND можно выдавать в среде Norton Commander'a, а ее действие сохраняется и после выхода из оболочки;

— команду APPEND можно использовать в сети для указания местоположения удаленных файлов с данными.

Примеры:

- APPEND C:\WORDSTAR
- APPEND /X /E
- APPEND C:\MSWORD;C:\DBASE

DOS 4.0. Для обеспечения возможности сокращения числа побочных эффектов в командной строке вместо *dir* допустима конструкция

dir [/X[:ON|:OFF]] [/PATH:{ON|OFF}]

Указанные в данном контексте переключатели влияют на то, в каких случаях файл ищется в каталоге *dir* в процессе выполнения команд ATTRIB, BACKUP, DIR, REPLACE, RESTORE и XCOPY, что позволяет управлять «видимостью» каждого из каталогов для различных средств DOS в отдельности. Эти переключатели определяют следующее:

/X[:ON] — задать расширенный режим поиска файлов, даже если он при первом выполнении команды APPEND не был установлен (переключатели /X и /X:ON эквивалентны);

/X:OFF — отключить расширенный режим поиска файлов, даже если он был установлен первым выполнением команды APPEND;

/PATH:ON — разрешить поиск по заданному маршруту файлов, в спецификациях которых наряду с составным именем может иметься также имя привода и/или маршрут;

/PATH:OFF — разрешить поиск по специфицированному маршруту только файлов, которые заданы лишь своими составными именами (без указания привода и маршрута).

Команда BREAK

Назначение: установка и отображение режима контроля нажатия комбинации клавиш Ctrl-Break.

Тип: внутренняя.

Синтаксис:

BREAK [ON|OFF]

Комментарии. Обычно DOS проверяет нажатие комбинации клавиш Ctrl-Break (Ctrl-C) только при обмене информацией с посимвольными устройствами (консолью и принтером) с тем, чтобы завершить выполнение программы по требованию пользователя. Ввод команды BREAK позволяет изменить режим контроля системой нажатия этой комбинации клавиш.

В командной строке допустим один из двух следующих аргументов:

ON — распространить контроль нажатия комбинации клавиш Ctrl-Break и на другие функции DOS, такие, как дисковый ввод-вывод;

OFF— переключить режим контроля в исходный (отключить режим, установленный аргументом ON).

Если ни один из аргументов не задан, то на экран дисплея выводится информация об установленном режиме (статус) контроля, т.е. ON или OFF.

По умолчанию (если команда BREAK не выдавалась) действует режим OFF.

Замечания:

— некоторые программы самостоятельно переключают режим контроля;

— альтернативным средством задания режима контроля нажатия комбинации клавиш Ctrl-Break является команда конфигурирования системы BREAK=;

— нажатие Ctrl-Break контролируется не выполняемой программой, а DOS, когда последняя получает управление по прерыванию определенного типа (по Ctrl-Break взводится соответствующий системный флаг, после чего текущая программа продолжает выполняться, а установка флага впоследствии анализируется DOS);

— при нажатии Ctrl-C действие Ctrl-Break эмулируется программно, вследствие чего Ctrl-C не всегда приводит к желаемому результату. Дело в том, что при нажатии Ctrl-C соответствующего прерывания для отметки этого факта в системе не возникает — расширенный код комбинации клавиш Ctrl-C просто размещается в буфере клавиатуры. Код Ctrl-C не будет считан DOS до тех пор, пока не будут прочитаны все предыдущие символы. Если же ввод с клавиатуры не осуществляется вовсе, то нажатие комбинации клавиш Ctrl-C останется незамеченным.

Пример: BREAK ON

Команда VERIFY

Назначение: установка и отображение режима контроля правильности записи информации на диски.

Тип: внутренняя.

Синтаксис:

VERIFY [ON|OFF]

Комментарии. В исходном состоянии DOS не проверяет возможности считывания информации после ее записи на диск. Команда VERIFY позволяет установить такой контроль.

В командной строке допустим один из двух аргументов:

ON — установить режим контроля правильности записи информации на диск путем ее последующего считывания;

OFF— отключить режим контроля.

Без аргумента команда VERIFY выводит на экран дисплея текущий статус (ON или OFF).

По умолчанию (когда VERIFY не выдавалась) действует OFF.

Замечания:

— команда VERIFY ON функционально аналогична переключателям /V в командах COPY и XCOPY, однако она действует на все последующие команды DOS и программы до выполнения VERIFY OFF;

— обычно запись на диск выполняется корректно, а задание VERIFY ON приводит к увеличению времени выполнения операций записи. Поэтому контроль записи целесообразно включать только при работе с особо ценной информацией, а также при записи на сбойные дискеты.

Пример: VERIFY ON

Команда DATE

Назначение: установка для DOS и отображение даты.

Тип: внутренняя.

Синтаксис:

DATE [*date*]

Комментарии. Установка даты для DOS влияет на корректность заполнения соответствующего поля элемента каталога при создании и обновлении файла, а также при создании подкаталога.

Если требуется установить дату и Вы знаете, в каком формате ее ввести, то при вызове команды DATE укажите дату аргументом *date*. Если Вам нужно вывести на экран дисплея текущую дату и затем, возможно, изменить ее (пользуясь предложенным форматом), то задайте команду DATE без аргумента. При этом появится, например, сообщение

```
Current date is Fri 11-2-91
Enter new date:
(Текущая дата — пятница, 2.11.91 г.
Введите новую дату:)
```

Вы можете просто нажать клавишу Enter (чтобы оставить дату без изменения) или ввести новую дату по аналогии с отображенной (номер месяца, номер дня и две последних цифры года с использованием в качестве разделителя символа «—»). Год допускается задавать полностью (четырьмя цифрами).

Замечания:

— формат отображения и задания даты устанавливается командой конфигурирования системы COUNTRY=;

— на ПЭВМ класса XT дата первоначально (при загрузке DOS) устанавливается по содержимому ПЗУ (например, 01-01-80). Поэтому команду DATE крайне желательно поместить в файл AUTOEXEC.BAT и явно задавать дату при загрузке, чтобы она была корректной. Если же этот файл отсутствует, то система при загрузке выдаст запрос на ввод даты автоматически;

— на ПЭВМ класса AT и старше, а также семейства PS/2 устанавливать дату после каждой загрузки не обязательно, так как они имеют встроенные часы с автономным питанием. Однако периодически возникает необходимость коррекции даты. При этом команда DATE может использоваться только с целью установки даты для текущего сеанса работы ПЭВМ, а при выключении питания ее действие отменяется. Чтобы переустановить непосредственно системные часы, требуется выполнить утилиту SETUP или другую подходящую утилиту (например, Norton Control Center из пакета Norton Utilities);

— система учитывает количество дней в каждом месяце, вследствие чего при наступлении нового дня дата сменится автоматически и корректно.

Пример: DATE 11-18-91

Команда TIME

Назначение: установка для DOS и отображение времени.
Тип: внутренняя.
Синтаксис:

TIME [*time*]

Комментарии. Установка времени для DOS влияет на корректность заполнения соответствующего поля элемента каталога при создании и обновлении файла, а также при создании подкаталога.

Если требуется установить время и Вы знаете, в каком формате его ввести, то при вызове команды TIME укажите время аргументом *time*.

Например, для США задаваемое время имеет следующий синтаксис:

чч:*мм*[:*сс*[.*лл*]]

где *чч* — часы (0..23);
мм — минуты (0..59);
сс — секунды (0..59);
лл — миллисекунды (0..99).

Минимально необходимо указать только часы и минуты.

Время зачастую отображается (но не командой TIME) в двенадцатичасовом формате (до полудня — окончание a, а после полудня — p).

Если Вам нужно вывести на экран дисплея текущее время и затем, возможно, изменить его (пользуясь предложенным форматом), то задайте команду TIME без аргумента. При этом появится сообщение вида

```
Current time is 8:45:17.95
Enter new time:
(Текущее время — 8.45.17,95.
Введите новое время:)
```

Вы можете просто нажать клавишу Enter (чтобы оставить время без изменения) или ввести новое время по аналогии с отображенным.

Замечания:

— формат отображения и задания времени устанавливается командой конфигурирования системы COUNTRY=;

— на ПЭВМ класса XT время первоначально (при загрузке DOS) устанавливается в нуль. Поэтому команду TIME желательно поместить в файл AUTOEXEC.BAT и явно задавать время при загрузке, чтобы оно было корректным. Если же этот файл отсутствует, то запрос на ввод времени выдается системой при загрузке DOS автоматически;

— на ПЭВМ класса AT и старше, а также семейства PS/2 устанавливать время после каждой загрузки DOS не обязательно, так как они имеют встроенные часы с автономным питанием. Целесообразно периодически контролировать время и при необходимости корректировать его командой TIME. Как и при использовании команды DATE, установка времени в этом случае будет действовать до выключения питания ПЭВМ. Чтобы изменить показания системных часов, нужно воспользоваться утилитой SETUP или другой подходящей утилитой.

Пример: TIME 14:09

Команда PROMPT

Назначение: изменение приглашения DOS.
Тип: внутренняя.
Синтаксис:

PROMPT [*string*]

Комментарии. Строка *string* задает вид приглашения DOS. Она может содержать любые символы, за исключением <, >, | и =, а также специальные комбинации двух символов. Произвольные символы выводятся в приглашении без каких-либо изменений. Специальные *двухсимвольные комбинации* начинаются с символа $ и задают определенную последовательность символов, которая и выводится в приглашении DOS. Иначе говоря, комбинации с $ являются в общем случае переменными (или параметрами), вместо которых подставляется их текущие значения.

DOS различает следующие специальные двухсимвольные комбинации и интерпретирует их таким образом:

$P — полная спецификация рабочего каталога;
$T — текущее время;
$D — текущая дата;
$V — номер версии DOS;
$N — имя текущего дисковода;
$H — символ BS;
$E — символ ESC (код 1BH, или 27);
$_ — маркер EOL (два символа — CR и LF);
$G — символ >;
$L — символ <;
$B — символ |;
$Q — символ =;
$$ — символ $;
$*x* — пусто, если *x* — одиночный символ и $*x* не совпадает ни с одной из перечисленных выше комбинаций.

При выводе приглашения DOS на экран управляющие символы, представляемые комбинациями $H и $_, интерпретируются стандартным образом ($H приводит к удалению предыдущего символа, а $_ — к продолжению выдачи приглашения с начала следующей строки). Однако управляющий символ ESC ($E) в данном контексте играет особую роль: он является признаком управляющей последовательности для драйвера ANSI.SYS (см. п. 5.10.3).

Нормальным (стандартным) приглашением DOS, устанавливаемым при загрузке системы, является NG (имя текущего привода и символ >). Выполнение команды PROMPT без аргумента *string* приводит к возврату именно к этому виду приглашения.

Замечания:

— команду PROMPT целесообразно помещать в файл AUTOEXEC.BAT;

— наиболее удачным и поэтому часто используемым приглашением DOS является PG;

— управляющие последовательности с $E для драйвера ANSI.SYS существенно расширяют возможности формирования приглашений и дополнительно позволяют использовать приглашения для переключения режимов работы дисплея;

— на первый взгляд бесполезная, комбинация $H позволяет уменьшить длину приглашения (например, удалить из текущего времени секунды и миллисекунды);

— ввод команды PROMPT приводит к немедленному отображению нового приглашения, т.е. первый раз оно посылается на экран командой, которая его устанавливает;

— при запуске вторичной копии КП из программы целесообразно установить уникальное приглашение, содержащее, в частности, информацию о способе возврата в программу;

— заданная в качестве аргумента строка помещается в окружение DOS в роли значения глобальной переменной PROMPT и выбирается для использования именно из него;
— команда PROMPT в среде оболочки Norton Commander не действует.

Примеры:

- PROMPT PG — отображать в приглашении полную спецификацию рабочего каталога, за которой следует символ > (скажем, C:\UTILS\NU>);
- PROMPT Пожалуйста, введите команду DOS: — отображать в приглашении DOS указанную строку;
- PROMPT THHHHH$G — отображать в приглашении DOS текущее время (часы и минуты), а также символ > (скажем, 12:31>);
- PROMPT Введите EXIT для возврата в PROG$_$N$G — отображать приглашение, состоящее из двух строк, в первой из которых выводится текст «Введите EXIT для возврата в PROG», а во второй — имя текущего привода и символ >.

Команда FASTOPEN

Назначение: ускорение открытия файлов и каталогов.
Тип: внешняя, несетевая.
Синтаксис:

FASTOPEN ◁*d*:[=*n*]▷...

Комментарии. При выполнении данной команды для каждого логического жесткого дисковода *d* в ОЗУ создается кэш из *n* элементов и загружается код, управляющий его работой. Если *n* не задано, то по умолчанию принимается 34. Диапазон допустимых значений этого аргумента составляет 10 — 999. В командной строке может быть указано до четырех конструкций *d*:[=*n*].

Каждый элемент кэша хранит информацию об имени файла (каталога) и о его размещении на жестком диске. Поэтому доступ к такому файлу (каталогу) может быть ускорен за счет извлечения сведений о его местоположении из кэша в ОЗУ, а не с жесткого диска, обладающего существенно меньшим быстродействием.

Элементы кэша заполняются не во время интерпретации команды FASTOPEN, а при первом открытии файлов и каталогов, если кэш еще имеет свободные элементы. Повторное открытие файлов (каталогов), зарегистрированных в кэше, осуществляется с использованием его содержимого. Занесенная в кэш информация никогда из него не удаляется, даже если обращение к зарегистрированным файлам (каталогам) больше ни разу не производится. Когда все элементы кэша будут заняты, пополнение его содержимого прекратится.

Замечания:
— команда FASTOPEN к НГМД неприменима;
— при каждом сеансе работы с DOS команда FASTOPEN может быть выполнена только один раз;
— команда FASTOPEN особенно полезна в случае разветвленной файловой структуры на жестком диске, так как в этом случае доступ к файлу требует нескольких обращений к вложенным друг в друга каталогам. Эффект команды увеличивается при фрагментации каталогов;
— чрезмерное увеличение размера кэша может привести к противоположному эффекту (к увеличению времени доступа к файлу за счет последовательного поиска в большом кэше);
— команда FASTOPEN увеличивает размер резидентной части DOS: каждый элемент кэша занимает 35 байт, да еще требуется дополнительная память для резидентной программы управления кэшем. Так, например, для *n*=100 нужно около 5,5 Кбайт;
— команда FASTOPEN не может уменьшить время обработки фрагментированных файлов, так как карты размещения файлов в кэше не фиксируются, да и в любом случае требуется обращение к таблице размещения файлов на диске.

Пример: FASTOPEN C:=200 D:

DOS 4.0. Синтаксис:

FASTOPEN ◁*d*:[={*n*|([*n*],*m*)}] [/X]▷...

Здесь *n* имеет тот же смысл, но по умолчанию принимается 10. Аргумент *m* задает число (в диапазоне 1 — 999) описателей *экстентов* (непрерывных областей файлов и каталогов) для каждого файла (каталога). Элемент кэша для файла (каталога) занимает около 48 байт, а каждый описатель экстента дополнительно требует 16 байт. Если число *m* не задано (указана конструкция =*n*, а не ([*n*],*m*)), то описатели экстентов не создаются.

Когда специфицирована конструкция (,*m*), кэш имен не создается, а формируется только кэш описателей экстентов.

Команда FASTOPEN усовершенствована в двух аспектах:

1) для файлов (каталогов) в кэше дополнительно сохраняются карты их физического размещения, для чего используются доступные описатели экстентов. Если файл (каталог) размещен в большем, чем *m*, количестве экстентов, то фиксируется информация только о первых *m* экстентах;

2) кэш можно разместить в отображаемой (LIM EMS 4.0) памяти, если указать переключатель /X.

Средства описания и использования экстентов файлов (каталогов) дополнительно увеличивают эффект применения команды FASTOPEN в случае фрагментированных файлов (каталогов). Для достижения наилучших результатов с переключателем /X используйте в команде SELECT значения по умолчанию.

Команда ASSIGN

Назначение: подмена одного дисковода другим.
Тип: внешняя.
Синтаксис:

ASSIGN [*d1*=*d2*]...

Комментарии. Данная команда используется, когда программный продукт жестко привязан к приводам с определенными именами, а пользователю требуется перенаправить ввод-вывод на другие дисководы. Команда ASSIGN должна быть введена до запуска таких программ. Ее выполнение сводится к тому, что привод *d1* подменяется приводом *d2* (все операции ввода-вывода, связанные с *d1*, перенаправляются на *d2*).

В команде ASSIGN можно указать несколько конструкций *d1*=*d2*. Если же не указано ни одной, то все предыдущие переназначения отменяются. Новое назначение отменяет старое.

Замечания:

— в качестве *d2* не допускается использовать неопределенное в системе имя;

— для обеспечения совместимости с будущими версиями DOS используйте вместо ASSIGN команду SUBST. Например, команды ASSIGN A=C и SUBST A: C:\ эквивалентны;

— в связи с тем что команда ASSIGN маскирует действительный тип устройства *d1*, ее нельзя использовать совместно с командами, требующими информации о приводе (BACKUP, RESTORE, LABEL, JOIN, SUBST и PRINT);

— команды FORMAT и DISKCOPY игнорируют переназначения, выполненные командой ASSIGN;

— коллизия имен приводов в командах ASSIGN, SUBST и JOIN недопустима;

— используйте ASSIGN только в случае крайней необходимости.

Пример:

- ASSIGN A=C B=C — подменить приводы A и B приводом C (перенаправить ввод-вывод с дисководов A и B на накопитель C).

Команда SUBST

Назначение: обозначение маршрута именем дисковода и отображение введенных обозначений.
Тип: внешняя, несетевая.
Синтаксис:

SUBST [*d*: *dir*]

или

SUBST *d*: /D

Комментарии. Команда SUBST в первой форме и с заданными аргументами позволяет обозначить спецификацию каталога *dir* именем привода *d*. При этом *d* оказывается фиктивным приводом, так как он реально не существует. Фиктивный привод впоследствии может использоваться в командах DOS аналогично логическому приводу. Это обеспечивает упрощение задания спецификаций файлов и каталогов. Вторым аргументом команды (*dir*) обязательно должна быть указана полная спецификация каталога.

Форма команды с переключателем /D служит для удаления заданного фиктивного привода.

Если команда SUBST специфицирована без аргументов, то на экран дисплея выводится информация обо всех имеющихся на данный момент фиктивных приводах.

Замечания:

— список допустимых имен фиктивных приводов определяется командой конфигурирования системы LASTDRIVE=. Если эта команда в CONFIG.SYS не указана, то допустимыми именами будут A, B, C, D и E. Команда LASTDRIVE= никаким образом не влияет на назначение имен логическим дисководам при загрузке DOS;

— если использованное в команде SUBST имя уже закреплено за логическим накопителем, то это назначение (с возможностью восстановления) отменяется и начинает действовать обозначение, установленное командой SUBST. В этом случае данная команда функционально заменяет команду ASSIGN, если спецификация *dir* определяет корневой каталог. Следовательно, SUBST можно считать обобщением команды ASSIGN;

— если использованное в команде SUBST имя не закреплено за логическим дисководом, то оно просто вводится для обозначения спецификации каталога без каких-либо побочных эффектов;

— в спецификации *dir* не допускается указывать явно или неявно текущий каталог;

— фиктивный привод *d* можно использовать в командах DOS не только как префикс к маршруту, но и в качестве самостоятельного привода, задав, к примеру, команду *d*: с той целью, чтобы соответствующий каталог, так же как и диск, стал текущим;
— для переназначения фиктивного привода необходимо предварительно его удалить;
— фиктивные приводы полезно использовать для обозначения поддеревьев файловой структуры жесткого диска при их использовании различными пользователями с целью разграничения доступа;
— после обозначения каталога *dir* однобуквенным именем явное указание *dir* в командах DOS остается возможным;
— после обозначения каталога именем привода этот каталог не может быть удален;
— с фиктивными приводами не могут работать команды BACKUP, CHKDSK, DISKCOMP, DISKCOPY, FDISK, FORMAT, LABEL, RECOVER, RESTORE и SYS;
— коллизия имен приводов в командах SUBST, ASSIGN и JOIN недопустима.

Пример. После выполнения команды

SUBST Z: C:\WORDPROC\LTRS

следующие две спецификации становятся эквивалентными:

Z:MYFILE.TXT
C:\WORDPROC\LTRS\MYFILE.TXT

Команда JOIN

Назначение: логическое подсоединение *(монтирование)* дисковода к каталогу диска на другом накопителе и отображение текущих подсоединений.
Тип: внешняя, несетевая.
Синтаксис:

JOIN [*d*: *dir*]

или

JOIN *d*: /D

Комментарии. Данная команда (в первой форме, с заданными аргументами) обеспечивает объединение файловых структур на жестких, гибких и/или виртуальных дисках в единую файловую структуру. Это особенно полезно, если ПЭВМ оборудована несколькими физическими НЖМД. Команда JOIN выполняет действия, обратные команде SUBST. Аргумент *d* задает имя логического привода, файловую структуру диска в котором требуется подсоединить к каталогу *dir*. После выполнения команды JOIN каталог *dir* будет обозначать *d*:\, а само имя *d* станет недоступным.
На аргументы накладываются следующие ограничения:
— привод *d* не должен быть текущим;
— спецификация *dir* должна указывать на подкаталог корневого каталога, причем этот подкаталог либо может существовать, но в этом случае обязан быть пустым, либо может отсутствовать (тогда он создается автоматически).
Команда JOIN без аргументов выдает на экран дисплея список всех текущих (активных) подсоединений.
Для отмены подсоединения логического привода *d* к какому-либо каталогу следует выполнить команду JOIN во второй форме (с переключателем /D), но этот каталог не должен быть рабочим.

Замечания:
— для переподсоединения привода к другому каталогу или другого привода к данному каталогу предварительно следует отменить соответствующее активное подсоединение;
— созданный каталог *dir* после отмены подсоединения не удаляется;
— все манипуляции с каталогом *dir*, если действует подсоединение, производятся по сути с файловой структурой на соответствующем логическом диске;
— с приводами, подсоединенными к каталогам, нельзя выполнять операции командами BACKUP, CHKDSK, DISKCOMP, DISKCOPY, FDISK, FORMAT, LABEL, RECOVER, RESTORE и SYS;
— не рекомендуется подсоединять привод к каталогу, лежащему на пути к каталогу, использованному в команде SUBST;
— не рекомендуется также допускать коллизии имен приводов в командах JOIN и ASSIGN.

Пример:

- JOIN E: C:\HARDDRV2 — подсоединить файловую структуру диска из привода E к каталогу \HARDDRV2 диска в приводе C.

Команда CTTY

Назначение: изменение стандартного УВВ DOS.
Тип: внутренняя.
Синтаксис:

CTTY *устройство*

Комментарии. Стандартным УВВ для DOS является CON (клавиатура и дисплей). Чтобы сменить стандартное УВВ, нужно выполнить команду CTTY, в качестве аргумента которой указать имя нового такого устройства. Это устройство обязательно должно обеспечивать как ввод, так и вывод информации. С нового стандартного УВВ можно вводить команды DOS. Сообщения DOS также будут выводиться на это устройство. Ввод-вывод программ, связанный со стандартным УВВ, будет соответствующим образом перенаправлен. Если же программа адресует конкретное устройство CON, то перенаправления ввода-вывода не произойдет.

Замечание: команда CTTY применяется редко, однако может быть полезна в командных файлах для отмены выдачи сообщений на экран дисплея при интерпретации команд.

Примеры:

- CTTY AUX — стандартным УВВ считать устройство, подсоединенное к первому адаптеру последовательного интерфейса;
- CTTY NUL — переназначить стандартный ввод-вывод на фиктивное устройство (запретить ввод команд и выдачу сообщений);
- CTTY CON — вернуть стандартный ввод-вывод на устройство CON.

Команда SELECT

Назначение: установка DOS на новый диск (жесткий или гибкий) и ее конфигурирование.
Тип: внешняя.
Синтаксис:

SELECT [*d*:] [*dir*] *страна клавиатура*

Комментарии. Аргументы определяют следующее:

d — привод, в который установлен фирменный системный диск, используемый в качестве источника системных файлов (по умолчанию — A);

dir — спецификация целевого каталога, задающая дисковод, на диск в котором следует перенести систему, и каталог, в который нужно поместить утилиты DOS (по умолчанию — B:);

страна — код страны, в соответствии с которым требуется сконфигурировать DOS (см. описание команды конфигурирования системы COUNTRY=);

клавиатура — двухбуквенный код клавиатуры, в соответствии с которым требуется сконфигурировать DOS (см. описание команды KEYB).

По команде SELECT выполняются следующие действия:
1) проверяется наличие драйвера COUNTRY.SYS на дискете в приводе *d*;
2) командой FORMAT инициализируется диск, указанный в *dir*, с переносом системы;
3) выполняется команда XCOPY *d: dir* для копирования необязательных системных файлов;
4) на целевом диске создается файл CONFIG.SYS с командой COUNTRY=*страна*;
5) на целевом диске создается файл AUTOEXEC.BAT с командой KEYB *клавиатура, ср.*

Замечание: команда SELECT упрощает подготовку нового системного диска, хотя те же действия могут быть выполнены другими средствами DOS, но это требует высокой квалификации пользователя.

Пример:

- SELECT A: C:\DOS33 033 FR — создать системный диск, сконфигурированный для использования во Франции.

DOS 4.0. Команда SELECT заметно усовершенствована, теперь является полноэкранной, диалоговой и позволяет создать файлы CONFIG.SYS, а также AUTOEXEC.BAT, содержащие всю требуемую информацию по конфигурированию системы.

Во время установки DOS на новый диск команда SELECT выполняет следующие функции (по указаниям пользователя) с целью конфигурирования DOS:

— формирует резидентную часть DOS требуемого размера в зависимости от емкости ОЗУ ПЭВМ;

— настраивает DOS на страну использования, устанавливая код страны и клавиатуры;

— конфигурирует DOS в соответствии с типом принтера и адаптером интерфейса, к которому подключен принтер (при необходимости пользователь может назначить адаптер последовательного интерфейса вместо параллельного);

— производит дополнительную настройку системы, обеспечивая переключение кодовых страниц, поддержку отображаемой памяти, расширенную поддержку дисплея (драйвер ANSI.SYS), повышение производительности файловой системы (командой FASTOPEN), поддержку дисплея посредством команды GRAFTABL, принтера посредством команды GRAPHICS, DOS командой SHARE, оболочки DOS Shell и виртуального диска в ОЗУ (драйвер RAMDRIVE.SYS или VDISK.SYS).

Результатом выполнения перечисленных функций являются полностью сформированные файлы AUTOEXEC.BAT и CONFIG.SYS. Все описанные действия могут быть реализованы другими средствами DOS, но это займет гораздо больше времени.

Команда SELECT предоставляет в распоряжение пользователя шесть основных экранов, в которых он может выбирать и устанавливать спецификации в зависимости от конфигурации оборудования и своих потребностей. На каждом экране пользователю предлагаются вопросы и варианты ответа на них. Предлагаемый ответ подсвечивается. Если он Вас не устраивает, то клавишами управления курсором можно выбрать требуемый и нажать клавишу Enter. С каждым экраном связан подробный интерактивный справочник, который вызывается клавишей F1.

Перед выполнением команды SELECT желательно иметь ответы на следующие вопросы, что поможет Вам без трудностей установить систему:

— каков объем ОЗУ ПЭВМ;
— принтером какого типа снабжена ПЭВМ;
— имеется ли в ПЭВМ отображаемая память.

Для полной установки DOS потребуется от одной до четырех пустых дискет в зависимости от их емкости и наличия НЖМД. При желании дополнительно установить DOS Shell подготовьте еще одну дискету.

Когда пользователь хочет создать полностью новый системный диск, следует загрузить DOS с фирменного системного диска Install. Если загрузка осуществлялась с 133-мм 360-Кбайт дискеты, то далее нужно:

— извлечь из дисковода дискету Install и установить фирменный системный диск Select;
— нажать клавишу Enter;
— извлечь из привода дискету Select и снова установить Install;
— нажать клавишу Enter.

После выполнения этих действий на дисплей будет выведен экран Welcome («Добро пожаловать»), что свидетельствует о начале выполнения команды SELECT.

В случае, когда загрузка DOS осуществлена с 89-мм 720-Кбайт дискеты, приведенную последовательность действий выполнять не требуется. Нужно только дождаться появления экрана Welcome.

Если пользователь хочет только лишь обновить на целевом (и являющемся системном) диске спецификации, установленные ранее, достаточно поместить в дисковод 133-мм диск Install или 89-мм диск Select и запустить с него утилиту SELECT, введя командную строку SELECT MENU. После этого появится экран Welcome.

Нажмите клавишу Enter, и будет выведен экран Introduction («Введение»), в котором описывается назначение клавиш:

ENTER	— Proceeds to next step of program (осуществляет переход к следующему шагу программы)
ESC	— Cancels current screen (аннулирует текущий экран)
TAB	— Moves cursor to next text entry field (перемещает курсор к следующему полю текстового описания)
PAGE UP/PAGE DOWN	— Scrolls information one page at a time (прокручивает одну страницу за раз)
F1	— Displays help information (отображает справочную информацию)
F3	— Exits Select program (выходит из программы SELECT)
LEFT/RIGHT	— Scrolls data horizontally (прокручивает данные по горизонтали)

После нажатия клавиши Enter появляется *первый экран*, позволяющий задать размер резидентной части DOS за счет изменения количества и емкости различных системных буферов и областей данных соответствующими указаниями в файле CONFIG.SYS. Возможны следующие варианты:

1 — для ПЭВМ с 256-Кбайт ОЗУ (резидентная часть DOS минимизируется);
2 — для ПЭВМ с 512-Кбайт ОЗУ (резидентная часть DOS будет иметь средний размер);
3 — для ПЭВМ с ОЗУ бо́льшим, чем 512 Кбайт (резидентная часть DOS будет максимальна по размеру).

Чем больше размер резидентной части DOS, тем быстрее она функционирует.

Второй экран позволяет пользователю настроить устанавливаемую DOS на использование в определенной стране путем задания кода страны и двухбуквенного кода клавиатуры. Меню экрана содержит все допустимые коды стран и клавиатур. Если дополнительно требуется задать возможность смены языка (переключения кодовых страниц), то следует установить Y («Да») для опции Code-Page Switching.

Третий экран обеспечивает выбор дисковода, на диск (диски) в котором требуется установить DOS. Если Вы выбрали жесткий диск, то появляется экран DOS Location («Размещение DOS»). Здесь требуется специфицировать каталог, в который будут скопированы многие системные файлы.

Четвертый экран позволяет настроить DOS в соответствии с имеющимся в ПЭВМ принтером и тем, к какому адаптеру интерфейса он подключен. Вам будут предложены все имеющиеся возможности, из которых можно выбрать до семи типов печатающих устройств.

Пятый экран обеспечивает установку оболочки DOS Shell.

Шестой экран позволяет отредактировать имеющиеся или создать новые файлы CONFIG.SYS и AUTOEXEC.BAT путем выбора параметров из предложенных.

Если DOS устанавливается на жесткий диск и он не разбит на разделы с логическими дисками, Вам будет предложено это сделать (см. описание команды FDISK), после чего DOS будет перенесена на жесткий диск.

В случае, когда целевой жесткий диск уже имеет файлы CONFIG.SYS и AUTOEXEC.BAT, то по команде SELECT будут созданы новые файлы CONFIG.400 и AUTOEXEC.400. Пользователю после завершения установки системы следует перенести содержащуюся в них информацию в старые файлы и осуществить рестарт DOS.

При установке системы на НГМД потребуется одна 1,44-Мбайт дискета, две либо три 720-Кбайт дискеты или четыре либо пять 360-Кбайт дискет (в зависимости от того, устанавливается ли DOS Shell). 1,44-Мбайт дискета будет иметь метку STARTUP, 720-Кбайт дискеты — метки STARTUP, WORKING и SHELL, а 360-Кбайт дискеты — STARTUP, WORKING1, WORKING2, WORKING3 и SHELL. Загрузка DOS должна осуществляться с дискеты STARTUP.

5.6.6. Команды управления системой

Команда COMMAND

Назначение: запуск вторичной копии КП DOS.
Тип: внешняя.
Синтаксис:

COMMAND [*dir*] [*устройство*] [/E:*n*] [/P] [/C *process*] [/F] [/D]

Комментарии. По команде COMMAND принудительно запускается вторичная копия КП из файла COMMAND.COM (исполняется именно данный файл). Первичная же копия КП, инициированная во время загрузки DOS, остается при этом в ОЗУ, но теряет активность.

Запуск вторичной копии КП используется для:

— *создания вложенных командных файлов,* причем в версиях DOS до 3.3 команда COMMAND была единственным средством для этого (с переключателем /C);

— *выдачи внутренней команды DOS* из выполняемой программы в автоматическом режиме, т.е. без участия пользователя (с переключателем /C);

— *имитации временного возврата в DOS* из выполняемой программы, в результате чего можно работать с КП в интерактивном режиме, а в нужный момент возобновить активность программы (без переключателей /P и /C);

— *увеличения размера области памяти для окружения DOS* (с переключателем /E);

— *частичного рестарта DOS* для устранения сбоев (с переключателем /P);

— *перевода КП в режим, при котором в случае ошибки ввода-вывода в зависимости от версии DOS автоматически генерируется ответ F для Fail или I для Ignore* (с переключателем /F).

Накладные расходы памяти, связанные с запуском вторичной копии КП, невелики, так как для хранения его резидентного модуля требуется всего 4 Кбайт ОЗУ.

Вновь запущенному КП передается копия окружения, сформированного до этого. Следовательно, никакие изменения в копии окружения не окажут воздействия на его оригинал.

Аргумент *dir* указывает, где находится файл с КП (COMMAND.COM). Спецификация *dir* служит для обновления значения глобальной переменной COMSPEC в окружении DOS. Значение COMSPEC, как мы уже знаем, используется при подгрузке транзитного модуля КП. Каталог *dir* задавать не обязательно, если COMSPEC в оригинале окружения правильно специфицирует файл COMMAND.COM. Несоответствие, требующее указания *dir*, может возникнуть только в случае, когда размещение рабочего файла COMMAND.COM перед запуском вторичной копии КП изменено, а значение переменной COMSPEC осталось немодифицированным. Отметим, что *dir* также способствует отысканию файла COMMAND.COM собственно при запуске вторичной копии КП. Однако в этом нет необходимости, так как можно указать маршрут перед COMMAND.

Задание аргумента *устройство* приводит к тому, что оно становится стандартным УВВ DOS (см. описание команды CTTY). По умолчанию принимается CON.

Переключатели определяют следующие действия:

/E:*n* — специфицирует (в байтах) размер области памяти для окружения DOS. Допустимы значения *n* от 128 до 32768, кратные 16-ти. По умолчанию принимается 128;

/P — обеспечивает запуск КП в интерактивном резидентном режиме с автоматическим выполнением файла AUTOEXEC.BAT. Возврат в первичную копию КП становится невозможным;

/C *process* — обеспечивает запуск КП с целью выполнения команды DOS, заданной посредством *process*, и автоматический возврат в родительский процесс, в частности в первичную копию КП DOS;

/F — обеспечивает запуск КП в режиме, когда в случае ошибки ввода-вывода (по которой выдается сообщение «Abort, ...») в зависимости от версии DOS автоматически генерируется ответ F или I;

/D — обеспечивает отключение выдачи запросов о дате и о времени в случае отсутствия файла автозапуска AUTOEXEC.BAT.

Переключатели /P и /C функционально несовместимы. Однако синтаксически их одновременное указание допустимо (при этом /P игнорируется). Если ни один из переключателей /P и /C не задан, то КП запускается в интерактивном режиме, но AUTOEXEC.BAT не выполняется и возможен возврат в родительский процесс по команде EXIT. Задавать в переключателе /C целесообразно только внутреннюю команду DOS, так как внешняя команда может быть выполнена без явного участия КП как автономная программа. Переключатель /F можно использовать, например, перед копированием данных с дефектной дискеты, чтобы избавиться от многократной явной выдачи одного и того же ответа. Данный переключатель может использоваться совместно с переключателем /P или /C. Переключатель /D имеет смысл специфицировать только вместе с переключателем /P.

Замечания:

— если число в переключателе /E выходит за допустимые границы, то на экран дисплея выводится сообщение «Involid environment size specified» («Специфицирован ошибочный размер окружения»). Однако выполнение команды продолжается, а для окружения выделяется память минимально или максимально возможного размера в зависимости от заданного *n*;

— если число в переключателе /E не кратно 16, то оно будет автоматически скорректировано без каких-либо сообщений об этом;

— пробелы между /C и *process* в командной строке необязательны;

— если переключатель /C не задан, то независимо от указаний в командной строке всегда создается вторичная копия КП;

— переключатели /F и /D не документированы, а использовать /F нужно в высшей степени осторожно;

— текущий привод и текущие каталоги при запуске вторичной копии КП не изменяются;

— вид приглашения DOS, а также режимы VERIFY и BREAK вторичной копией КП наследуются;

— режим, заданный переключателем /F, вторичной копией КП не наследуется;

— глубина вложенности копий КП не ограничивается.

Примеры:

- COMMAND /E:512 /P — запустить копию КП в резидентном интерактивном режиме с 512-Кбайт окружением;
- COMMAND /C DIR A:*.BAS — выполнить команду DIR A:*.BAS (имеет смысл задавать только в программе).
- COMMAND /F — запустить копию КП в нерезидентном интерактивном режиме с автоматической генерацией ответа F или I при возникновении ошибок ввода-вывода.

DOS 4.0. Минимальным размером окружения (и значением по умолчанию) является 160 Кбайт.

Команда EXIT

Назначение: выход из КП и возврат на предыдущий уровень.
Тип: внутренняя.
Синтаксис:

EXIT

Комментарии. Эта команда обеспечивает возврат управления интерактивной вторичной копией КП родительскому процессу (КП или программе). Команда EXIT не действует на первичную копию КП, запускаемую при загрузке DOS, и на вторичную копию, если последняя запущена с переключателем /P (см. описание команды COMMAND).

Окружение, сформированное в ходе работы вторичной копии КП, удаляется, а окружение родительского КП (или другого процесса) восстанавливается. Не возвращаются также новый вид приглашения DOS и режим, заданный командой COMMAND /F. Однако текущий привод и текущие каталоги, а также режимы VERIFY и BREAK остаются теми, которые установлены вторичной копией.

5.6.7. Информационные команды

Команда VER

Назначение: отображение производителя и номера версии загруженной DOS.
Тип: внутренняя.
Синтаксис:

VER

Комментарии. Если Вы хотите узнать номер версии DOS, с которой предстоит работать, введите VER и на экране дисплея появится сообщение вида

MS-DOS Version 3.30

Команда MEM

Назначение: отображение информации о распределении ОЗУ (карты памяти).
Тип: внешняя.
Синтаксис:

MEM [/PROGRAM | /DEBUG]

Комментарии. Эта команда имеется только в DOS, начиная с версии 4.0. Переключатели определяют объем выводимой информации:

/PROGRAM — вывести информацию о программах, загруженных в ОЗУ (о резидентных программах);

/DEBUG — вывести информацию о резидентных программах, внутренних драйверах и другие сведения.

Переключатели /PROGRAM и /DEBUG несовместимы.

Командой MEM отображаются также сведения о наличии и использовании отображаемой памяти (если установлен драйвер LIM EMS 4.0), а также расширенной памяти. Если ни один из переключателей в командной строке не задан, то выводятся только эти сведения.

Карта стандартной памяти состоит из четырех следующих колонок:

1) адрес начала области памяти (Address);
2) имя программы, использующей область памяти (Name);
3) размер области памяти в байтах (Size);
4) тип содержимого области памяти (Type).

Все числа приводятся в шестнадцатеричной системе счисления.

Различают такие *типы* содержимого областей памяти, как:

— *векторы прерываний* (Interrupt Vector);
— *область связи BIOS*, находящейся в ПЗУ (ROM BIOS Communication Area);
— *область связи DOS* (DOS Communication Area);
— *системная программа* (System Program), т.е. компонент DOS;
— *области памяти, выделяемые по командам конфигурирования системы* DEVICE=, FILES=, FCBS=, BUFFERS=, LASTDRIVE= и STACK=;
— *программа* (Program);
— *данные* (Data);
— *окружение* (Environment);
— *свободная область* (Free).

Под картой памяти выводятся нижеперечисленные сведения (в десятичной системе счисления):

— общий объем стандартной памяти (total memory);
— объем доступной части ОЗУ (available), обычно совпадающий с общим объемом;
— максимальный размер исполняемой программы (lagest executable program size), т.е. размер свободной области в стандартной памяти;
— общий объем отображаемой памяти (total EMS memory);
— размер свободной области в отображаемой памяти (free EMS memory);
— общий объем расширенной памяти (total extended memory);
— размер свободной области в расширенной памяти (available extended memory).

Единицей измерения является байт.

5.7. Инструментальные команды

Команды этой группы по сути являются интерфейсом простых инструментальных систем, поставляемых вместе с DOS, хотя строго говоря не содержащихся в ней. К инструментальным относятся следующие команды:

BASIC — осуществляет вызов интерпретатора языка BASIC;

DEBUG — обеспечивает отладку машинных программ (поиск и исправление ошибок);

EDLIN — осуществляет вызов построчного текстового редактора;

EXE2BIN — преобразует EXE-файлы в абсолютные или позиционно-независимые программные файлы;

LINK — обеспечивает компоновку программ.

Мы не будем рассматривать их подробно, так как они не являются лучшими в своем классе и не вписываются в задачи, решаемые ОС. Вместе с тем определенный интерес представляет команда EXE2BIN.

5.7.1. Команда EDLIN

Назначение: подготовка текстовых файлов.
Тип: внешняя.
Синтаксис:

EDLIN *file* [/B]

Комментарии. EDLIN — это построчный текстовый редактор (работает с отдельными строками текстового файла), хотя современные текстовые редакторы практически все являются экранными. Он служит для создания и редактирования небольших текстовых файлов, например с письмами или простыми программами. Тем не менее EDLIN обеспечивает:
— чтение и редактирование файлов любой длины;
— работу с файлами, содержащими внутри себя маркеры EOF;
— включение в создаваемый файл управляющих символов (после комбинации клавиш Ctrl-V).
Спецификация создаваемого или редактируемого файла задается аргументом *file*. Если указан переключатель /B, то в файле игнорируются маркеры EOF, чем и обеспечивается работа со всем его содержимым вместо части файла до обнаружения первого такого маркера.
EDLIN является интерактивной программой.

Замечание: редактор EDLIN используется редко из-за ограниченности его возможностей и трудности работы с ним.

5.7.2. Команда BASIC

Назначение: выполнение (интерпретация) программ, составленных на языке Basic.
Тип: внешняя.
Синтаксис:

BASIC[A] [*file*] [/F:*n1*] [/S:*n2*] [/C:*n3*] [/M:*n4*[,*n5*]] [/D]

Комментарии. Продукт BASICA в отличие от BASIC имеет графические средства. Аргумент *file* задает файл, содержащий программу на языке BASIC, которую требуется выполнить. Если этот аргумент опущен, то BASIC входит в интерактивный режим работы. В этом случае операторы программы вводятся с клавиатуры и интерпретируются.
Допустимы следующие переключатели:

/F:*n1* — устанавливает максимальное количество открытых файлов (по умолчанию — 3);

/S:*n2* — задает размер буфера на диске (по умолчанию — 128 байт);

/C:*n3* — специфицирует размер внутреннего буфера (по умолчанию — 256 байт);

/M:*n4*[,*n5*] — определяет максимальный размер (*n4*) рабочей области (по умолчанию — 64 Кбайт) и максимальный размер (*n5*) области для машинного кода;

/D — обеспечивает двойную точность для всех трансцендентных функций.

Замечание: система PC DOS содержит интерпретатор GWBASIC.

5.7.3. Команда LINK

Назначение: компоновка объектных модулей.
Тип: внешняя.
Синтаксис:

LINK [*file1*] [,[*file2*][,[*file3*][,*file4*]]] [;]

Комментарии. Компоновщик LINK формирует исполняемую перемещаемую программу в формате EXE-файла из нескольких объектных модулей (OBJ-файлов) с использованием объектных библиотек (LIB-файлов). Объектные модули генерируются Макроассемблерами и компиляторами с языков высокого уровня. Результат компоновки может быть непосредственно выполнен.
Компоновщик реализует следующие действия:
— *объединяет коды и данные,* содержащиеся в нескольких объектных модулях;

— *разрешает внешние ссылки* между ними;
— *подключает дополнительно модули из объектных библиотек* с целью окончательного разрешения внешних ссылок;
— *формирует EXE-файл*, структура которого описана в п. 5.2.2;
— *формирует листинг* (отчет о своей работе).
В командной строке можно указать до четырех аргументов:

file1 — спецификация объектного файла или спецификации нескольких объектных файлов, разделенные символом +;

file2 — спецификация результирующего EXE-файла;

file3 — спецификация файла, в который будет помещен листинг;

file4 — спецификация объектной библиотеки или спецификации нескольких объектных библиотек, разделенные символом +.

Вслед за аргументами можно указать множество различных переключателей, которые здесь не рассматриваются.
Если некоторые или все аргументы не заданы, а также если командная строка не завершается точкой с запятой, то компоновщик предложит Вам специфицировать отсутствующую информацию путем выдачи запросов.
Замечание: EXE-файл может быть преобразован в COM-файл командой EXE2BIN.

5.7.4. Команда DEBUG

Назначение: отладка программ.
Тип: внешняя.
Синтаксис:

DEBUG [*file* [*arglist*]]

Комментарии. DEBUG обеспечивает интерактивную отладку и выяснение поведения программ в формате EXE- и COM-файлов. Обычно он используется для отладки исполняемых программ, полученных с языка уровня Ассемблера.
DEBUG позволяет:
— *прослеживать выполнение и управлять выполнением программ*;
— *отображать текстовые файлы* в ASCII- и шестнадцатеричном форматах;
— *вносить незначительные изменения* в отлаживаемую программу и создавать небольшие программы на языке Ассемблера;
— *исследовать и модифицировать содержимое секторов* на диске.
Аргумент *file* определяет файл, содержащий отлаживаемую программу; *arglist* является списком аргументов для отлаживаемой программы и задается в том же виде, как и при запуске программы *file*.
Если команда DEBUG введена без аргументов, то пользователю будет предложено задать имя файла.

Замечание: DEBUG является интерактивной программой со своим множеством команд.

5.7.5. Команда EXE2BIN

Назначение: преобразование EXE-файла в двоичный формат, т.е. в абсолютный или позиционно-независимый программный файл.
Тип: внешняя.
Синтаксис:

EXE2BIN *file1* [*file2*]

Комментарии. Аргумент *file1* задает *исходный* (подлежащий преобразованию), а *file2* — *результирующий* файл. Если в *file1* не задано расширение, то по умолчанию предполагается EXE. Для *file2* по умолчанию принимается расширение BIN.
Размер исходного файла не должен превышать 64 Кбайт (максимальный размер сегмента), и в нем нельзя использовать сегмент стека.
В зависимости от содержимого EXE-файла команда EXE2BIN обеспечивает один из двух типов преобразования:
1) если начальный CS:IP (сегмент кода:указатель инструкции) в EXE-файле не задан, то осуществляется **чистое двоичное преобразование**. При этом в случае, когда требуется зафиксировать сегмент (т.е. когда программа содержит инструкции, подлежащие настройке по месту загрузки), на экран будет выдан запрос на ввод адреса сегмента. Результатом преобразования будет неперемещаемая машинная программа, которую нужно загружать в ОЗУ всегда именно по заданному адресу. Поэтому такая программа должна вызываться на выполнение только специальной прикладной программой, обеспечивающей загрузку по абсолютному адресу памяти. КП DOS не в состоянии это реализовать;

2) если начальным CS:IP является 0000:100H, то EXE2BIN осуществляет **генерацию перемещаемой машинной программы в формате COM-файла,** которая может быть запущена КП DOS. Поэтому для результирующего файла нужно обеспечить расширение COM (в частности, путем его переименования).

Преобразование первого типа сводится к получению абсолютной, а второго типа — к формированию позиционно независимой программы.

Замечание: форматы EXE- и COM-файлов описаны в п. 5.2.2.

Примеры:

- EXE2BIN MYPROG A:PROG — преобразовать файл MYPROG.EXE в двоичный формат и поместить результат в файл A:PROG.BIN;
- EXE2BIN MYPROG — преобразовать файл MYPROG.EXE в двоичный формат и поместить результат в файл MYPROG.BIN;
- EXE2BIN MYPROG PROG.COM — преобразовать файл MYPROG.EXE в двоичный формат и поместить результат в файл PROG.COM.

5.8. Перенаправление ввода–вывода и фильтры

Средства перенаправления ввода-вывода существенным образом расширяют возможности и повышают гибкость командного языка DOS. Сначала мы введем понятие перенаправления ввода-вывода, опишем средства перенаправления ввода-вывода, а затем рассмотрим команды-фильтры, которые базируются на этих средствах.

5.8.1. Общие положения

Под **перенаправлением ввода-вывода** понимают замену *источника данных* при вводе информации в программу или *адресата выводимых результатов* работы программы без ее модификации. Очевидно, источниками данных и адресатами результатов являются файлы, а также посимвольные устройства.

Необходимое условие возможности перенаправления ввода-вывода состоит в обеспечении независимости программ от файлов и устройств. Иными словами, спецификации файлов и имена устройств не должны в явном виде фигурировать в программе. Действительно, если, скажем, файл указан в ней непосредственно, то для замены файла без модификации программы не обойтись.

DOS позволяет сделать программы в рассмотренном смысле независимыми, обеспечивая:

1) *доступ к командной строке запуска программы* с целью извлечения аргументов, в качестве которых можно задать файлы (устройства);

2) возможность *использования* в программах так называемого *стандартного УВВ* (в роли которого, как правило, выступает CON) без его явного указания;

Тем не менее независимость программ от файлов (устройств) сама по себе не обеспечивает широких возможностей перенаправления ввода-вывода, особенно стандартного.

С целью организации перенаправления ввода-вывода команд и программ система поддерживает:

1) замену файлов и устройств как аргументов в командной строке, а также единообразную трактовку файлов и посимвольных устройств;

2) смену стандартного УВВ своими командами, такими, как MODE, CTTY и COMMAND;

3) замену стандартного УВВ в командной строке;

4) направление стандартного вывода одной программы (команды) на стандартный ввод другой программы (команды), т.е. организацию *конвейеров.*

Все перечисленные способы, за исключением первого, применимы только тогда, когда программа (команда DOS) использует стандартный ввод-вывод. Применимость первого способа следует из его формулировки и распространяется на все файлы (устройства).

Вопросы **замены файлов и устройств в командной строке и единообразной их трактовки** уже затрагивались нами ранее. Возможности этого наиболее ярко могут быть проиллюстрированы на примере команды COPY:

- COPY MYFILE.TXT TEXT1.TXT — скопировать содержимое файла MYFILE.TXT в файл TEXT1.TXT;
- COPY MYFILE.TXT TEXT2.TXT — то же, но в качестве адресата используется файл TEXT2.TXT;
- COPY MYFILE.TXT PRN — отпечатать содержимое файла MYFILE.TXT на принтере;
- COPY MYFILE.TXT NUL — скопировать содержимое файла MYFILE.TXT на фиктивное устройство с целью проверки файла на считываемость;
- COPY CON MYFILE.TXT — ввести информацию в файл MYFILE.TXT с клавиатуры.

Краткий отчет о своей работе команда COPY выдает на стандартное УВВ.

Концепция стандартного УВВ сводится к тому, что в системе зарегистрировано *виртуальное* УВВ, к которому обращаются команды (программы) и которое связывается с *реальным* УВВ.

Использование стандартного УВВ в программах имеет то преимущество, что программы получаются независимыми от устройства, причем имя этого устройства передавать в них в качестве аргумента не нужно.

Смена стандартного УВВ командами DOS обеспечивает перенаправление специфицированного для него ввода-вывода у всех запускаемых вслед за этим программ и выдаваемых команд.

Пусть запускаемая по *process* программа или выполняемая по *process* команда использует стандартное УВВ. Тогда **замену стандартного УВВ в командной строке** можно осуществить путем указания в ней одной из следующих конструкций:

process >*file* — перенаправляет стандартный вывод из *process* в файл *file*. Если файл отсутствует, то он создается, а если существует — то заменяется;

process >>*file* — добавляет стандартный вывод из *process* в конец существующего файла *file*. Если файл отсутствует, то он создается и тогда >>*file* эквивалентно >*file*;

process <*file* — перенаправляет стандартный ввод в *process* на существующий файл *file*.

Перечисленные средства могут комбинироваться в одной командной строке.

Примеры:

- DIR >FILELIST.TXT — вывести содержимое рабочего каталога не на экран дисплея, а в файл FILELIST.TXT;
- TYPE MYFILE.TXT >PRN — распечатать содержимое файла MYFILE.TXT на принтере вместо отображения на экране дисплея;
- BACKUP C:*.* A: /S >PRN — выполнить резервирование файловой структуры диска в приводе C на дискетах в приводе A и отпечатать отчет о проделанной работе на принтере;
- COPY MYFILE.TXT TEXT.TXT >NUL — скопировать содержимое файла, но не выдавать отчет;
- PROG <INPUT.TXT >OUTPUT.TXT — обеспечить ввод исходных данных в программу PROG из файла INPUT.TXT, а вывод результатов ее работы — в файл OUTPUT.TXT.

При перенаправлении стандартного ввода-вывода в рабочем каталоге, как правило, создаются временные файлы.

Вот почему при перенаправлении вывода команды DIR без аргумента (т.е. для рабочего каталога) в среде старых версий DOS Вы можете увидеть два новых файла, имена которых составлены из букв и цифр. Это именно те временные файлы. Один из способов избавления от их фиксации в выводе команды DIR состоит в смене текущего привода и задании дисковода в команде DIR явно.

Вместе со средствами перенаправления ввода-вывода полезной оказывается команда ECHO, используемая главным образом в командных файлах. Тем не менее ее можно вводить и с клавиатуры. Один из вариантов ее задания обеспечивает посылку на стандартное УВВ (на экран дисплея) специфицированной в качестве аргумента строки. Это позволяет выводить текст и управляющие символы на принтер или другое устройство, например:

- ECHO ТЕКСТ >PRN — отпечатать строку ТЕКСТ на принтере;
- ECHO ◄Ctrl-L► >PRN — послать на принтер управляющий символ, обеспечивающий прогон бумаги до начала следующей страницы.

Направление стандартного вывода одной программы (команды) на вход другой обеспечивается за счет использования конвейеров. **Конвейером** в этом контексте называется последовательность программ (команд), осуществляющих поэтапную обработку информационного потока. Конвейер синтаксически оформляется как

process1|*process2*|...*processN*

и представляет собой по сути единый, но сложный процесс. Средства конвейеризации программ реализуют один из методов их комплексирования в среде DOS.

Пример:

- ECHO Y|DEL *.* — обеспечить автоматический ответ Y (для «Да») на запрос «Are you sure?» при удалении всех элементов каталога.

Конвейеризация может сочетаться в одной командной строке с другими средствами перенаправления ввода-вывода.

Главным образом для использования в конвейерах DOS содержит три команды-фильтра. **Фильтром** называется программа, вводящая информационный поток, преобразующая его по определенному алгоритму и выводящая результат своей работы. К фильтрам DOS относятся следующие команды:

FIND — осуществляет поиск заданной последовательности символов в текстовых файлах;

MORE — обеспечивает постраничное отображение содержимого текстового файла;

SORT — сортирует строки текстового файла в лексикографическом порядке.

Заметим, что DOS обеспечивает, хотя и весьма ограниченные, возможности подмены поблочных устройств, даже если программа от них зависима (см. описание команд ASSIGN и SUBST).

5.8.2. Команда MORE

Назначение: постраничное (поэкранное) отображение содержимого текстового файла.
Тип: внешняя.
Синтаксис:

MORE [<*file*]

или

process | MORE

Комментарии. Фильтр MORE считывает текстовый файл со стандартного устройства ввода (которое практически всегда подменяется) и отображает его содержимое с паузой каждый раз после заполнения экрана дисплея. При этом в последней (нижней) строке экрана выдается сообщение —MORE—. Для вывода следующей страницы достаточно нажать на клавиатуре любую клавишу.

Замечания:
— ввод текстового файла с клавиатуры допустим, но делать это не имеет смысла. Если Вы все же забыли перенаправить стандартный ввод команды MORE, она будет ждать задания файла с клавиатуры. В этом случае следует нажать Ctrl-Z или F6, а затем — Enter;
— фильтр MORE DOS 3.3 создает временный файл в рабочем каталоге. Если этот диск полностью занят или защищен от записи, MORE работать не будет.

Примеры:

- DIR | MORE — осуществить постраничное отображение содержимого рабочего каталога;
- MORE < READ.ME — осуществить постраничное отображение содержимого файла READ.ME;
- TYPE READ.ME | MORE — то же.

5.8.3. Команда SORT

Назначение: сортировка строк текстового файла в лексикографическом (алфавитном) порядке.
Тип: внешняя.
Синтаксис:

SORT [/R] [/+*n*] [<*file*]

или

process | SORT [/R] [/+*n*]

Комментарии. Фильтр SORT считывает строки текстового файла со стандартного устройства ввода и сортирует их в лексикографическом (алфавитном) порядке в соответствии с выбранным кодом страны, а также установленной кодовой страницей. Стандартное устройство ввода часто подменяется, что отражено в синтаксисе команды.
В командной строке допустимы следующие переключатели:

/R — сортировать в обратном порядке (от Z к A и от 9 к 0);

/+*n*— учитывать при сортировке фрагменты строк, начинающиеся с *n*-го символа (*n* — число). Если этот переключатель не задан, то предполагается 1 (первый символ).

Замечания:
— фильтр SORT не различает одноименные строчные и прописные буквы;
— правильная сортировка строк с кириллицей не обеспечивается, если кодовая страница ее не содержит.

Примеры:

- SORT /R < MYFILE.TXT — отсортировать строки файла MYFILE.TXT в обратном порядке и вывести результат на экран дисплея;
- SORT /R < MYFILE.TXT > SORTRES.TXT — то же, но результат поместить в файл SORTRES.TXT;
- DIR | SORT — отсортировать содержимое каталога по имени файла и вывести результат на экран дисплея;
- DIR | SORT /+10 — то же, но сортировку осуществить по расширению имен файлов;
- DIR | SORT /+14 — то же, но сортировку осуществить по размеру файлов;
- DIR | SORT /+14 | MORE — то же, но осуществить постраничный вывод.

5.8.4. Команда FIND

Назначение: поиск заданной последовательности символов в одном или нескольких текстовых файлах.
Тип: внешняя.
Синтаксис:

FIND [/V] [/C] [/N] *"string"* [*file*]...

или

process|FIND [/V] [/C] [/N] *"string"*

Комментарии. Команда FIND считывает содержимое заданных в командной строке аргументами *file* текстовых файлов и осуществляет поиск в них строк, содержащих последовательность символов (подстроку) *string*. Если ни один из аргументов *file* не указан, то чтение информации производится построчно со стандартного устройства ввода (с клавиатуры). В этом случае его можно подменить, как показано во второй форме команды.

Когда ни один из переключателей не задан, фильтр FIND выдает на стандартное устройство вывода (экран дисплея) те строки текстовых файлов, в которых найдена подстрока *string*.

Аргумент *string* заключается в кавычки (но не в двойные апострофы). Если в string нужно указать кавычки, то их следует продублировать.

Допустимы следующие переключатели:

/V — выводить только строки, не содержащие подстроку *string*;

/C — вместо строк выводить их порядковые номера в соответствующих файлах;

/N — перед каждой выдаваемой строкой вывести и ее порядковый номер в файле.

Если одновременно указать переключатели /V и /C, то будут выданы номера всех строк, не содержащих подстроку *string*. Если задать только /C, то выведутся номера строк, содержащих подстроку *string*. В случае одновременного задания переключателей /C и /N последний из них игнорируется.

Замечание: фильтр FIND различает одноименные строчные и прописные буквы.

Примеры:

- FIND /N "ReadLn" MYPROG.PAS P.PAS — вывести на экран дисплея строки файлов MYPROG.PAS и P.PAS, содержащие подстроку ReadLn, а также номера этих строк;
- DIR|FIND /V "i" >FILE.LST — записать в файл FILE.LST содержимое рабочего каталога с игнорированием следующих строк:

```
Volume in drive ...
Directory of drive ...
... files ... bytes free
```

Каждая из них содержит символ i. В других же строках, описывающих элементы каталога, буква i встретиться не может, так как имена и расширения имен файлов представлены на диске прописными буквами;

- DIR|FIND /V "i"|SORT >FILE.LST — то же, но перед записью в FILE.LST список содержащихся в каталоге файлов отсортировать по именам.

5.9. Командные файлы

Команды DOS, служащие для использования в командных файлах, образуют простой, но довольно развитый, хотя и несколько архаичный язык программирования, выразительные возможности которого достаточны для решения практических задач. Чтобы составлять сложные командные файлы, желательно иметь навыки программирования на одном из процедурных языков высокого уровня, так как в наши цели не входит обучение этому. Тем не менее приводятся все необходимые сведения по средствам, используемым в командных файлах.

Сначала рассматриваются общие положения по разработке командных файлов. Затем описываются дополнительные команды DOS для использования в командных файлах. Подраздел завершается рядом примеров, большинство из которых решают практические задачи, часто возникающие перед пользователем. Советуем внимательно ознакомиться с примерами, так как они иллюстрируют многие полезные методы разработки командных файлов.

5.9.1. Общие сведения о командных файлах

Командным (пакетным — от batch) файлом называется последовательность команд DOS, записанная в текстовый файл и выполняемая путем ввода спецификации этого файла с клавиатуры аналогично единственной команде DOS. Такой файл представляет собой системную макрокоманду и является аналогом процедуры в программах. Наличие средств создания и использования командных файлов делает командный язык DOS *расширяемым*.

Командные файлы предназначены для упрощения задания и выполнения часто используемых последовательностей команд системы. При помощи этих файлов пользователь может без особого труда создать свой интерфейс с DOS. В частности, командные файлы полезны при выдаче команд, требующих многих аргументов и переключателей, чтобы каждый раз не вводить с клавиатуры длинную командную строку.

Командный файл может содержать любые команды, допустимые в командной строке, вводимой в ответ на приглашение DOS, и запросы на выполнение программ. Кроме того, имеются дополнительные команды, используемые главным образом только в таких файлах, хотя некоторые из них с определенными ограничениями могут быть заданы и с клавиатуры. Каждая строка командного файла может содержать только одну команду или запрос на выполнение программы.

Командные файлы обычно создаются каким-либо текстовым редактором, в частности EDLIN. Маленькие же файлы зачастую удобнее сформировать командой COPY CON *file*. Каждый командный файл должен иметь расширение BAT (от BATch).

Например, для создания командного файла, отображающего содержимое текущих каталогов дискет в приводах A и B, достаточно ввести с клавиатуры следующую информацию:

```
COPY CON DIRAB.BAT◄Enter►
CLS◄ Enter ►
DIR A:◄ Enter ►
DIR B:◄ Enter ►
◄ F6 ►◄ Enter ►
```

Командный файл запускается обычно путем ввода его спецификации в ответ на приглашение DOS, причем расширение BAT допускается не указывать, например: DIRAB. Кроме спецификации файла можно задать последовательность аргументов (в текстовом виде), отделенных друг от друга пробелами (одним или несколькими), запятой или точкой с запятой. В рамках же командного файла имеется возможность извлечь эти аргументы.

Командные файлы обрабатываются КП DOS построчно, что замедляет их выполнение, но вместе с тем позволяет создавать самомодифицирующиеся командные файлы. Каждая прочитанная строка отображается на экране дисплея (если не было предпринято специальных мер) и интерпретируется. Затем считывается следующая строка и т.д. Если во время выполнения командного файла пользователь извлечет из привода дискету, его содержащую, то DOS перед считыванием следующей строки предложит установить ее снова.

При необходимости прервать выполнение командного файла следует нажать комбинацию клавиш Ctrl-Break, в результате чего появится следующее сообщение:

```
Terminate bath job (Y/N)?
(Завершить пакетное задание (Y—да/N—нет)?)
```

В случае ответа Y выполнение командного файла принудительно прекратится, а N — будет продолжено.

При создании и использовании командных файлов полезно знать следующее:

— в конце одного командного файла можно специфицировать имя другого с тем, чтобы последовательно выполнить два файла путем указания в командной строке только первого из них. Следует иметь в виду, что указание командного файла внутри другого командного файла приводит к безусловной передаче управления без последующего возврата в него;

— внутри командных файлов допускается использовать все имеющиеся в DOS средства перенаправления ввода-вывода;

— стандартный ввод-вывод всего командного файла как единого целого перенаправлять не допускается;

— в начале строки командного файла можно указать символ @, вследствие чего данная строка не будет отображаться на экране дисплея перед ее интерпретацией (но символ @ не влияет на стандартный ввод-вывод команды или программы);

— текущий привод, текущие каталоги и состояние окружения DOS при входе в командный файл не изменяются.

Средства DOS для поддержки командных файлов позволяют:

— создавать линейные командные файлы;
— организовывать разветвления в командных файлах;
— создавать циклы в командных файлах;
— разрабатывать вложенные командные файлы;
— производить вывод информации из командных файлов;
— приостанавливать интерпретацию командных файлов для осуществления тех или иных действий;
— осуществлять параметризацию командных файлов;
— управлять отображением выполняемых строк командных файлов на экране дисплея;
— включать в командные файлы комментарии.

Тем не менее DOS не предоставляет удобных средств для создания интерактивных командных файлов. Хотя такая возможность в принципе и имеется, но требуется разработка специальных программ, выдающих в зависимости от ответа пользователя различные коды возврата. Эти программы включаются в командные файлы, и по возвращаемым ими кодам организуются разветвления. Вместо создания собственной универсальной или специализированной программы пользователь может воспользоваться одной из имеющихся на рынке утилит, например утилитой Batch Enhancer из комплекта Norton Utilities.

Если Вы хотите выполнять одну и ту же последовательность команд при каждой загрузке DOS, создайте файл AUTOEXEC.BAT и поместите его в корневой каталог системного диска. Обычно в файле автозапуска AUTOEXEC.BAT размещают команды DATE, TIME, PROMPT, SET, PATH и (при необходимости) APPEND, а также загружают различные резидентные программы, в частности, драйверы. Последней строкой файла автозапуска, как правило, запускается одна из оболочек DOS (в частности, Norton Commander).

5.9.2. Параметризация командных файлов

Параметризация командного файла сводится к возможности использования в его теле переменных и их означивание перед запуском командного файла на выполнение. Такая техника обеспечивает разработку универсальных командных файлов и настройку их на конкретные условия применения путем передачи этим файлам той или иной информации (в частности, имен обрабатываемых файлов). Важно то, что настройка командного файла не требует его модификации.

DOS поддерживает два *механизма (способа) передачи информации* в командные файлы:

1) *передачу через аппарат параметров*;

2) *передачу через окружение DOS.*

Первый способ состоит в том, что в теле командного файла используются параметры, а при его вызове задаются аргументы, замещающие их. В командном файле допускается указывать до девяти **параметров** с именами %1 — %9. Порядок задания аргументов в командной строке описан в предыдущем пункте. **Аргумент** может состоять из любой последовательности символов, за исключением пробела. Действует принцип позиционного соответствия параметров и аргументов, т.е. вместо параметра *%i* подставляется *i*-й аргумент. В случае отсутствия аргумента параметр заменяется пустой строкой, а избыточные аргументы отбрасываются. В теле командного файла допускается также использовать параметр %0, вместо которого подставляется (возможно, неполная) спецификация данного командного файла, взятая из командной строки. Полезность этого неочевидна, но мы все же продемонстрируем ее примерами несколько позже. При необходимости задания большего, чем 9, числа аргументов не обойтись без команды SHIFT. Она сыграет положительную роль также в случае указания различного количества аргументов при том или ином вызове командного файла. Если внутри командного файла, скажем в имени файла, требуется задать символ %, то его следует продублировать (%%), чтобы избежать коллизии с признаком параметра или начала ссылки на значение глобальной переменной.

В качестве примера рассмотрим командный файл PR, осуществляющий печать текстового файла с начала страницы:

```
@ECHO OFF
ECHO ◄Ctrl-L►  >PRN
TYPE %1 >PRN
```

Запустить его на выполнение можно командной строкой

```
PR MYTEXT.TXT
```

Второй способ передачи информации в командный файл сводится к предварительному присваиванию (командой SET) глобальной переменной требуемого значения и последующему использованию этого значения в теле командного файла. Чтобы получить в командном файле доступ к значению глобальной переменной, следует ее имя заключить в символы %. Вместо отсутствующей в окружении глобальной переменной подставляется пустая строка.

Например, командный файл, добавляющий к имеющимся новый маршрут поиска исполняемых файлов, можно представить следующим образом:

```
SET PATH=%PATH%;%1
```

Очевидно, глобальные переменные можно использовать в командных файлах аналогично переменным в программах на процедурных языках программирования: только что рассмотренная команда семантически эквивалентна оператору присваивания.

Однозначный ответ на вопрос о том, какой из рассмотренных способов передачи информации в командный файл лучше, отсутствует. Второй из них основан на побочном эффекте, что обычно считается плохой дисциплиной программирования. Однако только данный способ обеспечивает доступ к окружению DOS и избавляет пользователя от явного задания всех аргументов. Это особенно полезно, когда некоторые из них не меняются при последовательных запусках командного файла.

5.9.3. Использование символа @

Как уже отмечалось в п. 5.9.1, символ @, записанный в начале строки командного файла, запрещает ее отображение на экране дисплея перед исполнением. Это нужно, чтобы скрыть какую-то информацию, не портить изображение на экране или не захламлять экран ненужными сообщениями. Следует четко различать отображение команды на экране (*эхо-отображение*) и *вывод результатов* ее выполнения на дисплей. Второй тип сообщений символом @ не подавляется, а может использовать перенаправление ввода-вывода.

Например, если в командном файле имеется строка COPY A:MYFILE.TXT B:, то в результате ее интерпретации на экран будет выдано следующее:

```
C:>COPY A:MYFILE.TXT B:
1 File(s) copied
```

Если эту строку предварить символом @ (т.е. записать ее в виде @COPY A:MYFILE.TXT B:), то отобразится лишь последнее (второе) сообщение. И только в случае дополнительного перенаправления стандартного вывода никакого сообщения на экране дисплея не появится. Это наиболее удобно реализовать, поместив в командный файл строку @COPY A:MYFILE.TXT B: >NUL.

Символ @ действует только на ту строку командного файла, в начале которой он записан. Для управления эхо-отображением последовательностей строк следует использовать команду ECHO.

5.9.4. Дополнительные команды для командных файлов

В этом пункте мы рассмотрим следующие команды, которые используются главным образом только в командных файлах:

CALL — осуществляет вызов заданного командного файла с последующим возвратом в точку вызова;

ECHO — производит включение и отключение эхо-отображения строк командного файла, а также выдает текущий статус и отображает заданный текст (сообщение);

FOR — организует цикл в командном файле;

GOTO — обеспечивает безусловный переход к строке командного файла с заданной меткой;

IF — осуществляет разветвление в командном файле;

PAUSE — организует паузу при выполнении командного файла;

REM — самодокументирует командный файл;

SHIFT — изменяет соответствие параметров аргументам командного файла.

Перечисленные команды увеличивают выразительность и гибкость командных файлов.

Команда ECHO

Назначение: включение и отключение эхо-отображения строк командного файла при их интерпретации, а также выдача текущего статуса и отображение заданного сообщения.

Тип: внутренняя.

Синтаксис:

ECHO [ON|OFF|*сообщение*]

Комментарии. Обычно команды командного файла выводятся на экран дисплея (вместе с приглашением DOS) при их интерпретации КП. Если Вы хотите отключить эхо-отображение для последующих команд, задайте OFF. Это позволит скрыть некоторую информацию, не искажать имеющееся на экране изображение и не захламлять экран. Для включения эхо-отображения нужно указать ON. Если аргумент вообще отсутствует, то на экран (как стандартное устройство вывода) выдается текущий статус («ECHO is ON» или «ECHO is OFF»).

Когда в качестве аргумента задано *сообщение*, то оно без каких-либо изменений отображается на экране дисплея для информирования пользователя. Сообщение представляется любой последовательностью символов, содержащей, по крайней мере, один отличный от пробела символ. Сообщением считаются все символы в строке, за исключением единственного пробела после имени команды ECHO. Однако символы >, < и | в сообщении недопустимы.

Замечания:

— простота команды ECHO обманчива;

— эта команда является чуть ли не единственным средством DOS, обеспечивающим вывод информации из командных файлов;

— для исключения дублирования сообщений перед командами вида ECHO *сообщение* обычно размещают команду ECHO OFF (@ECHO OFF) или вместо них используют команды @ECHO *сообщение*;

— при отладке командного файла отключать эхо-отображение не следует;

— для отображения сообщения, состоящего из нескольких строк, следует поместить в командный файл несколько команд ECHO;

— для вывода на экран пустой строки с целью разграничения фрагментов текста нужно использовать команду в виде ECHO: или с сообщением из неотображаемого символа;

— перенаправление ввода-вывода в команде ECHO существенно расширяет ее возможности, обеспечивая посылку последовательностей символов на устройства (в частности, для управления ими) и в файлы (что является наиболее быстрым способом создания простейших текстовых файлов);

— включать эхо-отображение в конце командного файла нет необходимости, так как DOS делает это автоматически;

— команду ECHO можно использовать не только в командных файлах, но и вводить с клавиатуры, однако форма ECHO OFF в последнем случае действовать не будет.

Примеры:

- @ECHO OFF
  ECHO:
  ECHO Этот командный файл
  ECHO форматирует и проверяет
  ECHO новый диск
  ECHO:
  ECHO ON — выдать на экран дисплея сообщение «Этот командный файл форматирует и проверяет новый диск» на пяти строках;
- ECHO ◄ Ctrl-O► >PRN — установить IBM-совместимый принтер в режим печати сжатым шрифтом;
- ECHO ◄ Ctrl-L► >PRN — осуществить прогон бумаги на принтере до начала следующей страницы;
- ECHO NC >>C:\AUTOEXEC.BAT — добавить в конец файла AUTOEXEC.BAT строку, содержащую NC.

Последние три команды могут быть использованы как в командном файле, так и в ответ на приглашение DOS.

Команда PAUSE

Назначение: организация паузы при выполнении командного файла.
Тип: внутренняя.
Синтаксис:

PAUSE [*сообщение*]

Комментарии. Эта команда используется для приостановки интерпретации командного файла перед выполнением критических (опасных) операций или для выполнения пользователем определенных действий (например, установки дискеты). По команде PAUSE обработка командного файла временно прекращается и на экран выдается сообщение

Strike a key when ready...
(Затем нажмите любую клавишу)

Если пользователь в ответ на него нажмет комбинацию клавиш Ctrl-Break, то появится сообщение

Terminate batch job (Y/N)?
(Завершить пакетное задание (Y—да/N—нет)?)

При нажатии клавиши N выполнение командного файла будет продолжено, а Y — прекращено. Любой другой ответ приведет к повторению запроса.

Если пользователь в ответ на сообщение «Strike a key when ready...» нажмет любую клавишу пишущей машинки (обычно Space), то интерпретация командного файла будет продолжена без каких-либо дополнительных сообщений.

Текст «Strike a key when ready...» используется в качестве дополнения к информационному сообщению, которое выдано на экран раньше, в частности, при эхо-отображении команды PAUSE с заданным аргументом, а именно, сообщением. Аргументом может быть произвольная последовательность символов (текст), за исключением символов <, > и |.

Замечания:

— аргумент *сообщение* командой PAUSE в отличие от команды ECHO на стандартное устройство вывода не выдается. Поэтому если эхо-отображение отключено (командой ECHO OFF или символом @), то на экране появится только сообщение «Strike a key when ready...». В этом случае для предварительной выдачи информационного сообщения следует использовать команду ECHO;

— команду PAUSE можно вводить с клавиатуры, но полезность этого усмотреть трудно;

— в некоторых версиях DOS вместо «Strike a key when ready...» используется сообщение «Press a key when ready...» или «Press any key to continue...».

Примеры:

- выполнение команды

  PAUSE Установите дискету в привод B:

  приведет к отображению сообщений

  C:>PAUSE Установите дискету в привод B:
  Strike a key when ready...

- выполнение команд

```
@ECHO OFF
ECHO Установите дискету в привод B:
PAUSE
ECHO ON
```

обеспечит выдачу сообщений

```
Установите дискету в привод B:
Strike a key when ready...
```

- интерпретация последовательности команд

```
@ECHO OFF
ECHO Установите дискету в привод B:
ECHO и нажмите любую клавишу для продолжения
PAUSE > NUL
ECHO ON
```

приведет к появлению сообщений

```
Установите дискету в привод B:
и нажмите любую клавишу для продолжения
```

Команда REM

Назначение: самодокументирование командного файла.
Тип: внутренняя.
Синтаксис:

REM [*комментарий*]

Комментарии. Команда REM не приводит к выполнению каких-либо действий, но позволяет включать в тело командного файла текстовую информацию для его документирования (посредством аргумента *комментарий*, в качестве которого можно использовать любую последовательность символов без каких-либо исключений). Если *комментарий* состоит только из пробелов или вообще отсутствует, то это равносильно размещению в файле пустой строки для наглядности.

Замечания:
— команды REM обычно помещают во фрагменты командного файла с отключенным эхо-отображением (или используют символ @);
— команду REM удобно использовать при отладке командного файла для временного запрета интерпретации команды путем вставки REM перед ней. При необходимости REM можно легко удалить;
— строка с меткой также может быть использована для комментирования командного файла (см. описание команды GOTO).

Пример самодокументированного командного файла:

```
@ECHO OFF
REM Этот файл форматирует и тестирует новый диск в приводе %1
REM Он имеет имя CHECKNEW.BAT
REM
ECHO Установите дискету в привод %1
ECHO и затем нажмите любую клавишу
PAUSE >NUL
FORMAT %1 /V
CHKDSK %1
```

Этот файл может быть вызван на выполнение командной строкой CHECKNEW A:

Команда CALL

Назначение: вызов одного командного файла из другого с последующим возвратом в точку вызова.
Тип: внутренняя.
Синтаксис:

CALL *file* [*arglist*]

Комментарии. По данной команде осуществляется запуск командного файла, заданного посредством *file* с передачей ему указанных аргументов *(arglist)*. Этот файл должен существовать и иметь расширение BAT. В спецификации *file* расширение можно не указывать. Когда выполнение командного файла *file* завершается, управление возвращается в точку его вызова, т.е. в вызывающий файл на непосредственно следующую за CALL команду.

Замечания:

— команда CALL обеспечивает возможность создания вложенных командных файлов, что облегчает разработку и модификацию сложных командных файлов за счет их структуризации;

— в ранних (до 3.3) версиях DOS команда CALL отсутствовала и поэтому ее функции могла выполнить только команда COMMAND с переключателем /C. Последнюю допускается использовать для вызова с возвратом и в современных версиях DOS, однако это потребует больших накладных расходов и поэтому теряет всякий смысл;

— осуществлять перенаправление ввода-вывода в команде CALL не допускается;

— уровень вложенности командных файлов не должен превышать восьми;

— состояние эхо-отображения наследуется вызываемым командным файлом, но обратно измененное состояние не возвращается;

— с использованием команды CALL можно создавать рекурсивные (обращающиеся к самим себе) командные файлы, но нужно обеспечить корректное завершение рекурсивных вызовов.

Пример: CALL C:\BAT\CHECKNEW B:

Команда GOTO

Назначение: безусловный переход к строке командного файла с заданной меткой.
Тип: внутренняя.
Синтаксис:

GOTO [:]*метка*

Комментарии. Данная команда позволяет нарушить обычную линейную последовательность исполнения строк командного файла. После выполнения команды GOTO интерпретация командного файла продолжается со строки, текстуально следующей за строкой с заданной меткой. *Метка* представляет собой последовательность алфавитно-цифровых символов без использования разделителей (специальных символов). Длина метки не ограничивается, но DOS учитывает только первые 8 символов. Строка с меткой должна начинаться с символа :, непосредственно за которым указывается сама метка. За меткой (через пробел) может следовать любая последовательность символов, воспринимаемая как комментарий. Если командный файл не содержит строку с меткой, указанной в команде GOTO, то его выполнение по GOTO завершается.

Замечания:

— команда GOTO совместно с командой IF используется для организации в командных файлах циклов и разветвлений;

— строку с меткой можно использовать для комментирования командного файла (достаточно начать строку символом :, а не командой REM);

— на ошибочность задания метки в строке командного файла (но не в команде GOTO) DOS не реагирует, поэтому в «псевдокомментарии» допустимы любые символы;

— указание в GOTO несуществующей метки используется для прекращения выполнения командного файла;

— любая строка командного файла, начинающаяся с двоеточия, при его обработке игнорируется.

Пример:

```
@ECHO OFF
:AGAIN Простой циклический командный файл
ECHO Нажмите Ctrl-Break для остановки
GOTO AGAIN
```

Команда IF

Назначение: организация разветвлений в командных файлах.
Тип: внутренняя.
Синтаксис:

IF [NOT] *условие process*

Комментарии. Команда IF позволяет выполнить или пропустить указанный в ней *process* в зависимости от заданного *условия*. При отсутствии NOT специфицированный *process* выполняется, если *условие* принимает значение «истина», т.е. удовлетворяется. При задании NOT *process* выполняется, если *условие* принимает значение «ложь» (не удовлетворяется). В других случаях специфицированный *process* не выполняется, а просто пропускается и осуществляется интерпретация текстуально следующей строки. Команду IF можно прочитать следующим образом: если [не] *условие*, то выполнить *process*.

Условие может быть задано одним из следующих способов:

EXIST *pattern* — существует, по крайней мере, один файл, сопоставимый со спецификацией шаблона *pattern*;

string1 = = *string2* — символьные строки *string1* и *string2*, возможно, после замещения в них параметров аргументами, идентичны. Строки могут содержать любые символы, за исключением следующих: пробел, запятая, =, ;, <, > и |. В ходе сравнения проводится различие между одноименными строчными и прописными буквами;

ERRORLEVEL *n* — последняя выполненная программа или внешняя команда DOS (но не обязательно по указаниям из предыдущей строки командного файла) выдала код возврата, не меньший (не обязательно равный!) чем целое число *n*;

В связи с тем, что код возврата анализируется не на равенство, а на больше либо равно, **для организации разветвления по многим направлениям** следует использовать следующую схему:

```
process
IF ERRORLEVEL n GOTO меткаN
...
IF ERRORLEVEL 2 GOTO метка2
IF ERRORLEVEL 1 GOTO метка1
REM IF EROORLEVEL 0
...
GOTO END
:метка1
...
GOTO END
:метка2
...
GOTO END
...
:меткаN
...
:END
```

Отсюда видно, что анализ кода возврата нужно начинать с наибольшего возможного значения. Кроме того, даже если альтернативная ветвь состоит из единственной команды, последнюю нельзя указывать непосредственно за условием, так как в этом случае после ее выполнения удовлетворится условие и в следующей команде IF, что приведет к неправильной работе командного файла. Корректным будет только безусловный переход к фрагменту командного файла, реализующему альтернативную ветвь, и выходу из нее снова по команде GOTO. Очевидно, при этом альтернативная ветвь может содержать и несколько команд.

Если ситуация отличается от описанной, но Вам все же требуется при удовлетворении какого-либо условия выполнить несколько команд, то в качестве альтернативы рассмотренному способу можно оформить эти команды отдельным командным файлом и вызвать его в команде IF по CALL.

Замечания:

— команда IF совместно с GOTO позволяет создавать в командных файлах не только разветвления, но и циклы;

— если после *условия* в команде IF указать спецификацию командного файла, то возврат обратно (как после CALL) не произойдет;

— команды IF могут быть вложенными;

— любую программу можно организовать таким образом, чтобы она выдавала код возврата, зависящий от каких-либо условий, что позволяет (с привлечением команды IF) создавать интерактивные командные файлы;

— некоторые внешние команды (см. подраздел 5.6) также выдают коды возврата, которые можно использовать для организации разветвлений в командных файлах;

— если в качестве одной из строк конструкции *string1* = = *string2* требуется задать пустую строку, то следует обе строки заключить в одни и те же разделители, например: *.string1.* = =.., *string1"* = = "", */string1/* = =// или даже */string1* = =/.

Примеры:

- IF NOT EXIST D:BASCOM.COM COPY C:\BASIC\BASCOM.COM D: — если файл BASCOM.COM на диске в приводе D не существует, то скопировать его туда;
- IF %1 = =LEX CHDIR C:\EDITORS\LEX
  IF %1 = =lex CHDIR C:\EDITORS\LEX — установить на диске в приводе C текущим каталог \EDITORS\LEX, если первым аргументом командного файла задан каталог LEX или lex;
- BACKUP C:*.* A: /S
  IF ERRORLEVEL 3 GOTO TROUBLE — если команда BACKUP завершается с кодом возврата 3 или выше, то осуществить переход к метке TROUBLE;
- IF "%1" = = " GOTO M1
  CD %1
  GOTO M2
  :M1
  CD C:\LANGS\TP:
  M2 — если аргумент при запуске командного файла задан, то установить текущий каталог по нему; в противном случае изменить текущий каталог на C:\LANGS\TP;

- ```
  @ECHO OFF
  :BEGIN
  FORMAT A: /S
  IF NOT ERRORLEVEL 1 GOTO END
  ECHO Системный диск не создан
  GOTO BEGIN
  :END
  ECHO Системный диск создан
  ```
  — форматировать дискету в приводе A и переносить на нее систему до тех пор, пока эти операции не завершатся успешно (с кодом возврата 0);
- IF /%1==/ CALL TEST MYFILE.TXT — если аргумент при запуске командного файла не задан, то выполнить командный файл TEST.BAT с аргументом MYFILE.TXT.

Команда FOR

Назначение: организация цикла в командном файле.
Тип: внутренняя.
Синтаксис:

FOR *%%символ* IN *(список)* DO *process*

Комментарии. Данная команда служит для многократного выполнения *process* с различными значениями параметра цикла (локальной переменной).

Параметр цикла (в синтаксисе команды — *символ*) представляется единственным символом, отличным от цифры и %, чтобы он не вступал в противоречие с параметром командного файла. Обычно параметр цикла обозначается буквой.

Список — это последовательность цепочек символов, разделенных пробелами или запятыми. В качестве элемента списка (цепочки символов) можно задать и спецификацию шаблона файла, но тогда этот элемент должен быть единственным (остальные элементы игнорируются). Он обозначает список файлов, с ним сопоставимых.

В аргументе *process*, задающем выполнение команды или программы, как правило, фигурирует параметр цикла.

Семантика команды FOR состоит в том, что указанный *process* многократно выполняется с каждым значением параметра цикла из заданного *списка*, причем в том порядке, в котором эти значения перечислены.

Если требуется поместить в тело цикла (после DO) несколько команд, то следует записать их в отдельный командный файл и использовать для его вызова команду CALL.

Замечания:

— команда FOR не обладает полнотой (не позволяет организовать все возможные циклы). Поэтому в ряде случаев необходимо пользоваться комбинациями команд IF и GOTO. Вместе с тем FOR позволяет решить многие задачи наиболее простым способом;

— команды FOR не могут быть вложенными;

— спецификация командного файла в качестве *process* недопустима;

— параметр цикла начинается с двух символов %, чтобы КП DOS было легче отличить его от параметра командного файла и значения глобальной переменной;

— команда FOR может быть использована не только в командном файле, но и непосредственно введена с клавиатуры. В последнем случае параметр цикла должен начинаться с одного символа %.

Примеры:

- FOR %%D IN (A, B, C) DO DIR %%D:*.* — выдать на экран содержимое текущих каталогов дисков в приводах A, B и C;
- FOR %%F IN (*.TXT) DO TYPE %%F — отобразить содержимое всех TXT-файлов, находящихся в рабочем каталоге (единственной командой TYPE этого достичь нельзя, так как она не допускает шаблонов);
- FOR %%F IN (*.ASM) DO MASM %%F — провести ассемблирование всех ASM-файлов из рабочего каталога;
- FOR %%F IN (*.PAS) DO CALL COMPILE %%F — выполнить командный файл COMPILE.BAT для каждого PAS-файла из рабочего каталога в качестве аргумента;
- FOR %F IN (A:*.*) DO DEL C:%F — удалить из текущего каталога диска в приводе C все файлы, содержащиеся в текущем каталоге диска в приводе A, например, после ошибочного копирования файлов в непустой каталог. Эта команда (так как проставлен единственный символ %) может быть введена только в командной строке.

Команда SHIFT

Назначение: изменение соответствия параметров аргументам командного файла (сдвиг параметров относительно аргументов на одну позицию вправо).
Тип: внутренняя.
  ```

Синтаксис:

SHIFT

Комментарии. Команда SHIFT полезна в тех случаях, когда в командном файле требуется обеспечить, по крайней мере, одну из следующих возможностей:

1) обработку неопределенного числа аргументов;
2) обработку более девяти аргументов;
3) однообразную циклическую обработку аргументов.

При каждом выполнении команды SHIFT i-й (i = 1,...,8) параметр командного файла принимает значение (i + 1)-го параметра. Значение параметра %0 теряется, а параметр %9 связывается с первым слева из еще свободных аргументов. Иными словами, по SHIFT осуществляется сдвиг параметров относительно аргументов в сторону увеличения на единицу номеров аргументов.

Замечание: DOS не имеет команды, действие которой противоположно команде SHIFT. В частности, после однократного сдвига доступ к первому аргументу (даже посредством %0) становится невозможным, если не сохранить его в качестве значения глобальной переменной.

Пример командного файла, удаляющего заданные списки файлов:

```
@ECHO OFF
REM Файл MULTIDEL.BAT: удаляет списки файлов
REM Пример использования: MULTIDEL TEMP.TXT, *.BAK, TEST.PAS
:LOOP
DEL %1
SHIFT
IF NOT .%1.==.. GOTO LOOP
```

5.9.5. Примеры командных файлов

1. Командный файл SHOWANY.BAT, обеспечивающий постраничное отображение содержимого заданного текстового файла (с паузой после заполнения каждого экрана):

```
@ECHO OFF
IF /%1=/ GOTO ERROR1
IF NOT EXIST %1 GOTO ERROR2
MORE <%1
GOTO END
:ERROR1 Не задан аргумент
ECHO Вы должны специфицировать файл
GOTO END
:ERROR2 Файл отсутствует
ECHO Файл %1 отсутствует
:END
```

2. Командный файл SHIFTEST.BAT для исследования работы команды SHIFT:

```
@ECHO OFF
ECHO %0
:BEGIN
IF /%1==/ GOTO END
ECHO %1 %2 %3 %4 %5 %6 %7 %8 %9
SHIFT
GOTO BEGIN
:END
```

После его подготовки попробуйте задать

SHIFTEST A B C D E F G H I J K L M N O P Q R S

3. Командный файл ASMTOCOM.BAT для получения COM-программы из программы на языке Макроассемблера:

```
@ECHO OFF
MASM %1
LINK %1
DEL %1.OBJ
EXE2BIN %1 %1.COM
DEL %1.EXE
```

Если программа на языке Макроассемблера находится в файле MYPROG.ASM, то для ее обработки в ответ на приглашение DOS следует ввести

ASMTOCOM MYPROG

Расширение специфицировать нельзя.

4. Командный файл SETPATH.BAT, обеспечивающий добавление маршрутов поиска исполняемых файлов к уже имеющимся и отображение всех установленных маршрутов:

```
@ECHO OFF
:START
IF /%1==/ GOTO END
SET PATH=%PATH%;%1
SHIFT
GOTO START
:END
PATH
```

Пример использования:

```
SETPATH  C:\EDITORS\TEXT\ME  C:\DOS33
```

Рассмотренный командный файл работает нормально, только если значение глобальной переменной PATH помещается на одной строке экрана.

5. Командный файл DELBAK.BAT для удаления резервных копий файлов (BAK-файлов) из заданного каталога, если он указан, или из рабочего каталога, если аргумент не специфицирован:

```
@ECHO OFF
SET #DIR=
IF NOT /%1==/ SET #DIR=%1\
IF EXIST %#DIR%*.BAK GOTO DELETE
ECHO Нет BAK-файлов для удаления
GOTO END
:DELETE
DIR %#DIR%*.BAK
ECHO Если Вы не хотите удалять перечисленные файлы,
ECHO то нажмите Ctrl-Break, а иначе -
ECHO любую клавишу для продолжения
PAUSE >NUL
DEL %#DIR%*.BAK
ECHO *** Файлы удалены ***
:END
```

В этом файле использована глобальная переменная с уникальным именем #DIR. Сначала ее значение сбрасывается командой SET, чтобы оно обязательно было пустым, а не осталось установленным после предыдущего вызова файла DELBAK.BAT. Затем, если аргумент задан, то к нему справа приписывается символ \ и результат сохраняется в качестве значения глобальной переменной #DIR. В последующих командах оно применяется в качестве префикса к составным именам BAK-файлов. Имейте в виду, что указание *d:* в качестве аргумента приведет к удалению BAK-файлов не в текущем, а в корневом каталоге диска в приводе *d*. Поэтому если Вам требуется задать текущий каталог, то используйте конструкцию *d:.* .

6. Командный файл LEX.BAT для вызова текстового редактора ЛЕКСИКОН с автоматической загрузкой в него последнего из ранее использованных для этой же цели текстовых файлов:

```
@ECHO OFF
IF /%1==/ GOTO NOARG
SET #DOC=%1
:NOARG
LEXE %#DOC%
CLS
```

Здесь предполагается, что файл LEX.EXE переименован в LEXE.EXE, чтобы не возникало коллизий с файлом LEX.BAT. Если в ответ на приглашение DOS ввести

```
LEX C:\TXT\MYFILE.TXT
```

то спецификация указанного файла будет запомнена в глобальной переменной #DOC, а сам файл — загружен в редактор. Значение #DOC будет затем использоваться при запуске файла LEXE.EXE командной строкой LEX без аргумента, обеспечивая загрузку в редактор последнего из заданных файлов.

7. Командный файл BLANK.BAT для гашения экрана на период до нажатия любой клавиши:

```
PROMPT $e[0;30m$e[2J
PAUSE
PROMPT $e[0m
CLS
@ECHO OFF
PROMPT $P$G
```

Для того чтобы понять работу этого командного файла, нужно ознакомиться с материалом, изложенным в п. 5.11.1. Файл BLANK.BAT гасит экран дисплея и оставляет его в таком состоянии до тех пор, пока не будет нажата какая-либо клавиша. Этот файл полезно использовать при непродолжительных перерывах в работе, чтобы, не выключая ПЭВМ, погасить экран и тем самым продлить срок его службы. Часто же выключать и включать ПЭВМ не рекомендуется.

Первой строкой файл BLANK.BAT посылает Escape-последовательность ←[0;30m←[2J на экран, а драйвер ANSI.SYS перехватывает и интерпретирует ее. Нуль задает обычные атрибуты монохромного экрана (фон — черный); 30m устанавливает черный цвет для текста, чтобы последующие сообщения были невидимыми, а символы 2J обеспечивают очитку экрана. Команда PAUSE приостанавливает интерпретацию командного файла. После нажатия произвольной клавиши выполнится вторая команда PROMPT, и экран вернется к своему нормальному состоянию. Третья же команда PROMPT установит удобную подсказку DOS. Команду @ECHO OFF нельзя выдать в начале командного файла, так как в этом случае Escape-последовательности не будут посланы на экран и поэтому не будут перехвачены драйвером ANSI.SYS. Команда CLS необходима (по крайней мере, в PC DOS 3.3 и MS-DOS 4.0) для «срабатывания» предшествующей ей PROMPT.

8. Для автоматического ответа на запрос команды DOS или программы в командном файле можно использовать следующую схему:

```
...
ECHO Y >YES
process <YES
DEL YES
...
```

Команда ECHO создает текстовый файл YES с ответом Y, затем содержимое файла YES подается на стандартный ввод процесса, чем имитируется ввод ответа Y с клавиатуры. После завершения процесса файл YES удаляется. Естественно, аналогичным образом можно задать любые требуемые ответы.

Недостаток описанного способа состоит в необходимости создания и удаления файла, что требует много времени. Устранить этот недостаток можно путем создания постоянных текстовых файлов с различными вариантами ответов и их использования во всех командных файлах.

Еще один, по всей видимости наилучший, способ автоматического ввода ответа в процесс состоит в использовании строки вида

```
ECHO Y|process
```

9. Если Вы хотите иметь команды для сохранения рабочего каталога в определенный момент и его восстановления после установки нового рабочего каталога, то создайте, например, в каталоге C:\BAT следующие файлы:

```
CDFILE:      CD

SAVEDIR.BAT: @ECHO OFF
             CD >C:\BAT\CURDIR
             COPY C:\BAT\CDFILE+C:\BAT\CURDIR C:\BAT\RETURN.BAT
```

Текстовый файл CDFILE содержит «начало» команды CD. Его нужно сформировать аккуратно, по крайней мере, с одним пробелом после CD и без маркера EOL. Командный файл SAVEDIR.BAT обеспечивает запись спецификации рабочего каталога в текстовый файл CURDIR и формирование командного файла RETURN.BAT для восстановления этого каталога.

Для удобства работы одним из маршрутов поиска исполняемых файлов должен быть C:\BAT.

Пусть рабочим является каталог C:\EDITORS\TEXT\LEX. Если вызвать командный файл SAVEDIR (без аргументов), то будет сформирован командный файл RETURN, содержащий команду

```
CD C:\EDITORS\TEXT\LEX
```

После этого можно менять рабочие каталоги командой CD (но не командой *d*:!). Для возврата в LEX достаточно будет ввести с клавиатуры RETURN.

Каждое последующее выполнение файла SAVEDIR отменяет действие предыдущего без возможности восстановления.

Описанная техника облегчает работу в случае наличия на жестком диске разветвленной файловой структуры.

10. Командный файл FULLCOPY.BAT для резервирования содержимого жесткого диска на дискетах в приводе A:

```
@ECHO OFF
ATTRIB +A C:\*.* /S
:LOOP
ECHO Установите отформатированную дискету в привод A
ECHO и затем нажмите любую клавишу
PAUSE >NUL
XCOPY C:\*.* A: /S /M /E /V
IF ERRORLEVEL 4 GOTO LOOP
ECHO Резервирование завершено
```

Команда ATTRIB устанавливает атрибут A у всех файлов. XCOPY копирует на дискету в приводе A файловую структуру с жесткого диска. У всех скопированных файлов атрибут A сбрасывается. Выполнение правильно заданной команды XCOPY (а в данном случае это так) завершается в одном из следующих случаев:

1) дискета заполнена, но не все файлы еще зарезервированы (код возврата 4);

2) все файлы зарезервированы, файлов для копирования не найдено или выполнение команды прервано по Ctrl-Break (коды возврата 0, 1 и 2 соответственно);

3) произошла критическая ошибка ввода-вывода (код возврата 5).

В случаях 1 и 3 команда IF передает управление на метку LOOP для продолжения (возобновления) резервирования на следующей дискете. При этом уже скопированные файлы не резервируются, так как их атрибуты A сброшены.

Если произошел случай 2, то резервирование завершается с выдачей соответствующего сообщения.

Конечно, можно ввести дополнительный анализ кода возврата 5.

В MS-DOS 4.0 рассмотренный командный файл не используйте, так как в команде XCOPY имеется ошибка (при нехватке места на целевом диске выдается код возврата 0).

11. Резервирование всего содержимого жесткого диска требует много времени и большого количества дискет. Поэтому при повторном резервировании лучше в команде XCOPY использовать переключатель /D для копирования только тех файлов, которые созданы и обновлены после даты последнего резервирования. При этом для корректного выбора файлов требуется обеспечить работу DOS все время с правильной датой (см. описание команды DATE в п. 5.6.5).

Кроме того, пользователю нужно запоминать или записывать дату последнего резервирования файлов для указания ее в переключателе /D. Чтобы избавиться от этого, можно создать следующие командные файлы:

```
DATECOPY.BAT: @ECHO OFF
              CALL BACKDATE
              ECHO|MORE|DATE|FIND /V "Enter" >BACKDATE.BAT

CURRENT.BAT:  @ECHO OFF
              ATTRIB +A C:\*.* /S
              :LOOP
              ECHO Установите отформатированную
              ECHO дискету в привод A
              PAUSE >NUL
              XCOPY C:\*.* A: /S /M /E /V /D:%4
              IF ERRORLEVEL 4 GOTO LOOP
              ECHO Резервирование закончено
```

Для того чтобы использовать эти файлы, нужно единственный раз сформировать файл BACKDATE.BAT с помощью команды

```
ECHO|MORE|DATE|FIND /V "Enter" >BACKDATE.BAT
```

Этот файл будет содержать только строку, аналогичную следующей:

```
Current date is Sat 1-26-1991
(Текущая дата — суббота 26.01.1991 г.)
```

Действительно, команда MORE при отсутствии подлежащего отображению текста (от команды ECHO) генерирует маркер EOL и тем самым моделирует нажатие клавиши Enter в ответ на запрос текущей даты командой DATE. Команда же FIND отсекает из вывода команды DATE вторую строку (Enter new date (dd-mm-yy):).

Можно создать файл BACKDATE.BAT и любыми другими средствами, но его содержимое обязательно должно иметь указанный вид.

Для резервирования новых файлов (созданных в день последнего резервирования или позже) теперь достаточно инициировать выполнение командного файла DATECOPY. Он вызывает файл BACKDATE.BAT, который интерпретируется как командный. При этом запускается командный файл CURRENT.BAT с четырьмя аргументами (date, is, день недели и дата). Четвертый аргумент (дата) замещает параметр %4 в переключателе /D команды XCOPY. После завершения резервирования управление возвращается в файл DATECOPY.BAT и выполняется команда DATE, обновляющая файл BACKDATE.BAT текущей датой, т.е. датой последнего резервирования. Поэтому после запуска DATECOPY через некоторый период времени резервирование пройдет в соответствии с ожиданиями (будут скопированы все файлы с учетом даты в BACKDATE.BAT).

В MS-DOS 4.0 данные командные файлы не используйте по той же причине, что и командный файл из предыдущего примера.

5.10. Конфигурирование системы

Под **конфигурированием** системы понимается настройка системы для достижения поставленных целей.

Рассматриваемые здесь средства DOS обеспечивают:

— настройку системы на конкретную конфигурацию оборудования ПЭВМ;

— адаптацию системы к потребностям пользователя;

— повышение эффективности функционирования DOS и производительности ПЭВМ в целом.

Вначале мы приведем общие сведения о конфигурировании системы, затем опишем команды конфигурирования, а в заключение — внешние системные драйверы устройств.

5.10.1. Общие положения

Конфигурирование DOS осуществляется MP BIOS во время загрузки системы по указаниям, заданным в виде специальных команд в файле CONFIG.SYS. В процессе конфигурирования реализуются следующие действия:

— устанавливаются режимы работы DOS;

— подключаются внешние драйверы устройств;

— осуществляется параметризация DOS с выделением необходимых областей памяти (для буферов ввода-вывода и т.п.);

— производится выбор и, возможно, настройка КП;

— осуществляется настройка на указанную страну;

— загружаются резидентные программы (только для DOS 4.0).

Если файл CONFIG.SYS отсутствует, то при конфигурировании используются имеющиеся умолчания. В случае применения для конфигурирования DOS файла CONFIG.SYS он должен находиться в корневом каталоге системного диска.

Этот файл представляет собой обычный ASCII-файл и может быть создан традиционными средствами. Каждая его строка должна содержать команду конфигурирования системы вида (за исключением команды REM)

имя команды = *аргументы* [*переключатели*]

Символ = может отделяться слева и справа любым количеством пробелов.

Все команды конфигурирования системы независимы и поэтому в смысле синтаксиса могут указываться в файле CONFIG.SYS в любом порядке. Однако если учитывать семантику команд, а также их последовательную обработку в ходе загрузки системы, то порядок задания команд в файле CONGIG.SYS (особенно команд DEVICE=) должен быть предварительно продуман.

Если Вы изменили содержимое файла CONFIG.SYS и хотите сконфигурировать DOS в соответствии с новыми командами, то следует перезагрузить систему.

Подавляющее число функций конфигурирования системы не может быть выполнено другими средствами DOS.

Вместе с тем дополнительное конфигурирование и реконфигурирование системы можно осуществить уже изученными нами командами реконфигурирования (см. п. 5.6.5).

5.10.2. Команды конфигурирования системы

Существуют следующие команды конфигурирования системы (для файла CONFIG.SYS):

BREAK= — устанавливает режим контроля нажатия комбинации клавиш Ctrl-Break;

BUFFERS= — устанавливает число буферов для дискового ввода-вывода;

COUNTRY= — настраивает DOS на использование в заданной стране;

DEVICE= — подключает внешние драйверы устройств;

DRIVPARM= — параметризует драйверы поблочных устройств;

FCBS= — устанавливает максимальное число одновременно открытых FCB (при разделении файлов);

FILES= — устанавливает максимальное число одновременно открытых файлов при использовании определенных системных вызовов;

INSTALL= — загружает и выполняет резидентные программы;

LASTDRIVE= — задает список допустимых имен логических приводов для команды SUBST;

REM — комментирует файл CONFIG.SYS;

SHELL= — обеспечивает настройку и замену стандартного КП DOS на другой;

STACKS= — устанавливает число и размер стеков для аппаратных прерываний;

SWITCHAR= — устанавливает символ переключателя для командной строки;

SWITCHES= — обеспечивает эмуляцию 84-клавишной клавиатуры.

Мы используем символ =, чтобы отличать команды конфигурирования от других команд DOS.

Все команды конфигурирования, за исключением BREAK=, DEVICE=, DRIVPARM=, INSTALL= и REM, могут указываться в файле CONFIG.SYS только один раз.

Команда BREAK=

Назначение: установка режима контроля нажатия комбинации клавиш Ctrl-Break.

Синтаксис:

BREAK={ON|OFF}

Комментарии. При использовании данной команды достигается тот же эффект, что и путем ввода команды BREAK (см. п. *5.6.5*) с клавиатуры или ее выдачи из командного файла, в частности, AUTOEXEC.BAT. Единственное отличие состоит в том, что команда BREAK= не позволяет отобразить установленный режим.

По умолчанию (когда команда BREAK= не задана) предполагается OFF.

Пример:

- BREAK=ON — включить контроль нажатия комбинации клавиш Ctrl-Break (Ctrl-C).

Команда SWITCHAR=

Назначение: установка символа переключателя для командной строки.

Синтаксис:

SWITCHAR=*символ*

Комментарии. Эта команда не документирована (не описана в документации), но реально поддерживается. С ее помощью задается одиночный *символ*, который будет в последующем использоваться при указании переключателей в командных строках. По умолчанию принимается символ /.

Замечание: изменить признак переключателя можно, например, с целью использования символа / в именах файлов. Однако полезность данной команды сомнительна, так как файловые структуры и командные файлы оказываются непереносимыми на другие ПЭВМ.

Пример:

- SWITCHAR=— — установить символ — вместо / для индикации переключателя в командной строке.

Команда DEVICE=

Назначение: подключение к DOS внешних драйверов устройств.

Синтаксис:

DEVICE=*file* [*arglist*] [*переключатели*]

Комментарии. Спецификация *file* указывает на файл, содержащий двоичный образ подключаемого драйвера, т.е. машинную программу. Драйверы оформляются специальным образом и обычно имеют расширение SYS или BIN. С DOS поставляется совокупность внешних системных драйверов, каждый из которых может быть подключен командой DEVICE=. Они описываются в следующем подразделе.

arglist — это последовательность (список) аргументов, обеспечивающих конфигурирование драйвера.

Переключатели играют ту же роль.

Если Вы приобрели дополнительное ПУ (например, сканер или «мышь»), то с ним же обычно должны получить и драйвер для управления устройством. Если этот драйвер имеет расширение SYS или BIN, то подключите его командой DEVICE=, а если COM, — укажите его в файле AUTOEXEC.BAT. В результате этого появится возможность использовать ПУ аналогично стандартным устройствам, поддерживаемым DOS, путем обращения к нему по имени.

По своему усмотрению Вы можете подключить внешние драйверы и для уже зарегистрированных в DOS устройств.

Замечания:

— умолчания для команды DEVICE= отсутствуют;

— внешние системные драйверы COUNTRY.SYS и KEYBOARD.SYS загружаются DOS автоматически (при использовании соответствующих команд DOS). Если специфицировать их командами DEVICE=, то DOS во время загрузки просто зависнет;

— команда DEVICE= увеличивает размер резидентной части DOS.

Пример: DEVICE=D:\DOS33\ANSI.SYS

Команда BUFFERS=

Назначение: установка числа буферов для дискового ввода-вывода, которые будут созданы и задействованы DOS.

Синтаксис:

BUFFERS = *n*

Комментарии. Аргумент *n* задает число буферов, которые требуется создать. Допустимы значения в диапазоне 1 — 99. По умолчанию (если команда BUFFERS= отсутствует) устанавливается одно из следующих значений:

для ПЭВМ с минимальными аппаратными ресурсами	— 2
при наличии дисковода емкостью более 360 Кбайт	— 3
при наличии ОЗУ емкостью 128 — 255 Кбайт	— 5
при наличии ОЗУ емкостью 256 — 511 Кбайт	— 10
при наличии ОЗУ емкостью не менее 512 Кбайт	— 15

Буфер — это область ОЗУ, которую DOS использует для временного хранения информации при обмене с дисками. В каждом буфере запоминается 512-Кбайт дисковый сектор.

Когда программа запрашивает ввод информации с диска (даже одного байта), DOS считывает соответствующий 512-байт сектор в буфер и затем передает программе требуемую порцию информации (логическую запись). Если следующий запрос выполнен для этого же сектора, то повторное считывание с диска не производится, а просто в программу из буфера передается очередная запись. Если же требуемых данных в буфере нет, то осуществляется считывание в буфер следующего сектора. Вывод информации на диск производится также через буфер. Описанная техника позволяет ускорить обмен информацией за счет физической передачи достаточно больших порций информации, что уменьшает долю накладных расходов.

При увеличении числа буферов повышается и вероятность того, что требуемые данные в момент запроса окажутся в одном из них (в частности, при одновременной работе с несколькими файлами). Это справедливо потому, что обмен информацией между буферами и диском осуществляется асинхронно и параллельно выполнению программы (обменом управляет не МП, а СПДП). Накладные же расходы по пересылке данных в памяти невелики. Следовательно, увеличение числа буферов ведет к повышению производительности ПЭВМ.

Однако чрезмерно большое количество буферов дает обратный эффект: увеличиваются накладные расходы по анализу статуса буферов, а само по себе такое число буферов избыточно и к выигрышу собственно при обмене информацией уже не ведет. Замедление обмена наблюдается при формировании около 50 буферов.

Для обычных программ, таких, как текстовые редакторы, оптимальное значение *n* принадлежит диапазону 10 — 20. При наличии на жестком диске разветвленной файловой структуры лучше выбрать *n* из диапазона 20 — 30.

Для различных ПЭВМ целесообразно устанавливать следующие значения *n*:

IBM PS/2 (модели 50 — 80)	— 32
IBM PC AT с 20 — 30-Мбайт НЖМД	— 32
IBM PC XT с 10-Мбайт НЖМД	— 16
IBM PC (без НЖМД)	— 4

Замечания:

— правильная установка *n* обеспечивает повышение скорости обмена информацией примерно на 10%;

— наилучшее значение *n* для заданной конфигурации оборудования и определенного класса задач можно определить только опытным путем;

— увеличение числа буферов ведет к увеличению резидентной части DOS (каждый буфер требует 528 байт памяти);

— пользователь должен, по крайней мере, обеспечить такое количество буферов, которого достаточно для хранения всей FAT (таблицы размещения файлов), так как она читается часто.

Пример: BUFFERS = 25

DOS 4.0. Команда задается в виде

BUFFERS = *n*[, *m*] [/X]

Здесь *n* — имеет тот же смысл;

m — максимальное число секторов, которые могут быть считаны или записаны за одну операцию обмена. Допустимы значения от 1 до 8, а по умолчанию принимается 1. Если m > 1, то считывание будет осуществляться с упреждением, что может дополнительно повысить скорость обмена;

/X — предписывает разместить буфера в отображаемой памяти (если она полностью занята или отсутствует, то данный переключатель игнорируется).

Когда /X задан, максимально допустимым значением *n* является 10000.

Система PC DOS 4.0 содержит ошибку, в результате которой созданные в отображаемой памяти буфера портятся, если эта память интенсивно используется другими программами. Поэтому для PC DOS переключатель /X лучше не задавать. В системе же MS-DOS 4.0 описанная ситуация обычно не возникает.

Команда FCBS=

Назначение: установка максимального числа одновременно открытых FCB, когда включены в работу средства разделения файлов.

Синтаксис:

FCBS = *n*, *m*

Комментарии. Аргумент *n* задает максимальное число FCB, которые могут находиться в открытом состоянии (а следовательно, и максимальное число одновременно открытых файлов). Аргумент *m* определяет, сколько первых из открытых FCB (по времени открытия) требуется защитить от автоматического закрытия, когда производится попытка открытия новых FCB, а уже открыто *n* FCB. Числа *n* и *m* не должны превышать 255. По умолчанию подразумевается команда FCBS = 4,0.

Программный интерфейс DOS предлагает две группы средств для организации обмена с файлами:

1) *средства, ориентированные на FCB*;

2) *средства, ориентированные на обработчик* (handle).

Первые из них были введены в ранних версиях DOS и сохранены в последующих для обеспечения совместимости снизу вверх. Они используются старыми программными продуктами.

Вторая группа средств является новой, более совершенной и доступна только в последних версиях DOS.

Команда FCBS= имеет отношение к средствам, ориентированным на FCB. FCB — это информационная структура, которая используется для управления открытыми файлами. Если *m* установлен в нуль, то при открытии каждого FCB, когда уже открыто *n* FCB, происходит закрытие первого (самого старого) FCB. Если *m* не равен нулю, то в данной ситуации первые *m* файлов остаются открытыми, а закрывается (*m* + 1)-й FCB.

Замечания:

— команду FCBS= следует применять только совместно с командой SHARE (см. п. 5.6.3) при условии, что выполняемые программы используют средства, ориентированные на FCB;

— число *m* не должно превосходить *n*. В предельном случае (когда *m* = *n*) DOS никогда не сможет самостоятельно (автоматически) закрыть FCB, вследствие чего могут возникнуть ситуации, при которых нельзя будет открыть новый файл;

— увеличение *n* ведет к возрастанию размера резидентной части DOS.

Пример: FCBS = 4,2

Команда FILES=

Назначение: установка максимального числа одновременно открытых файлов при пользовании средствами обмена, ориентированными на обработчик.

Синтаксис:

FILES = *n*

Комментарии. Единственный аргумент *n* определяет количество файлов, которые одновременно могут находиться в открытом состоянии. Значение *n* должно лежать в диапазоне 5 — 99 (по умолчанию принимается 8).

Замечания:

— многие программы требуют, чтобы *n* было не менее 20;

— используйте команду FILES=, если на экране появится сообщение об исчерпании обработчиков файлов (file handles);

— никогда не устанавливайте *n* меньше 6, иначе у Вас возникнут проблемы даже при выполнении команд DOS (например, FORMAT);

— без перезагрузки DOS увеличить число обработчиков файлов можно путем вызова через программный интерфейс функции 67H по прерыванию 21H;

— увеличение *n* ведет к возрастанию размера резидентной части DOS.

Пример: FILES = 20

DOS 4.0. Значение *n* должно принадлежать диапазону 8 — 255 (умолчание — 8). Тем не менее через программный интерфейс можно установить значение вплоть до 65534.

Команда DRIVPARM=

Назначение: параметризация драйверов поблочных ПУ (обычно — внутренних драйверов для поддержки нестандартных устройств).

Синтаксис:

DRIVPARM = /D: *n* [/C] [/I] [/N] [/H: *h*] [/S: *s*] [/T: *t*] [/F: *f*]

Комментарии. Данная команда имеется и в DOS 3.3, но документирована только в DOS 4.0. Она модифицирует драйвер поблочного устройства путем изменения используемых им параметров ПУ. В результате этого обеспечивается поддержка драйвером другого формата носителя информации на устройстве, если оно исходя из своих аппаратных возможностей способно на это. Обычно командой DRIVPARM= параметризуют внутренние драйверы устройств, находящиеся в ПЗУ (табличная информация, содержащая параметры для них, переписывается в ОЗУ при загрузке DOS).

В команде DRIVPARM= указывается не драйвер, который требуется параметризовать, а конкретное устройство. Поэтому если драйвер обслуживает несколько устройств, то он параметризуется только в плане управления заданным (единственным) устройством.

В команде указываются следующие переключатели (все из них, кроме /D, являются факультативными):

/D: *n* — задает физический номер привода *n* (от 0 до 255). Обычно 0 соответствует A, 1 — B, 2 — C и т.д.;

/C — информирует DOS о том, что дисковод аппаратно способен распознавать ситуацию, когда диск установлен, но дверца открыта. Такой дисковод реагирует на открытую дверцу так, как будто бы диск не был установлен вовсе. Узнать о возможностях дисковода можно, изучив документацию на него. Современные дисководы, как правило, распознают описанную ситуацию;

/I — информирует DOS о том, что указанный 89-мм дисковод совместим на аппаратном уровне с оборудованием ПЭВМ. Управление таким дисководом может быть осуществлено стандартным контроллером НГМД. Переключатель /I используется в случае, когда BIOS не поддерживает 89-мм дисководы, позволяя обеспечить доступ к ним;

/N — информирует DOS о том, что в команде специфицировано устройство с несъемным носителем информации (например, НЖМД);

/H: *h* — задает число головок *h* (от 1 до 99). По умолчанию принимается 2;

/S: *s* — устанавливает число секторов на дорожке *s* (от 1 до 99). Значением по умолчанию является 9, если не задан переключатель /F;

/T: *t* — задает число дорожек *t* на каждой стороне носителя информации (от 1 до 999). Значение по умолчанию отсутствует;

/F: *f* — специфицирует формат носителя, где *f* — одна из следующих цифр:

0 — для дискеты емкостью 160, 180, 320 или 360 Кбайт;
1 — для дискеты емкостью 1,2 Мбайт;
2 — для дискеты емкостью 720 Кбайт (89 мм);
5 — для НЖМД;
6 — для НМЛ;
7 — для дискеты емкостью 1,44 Мбайт (89 мм).

По умолчанию принимается 2.

Замечания:

— команда DRIVPARM= не подключает новый драйвер устройства, а только параметризует уже имеющийся в системе (возможно, внешний, установленный ранее командой DEVICE=);

— по умолчанию (если DRIVPARM= не задана) никакая параметризация не выполняется;

— переключатели /C, /I и /N необходимы потому, что аппаратные возможности устройств системой автоматически не распознаются;

— переключатель /F зачастую исключает необходимость детализации формата носителя информации переключателями /H, /S и /T;

— переключатели /H, /S, /T и /F аналогичны одноименным переключателям команды FORMAT;

— команда DRIVPARM= обеспечивает использование установленного формата носителя всеми средствами DOS;

— команда DRIVPARM= отменяет заданный по умолчанию формат носителя на указанном устройстве.

Пример:

- если ПЭВМ снабжена НМЛ, который имеет имя D и обслуживается драйвером, поддерживающим 20 дорожек по 40 секторов, а требуется установить формат «10 дорожек по 99 секторов», то укажите в файле CONFIG.SYS команду

DRIVPARM=/D:3 /H:1 /S:99 /T:10 /F:6

Команда LASTDRIVE=

Назначение: задание списка допустимых имен логических приводов для команды SUBST.
Синтаксис:

LASTDRIVE=*d*

Комментарии. Данная команда указывает, что в качестве имен логических приводов можно использовать буквы от A до *d* включительно. По умолчанию (если команда отсутствует) устанавливается список A — E. DOS для каждого имени логического привода выделяет область ОЗУ размером 80 байт. При наличии реального привода соответствующая область хранит текущий каталог диска, в него установленного. Для привода, созданного командой SUBST, соответствующая область содержит полную спецификацию каталога, который этим приводом представляется.

Замечания:

— увеличение длины списка имен логических приводов ведет к возрастанию размера резидентной части DOS;

— команда LASTDRIVE= полезна и в случае, когда ПЭВМ подключена к сети, чтобы обеспечить доступ к сетевым приводам;

— драйвер DRIVER.SYS выделяет 80-байт области самостоятельно и поэтому не требует задания команды LASTDRIVE=;

— для реальных логических приводов 80-байт области выделяются автоматически без использования сведений из команды LASTDRIVE=.

Пример:

- LASTDRIVE=M — обеспечить возможность работы с именами логических приводов от A до M.

Команда STACKS=

Назначение: установка числа и размера стеков для аппаратных прерываний.
Синтаксис:

STACKS=*n*,*s*

Комментарии. По данной команде DOS выделяет необходимую область памяти и формирует в ней пул из *n* (от 0 до 64) стеков размером в *s* (от 0 до 512) байт. По умолчанию (при отсутствии команды STACKS=) для IBM PC XT и IBM PC Portable принимается STACKS=0,0, а для других ПЭВМ — STACKS=9,128. Эти стеки обеспечивают решение проблем, связанных с обработкой повторных (когда прерывание происходит до окончания обработки предыдущего прерывания того же типа) и одновременно возникающих аппаратных прерываний. При каждом аппаратном прерывании DOS выделяет для него стек из имеющегося пула и обеспечивает доступ обработчика прерывания к стеку. После обработки прерывания ставший больше ненужным стек возвращается в пул. Если STACKS=0,0, то DOS не переключает стеки при возникновении прерываний, а использует единственный.

Замечание: увеличение размера пула стеков ведет к возрастанию размера резидентной части DOS.

Пример:

- STACKS=8,512 — сформировать пул из 8 стеков размером 512 байт каждый.

Команда SHELL=

Назначение: обеспечение настройки и замены стандартного КП DOS (COMMAND.COM) на новый.
Синтаксис:

SHELL=*file* [*arglist*] [*переключатели*]

Комментарии. Данная команда включается в CONFIG.SYS в случаях, когда требуется:

1) использовать вместо стандартного КП DOS другую программу;

2) установить размер окружения DOS для стандартного КП, отличающийся от принимаемого по умолчанию (т.е. параметризовать КП DOS);

3) загрузить COMMAND.COM не из корневого каталога системного диска, а из другого каталога.

4) изменить режим работы КП DOS.

Спецификация *file* задает файл, содержащаяся в котором программа будет использоваться в качестве КП. Аргументы, указанные в списке *arglist*, и переключатели параметризуют эту программу, а также задают режимы ее работы.

Если команда SHELL= не задана, то используется COMMAND.COM из корневого каталога системного диска.

DOS при обработке команды SHELL= автоматически записывает в окружение строку COMSPEC=*file*.

Замечания:

— если требуется использовать COMMAND.COM, находящийся в корневом каталоге системного диска, и изменять размер окружения, а также режим работы КП не нужно, то команду SHELL=\COMMAND.COM можно не указывать (значение глобальной переменной COMSPEC установится автоматически);

— если Вы решили специфицировать COMMAND.COM в команде SHELL=, то обратитесь к описанию команды COMMAND в п. 5.6.6 для правильного выбора переключателей с целью установки резидентного режима работы КП (переключатель /P обязателен!).

Примеры:

- SHELL=C:\DOS33\COMMAND.COM /E:1000 /P — запустить КП DOS из каталога C:\DOS33 и выделить под окружение 16-Кбайт область ОЗУ;
- SHELL=C:\BIN\NEWSHELL — в качестве КП использовать программу в указанном файле.

Команда COUNTRY=

Назначение: настройка DOS для использования в заданной стране.
Синтаксис:

COUNTRY=*c*[, [*cp*][, *file*]]

Комментарии. Данная команда служит для установки формата даты и времени, принятого в заданной стране.

В качестве аргументов указываются:

c — код страны;

cp — номер кодовой страницы, допустимой для заданной страны;

file — спецификация файла-драйвера, обеспечивающего работу DOS в соответствии с соглашениями, принятыми в заданной стране.

Драйвер, специфицированный посредством *file*, автоматически подключается к системе. Если аргумент *file* не задан, то выбирается внешний системный драйвер COUNTRY.SYS из корневого каталога системного диска. Этот драйвер поддерживает все используемые в DOS коды стран. Поэтому аргумент *file* нужно задавать только в случаях, когда используется другой драйвер или файл COUNTRY.SYS находится в другом каталоге.

Номер кодовой страницы можно не задавать.

Если команда COUNTRY не указана, то устанавливается код страны 001 (США).

Замечания:

— полная информация по настройке DOS на ту или иную страну содержится в описании команды MODE: поддержка кодовых страниц (см. п. 5.6.4);

— номер кодовой страницы в команде COUNTRY= обычно не указывается;

— COUNTRY= влияет на задание команд DATE, BACKUP, RESTORE и TIME, а также на формат отображения даты и времени этими и другими командами DOS.

Пример:

- COUNTRY=033,,C:\DOS33\COUNTRY.SYS — настроить DOS на использование во Франции (033) и указать на необходимость подключения драйвера из файла C:\DOS33\COUNTRY.SYS.

Команда INSTALL=

Назначение: загрузка и выполнение резидентной программы во время обработки файла CONFIG.SYS.
Синтаксис:

INSTALL=*file* [*arglist*] [*переключатели*]

Комментарии. Данная команда поддерживается только в DOS версии 4.0 и выше. Спецификация *file* задает файл с подлежащей выполнению резидентной программой. Аргументы *(arglist)* и переключатели относятся к устанавливаемой резидентной программе.

Замечания:

— по умолчанию никакая резидентная программа не выполняется;

— команда INSTALL= используется главным образом для того, чтобы предотвратить многократную установку резидентных программ при повторных запусках файла AUTOEXEC.BAT в ходе частичного рестарта системы;

— при установке резидентной программы через INSTALL= имеется возможность «сделать невидимыми» некоторые векторы прерываний до загрузки КП DOS;

— использование команды INSTALL= для установки резидентных программ вместо их явного запуска (в частности, из файла AUTOEXEC.BAT) позволяет несколько уменьшить размер резидентных модулей за счет сокращения размера копий окружения DOS, а также скрыть от пользователя-непрофессионала факт запуска на выполнение ряда системных программ;

— установка всех резидентных программ командой INSTALL= обеспечивает возможность динамического изменения размера окружения DOS в зависимости от потребностей, так как оно ничем теперь снизу не ограничивается;

— путем указания в INSTALL= могут быть выполнены внешние команды DOS FASTOPEN, KEYB, NLSFUNC и SHARE, так как они загружают в ОЗУ резидентный код;

— расширение (EXE или COM) в аргументе *file* необходимо обязательно указывать.

Пример:

- INSTALL=C:\DOS33\FASTOPEN.EXE C:=100 — выполнить команду FASTOPEN;

- INSTALL=C:\SYS\UNISCR.COM — загрузить в знакогенератор видеосистемы кодовую таблицу с кириллицей;
- INSTALL=C:\SYS\UNIKBD.COM /IB4/IS/M:C:\SYS\UNIKBD.STD/R — установить драйвер русской клавиатуры.

Команда SWITCHES=

Назначение: обеспечение эмуляции старой (84-клавишной) клавиатуры.
Синтаксис:

SWITCHES=/K

Комментарии. Эта команда имеется только в PC DOS 4.0 и используется в случаях, когда Вы работаете с программами, не реагирующими на нажатие дополнительных клавиш управления курсором на 101-клавишной клавиатуре.

Замечание: большинство программ работает со 101-клавишной клавиатурой корректно, а поэтому необходимость указания данной команды в файле CONFIG.SYS возникает редко.

Команда REM

Назначение: комментирование файла CONFIG.SYS.
Синтаксис:

REM *комментарий*

Комментарии. Команда REM доступна только в DOS 4.0 и полностью эквивалентна одноименной команде DOS для командных файлов (см. п. 5.9.4).

5.10.3. Внешние системные драйверы устройств

В комплекте DOS содержатся следующие внешние драйверы, каждый из которых может быть подключен к системе командой конфигурирования DEVICE=:

ANSI.SYS — обеспечивает возможности расширенного управления устройством CON (клавиатурой и дисплеем);

DISPLAY.SYS — поддерживает переключение кодовых страниц для дисплея;

DRIVER.SYS — создает новый логический дисковод;

EMM386.SYS — эмулирует отображаемую память в расширенной памяти ПЭВМ на базе МП 80386 или 80486;

HIMEM.SYS — управляет расширенной памятью в соответствии со спецификацией XMS 2.0 и HMA-памятью. Имеется в DOS, только начиная с версии 4.0;

PRINTER.SYS — поддерживает переключение кодовых страниц для принтера;

VDISK.SYS — создает виртуальные диски (в MS-DOS, имеет имя RAMDRIVE.SYS);

SMARTDRV.SYS — создает в дополнительной памяти кэш по чтению для обмена информацией с НЖМД (имеется в DOS, только начиная с версии 4.0). В PC DOS именуется как IBMCACHE.SYS;

XMA2EMS.SYS — управляет отображаемой памятью в соответствии со спецификацией LIM EMS 4.0. Поддерживается DOS, только начиная с версии 4.0;

XMAEM.SYS — эмулирует плату отображаемой памяти в расширенной памяти ПЭВМ на базе МП 80386 или 80486.

Изучать перечисленные драйверы будем в соответствии со схемой, использованной нами при описании команд. В секции синтаксиса при этом будет приводиться формат команды DEVICE= для подключения рассматриваемого драйвера.

Драйвер ANSI.SYS

Назначение: обеспечение возможности расширенного управления устройством CON (клавиатурой и дисплеем) через пользовательский интерфейс DOS.
Синтаксис:

DEVICE=[*dir*\]ANSI.SYS

Комментарии. Драйвер ANSI.SYS используется для:
— переключения текстовых режимов работы дисплея, а также установки цветов фона и выводимых символов;
— позиционирования курсора на экране дисплея;
— переопределения клавиш клавиатуры и их комбинаций.

Если спецификация *dir* не задана, то файл ANSI.SYS ищется в корневом каталоге системного диска.

Передача сообщений (команд) установленному драйверу ANSI.SYS осуществляется посредством Escape-последовательностей ANSI (Американского Национального института стандартов). Называются эти управляющие последовательности Escape-последовательностями потому, что каждая из них начинается символом ESC. Драйвер ANSI.SYS перехватывает весь вывод на дисплей с целью выделения таких последовательностей и выполнения действий, в них задаваемых. Escape-последовательности мы рассмотрим в следующем подразделе.

Пример:

- DEVICE=C:\DOS33\ANSI.SYS — подключить драйвер ANSI.SYS из каталога C:\DOS33.

DOS 4.0. Синтаксис:

DEVICE=[*dir*\]ANSI.SYS [/X] [/K] [/L]

Переключатели интерпретируются следующим образом:

/X — разрешить переопределение драйвером ANSI.SYS тех расширенных кодов клавиш, которые содержат в первом байте 0EH. Этот переключатель для стандартной американской клавиатуры не действует, так как все расширенные коды клавиш, ею генерируемые, начинаются с 00H;

/K — обеспечить эмуляцию старой (84-клавишной) клавиатуры (см. описание команды SWITCHES= в предыдущем пункте);

/L — обеспечить сохранение установленного командой MODE режима дисплея даже в том случае, когда выполняемая программа пытается его изменить.

Кроме того, в DOS 4.0 драйвером ANSI.SYS поддерживаются все текстовые режимы работы дисплейных адаптеров EGA и VGA.

Переключатель /L срабатывает не всегда, а также может приводить к «зависанию» системы.

Драйвер DISPLAY.SYS

Назначение: поддержка переключения кодовых страниц для дисплея.

Синтаксис:

DEVICE=[*dir*\]DISPLAY.SYS CON[:]=([*mun*][,[*hwcp*][,{*n*|(*n*,*m*)}]])

Комментарии. Команда DEVICE= в приведенном формате подключает драйвер DISPLAY.SYS, который активизирует заданную аппаратную кодовую страницу и выделяет указанное количество буферов для последующей загрузки составленных кодовых страниц.

Спецификация *dir* определяет каталог с файлом DISPLAY.SYS (по умолчанию — корневой каталог системного диска), а аргументы драйвера DISPLAY.SYS задают следующее:

mun — тип дисплейного адаптера (MONO, CGA, EGA или LCD). Если тип опущен, то осуществляется автоматическое распознавание адаптера. Тип EGA указывается как для адаптера EGA, так и для адаптера VGA;

hwcp — номер одной из поддерживаемых адаптером аппаратных кодовых страниц, которую требуется активизировать. Допустимы следующие номера, если соответствующие кодовые страницы поддерживаются адаптером аппаратно: 437, 850, 860, 863 и 865. Кодовая страница по умолчанию определяется оборудованием (обычно — 437);

n — количество буферов, которые требуется создать для последующей загрузки составленных кодовых страниц (по одному буферу на кодовую страницу). Допустимо значение в диапазоне 0 — 12 (для MONO и CGA нужно указать 0, для EGA — можно 2, а для LCD — 1);

m — число шрифтов, поддерживаемых для каждой кодовой страницы. Например, EGA поддерживает два шрифта (с матрицами 8x8 и 8x14 точек).

Замечания:

— полная информация о переключении кодовых страниц содержится в описании команды MODE: поддержка кодовых страниц (см. п. 5.6.4);

— если тип адаптера автоматически распознается неверно, то следует его явно задать в качестве аргумента;

— число шрифтов обычно не указывается.

Пример:

- DEVICE=C:\DOS33\DISPLAY.SYS CON=(EGA,437,2) — активизировать аппаратную кодовую страницу 437 (для США) и сформировать два буфера для составленных кодовых страниц.

Драйвер PRINTER.SYS

Назначение: поддержка переключения кодовых страниц для принтера.

Синтаксис:

DEVICE=[*dir*\]PRINTER.SYS LPT*n*=(*mun*[,[*hwcp* | (*hwcp*[, *hwcp*]...)][, *m*]])

Комментарии. Команда DEVICE= в данном формате подключает драйвер PRINTER.SYS, который активизирует заданную аппаратную кодовую страницу и выделяет указанное количество буферов для последующей загрузки составленных кодовых страниц.

Спецификация *dir* определяет каталог с фалом PRINTER.SYS (по умолчанию — корневой каталог системного диска), а аргументы драйвера PRINTER.SYS задают следующее:

n — номер (1, 2 или 3) адаптера параллельного интерфейса, к которому подключен принтер;

mun — тип принтера. Допустимы следующие типы:

4201 — семейство IBM 4201 Proprinter или IBM 4202 Proprinter XL;
5202 — IBM 5202 Quietwriter III;

hwcp — номер активизируемой аппаратной кодовой страницы. Если задано несколько номеров, то первая из указанных кодовых страниц активизируется, а остальные подготавливаются;

m — количество буферов, которые требуется создать для последующей загрузки составленных кодовых страниц (по одному буферу на кодовую страницу).

Замечания:

— полная информация о переключении кодовых страниц содержится в описании команды MODE: поддержка кодовых страниц (см. п. 5.6.4);

— если Ваш принтер несовместим ни с одним из перечисленных, то подключать драйвер PRINTER.SYS не следует;

— вместо LPT1 можно указать PRN;

— если указывается единственная аппаратная кодовая страница, то внутренние скобки можно опустить.

Пример:

- DEVICE=C:\DOS33\PRINTER.SYS PRN=(4201,437,2) — активизировать аппаратную кодовую страницу 437 (для США) и сформировать два буфера для составленных кодовых страниц.

DOS 4.0. Поддерживается тип 4208 для принтеров IBM 4207 или 4208, а также IBM Proprinter X24 или XL24.

Драйвер DRIVER.SYS

Назначение: создание нового логического привода.

Синтаксис:

DEVICE=[*dir*\]DRIVER.SYS /D:*n* [/C] [/H:*h*] [/S:*s*] [/T:*t*] [/F:*f*]

Комментарии. Данный драйвер используется для:

— поддержки внешних (подключенных к ПЭВМ дополнительно) НГМД;

— создания дополнительного логического (фиктивного) дисковода, эквивалентного по формату уже имеющемуся и обозначающего тот же физический привод;

— создания дополнительного логического (фиктивного) дисковода с другим форматом по сравнению с имеющимся, но обозначающего тот же физический привод.

Спецификация *dir* указывается, если файл DRIVER.SYS содержится в каталоге, отличном от корневого каталога системного диска.

Для драйвера специфицируются следующие переключатели:

/D:*n* — задает физический номер НГМД *n* (от 0 до 127), причем 0 обозначает первый НГМД, 1 — второй, 2 — третий (обычно первый внешний НГМД) и т.д. или физический номер НЖМД *n* (128 — первый НЖМД, а 129 — второй);

/C — информирует драйвер о том, что дисковод аппаратно способен распознавать ситуацию, когда диск установлен, но дверца не закрыта. Такой дисковод реагирует на открытую дверцу, как будто бы диск не был установлен. Узнать об особенностях дисковода можно, изучив документацию на него. Современные дисководы, как правило, распознают описанную ситуацию;

/H:*h* — задает число головок *h* (от 1 до 99, по умолчанию — 2);

/S:*s* — устанавливает число секторов на дорожке *s* (от 1 до 99, по умолчанию — 9);

/T:*t* — задает число дорожек на каждой стороне дискеты *t* (от 1 до 999, по умолчанию — 80);

/F:*f* — специфицирует формат дискеты, где *f* — номер формата, задаваемый одной из следующих цифр:

0 — для дискеты емкостью 160, 180, 320 или 360 Кбайт;
1 — для дискеты емкостью 1,2 Мбайт;

2 — для дискеты емкостью 720 Кбайт (89 мм);
7 — для дискеты емкостью 1,44 Мбайт (89 мм).

По умолчанию принимается 2.

Замечания:
— в отличие от команды DRIVPARM= драйвер DRIVER.SYS не модифицирует формат зарегистрированного логического дисковода, а создает новый логический дисковод на новом или уже зарегистрированном приводе;
— об именах, назначенных созданным логическим дисководам, пользователь будет оповещен при загрузке DOS;
— переключатели, воспринимаемые драйвером DRIVER.SYS, аналогичны тем, которые указываются в команде DRIVPARM=;
— если ПЭВМ содержит единственный НГМД, то подключать драйвер DRIVER.SYS для создания дополнительного логического привода, идентичного имеющемуся, не следует (он будет создан автоматически и получит имя B);
— переключатель /C необходим потому, что аппаратные возможности НГМД драйвером автоматически не распознаются;
— переключатель /F зачастую исключает необходимость детализации формата дискеты переключателями /H, /S и /T;
— драйвером DRIVER.SYS можно создать 720-Кбайт логические дисководы на ПЭВМ типа ЕС 1840/41, поварьировав значениями в переключателе /D для имитации внешних 720-Кбайт 89-мм приводов.

Примеры:

- если ПЭВМ снабжена одним 1,2-Мбайт и одним 1,44-Мбайт НГМД, а Вы хотите их использовать для копирования файлов (по COPY) с дискеты на дискету того же формата без задействования жесткого диска в качестве промежуточного хранилища информации, то поместите в CONFIG.SYS команды

```
DEVICE=DRIVER.SYS /D:0 /F:1
DEVICE=DRIVER.SYS /D:1 /F:7
```

  В результате этого каждый НГМД станет доступным под двумя именами, которые можно будет указать в команде COPY;

- если Вы подключили к ПЭВМ внешний 89-мм 720-Кбайт НГМД, то в CONFIG.SYS укажите

```
DEVICE=DRIVER.SYS /D:2
```

  Продублировав данную строку, получите на этом физическом дисководе два логических.

Драйвер VDISK.SYS

Назначение: создание виртуального диска.
Синтаксис:

DEVICE=[*dir*\]VDISK.SYS [*m*] [*n*] [*p*] [/E[:*q*]]

Комментарии. *Виртуальным диском (ОЗУ-диском)* является аналог жесткого диска, созданный в оперативной памяти. Он применяется для хранения часто используемых файлов, чтобы снизить время доступа к ним. Файлы на виртуальный диск удобно копировать командами, размещенными в AUTOEXEC.BAT. Недостаток виртуального диска состоит в том, что при выключении питания все его содержимое теряется.

Спецификация *dir* определяет каталог, содержащий файл VDISK.SYS, если этот файл не находится в корневом каталоге системного диска.

Аргументы задают следующее:

m — размер виртуального диска в Кбайт (от 16, по умолчанию — 64);

n — размер сектора в байтах. Допустимы значения 128, 256 и 512. По умолчанию принимается 128;

p — число элементов в корневом каталоге виртуального диска (от 2 до 512, по умолчанию — 64).

Если имеется переключатель /E, то виртуальный диск размещается в расширенной памяти; число *q* задает количество секторов, передаваемых между расширенной памятью и стандартным ОЗУ посредством одной операции чтения-записи (от 1 до 8, по умолчанию — 8). Если переключателя /E нет, то виртуальный диск создается в стандартном ОЗУ.

Замечания:
— виртуальный диск, созданный в стандартном ОЗУ, существенно увеличивает размер резидентной части DOS. Поэтому если расширенная память имеется, то целесообразно указывать переключатель /E;

— имя привода, под которым можно обращаться к виртуальному диску, будет сообщено во время загрузки DOS;

— форматировать виртуальный диск перед использованием не следует;

— для создания более одного виртуального диска нужно специфицировать несколько команд DEVICE=VDISK.SYS ...;

— не размещайте на виртуальном диске создаваемые файлы с ценной информацией;

— если выполняемая программа одновременно использует два файла, то разместите один из них на виртуальном диске, чтобы избавить физический дисковод от частого перемещения головок с одного файла на другой;

— количество элементов корневого каталога виртуального диска округляется в бóльшую сторону до числа, кратного количеству элементов в секторе (каждый элемент каталога занимает 32 байт). Так, если задано 25 элементов корневого каталога, то при 512-байт размере сектора реально будет создано 32 элемента. Действительно, каждый сектор содержит 16 элементов каталога, а 32 — ближайшее к 25 большее число, кратное 16. Описанные действия выполняются с целью наиболее полного использования памяти (сектор, отведенный под каталог, задействуется им полностью);

— если Вы хотите опустить ряд аргументов драйвера, то придерживайтесь следующего правила: i-й аргумент можно не указывать, только если не заданы все последующие аргументы;

— размер сектора в 512 байт обеспечивает наивысшее быстродействие виртуального диска;

— файл VDISK.SYS поставляется с PC DOS. Драйвер же виртуального диска системы MS-DOS содержится в файле RAMDRIVE.SYS;

— драйвер VDISK.SYS управляет расширенной памятью непосредственно, а поэтому никаких дополнительных драйверов не требуется.

Примеры:

- DEVICE=VDISK.SYS 30 — создать 30-Кбайт виртуальный диск в стандартном ОЗУ;
- DEVICE=VDISK.SYS 1500 512 /E — создать 1,5-Мбайт виртуальный диск с размером сектора в 512 байт в расширенной памяти.

DOS 4.0. Синтаксис:

DEVICE=[*dir*]VDISK.SYS [*m*] [*n*] [*p*] [/E[:*q*]] [/A[:*q*]]

Здесь аргументы имеют тот же смысл, однако *n* дополнительно может быть равно 1024, а по умолчанию принимается 512; *p* теперь может принадлежать диапазону 4 — 1024.

Переключатели /E и /A несовместимы. /E, как и ранее, обеспечивает создание виртуального диска в расширенной памяти, а переключатель /A — в отображаемой. Число *q* имеет тот же смысл. Используйте переключатель /A, только если расширенная память отсутствует или ее размера оказывается недостаточно (отображаемую память лучше задействовать в других целях, получив от этого преимущества, заключающиеся в отсутствии пересылки данных в стандартное ОЗУ и обратно). Если виртуальный диск все же создается в отображаемой памяти, то предварительно установите драйвер для управления ею, например XMA2EMS.SYS.

Драйвер SMARTDRV.SYS

Назначение: создание в дополнительной памяти ПЭВМ кэша по чтению для обмена информацией с НЖМД.

Синтаксис:

DEVICE=[*dir*\]SMARTDRV.SYS [*s*] [/A]

Комментарии. Драйвер SMARTDRV.SYS имеется только в DOS, начиная с версии 4.0. Его можно использовать лишь на такой ПЭВМ, которая снабжена НЖМД и имеет дополнительную (отображаемую или расширенную) память. Этот драйвер создает в дополнительной памяти кэш и осуществляет управление им. Считанная с жесткого диска информация заносится в кэш и сохраняется в нем до тех пор, пока не будет вытеснена другими данными. Если выполняемым программам требуется информация, имеющаяся в кэше, то обращение к диску не производится, а осуществляется передача прямо из кэша. Это позволяет повысить скорость считывания данных с «диска». Таким образом, кэш можно считать «старшим братом» буферов ввода-вывода. Вместе с тем драйвер SMARTDRV.SYS не обеспечивает использование кэша при записи информации на диск. Кэш дает наибольший эффект, когда одновременно используются несколько файлов, а также при свопинге.

Спецификация *dir* определяет каталог, содержащий файл SMARTDRV.SYS, если он не находится в корневом каталоге системного диска.

Аргумент *s* задает размер кэша в Кбайт. По умолчанию принимается 256, если кэш размещается в расширенной памяти, или вся отображаемая память, если кэш создается в последней.

Переключатель /A предписывает разместить кэш в отображаемой памяти. Если он не задан, то кэш создается в расширенной памяти.

Замечания:

— драйвер SMARTDRV.SYS несовместим с другими функционально аналогичными программами;

— драйвер SMARTDRV.SYS эффективнее функционирует в отображаемой памяти;

— дисковый кэш зачастую лучше виртуального диска, так как освобождает пользователя от каких-либо дополнительных действий по размещению файлов на виртуальном диске и обладает большей гибкостью, храня содержимое активных файлов, а не тех, которые явно специфицированы;

— при создании кэша необходимость (и целесообразность) использования команд FASTOPEN и BUFFERS= отпадает;

— созданный драйвером SMARTDRV.SYS кэш не обслуживает НГМД;

— файл с именем SMARTDRV.SYS имеется только в MS-DOS 4.0. В системе же PC DOS данный драйвер размещается в файле IBMCACHE.SYS, а синтаксис его запуска не документирован.

Примеры:

- DEVICE=SMARTDRV.SYS 1024 /A — создать в отображаемой памяти кэш емкостью 1 Мбайт;
- DEVICE=SMARTDRV.SYS /A — создать в отображаемой памяти кэш с использованием всего ее доступного объема.

Драйвер XMA2EMS.SYS

Назначение: управление отображаемой памятью в соответствии со спецификацией LIM EMS 4.0.
Синтаксис:

DEVICE=[*dir*\]XMA2EMS.SYS P0=*n1* P1=*n2* P2=*n3* P3=*n4* [P254=*n5*] [P255=*n6*] [/X:*m*]

или

DEVICE=[*dir*\]XMA2EMS.SYS FRAME=*n* [P254=*n5*] [P255=*n6*] [/X:*m*]

Комментарии. Драйвер XMA2EMS.SYS поддерживается только DOS, начиная с версии 4.0. Все числа *n* и *n1* — *n6* являются адресами, выражаемыми номерами параграфов (16-байт блоков) и задаваемыми в шестнадцатеричной системе счисления без явного указания символа H. Десятичное число *m* специфицирует размер памяти в страницах (16-Кбайт блоках).

Аргументы задают следующее:

P*i* (*i*=0,...,3) — номер параграфа в стандартном адресном пространстве, начиная с которого нужно создать *i*-ю 16-Кбайт страницу окна для отображаемой памяти;

FRAME — номер параграфа в стандартном адресном пространстве, начиная с которого следует последовательно и непрерывно разместить все четыре страницы окна отображаемой памяти. Аргумент FRAME несовместим с аргументами P*i*, позволяя более быстрым, но менее универсальным способом специфицировать окно;

P254 — номер параграфа в стандартном адресном пространстве, начиная с которого требуется разместить страницу, используемую командой FASTOPEN /X и драйвером VDISK.SYS /A. Если аргумент P254 не задан, то разместить буфер команды FASTOPEN и создать виртуальный диск в отображаемой памяти не удастся;

P255 — используется аналогично предыдущему аргументу, но применительно к команде конфигурирования BUFFERS=.

Таким образом, число 0, 1, 2 или 3 вслед за P в имени аргумента задает номер 16-Кбайт страницы окна отображаемой памяти, а само значение аргумента — базовый адрес этой страницы в стандартном адресном пространстве, выраженный номером первого параграфа. Другие аргументы обеспечивают обращение к заданным страницам отображаемой памяти минуя окно, но требуют наличия в стандартном адресном пространстве дополнительных неиспользуемых фрагментов.

Переключателем /X:*m* определяется объем используемой драйвером отображаемой памяти в страницах. Допустимы значения *m* в диапазоне от 4 (т.е. 64 Кбайт) до величины всей емкости имеющейся отображаемой памяти. По умолчанию принимается максимально возможное значение.

Замечания:

— после установки драйвера XMA2EMS.SYS выполняемые программы могут использовать все средства доступа к отображаемой памяти, предоставляемые функциями DOS по прерыванию 21H;

— подключите драйвер XMA2EMS.SYS до создания в отображаемой памяти виртуального диска, а также размещения в ней буферов командами BUFFERS= и FASTOPEN;

— для создания четырехстраничного окна отображаемой памяти обычно используется диапазон номеров параграфов от C000 до E000, которые по сути являются сегментными адресами;

— в аргументе P*i* допускается указывать номера страниц, отличные от перечисленных;

— если драйвер запущен в работу без аргументов, то он лишь выдает справочную информацию о доступной отображаемой памяти;

— если XMA2EMS.SYS указан в файле CONFIG.SYS после драйвера XMAEM.SYS, то переключатель /X отвергается, а используется значение, установленное ранее.

Примеры:

- DEVICE=XMA2EMS.SYS FRAME=D000 P254=C000 P255=C400 /X:64 — задействовать 64 16-Кбайт страницы отображаемой памяти для программ и 2 страницы для использования DOS, разместив окно по адресу D0000H, а страницы для DOS — по адресам C0000H и C4000H;
- DEVICE=XMA2EMS.SYS P0=D000 P1=D400 P2=D800 P3=DC00 P254=C000 P255=C400 /X:64 — то же.

Драйвер HIMEM.SYS

Назначение: управление расширенной памятью в соответствии со спецификацией XMS 2.0 и HMA-памятью.

Синтаксис:

DEVICE=[*dir*\]HIMEM.SYS [/HMAMIN=*n*] [/NUMHANDLES=*m*]

Комментарии. Драйвер HIMEM.SYS работает под управление DOS версии 3.0 и выше, но поставляется лишь с системой, начиная с версии 4.0. Он выполняет следующие функции:

1) обеспечивает доступ выполняемых программ к расширенной памяти в соответствии со спецификацией XMS 2.0;

2) обеспечивает возможность использования верхней HMA-памяти, но ничего в нее не загружает.

Каталог *dir* следует указать лишь тогда, когда файл HIMEM.SYS не находится в корневом каталоге системного диска.

После подключения драйвера к системе на экране появится сообщение

HIMEM: DOS XMS Driver, Version 2.04 — 8/17/88
Copyright 1988, Microsoft Corp.

Переключатель /HMAMIN=*n* специфицирует минимальный размер (в Кбайт) резидентной программы, которая может быть загружена в HMA-память (конечно, если она соответствующим образом оформлена). Значение *n* лежит в диапазоне 0 — 63. По умолчанию принимается 0, т.е. в HMA-память «пропускаются» все резидентные программы. Они размещаются в верхней памяти в порядке их загрузки. Переключатель /HMAMIN служит для эффективного ее использования, обеспечивая загрузку в HMA-память более крупных программ, а остальных — в стандартную память.

Например, если Вы устанавливаете сначала резидентную программу размером 15 Кбайт, а затем — 60 Кбайт и они обе способны размещаться в HMA-памяти, то укажите /HMAMIN=60. В этом случае первая программа будет загружена в стандартную память, а вторая — в верхнюю, что по сути сэкономит 60 Кбайт. Если переключатель не задан, то в HMA-память загрузится только первая программа. Для второй же в 64-Кбайт области не хватит места и поэтому она будет размещена в стандартной памяти. Это даст выигрыш всего в 15 Кбайт. Таким образом, задав переключатель /HMAMIN=60, Вы высвободите 45 Кбайт по сравнению с его отсутствием.

Когда Вы явно специфицировали переключатель /HMAMIN, при установке драйвера на экране появится сообщение

Minimum HMA size set to *n*
(Минимальный размер HMA-памяти установлен в *n*)

Переключатель /NUMHANDLES=*m* задает максимальное число одновременно используемых обработчиков для блоков расширенной памяти (EMB — Extended Memory Block). Это число (*m*) должно лежать в диапазоне 1 — 128, а по умолчанию принимается 32. Имейте в виду, что каждый обработчик дополнительно требует 6 байт в стандартной памяти. Поэтому специфицируйте переключатель явно только тогда, когда для выполняемых программ обработчиков, заданных по умолчанию, не хватает.

Если Вы указали переключатель /NUMHANDLES, то при установке драйвера на экран будет выдано сообщение

m extended memory handles available
(Доступно *m* обработчиков расширенной памяти)

Драйвер HIMEM.SYS допускает также следующие недокументированные переключатели:

/INT15=*p* — зарезервировать *p* Кбайт расширенной памяти для доступа к ней через прерывание 15H вместо спецификации XMS. Это необходимо для работы ряда программных продуктов (в частности, Paradox, QEMM, Oracle, Turbo EMS), которые обращаются к расширенной памяти непосредственно. В В качестве *p* допустимо значение из диапазона 64 — 65535, а по умолчанию принимается 0, т.е. вся расширенная память становится доступной в соответствии со спецификацией XMS;

/MACHINE:*c* — специфицировать тип адаптера адресной линии A20 в терминах кода или номера, если этот адаптер автоматически не распознается:

Код	Номер	Адаптер (тип ПЭВМ)
AT	1	IBM PC AT
	1	COMPUADD 386
	1	JDR 386/33
PS2	2	IBM PS/2
	2	Datamedia 386/486
	2	UNISYS PowerPort
PT1CASCADE	3	Phoenix Cascade BIOS
HPVECTRA	4	HP Vectra (A и A+)
ATT6300PLUS	5	AT&T 6300 Plus
ACER1100	6	Acer 1100
TOSHIBA	7	Toshiba 1600, 1200XE и 5100
WYSE	8	Wyse 12,5 МГц 286
	8	COMPUADD 386
	8	Hitachi HL500C
	8	Intel 301z или 302
TULIP	9	Tulip SX
ZENITH	10	Zenith ZBIOS
AT1	11	IBM PC AT (для будущего использования)
AT2	12	IBM PC AT (для будущего использования)
CSS	12	CSS Lab
AT3	13	IBM PC AT (для будущего использования)
PHILIPS	13	Philips
FASTHP	14	HP Vectra
BULLMICRAL	16	Bull Micral 60

/A20CONTROL:{ON|OFF} — брать на себя (ON) или нет (OFF) контроль за линией A20 даже в том случае, если она во время загрузки DOS была активной. По умолчанию принимается ON. Если указано OFF, то контроль со стороны драйвера за линией A20 будет осуществляться только в том случае, когда при подключении драйвера она была пассивной;

/SHADOWRAM:{ON|OFF} — выключать (OFF) или нет (ON) теневое ОЗУ. Теневое ОЗУ, являющееся областью памяти в стандартном адресном пространстве, хранит копию BIOS и подменяет ПЗУ для ускорения доступа. Если теневое ОЗУ выключается, то его область будет использоваться драйвером. По умолчанию принимается ON;

/CPUCLOCK:{ON|OFF} — корректировать (ON) или нет (OFF) тактовую частоту ПЭВМ. По умолчанию принимается OFF. ON можно указать в случае, когда при подключении драйвера тактовая частота изменяется. Однако при этом работа драйвера замедлится.

Замечания:

— очень важным свойством драйвера является то, что он позволяет разгрузить до 64 Кбайт стандартной (640-Кбайт) памяти;

— после установки драйвера резидентные программы в HMA-память автоматически не загружаются (они должны быть оформлены специальным образом);

— драйвер может быть запущен только на ПЭВМ класса AT и старше, имеющей расширенную память размером не менее 64 Кбайт;

— если расширенная память управляется другими программными продуктами, не использующими XMS-интерфейс, то сконфигурируйте их так, чтобы, по крайней мере 64-Кбайт область, не задействовалась;

— если Вы хотите установить драйверы, обращающиеся к расширенной памяти по стандарту XMS, то укажите HIMEM.SYS в файле CONFIG.SYS перед ними;

— если ПЭВМ не снабжена расширенной памятью, но имеет плату отображаемой памяти, то посредством переключателей на последней обычно можно преобразовать часть отображаемой памяти в расширенную для использования драйвером HIMEM.SYS.

Пример:

```
DEVICE=HIMEM.SYS /HMAMIN=40 /NUMHANDLES=128
```

Драйвер EMM386.SYS

Назначение: эмуляция отображаемой памяти в расширенной памяти на ПЭВМ с МП 80386(SX) или 80486(SX) в соответствии со стандартом LIM EMS 4.0.

Синтаксис:

DEVICE=[*dir*\]EMM386.SYS [*s*] [X:*m*—*n*]... [M*x*]

Комментарии. Драйвер EMM386.SYS работает под управлением DOS, начиная с версии 3.1.

Если файл EMM386.SYS не находится в корневом каталоге системного диска, то специфицируйте dir.

Аргументом *s* задается Кбайт размер расширенной памяти, используемой драйвером, Кбайт. По умолчанию принимается 256, а рекомендуется указывать 512.

Аргумент X:*m*-*n* специфицирует диапазон адресов в стандартном адресном пространстве, которые не должны использоваться в качестве 64-Кбайт окна отображаемой памяти. Границы этой области определяются значениями *m* и *n*, являющимися номерами параграфов (16-Кбайт блоков) и представляемыми в шестнадцатеричной системе счисления без явного указания символа H.

Аргумент M*x* явно указывает, по какому адресу в стандартном адресном пространстве разместить 64-Кбайт окно отображаемой памяти. Значение *x* выбирается из следующего списка при условии, что оно не вступает в конфликт с аргументом X:

x	Адрес в параграфах
0	C000
1	C400
2	C800
3	CC00
4	D000
5	D400
6	D800
7	DC00
8	E000

Драйвер автоматически распознает большинство плат расширения, создающих занятые окна в стандартном адресном пространстве (диапазон адресов 640 Кбайт — 1 Мбайт) и формирует свое окно в еще незадействованной области. Поэтому аргументы X и M следует задавать только в редких случаях, когда возникают коллизии между драйвером EMM386.SYS и платами расширения.

Замечания:

— драйвер может быть установлен только на ПЭВМ с аппаратно-программным интерфейсом по стандарту AT;

— все программные продукты, пользующиеся услугами драйвера, должны загружаться после его установки.

Примеры:

- DEVICE=EMM386.SYS 1024
- DEVICE=EMM386.SYS 512 X:C400—C7FF X:E000—E3FF

Драйвер XMAEM.SYS

Назначение: эмуляция платы отображаемой памяти в расширенной памяти на ПЭВМ с МП 80386(SX) или 80486(SX).

Синтаксис:

DEVICE=[*dir*\]XMAEM.SYS [*n*]

Комментарии. Этот драйвер имитирует наличие в ПЭВМ платы отображаемой памяти, используя под нее расширенную память. Единственный аргумент *n* указывает в 16-Кбайт страницах, какой объем расширенной памяти требуется преобразовать в отображаемую. По умолчанию используется вся доступная расширенная память.

Замечания:

— драйвер XMAEM.SYS должен устанавливаться перед драйвером XMA2EMS.SYS;

— пара драйверов XMAEM.SYS и XMA2EMS.SYS эквивалентна единственному драйверу EMM386.SYS.

Пример: DEVICE=XMAEM.SYS

5.11. Управление посимвольными устройствами

В рамках данного подраздела мы будем трактовать управление ПУ в узком смысле (главным образом в плане установки режимов их работы, хотя будут затрагиваться и некоторые другие вопросы).

Пользователь может управлять посимвольными устройствами следующими способами:

1) *аппаратно* с использованием имеющихся на устройстве клавиш и переключателей;

2) с помощью предназначенных для этого *команд DOS*;

3) *путем посылки на устройство специальных управляющих последовательностей* (команд);

4) *с использованием средств, предоставляемых различными программными продуктами*, в частности системами программирования и текстовыми, а также графическими редакторами.

Однозначно ответить на вопрос о том, какой из способов наилучший, не представляется возможным. Все зависит от того, что пользователю требуется. Третий способ обладает самыми богатыми возможностями, но он в то же время сложнее всех остальных.

Команды DOS для управления посимвольными устройствами уже рассмотрены нами в п. 5.6.4 и подразделе 5.10. Изучение программных продуктов, способных управлять устройствами, в наши цели не входит.

Поэтому остается обсудить два из перечисленных способов: аппаратный и с использованием управляющих последовательностей. Ограничимся рассмотрением только часто используемых ПУ: консолью (дисплеем и клавиатурой), а также точечно-матричным принтером.

5.11.1. Управление дисплеем и клавиатурой с использованием Escape–последовательностей

Консоль обычно не предоставляет каких-либо аппаратных возможностей по управлению ею, за исключением регулировки яркости и контрастности изображения на экране дисплея.

Для использования управляющих последовательностей (в данном случае — Escape-последовательностей) требуется осуществить подключение к DOS драйвера ANSI.SYS (см. описание команды DEVICE= в п. 5.10.2 и драйвера ANSI.SYS в п. 5.10.3).

Escape-последовательность представляет собой последовательность символов, первым из которых является литера ESC, имеющая код 27 (1BH) и графическое изображение ←. Для воздействия на консоль Escape-последовательность нужно послать на дисплей. Она будет перехвачена драйвером ANSI.SYS, который и выполнит требуемые действия.

Структура Escape-последовательности такова. Сначала следует пара символов ←[. Затем располагаются *операнды* — числа, разделенные точкой с запятой. А завершается последовательность однобуквенной *командой.*

В Escape-последовательностях одноименные строчные и прописные буквы не рассматриваются как эквивалентные. Все числовые величины (которые должны задаваться в символьном виде) представляются в последовательностях с использованием десятичной системы счисления. Строки и столбцы экрана нумеруются, начиная с 1. Символ [принадлежит последовательности, а не является метасимволом.

Допустимы следующие Escape-последовательности:

1) *для управления курсором:*

← [*l*;*c*H — позиционирует (устанавливает) курсор в *c*-ю позицию *l*-й строки. По умолчанию (← [H) курсор устанавливается в верхний левый угол экрана;

← [*l*;*c*f — то же;

← [*n*A — перемещает курсор вверх на *n* строк без изменения позиции внутри строки. По умолчанию (← [A) принимается 1. Если курсор уже находится в верхней строке, то последовательность игнорируется;

← [*n*B — перемещает курсор вниз на *n* строк без изменения позиции внутри строки. По умолчанию (← [B) принимается 1. Если курсор уже находится в нижней строке, то последовательность игнорируется;

← [*n*C — перемещает курсор по строке вправо на n позиций. По умолчанию (← [C) принимается 1. Если курсор уже находится в крайней правой позиции строки, то последовательность игнорируется;

← [*n*D — перемещает курсор по строке влево на *n* позиций. По умолчанию (← [D) принимается 1. Если курсор уже находится в крайней левой позиции, то последовательность игнорируется;

← [s — запоминает текущую позицию курсора (номера строки и позиции в строке, т.е. колонки) для обеспечения возможности последующей его установки в эту позицию. Более одной позиции запомнить нельзя;

← [u — восстанавливает позицию курсора (устанавливает курсор в позицию, запомненную последней из последовательностей ← [s);

← [6n — выводит на экран текущую позицию курсора в форме ← [*l*; *s*R, где *l* — номер строки, а *s* — номер столбца;

2) *для удаления символов с экрана дисплея:*

← [2J — очищает экран и устанавливает курсор в его левый верхний угол;

← [K — удаляет все символы от курсора до конца строки (включая позицию курсора);

3) *для установки режима работы дисплейного адаптера и выбора цветов:*

← [*s*;...*s*m — устанавливает атрибуты дисплея в текстовом режиме, которые действуют до тех пор, пока не будут изменены такой же последовательностью. Однако атрибуты

уже выведенной информации не изменяются, вследствие чего целесообразно предварительно очищать экран. В качестве атрибутов *s* в любом порядке можно задать режим отображения символов, цвет символов и цвет фона. Если какой-либо из атрибутов не указан, то продолжает действовать установленный.

Допустимы следующие режимы отображения символов:

0 — сбросить все атрибуты, установив исходные (белые символы на черном фоне);
1 — повысить интенсивность выводимых символов, что в совокупности с допустимыми восемью цветами позволяет получить 16 цветов;
4 — выводить символы с подчеркиванием (только для монохромного дисплея);
5 — обеспечить мерцание выводимых символов;
7 — включить режим реверсивного отображения (символы выводить фоновым цветом, а фон — цветом символов);
8 — отключить вывод (задать черный цвет для фона и символов).

Можно установить следующие цвета символов (текста):

30 — черный (Black);
34 — синий (Blue);
32 — зеленый (Green);
36 — бирюзовый (Cyan);
31 — красный (Red);
35 — пурпурный (Magenta);
33 — коричневый, или охристый (Yellow);
37 — серебристый (White).

В качестве цвета фона можно задать такие же цвета:

40 — черный (Black);
44 — синий (Blue);
42 — зеленый (Green);
46 — бирюзовый (Cyan);
41 — красный (Red);
45 — пурпурный (Magenta);
43 — коричневый, или охристый (Yellow);
47 — серебристый (White).

←[=*s*h — устанавливает заданный числом *s* режим (текстовый или графический) работы дисплейного адаптера. Допустимы следующие режимы:

0 — 40x25 символов, монохромный;
1 — 40x25 символов, цветной;
2 — 80x25 символов, монохромный;
3 — 80x25 символов, цветной;
4 — 320x200 точек, цветной;
5 — 320x200 точек, монохромный;
6 — 640x200 точек, черно-белый (двухцветный);
7 — режим переноса конца длинной строки на следующую строку (иначе — отсечение);
14 — 640x200 точек, цветной;
15 — 640x350 точек, монохромный;
16 — 640x350 точек, цветной;
17 — 640x480 точек, цветной;
18 — 640x480 точек, цветной;
19 — 320x200 точек, цветной.

Режимы 0 — 3 и 7 являются текстовыми, а остальные — графическими. Возможность установки того или иного режима зависит от типов дисплейного адаптера и дисплея. Режимы 14 — 19 доступны в DOS, начиная с версии 4.0 (для EGA и VGA). Умолчание для данной последовательности отсутствует;

←[=*s*l — то же, но ←[=7l отменяет перенос длинных строк;

4) *для переопределения клавиш клавиатуры и их комбинаций:*

←[*k*;...*k*p — переопределяет единственную клавишу или комбинацию клавиш таким образом, что при ее нажатии в буфер клавиатуры будет помещаться отличный от стандартного код (последовательность кодов). Это позволяет определять пользовательские макрокоманды, но в отличие от командных файлов — только без параметров. Здесь *k* — десятичное число или последовательность символов, заключенная в кавычки. Переопределяемая клавиша (комбинация клавиш) представляется одним или двумя первыми *k*. Если клавиша (комбинация клавиш) стандартно генерирует один из кодов ASCII, то ее можно задать либо соответствующим кодом (в десятичной системе счисления), либо соответствующим символом, заключенным в кавычки. Так, символ A можно представить как 65 (см. подраздел 5.3) или как "A". В случае, когда начальный операнд *k* является строкой более чем из одного

символа, заключенной в кавычки, то первый из них обозначает переопределяемую клавишу. Если клавиша (комбинация клавиш) стандартно генерирует расширенный код клавиши, то ее можно представить только двумя числами, разделенными точкой с запятой (см. табл. 5.14 в п. 5.4.2). Остальные *k* и, возможно, остаток первого *k*, задают последовательность символов, которая будет генерироваться при нажатии указанной клавиши (комбинации клавиш). В частном случае эту последовательность символов можно представить единственной строкой. Однако если требуется задать управляющие символы или расширение ASCII, то придется использовать соответствующие коды. Кроме того, можно специфицировать и расширенные коды клавиш. Для восстановления прежнего смысла нажатия клавиши (комбинации клавиш) следует продублировать ее код, в частности расширенный код клавиши.

Конечно, на устройство CON можно выдать несколько Escape-последовательностей подряд, но каждая из них должна начинаться с символов ←[.

Примеры Escape-последовательностей:

- ←[10;30H — переместить курсор в 30-ю позицию 10-й строки экрана;
- ←[3B — сместить курсор вниз на три строки;
- ←[1;34;41m — обеспечить вывод последующего текста голубым цветом на красном фоне;
- ←[=4h — переключить дисплейный адаптер в цветной графический режим с разрешением 320x200 точек;
- ←["\";"?"p — переопределить клавишу \ на ?;
- ←[92;63p — то же (92 — код символа \, а 63 — код символа ?);
- ←["\?"p — то же;
- ←["\";"?"p←["?";"\"p — поменять местами клавиши \ и ?;
- ←[4;"DIR C:";13p — переопределить комбинацию клавиш Ctrl-D так, чтобы при ее нажатии вводилась строка DIR C:◄Enter► (13 — код символа CR);
- ←[0;68;"DIR";13p — переопределить клавишу F10 так, чтобы при ее нажатии выдавалась команда DIR (вводилась строка DIR◄Enter►);
- ←[0;68;0;68p — восстановить стандартную интерпретацию клавиши F10.

Для вывода Escape-последовательности на дисплей можно воспользоваться одним из следующих способов:

1) создать программным способом или текстовым редактором текстовый файл с Escape-последовательностью и вывести его содержимое на экран командой TYPE;

2) создать текстовым редактором командный файл с командой ECHO, в качестве аргумента которой указана Escape-последовательность, и выполнить его. Пример такого командного файла:

```
@ECHO OFF
ECHO ←[0;68;"DIR";13p
```

3) использовать команду PROMPT с двухсимвольной комбинацией $E (см. п. 5.6.5), например:

```
PROMPT $E[0;68;"DIR";13p,
```

вслед за которой нужно ввести команду для восстановления приглашения DOS прежнего вида. Тогда первой командой будет единственный раз выдано приглашение, содержащее Escape-последовательность, а второй — «нормальное» приглашение. Команды PROMPT такого вида удобнее специфицировать в командном файле и затем его при необходимости выполнять.

Кроме того, Escape-последовательность можно вывести на экран непосредственно из программы. В качестве примера приведем программу на языке Turbo Pascal версий 5.0 — 6.0, иллюстрирующую применение Escape-последовательностей:

```
Program ShowANSI;
Begin
    Write(#27'[5;31;44m');          {установка следующих атрибутов дисплея:
                                        мерцающие символы;
                                        цвет символов — красный;
                                        цвет фона — синий. }
    Write(#27'[2J');                                {очистка экрана}
    Write(#27'[13;15H');        {позиционирование курсора для вывода текста}
    Write ('Вы видите красные мерцающие символы на синем фоне');
    ReadLn;                                 {пауза до нажатия клавиши Enter}
    Write(#27'[0m');            {установка исходных атрибутов дисплея}
    Write(#27'[2J')                                 {очистка экрана}
End.
```

Эта программа обеспечивает вывод текста «Вы видите красные мерцающие символы на синем фоне» действительно красными мерцающими символами в середине синего экрана.

Тем не менее выдача Escape-последовательнотей из программ на языках высокого уровня не является целесообразной, так как такие языки предоставляют в распоряжение пользователя гораздо более совершенные средства управления видеосистемой. В программах же на языках уровня Ассемблера применение Escape-последовательностей оправдано.

Символ ESC невозможно передать системе с клавиатуры (в частности, в команде ECHO), так как он перехватывается DOS и интерпретируется как немедленная отмена командной строки. Аналогично многие текстовые редакторы реагируют на нажатие клавиши Esc специальным образом, и поэтому их использовать для подготовки Escape-последовательностей не представляется возможным. Однако иногда (например, в текстовом редакторе ЛЕКСИКОН) все же можно задать символ ESC путем набора кода 27 на малой цифровой клавиатуре на фоне нажатой клавиши Alt. Редактор же EDLIN позволяет вводить практически любые управляющие символы, в частности ESC.

Составление программы для формирования текстового файла с Escape-последовательностями трудоемко.

По указанным причинам для пользователя наилучшим является третий способ формирования и вывода Escape-последовательностей (в этом случае символ ESC задается косвенно — через $E).

5.11.2. Управление принтером

В качестве примера для рассмотрения выберем 9-игольчатый точечно-матричный принтер среднего класса EPSON LX-800 японской корпорации Seiko Epson, который поставляется с ПЭВМ серии ЕС, а также с другими IBM-совместимыми машинами. Многие производители печатающих устройств обеспечивают совместимость своих изделий с принтерами фирмы Epson.

Сначала дадим общую характеристику возможностей принтера, а затем опишем управление им аппаратно и при помощи управляющих последовательностей. После этого рассмотрим методы определения новых символов и печати иллюстраций.

Общая характеристика принтера EPSON LX-800

Печатающее устройство EPSON LX-800 обладает следующими техническими характеристиками:

Метод печати (принцип действия): точечно-матричный;
Цвет печати: черный;
Режимы работы: текстовый и графический;

Градации качества печати в текстовом режиме:

Draft (черновая печать);
NLQ (печать с качеством, близким к типографскому);

Максимальная высота печатного символа: 3,1 мм;
Ширина символа: 2,1 или 1,05 мм с возможностью увеличения вдвое;
Плотность печати: 10, 12, 17 или 20 символ/дюйм с возможностью уменьшения вдвое;
Число символов в строке: 80, 96, 132 или 160 (узкая каретка) с возможностью уменьшения вдвое;
Межстрочный интервал: 1/6 дюйма (4,233 мм) или программируется с дискретностью 1/216 дюйма (0,118 мм);

Скорость печати:

180 символ/с при печати с плотностью 12 символ/дюйм и качеством Draft;
150 символ/с при печати с плотностью 10 символ/дюйм и качеством Draft;
25 символ/с при печати с плотностью 10 символ/дюйм и качеством NLQ;

Разрешающая способность в графическом режиме (плотность печати):

по вертикали — 72 точка/дюйм (2,835 точка/мм);
по горизонтали — до 240 точка/дюйм (9,449 точка/мм);

Направление печати: двунаправленное в текстовом режиме;
однонаправленное в графическом режиме;
Число копий (под копирку): до 3 при условии, что толщина пакета не превышает 0,25 мм;
Ширина бумаги: рулонная: 4 — 10 дюймов (102 — 254 мм);
листовая: 7,15 — 8,5 дюймов (182 — 216 мм);
Срок службы печатающей головки: 200 млн. символов при 14 точках в символе;
Габариты: 399x308x91 мм;
Масса: 5,1 кг;
Напряжение: 220 В переменного тока;
Потребляемая мощность: 70 В·А.

В текстовом режиме обеспечивается печать с использованием различных стилей. Под **стилем** будем понимать совокупность характеристик печатаемых символов и их расположения в строке.

Стиль определяется:

1) *шрифтом* (формой символов);
2) *шагом* (плотностью размещения символов в строке);
3) *выделением символов*;
4) *качеством печати.*

Принтер EPSON LX-800 поддерживает один **шрифт** (без названия) в режиме Draft и два **шрифта** (Roman и Sans Serif) в режиме NLQ. Шрифт Roman является прямым шрифтом, аналогичным шрифту пишущей машинки. Шрифт Sans Serif — это так называемый рубленый шрифт (символы угловаты, т.е. менее округлы, чем в предыдущем шрифте). Шрифты могут задаваться как аппаратно (переключателями или клавишами), так и посредством управляющих последовательностей.

Следующие две составляющие стиля являются по сути *модификаторами шрифта,* видоизменяя его тем или иным образом. Поэтому можно считать, что EPSON LX-800 обеспечивает печать гораздо бо́льшим количеством шрифтов.

Поддерживаются следующие модификаторы шрифта, определяющие **шаг**:

— нормальный (Pica);
— полусжатый (Elite);
— сжатый (compressed, или condenced);
— растянутый (expanded, или double-width).

Нормальный шрифт соответствует французскому типографскому стандарту, распространенному в Европе (в английской системе ему соответствует «цицеро»).

Полусжатый шрифт отличается от нормального только уменьшенным интервалом между символами.

Модификатор *«сжатый»* используется совместно с нормальным или полусжатым шрифтами, увеличивая плотность размещения символов примерно на 70% (см. табл. 5.20).

Таблица 5.20

Влияние модификаторов шага на плотность размещения символов

Модификаторы	Ширина символа, мм	Плотность, символ/дюйм	Максимальное число символов в строке
нормальный	2,1	10	80
полусжатый	2,1	12	96
нормальный, сжатый	1,05	17	132
полусжатый, сжатый	1,05	20	160

Модификатор *«растянутый»* совместим со всеми представленными в табл. 5.20 комбинациями и приводит к увеличению ширины символов, а также промежутков между ними вдвое. Поэтому плотность символов и максимальное число символов в строке уменьшаются ровно в два раза.

В принтере EPSON LX-800 реализованы следующие возможности **выделения** символов (т.е. текста):

— *полужирная печать* (символ выводится два раза с небольшим смещением печатающей головки вправо при повторной печати);

— *двухпроходная печать* (каждая строка выводится два раза с незначительной протяжкой бумаги перед вторым проходом печатающей головки);

— *печать курсивом* (с наклоном);

— *печать с подчеркиванием.*

Двухпроходная печать аналогична режиму NLQ, и поэтому в NLQ она игнорируется, хотя и может специфицироваться.

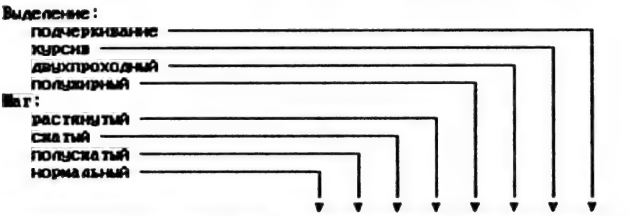
За некоторым исключением перечисленные модификаторы совместимы между собой и с модификаторами, определяющими шаг. Информация по совместимости модификаторов шрифта и способам их задания (установки) представлена в табл. 5.21. В ней приняты следующие обозначения:

Т — переключатели (тумблеры), расположенные на задней стенке принтера;
К — клавиши на пульте управления, размещенного в правом ближнем углу крышки печатающего устройства;
П — управляющие последовательности;
D — режим Draft;
Q — режим NLQ.

В колонке «Способы установки» указаны альтернативные способы задания модификаторов, а в других клетках определяется совместимость модификаторов, которая зависит от выбранного

Таблица 5.21

Способы установки и совместимость модификаторов шрифта



Модификатор		Способы установки								
Шаг	нормальный	К, П		–	D&Q	D&Q	D&Q	D	D&Q	D&Q
	полусжатый	К, П	–		D	D&Q	D&Q	D	D&Q	D&Q
	сжатый	Т, К, П	D&Q	D		D&Q	D	D	D	D&Q
	растянутый	П	D&Q	D&Q	D&Q		D&Q	D	D&Q	D&Q
Выделение	полужирный	К, П	D&Q	D&Q	D	D&Q		–	D&Q	D&Q
	двухпроходный	К, П	D	D	D	D	–		D	D&Q
	курсив	Т, П	D&Q	D&Q	D	D&Q	D&Q	D		D&Q
	подчеркивание	П	D&Q	D&Q	D&Q	D&Q	D&Q	D&Q	D&Q	

качества печати. Если специфицировано D, то модификаторы совместимы только в режиме Draft, а если D&Q — то как в режиме Draft, так и в режиме NLQ. Автономно могут использоваться только два взаимно исключающих друг друга модификатора: «нормальный» и «полусжатый».

Принтер EPSON LX-800 снабжен СППЗУ, в котором записаны четыре пары *(аппаратных) кодовых страниц*:

1) две кодовые страницы, соответствующие основной кодировке кириллицы;
2) две кодовые страницы для курсива;
3) две кодовые страницы, соответствующие кодовой странице 437 (см. табл. 5.3 в подразделе 5.3);
4) две кодовые страницы, соответствующие альтернативной кодировке (см. табл. 5.5 в подразделе 5.3).

Кодовые страницы в каждой паре различаются тем, что в первой из них кодами 80H — 9FH представляются те же символы, что и кодами 00H — 1FH, т.е. осуществлено дублирование двух колонок кодовой таблицы. Вторая кодовая страница пары совпадает с соответствующим ей оригиналом (прототипом). Кодовые страницы для курсива содержат в первой половине символы прямого, а во второй — наклонного начертания. Русские буквы в них не входят.

Требуемая пара кодовых страниц выбирается путем соответствующей установки переключателей или выдачи управляющей последовательности. Одна из кодовых страниц пары может быть выбрана только управляющей последовательностью, а другая устанавливается автоматически при включении питания.

Дополнительно к поддержке различных кодовых страниц принтер EPSON LX-800 обеспечивает *смену символов* с кодами 23H, 24H, 40H, 5BH, 5CH, 5DH, 5EH, 60H, 7BH, 7CH, 7DH и 7EH в зависимости от выбранного *набора национальных символов*. Требуемый набор (один из восьми) устанавливается переключателями или управляющей последовательностью. Дополнительно к этому можно задействовать еще пять наборов, но только путем посылки на принтер управляющей последовательности. С целью полной поддержки кодовой страницы 437, основной и альтернативной кодировки следует выбрать набор национальных символов для США.

Кодовые таблицы и наборы национальных символов хранятся в СППЗУ в виде совокупностей матриц символов. Матрицы описывают порядок вычерчивания соответствующих символов по точкам, а требуемая матрица во время печати выбирается в соответствии с кодом символа и установленной кодовой страницей (с учетом набора национальных символов). Поэтому часть СППЗУ, хранящую матрицы символов, называют **знакогенератором**. Матрицы символов различаются не только в зависимости от кодовой страницы, но и от уровня качества печати. В принципе знакогенератор можно перепрограммировать, но для этого требуется специальное оборудование.

Матрицы для модифицированных шрифтов в СППЗУ отсутствуют. Требуемые символы в этих условиях формируются из матриц для режима Draft (или NLQ) при помощи микропрограмм, записанных в СППЗУ.

Рассматриваемый принтер вместе с тем позволяет *заменить до шести стандартных символов специально разработанными*. Чтобы произвести замену, нужно подготовить матрицы символов и записать их в предназначенное для этого ОЗУ небольшого объема, находящееся в принтере. Режимы Draft и NLQ снабжаются отдельными матрицами.

Возможности принтера EPSON LX-800 по *форматированию текста* сводятся к следующему:

— задание межстрочного интервала;

— автоматическое разбиение текста на страницы (даже при отсутствии в выводимых данных команд на перевод страницы) или печать текста непрерывным потоком (с возможностью перевода страницы по командам в выводимой информации);

— печать надстрочных и подстрочных индексов;

— выключка (выравнивание текста) по левой или правой границе, а также центрирование;

— горизонтальная и вертикальная табуляция.

Аппаратный способ управления принтером весьма ограничен по своим возможностям, но удобен для пользователей среднего уровня квалификации, а также для начальной установки режимов работы принтера с целью единообразного вывода всего текста.

Управляющие последовательности (команды принтера) реализуют все его возможности и позволяют печатать фрагменты текста различными стилями, так как сами команды могут быть помещены в выводимый текст. Этот способ управления принтером доступен только квалифицированным пользователям.

Команды DOS и различные программные продукты используют команды принтера, но в общем случае — ограниченное их подмножество. Программы, способные управлять принтерами, учитывают их особенности, как правило, предлагая задать пользователю модель устройства. Если принтер EPSON LX-800 програ́ммным продуктом не поддерживается, то следует выбрать одну из следующих моделей, которые перечислены в порядке уменьшения предпочтения: FX-800, EX-800, LX-80, FX-85, FX-80, RX-80, принтер EPSON или принтер для черновика.

Все установки, сделанные любыми способами, за исключением переключателей, при отключении питания сбрасываются.

Использование переключателей и клавиш

Принтер EPSON LX-800 имеет двенадцать *переключателей*, расположенных на задней стенке устройства и разбитых на группы. 8 переключателей первой группы обозначаются как 1-1 — 1-8, а 4 тумблера второй группы — как 2-1 — 2-4. Положение переключателя можно изменить тонким острым предметом, например отверткой или шариковой ручкой. Состояние «Включен» (ON) соответствует верхнему положению переключателя.

Состояние переключателей определяет режимы работы, в которые устанавливается принтер при включении питания или при получении команды ESC @. Затем режимы могут быть изменены *клавишами* и/или управляющими последовательностями.

Смену положения переключателей нужно производить только при отключенном питании принтера.

Функции переключателей сведены в табл. 5.22 — 5.24. При выборе режима NLQ переключателем 1-5 (табл. 5.22) устанавливается шрифт Roman. Для использования в СНГ нужно включить переключатели 1-3, 1-4, 1-6, 1-7 и 1-8 (табл. 5.23 и 5.24). При этом будет выбрана альтернативная кодовая таблица (никакая дополнительная команда не требуется). Вот почему драйвер, поддерживающий на принтере русский шрифт, подключать нет никакой необходимости.

Включение питания осуществляется выключателем, расположенным на левой стенке принтера.

После подачи питания выбор режимов работы принтера может быть осуществлен при помощи пульта управления, на котором расположены четыре индикаторные лампочки и три клавиши (рис. 5.15).

Индикаторные лампочки информируют о состоянии печатающего устройства:

POWER (ПИТАНИЕ) — загорается при включении питания;

READY (ГОТОВО) — горит, когда принтер готов к приему данных (во время печати мигает);

PAPER OUT (КОНЕЦ БУМАГИ) — загорается, когда в принтере заканчивается лист (рулон) бумаги или бумага неправильно установлена;

ON LINE (НЕАВТОНОМНЫЙ) — горит, когда принтер находится в логической связи с ПЭВМ, т.е. может принимать данные (эта лампочка совмещена с клавишей ON/OFF LINE).

Сенсорные клавиши имеют следующее назначение:

ON/OFF LINE (НЕАВТОНОМНЫЙ/АВТОНОМНЫЙ) — переключает принтер попеременно в неавтономное и автономное состояние;

NLQ/FORM FEED (РЕЖИМ NLQ/ПЕРЕВОД СТРАНИЦЫ) — в автономном состоянии приводит к прогону рулонной бумаги до начала следующей страницы, а в неавтономном — к выбору режима печати NLQ. Дополнительные нажатия этой клавиши в неавтономном состоянии принтера обеспечивают попеременный выбор шрифта Roman или Sans Serif (звуковая сигнализация срабатывает два или три раза соответственно);

Таблица 5.22

Функции переключателей

Номер переключателя	Функция	ВКЛ (ON)	ВЫКЛ (OFF)	Заводская установка
1-1	Шаг	сжатый	нормальный	ВЫКЛ
1-2	Вид символа "нуль"	Ø	0	ВЫКЛ
1-3	Кодовая страница	см. табл. 5.23		ВЫКЛ
1-4				ВКЛ
1-5	Качество печати	NLQ	Draft	ВЫКЛ
1-6	Набор национальных символов	см. табл. 5.24		ВКЛ
1-7				ВЫКЛ
1-8				ВКЛ
2-1	Длина страницы	12 дюймов	11 дюймов	ВКЛ
2-2	Устройство подачи листовой бумаги	действует	не действует	ВЫКЛ
2-3	Разбиение на страницы при печати на рулонную бумагу	действует	не действует	ВЫКЛ
2-4	Автоматический перевод строки по символу CR	действует	зависит от наличия символа LF	ВЫКЛ

Таблица 5.23

Выбор пары кодовых страниц

Номер пары кодовых страниц	Номер переключателя	
	1-3	1-4
2	ВЫКЛ	ВЫКЛ
1	ВЫКЛ	ВКЛ
3	ВКЛ	ВЫКЛ
4	ВКЛ	ВКЛ

Таблица 5.24

Установка набора национальных символов

Страна	Номер переключателя		
	1-6	1-7	1-8
США	ВКЛ	ВКЛ	ВКЛ
Франция	ВКЛ	ВКЛ	ВЫКЛ
Германия	ВКЛ	ВЫКЛ	ВКЛ
Великобритания	ВКЛ	ВЫКЛ	ВЫКЛ
Дания	ВЫКЛ	ВКЛ	ВКЛ
Швеция	ВЫКЛ	ВКЛ	ВЫКЛ
Италия	ВЫКЛ	ВЫКЛ	ВКЛ
Испания	ВЫКЛ	ВЫКЛ	ВЫКЛ

DRAFT/LINE FEED (РЕЖИМ DRAFT/ПЕРЕВОД СТРОКИ) — в автономном состоянии приводит к протяжке бумаги на одну строку, а в неавтономном — к выбору режима печати Draft (звуковая сигнализация срабатывает один раз). Эта кнопка управляет также автоматической загрузкой в принтер следующего листа бумаги (AUTO LOAD) в автономном состоянии, если бумага не заправлена.

С целью проверки работоспособности принтера и получения образцов шрифтов можно осуществить его *автономное тестирование*. Для этого следует:

1) заправить бумагу;

2) отключить питание и затем снова его включить, удерживая кнопку DRAFT нажатой.

Принтер начинает печать различных символов с тем качеством и тем шрифтом, которые установлены переключателем 1-5. Удерживать клавишу DRAFT далее нет необходимости. Тестирование прекращается при отключении питания или окончании бумаги.

Если в соответствии с п. 2 удерживать не клавишу DRAFT, а клавишу NLQ, то независимо от положения переключателя 1-5 тестирование будет осуществляться в режиме NLQ (шрифты Roman и Sans Serif будут чередоваться).

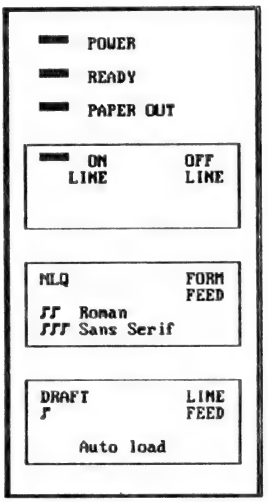


Рис. 5.15. Пульт управления

Клавиши могут использоваться также для выбора одного или нескольких модификаторов шрифтов (см. табл. 5.21). Чтобы сделать это, следует включить режим выбора модификатора и затем установить требуемый модификатор (или несколько модификаторов).

Для включения режима выбора модификатора требуется:

1) перевести устройство в неавтономное состояние;
2) удерживая клавишу ON/OFF LINE нажатой, нажать и клавишу FORM FEED.

В результате этих действий вырабатывается короткий звуковой сигнал и лампочка ON LINE начинает мигать. В этом режиме клавиши выполняют следующие функции:

ON/OFF LINE — осуществляет указание модификатора, который требуется задать;

FORM FEED — осуществляет фиксацию указанного модификатора;

LINE FEED — отменяет режим выбора модификатора шрифта.

Для *установки модификатора шрифта* после включения режима его выбора следует:

1) определить номер n требуемого модификатора в соответствии со следующим:

1 — полужирный шрифт;
2 — двухпроходный шрифт;
3 — сжатый шрифт;
4 — полусжатый шрифт;

2) нажать клавишу ON/OFF LINE n раз для указания требуемого модификатора;
3) нажать клавишу FORM FEED для фиксации модификатора;
4) нажать клавишу LINE FEED для отмены режима выбора модификатора;
5) нажать клавишу ON/OFF LINE для перевода принтера в неавтономное состояние.

Если требуется установить два совместимых модификатора с номерами n и m, причем $n<m$, то следует выполнить пункты 1 — 3 для n, а затем — пункты 2 и 3 для $m-n$. Задание трех совместимых модификаторов с номерами n, m и k при условии, что $n<m<k$, осуществляется аналогично: сначала фиксируется модификатор n, затем — m—n и, наконец, k—m.

Например, для установки модификаторов «сжатый» и «полусжатый» следует:

1) нажать клавишу ON/OFF LINE 3 раза;
2) нажать клавишу FORM FEED;
3) нажать клавишу ON/OFF LINE 1 раз;
4) нажать клавишу FORM FEED;
5) нажать клавишу LINE FEED;
6) нажать клавишу ON/OFF LINE.

Если в печатаемом документе есть управляющие последовательности, то они будут отменять соответствующие выбранные клавишами модификаторы. В противном случае модификаторы будут действовать до тех пор, пока не отключено питание, так как, например, модификатор «нормальный» не может быть задан клавишами. Иными словами, переустановка модификаторов клавишами невозможна.

Принтер EPSON LX-800 обеспечивает *выдачу дампа данных*, т.е. в точности той информации, которая получена устройством. Это позволяет проверить, как те или иные коды интерпретируются принтером. Такая возможность облегчает для опытного пользователя выявление причин неправильной печати.

Для печати дампа следует, удерживая клавиши FORM FEED и LINE FEED в нажатом состоянии, включить питание. На бумаге отпечатается сообщение «Data Dump Mode». После этого клавиши можно отпустить. В таком режиме выдача данных на принтер приводит к печати строк, каждая из которых состоит из трех полей — номера строки, шестнадцатеричных кодов и их символьной интерпретации. Если код во втором поле не имеет графического представления (является для принтера управляющим), то в третьем поле он представляется точкой в соответствующей позиции. В режиме дампирования принтер на управляющие последовательности больше никак не реагирует. Для печати последней строки полученных принтером данных нужно нажать клавишу ON/OFF LINE. После этого режим печати дампа отменяется.

Команды принтера

Все модели принтеров фирмы Seiko Epson, а также многие другие печатающие устройства управляются **командами (управляющими последовательностями)** ESC/P (Epson Standard Code for Printer — стандартный код фирмы Epson для принтера). Эти команды пересылаются в принтер вперемежку с текстом, который требуется отпечатать. Принтер выделяет их из информационного потока и выполняет предписанные командами действия. В печатаемом документе команды никак не отображаются, но приводят к изменению стиля документа и к его форматированию.

Команда состоит из кода операции, за которым могут следовать несколько операндов (аргументов). *Код операции* однозначно определяет команду и представляется либо управляющим символом ASCII, либо (так как управляющих символов всего 33) двумя символами — ESC и еще каким-либо, в том числе цифрой (но не числом, а символом). *Операнды* модифицируют (уточняют) действие команды или поставляют необходимую для ее выполнения информацию. Каждый операнд размещается в одном байте.

Некоторые команды допускают два альтернативных способа задания операнда *n*, где *n* — *число* 0 или 1: один из них состоит в спецификации операнда его *двоичным кодом* (00H или 01H), а другой — как *символа* (30H или 31H — см. табл. 5.3 в подразделе 5.3). Такие возможности мы будем обозначать через <*n*>.

Команды будем записывать последовательностью обозначений входящих в них символов (графические представления давать нецелесообразно из-за того, что получить их для управляющих символов совсем не просто). Шестнадцатеричную и десятичную кодировки можно узнать, используя табл. 5.3 и 5.4 из подраздела 5.3. *Числовые операнды* (все они должны представляться двоичными кодами) будут обозначаться одиночными строчными латинскими буквами, набранными курсивом. Возможно приписывание справа номера операнда. *Символьные операнды* обозначаются так же, но для них делаются соответствующие оговорки.

Если не отмечено противное, *символ* О представляет букву, а не цифру.

Принтер EPSON LX-800 воспринимает команды следующих функциональных групп:

1) *команды подготовки принтера к работе:*

ESC @ — инициализация принтера (как при включении питания). Текущая выдача немедленно отменяется;

DC1 — выбор принтера (перевод в неавтономное состояние), если он был переведен в автономное состояние командой DC3. Устройство, установленное в автономное состояние клавишей ON/OFF LINE, не выбирается;

DC3 — невыбор принтера (перевод в автономное состояние до приема команды DC1). При этом устройство не может быть выбрано нажатием клавиши ON/OFF LINE;

ESC *s* <*n*> — включение (*n* = 1) или выключение (*n* = 0) режима пониженной скорости для уменьшения уровня шума;

ESC < — установка режима однонаправленной печати для текущей строки (отменяется возвратом каретки) с целью более точного позиционирования печатающей головки;

ESC U <*n*> — включение (*n* = 1) или выключение (*n* = 0) режима однонаправленной печати для более точного позиционирования печатающей головки;

ESC 8 — запрет срабатывания датчика конца бумаги с целью печати до конца листа;

ESC 9 — разрешение срабатывания датчика конца бумаги, в результате чего примерно за 30 мм до конца бумажного листа выдается звуковой сигнал и печать приостанавливается до заправки следующей страницы;

ESC EM *c* — включение (*c* — символ 4) или отключение (*c* — символ-цифра 0) устройства для подачи листовой бумаги, которое поставляется факультативно. Если его у Вас нет, то не пользуйтесь этой командой. Команда отменяет установку переключателя 2-2;

BEL — звонок (вырабатывается короткий звуковой сигнал);

2) *команды управления выводом данных:*

CR — возврат каретки (печатается строка из буфера принтера и печатающая головка возвращается в крайнее левое положение). Если переключатель 2-4 включен, то дополнительно осуществляется перевод строки;

CAN — отмена строки (текущая строка не печатается, но команды воспринимаются);

DEL — забой символа (предыдущий символ не печатается, но если он принадлежит команде, то принтером воспринимается);

3) *команды вертикального/горизонтального перемещения:*

FF — перевод формата (данные из буфера принтера печатаются на бумаге, и рулонная бумага прогоняется до начала следующей страницы в соответствии с ее текущей длиной, а бумажный лист просто выталкивается из принтера);

ESC C *n* — установка длины страницы (но не длины поля текста на странице) равной *n* строкам при текущем интервале между строками. Число *n* должно находиться в интервале 1 — 127. Началом страницы считается текущая строка;

ESC C NUL *n* — установка длины страницы (но не длины поля текста на странице) равной *n* дюймам (1 — 22). Началом страницы считается текущая строка;

ESC N *n* — установка межстраничного интервала равным *n* (1 — 127) строкам (если на странице помещается *m* строк, то поле текста будет состоять из *m—n* строк, за которым до начала следующей страницы будет пропущено *n* строк). Переключатель 2-3 выполняет аналогичную функцию, но задает стандартный интервал. Данная установка отменяется командами ESC O, ESC C и ESC C NUL;

ESC O — отмена разбиения на страницы, сделанного командой ESC N. Установка переключателя 2-3 также отменяется;

LF — перевод строки (печатается текущая строка из буфера принтера, и бумага перемещается вперед на одну строку с учетом текущего интервала между строками);

ESC 0 — установка 1/8-дюймового (3,175-мм) межстрочного интервала (0 — цифра в символьном виде);

ESC 1 — установка 7/72-дюймового (2,469-мм) межстрочного интервала;

ESC 2 — установка 1/6-дюймового (4,233-мм) межстрочного интервала;

ESC 3 *n* — установка *n*/216-дюймового межстрочного интервала (*n* — число в интервале 0 — 255);

ESC A *n* — установка *n*/72-дюймового межстрочного интервала (*n* число в интервале 0 — 85);

ESC J *n* — перевод строки на *n*/216 дюйма (*n* — число в диапазоне 0 — 255); возврат каретки не производится, и действующая установка межстрочного интервала не отменяется;

VT — вертикальная табуляция (бумага прогоняется до позиции следующей вертикальной табуляции для канала, установленного командой ESC /; если никакой канал не установлен, то используется канал 0; когда позиции вертикальной табуляции не установлены, бумага перемещается на одну строку);

ESC B *n1 n2* ... NUL — установка до 16 позиций вертикальной табуляции в терминах текущего межстрочного интервала. Числа *n1*, *n2*, ... (в диапазоне 1 — 255) перечисляются в порядке возрастания и указывают позиции вертикальной табуляции. Все позиции запоминаются для канала 0 (см. команду ESC b). Последующие изменения межстраничного интервала влияния на сделанную установку позиций вертикальной табуляции не оказывают. Команда ESC B NUL приводит к гашению текущей установки;

ESC b *c n1 n2* ... NUL — установка позиций вертикальной табуляции для канала *c* (0 — 7). В остальном команда аналогична ESC B;

ESC / *c* — выбор канала вертикальной табуляции (0 — 7). Все последующие команды VT используют этот канал табуляции;

ESC I *n* — установка левого поля равным *n* колонкам в терминах текущего шага;

ESC Q *n* — установка правого поля равным *n* колонкам в терминах текущего шага;

BS — возврат на шаг (печатается текущая строка из буфера принтера, после чего печатающая головка перемещается на шаг назад и устанавливается тем самым

на последний отпечатанный символ). Эта команда может быть выполнена несколько раз, вплоть до левого поля, после чего она игнорируется. Команда BS также игнорируется после команды ESC a 1, 2 или 3. Если BS непосредственно следует за псевдографическим символом, то правильное положение печатающей головки при выводе последующих данных не гарантируется. BS можно использовать для комбинирования (наложения) символов;

ESC e *n s* — установка величины приращения при горизонтальной или вертикальной табуляции. Если *n* = 0, то устанавливаются позиции горизонтальной табуляции с интервалом *s* пробелов. Максимальные значения *s*:

для нормального шага — 21;
для полусжатого шрифта — 25;
для сжатого шрифта — 36.

Если *n* = 1, то устанавливаются позиции вертикальной табуляции с интервалом *s* строк;

ESC f *n s* — горизонтальный/вертикальный пропуск. Если *n* = 0, то вставляется s пробелов (до 107). Если *n* = 1, то осуществляется перевод *s* строк;

HT — горизонтальная табуляция (печатающая головка перемещается вперед до следующей позиции горизонтальной табуляции). Стандартная установка соответствует приращению, равному 8 символам с нормальным шагом. Последующее изменение шага на позиции табуляции влияния не оказывает;

ESC D *n1 n2* ... NUL — установка до 32 позиций горизонтальной табуляции *n1*, *n2*, ... (в порядке возрастания и в диапазоне 1 — 137). Команда ESC D NUL отменяет все установки. Установка после включения питания или команды ESC @ соответствует каждому восьмому символу;

4) *команды выбора печатного стиля:*

ESC x <*n*> — выбор качества печати: режима Draft (*n* = 0) или NLQ (*n* = 1);

ESC k <*n*> — выбор шрифта Roman (*n* = 0) или Sans Serif (*n* = 1) в режиме NLQ;

ESC ! *n* — выбор модификаторов шрифта. Операндом *n* задается сумма номеров следующих устанавливаемых, совместимых между собой (см. табл. 5.21) модификаторов:

0 — нормальный шаг;
1 — полусжатый шрифт;
4 — сжатый шрифт;
8 — полужирный шрифт;
16 — двухпроходный шрифт;
32 — растянутый шрифт;
64 — курсив;
128 — подчеркивание.

ESC P — выбор нормального шага;

ESC M — установка печати полусжатым шрифтом;

SI — установка печати сжатым шрифтом;

ESC SI — то же;

DC2 — отмена печати сжатым шрифтом, установленной командой SI (ESC SI), клавишами или переключателем 1-1;

SO — установка печати растянутым шрифтом (только для текущей строки). Отменяется возвратом каретки или командой DC4;

ESC SO — то же;

DC4 — отмена печати растянутым шрифтом, заданной командой SO (ESC SO), однако действие команд ESC W и ESC ! не блокируется;

ESC W <*n*> — установка (*n* = 1) или отмена (*n* = 0) печати растянутым шрифтом;

ESC E — установка печати полужирным шрифтом;

ESC F — отмена печати полужирным шрифтом;

ESC G — выбор двухпроходного режима печати;

ESC H — отмена двухпроходного режима печати;

ESC S NUL — выбор режима верхнего индекса. Последующие символы при печати будут иметь высоту, составляющую 2/3 от нормальной высоты, и располагаться в верхней части строки. Вместо NUL можно использовать символ-цифру 0;

ESC S SOH — выбор режима нижнего индекса. Последующие символы при печати будут иметь высоту, составляющую 2/3 от нормальной высоты, и размещаться в нижней части строки. Вместо SOH можно использовать символ 1;

ESC T — отмена режима верхнего (нижнего) индекса;

ESC — <*n*> — включение (*n* = 1) или выключение (*n* = 0) режима подчеркивания;

ESC a *n* — выключка (выравнивание) в режиме NLQ. Допустимы следующие значения *n*:

0 — установка выключки с левой стороны;
1 — установка центрирования;
2 — установка выключки с правой стороны;
3 — установка полной выключки.

Стандартная установка соответствует *n* = 0. Полная выключка выполняется, когда буфер становится полным, или при получении команды CR, VT, LF или FF. Команды HT и BS не действуют, если *n* не равно 0. При *n* = 3 в пределах параграфа не должно быть команды CR;

5) *команды выбора кодовой страницы:*

ESC t *n* — выбор пары кодовых страниц. Допустимые значения *n*:

0 — выбор пары кодовых страниц для курсива;
1 — выбор пары кодовых страниц, соответствующих кодовой странице 437.

Действие переключателей 1-3 и 1-4 отменяется;

ESC 4 — выбор режима курсива. Команда допустима даже при установке других кодовых страниц, однако псевдографические символы курсивом печататься не будут. В режиме Draft наклонные символы печатаются с пониженной скоростью;

ESC 5 — отмена режима курсива;

ESC R *n* — выбор набора национальных символов. Допустимы следующие значения *n*:

0 — для США
1 — для Франции
2 — для Германии
3 — для Великобритании
4 — для Дании (I)
5 — для Швеции
6 — для Италии
7 — для Испании (I)
8 — для Японии (II)
9 — для Норвегии
10 — для Дании (II)
11 — для Испании (II)
12 — для Латинской Америки

ESC 6 — выбор второй кодовой страницы из пары;

ESC 7 — выбор первой кодовой страницы из пары;

6) *команды определения символов* (дополнительно см. следующий подпункт):

ESC & NUL *n m d1 d2* ... NUL NUL — загрузка в принтер матриц определенных пользователем символов с кодами от *n* до *m* (в диапазоне 58 — 63, или 3AH — 3FH). Числа *d1, d2,* ... определяют вид символов;

ESC : NUL NUL NUL — копирование стандартных матриц символов с кодами 58 — 63 из СППЗУ в ОЗУ принтера с тем, чтобы можно было затем определить новые символы;

ESC % *n* — выбор стандартного (*n* = 0) или определенного пользователем (*n* = 1) набора символов. Используется совместно с командой ESC & (перед ней или после нее);

7) *команды выбора графического режима* (дополнительно см. соответствующий подпункт):

ESC K *n1 n2* — выбор восьмиигольчатого режима с одинарной плотностью (60 точка/дюйм) с общим числом колонок *n*1 + *n*2x256;

ESC L *n1 n2* — выбор восьмиигольчатого графического режима с двойной плотностью (120 точка/дюйм) с общим числом колонок *n*1 + *n*2x256;

ESC Y *n1 n2* — выбор высокоскоростного восьмиигольчатого графического режима с двойной плотностью (120 точка/дюйм) с общим числом колонок *n*1 + *n*2x256;

ESC Z *n1 n2* — выбор восьмиигольчатого графического режима с учетверенной плотностью (240 точка/дюйм) с общим числом колонок *n*1 + *n*2x256;

ESC * *m n1 n2*— выбор восьмиигольчатого графического режима с номером *m* и общим числом колонок *n*1 + *n*2x256;

ESC ? *s m* — модификация восьмиигольчатого графического режима. Здесь *s* — одна из букв K, L, Y или Z, задающая модифицируемый восьмиигольчатый графический режим, а *m* — номер нового режима;

ESC ^ *m n1 n2*— выбор девятиигольчатого графического режима с одинарной ($m = 0$) или двойной ($m = 1$) плотностью и общим числом колонок $n1 + n2 \times 256$. Требуется два числа для каждой печатаемой колонки.

Для выдачи команды (последовательности команд) на принтер можно воспользоваться одним из следующих способов:

1) создать программным способом или в ряде случаев текстовым редактором файл с командами и вывести его на принтер DOS-командой COPY *file* PRN или TYPE *file* >PRN;

2) создать текстовым редактором командный файл с DOS-командой ECHO, в качестве аргумента которой указана последовательность команд, а стандартный вывод перенаправлен на принтер, и выполнить его. В ряде случаев можно выдать команду ECHO и в ответ на приглашение DOS. Например, для возврата на шаг достаточно задать DOS-команду

ECHO ◄Ctrl-H► >PRN

3) вывести команды на принтер непосредственно из программы.

В связи с тем что наряду с управляющими символами команды принтера зачастую содержат и числовые аргументы, наиболее общими и удобными способами являются первый (с разработкой программы) и третий.

Все установленные командами режимы при отключении питания сбрасываются.

Не забудьте после видимого окончания печати текста перевести принтер в режим OFF LINE (автономный), чтобы он смог отпечатать последнюю строку, если она не завершается символом CR.

Определение новых символов

Взамен имеющихся в выбранной кодовой странице символов с кодами 58 — 63 можно определить новые символы с произвольным графическим изображением. В результате этого стандартные символы становятся недоступными, а вместо них будут печататься новые символы. Для восстановления исходного состояния принтера нужно отключить и включить питание на нем либо выдать команду ESC % с нулевым операндом.

Определение нового символа включает следующие этапы:

1) *конструирование символа*;

2) *кодирование матрицы символа*;

3) *загрузку матрицы символа в ОЗУ принтера*.

Детали выполнения этих этапов зависят от того, для какого уровня качества печати определяется новый символ. Поэтому мы рассмотрим формирование нового символа отдельно для режимов Draft и NLQ.

Режим Draft. Этап *конструирования* сводится к представлению символа точками в матрице, состоящей из 11 колонок и 9 рядов (по ряду на иглу). Первая и последняя колонки не используются, так как служат для разделения символов. Кроме того, для каждого отдельного символа задействуются только восемь из девяти рядов, причем обычно — верхние. Другая возможность состоит в использовании восьми нижних рядов (для случая, когда символ или его часть требуется несколько опустить вниз). Исходя из сказанного типичная заготовка для матрицы имеет вид, показанный на рис. 5.16. Двойной линией отделен неиспользуемый ряд. Ряды и колонки пронумерованы.

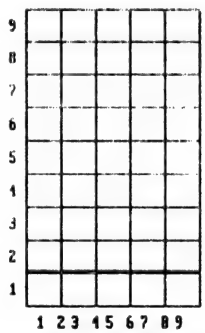


Рис. 5.16. Сетка для конструирования символов (режим Draft)

Точки можно ставить на пересечении каждого ряда и колонки. Единственное ограничение состоит в том, что никакие две точки не могут размещаться на пересечениях одного ряда и соседних колонок. Однако если допущено случайное перекрытие точек, то это не приведет к ошибке: просто одна из точек не будет печататься. Пример матрицы символа ↑ представлен на рис. 5.17.

При *кодировании матрицы символа* каждая колонка представляется числом. Для этого колонка матрицы рассматривается как однобайтовое двоичное число, в котором точка обозначает единицу, а ее отсутствие — нуль. Девятый ряд соответствует старшему, а второй — младшему разряду.

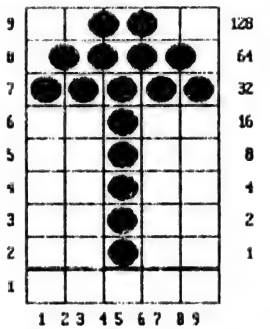


Рис. 5.17. Матрица символа ☂ для режима Draft

Затем каждое двоичное число переводится в десятичную систему счисления. Процесс представления колонки десятичным числом можно упростить, если использовать веса рядов сетки, записанные справа от матрицы. При этом достаточно сложить веса, которым соответствует точка в данной колонке. Для нашего примера получится следующий код матрицы:

32, 64, 32, 192, 63, 192, 32, 64, 32

Конструирование символа и кодирование матрицы для нижних восьми игл осуществляются аналогично, но при этом верхний ряд сетки не используется, а задействуется нижний.

Загрузка матрицы символа в ОЗУ принтера производится путем выдачи команд ESC :, ESC % и ESC & при условии, что принтер находится в режиме Draft. Команда ESC & в этом случае должна иметь формат

ESC & NUL *n m q c1 c2 ... c9* NUL NUL

где *n=m* — код нового символа (выбирается разработчиком в диапазоне 58 — 63);

q — число 128, если не задействуется нижний ряд, или число 0, если не задействуется верхний ряд сетки;

c1 c2 ... c9 — код матрицы символа.

Для рассматриваемого примера (если мы хотим новым символом заместить символ <, имеющий код 60) данная команда будет представлена следующим образом:

ESC & NUL 60 60 128 32 64 32 192 63 192 32 64 32 NUL NUL

Надо иметь в виду, что при выдаче команды числа должны быть закодированы в двоичной системе счисления, а не в ASCII.

Загрузить матрицу символа в ОЗУ принтера можно, в частности, следующей программой на языке Turbo Pascal:

```
Program LoadDraftUserSymbol;
Uses Printer;
Var
   c:string;
Begin
     Write(Lst, #27':'#0#0#0);                          {выдача команды ESC :}
     Write(Lst, #27'%'#1);                              {выдача команды ESC %}
     c:=  #27'&'#0#60#60#128+                  {формирование команды ESC &}
          #32#64#32#192#63#192#32#64#32#0#0;
     Write(Lst, c)                                      {выдача команды ESC &}
End.
```

После выполнения этой программы вместо символа будет печататься символ ☂.

Определить несколько новых символов можно одним из следующих способов:

1) для каждого символа выдать команду ESC & с уникальным кодом *n=m*;

2) выдать единственную команду ESC &, в которой *n* не равно *m*, *n* задает первый код символа, *m* задает последний код символа, а последовательности *q c1 c2 ... c9* специфицируются подряд для каждого символа.

Режим NLQ. При *конструировании* символа для этого режима можно задействовать 12 колонок и 18 рядов (рис. 5.18), так как печать будет осуществляться за два прохода печатающей головки. Точки на такой сетке можно представлять на каждом пересечении ряда и колонки без каких-либо ограничений, однако обычно слева и справа оставляют свободным один или два ряда для разграничения символов. В связи с тем что точки могут перекрываться, их принято обозначать

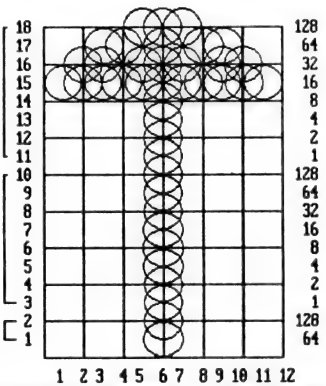


Рис. 5.18. Матрица символа ☂ для режима NLQ

незаштрихованными окружностями. В качестве примера на рисунке изображена матрица того же символа ☂.

Очевидно, столбец матрицы символа для режима NLQ невозможно представить двоичным числом в формате байта (требуется 18 разрядов). Поэтому каждый столбец *кодируется* триадой. Необходимое для этого разбиение показано в левой части рисунка, а в правой его части проставлены веса разрядов. Кодом матрицы для рассматриваемого примера будет являться следующая последовательность 36 чисел:

16, 0, 0, 48, 0, 0, 112, 0, 0, 80, 0, 0, 176, 0, 0, 255, 255,
192, 176, 0, 0, 80, 0, 0, 112, 0, 0, 48, 0, 0, 16, 0, 0, 0, 0

Загрузка матрицы в ОЗУ принтера, находящегося в режиме NLQ, производится аналогично режиму Draft, но команда ESC & должна иметь следующий формат:

ESC & NUL *n m* NUL 12 NUL *c1 c2 ... c36*

Здесь *n* и *m* имеют тот же смысл, что и раньше, число 12 указывает количество колонок, а числа *c1 c2 ... c36* являются кодом матрицы символа.

Требуемые команды могут быть выданы из программы на языке Turbo Pascal следующим образом:

```
Program LoadNLQUserSymbol;
Uses Printer;
Var
    c:string;
Begin
    Write(Lst, #27':'#0#0#0);
    Write(Lst, #27'%'#1);
    c:=#27'&'#0'<'#0#12#0+
      #16#0#0#48#0#0#112#0#0#80#0#0#176#0#0#255#255#192+
      #176#0#0#80#0#0#112#0#0#48#0#0#16#0#0#0#0#0;
    Write(Lst, c)
End.
```

Определенный описанным способом символ доступен как в шрифте Roman, так и в шрифте Sans Serif.

Имеется возможность определения (в том числе одних и тех же) символов как в режиме Draft, так и в режиме NLQ при одном и том же сеансе работы принтера.

Точечная графика

Графические команды принтера позволяют печатать на бумаге любое изображение, состоящее из расположенных в определенных местах точек. Оно формируется путем горизонтального перемещения печатающей головки и печати точек в нужных местах. В основном режиме задействуются только верхние 8 игл. Затем бумага протягивается на ширину строки (расстояние между девятой и второй иглами) и печатается следующая строка. Этот процесс может быть продолжен до тех пор, пока на бумаге не получится требуемое изображение.

Комбинации точек в каждой колонке печатаемой строки задаются последовательностью чисел, расположенных после команды выбора графического режима.

Разница между текстовым и графическим режимами состоит не в различных способах печати, а в том, как принтер управляется. В *текстовом режиме* определено соответствие между кодами символов и их представлением соответствующей матрицей. Поэтому прием кода символа принтером приводит к его вычерчиванию на бумаге. Для *графического режима* никакого аналогичного соответствия не вводится. Каждый полученный принтером однобайтовый код предписывает

отпечатать очередную колонку определенной комбинацией точек. Колонки кодируются полностью аналогично режиму Draft (верхние 8 игл).

Разрешающая способность принтера по вертикали определяется расстоянием между соседними иглами, которое составляет 1/72 дюйма. Конечно, можно было бы ее повысить путем печати строки за несколько проходов, протягивая бумагу перед каждым последующим проходом на небольшое расстояние. Это привело бы к уменьшению скорости печати (которая и так в графическом режиме мала) и к усложнению логики управления принтером. В модели EPSON LX-800 многопроходная печать в графическом режиме не реализована.

Разрешение принтера по горизонтали определяется только возможностями управления печатающей головкой и в принтере EPSON LX-800 может достигать 240 точка/дюйм.

В командах выбора графического режима указывается *количество колонок*, которые требуется отпечатать на данной строке. Коды колонок следуют непосредственно за командой. После отпечатывания заданного количества колонок принтер автоматически возвращается в текстовый режим. В каждой строке можно разместить почти 2000 колонок. Это число не может быть представлено в формате байта. Поэтому количество подлежащих печати колонок *n* задается двумя операндами *n1* и *n2*, где *n2* — целочисленное частное, а *n1* — остаток от деления *n* на 256.

Для перевода строки при печати изображений целесообразно использовать команду ESC A 8.

Принтер поддерживает четыре специализированных и одну универсальную команду (ESC *) выбора *восьмиигольчатого* графического режима, которые имеют следующий формат:

ESC K *n1 n2*
ESC L *n1 n2*
ESC Y *n1 n2*
ESC Z *n1 n2*
ESC * *m n1 n2*

Каждый графический режим обеспечивает свое горизонтальное разрешение, или *плотность печати* (табл. 5.25).

Таблица 5.25

Восьмиигольчатые графические режимы

Режим	Специализированная команда	m (для ESC *)	Горизонтальное разрешение, точка/дюйм
с одинарной плотностью	ESC K	0	60
с двойной плотностью	ESC L	1	120
высокоскоростной с двойной плотностью	ESC Y	2	120
с учетверенной плотностью	ESC Z	3	240
ЭЛТ 1	нет	4	80
графопостроитель	нет	5	72
ЭЛТ 2	нет	6	90

Режимы 4 и 6 обеспечивают согласование печатаемого изображения с изображением на экране дисплея. В этом случае отношение горизонтальной плотности к вертикальной на экране дисплея и на принтере будет одинаковым, т.е. изображение может быть передано без сжатия (растяжения). Режим 5 устанавливает горизонтальное разрешение равным вертикальному. В режимах 2 и 3 печатающая головка не может печатать две последовательных точки (в соседних колонках) одной иглой.

Для оперативного (без модификации всей программы) изменения графического режима можно использовать команду ESC ? *s m*. Она позволяет присвоить другой восьмиигольчатый графический режим одной из специализированных графических команд, заданной посредством *s* (K, L, Y или Z). Это приводит к изменению ширины печатаемого изображения без модификации высоты. Конечно, команда ESC ? должна быть выдана до специализированной графической команды.

Команда ESC ^ обеспечивает выбор *девятиигольчатого* графического режима с *одинарной* или *двойной* плотностью. Она используется редко.

Подлежащий печати рисунок обычно конструируется на миллиметровой бумаге. Нужно определиться в выборе графического режима в зависимости от потребной скорости печати (чем выше разрешение, тем она ниже) и качества изображения.

При *конструировании* рисунка нужно придерживаться следующих правил:

1) независимо от режима никакие точки не могут размещаться на горизонтальных линиях, а только по одной между ними;

2) в режиме с одинарной плотностью никакие точки не могут находиться на вертикальных линиях, а только по одной между ними;

3) в высокоскоростном режиме с удвоенной плотностью точки могут находиться и на вертикальных линиях, но никакие две точки не должны располагаться по горизонтали рядом;

4) в режиме с удвоенной плотностью точки могут находиться на вертикальных линиях без каких-либо ограничений;

5) в режиме с учетверенной плотностью действует правило 3, но для рядов рисунка лучше использовать не каждую строку миллиметровой бумаги, а через одну, чтобы примерно выдержать отношение между вертикальной и горизонтальной плотностью.

В качестве примера на рис. 5.19 представлен результат конструирования стрелки для печати в высокоскоростном режиме с двойной плотностью. Очевидно, такую стрелку можно разместить только на двух строках (8 рядов в одной и 4 ряда в другой строке). В верхней части рисунка проставлены коды колонок первой, в нижней части рисунка — второй строки. Такая стрелка может быть отпечатана следующей программой, составленной на языке Turbo Pascal:

```
Program PrintPointer;
Uses Printer;
Var
    c: string;
Begin
    Write(Lst, #27'Y'#41#0);                                    {установка режима}
    c:=#64#32#80#8#69#2#64#0#64#0+          {формирование первой строки}
      #64#0#64#0#32#0#16#0#8#0#8#0#8#0#8#0+
      #8#0#8#0#8#0#8#0# 184#64#32#16#8#5#2;
    Write(Lst, c);                                              {печать первой строки}
    Write(Lst, #27'A'#8);                          {установка межстрочного интервала}
    Writeln(Lst);                                               {переход к новой строке}
    Write(Lst, #27'Y'#41#0);                                    {установка режима}
    c:=#16#32#80#128#16#0#16#0#16+          {формирование второй строки}
      #0#16#0#16#0#32#0#64#0#128#0#128#0#128#0#128#0+
      #128#0#128#0#128#0#128#0#232#16#32#64#128#0#0;
    Writeln(Lst, c)                                             {печать второй строки}
End.
```

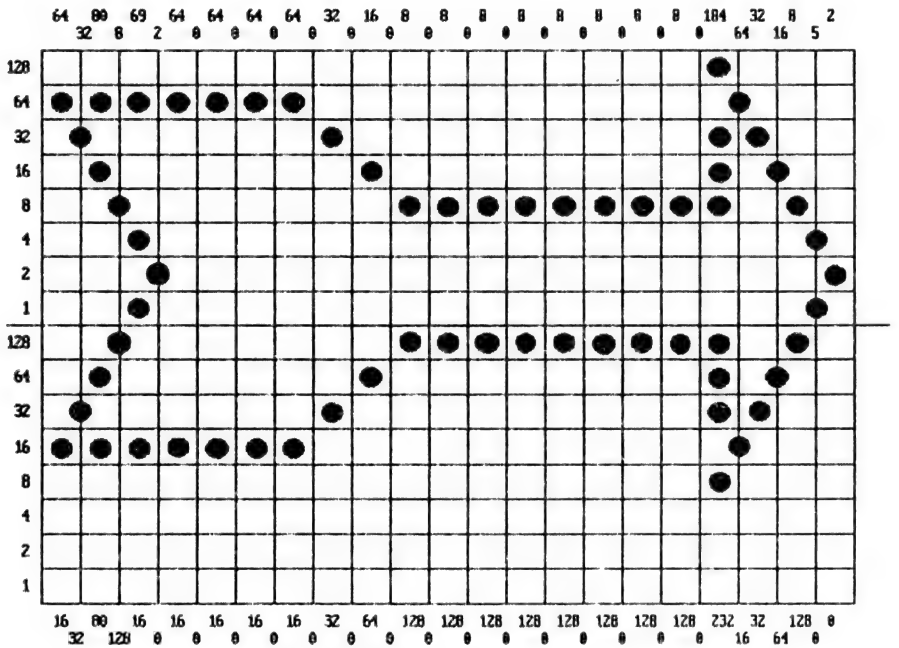


Рис. 5.19. Конструирование стрелки

Несмотря на то что графические команды принтера позволяют реализовать все его возможности, описанный метод подготовки иллюстраций весьма трудоемок. Поэтому им пользуются в основном разработчики графических программных продуктов, а также программисты при необходимости обеспечить высококачественный вывод результатов выполнения программы. Пользователю же на выбор предоставляется множество таких программных изделий (в частности, графических редакторов), которые существенно облегчают процесс формирования и печати изображений.

5.12. Размещение информации на магнитных дисках

Обсуждаемые в данном подразделе вопросы уже неоднократно (но отнюдь не полностью) затрагивались нами в пунктах 2.4.1 и 2.4.2, а также в подразделах 5.2 и 5.6. Теперь настало время обобщить разрозненные сведения и подробно изучить структуру системной области гибких, а также жестких дисков.

Пользователь-непрофессионал этот материал может пропустить.

В понятие **формата носителя информации** (здесь — магнитного диска) входят структура информации на нем и способы адресации элементов этой структуры.

Принято различать *физический* и *логический* форматы дисков, которые мы сейчас и рассмотрим.

5.12.1. Физический формат диска

На данном уровне рассмотрения принимается во внимание то, что информация размещается в секторах фиксированной длины, расположенных на концентрических дорожках поверхности диска (см. рис 2.6 в п. 2.4.1). Число секторов на дорожке конкретного носителя также постоянно.

Диск может иметь одну или две рабочих поверхности. НЖМД содержит один или более дисков (часто — два). Однако обычно под жестким диском понимают весь пакет магнитных дисков. Число дорожек на поверхности дискеты (число цилиндров), как правило, составляет 40 или 80. Жесткие диски могут иметь 305, 614 или другое число цилиндров. Количество рабочих поверхностей и цилиндров является *аппаратной* характеристикой дисковода. Тем не менее программист, непосредственно управляющий контроллером привода, может (в рамках допустимого диапазона) изменить число цилиндров.

Число секторов на дорожке задается *программно* (драйвером устройства). Стандартными для различных НГМД являются величины 8, 9, 15 и 18. Жесткие диски обычно имеют 17, 32 или более секторов на дорожке. Управляя контроллером привода непосредственно, можно изменить приведенные значения.

Каждый сектор состоит из *поля данных* и *поля служебной информации*, ограничивающей и идентифицирующей его. Размер сектора (точнее — емкость поля данных) также устанавливается драйвером. Пользовательский интерфейс DOS поддерживает единственный размер сектора — 512 байт. BIOS же непосредственно предоставляет возможности работы с секторами размером 128, 256, 512 или 1024 байт. Если управлять контроллером непосредственно, то можно обрабатывать и секторы с другими размерами.

Число цилиндров, секторов на дорожке и размер сектора устанавливаются при форматировании диска и в последующем без повторной его инициализации изменены быть не могут.

Заметим, что изощренным программированием контроллера дисковода можно добиться форматирования диска с различными числом секторов на дорожках и размерами секторов. Такая техника часто используется разработчиками ПО с целью защиты программных продуктов от несанкционированного копирования. Например, на одной из дорожек можно создать дополнительный сектор, содержащий ключ (пароль). Программный продукт при запуске проверяет его наличие и реагирует на его отсутствие прекращением работы. Системные средства копирования дисков и файлов оставят этот сектор незамеченным, вследствие чего несанкционированная копия программного изделия функционировать не будет. Описанная техника широко применялась для 8-секторных дискет, где остается место для девятого сектора. Имеются и такие системы защиты, которые записывают ключи в межсекторные промежутки.

Физический адрес сектора на диске представляется триадой [t-h-s], где t — *номер цилиндра* (дорожки на поверхности диска), h — *номер рабочей поверхности диска* (магнитной головки), а s — *номер сектора на дорожке*. Номер цилиндра t лежит в диапазоне 0..T—1, где T — *количество цилиндров*. Номер рабочей поверхности диска h принадлежит диапазону 0..H—1, где H — *число магнитных головок* в приводе. Номер сектора на дорожке s указывается в диапазоне 1..S, где S — *количество секторов на дорожке*. Например, триада [1-0-2] адресует сектор 2 на дорожке 0 (обычно верхняя рабочая поверхность) цилиндра 1. В дальнейшем мы будем пользоваться именно этими обозначениями.

Напомним, что обмен информацией между ОЗУ и дисками физически осуществляется только секторами.

5.12.2. Логический формат гибкого и жесткого диска

Каждая *дискета* обычно рассматривается DOS как единственный *логический диск*. Однако на ПЭВМ серии ЕС это не так. Дело в том, что НГМД этих компьютеров поддерживают 720-Кбайт дискеты. Вместе с тем драйвер дисководов обслуживает только дискеты емкостью 360 Кбайт. Указанное несоответствие не выдерживает никакой критики и вынуждает системных программистов разрабатывать, а пользователей — с муками подключать внешние несистемные драйверы, чтобы обеспечить вместо поддержки 360-Кбайт дискет (или наряду с этим) обслуживание 2x360-Кбайт и/или 720-Кбайт ГД. В первом случае на одном физическом приводе создается два логических дисковода, а на каждой дискете — два логических диска.

Жесткий диск организуется иначе. Он может быть подразделен на несколько *разделов*, используемых различными ОС. Максимальное число разделов равно четырем. Собственно DOS может использовать один или два раздела. Первый из них должен быть *первичным разделом* DOS, второй — может быть только *расширенным разделом DOS* (см. описание команды FDISK в п. 5.6.1). В первичном разделе DOS может быть сформирован только один логический диск, а в расширенном — любое их количество. Каждый логический диск «управляется» своим логическим приводом.

На логическом уровне считается, что секторы логического диска имеют непрерывную нумерацию от 0 до N—1, где N = T x H x S — *количество секторов на диске*. Соответствие между физическим адресом сектора и его **логическим номером** n (для дискет) определяется следующей формулой:

$$n = (t \times H \times S) + (h \times S) + s - 1 \qquad (5.1)$$

Таким образом, сначала (начиная с нуля) нумеруются секторы на нулевой дорожке нулевой поверхности, затем — на нулевой дорожке первой поверхности и т.д. После перенумерации секторов на нулевых дорожках всех поверхностей описанный процесс повторяется для первой и всех последующих дорожек.

Например, для 360-Кбайт дискеты секторам присваиваются логические номера в порядке, показанном на рис. 5.20.

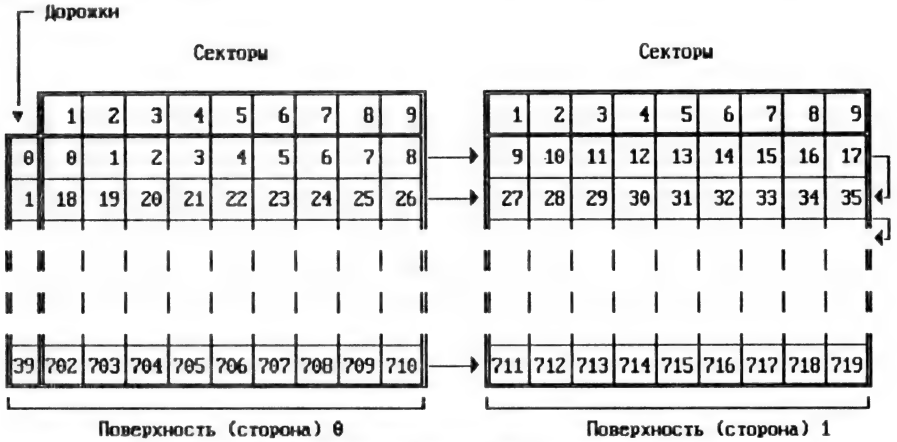


Рис. 5.20. Логическая нумерация секторов на дискете

Каждому логическому диску на винчестере соответствует своя *(относительная) логическая нумерация*. Физическая же адресация жесткого диска *сквозная*. В формуле (5.1) это нужно учитывать, вычитая из вычисленного n *абсолютный логический номер* первого сектора данного логического диска, чтобы получить смещение в секторах относительно его начала (т.е. *относительный логический номер*).

Формулу (5.1) можно использовать и в обратном направлении для определения физического адреса сектора по его логическому номеру:

$$t = n \;/\; (H \times S) \qquad (5.2)$$

$$h = (n - (t \times H \times S)) \;/\; S \qquad (5.3)$$

$$s = n - ((t \times H \times S) + (h \times S)) + 1 \qquad (5.4)$$

Здесь символ / обозначает целочисленное деление.

Приведенные выше формулы для дискет с единственным логическим диском можно использовать без каких-либо поправок. Для жестких дисков [как и с формулой (5.1)] дело обстоит несколько иначе: полученные t, h и s нужно увеличить на значения, соответствующие задействованной в других целях предшествующей области дискового пространства.

Версии DOS до 4.0 поддерживают логические диски, емкость которых не превышает 32 Мбайт. Это связано с тем, что для указания логического номера сектора используется 16-разрядное слово (512 байт x 2^16 = 32 Мбайт). Однако имеются драйверы, снимающие данное ограничение за счет укрупнения секторов на логическом уровне. В DOS 4.0 для указания логического номера сектора используется 32 разряда (двойное слово), так что в принципе можно адресовать 512 байт x 2^32. При снятии этого ограничения начинает играть роль другой фактор: размер кластера в сочетании с числом двоичных разрядов, используемых для указания кластера (см. подпункт «Таблица размещения файлов»).

Логическое дисковое пространство любого логического диска делится на две области (рис. 5.21): *системную область* и *область данных*.

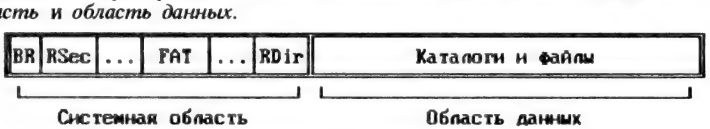


Рис. 5.21. Логическая структура логического диска

Системная область логического диска создается и инициализируется при форматировании, а в последующем обновляется при манипулировании файловой структурой. **Область данных** логи-

ческого диска содержит файлы и каталоги, подчиненные корневому. Она в отличие от системной области доступна через пользовательский интерфейс DOS.

Системная область состоит из следующих, расположенных в логическом адресном пространстве подряд, компонентов:

1) *загрузочной записи* (BR — Boot Record);
2) *зарезервированных секторов* (RSec — Reserved Sector);
3) *таблицы размещения файлов* (FAT — File Allocation Table);
4) *корневого каталога* (RDir — Root Directory).

BR находится в секторе с физическим адресом [0-0-1] (для дискеты) и содержит *блок параметров диска* (DPB — Disk Parameter Block), а также *системный загрузчик* (SB — System Bootstrap). Сектор, содержащий BR, называется *стартовым*.

За BR могут располагаться несколько **RSec**, используемых DOS. Стартовый сектор логического диска с BR относится к числу RSec. Обычно это единственный RSec на логическом диске.

FAT является очень важной информационной структурой. Она представляет собой карту (образ) области данных, в которой описывается состояние каждого кластера и связываются в цепочку принадлежащие одному файлу (некорневому каталогу) кластеры. *Кластер* — это минимальная единица дисковой памяти, выделяемая файлу (или некорневому каталогу). Каждый из них занимает целое число кластеров. Последний кластер при этом может быть задействован не полностью. Кластер представляет собой один или несколько смежных секторов в логическом дисковом адресном пространстве (точнее — только в области данных). На дискетах кластер занимает один или два сектора, а на жестких дисках — обычно четыре или восемь секторов. Логическое разбиение области данных на кластеры как совокупности секторов взамен использования одиночных секторов имеет следующий смысл:

— уменьшается возможная фрагментация файлов;
— уменьшается размер FAT, а следовательно, и объем системной области логического диска;
— ускоряется доступ к файлу, так как в несколько раз сокращается длина цепочек фрагментов дискового пространства, выделенных для него.

Однако слишком большой размер кластера ведет к неэффективному использованию области данных, особенно в случае большого количества маленьких файлов.

В связи с тем, что FAT используется при доступе к диску очень интенсивно, она обычно загружается в ОЗУ (в буфера ввода-вывода или кэш) и остается там настолько долго, насколько это возможно.

В связи с чрезвычайной важностью FAT она обычно хранится в двух идентичных экземплярах (за исключением FAT виртуального диска), второй из которых непосредственно следует за первым. Обновляются копии FAT одновременно. Используется же только первый экземпляр. Если он по каким-либо причинам окажется разрушенным, то произойдет обращение ко второму экземпляру.

RDir является корнем древовидной файловой структуры логического диска и не может быть удален никакими средствами. В связи с тем, что память под RDir выделяется статически, имеется ограничение количества содержащихся в нем файлов и подкаталогов.

Некорневые каталоги в области данных размещаются аналогично файлам (динамически), вследствие чего на их длину ограничение не накладывается.

Характеристики используемых физических и логических форматов дискет приведены в табл. 5.26. Форматы SS/QD и DS/QD DOS непосредственно не поддерживаются. Первый из них используется в ПЭВМ ИСКРА-1030, а второй — в ПЭВМ ЕС. Дополнительно к приведенным система обслуживает несколько форматов 8-дюймовых дискет, но сейчас в ПЭВМ они не применяются. Наибольшее распространение в настоящее время получили форматы с емкостью 360 Кбайт, 1,2 Мбайт, 720 Кбайт (3,5 дюйма) и 1,44 Мбайт.

В BR и FAT хранится *дескриптор носителя*, чтобы можно было быстро определить формат логического диска. Однако он обладает определенной неоднозначностью.

Никакой информации о *скрытых* (hidden) *секторах* автор, к сожалению, не имеет.

Особенности формата логического диска на винчестере состоят в следующем:

— увеличен размер кластера;
— увеличены размеры FAT и RDir;
— элементы FAT могут быть 16-битными;
— максимальное число элементов корневого каталога обычно составляет 512;
— дескриптором носителя является F8H.

На *жестком диске* имеется односекторная **главная загрузочная запись** (MBR — Master Boot Record), содержащая *внесистемный загрузчик* (NSB — Non-System Bootstrap), а также *таблицу разделов* (PT — Partition Table) и имеющая физический адрес [0-0-1]. Таким образом, в стартовом секторе физического жесткого диска находится не BR, а MBR.

PT описывает размещение и характеристики имеющихся на винчестере разделов.

NSB служит для копирования в ОЗУ SB из BR логического диска в активном разделе и передачи на него управления, что осуществляется при загрузке ОС.

Вслед за MBR размещаются *разделы* (см. рис. 5.14 в п. 5.6.1).

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

Таблица 5.26

Стандартные форматы дискет

Диаметр, дюймов	5,25	5,25	5,25	5,25	5,25	5,25	5,25	3,5	3,5
Условное обозначение формата	SS/DD	SS/DD	DS/DD	DS/DD	SS/QD	DS/QD	DS/HD	DS/QD	DS/HD
Типы дискет	SS/DD, DS/DD	SS/DD, DS/DD	DS/DD	DS/DD	SS/DD, DS/DD	DS/DD	DS/HD	DS/QD	DS/HD
Число рабочих поверхностей	1	1	2	2	1	2	2	2	2
Число цилиндров	40	40	40	40	80	80	80	80	80
Число секторов на дорожке	8	9	8	9	9	9	15	9	18
Размер сектора, байт	512	512	512	512	512	512	512	512	512
Число секторов в кластере	1	1	2	2	2	2	1	2	1
Число зарезервированных секторов	1	1	1	1	1	1	1	1	1
Размер FAT, секторов	1	2	1	2	2	3	7	3	9
Число копий FAT	2	2	2	2	2	2	2	2	2
Размер RDir, секторов	4	4	7	7	7	7	14	7	14
Максимальное число элементов в RDir	64	64	112	112	112	112	224	112	224
Число секторов на дискете	320	360	640	720	720	1440	2400	1440	2880
Дескриптор носителя, H	FE	FC	FF	FD	FC	F9	F9	F9	F0
Число скрытых секторов	0	0	0	0	0	0	0	0	0
Емкость, Кбайт	160	180	320	360	360	720	1200	720	1440

Расширенный раздел DOS можно рассматривать как жесткий диск в миниатюре: он содержит **вторичную MBR** (SMBR — Secondary MBR), в состав которой вместо PT входит *таблица логического диска* (LDT — Logical Disk Table), ей аналогичная. LDT описывает размещение и характеристики раздела, содержащего единственный логический диск, а также может специфицировать следующую SMBR. Следовательно, если в расширенном разделе DOS создано m логических дисков, то он содержит m SMBR, связанных в список. Каждый элемент этого списка описывает соответствующий логический диск и ссылается (кроме последнего) на следующий элемент списка.

Теперь мы переходим к детальному рассмотрению форматов компонентов системной области дисков. Подчеркнем, что если поле является числовым, а также занимает слово или двойное слово, то обычно младший байт слова (младшее слово двойного слова) размещается на диске первым. Это связано с удобством его обработки в ОЗУ после считывания.

Структура загрузочной записи

BR является самой первой на логическом диске (на дискете — имеет физический адрес [0-0-1]). Она состоит, как мы уже знаем, из двух частей — DPB и SB. DPB служит для идентификации физического и логического форматов логического диска, а SB играет существенную роль в процессе загрузки DOS (см. пп. 5.2.1 и 5.2.4). Первые два байта BR занимает команда безусловного перехода JMP на SB. Третий байт содержит код 90H (NOP — нет операции). Далее располагается восьмибайтовый системный идентификатор, включающий информацию о фирме-разработчике и версии DOS (например, IBM 3.3). Затем следует DPB, а после него — SB. DPB, создаваемые DOS 3.3 и DOS 4.0, различны (табл. 5.27 и 5.28). Другие же системные области диска в этих версиях DOS различий не имеют (но допустимы дополнительные значения в некоторых полях).

Одна из двух возможных структур BR определяется по содержимому байта со смещением 26H (для DOS 4.0 он хранит код 29H).

В новой BR увеличен размер ряда полей и присутствуют дополнительные поля. Поле метки тома дублирует соответствующий элемент корневого каталога. В качестве имени файловой системы (смещение 36H) может быть задано FAT12, FAT16 или HPFS с дополнением пробелами справа. Первые две системы являются неотъемлемой частью DOS и различаются только разрядностью номера кластера (12 или 16), от чего зависит структура FAT. HPFS — это дополнительная высокопроизводительная файловая система, доступная в OS/2.

Таблица 5.27

Структура BR в DOS 3.3

Смещение поля, байт	Длина поля, байт	Содержимое поля
00H(0)	3	безусловный переход на начало SB
03H(3)	8	системный идентификатор
0BH(11)	2	размер сектора, байт
0DH(13)	1	число секторов в кластере
0EH(14)	2	число зарезервированных секторов
10H(16)	1	число копий FAT
11H(17)	2	максимальное число элементов RDir
13H(19)	2	число секторов на логическом диске
15H(21)	1	дескриптор носителя
16H(22)	2	размер FAT, секторов
18H(24)	2	число секторов на дорожке
1AH(26)	2	число рабочих поверхностей
1CH(28)	2	число скрытых секторов
1EH(30)		SB
1FEH(510)	2	сигнатура (слово AA55H)

Таблица 5.28

Структура BR в DOS 4.0

Смещение поля, байт	Длина поля, байт	Содержимое поля
00H(0)	3	безусловный переход на начало SB
03H(3)	8	системный идентификатор
0BH(11)	2	размер сектора, байт
0DH(13)	1	число секторов в кластере
0EH(14)	2	число зарезервированных секторов
10H(16)	1	число копий FAT
11H(17)	2	максимальное число элементов RDir
13H(19)	2	число секторов на логическом диске, если его размер не превышает 32 Мбайт; иначе - 0000H
15H(21)	1	дескриптор носителя
16H(22)	2	размер FAT, секторов
18H(24)	2	число секторов на дорожке
1AH(26)	2	число рабочих поверхностей
1CH(28)	4	число скрытых секторов
20H(32)	4	число секторов на логическом диске, если его размер превышает 32 Мбайт
24H(36)	1	тип логического диска (00H - гибкий, 80H - жесткий)
25H(37)	1	пусто (резерв)
26H(38)	1	маркер с ходом 29H
27H(39)	4	серийный номер тома
2BH(43)	11	метка тома
36H(54)	8	имя файловой системы
3EH(62)		SB
1FEH(510)	2	сигнатура (слово AA55H)

Структура каталога

Корневой и некорневые каталоги имеют одинаковую структуру. Но RDir в отличие от других каталогов расположен в системной области логического диска, имеет ограниченный размер и не может быть фрагментированным.

Каталог состоит из последовательности 32-байт *элементов*. Каждый элемент каталога описывает входящий в него файл или каталог, содержит метку тома или является свободным. Отметим, что вся служебная информация о файле (каталоге), за исключением исчерпывающих сведений о его размещении в области данных логического диска, находится в одном из элементов того каталога, в котором он логически содержится. Сам же файл хранит только данные.

Элемент каталога состоит из восьми *полей* (табл. 5.29). Детализируем содержимое большинства из них.

Поле имени содержит имя файла (каталога), дополненное при необходимости пробелами справа. Код в первом байте этого поля определяет описываемый данным элементом объект или состояние данного элемента каталога. Заметим, что тип объекта здесь задается лишь частично.

Таблица. 5.29

Структура элемента каталога

Смещение поля, байт	Длина поля, байт	Содержимое поля
00H(0)	8	имя файла (каталога)
08H(8)	3	расширение имени файла (каталога)
0BH(11)	1	атрибуты файла
0CH(12)	10	резерв (для будущих расширений)
16H(22)	2	время создания файла (каталога)
18H(24)	2	дата создания файла (каталога)
1AH(26)	2	номер первого кластера файла (каталога)
1CH(28)	4	размер файла (каталога), байт

Для полной его идентификации привлекается и поле атрибутов. Содержимое первого байта поля имени интерпретируется следующим образом:

00H — элемент каталога еще ни разу не использовался. Это означает, что все последующие элементы являются пустыми, а следовательно, можно прекратить поиск по каталогу с уверенностью в том, что весь каталог просмотрен. Такая техника сокращает время поиска в каталогах, но элементы, по каким-либо причинам оказавшиеся размещенными после первого пустого элемента, становятся недоступными;

05H — первый символ имени файла (каталога) имеет код E5H, т.е. является символом *σ*. Такая перекодировка используется потому, что код E5H применяется по другому назначению;

2EH — элемент каталога описывает данный каталог (ссылка на себя). Число 2EH является кодом символа «точка». Если и второй байт поля имени содержит код 2EH, то элемент каталога описывает родительский каталог (..) данного каталога. Остальные байты полей имени и расширения содержат пробелы;

E5H — элемент использовался файлом (каталогом), но уже освобожден. Этот код записывается при удалении существующего файла (каталога), но больше никакие изменения в элемент каталога не вносятся.

Любое другое содержимое первого байта поля имени интерпретируется как код первого символа в имени файла (каталога).

Поле расширения хранит расширение имени файла (реже — каталога), дополненное при необходимости пробелами справа (аналогично имени).

Имя и расширение можно сформировать практически из любых символов, но пользовательский интерфейс DOS налагает свои ограничения (см. подпункт «Файлы» в п. 5.2.2). Напомним, что строчные буквы при вводе автоматически перекодируются в прописные. Это относится и к специфическим буквам национальных алфавитов, погруженных в стандартные кодовые страницы. Если же используемая Вами версия DOS непосредственно не поддерживает кириллицу, то о новом коде русской буквы Вам узнать заранее (до создания файла и каталога) не удастся. Чтобы избежать неприятностей, используйте для именования файлов (каталогов) только «надежные» в этом плане символы в соответствии с п. 5.2.2. Используя низкоуровневые редакторы диска, можно записать в поля имени и расширения любые символы, по крайней мере, удивив тем самым других пользователей.

В *поле атрибутов* каждый бит задает определенный атрибут файла (см. п. 5.2.2) или описывает представляемый данным элементом каталога объект. Структура этого поля показана в табл. 5.30. Если байт атрибутов содержит код 08H, то поля имени и расширения представляют метку тома. В случае, когда в байте атрибутов записан код 10H, считается, что данный элемент каталога описывает каталог.

Поле времени содержит время создания (последней модификации) файла или время создания каталога. Код в этом поле рассматривается как целое без знака, полученное по следующей формуле:

$$\text{часы} \times 2048 + \text{минуты} \times 32 + \text{секунды} / 2$$

Аналогично *поле даты* содержит дату создания (последней модификации) файла или дату создания каталога, которая представляется целым беззнаковым числом, полученным по формуле

$$(\text{год} - 1980) \times 512 + \text{месяц} \times 32 + \text{день}$$

Следовательно, календарь поддерживается только на 1980 — 2108 гг.

Таблица. 5.30

Структура поля атрибутов файла

Биты	Маска	Значение
7 6 5 4 3 2 1 0		
. . . . 1	01H	файл имеет атрибут R
. . . . . . 1 .	02H	файл имеет атрибут H
. . . . . 1 . .	04H	файл имеет атрибут S
. . . . 1 . . .	08H	элемент описывает метку тома
. . . 1	10H	элемент описывает каталог
. . 1	20H	файл имеет атрибут A
. . . .		резерв
. . . .		резерв

Поле со смещением 1AH содержит *номер первого кластера*, принадлежащего файлу (каталогу). Список остальных кластеров хранится в FAT. Для файлов, которым не выделено места (например, для открытых и сразу закрытых файлов), это поле хранит код 0000H.

Поле размера содержит длину файла (каталога), измеряемую в байтах. Для текстовых файлов заданная таким образом длина иногда не совпадает с реальной длиной, определяемой по первому символу EOF в файле. Напомним: это происходит потому, что для повышения быстродействия некоторые текстовые редакторы осуществляют обмен с диском на логическом уровне не байтами (символами), а логическими записями, содержащими несколько байтов. Если последняя логическая запись является неполной, то реальная длина файла будет меньшей, чем задано в поле размера. К неприятностям такая несогласованность обычно не приводит, так как DOS позволяет интерпретировать оба признака конца файла (переключателями /A и /B команды COPY). Ситуация, когда длина текстового файла является большей, чем указано в элементе каталога, считается ошибочной. Максимально возможный размер файла определяется DOS по содержимому FAT, т.е. исходя из наличия свободного пространства на диске.

Структура таблицы размещения файлов

FAT является образом области данных логического диска и содержит исчерпывающую информацию об использовании, а также о состоянии каждого кластера. В частности, она хранит сведения о размещении каждого файла и некорневого каталога.

FAT состоит из последовательности **элементов**, каждый из которых, за исключением первых двух, описывает соответствующий кластер.

Первые два элемента (с номерами 0 и 1) хранят копию дескриптора носителя (самый первый, т.е. младший, байт), дополненную до требуемого количества байтов двоичными единицами (FFH). Между остальными элементами FAT и кластерами области данных устанавливается позиционное соответствие: m-й (m=2,...,M+1, где M — *число кластеров в области данных*) элемент FAT описывает m-й кластер области данных. Последние нумеруются, начиная с двух, а не с нуля, для удобства. Кластеры располагаются в логическом дисковом пространстве последовательно, в порядке возрастания их номеров и покрывают всю область данных.

В зависимости от M для хранения элемента FAT отводится 12 или 16 битов (разрядов). 16-разрядный формат используется, только если число секторов превышает 20740 (5104H), что соответствует емкости логического диска в 10 Мбайт. Поэтому о таком формате можно говорить только применительно к достаточно большому логическому диску на винчестере. 12-разрядный формат FAT используется для всех дискет и небольших логических дисков на винчестере. С целью экономии дискового пространства два соседних 12-разрядных элемента FAT размещаются в трех байтах, однако такое решение усложняет доступ к ним.

Каждый m-й элемент FAT содержит либо номер следующего кластера, принадлежащего файлу (некорневому каталогу), либо *специальный* код. В качестве специальных используются следующие значения:

(0)000H — кластер свободен;

(F)FF0H..(F)FF6H — кластер зарезервирован для использования DOS;

(F)FF7H — кластер является дефектным;

(F)FF8H..(F)FFFH — кластер является последним в файле (некорневом каталоге).

Наличие любого другого кода означает, что данный кластер принадлежит какому-либо файлу (некорневому каталогу), а сам код является номером следующего кластера в этом файле (некорневом каталоге).

Пример фрагмента FAT и ее связи с каталогом приведен на рис. 5.22. Легко видеть, что файл MYFILE.TXT занимает кластеры 6, 7, 14, 15, 17 и 18.

Вновь создаваемым или пополняемым файлам (каталогам) выделяются ближайшие к началу FAT, но еще свободные элементы и соответствующие им кластеры.

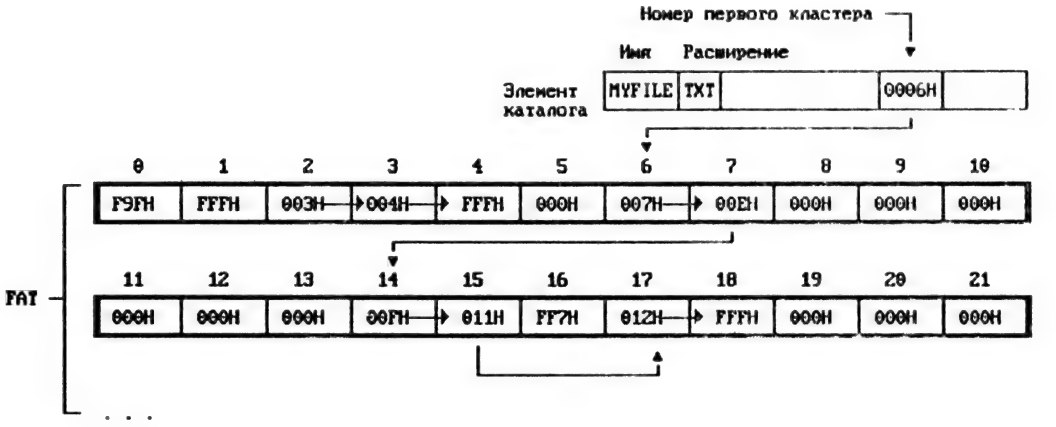


Рис. 5.22. Фрагмент FAT

В случае удаления файла все занятые им элементы FAT обнуляются. Тем не менее если файл не был фрагментирован, то имеются все шансы восстановить его, так как номер первого кластера файла в элементе каталога сохраняется. Конечно, занимаемые удаленным файлом кластеры не должны еще использоваться другими файлами.

Обычно ни программист, ни тем более пользователь ПЭВМ не испытывает потребности в непосредственном доступе к элементам FAT. Однако, если на диске возникли логические или физические дефекты и никакие системные средства не смогли восстановить потерянную в результате этого информацию, квалифицированный программист должен попытаться сделать все возможное путем написания специальной программы. Эта программа, естественно, должна напрямую обращаться к компонентам системной области диска, в частности к FAT.

Если FAT имеет 12-разрядный формат, то получить содержимое m-го элемента FAT поможет следующий алгоритм:

1) умножить m на 1,5 (каждый элемент FAT занимает полтора байта);
2) результат умножения (без дробной части) использовать как смещение в FAT для чтения слова (двух байтов);
3) если m было четным, то выделить из прочитанного слова первые 12 бит, если нечетным — последние.

Для FAT в 16-разрядном формате эта же задача решается гораздо проще:

1) умножить m на 2 (каждый элемент FAT занимает 2 байта);
2) использовать результат умножения как смещение в FAT для чтения слова.

Чтобы получить логический номер сектора, являющегося первым в кластере с заданным номером m, следует:

1) вычесть 2 из m;
2) умножить результат на количество секторов в кластере;
3) определить количество секторов, занимаемых системной областью логического диска;
4) сложить полученные по пп. 2 и 3 величины.

Технически доступ к файлу на диске осуществляется DOS следующим образом:

1) отыскивается соответствующий элемент каталога;
2) по содержимому FAT определяется номер требуемого кластера (если файл читается не с начала или его считывание продолжается);
3) номер кластера преобразуется в логический номер сектора;
4) выдается прерывание 25H для прямого чтения сектора с диска, по которому в работу включается драйвер НМД;
5) драйвер преобразует логический номер сектора в его физический адрес [см. формулы (5.2) — (5.4)] и обеспечивает управление приводом по считыванию требуемого сектора, содержимое которого записывается в предназначенный для этого буфер в ОЗУ.

Структура главной загрузочной записи и таблицы разделов

MBR делится на две логические части — NSB и PT. NSB обеспечивает загрузку DOS или какой-либо другой ОС с логического диска в активном разделе путем анализа содержимого PT.

PT находится в конце MBR и содержит четыре строки в соответствии с максимальным количеством разделов на винчестере. Каждую строку PT назовем **описателем раздела**.

Структура MBR представлена в табл. 5.31.

Таблица 5.31

Структура MBR

Смещение поля, байт	Длина поля, байт	Содержимое поля	
000H(0)	446	MSB	
1BEH(446)	16	описатель первого раздела	PT
1CEH(462)	16	описатель второго раздела	
1DEH(478)	16	описатель третьего раздела	
1EEH(494)	16	описатель четвертого раздела	
1FEH(510)	2	сигнатура (слово AA55H)	

Описатель раздела специфицирует размещение и характеристики соответствующего раздела жесткого диска (табл. 5.32).

Таблица 5.32

Структура описателя раздела

Смещение поля, байт	Длина поля, байт	Содержимое поля
00H(0)	1	статус раздела
01H(1)	1	номер поверхности винчестера, на которой начинается раздел
02H(2)	2	номера цилиндра (6 бит) и сектора (10 бит) на дорожке, с которых начинается раздел
04H(4)	1	системный код
05H(5)	1	номер поверхности винчестера, на которой заканчивается раздел
06H(6)	2	номера цилиндра (6 бит) и сектора (10 бит) на дорожке, которыми заканчивается раздел
08H(8)	4	логический номер начального сектора раздела
0CH(12)	4	размер раздела, секторов

Статус раздела может иметь два значения: 00H — для пассивного раздела или 80H — для активного раздела (из него будет загружаться ОС).

Значения *полей описателя раздела со смещениями 02H и 06H* вычисляются по следующей формуле:

$$(t \times 64) + s$$

Возможны следующие *системные коды*:

0 — раздел не создан (неизвестный раздел);

1 — первичный раздел DOS или раздел OS/2 с 12-битной FAT;

2 — раздел с CP/M;

3 — раздел с Xenix;

4 — первичный раздел DOS или раздел OS/2 с 16-битной FAT;

5 — расширенный раздел DOS;

6 — первичный раздел DOS 4.0 или раздел OS/2 1.10 размером более 32 Мбайт (это единственный идентификатор «большого» раздела);

7 — раздел для HPFS (OS/2).

Логический номер начального (стартового) сектора является абсолютным (отсчитывается от сектора [0-0-1] жесткого диска) и используется в качестве базового адреса для обеспечения относительной адресации внутри раздела.

Первый раздел всегда начинается по адресу [0-1-1], сразу вслед за MBR.

Структура расширенного раздела DOS

Расширенный раздел DOS содержит один или несколько логических дисков, для каждого из которых создается SMBR. Все SMBR связываются в цепочку (список). Ссылка на первый SMBR этого списка находится в PT MBR. Логическая структура расширенного раздела DOS представлена на рис. 5.23.

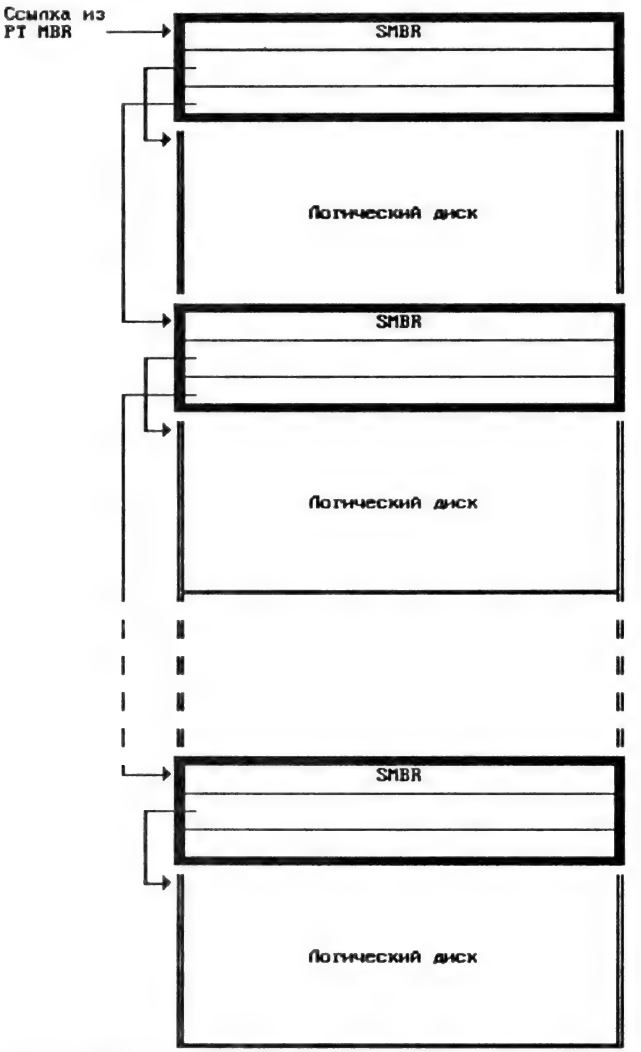


Рис. 5.23. Логическая структура расширенного раздела DOS

Структура SMBR аналогична MBR, но NSB отсутствует, а вместо PT используется LDT. Последняя имеет тот же формат, что и PT, но содержит два описателя вместо четырех. ***Первый описатель*** специфицирует раздел, аналогичный первичному (системный код 01H или 04H). Адреса начала и конца раздела указываются в этом описателе относительно соответствующей SMBR. ***Второй описатель*** LDT специфицирует следующую SMBR (системный код 05H) или является пустым (системный код 00H). Адреса здесь задаются абсолютно.

Производственное издание

Богумирский Борис Сергеевич,
преподаватель Военного инженерно-космического
института имени А.Ф.Можайского,
кандидат технических наук

Руководство пользователя ПЭВМ

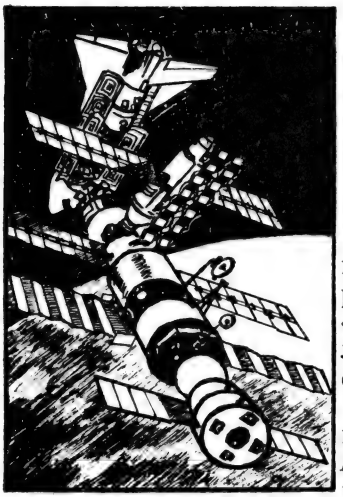
В двух частях
Часть 1

Художник Н.И. Богумирская

Подписано в печать с оригинал-макета 14.10.92. Формат $70 \times 100^{1}/_{16}$.
Гарнитура Таймс. Печать офсетная. Усл. печ. л. 29,01. Уч.-изд. л. 53,82.
Тираж 50 000 экз. Заказ № 10.

Ассоциация «OILCO»
199034, Санкт-Петербург, В.О., 9-я линия, д. 4, а/я 0075, тел. 164-48-52

Отпечатано с оригинал-макета в ГПП «Печатный Двор». 197110, Санкт-Петербург, Чкаловский пр., 15



ВОЕННЫЙ ИНЖЕНЕРНО-КОСМИЧЕСКИЙ КРАСНОЗНАМЕННЫЙ ИНСТИТУТ им. А. Ф. МОЖАЙСКОГО

ОБЪЯВЛЯЕТ ПРИЕМ
СЛУШАТЕЛЕЙ И КУРСАНТОВ НА ПЕРВЫЕ КУРСЫ

Институт готовит высококвалифицированных офицеров-инженеров с высшим военно-специальным образованием для космических частей (космодромов *«Байконур»* и *«Плесецк»*, Центров и пунктов управления космическими аппаратами) и ряда других частей Министерства обороны.

По окончании ВИККИ им. А. Ф. Можайского выпускникам присваивается воинское звание *«ЛЕЙТЕНАНТ»* и выдается диплом общепринятого образца по одной из 27 специальностей:

- на факультете **КОНСТРУКЦИЙ РАКЕТ-НОСИТЕЛЕЙ И КОСМИЧЕСКИХ АППАРАТОВ**:
  - инженеров-механиков (по летательным аппаратам и технологическому оборудованию);
  - инженеров-баллистиков;
- на факультете **СИСТЕМ УПРАВЛЕНИЯ И ВЫЧИСЛИТЕЛЬНОЙ ТЕХНИКИ**:
  - инженеров-электриков (по системам управления летательных аппаратов);
  - инженеров электронной техники (по электронно-вычислительной технике);
  - инженеров электронной техники (по оптико-электронным средствам);
  - инженеров электронной техники (по специальным средствам и системам летательных аппаратов);
- на факультете **РАДИОЭЛЕКТРОНИКИ**:
  - инженеров радиоэлектронной техники (по радиоэлектронным средствам);
  - инженеров электронной техники (по радиоэлектронным системам летательных аппаратов);
- на **ИНЖЕНЕРНО-ТЕХНИЧЕСКОМ** факультете:
  - инженеров-строителей (по наземным и подземным сооружениям и системам их жизнедеятельности);
  - инженеров-механиков (по техническим системам наземных комплексов);
  - инженеров-электриков (по электроснабжению объектов);
- на факультете **СБОРА И ОБРАБОТКИ ИНФОРМАЦИИ**:
  - инженеров-геофизиков (по геофизическому обеспечению и средствам сбора и обработки геофизической информации);
  - инженеров-математиков (по математическому обеспечению автоматизированных систем обработки информации);
  - радиоинженеров (по радиоэлектронным и радиотехническим средствам);
  - инженеров электронной техники-оптиков (по оптико-электронным приборам);
- на факультете **АВТОМАТИЗИРОВАННЫХ СИСТЕМ УПРАВЛЕНИЯ И СВЯЗИ**:
  - радиоинженеров (по радио- и электропроводной связи);
  - инженеров-системотехников (по автоматизированным системам управления);
  - инженеров-математиков (по математическому обеспечению автоматизированных систем управления);
  - инженеров-математиков (по автоматизированной обработке информации);
- на командном факультете **ОРГАНИЗАЦИИ И ПРИМЕНЕНИЯ КОСМИЧЕСКИХ СИСТЕМ**:
  - инженеров-электромехаников.

Срок обучения в институте 5 лет.
Профессиональный отбор проводится с 5 по 30 июля.

На экзамены по общеобразовательной подготовке выносятся дисциплины:
математика (письменно),
физика (устно),
русский язык и литература (письменно).

Желающие поступить в ВИККИ им. А. Ф. Можайского подают заявления в районный (городской) военный комиссариат по месту жительства до мая года поступления.

Адрес института: 197082, Санкт-Петербург, Ждановская ул., 13, Приемная комиссия.
Телефоны: 235-88-74, 235-87-23.


OILCO R
Добыча, переработка, реализация
нефтепродуктов и продуктов нефтехимии
R

ВЫГОДНЫЙ БЕЗОПАСНЫЙ САМОЛЕТ — ЭТО РЕАЛЬНОСТЬ

Ассоциация «OILCO» предлагает бортовую телевизионную систему визуализации для пилота, обеспечивающую повышение безопасности взлета и посадки объектов авиационной техники в условиях ограниченной видимости.

Система обеспечивает:

- автоматическую индикацию с помощью уникальной телевизионной камеры взлетно-посадочной полосы (ВПП);
- определение в реальном масштабе времени специализированным вычислителем значений курса самолета относительно оси ВПП, дальности до ВПП, высоты и скорости;
- визуальное отображение наблюдаемой ВПП и численных значений параметров движения на экране монитора или лобовом стекле кабины.

Система автоматически формирует сигнал, запрещающий взлет или посадку при отклонении параметров движения самолета от допустимых.

Система работоспособна при метеорологической дальности видимости не менее 150—200 м.

Использование разработанной системы не требует проведения доработок имеющегося бортового оборудования объектов авиационной техники.

Система уникальна и запатентована.

Наша система обеспечивает безопасность полета, повышает эффективность использования аэропортов.

ГЛАВНОЕ — МЫ ОБЕСПЕЧИВАЕМ ВАМ БЕЗОПАСНОСТЬ.
ДЛЯ САМОЛЕТОВ С НАШЕЙ СИСТЕМОЙ НЕТ ПЛОХОЙ ПОГОДЫ.

Телефон: 164-48-52 **Факс: 164-48-52**


OILCO®
R
OIL
AVIA
AJAX

КРАСОТА, ЗДОРОВЬЕ, ВКУС, ВКУС К ЖИЗНИ

Продукция нашей фирмы «OILGREEN» представляет собой экстракты, полученные из различных видов растений методом экстракции при сверхвысоком давлении в атмосфере углекислого газа. Экстракты содержат витамины, органические кислоты, аминокислоты, эфирные масла, пигменты, дубильные вещества и многое другое. Полученные при сверхкритических условиях, экстракты не имеют в своем составе тяжелых металлов, пестицидов, нитратов и не концентрируют радионуклидов. С помощью углекислоты существуют технологии для экстракции полезных веществ из более чем 300 видов растений.

Экстракт представляет собой комплекс биологически активных веществ, выделяемых из экологически чистого сырья, при этом соотношения компонентов в экстракте созданы природой, что придает ему особую ценность.

Экстракты могут использоваться в пищевой, парфюмерной, косметической и фармацевтической промышленности.

ДАЖЕ НЕСПЕЦИАЛИСТУ ЯСНО: ЭКСТРАКТЫ — ЭТО БУДУЩЕЕ УЖЕ СЕГОДНЯ.

Телефон: 164-48-52 Факс: 164-48-52

OIL GREEN

OILCO R

НОВЕЙШИЕ ТЕХНОЛОГИИ — ЭТО НАШ СТИЛЬ

Фирма «OILVAC» предлагает технологию напыления разнообразных покрытий на любые материалы в условиях атмосферы или вакуума. Высокая скорость частиц напыляемого порошка (2 км/с) позволяет снизить требования к качеству подготовки покрываемой поверхности.

Для нашей технологии не существует материалов, на которых напыление не зафиксируется.

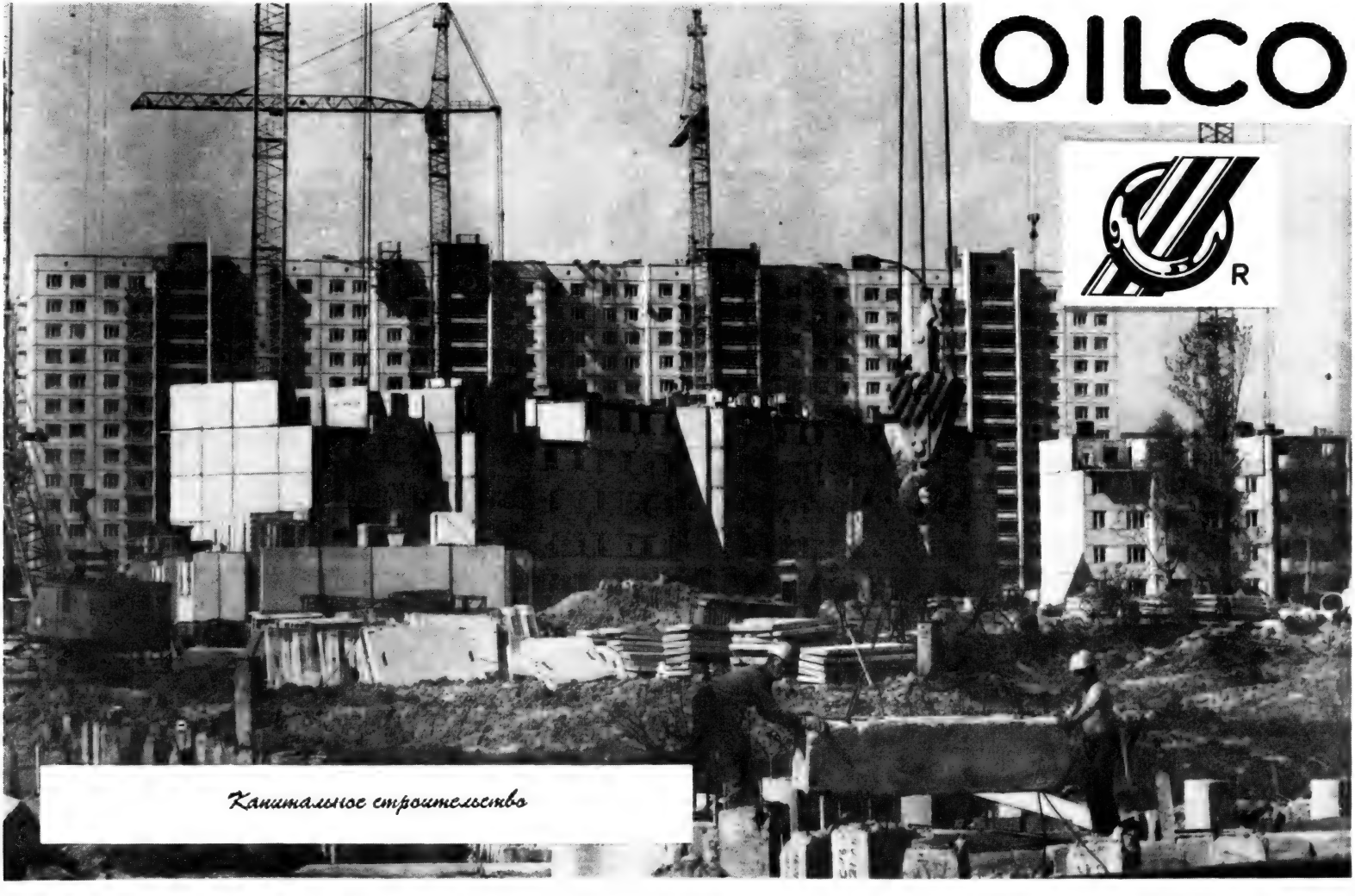
Предлагаемая технология настолько гибка, что позволяет получать фильтры на фольгах из любых металлов и большинства неметаллических материалов. Толщина металлических пластин может достигать 0.2 мм, контролируемый диаметр отверстий лежит в диапазоне 0.1 мкм — 10 мм.

Плотность перфорации может составлять 1000 отверстий/мм^2, площадь фильтра, получаемого за 1 цикл, — 1 м^2. В ходе перфорирования не происходит изменения химического состава материала.

Области применения наших технологий не поддаются перечислению.

Телефон: 164-48-52 Факс: 164-48-52

OILCO R


OILCO
R
Капитальное строительство

ДЛЯ ЗАМЕТОК

ДЛЯ ЗАМЕТОК

Б.С. Богумирский РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ ПЭВМ