МГАПИ

МИНИСТЕРСТВО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

МОСКОВСКАЯ ГОСУДАРСТВЕННАЯ АКАДЕМИЯ
ПРИБОРОСТРОЕНИЯ И ИНФОРМАТИКИ

Кафедра ‘Персональные компьютеры и сети’

Баканов В.М.

Программирование мультимедиа-систем

Учебное пособие

Москва 2004

УДК 681.3

Программирование мультимедиа-систем: Учеб. пособие / В.М.Баканов; МГАПИ. Москва, 2004. 122 с. ISBN 0-000-0000-0

Рекомендовано Ученым Советом МГАПИ в качестве учебного пособия для специальности 2201.

Предлагаемое издание предназначено для студентов IV-V курсов, обучающихся по дисциплине ‘Программирование мультимедиа-систем’ специальности 2201 и смежным или создающих программное обеспечение (ПО) указанного класса в сред**е** различных операционных систем (ОС), может быть применено в качестве курса лекций и для самостоятельной работы. В предлагаемой работе рассматриваются в основном мультимедиа-системы на основе IBM PC-совмест**и**мых ПЭВМ и ОС Windows, однако основные положения верн**ы** и для ЭВМ общего класса.

В работе изложены основы функционирования штатного ПО поддержки устройств мультимедиа, содержится практический материал для разработчиков оригинального программного обеспечения указанных устройств, дан обзор современных аппаратных средств мультимедиа и приведены данные об основных возможностях этих систем (с информативной целью приводятся WEB-адреса разработчиков мультимедиа-устройств и ПО для их функционирования).

Практические примеры рассчитаны на использование интегрированных ср**е**д Visual C++ корпорации Microsoft Corp. и C++Builder/Delphi фирмы Borland Int. и выполняются при поддержке 32-х битовых ОС семейства Windows; предполагается знание осн**о**в языка C/C++.

Последняя версия данного методического пособия может быть получена в виде файла http://pilger.mgapi.edu/metods/m_media.zip.

Табл.25. Ил.17. Библиограф.: 7 назв.

Печатается по решению Редакционно-издательского совета Московской государственной академии приборостроения и информатики.

Научный редактор:	профессор	Михайлов Б.М.
Рецензенты:	профессор	Рощин А.В.
	профессор	Степанова Т.А.

$Л\frac{1402010000}{ЛР020418-97}$

ISBN 0-000-0000-0

© В.М.Баканов, 2004

СОДЕРЖАНИЕ

Стр.

ВВЕДЕНИЕ

Слово ‘мультимедиа’ (*multimedia*, иногда ‘малтимедиа’) представляет собой сочетание двух англоязычных слов - ‘много’ и ‘поср**е**дник, способ, обстановка, контактное воздействие’. Таким образом, термин мультимедиа можно формально перевести как ‘множество способов воздействия’; фактически же понятие мультимедиа подразумевает множество различных методов хранения и представления информации в форме звука, изображения, тактильных и др. воздействий на органы чувств человека Т.о. мультимедиа-системы (ММС) суть системы обработки и представления мультимедиа-данных (ММД).

Следует различать информационное наполнение (контент, *contents*) и программное обеспечение ММС (трудноалгоритмизируемый аспект дизайна ММС вынужденно включен в технологию разработки ПО). Основные виды ММД : текст, графика, анимация, звук, видео, тактильные ощущения; условно можно разделить ММД на справочные, учебные, игровые, управляющие (в большинстве случаев разделы классификации перекрываются).

Мультимедиа является технологией представления сенс**о**рной (доступной через органы чувств) информации в максимально близкой человеку форме и имеет два аспекта - аппаратный и программный (психологические, социальные и т.п. аспекты в данной работе рассматриваются крайне узко).

Аппаратная сторона мультимедиа обеспечивает непосредственный д**о**ступ к информации указанного типа и представляется в виде (с начала 90-х годов ставшими обычными для ПЭВМ) стандартными средствами - видеоадаптерами, мониторами, дисководами жестких дисков и приводами CD-ROM, звуковыми картами и специализированным оборудованием (например, ставшими модными в последнее время очками и шлемами виртуальной реальности), причем развитие номенклатуры подобных устройств носит поистине взрывн**о**й характер.

Программная сторона мультимедиа разделяется на чисто прикладную (приложения, предоставляющие пользователю информацию в определенном виде), специализированную (программные средства для создания мультимедийных приложений) и системную (компоненты конкретной ОС, специально ориентированные на поддержку мультимедийных возможностей).

Мультимедиа является самой молодой из известных областей применения ЭВМ и до сих находится в близкой к начальной стадии развития. Исторически первоначально ЭВМ разрабатывались для обработки весьма специфически организованной (в числовом и только числовом виде) информации, однако большинство ММД труднопредставимы и труднообра-

батываемы в числовом виде (требуют огромных объемов памяти и мощности процессоров для обработки). В результате профессиональное создание и обработка мультимедиа-информации до сегодняшних дней остается дорогостоящей и не всем доступной процедурой. Дальнейшее развитие мультимедиа затруднено сложностями с (необходимым) расширением средств воздействия - например, до сих пор полностью не решен вопрос механизма вкуса и запахов (что исключает возможность включения этих воздействий в арсенал средств мультимедиа), технология тактильных воздействий в данное время также чрезмерно сложна и малоэффективна, идеи о прямом воздействии мультимедиа-информации на мозг человека находятся пока в области, близкой фантастике (хотя и научной).

Традиционные ОС разрабатывались исходя из условия эффективности при обработке числовой информации и обычно малоэффективны при обработке мультимедиа-данных. По-видимому, реально приближение времени специализированных (например, ориентированных на обработку мультимедиа-информации) ОС (например, NextStep и BeOS, см. www.beincorporated.com); сказанное, естественно, относится и к архитектуре ЭВМ, которая уже сейчас претерпевает существенные изменения в связи с развитием мультимедиа.

Целью работы является дать начальные теоретические и практические понятия и сведения о программном и аппаратном обеспечении систем мультимедиа, сведения о представлении (в том числе принципах компрессии) и обработке информации. Так как мультимедиа развивается лавинообразно, часть приведенных материалов уже через несколько месяцев окажется устаревшей, однако общие направления развития ММС обычно стабильны в течении десятилетия.

При ознакомлении с пособием весьма желательна работа с указанными литературными источниками и InterNet-ссылками, а также практика на ЭВМ; данная основа и постоянная практика позволят программисту стать профессионалом.

В качестве основной операционной системы выбрана ОС **Windows** фирмы **Microsoft Corp.** (MS) как наиболее часто применяющаяся на ПЭВМ; при необходимости приводятся сведения об особенностях мультимедиа-программирования в среде иных ОС.

Далее в тексте названия ПО и аппаратных средств выделены ‘жирным’ начертанием, курсивом отмечены названия фирм-производителей (рекомендуется использовать их в качестве ключевых слов при поиске информации в сети InterNet).

1. СТАНДАРТНЫЕ НОСИТЕЛИ МУЛЬТИМЕДИА-ИНФОРМАЦИИ

Особенностью информации мультимедиа является в первую очередь ее значительные объемы – обычным является работа с файлами объемом в несколько гигабайт (здесь существенно ограничение ОС Windows'9x на максимальный объем файла в 2 Гбайта). Вторая особенность заключается в необходимости обеспечения высокой скорости записи и воспроизведения данных – нередки требования в обмене информацией при скорости 10-50 Мбайт/сек.

Традиционно первыми носителями ММ-информации явились магнитные ленты. В современных видеокамерах и видеомагнитофонах непрофессионального класса ('*home video*') обычно используется рассчитанная на $30 \div 60$ мин записи магнитная лента. Современные модели видеокамер оснащены разъемами для вывода аудиовидеоинформации на видеомагнитофон и/или ПЭВМ; применяют аналоговые разъемы типа **RCA** ('*колокольчики*', *'тюльпаны'*) и **S-Video** (*Separate Video*), цифровые модели используют интерфейс **USB** (www.usb.org) и **IEEE 1394** (***FireWire, i.Link, Digital Link***), профессиональным (студийным) интерфейсом является **YUV** (*professional video*). Для копирования относительно небольших объемов информации используют карты памяти **Memory Stick** (объемом нескольких Мбайт).

Только после оцифровки аналогового сигнала появляется возможность оценить его объем в байтах. Современные накопители на твердых дисках ('*винчестерах*') обладают объемом запоминаемых данных $120 \div 250$ Гбайт при стоимости менее 1 US$ за гигабайт при времени произвольного доступа порядка 10^{-2} сек; на таком диске может храниться многочасовое видео.

CD (*compact disk*) представляют собой рассчитанные на считывание и запись информации лучом маломощного твердотельного лазера многослойные стеклопластиковые носители диаметром 120 и толщиной 1,2 мм и разделяются на однократно записываемые на заводе-изготовителе **CD-ROM**, поставляемые без записи и могущие быть однократно записанными пользователем **CR-R** и имеющие возможность многократной перезаписи **CD-RW**.

Информация на CD представлена в виде *питов* (каждый пит несет 1 бит информации) – участков с измененной отражающей способностью размером порядка $(0{,}5 \div 3) \times 10^{-3}$ мм, последовательно размещенных вдоль единственной идущей от наружной к внутренней частей диска дорожки, запись осуществляется посредством прожига более мощным твердотельным лазером расположенного внутри тела диска отражающего слоя (серебро или золото для CD-R).

CD-ROM используют специальную файловую систему, описанную международным стандартом **ISO 9660** (*International Standart Organization*, который совпадает с основными положениями стандарта **HSG** – *High Sierra Group*), большинство приводов CD-ROM отвечают спецификации

MPC (*Multimedua PC*). Для преобразования содержимого кат**а**логов данного формата в стандартный формат MS DOS служит драйвер **MSCDEX.EXE** (в**ы**зов описывается в *AUTOEXEC.BAT*), MSCDEX.EXE применяется совместно с драйвером производителя данной модели пр**и**вода CD-ROM (загр**у**зка из *CONFIG.SYS*).

CD-ROM или CD-R диск имеют объем 640÷700 Мбайт, соответствующая длительность звучания 74÷80 мин определена фирмой **Sony** на основе длительности популярной в Японии 9-й симфонии Бетховена – 72,73 мин, единичная скорость считывания информации составляет 150 Кбайт/сек (известны устройства считывания с 52-х кратной скоростью), принципы хранения и д**о**ступа к информации определяются т.н. ‘Красной’-, ‘Желтой’-, ‘Зеленой’-, ‘Оранжевой книгами’ и др. стандартами. За счет определенных технологических ухищрений объем DVD-дисков достигает 4,7÷17 Гбайт.

Цвет поверхности однократно записываемых (**CD-R**) лазерных дисков определяется технологией и может быть **Цианин** (*Cianine*, при золотом отражающем слое цвет рабочей поверхности темно-зеленый, при серебряном – светло-голубой), **Фталоцианин** (*Phtalocianine*, часто золотой отражающий слой, желто-зеленый оттенок рабочей поверхности) или **Азо** (*AZO*, серебряный отражающий слой, насыщенно синий цвет рабочей поверхности); каждая технология имеет свои достоинства и недостатки.

Твердотельные носители (**CompactFlash, SmartMedia Card, MultiMedia Card, Sequre Digital, Miniature Card** и т.п.) применяются для целей хранения аудиовидеоинформации в цифровых фотокамерах и диктофонах, MP3-плеерах; вследствие быстрого роста их емкости (до 1 Гбайт и выше) наблюдается приближение функциональности цифровых фотокамер к видеокамерам.

Контрольные вопросы

1. Каковы основные виды носителей мультимедиа-информации ? В чем заключается их общая особенность ?
2. В чем заключается принцип записи информации на лазерные диски (CD) ? Что является физической единицей информации на CD ?
3. Для каких целей используются твердотельные носители информации и почему ?

2. МЕТОДЫ ЗАПИСИ И ВОСПРОИЗВЕДЕНИЯ СТАТИЧЕСКИХ ИЗОБРАЖЕНИЙ

3.1. **Способы кодирования цвета при записи и воспроизведении изображений**

Способы представления цвета имеют свою теоретическую и практическую историю. Согласно современным понятиям свет является электромагнитным излучением, причем человеческий глаз воспринимаем лучи с длиной волны приблизительно от 400×10^{-6} мм (фиолетовый) до 700×10^{-6} мм (красный). Данная шкала *является непрерывной*, а понятие ‘цвета’ относится всего лишь к *определенной части* этой шкалы (от синего к красному согласно повышению длины волны).

Человеческий глаз различает сотни цветовых оттенков (известное ‘зрение художника’, якобы м**о**гущее воспринимать многие тысячи цветов, скорее всего является следствием чрезмерной эмоциональности в суждениях). Открытый И.Ньютоном способ разложения цвета на 7 составляющих оказался чрезмерно сложным для практической реализации. Согласно *трехкомпонентной теории* цветового зрения (не единственной из существующих) цвет представляется в виде суперпозиции трех основных цвет**о**в – красного (R), зеленого (G) и синего (B). В 1931 г. решением Международной Комиссии по Освещению (*CIE, Commision International de l’Eclairage*) были стандартизированы монохроматические цвет**а** цветового излучения с дл**и**нами волн соответственно : красный цвет - 700×10^{-6}, синий – $546{,}1\times10^{-6}$ и красный – $435{,}8\times10^{-6}$ мм. На рис. 3.1 схематично показаны схемы основных современных моделей цвета – **RGB**, **CMYK** и **Lab**.

Модель **RGB** (*Red, Green, Blue*) является, пожалуй, наиболее простой и естественной из существующих. Здесь цвет представляется суммой интенсивностей трех составляющих цвета, при этом смешение трех цвет**о**в в одинаковой пропорции дает белый (при максимальной интенсивности составляющих) или серый (при меньшей, но равной, интенсивности составляющих; при нулевой интенсивности составляющих имеем черный цвет). Эта модель именуется *аддитивной* (основанной на *сложении* трех составляющих цвета) и напрямую реализуется в современных сканерах и электроннолучевых трубках мониторов. В Windows модель RGB поддерживается широко – известны системные функции получения полного цвета по его составляющим RGB(Red,Green,Blue) и выделения интенсивности N-ной компоненты ($N\in[R,G,B]$) цвета GetNValue(RGB_Value); при этом интенсивность каждого из цвет**о**в Red, Green, Blue кодируется целым числом от 0 до 255.

Модель RGB имеет и недостатки – цвет**а** на экране монитора могут отличаться от полученных цветоделением, существует взаимозависимость цветовых каналов (при увеличении яркости одного канала в других каналах яркость уменьшается). Развитием RGB-модели является **RGBA** (*Red,*

Green, Blue, Alpha), позволяющая учитывать прозрачность элементов изображения (канал *Alpha*).

Модель RGB совершенно неприменима при цветной печати, когда цвета фактически не суммируются, а вычитаются из белого цвета бумаги (при печати суммирование трех красок равной интенсивности дает не белый, а наоборот – близкий к черному - цвет). При этом в модели **CMYK** используются дополнительные к RGB цвета – голубой (Cyan), пурпурный (Magenta) и желтый (Yellow), модель получила название *субтрактивной*. Для получения серых оттенков приходится давать избыток голубой составляющей, интенсивность Cyan на 10÷20% больше, чем пурпурной и желтой. В реальных цветных принтерах используется дополнительная емкость с черной краской, так как смешение CMY при их полной интенсивности все же не позволяет получить истинно черный цвет (отсюда символ K, см. рис. 2.1).

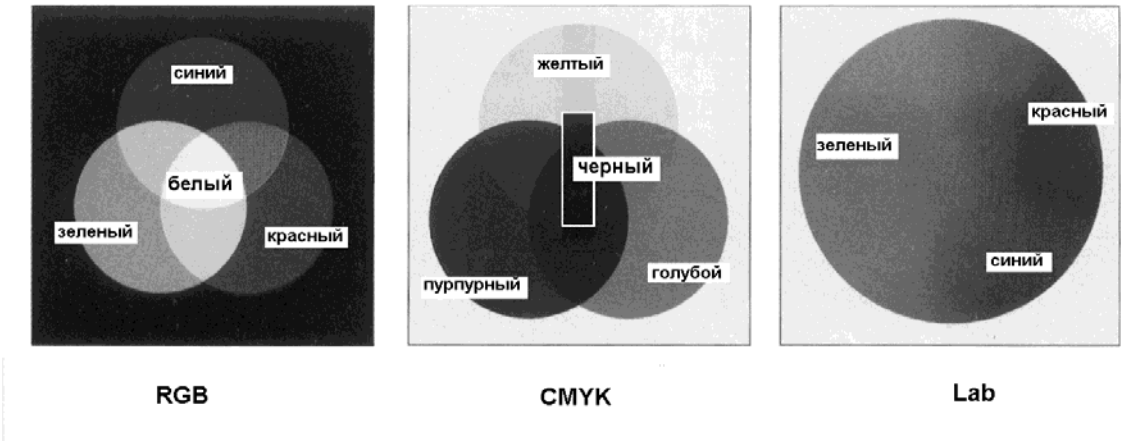

Рис. 2.1. Основные модели представления цвета

Недостатки цветовой модели CMYK – узкий цветовой диапазон, неточное отображение цветов CMYK на мониторе и больший (по сравнению с RGB) расход памяти при реализации.

Самым широким охватом обладает и наиболее точно описывает параметры цвета модель **Lab**. Ее достоинством является полное разделение информации о цвете и яркости, модель часто используется в качестве внутренней во многих программных продуктах для пересчета при переводе цветов из одной модели в другую. Современные пакеты работы с изображениями обязательно позволяет учитывать качество реальных устройств воспроизведения и корректировать цветопередачу (при этом используется понятие *цветовой температуры* – величины, тесно связанной с *амплитудно-частотной характеристикой* данного устройства в диапазоне видимого цвета).

В некоторых случаях профессионалы предпочитают работать с системой HSV, название которой является аббревиатурой терминов *оттенок* (Hue) – *насыщенность* (Saturation) – *яркость* (Value).

Множественность моделей говорит о сложности попыток представления цвета; несмотря на явные достижения и практическое использование работа в этом направлении продолжается в связи с постоянным повышением требований к качеству воспроизведения изображений при экранной демонстрации и получении печатной продукции.

3.2. **Основные форматы файлов изображений**

К настоящему времени число форматов (в случае Window формат обычно определяется расширением имени файла) представления изображений определяется десятками и практически не растет далее (чего нельзя сказать о методах компрессии данных). Формат в большинстве случае определяется расширением имени файла (для MS Windows), однако в некоторых случаях информация в файле с одним и тем же расширением может оказаться сжатой различными методами.

Наипростейшим случаем сохранения растрового (см. подраздел 2.2.1 данной работы) изображения является последовательное кодирование триад цвета в каждой точке изображения; при этом объем файла изображения будет не менее $X \times Y \times 3 \times N$ байт (X,Y – ширина и высота изображения в точках, N – число байт кодирования интенсивности каждого цвета). При размерах изображения 640×480 и ‘глубине’ цвета в 2 байта (2^{16}=65536 градаций) размер файла не менее $640 \times 480 \times 3 \times 2 \approx 2$ Мбайт ! Учитывая, что в настоящее время обычно применяется глубина цвета в 3 или 4 байта (2^{24}=16’777’216 или 2^{32}=4’294’967’296 градаций интенсивности цвета) и значительно б**о**льшие размеры изображения, хранение полноцветных изображения без компрессии практически невозможно (о методах компрессии см. подраздел 2.3).

Одним из (исторически) первых форматов сохранения изображений явился точечный рисунок Windows (расширение имени файла .BMP, .RLE или .DIB; именно с BMP-форматом работает Windows-штатный графический редактор MS Paint. Бл**и**зким к BMP является формат ICO, до сих пор применяемый для сохранения изображений в виде маленьких ‘иконок’.

Файлы формата **PCX** (*PC Paintbrush*) использовались в основном в MS DOS и Windows’3x, поддерживается сжатие (*компрессия*) по методу **ZSoft**.

Графические файлы формата **TIF** (*Tagged File Format*) поддерживают все глубины цветности и используют сжатие.

Формат **GIF** (*Graphics Interchange Format*) поддерживает только 256-цветные изображения и (в современных версиях GIF87/GIF89) последова-

тельность изображений (анимация); для использования на страницах **HTML** (*HiperText Markup Language*) важно свойство ‘прозрачности’ (*transparent*) GIF-файлов. Для создания и редактирования анимированных GIF-файлов имеется большое количество ПО (например, Ulead GIF Animator, www.ulead.com).

JPG-файлы (*JPEG File Interchange Format*) являются сильно компрессованными (возможно выбирать уровень компрессии в ущерб качеству изображения) и практически монополизировали InterNet (кроме небольших анимированных ‘ярлычков’, где применяется GIF).

Формат **DXF** активно применяется фирмой *AutoDESC* (www.autodesc.com) в пакете **AutoCAD** и является стандартом обмена векторной графикой; DXF-файл является текстовым, поддерживает определения сложных объектов, вложенность блоков и др. Текстовый формат этих файлов способствовал их широкому распространению, т.к. (относительно) несложно разрабатывать пользовательские программы для считывания, анализа и создания DXF-файлов. Специально для применения в сети InterNet фирма AutoDESC разработала формат **DWF** (*Drawing WEB File*).

Продвигаемый MS формат **WMF** (*Windows Metafiles Format*) поддерживает векторную графику (и поэтому изображения легко масштабируются) и позиционируется как средство поддержания объектов галлереи кадров (*Microsoft Clip Gallery*).

Проблема авторизации изображений решен путем внедрения в файл изображения т.н. *цифровой метки* (компания *Digimarc Corp*., www.digimarc.com). Эффективность методики настолько высока, что единожды внедренная метка обнаруживается даже после сложных манипуляций с изображением и сканирования картинки, технология принята штатной в продуктах фирм *Adobe* (www.adobe.com) и *Corel* (www.corel.com).

Формат **PDF** (*Portable Document Format*) той же фирмы является форматов электронных документов и может включать текст, графику (как растровую, так и векторную) и иные данные.

Большое количество других форматов изображений используется не столь часто и здесь не рассматривается; некоторые из вышерассмотренных форматов изображений используются и при создании видеофильмов (см. ниже). Практически все форматы изображений приг**о**дны для использования в качестве объектов для предложенной MS технологии внедрения или связывания объектов (**OLE**); причем ‘внутри’ конкретного приложения изображения сохраняются (в случае внедрения) в специфичном формате, для перекодирования применяются (автоматически применяемые) т.н. ‘графические фильтры’.

2.2.1. **Методы представления графической информации**

Традиционно используются два метода представления графической информации – растровый и векторный.

При представлении графической информации в *растровом виде* используется технология хранения информации о каждом пикселе (*pixel- picture element*); пиксел является неделимой единицей - точкой изображения, данные обычно хранятся последовательно в формате одномерного массива, а ширина и высота изображения в пикселах описываются в заголовке файла), сохраняются данные о цвете (в единицах 2^1=2 цвет**а**, 2^8=256 цвет**о**в и т.д.; также сохраняется информация о *палитре* – текущей таблице соответствий представляемого цвета и его кода). При моделировании объемных (трехмерных) объектов используется *воксел* (*voxel – volume picture element*), см. раздел 6.

Историческим ан**а**логом данного метода явилась, вероятно, давно отработанная технология передачи и приема телевизионных изображений (такая же технология применяется при *сканировании изображений*). Типичным представителем этой методики является входящий в штатное системное ПО фирмы MS пакет **Microsoft Paint**; в настоящее время практически все графические редакторы поддерживают растровую графику. Практически все современные системы сохранения движущихся изображений (*movie*) используют растровый способ представления графической информации.

Вторым методом представления графических изображений является *векторный способ*. При этом неделимой единицей является вектор – определяемая начальной и конечной координатами прямая линия; аттрибутами являтся цвет (включая палитру), толщина, тип (сплошная, штрихпунктирная и т.д.) линии. Вырождением линии (вектора) является точка (фактически *растровый способ* представления графической информации). На основе векторов строятся и более сложные графические примитивы – дуги, овалы, гладкие линии произвольной формы и т.д.

Данный метод является естественным для представления информации в виде чертежей, типичным представителем является пакет создания чертежной документации **AutoCAD** (соответствующие файлы формата DXF являются текстовыми и содержат описания графических примитивов в векторном виде); другим представителем пакетов векторной графики является **CorelDraw** (www.corel.com). Размеры файлов при векторном способе обычно значительно меньше, скорость же отрисовки изображений на устройствах вывода практически не отличается. Это объясняется почти 100% применением растровых дисплеев (применение векторных дисплеев в настоящее время ограничено), при этом изображение любых векторных примитивов сводится к (программной) конвертации в растровый формат; используется линейная или круговая интерполяция путем ‘засвечивания’

ближайших к вектору точек растра по *методу Брезенхама* (Bresenham, [1]). Заметим, что векторные графопостроители в настоящее время широко распространены и хорошо согласуются (по форматам передаваемых данных) с технологией векторной графики.

Конвертация между двумя указанными видами представления графической информации тривиальна лишь при переходе от векторной к растровой графике (*метод Брезенхама*), обратный переход требует значительных ухищрений (известны, например, конверторы сканированных растровых изображений в векторный формат **AutoCAD**'а; при этом особенная сложность заключается в распознавании участков растра в районе 'стыковки' векторов, что обычно требует вмешательства оператора).

Существенно различаются для векторной и растровой графики процедуры линейного масштабирования. Если объекты векторной графики масштабируются элементарно, масштабирование растровой графики существенно сложнее (примитивное масштабирование в этом случае приводит лишь к превращению пикселов в прямоугольные образования) – применяются специальные алгоритмы заполнения и сглаживания [1]. Однако эти и более сложные (нелинейные преобразования) легко реализуются вычислительными возможностями ПЭВМ. Более сложные функции класса повышения резкости, оконтуривания, выделения градиентов и областей с заданными свойствами и др. определены лишь для растровой графики; большой набор предопределенных фильтров для подобных преобразований дост**у**пен в пакете **Adobe Photoshop** (www.adobe.com), задаваемые пользователем фильтры удобноприменимы в пакете **Paint Shop Pro** (фирма *Jasc, Inc*., www.jasc.com).

Одна из простых операций такого рода – *локальная цифровая фильтрация*, осуществляемая путем взвешенного суммирования яркостей пикселов, находящихся в некоторой окрестности текущего пиксела [1].

2.2.2. **Текстовые данные в мультимедиа**

Текстовые данные (независимо от типа письма - иероглифического, алфавитного, смешанного) фактически являются частью представления информации в виде статических изображений (графики) и в целом описываются, обрабатываются и представляются теми же методами. Особенностью текста является его вторичность (по отношению к первичности речи, кодовым выражением которой текст формально и является), вследствие чего появляются дополнительные функции ММС: распознавания речи и обратная - речевого воспроизведения текста; эти функции становятся штатными даже для ОС (в частности, небызызвестного проекта **Merlin** фирмы IBM). К сходным проблемам относится и вопрос распознавания символов (технология **OCR** - *Optical Character Recognition*), в настоящее

время удовлетворительно решенный даже в ‘карманных’ ПЭВМ и машинного перевода (в том числе перевода ‘на лету’ в сети InterNet - например, приложение **Promt WebView**, см. www.promt.ru/rus/products/webview) - www.promt.ru, www.translate.ru, www.star.spb.ru. Фирма MS на сайте www.microsoft.com/downloads предлагает специализированную библиотеку разработчика систем распознавания речи **Microsoft Speech API**, системы распознавания и преобразования текста в речь **Microsoft Speech Recognition** и **Microsoft Text-to-Speech**; функциями речевого управления должен обладать пакет MS Office 10.

Символы внутримашинно представлены численным кодом (обычно 8-ю двоичными разрядами, перспективная кодировка UNICODE использует 16 бит и позволяет единообразно представить символы 2^{16}= 65536 языков мира); наличие оставшихся от первых лет компьютерной эпохи нескольких таблиц кодировок (‘кодовых страниц’ - например, Windows-1251, Koi8-R и др.) создает трудности при работе. Наиболее распространенным в сред**е** Windows текстовым (с элементами графики) редактором (*текстовым процессором*) является **MS Word** (www.microsoft.com/rus), из популярных *настольных издательских систем* следует упомянуть Adobe **PageMaker** (www.adobe.com), Xerox **VenturaPublisher** (www.xerox.com) и **Quark XPress** (*Quark, Inc.*, www.quark.com).

Действие OCR-систем заключается в сопоставлении печатным символам (обычно представляемым в виде сканированного изображения) кодовому набору алфавита, ‘понимаемому’ конкретным ПО обработки текстов (изображению символа ставится в соответствие его числовой код). Одной из распространенных OCR-систем является **FineReade**r фирмы *ABBYY Software* (www.abbyy.ru). Последние версии продуктов этой фирмы (ABBYY FineReader Рукопись) позволяют распознавать формы (технология Document Capture - ‘захват документа’), например, бланки налоговых деклараций (с занесением информации из определенных полей бланка в поля базы данных).

Комплекс **Cognitive Forms** принадлежит к классу **OCR/ICR/OMR** (*Optical Character Recognition / Intelligent Character Recognition / Optical Mark Recognition* - оптическое распознавание печатных символов / распознание рукописных символов / оптическое распознание меток) и реализует трехуровневую технологию распознания.

Для представления текстовой информации в приятной человеку форме используются шрифты. *Шрифт* (*гарнитура*) - набор символов, схожих по графическим особенностям. Начертание описывает характерные особенности шрифта (**bold** - жирный, *italic* - курсивный, normal - прямой). *Кегль*, или *размер* шрифта (*size*) определяется высотой прописной буквы, измеренной в *пунктах* (*points*); один пункт равен 1/72 дюйма (0,353 мм), в шрифте размером 12 пунктов прописные буквы имеют высоту 1/6 дюйма.

Эффекты предоставляют возможность применить к выбранному шрифту различные способы оформления - подчеркивание, зачеркивание, оконтуривание, капитель, закрашивание в различные цвет**а** и т.п.

Растровые шрифты имеют фиксированные форму и размеры (например, шрифт MS Sans Serif), причем при масштабировании (только целочисленном) форма символов искажается (возникает ‘ступенька’). *Векторные* (*масштабируемые, контурные*) шрифты (например, Modern) строятся ‘точка за точкой’ при помощи специального штатного для ОС Windows ПО (**GDI** - *Graphic Device Interface*) и допускают масштабирование в любое число раз без искажений, однако для их отрисовки требуются значительные ресурсы. Именуемая **TrueType** разновидность векторных шрифтов (например, Arial) приг**о**дна для вывода как на экран так и на принтер и допускает масштабирование на размер от 1 до 999 пунктов. Близкими к *TrueType* являются шрифт**ы** в формате **PostScript** (предложенный Adobe и ставший всеобщим стандартом язык описания макета страницы, PostScript обеспечивает высококачественный вывод изображений, графики и текста, поддерживая при этом повороты, увеличение и уменьшение символов, для вывода изображений используется интерпретатор PostScript в принтере или в ПЭВМ); для принтеров Hewlett-Packard LaserJet, DeskJet возможно использование технологии **PCL** (*Printer Control Language*), позволяющей осуществлять форматирование распечатываемой страницы в самом принтере.

Шрифты типа TrueType при отрис**о**вке строятся на основе р**е**перных точек, соединенных плавными кривыми (используются квадратичные B-сплайны); ОС Windows имеет штатный набор функций для работы с этими кривыми. Современное ПО создания новых шрифтов (**Fontographer** фирмы *Macromedia, Inc.*, www.macromedia.com; **Font Lab** фирмы *Adobe*, www.adobe.com и др.) позволяет разрабатывать формы символов в графическом диалоге с пользователем, задавая базовые точки и соединяя их кривыми. Деятельность разработчиков шрифт**о**в координирует ежегодная конференция **ATypI** (*Association Typographique Internationale*, www.atypi.org).

Чисто технической сложностью является работа пользователя с текстовыми данными на фоне (растровой или векторной графики). В примитивных графических редакторах класса **MS Paint** после ввода текста его редактирование невозможно, так как он преобразуется в растр. В более мощных редакторах текст сохраняется как отдельный объект (с указанием аттрибутов - фонта, размера, цвета и др.) и при этом отображается в растровом или векторном виде; редактирование объекта позволяет легко изменять текст (и его аттрибуты).

2.3. **Методы сжатия файлов изображений**

Сжатие информации в современных ПЭВМ реализуется программно, причем в большинстве случаев пользователю-непрофессионалу нет необходимости знать, какой метод компрессии используется - ОС самостоятельно применяет нужный кодек, анализируя заголовок медиафайла (в случае отсутствия необходимого кодека выдается соответствующее сообщение). В случае небольшого количества цветов для сжатия используется метод группового кодирования (*run-length encoding*), при этом последовательность одинаковых точек заменяется специальными кодами, несущими информацию о цвете и числе повторов пиксела (т.н. Group 3 метод сжатия, разработанный впервые для факс-аппаратов). Метод предсказания позволяет предсказать цвет следующего пиксела, на этом основана технология сжатия JBIG. Разработано множество несложных, но эффективных приемов кодирования (например, учет т.н. вертикальной избыточности изображения и др.).

К другой группе относятся т.н. методы сжатия с потерями (*lossy compression*), наиболее известным из которых является JPEG (назван согласно аббревиатуре утвердившего его международного объединения *Joint Photographic Experts Group*). Метод основан на том, что человеческий глаз более чувствителен к изменению яркости, а не цвета, и к градациям цвета, а не резкому его изменению. JPEG в основном оперирует информацией о яркости, опуская некоторые данные о цвете, и вместо резкого изменения цвета поддерживает плавные переходы. В результате формат JPEG весьма эффективен при сжатии фотографических изображений, но вызывает заметные искажения четких очертаний контурных рисунков (в которых важна именно резкая смена цвета); мелкие детали изображения могут быть потеряны.

Наиболее распространенными (предлагались еще в пакете **Video for Windows** для Windows'3x, подробнее см. подраздел 4.4.1) штатными для Windows к**о**деками (*codec*, системный драйвер для потокового кодирования медиаинформации) являются MS **Video 1**, **Microsoft RLE** (*Run Length Encoded*), **Indeo** (**In**telVi**deo**, особенно эффективен при работе с 24-битовой глубиной цвета), **Cinepack**; некоторые пригодны и для работы с пот**о**ковым видео.

2.4. **Программное обеспечение создания и обработки изображений**

Наиболее известным ПО для работы с видеоизображениями в настоящее время является известный любому пользователю ПЭВМ PC пакет MS Paint. Пакет позволяет работать с файлами формата BMP, GIF, TIFF, ICO, PNG, JPG, является типичным пакетом растровой графики, позволяет выполнять простейшие функции рисования (создание точек, прямых, эл-

липсов и др. простых геометрических фигур) и редактирования (масштабирование, перенос части изображения, замену палитры и др.). Достоинством пакета является его (чрезмерная) простота, именно поэтому он рекомендуется начинающим.

Одним из наиболее широкообъемлющих по возможностям является пакет растровой графики **PhotoShop** фирмы *Abobe* (www.adobe.com). В пакет включено большое количество модулей геометрического и цветового (линейного и нелинейного) преобразования изображений, возможен учет цветовой настройки устройств вывода (дисплея и принтера). Photoshop (особенно последних версий) 'тяжел' (требует больших ресурсов памяти и производительности процессора) для использования на ПЭВМ. Простым, но обладающим широкими возможностями, пакетом векторной графики является **Paint Shop Pro**).

Типичным ПО для работы с векторной графикой является **CorelDraw** (www.corel.com), возможна отрисовка большого количества графпримитивов (к тому же могущих располагаться в различных слоях и соответственно компоноваться); при сохранении изображений поддерживается большое количество форматов.

Нет возможности (и необходимости) перечислять сотни существующих на данный момент пакетов обработки изображений; важным свойством каждого является возможность конвертации (в том числе групповой) форматов графических файлов.

Контрольные вопросы

1. Чем является согласно современным представлением цвет ?
2. Какие модели синтеза цвета наиболее часто применяются в компьютерных технологиях и чем они отличаются ?
3. Какие основные форматы графических файлов известны и в каких приложениях они используются ?
4. На каких основных принципах основаны технологии компрессации файлов изображений ? Что такое сжатие с потерями и без потерь ?
5. Чем отличаются растровая и векторная графика ?
6. Что такое шрифты TrueType ? Каким образом они строятся и отображаются ?
7. Перечислить наиболее распространенные пакеты для работы с растровой и векторной графикой

3. ОСНОВЫ ЗАПИСИ, СИНТЕЗА И ВОСПРОИЗВЕДЕНИЯ ЗВУКА

3.1. Методы преобразования информации при записи, синтезе и воспроизведении звука

Звук представляет собой колебания физической сред**ы** (обычно воздуха) частотой приблизительно $20 \div 20000$ Гц, все современные системы обработки звука основаны на преобразовании этих колебаний в электрический сигнал, последующей его (ан**а**логовой или цифровой) обработки и вывода вновь в виде колебаний физической сред**ы**. Эффект стереофонии достигается временн**о**й разницей колебаний, легко улавливаемой благодаря наличию приблизительно 20-сантиметровой базы между приемниками аудиоинформации – ушами (разница порядка 7×10^{-4} сек).

В самом начале своей истории компьютер фирмы IBM был оснащен примитивным динамиком, позволявшем (посредством драйвера SPEAKER.DRV) воспроизводить звуки (одновременно) одного тона без регулировки уровня звука; именно в это время были разработаны основные принципы преобразования звука для бытовых компьютеров.

Первый шаг к более серьезной работе со звуком был сделан в 1987 г., когда фирма **Creative Labs** (www.creative.ru) разработала *Creative Music System* (C/MS), представлявший собой 12-голосный стереомузыкальный синтезатор, начавший распространяться в 1989 г. под маркой *Game Blaster*. Огромный коммерческий успех этой карты привел в скором времени к появлению других подобных карт, наиболее известной из которых является карта **AdLib**; в основе их функционирования лежит методё известный как нижеописанный ‘синтез путем частотной модуляции’ (*FM Syntesis*, см. ниже).

Запись произвольного звука осуществляется путем прямой оцифр**о**вки аналогового сигнала, представляющего собой электрическую копию звукового давления (преобразователем является датчик звукового давления - микрофон). Частота оцифровки (частота преобразования) называется частотой выборки сигнала и по известной теореме Котельникова-Найквиста должна быть не ниже удвоенного значения максимальной частоты преобразуемого сигнала (например, если спецификация MPC Level 1 определяет частоту преобразования 11 kГц, то верхний предел записываемой частоты составляет около 5 kГц).

Преобразование аналогового сигнала в цифровую форму выполняет аналого-цифровой преобразователь (АЦП), служащий для дискретизации сигнала по времени (частота оцифровки) и квантования по уровню (собственно цифровое представление сигнала). Обычно в АЦП применяется технология преобразования с импульсно-кодовой модуляцией (PCM, *Pulse Code Modulation*). Временн**ы**е промежутки между моментами преобразования сигнала называют интервалами выборки (*Sampling Interval*); эта величина обратно пропорциональна частоте выборки, или сэмплингом (*Sampling Rate*). Амплитуда аналогового сигнала (*Sample Value*) при каждом преобразовании делится (квантуется) по уровню и кодируется в соответствующий параллельный цифровой код (*Digital Sample*), время преобра-

зования аналогового сигнала в цифровой код именуется временем выборки (*Sampling Time*), рис. 3.1.

Разрешающей способностью АЦП называется наименьшее значение ан**а**логового сигнала, которое приводит к изменению цифрового кода. Например, если АЦП выдает 8-разрядный код, разрешающая способность равна $1/(2^8)=1/256$ от максимальной амплитуды аналогового сигнала (около 0,4% в относительных единицах), 16-разрядный АЦП имеет точность представления сигнала не хуже $1/(2^{16})=1/65536$ (0,0015%).

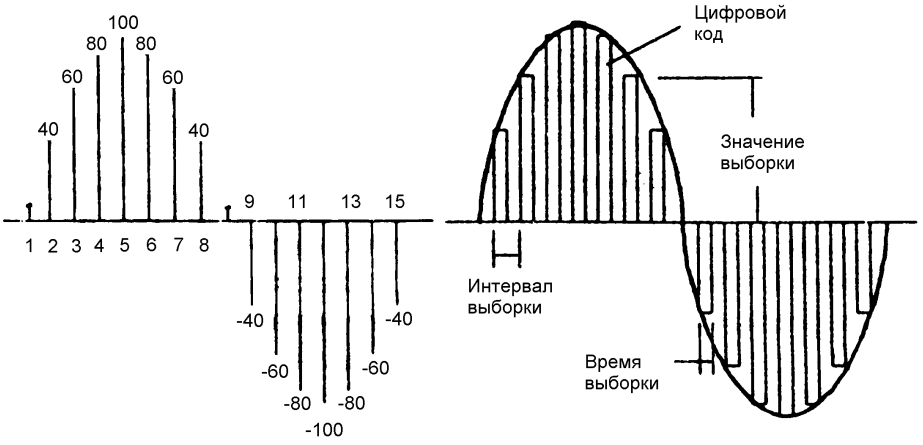

Рис. 3.1. Характеристики процесса преобразования между ан**а**логовым и цифровым сигналами

С увеличением разрядности АЦП растет его динамический диапазон (каждый дополнительный бит соответствует увеличению приблизительно на 6 дБ). 8-разрядное преобразование обеспечивает динамический диапазон 48 дБ (качество кассетного магнитофона), 12-разрядное – 72 дБ (качественный катушечный магнитофон), 16-разрядное – 96 дБ (качество аудио компакт-диска).

Полученный с АЦП параллельный код разрядностью $8 \div 16$ последовательно (побитно) записывается с частотой сэмплинга в аудиофайл (при необходимости используется буфферизация), при этом поток несжатых цифровых аудиоданных вел**и**к (см. таблицу ниже); ниже будут приведены методы снижения потока данных и размеров аудиофайлов.

Частота (kГц)	*Разрешение (бит)*	*Режим*	*Качество звучания*	*Скорость передачи данных (kb/s)*	*Размер файла минутной записи (Мбайт)*
11,025	8	моно	телефонная линия	10	0,66

11,025	8	стерео		21	1,3
11,025	16	моно			1,3
11,025	16	стерео			2,6
22,05	8	моно	радио-трансляция	21	1,3
22,05	8	стерео		43	2,6
22,05	16	моно			2,6
22,05	16	стерео			5,3
44,1	8	моно			2,6
44,1	8	стерео			5,3
44,1	16	моно		86	5,3
44,1	16	стерео	запись на CD	172	10,5

В студийной работе происходит переход на стандарт 96 kГц/24 бита, который по теоретически достижимому качеству пока заметно перекрывает возможности существующих звуковых систем.

Заметим, что АЦП (также как и ЦАП - цифро-аналоговые преобразователи) выполняют свои функции аппаратно, не загружая ЦП (центральный процессор); последний управляет только режимами работы АЦП и ЦАП. Обобщенные схемы записи и воспроизведения звука приведены на рис.3.2 и 3.3; именно так создаются и воспроизводятся широкоизвестные **WAV**-файлы (вопросы сжатия и распаковки ‘на лету’ аудиоданных обсуждаются ниже).

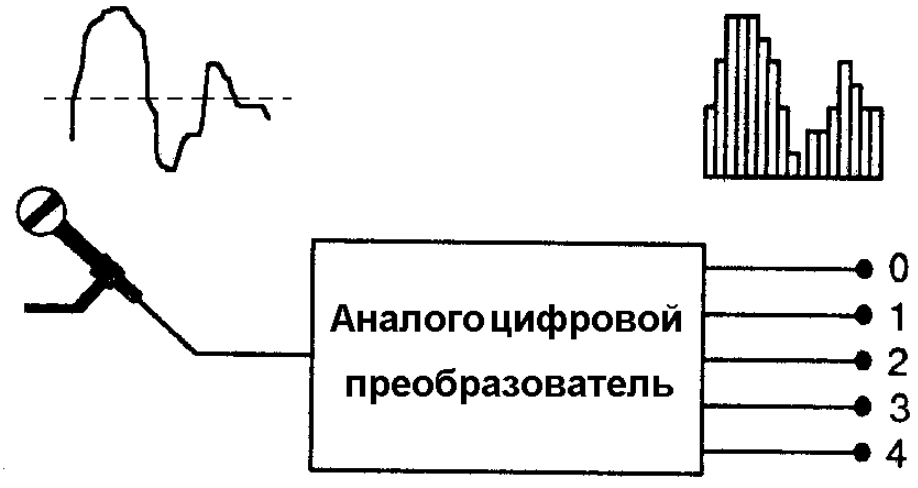

Рис. 3.2. Аналого-цифровое преобразование при записи звука

Рис. 3.3. Цифро-аналоговое преобразование при воспроизведении звука

ЦАП выполняет обратное (цифро-аналоговое) преобразование, в результате работы ЦАП получается ступенчатый сигнал, представляющий собой исходный аналоговый сигнал плюс обусловленная сэмплингом высокочастотная составляющая (лежащая выше верхнего предела слышимых частот и поэтому легко фильтруемая), рис.4.4.

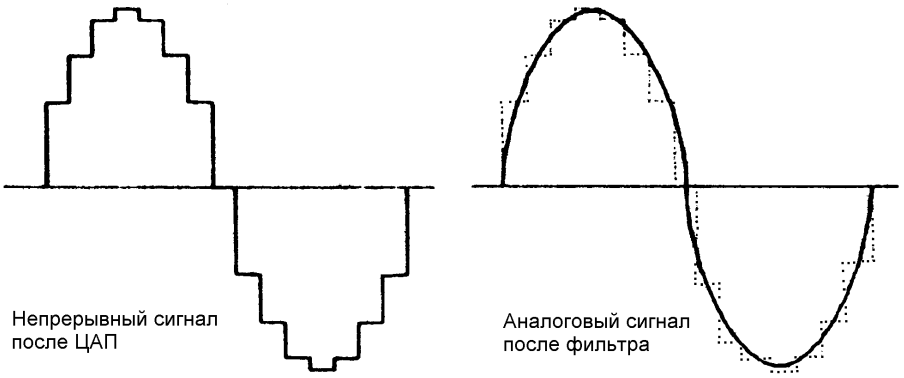

Рис. 3.4. Сглаживание ступенчатого сигнала после ЦАП при воспроизведении звука

Сказанное относится к записи и воспроизведению произвольного звука, во многих случаях возможно значительно сократить объем хранимых данных путем *синтеза* (создания) звука. При этом мы лишаемся возможности работы с произвольным звуком, остается лишь возможность обработки некоторого (моделируемого) подмножества звучаний. Достаточно широко применяются звуковые карты, оснащенные **DSP** (*Digital Signal*

Processor), обладающие многими дополнительными возможностями обработки звука (распознавание речи, реверберация, спецэффекты типа 3-х мерного звучания и др.).

Наиболее часто применяют цифровой FM-синтез звука, основы которого заложены в конце 70-х годов студентом Стенфордского университета Джоном Чоунингом (*John Chowning*). Несколько десятилетий ранее Роберт Муг (*Robert Moog*) реализовал в серии своих всемирно известных синтезаторов аналоговый вариант FM-синтеза путем использования генераторов огибающей, управляющих амплитудой отдельных VOC-генераторов (*Voltage-Controlled Oscillator*).

В цифровом FM-синтезе каждый из описанных управляемых генераторов называется *оператором.* В операторе выявляются два базовых элемента: *фазовый модулятор* и *генератор огибающей*. Фазовый модулятор задает частоту (высоту) звука, а генератор огибающей - его амплитуду (громкость); см. общую схему рис. 3.5.

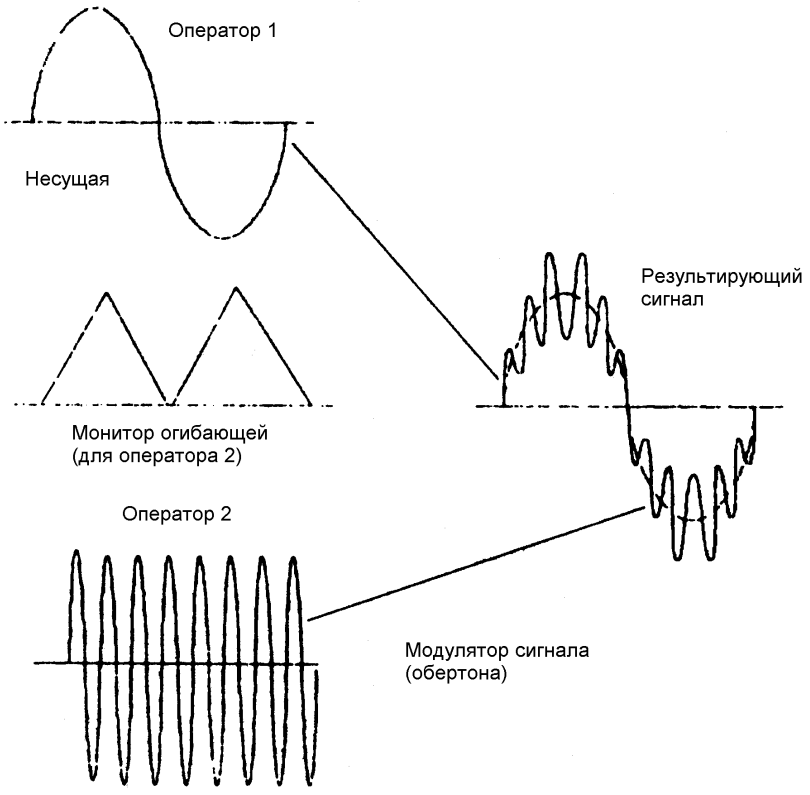

Рис. 3.5. Генерация сигналов с заданной огибающей при получении звука посредством FM-синтеза

В большинстве случаев для синтеза одного инструмента достаточно двух операторов - оператора несущей (основной тон) и оператора модулирующей частоты (*обертон*). Например, для струнных инструментов (фортепиано, гитара и др.) можно выделить общие моменты - при нажатии

произвольной клавиши (возбуждении колебаний струны) амплитуда сначала быстро возрастает до максимума, затем несколько спадает, после чего следует относительно продолжительный участок медленного падения амплитуды и, наконец, участок быстрого затухания. Описанные стадии сигнала носят названия *Attack, Decay, Sustain* и *Release* соответственно, поэтому сам генератор огибающей именуется **ADSR**-генератором (по первым буквам фаз сигнала, см. рис. 3.6).

Обычно пара операторов определяет *голос*; современные наборы микросхем для FM-синтеза звука содержат до $36 \div 40$ голосов, осуществляя различные режимы (алгоритмы) FM-синтеза (в том числе и самые сложные, предполагающие использовать 18 и более операторов для синтеза речи). В звуковых картах обычно присутствует специальный *генератор шума*, обрабатываемый одним оператором (оператором огибающей).

Кроме FM-синтеза, в высококачественных звуковых картах используется табличный или WT-синтез (*Wave Table synthesis*); такие устройства именуют также *синтезаторами выборок* или *сэмплерами* (*Samples*). Идея применения WT-синтеза состоит в использовании специальных алгоритмов, позволяющих по одному лишь характерному тону (*выборке*) музыкального инструмента воспроизвести все остальные тона (фактически восстановить его полное звучание).

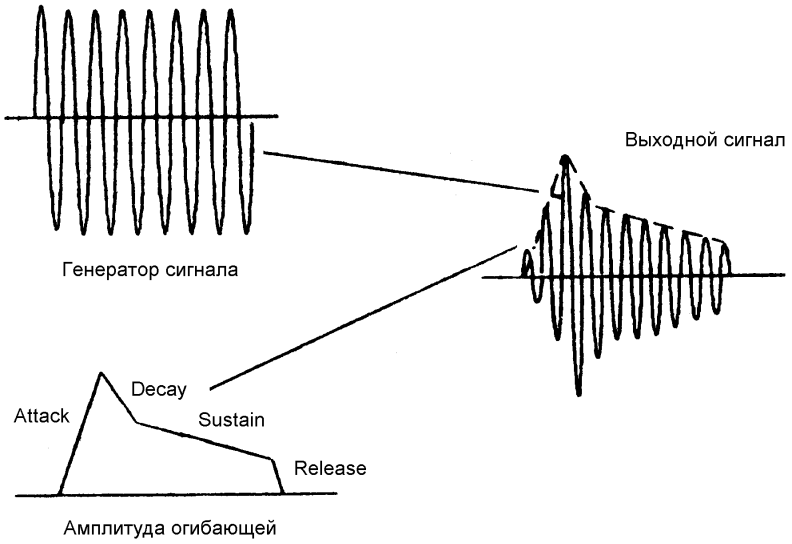

Рис. 3.6. Синтез звука при использовании генератора огибающей (ASDR-генератор)

Выборки сигналов (таблицы) сохраняются в ROM (*Read Only Memory*) или программно загружаются в RAM (*Random Access Memory*) звуковой карты, после чего специализированный WT-процессор выполняет операции над выборками сигнала, изменяя их амплитуду и частоту (рис. 4.7). При этом генерируемое WT-методом звучание ближе к звуку реальных инструментов, нежели при FM-технологии. Дополнительную гибкость WT-

методу дает возможность простого изменения таблиц выборок; многие карты поддерживают как FM- так и WT-синтез.

Файлы для генерации звука посредством FM-технологии имеют расширение **MID** (от MIDI - *Musical Digital Interface*, совмест**и**мым форматом является **RMI**) и содержат ссылки на ноты (кодируемые числами), их длительность и тип музыкального инструмента (до 200 инструментов в современных картах). MID-файлы естественным образом могут быть воспроизведены и на поддерживающих WT-синтез звуковых картах.

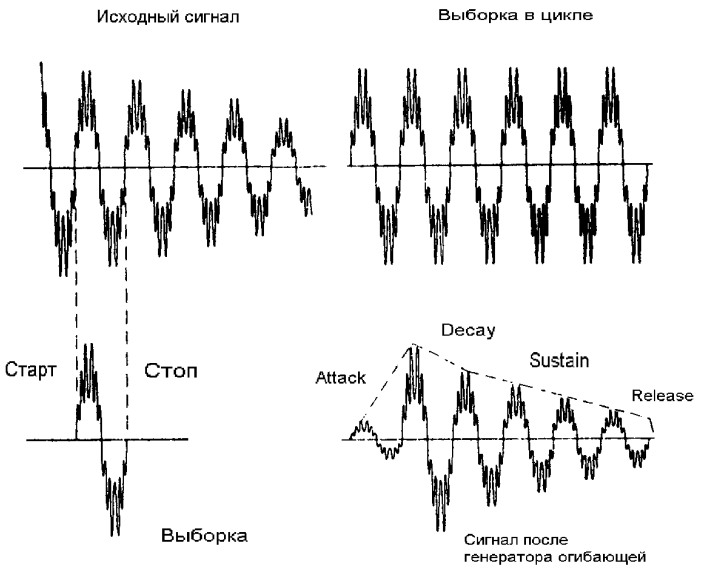

Рис. 3.7. К описанию технологии WT-синтеза звука

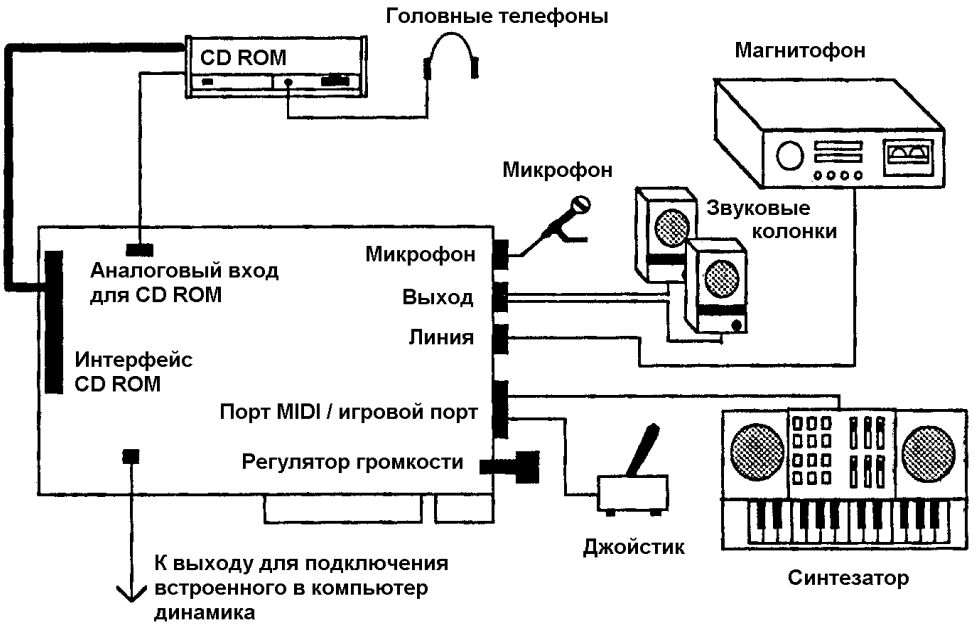

Рис. 3.8. Типовое подключение внешних устройств к звуковой карте IBM PC

3.2. **Методы сжатия информации при работе со звуком**

Чем более объем памяти WT-карты, тем реалистичнее звучание (ибо в памяти хранится больше образцов, записанных с более высоким разрешением). Стандарт *General MIDI* описывает более 200 инструментов, для хранения образцов их звучания (таблиц) требуется не менее 8 Мбайт памяти (минимум 20 Кбайт для каждого образца).

Известен WF-метод (*Wave Form*) генерации звучания, основанный на преобразовании звуков в сложные математические формулы и дальнейшем применения этих формул для управления мощным процессором с целью воспроизведения звука; от WF-синтеза ожидают еще лучшей (относительно FM и WT-технологий) реальности звучания музыкальных инструментов при ограниченных объемах звуковых файлов .

Типовая схема подключения внешних устройств к IBM PC-ориентированной звуковой плате (*карте*) приведена на рис. 4.8.

Для сокращения потока данных используются иные (отл**и**чные от PCM) методы кодирования аналогового сигнала. Например, известна существенно сокращающая объем хранимых данных техника кодирования, основанная на известных характеристиках аналогового сигнала; при т.н. μ-кодировании ан**а**логовый сигнал преобразуется в цифровой код, определяемый логарифмом величины сигнала (а не его линейным преобразованием). Недостаток метода - необходимость иметь априорную информацию о характеристиках исходного сигнала.

Известны методы преобразования, не требующие априорной информации об исходном сигнале. При *дифференциальной импульсно-кодовой модуляции* (DPCM*, Differential Pulse Code Modulation*) сохраняется только разность между текущим и предшествующим уровнями сигнала (разница требует для цифрового представления м**е**ньшего количества бит, чем полная величина амплитуды). При *дельта-модуляции* (DM*, Delta Modulation*) каждая в**ы**борка состоит всего из одного бита, определяющего знак изменения исходного сигнала (увеличение или уменьшение); дельта-модуляция требует повышенной частоты сэмплинга. Технологии дифференциальной импульсно-кодовой модуляции связаны с накапливающейся со временем ошибкой, поэтому применяются специальные меры периодической калибровки АЦП.

Наибольшее распространение при записи звука получила *адаптивная импульсно-кодовая модуляция* (*ADPCM, Adaptive Pulse Code Modulation*), использующая 8- или 4-разрядное кодирование для разности сигналов. Технология впервые была применена фирмой *Creative Labs* и обеспечивает сжатие данных до 4:1.

Однако часто применяются иные (программные) методы сжатия/распаковки аудиоинформации; среди них в последнее время наиболее популярен формат **MP3**, разработанный институтом **Fraunhofer IIS** (*Fraunhofer Institute Integrierte Schaltungen*, www.iis.fhg.de) и фирмой THOMSON (полная спецификация формата MP3 опубликованы на сайте www.mp3tech.org). Полное название стандарта MP3 звучит MPEG-Audio Layer-3 (где **MPEG** суть *Moving Picture Expert Group*, не путать с предназначенным для использовании в телевидении высокой четкости стандартом MPEG-3).

MP3-кодирование данных происходит посредством выделения независимых отдельных блоков данных - фреймов. Для этого исходный сигнал при кодировании разбивается на равные по продолжительности участки, именуемые фреймами и кодируемые отдельно (для дополнительного снижения объема данных применяется сжатие с применением *алгоритма Хеффмена*); при декодировании сигнал формируется из последовательности декодированных фреймов. Процесс кодирования требует ощутимого времени, декодирование (при воспроизведении) осуществляется ‘на лету’.

MP3-формат обеспечивает наилучшее качество звука при минимальном объеме файла. Это достигается учетом особенностей человеческого слуха, в том числе эффекта маск**и**рования слабого сигнала одного диапазона частот более мощным сигналом соседнего диапазона (когда он имеет место) или мощным сигналом предыдущего фрейма, вызывающего временное понижение чувствительности уха к сигналу текущего фрейма (проще говоря, удаляются второстепенные звуки, которые не слышатся человеческим ухом из-за наличия в данный/предыдущий момент другого -

более громкого звука). Также учитывается неспособность большинства людей различать сигналы, по мощности лежащие ниже определенного уровня, разного для разных частотных диапазонов. Этот процесс называется *адаптивным кодированием* и позволяет экономить на наименее зн**а**чимых с точки зрения восприятия человеком деталях звучания. Степень сжатия (следовательно и качество), определяются не форматом MP3, а *шириной потока данных* при кодировании.

Аудиоинформация, сжатая по такой технологии, может передаваться потоком (streaming), а может храниться в файлах формата MP3 или WAV-MP3. Отличие второго от первого состоит в наличии дополнительного заголовка WAV-файла, что позволяет при наличии MP3 - кодека (codec, кодер и декодер в комплексном исполнении) в системе использовать для работы с таким файлом стандартные средства Windows. Параметры компрессии при кодировании файла можно варьировать в широких пределах. Качество, неотличимое большинством рядовых слушателей от качества CD, достигается при скорости передачи (*bitrate, битрейт*) 112÷128 Кбайт в секунду; при этом сжатие составляет примерно 14:1 относительно исходного объема. Специалисты обычно требуют скорости передачи 256÷320 Кбайт/сек (это соответствует всего лишь двойной скорости CD-проигрывателя, но для большинства отечественных InterNet - линий недоступна).

Принципиальной особенностью MPEG-кодирования (как видео-, так и аудиоинформации) является *компрессия с потерями*. После упаковки и распаковки звукового файла методом MP3 результат *не идентичен оригиналу* ‘бит в бит’. Напротив, упаковка целенаправленно исключает из упаковываемого сигнала несущественные компоненты, что и приводит к чрезвычайному возрастанию коэффициента сжатия (сжатие до 96:1 при качестве телефонного канала).

Для MP3 также написано множество удобного программного обеспечения. Налажено производство аппаратных (карманных и автомобильных) MP3 плееров (MP3 поддерживает до 5 каналов).

На рубеже 1998÷1999 г. фирма **XingTech** (www.xingtech.com) первая использовала технологию *переменного битрейта* (VBR, *Variable Bite Rate*). В случае VBR задается максимальный допуст**и**мый уровень потерь, а кодер выбирает минимальный битрейт, достаточный для выполнения поставленной задачи. Стоящие рядом в конечном потоке фреймы могут оказаться в итоге закодированными с разными параметрами.

По расчетам специалистов MP3 останется актуальным в ближайшее десятилетие (даже несмотря на существование форматов AAG и VQF и продвигаемого MS формата **WMA**). О существовании иных к**о**деров (преобразователей информации из одного формата в другой) см. www.sulaco.org/mp3/free.html и www.xiph.org.

Возможным конкурентом MP3 в (не столь близком) будущем может стать формат MPEG-4 (точнее, его аудиокомпонента), основанный на объектном подходе к звуковым сценам (язык **BIFS** позволяет располагать источники звука в трехмерном пространстве сцены, управлять их характеристиками и применять к ним эффекты независимо друг от друга и т.д., в следующих версиях предполагается добавление возможности задания акустических параметров среды).

Для кодирования аудиообъектов MPEG-4 предлагает наборы инструментов как для ‘живых’ звуков, так и для синтезированных. MPEG-4 устанавливает синтаксис двоичных потоков и процесс декодирования в терминах наборов инструментов, что позволяет применять различные алгоритмы сжатия. Диапазон предлагаемых стандартом скоростей потока для кодирования живых звуков - от 2 до 128 Кбайт/сек и выше. При кодировании с переменным потоком минимальная средняя скорость может оказаться еще меньше (порядка 1,2 Кбайт/сек). Для звука высшего качества применяется алгоритм AAC, который дает качество лучше, чем у CD при потоке в 10 с лишним раз меньше. Другой возможный алгоритм кодирования живого звука - **TwinVQ**. Для кодирования речи предлагаются алгоритмы **HVXC** (*Harmonic Vector eXcitation Coding*) для скоростей потока $2 \div 4$ Кбайт/сек и **CELP** (*Code Excited Linear Predictive*) для скоростей $4 \div 24$ Кбайт/сек.

MPEG-4 предполагает возможность синтеза речи. На входы синтезатора поступает проговариваемый текст, а также различные параметры ‘окраски’ голоса - ударения, изменения высоты тона, скорости произнесения фонем и т. п. Можно также задать для ‘говорящего’ пол, возраст, акцент и др. В текст можно вставлять управляющую информацию, обнаружив которую синтезатор синхронно с произнесением соответствующей фонемы передаст параметры или команды другим компонентам системы (например, параллельно с голосом может генерироваться поток параметров для анимации лица). Как и всегда, MPEG-4 задает правила работы, интерфейс синтезатора, но не его внутреннее устройство.

Интересная часть ‘звуковой’ составляющей - средства синтеза произвольных звуков и музыки. MPEG-4 предлагает в качестве стандарта подход, разработанный в колыбели многих передовых технологий - **MIT Media Lab**. и названный SA (*Structured Audio*, Структурированный Звук). Это не конкретный метод синтеза, а формат описания методов синтеза, в котором можно задать любой из существующих методов (а также, как утверждается, будущих). Для этого предлагаются два языка - **SAOL** (*Structured Audio Orchestra Language*) и **SASL** (*Structured Audio Score Language*). Первый задает оркестр, а второй - то, что этот оркестр должен играть. Оркестр состоит из инструментов, каждый инструмент представлен сетью элементов цифровой обработки сигналов - синтезаторов, цифровых фильтров, которые все вместе и синтезируют нужный звук. С помощью SAOL

можно запрограммировать практически любой нужный инструмент, природный или искусственный звук. Сначала в декодер загружается набор инструментов, а затем поток данных SASL заставляет этот оркестр играть, управляя процессом синтеза; таким образом обеспечивается одинаковое звучание на всех декодерах при очень низком входном потоке и высокой точности управления. С появлением MPEG-4 фактически обретает более реальные и понятные очертания идея ITV (*Interactive TeleVision, Интерактивное Телевидение*), о котором спорят уже несколько лет и под которым каждый понимает нечто свое (от простого ‘видео-по-запросу’ до детективов с многовариантным развитием сюжета и участием зрителя).

Данные о MPEG-4 приведены в основном для информации о современных тенденциях записи и синтеза медиаданных, интересующихся отсылаем к cselt.it/mpeg и www.mpeg.org. В конце 2000 г. группа разработчиков MPEG планировала объявить об окончании работы над стандартом MPEG-7 (официальное название - *Multimedia Content Description Interface*).

3.3. **Простые способы воспроизведения звука**

В простейшем случае приложение должно выдавать звуковые сигналы или проигрывать небольшие звуковые сообщения; для этих целей можно воспользоваться API-Windows функциями **MessageBeep** и **sndPlaySound**.

Первая из упомянутых функций имеет прототип

```
void
MessageBeep(UINT  uAlert);
```

При установленном драйвере звукового адаптера проигрывается звуковой фрагмент, указанный передаваемым в качестве функции кодом; при отсутствии драйвера выдается лишь короткий ‘бип’ из встроенного динамика.

Числовой параметр определяется значениями строк ключа **HKEY_CURRENT_USER\AppEvents\Schemes\Apps\Default** системного реестра, связь событий с нужными звуками производится с помощью приложения Control Panel; связь параметра функции с описанием некоторых событий приведены в нижерасположенной таблице

Значения параметра функции MessageBeep	*Описание*
-1	Стандартный звуковой сигнал, выдаваемый на встроенный динамик
MB_ICONASTERISK	Проигрывается WAV-файл, описанный строкой SystemAsterisk
MB_ICONEXLAMATION	Аналогично, строка SystemExclamation

MB_ICONHAND	Аналогично, строка SystemHand
MB_ICONQUESTION	Аналогично, строка SystemQuestion
MB_OK	Аналогично, строка SystemDefault

Функция **MessageBeep** пытается проиграть звуковой фрагмент в асинхронном (фоновом) режиме; в случае невозможности этого управление возвращается только после окончания проигрывания.

Произвольный звуковой файл проигрывается функцией **sndPlaySound**, находящейся в библиотеке **mmsystem.dll**

```
BOOL
sndPlaySound(LPSTR  lpszSoundFile,
             UINT  wFlags);
```

Через первый параметр функции передается ссылка на ресурс, содержащего звуковой фрагмент (имя WAV-файла, идентификатор ресурса приложения или вышеописанную текстовую строку); параметр **wFlag** определяет способ проигрывания звукового фрагмента (нижеприведенные значения можно комбинировать при помощи операции ИЛИ)

Значения параметра wFlag	*Описание*
SND_SYNC	Определяет синхронный режим работы (функция **sndPlaySound** вернет управление только после завершения проигрывания звукового фрагмента)
SND_SAYNS	Определяет асинхронный режим работы (функция **sndPlaySound** вернет управление немедленно, проигрывание звукового фрагмента будет выполняться в фоновом режиме параллельно с работой приложения)
SND_NODEFAULT	Если указанный фрагмент не найден, функция **sndPlaySound** сразу вернет управление приложению. Если флаг **SYNC_NODEFAULT** не указан и файл не найден, будет воспроизведен стандартный системный звук (соответствующий строке **SystemDefault**); если и это невозможно - звучания а не будет, а функция вернет **FALSE**
SND_MEMORY	Используется для проигрывания загруженных в оперативную память звуковых файлов (например, из ресурсов приложения)
SND_LOOP	При указании **SND_ASYNC** проигрывание зацикливается (останов возможен только вызовом **sndPlaySound** с первым нулевым параметром)
SND_NOSTOP	При осуществлении проигрывания в данный момент возвращается **FALSE**

Во всех случаях (кроме использования флага **SND_NOSTOP**) функция **sndPlaySound** возвращает **TRUE**, если в данный момент проигрыва-

ние выполняется и **FALSE** в противоположном случае. При использовании функций **MessageBeep** и **sndPlaySound** WAV-файл должен полностью помещаться в физическую память.

В работе [6] приведен полный С-код приложения **SNDPLAY**, демонстрирующего различные способы использования функции **sndPlaySound**.

3.4. **Интерфейс управляющих строк MCI**

MCI (*Media Control Interface*) представляет собой универсальный, независимый от особенной аппаратуры интерфейс, предназначенный для управления устройствами мультимедиа (звуковые и видеоадаптеры, устройства чтения звуковых компакт-дисков и лазерных видеодисков). Возможности интерфейса MCI удовлетворяют практически всем потребностям любого приложения мультимедиа, предназначенного для записи и воспроизведения звуковой или видеоинформации; при необходимости обработки данных на низком уровне или в реальном времени (редактирование и преобразование WAV-файлов, распознавание речи и/или образов, преобразование речи в режиме реальном времени) может использоваться нижеописанным интерфейсом низкого уровня. Ниже подробно описано использование интерфейса MCI при работе со звуком, однако MCI также эффективен при работе с видео; известный **Delphi**- и **C++Builder**-компонент **MediaPlayer** (подробнее см. подраздел 5.3.4) является всего лишь надстройкой над уровнем MCI-инструкций.

Достаточно подробное описание интерфейса MCI приведено в работе [5]; там же дано множество фрагментов исходного кода и готовых (несложных) приложений. Все функции интерфейса MCI экспортируются из библиотеки **mmsystem.dll**; эти функции непосредственно обращаются к драйвер**а**м устройств ввода/вывода и к функциям низкого уровня, определенным в той же библиотеке. Заметим, что не все команды могут быть выполнены конкретным устройством мультимедиа (например, не все устройства работы с лазерными дисками имеют возможность их записи), поэтому перед использованием устройства необходимо программно выяснить его возможности (для чего имеются специальные функции).

Приложения могут использовать два типа программного интерфейса MCI - основанный на использовании текстовых команд *интерфейс управляющих строк* (*Command-String Interface*) и основанный на посылке сообщений *интерфейс управляющих событий* (*Command-Message Interface*, см. подраздел 3.5 данной работы).

Интерфейс управляющих строк удобен для использования в системах программирования высокого уровня; например, для проигрывания звукового файла **ding.wav** достаточно передать звуковому адаптеру следующую последовательность управляющих строк

```
open ding.wav type waveaudio alias snd wait
play snd wait
close snd
```

При использовании интерфейса управляющих строк используется функция **mciSendString**, которой в качестве первого параметра передается указатель на строку команды

```
DWORD
mciSendString(LPSTR lpstrCommand, // строка управления
              LPSTR lpstrReturnString, // буфер для результата
              UINT wReturnLength, // размер этого буфера
              HANDLE hCallback); // идентификатор окна извещения
```

Здесь **lpstrCommand** - дальний указатель на текстовую управляющую строку, **lpstrReturnLength** - размер буфера для занесения результата выполнения команды (в текстовом виде, для игнорирования результата можно использовать **NULL**), **wReturnLength** - размер этого буфера, **hCallback** - идентификатор получающего сообщение **MM_MCINOTIFY** окна (сообщение посылается после завершения операции устройством, для игнорирования сообщения можно использовать **NULL**).

В нижеприведенной таблице приведены возвращаемые функцией **mciSendString** значения (при успехе возвращается нуль)

Значение	*Описание*
MCIERR_BAD_CONSTANT	Указана недопустимая для данной команды константа
MCIERR_BAD_INTEGER	Указано недопустимое для данной команды значение
MCIERR_DUPLICATE_FLAGS	Двойное определение параметра или значения
MCIERR_MISSING_DEVICE_ NAME	В управляющей строке не указано имя устройства, драйвера, файла или алиас (альтернативное имя)
MCIERR_MISSING_STRING_ ARGUMENT	Не указан обязательный параметр команды
MCIERR_NEW_REQUIRED_ ALIAS	При использовании параметра **new** не указан алиас
MCIERR_NO_CLOSING_ QUOTE	В команде отсутствуют закрывающие двойные кавычки
MCIERR_NOTIFY_ON_AUTO_ OPEN	Для автоматически открываемого устройства нельзя указывать флаг **notify**
MCIERR_PARAM_OVERFLOW	Строка параметра не помещается в буфер (размер оного следует увеличить)
MCIERR_PARSER_INTERNAL	Ошибка в драйвере устройства (возможно, драйвер следует заменить новым, более поздней вер-

	сии)
MCIERR_UNRECOGNIZED_ KEYWORD	Не распознан параметр управляющей строки

Для преобразования полученного от функции **mciSendString** кода ошибки в текстовую строку можно воспользоваться функцией **mciGetErrorString**, которой необходимо передать двойное слово кода ошибки (текстовая строка будет возвращена в параметре lpstrBuffer)

```
UINT
mciGetErrorString(DWORD  dwError,  //  код ошибки
                  LPSTR  lpstrBuffer,  //  буфер для записи текстовой
                                       //  строки ошибки
                  UINT  wLength);  //  размер этого буфера
```

При успешном завершении функция **mciGetErrorString** возвращает **TRUE**; при невозможности сопоставить заданному коду текстового описания возвращается **FALSE**.

В работе [6] приведен полный С-код приложения **MCISTRRVW**, демонстрирующего использование строчного интерфейса MCI для воспроизведения звукового файла; использование команд интерфейса управляющих строк для управления CD ROM и проигрывания MIDI-файлов поясняется также в подразделах 3.7.1 и 3.8.1 данной работы.

4.5. **Интерфейс управляющих сообщений MCI**

Более тесное и гибкое взаимодействие между разработанным с использованием языка C/C++ приложением и устройством мультимедиа можно достичь при использовании интерфейса управляющих сообщений; при этом используется функция **mciSendCommand**, которой в качестве второго параметра передается код соответствующего управляющего сообщения.

Прототип функции **mciSendCommand** (см. файл **mmsystem.h**) приведен ниже

```
DWORD
mciSendCommand(UINT  wDeviceID,  //  идентификатор устройства
               UINT  wMessage,  //  код сообщения
               DWORD  dwParam1,  //  флаги команды
               DWORD  dwParam2);  //  указатель на структуру
                                  //  параметров
```

Здесь **wDeviceID** - идентификатор управляемого устройства (для сообщения **MCI_OPEN** не используется, т.к. идентификатор создается в ре-

зультате выполнения именно этой команды), **wMessage** - код сообщения, **dwParam1** - флаги команды, **dwParam2** - указатель на структуру параметров (формат коей зависит от кода сообщения).

Функция **mciSendCommand** возвращает нуль при нормальном завершении или код ошибки (текстовое описание ошибок можно получить с помощью функции **mciGetErrorString**, передав ей этот код в качестве параметра).

Нижеприведенный фрагмент кода открывает устройство ‘waveaudio’ (будет открыт файл, путь к которой записан в переменной **szFileAudio**)

```
// определение переменных
MCI_OPEN_PARMS  mciOpen;
DWORD  dwFlags,  dwRc;

// заполнение полей структуры  mciOpen
mciOpen.lpstrDeviceID = (LPSTR) "waveaudio";
mciOpen.lpstrElementName = (LPSTR) szFileName;
mciOpen.dwCallback = 0;
mciOpen.wDeviceID = 0;
mciOpen.wReserved = 0;
mciOpen.lpstrAlias = NULL;
dwFlags = MCI_OPEN_TYPE | MCI_OPEN_ELEMENT | MCI_WAIT;

// собственно открытие файла с помощью функции  mciSendCommand
dwRc = mciSendCommand(0,  MCI_OPEN,  dwFlags,
                      (DWORD) (LPVOID) &mciOpen);
```

После выполнения данного фрагмента в переменную **dwrc** будет занесен код результата завершения; при успешном выполнении в поле **wDeviceID** структуры **mciOpen** заносится идентификатор открытого устройства.

В следующем примере закрывается мультимедиа - устройство с определенным ранее идентификатором **mciOpen.wDeviceID**

```
// определение переменных
MCI_GENERIC_PARMS  mciGen;
DWORD  dwRc;

// заполнение полей структуры  mciOpen
mciGen.dwCallback = 0;

// собственно выполнение функции  mciSendCommand
dwRc = mciSendCommand(mciOpen.wDeviceID,  MCI_CLOSE,  MCI_WAIT,
                      (DWORD) (LPMCI_GENERIC_PARMS) &mciGen);
```

В работе [6] приведен полный С-код приложения **MCIWAVER**, демонстрирующего использование интерфейса управляющих сообщений MCI для воспроизведения звуковых WAV-файлов; использование этого интерфейса для управления CD ROM и проигрывания MIDI-файлов приведено также в подразделах 3.7.2 и 3.8.2 данной работы.

3.6. **Интерфейс низкого уровня**

Программным интерфейсом низкого уровня удобно пользоваться в случае необходимости иметь непосредственный д**о**ступ к буферам, содержащим мультимедиа-данные; для случая работы со звуковыми файлами интерфейс обеспечивается несколькими функциями, имеющими префикс *wave* (например, **waveInOpen**, **waveOutOpen**, **waveOutWrite**, **waveAddBuffer** и т.д., причем функции экспортируются из файла **mmsystem.dll**).

Общая последовательность использования интерфейса низкого уровня заключается в чтении и проверке формата заголовка WAV-файла, открытия устройства вывода с указанием конкретного формата звуковых данных, блочного чтения данных WAV-файла, подготовки специальной функцией для вывода и передаче данных драйверу устройства вывода (драйвер выводит их на звуковой адаптер). При этом приложение должно самостоятельно подготовит блоки данных в оперативной памяти.

Запись звуковых данных осуществляется аналогично. В первую очередь открывается устройство ввода (ему указывается формат звуковых данных), далее заказывается один или несколько блоков памяти (они подготавливаются для ввода путем вызова специальной функции), подготовленные таким образом блоки по мере необходимости передаются драйверу устройства ввода (драйвер заполняет их записанными звуковыми данными). С целью сохранения записанных данных в WAV-файле приложение должно сформировать и записать в файл заголовок WAV-файла и звуковые данные из подготовленных и заполненных драйвером устройства ввода блоков памяти.

Программный интерфейс низкого уровня требует внимательного учета всех деталей процесса записи и воспроизведения (в отличие от интерфейса MCI, где многие параметры принимаются по умолчанию). Как всегда, б**о**льшая трудоемкость программирования интерфейса низкого уровня компенсируется повышенной гибкостью и возможностью работы со звуковыми данными в реальном времени.

3.6.1. **Формат wav-файла, информация о RIFF-структуре файлов**

Имеющие отношение к мультимедиа данные (звук, видео и др.) хранятся в файлах т.н. **RIFF**-формата (*Resource Interchange File Format* -

формат файла для обмена ресурсами). Как содержащие звук WAV-файлы, так и **AVI**-файлы, содержащие видеоинформацию, имеют формат RIFF.

Файл формата RIFF содержит вложенные фрагменты (*chunk's*); внешний фрагмент состоит из заголовка и области данных (рис. 3.9).

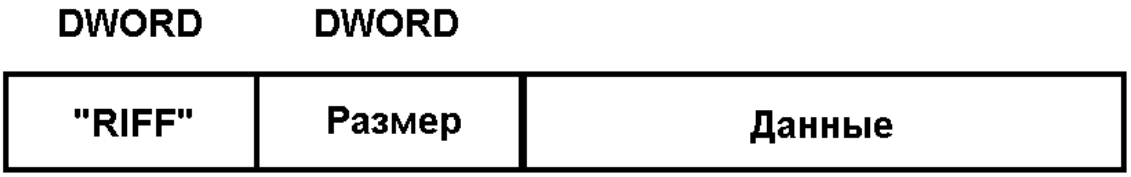

Рис. 3.9. Внешний фрагмент RIFF

Первое двойное слово заголовка содержит четырехбуквенный код FOURCC, идентифицирующий хранящиеся во фрагменте данные. Второе двойное слово заголовка представляет собой размер области данных в байтах (без учета размера самого заголовка).

Область данных имеет переменную длину, однако она должна быть выравнена на границу слова (при необходимости дополняется в конце нулевым байтом до целого числа слов).

Важно понять, что формат RIFF не описывает конкретный формат данных; практически файл в RIFF-формате может содержать любые мультимедиа-данные, причем формат конкретных данных зависит от типа этих данных (RIFF является скорее стандартом описания контейнера данных).

Обозначенная на рис. 3.9 как 'Данные' область может содержать внутри себя другие фрагменты. Для содержащего звуковые данные файла (WAV-файл) эта область содержит идентификатор данных 'WAVE', фрагмент формата звуковых данных 'fmt' (три символа 'fmt' и пробел в конце), а также фрагмент звуковых данных (рис. 3.10).

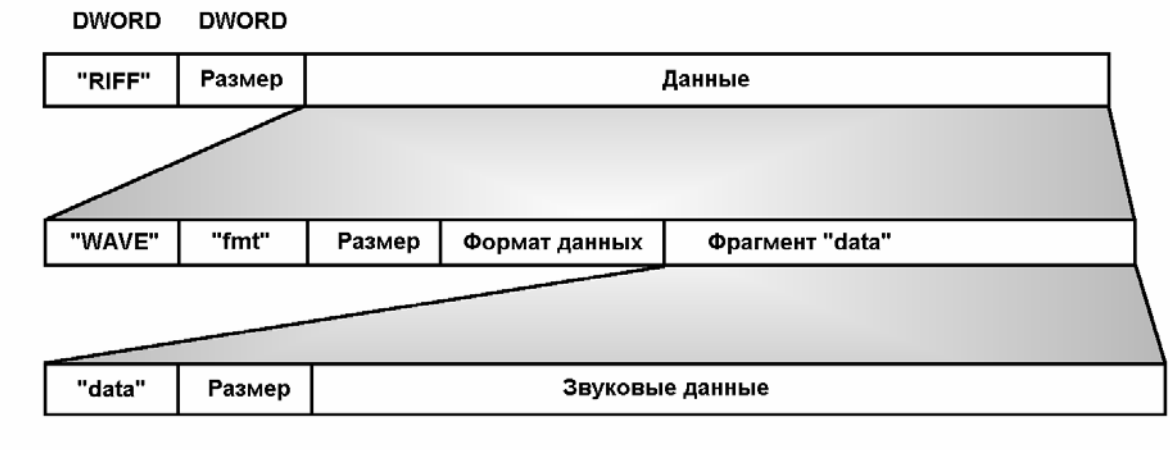

Рис. 3.10. Формат WAV-файла (в структуре RIFF)

Файл может дополнительно содержать фрагменты других типов, поэтому не следует предполагать, что заголовок WAV-файла имеет фиксированный формат. Например, в формате могут присутствовать фрагменты ‘LIST’ или ‘ABOUT’, содержащие информацию о правах копирования и описание самого мультимедиа-файла.

Означенная на рис. 3.10 как ‘Формат данных’ область описывает звуковые данные. Формат этой области для файлов PCM (записанных с использованием импульсно-кодовой модуляции) соответствуют структуре **PCMWAVEFORMAT**, определенной в файле **mmsystem.h** следующим образом

```
typedef  struct  pcmwaveformat_tag
{
   WAVEFORMAT  wf;
   WORD  wBitsPerSample;
}  PCMWAVEFORMAT;

typedef  PCMWAVEFORMAT          *PPCMWAVEFORMAT;
typedef  PCMWAVEFORMAT NEAR  *NPPCMWAVEFORMAT;
typedef  PCMWAVEFORMAT  FAR   *LPPCMWAVEFORMAT;
```

Структура **WAVEFORMAT** описана в файле **mmsystem.h** следующим образом

```
typedef  struct  waveformat_tag
{
   WORD  wFormat Tag;  // тип формата
   WORD  nChannels;  // количество каналов (моно или стерео)
   DWORD  nSanplesPerSec; // частота дискретизации
   DWORD  nAvgBytesPerSec; // скорость потока данных
   WORD  nBlockAlign; // выравнивание блока данных
}  WAVEFORMAT;

typedef  WAVEFORMAT          *PWAVEFORMAT;
typedef  WAVEFORMAT  NEAR  *NPWAVEFORMAT;
typedef  WAVEFORMAT  FAR    *LPWAVEFORMAT;
```

Поле **wFormatTag** описывает тип формата звуковых данных (для поддерживаемой стандартной библиотекой **mmsystem.dll** метода импульсно-кодовой модуляции PCM в этом поле должно находиться определенное в файле **mmsystem.h** значение **WAVE_FORMAT_PCM**)

```
#define WAVE_FORMAT_PCM  1
```

Поле **nChannels** содержит количество каналов, в нем могут находиться значения 1 (моно) или 2 (стерео).

В поле **nSamplesPerSec** записывается частота дискретизации (количество выборок сигнала в секунду). В этом поле могут находиться стандартные (11025 kГц, 22050 kГц или 44100 kГц) либо нестандартные значения (такие как 5000 kГц или 4400 kГц), однако не все драйверы звуковых адаптеров могут корректно работать с нестандартными частотами дискретизации.

Поле **nAvgBytePerSec** содержит *среднюю* скорость потока данных (количество байт в секунду, передаваемых драйверу устройства или получаемых от него). Эта информация может использоваться приложением для оценки размера необходимого для размещения звуковых данных буфера (например, для монофонического сигнала с дискретизацией 8 бит значение скорости численно совпадает со значением частоты дискретизации, для стереофонического сигнала с дискретностью 8 бит она вдвое выше). Точное значение величины **nAvgBytePerSec** рассчитывается по формуле

nAvgBytePerSec=(nChannel $\times$ **nSamples** $\times$ **wBitsPerample) / 8**

В поле **nBlockAlign** находится информация о выравнивании блока в байтах, причем

nBlockAlign=(nChannels $\times$ **wBitsPerSample) / 8**

Поле **wBitPerSample** определяет дискретность сигнала (количество бит, используемое для одной выборки сигнала); обычно используются значения 8 или 16.

Формат самих звуковых данных зависит от количества каналов и от дискретности.

Для монофонического сигнала с дискретностью 8 бит звуковые данные представляют собой массив однобайтовых значений, каждое из которых является выборкой сигнала.

Для стереофонического сигнала с дискретностью 8 бит звуковые данные имеют формат массива двухбайтовых слов, причем младший байт слова соответствует левому каналу, а старший - правому.

Формат звуковых данных с дискретностью 16 бит выглядит аналогично (для монофонического сигнала данные хранятся в массиве 16-битовых слов; для стереофонического используется массив двойных слов, причем младшему слову соответствует левый канал, а старшему - правый).

Диапазон изменения значений выборок определяется дискретизацией. Для 8-битовых данных диапазон составляет от 0 до 255 (0xff), причем отсутствию сигнала (полная тишина) соответствует значение 128 (0x80); для

16-битовых значений диапазон изменения составляет от –32768 (-0x8000) до 32767 (0x7fff), отсутствию сигнала соответствует значение 0.

3.6.2. **Функции для работы с файлами**

Библиотека **mmsystem.dll** содержит удобные функции, специально предназначенные для работы с файлами RIFF-формата (хотя можно пользоваться стандартные функции ввода - вывода или функции семейства **_hread** и **_hwrite**). Специализированные функции успешно работают с блоками памяти б**о**льшого размера (более 64 Кбайт), так как звуковые данные редко помещаются в одном сегменте памяти.

Открытие файла совершается функцией **mmioOpen**. Эта функция может открыть файл для буферизованного или небуферизованного ввода или для работы с файлом в оперативной памяти (полностью возможности указанной функции приведены в поставляемой совместно с MS SDK документации). Ниже приведен прототип функции **mmioOpen**

```
HMMIO
mmioOpen(LPSTR  szFilename,
         LPMMIOINFO  lpmmioinfo,
         DWORD  dwOpenFlags);
```

Здесь **szFilename** - дальний указатель на текстовую строку, содержащую путь к открываемому файлу, **lpmmioinfo** - указатель на содержащую дополнительные параметры для операции открытия файла структуру **MMIOINFO** (может быть задан как **NULL**), **dwOpenFlags** - определяющие режим открытия файла флаги.

При успехе функция **mmioOpen** возвращает (нестандартный) идентификатор открытого файла, который можно использовать только в функциях с префиксом **mmio**. Код ошибки можно определить, анализируя поле **wErrorRet** структуры **MMIOINFO**.

Формат структуры **MMINFO** описан в файле **mmsystem.h** как

```
typedef  struct  _MMIOINFO
{
   // поля общего назначения
   DWORD  dwFlags;  // общий флаг состояния
   FOURCC  fccIOProc;  // код идентификации процедуры
                       // ввода - вывода
   LPMMIOPROC   pIOProc; // указатель на процедуру
                         // ввода - вывода
   UINT  wErrorRet;  // код завершения
   HTASK  htask;  // идентификатор локальной процедуры
                  // ввода - вывода
```

```
   // поля для буферизированного ввода - вывода
   LONG  ccBuffer;  // размер буфера (или 0L)
   HPSTR  pchBuffer;  // указатель на буфер (или NULL)
   HPSTR  pchNext;  // указатель на следующий байт при
                    // операциях чтении или записи
   HPSTR  pchEndRead;  // указатель на последний прочитанный байт
   HPSTR  pchEndWrite;  // указатель на последний записанный байт
   LONG  lBufOffset;  // дискового смещение начала буфера
   // поля для процедур ввода - вывода
   LONG  lDiskOffset;  // дисковое смещение для следующей
                       // операции чтения или записи
   DWORD  adwInfo[3];  // дополнительные данные
                       // для типа MMIOPROC
   // прочие поля
   DWORD  dwReserved1;  //  зарезервировано
   DWORD  dwReserved2;  //  зарезервировано
   HMMIO  hmmio;  // идентификатор открытого файла
} MMIOINFO;

typedef  MMIOINFO          *PMMIOINFO:
typedef  MMIOINFO NEAR *NPMMIOINFO;
typedef  MMIOINFO FAF     *LPMMIOINFO;
```

Структура **MMIOINFO** дает возможность задавать многочисленные способы работы с файлами - можно использовать файлы в памяти, можно определить собственную процедуру для выполнения нестандартного ввода или вывода или работать с открытыми средствами MS-DOS идентификаторами файлов. В простейших случаях можно указать второй параметр функции **mmioOpen** как **NULL** и не использовать структуру **MMIOINFO** вообще

```
hmmio=mmioOpen((LPSTR)lpszFilename, NULL,
                    MMIO_READ | MMIO_ALLOCBUF);
```

Последний параметр функции **mmioOpen** предназначен для определения режима открытия файла в виде логической комбинации ИЛИ отдельных флагов, ниже приведен их список

Идентификатор флага	*Описание режима открытия файла*
MMIO_READ	Чтение
MMIO_WRITE	Запись
MMIO_READWRITE	Чтение и запись
MMIO_CREATE	Создание нового файла (если файл с таким именем уже существует, он обрезается до нулевой длины)
MMIO_DELETE	Удаление файла (если удаление выполнено корректно, возвращается **TRUE**, в противном случае - **FALSE**)

MMIO_PARSE	Создание текстовой строки, содержащей полный путь к файлу на основе пути, переданного функции через параметр **szFilename**; результат замещает буфер **szFilename**
MMIO_EXIST	Определяет существование указанного файла (при существовании создается текстовая строка, содержащая полный путь к файлу)
MMIO_ALLOCBUF	Файл открывается для буферизованного ввода - вывода (по умолчанию размер буфера равен 8 Кбайт; изменить размер буфера можно заданием его в поле **cchBuffer** структуры **MMIOINFO**)
MMIO_COMPAT	Файл открывается в режиме совместимости (файл может быть открыт несколько раз)
MMIO_EXCLUSIVE	Файл будет открыт в монопольном режиме
MMIO_DENYWRITE	Запрещает другим приложения открывать этот файл на запись
MMIO_DENYREAD	Запрещает другим приложения открывать этот файл на чтение
MMIO_DENYNONE	Другие приложения могут открывать файл и на запись, и на чтение
MMIO_GETTEMP	Создание текстовой строки для открытия временного файла (строка записывается в буфер, адрес которого передается через первый параметр функции; собственно открытие файла не выполняется)

В нижеследующем фрагменте кода выполняется создание файла, открываемого и на чтение, и на запись

```
hFile=mmioOpen(szFileName,  NULL,
                MMIO_CREATE | MMIO_READWRITE);
```

Закрытие файла, открытого функцией **mmioOpen** (после завершения работы с ним) должно осуществляться функцией **mmioClose**.

```
UINT
mmioClose(HMMIO  hmmio,  UINT  wFlags);
```

Параметр **hmmio** - полученный ранее с помощью функции **mmioOpen** идентификатор открытого файла, **wFlags** - определяющие режим закрытия файла флаги (можно указать флаг **MMIO_FHOPEN** для закрытия файла, открытого средствами MS-DOS). При успехе возвращается нулевое значение, в противном случае - код ошибки.

Запись в открытый с помощью функции **mmioOpen** файл осуществляется с помощью функции **mmioWrite**. Эта функция позволяет за единичный вызов записать в файл блок данных размером большим 64 Кбайт; по-

сле записи осуществляется перемещение текущей позиции в файле вперед на количество записанных байт.

```
LONG
mmioWrite(HMMIO  hmmio,  // идентификатор открытого файла
          HPSTR  hpBuff,  // указатель на буфер данных
          LONG  dwBytes);  // размер буфера
```

При успехе функция **mmioWrite** возвращает количество записанных байт или -1 при возникновении ошибки.

Чтение из открытого с помощью функции **mmioOpen** файла выполняется функцией **mmioRead**, также позволяющей за один вызов прочитать блок данных размером более 64 Кбайт; после чтения осуществляется перемещение текущей позиции в файле вперед на количество прочитанных байт.

```
LONG
mmioRead(HMMIO  hmmio,  // идентификатор открытого файла
          HPSTR  hpBuff,   // указатель на буфер данных
          LONG  dwBytes);  // размер буфера
```

При успехе функция **mmioRead** возвращает количество прочитанных байт или -1 при возникновении ошибки; при достижении конца файла возвращается нулевое значение.

Позиционирование в открытом с помощью функции **mmioOpen** файла выполняется функцией **mmioSeek**.

```
LONG
mmioSeek(HMMIO  hmmio,  // идентикатор открытого файла
          LONG  dwOffset,  // смещение для текущей позиции
          int  nOrigin);  // код отсчета смещения
```

Величины **dwOffset** и **nOrigin** интерпретируются обычным для файловых операций в языке С образом. При успехе функция **mmioSeek** возвращает новое смещение текущей позиции в файле (от начала файла, в байтах) или -1 при возникновении ошибки.

Для работы с RIFF-файлами в библиотеке **mmsystem.dll** присутствуют специальные функции, сильно облегчающие работу с фрагментами RIFF-файлов (хотя можно использовать обычные функции ввода - вывода, в том числе описанные выше). Эти специализированные функции помогают заполнить четырехбайтовый идентификатор фрагмента, найти в файле нужный фрагмент и установить на него (или за него) текущую позицию файла, создать новый фрагмент в новом файле.

При формировании нового фрагмента удобна функция **mmioFOURCC**, позволяющая создать четырехбуквенный код фрагмента из отдельных букв

```
FOURCC
mmioFOURCC(CHAR ch0,  // первая буква кода
           CHAR ch1,   // вторая - . - . - . -
           CHAR ch2,   // третья - . - . - . -
           CHAR ch3);  // четвертая - . - . - . -
```

Здесь параметры **ch0**, **ch1**, **ch2**, **ch3** - коды букв, образующих четырехбуквенный код; функция возвращает значение сформированного четырехбуквенного идентификатора, который можно использовать при формировании нового фрагмента.

Нетрудно догадаться, что функция **mmioFOURCC** реализована в виде макроса, упаковывающего четыре байта в двойное слово

```
#define mmioFOURCC(ch0,ch1,ch2,ch3)   \
                ((DWORD) (BYTE) (ch0) |          \
                ((DWORD) (BYTE) (ch1) << 8) |    \
                ((DWORD) (BYTE) (ch2)  << 16) |  \
                ((DWORD) (BYTE) (ch3)  <<  24))
```

Для форматирования, например, идентификатора "WAVE" можно воспользоваться вышеописанным макросом следующим образом

```
FOURCC  fourccWaveID;
fourccWaveID=mmmioFOURCC('W', 'A', 'V', 'E');
```

С помощью функции **mmioStringToFOURCC** можно сформировать четырехбуквенный идентификатор не из отдельных букв, а из строки символов.

```
FOURCC
mmioStringToFOURCC(LPCSTR szString, UINT wFlags);
```

Здесь **szString** - указатель на преобразуемую строку, закрытую двоичным нулем, при **wFlags=MMIO_TOUPPER** все символы строки будут преобразованы в заглавные. Функция **mmioStringToFOURCC** возвращает значение четырехбуквенного идентификатора, могущего быть использованным при формировании нового фрагмента; пример использования функции приведен ниже

```
FOURCC fourccWaveID;
```

```
fourccWaveID=mmioStringToFOURCC("wave", MMIO_TOUPPER);
```

Для создания нового фрагмента в RIFF-формате удобно использовать функцию **mmioCreateChunk**; новый фрагмент создается в текущей позиции файла, предварительно открытого с помощью функции **mmioOpen**

```
UINT
mmioCreateChunk(HMMIO  hmmio, // идентификатор файла
                LPMMCKINFO  lpck, // указатель на
                                  // структуру MMCKINFO
                UINT  wFlags);  // тип фрагмента
```

При **wFlags=MMIO_CREATERIFF** создается фрагмент "RIFF", при **wFlags=MMIO_CREATELIST** создается фрагмент "LIST". При нормальном завершении работы функция возвращает нулевое значение, в противном случае - код ошибки.

Структура **MMCKINFO** и указатели на нее определены в файле **mmsystem.h** как

```
typedef  struct _MMCKINFO
{
   FOURCC  ckid;
   DWORD  cksize;
   FOURCC  fccType;
   DWORD  dwDataOffset;
   DWORD  dwFlags;
}  MMCKINFO;

typedef  MMCKINFO          *PMMCKINFO;
typedef  MMCKINFO NEAR *NPMMCKINFO:
typedef  MMCKINFO FAR    *LPMMCKINFO;
```

В нижеприведенной таблице приведено описание отдельных полей этой структуры

Идентификатор поля	*Описание*
ckid	Код, соответствующий четырехбуквенному идентификатору фрагмента
cksize	Размер фрагмента в байтах без учета идентификатора фрагмента; поля длины фрагмента и дополнительных байтов выравнивания, которые могут находиться в конце фрагмента
fccType	Тип фрагмента
dwDataOffset	Смещение области данных относительно начала файла (в байтах)
dwFlags	Нуль или **MMIO_DIRTY** (в последнем случае длина

	фрагмента может быть изменена, поэтому для ее обновления следует вызвать функцию **mmioAscend**); флаг **MMIO_DIRTY** может быть установлен при создании фрагмента функцией **mmioCreateChunk**

В нижеприведенном примере создается новый файл, подготавливается структура **MMCKINFO**, а затем создается фрагмент "RIFF", для чего вызывается функция **mmioCreateChunk**

```
hFile=mmioOpen(szFileName,  NULL,
                    MMIO_CREATE | MMIO_READWRITE);
if (hFile != NULL)
{
   ck.ckid=MMIO_CREATERIFF;
   ck.cksize=waveioocbln.lpWaveHdr->dwBytesRecorded
              + sizeof(PCMWAVEFORMAT) + 20;
ck.fccType=mmioFOURCC('W', 'A', 'V', 'E');
mmioCreateChunk(hFile,  (LPMMCKINFO) &ck, MMIO_CREATERIFF);
}
```

С целью поиска нужного фрагмента внутри RIFF-файла нет необходимости выполнять побайтное чтение файла и анализ его внутренней структуры; найти необходимый фрагмент и выполнить позиционирование относительно этого фрагмента можно с помощью функций **mmioDescend** и **mmioAscend**.

Функция **mmioDescend** ищет заданный фрагмент начиная с текущей позиции; при нахождении фрагмента текущая позиция будет установлена на область данных (она расположена на 8 байт ближе к концу файла от начала фрагмента, см. рис 3.9).

```
UINT
mmioDescend(HMMIO  hmmio, // идентификатор файла
                LPMMCKINFO  lpck, // указатель на структуру
                                  // MMCKINFO для
                                  // текущего фрагмента
                LPMMCKINFO lpckParent, // указатель на стр-ру
                                       // MMCKINFO для
                                       // внешнего фрагмента
                UINT  wFlags);  // режим поиска
```

Здесь **lpckParent** - указатель на структуру **MMCKINFO**, описывающую внешний фрагмент, внутри которого выполняется поиск. В качестве внешнего фрагмента могут выступать только фрагменты 'RIFF' или 'LIST'; если внешний фрагмент отсутствует, этот параметр можно указывать как **NULL**. Если указан **wFlags=MMIO_FINDCHUNK**, выполняется

поиск фрагмента, заданного своим идентификатором, если **wFlags=MMIO_FINDLIST** - выполняется поиск фрагмента внутри фрагмента ‘LIST’, при **wFlags=MMIO_FINDRIFF** - внутри фрагмента ‘RIFF’.

При нормальном завершении функция **mmioDescend** возвращает нулевое значение, в противном случае - код ошибки.

В нижеприведенном примере кода открывается на чтение WAV-файл, затем в нем выполняется поиск фрагментов ‘WAVE’ и ‘fmt’

```
hmmio=mmioOpen((LPSTR)lpszFileName,  NULL,
                        MMIO_READ | MMIO_ALLOCBUF);
if (!hmmio)
   return WIOERR_FILEERROR;
memset(&ckRIFF, 0, sizeof(MMCKINFO));
ckRIFF.fccType=mmioFOURCC('W', 'A', 'V', 'E');
if (mmioDescend(hmmio, &ckRIFF, NULL, MMIO_FINDRIFF))
{
   mmioClose(hmmio, 0);
   return WIOERR_BADFORMAT;
}
memset(&ckFMT, 0, sizeof(MMCKINFO));
ckFMT.ckid=mmioFOURCC('f', 'm', 't', ' ');
if (mmioDescend(hmmio, &ckFMT, &ckRIFF, MMIO_FINDCHUNK))
{
   mmioClose(hmmio, 0);
   return WIOERR_BADFORMAT;
}
```

Функция **mmioAscend** предназначена для продвижения текущей позиции к началу следующего фрагмента.

```
UINT
mmioAscend(HMMIO  hmmio, // идентификатор открытого файла
           LPMMCKINFO  lpck,  // указатель на стр-ру MMCKINFO
           UINT  wFlags);  // режим поиска
```

Структура **MMCKINFO** должна быть предварительно заполнена функцией **mmioDescend** или **mmioCreateChunk**, параметр **wFlags** не используется и должен быть нулевым.

При нормальном завершении функция **mmioAscend** возвращает нулевое значение, в противном случае - код ошибки.

3.6.3. Программное определение возможностей звуковых устройств мультимедиа

В системе могут быть установлены устройства для записи и воспроизведения звука методом импульсно-кодовой модуляции PCM (waveform audio), устройства для записи и проигрывания музыкальных MIDI-файлов, дополнительные (auxiliary) устройства (например, проигрыватель звуковых компакт-дисков) и другие.

Библиотека **mmsystem.dll** содержит набор функций, с помощью которых приложение может определить состав устройств и их возможности.

Не имеющая параметров функция **waveOutGetNumDevs** возвращает количество устройств, способных воспроизводить записанные с использование импульсно-кодовой модуляции звуковые данные; функция **waveInGetNumDevs** возвращает количество устройств, способных записывать такие данные.

Количество пригодных для записи и воспроизведения MIDI-файлов устройств можно узнать при помощи функций **midiOutGetNumDevs** и **midiInGetNumDevs** соответственно.

Для определения дополнительных устройств предназначена функция **auxGetNumDevs**.

Для определения возможностей устройств используются функции **waveOutGetDevCaps** (возможности устройств вывода данных, записанных методом импульсно-кодовой модуляции), **waveInGetDevCaps** (возможности устройств записи данных методом импульсно-кодовой модуляции), **midiOutGetDevCaps** (возможности устройств воспроизведения в формате MIDI) и **auxGetDevCaps** (возможности дополнительных устройств).

Для вышеперечисленных функций с суффиксом **GetDevCaps** в качестве первого параметра указывается идентификатор устройства, изменяющийся от нуля (для первого устройства) до полученного с помощью функций **waveOutGetNumDevs**, **waveInGetNumDevs**, **midiOutGetNumDevs**, **midiInGetNumDevs** и **auxGetNumDevs** значения.

Второй параметр является дальним указателем на структуру, формат которой зависит от типа устройства. Это может быть структура **AUXCAPS** (дополнительное устройство), **MIDIINCAPS** (устройство для ввода данных MIDI), **MIDIOUTCAPS** (устройство вывода MIDI-данных), **WAVEINCAPS** (устройство ввода данных, записанных методом импульсно-кодовой модуляции) или **WAVEOUTCAPS** (устройство вывода данных, записанных методом импульсно-кодовой модуляции).

Третий параметр определяет размер соответствующей структуры в байтах.

Структура **AUXCAPS** определяется следующим образом

```
typedef  struct  auxcaps_tag
{
   UINT  wMid;  // код разработчика драйвера
   UINT  wPID;  // код устройства
```

```
   VERSION  vDriverVersion;  // версия драйвера
   char  szPname[MAXPNAMELEN];  // название устройства
   UINT  wTechnology;  // тип устройства
   DWORD  dwSupport;  // поддерживаемые функции
}  AUXCAPS;

typedef  AUXCAPS          *PAUXCAPS;
typedef  AUXCAPS  NEAR  *NPAUXCAPS;
typedef  AUXCAPS  FAR    *LPAUXCAPS;
```

Поля **wMid**, **wPid**, **vDriverVersion** и **szPname** определены во всех структурах, используемых для определения возможностей устройств мультимедиа.

Поле **wTechnology** специфично для структуры **AUXCAPS**. В нем могут быть установлены флаги **AUXCAPS_AUXIN** (имеется звуковой вход от внутреннего устройства проигрывания компакт-дисков) и **AUXCAPS_AUXIN** (предусмотрен звуковой вход от входной линии, расположенной на плате звукового адаптера).

Поле **dwSupport** может содержать флаги **AUXCAPS_VOLUME** (есть возможность регулировки громкости) и **AUXCAPS_LRVOLUME** (есть возможность раздельной регулировки громкости для левого и правого канала).

Структура **MIDIINCAPS** содержит только те поля, которые являются общими для всех структур, предназначенных для определения возможностей устройств мультимедиа

```
typedef  struct  midiincaps_tag
{
   UINT  wMid;
   UINT  wPid;
   VERSION  vDriverVersion;
   char  szPname[MAXPNAMELEN];
}  MIDIINCAPS;

typedef  MIDIINCAPS          *PMIDIINCAPS;
typedef  MIDIINCAPS  NEAR *NPMIDIINCAPS;
typedef  MIDIINCAPS  FAR   *LPMIDIINCAPS;
```

Структура **MIDIOUTCAPS** дополнительно содержит поля **wTechnology** (тип устройства), **wVoices** (количество голосов для встроенного синтезатора), **wChannelMask** (количество каналов для встроенного синтезатора) и **dwSupport** (поддерживаемые функции)

```
typedef  struct  midioutcaps_tag
{
```

```
   UINT  wMid;
   UINT  wPid;
   VERSION  vDriverVersion;
   char  szPname[MAXPNAMELEN];
   UINT  wTechnology;
   UINT  wVoices;
   UINT  Notes;
   UINT  wChannelMask;
   DWORD  dwSupport;
} MIDIOUTCAS;

typedef  MIDIOUTCAPS          *PMIDIOUTCAPS;
typedef  MIDIOUTCAPS  NEAR *NMIDIOUTCAPS:
typedef  MIDIOUTCAPS  FAR   *LPMIDIOUTCAPS;
```

В поле **wTechnology** могут находиться значения **MOD_MIDIPORT** (устройство является аппаратным портом MIDI), **MOD_SQSYNTH** (устройство является синтезатором с выходным сигналом прямоугольной формы), **MOD_FMSYNTH** (FM-синтезатор - т.е. синтезатор с частотной модуляцией) и **MOD_MAPPER** (устройство отображения Microsoft MIDI Mapper).

Структура **WAVEINCAPS** предназначена для определения возможностей устройств ввода звуковых сигналов с использованием импульсно-кодовой модуляции

```
typedef  struct waveincaps_tag
{
   UINT  wMid;
   UINT  wPid;
   VERSION  vDriverVersion;
   char  szPname[MAXPNAMELEN];
   UINT  wFormats;
   UINT  wChannels;
} WAVEINCAPS;

typedef  WAVEINCAPS          *PWAVEINCAPS;
typedef  WAVEINCAPS  NEAR  *NPWAVEINCAPS;
typedef  WAVEINCAPS  FAR    *LPWAVEINCAPS;
```

В поле **wChannels** содержится количество каналов (1 для моно и 2 для стерео).

В поле **dwFormats** могут располагаться флаги, соответствующие стандартным форматам звуковых данных, поддерживаемых данным устройством. Флаги объединяются при помощи логической операции ИЛИ, для них в файле **mmsystem.h** определены нижеследующие символические константы

Константа	*Частота дискретизации, количество каналов (моно, стерео) и количество бит для представления сигнала*
WAVE_FORMAT_1M08	11,025 kГц, моно, 8 бит
WAVE_FORMAT_1S08	11,025 kГц, стерео, 8 бит
WAVE_FORMAT_1M16	11,025 kГц, моно, 16 бит
WAVE_FORMAT_1S16	11,025 kГц, стерео, 16 бит
WAVE_FORMAT_2M08	22,5 kГц, моно, 8 бит
WAVE_FORMAT_2S08	22,5 kГц, стерео, 8 бит
WAVE_FORMAT_2M16	22,5 kГц, моно, 16 бит
WAVE_FORMAT_2S16	22,5 kГц, стерео, 16 бит
WAVE_FORMAT_4M08	44,1 kГц, моно, 8 бит
WAVE_FORMAT_4S08	44,1 kГц, стерео, 8 бит
WAVE_FORMAT_4M16	44,1 kГц, моно, 16 бит
WAVE_FORMAT_4S16	44,1 kГц, стерео, 16 бит

Структура **WAVEOUTCAPS** предназначена для определения возможностей устройств вывода звуковых сигналов с использованием импульсно-кодовой модуляции

```
typedef  struct waveoutcaps_tag
{
   UINT  wMid;
   UINT  wPid;
   VERSION  vDriverVersion;
   char  szPname[MAXPNAMELEN];
   UINT  wFormats;
   UINT  wChannels;
   DWORD  dwSupport;
}  WAVEOUTCAPS;

typedef  WAVEOUTCAPS          *PWAVEOUTCAPS;
typedef  WAVEOUTCAPS  NEAR  *NPWAVEOUTCAPS;
typedef  WAVEOUTCAPS  FAR    *LPWAVEOUTCAPS;
```

Поле **dwSupport** содержит флаги, соответствующие различным возможностям вывода, символические константы для них определены в файле **mmsystem.h** и представлены ниже

Константа	*Описание*
WAVECAPS_PITCH	Поддерживается изменение высоты тона
WAVECAPS_PLAYBACKRATE	Изменение скорости проигрывания
WAVECAPS_VOLUME	Управление громкостью
WAVECAPS_LRVOLUME	Раздельное управление громкостью для левого и правого каналов

WAVECAPS_SYNC	Драйвер устройства вывода работает в синхронном режиме (во время проигрывания работа приложения приостанавливается)

В работе [6] приведен полный С-код приложения **DRVLIST**, служащего для исследования установленных в системе конфигурации драйверов устройств мультимедиа.

3.6.4. **Воспроизведение звука**

Для воспроизведения звуковых файлов на низком уровне после определения возможностей устройства вывода необходимо открыть устройство, это удобно сделать с помощью функции **waveOutOpen**.

```
UINT
waveOutOpen(LPHWAVEOUT  lphWaveOut,
            UINT  wDeviceID,
            LPWAVEFORMAT  lpFormat,
            DWORD  dwCallbackInstance,
            DWORD  dwFlags);
```

Здесь **lphWaveOut** - дальний указатель на переменную типа **HWAVEOUT.** В эту переменную будет записан идентификатор устройства вывода, который необходим для выполнения всех операций с устройством. Функция **waveOutOpen** также может быть использована для определения возможности воспроизведения звуковых данных заданного формата (в том числе нестандартного), в этом случае параметр **lphWaveOut** может иметь значение **NULL**, дополнительно в параметре **dwFlags** следует установить флаг **WAVE_FORMAT_QUERY**.

Через параметр **wDeviceID** приложение должно передать функции **waveOutOpen** номер устройства вывода, которое оно собирается открыть или константу **WAVE_MAPPER**, определенную в файле **mmsystem.h**. В первом случае номер устройства может лежать в пределах от нуля до полученного с помощью функции **waveOutGetNumDevs** значения. Обычно используется константа **WAVE_MAPPER**, при этом функция **waveOutOpen** пытается самостоятельно выбрать и открыть устройство вывода, подходящее для проигрывания звуковых данных указанного формата.

Через параметр **lpFormat** приложение должно передать функции **waveOutOpen** адрес заполненной структуры **WAVEFORMAT** (эта структура и указатели на нее описаны выше).

Параметр **dwCallback** передает функции **waveOutOpen** адрес функции обратного вызова. Эту функцию будет вызывать драйвер устройства вывода при возникновении событий, имеющих отношение к проигрыва-

нию блока данных. При использовании функции обратного вызова в параметре **dwFlags** следует установить флаг **CALLBACK_FUNCTION**.

При использовании функции обратного вызова могут возникнуть проблемы, связанные с имеющимися ограничениями на функционирование подобных функций; поэтому с целью извещения приложения о возникновении события заключается в посылке сообщений оконной функции. Для этого параметр **dwCallback** должен содержать идентификатор окна, а в параметре **dwFlags** необходимо установить флаг **CALLBACK_WINDOW**.

Параметр **dwCallbackInstance** является идентификатором данных, передаваемым функции обратного вызова (этот параметр не используется совместно с флагом **CALLBACK_WINDOW**).

В поле **dwFlags** можно указывать следующие флаги

Идентификатор флага	*Описание*
WAVE_FORMAT_QUERY	Функция **waveOutOpen** вызывается только с целью проверки возможности использования формата звуковых данных, определенных в структуре **WAVEFPRMAT**, адрес которой передается через параметр **lpFormat**. Этим способом можно проверить, способно ли данное устройство работать с нестандартной частотой дискретизации
WAVE_ALLOWSYNC	Флаг необходимо использовать для открытия синхронного устройства вывода, во время работы которого все приложения блокируются
CALLBACK_WINDOW	Для извещения о наступлении событий используется окно, идентификатор которого передается через параметр **dwCallback**
CALLBACK_FUNCTION	Для извещения о наступлении событий используется функция обратного вызова, адрес которой передается через параметр **dwCallback**

При нормальном завершении функция **waveOutOpen** возвращает нулевое значение, в противном случае - описанный в нижележащей таблице код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_BADDEVICEID	Указан неправильный номер устройства вывода
MMSYSERR_ALLOCATED	Это устройство уже открыто
MMSYSERR_NOMEM	Для выполнения операции недостаточно памяти
WAVEERR_BADFORMAT	Указанный формат звуковых данных не поддерживается драйвером устройства вывода
WAVEERR_SYNC	Была выполнена попытка открыть синхронное устройство вывода без использования

	флага **WAVE_ALLOWSYNC**

Обычно при проигрывании WAV-файлов приложение вызывает функцию **waveOutOpen** дважды. Первый раз она вызывается для проверки возможности проигрывания звуковых данных заданного формата

```
if (waveOutOpen(NULL, WAVE_MAPPER,
                (WAVEFORMAT FAR *) lpwiocb->lpFmt,
                NULL, 0L,
                WAVE_FORMAT_QUERY | WAVE_ALLOWSYNC))
{
   // сообщение о том, что воспроизведение невозможно
}
```

Если указанный формат поддерживается драйвером, приложение может открыть устройство вывода, например, следующим образом

```
rc=waveOutOpen(&hWaveOut,  WAVE_MAPERR,
                (WAVEFORMAT FAR *) lpwiocb->lpFmt,
                (UINT) hwnd, 0L,
                CALLBACK_WINDOW | WAVE_ALLOWSYNC);
```

Указанная методика позволяет определить возможность работы с нестандартными форматами.

После открытия устройства можно приступать собственно к проигрыванию WAV-файла или звуковых данных; для чего необходимо подготовить и передать драйверу устройства вывода блоки данных, содержащие звуковую информацию (формат этих данных должен соответствовать указанному при открытии устройства).

Блоки данных, передаваемые драйверу, должны быть заказаны у системы как глобальные с флагами **GMEM_MOVEABLE** и **GMEM_SHARE**.

Перед передачей блока драйверу необходимо его подготовить при помощи функции **waveOutPrepareHeader**

```
UINT
waveOutPrepareHeader(HWAVEOUT hWaveOut,
                     LPWAVEHDR  lpWaveOutHdr,
                     UINT  wSize);
```

Здесь **hWaveOut** - полученный ранее от функции **waveOutOpen** идентификатор устройства вывода, **lpWaveOutHdr** - адрес описывающей передаваемый блок данных заполненной структуры **WAVEHDR**; описание данной структуры приведено ниже

```
typedef  struct  wavehdr_tag
{
   LPSTR  lpData;  // адрес блока данных
   DWORD  dwBufferLength;  // размер блока данных
   DWORD  dwBytesRecorded;  // количество записанных байт
                                 //  (только при записи)
   DWORD  dwUser;  // пользовательские данные
   DWORD  dwFlags;  // флаги состояния буфера данных
   DWORD  dwLoops;  // кратность проигрывания файла
                          //  (только при воспроизведении)
   struct  wavehdr_tag  far *lpNext;  // зарезервировано
   DWORD  reserved;  // зарезервировано (не используется)
}  WAVEHDR;

typedef  WAVEHDR          *PWAVEHDR;
typedef  WAVEHDR  NEAR *NPWAVEHDR;
typedef  WAVEHDR  FAR   *LPWAVEHDR;
```

После заказа блока памяти функцией **GlobalAlloc** с флагами **GMEM_MOVEABLE** и **GMEM_SHARE** необходимо зафиксировать его функцией **GlobalLock**; полученный в результате фиксирования адрес блока записывается в поле **lpData** структуры **WAVEHDR**, а размер блока - в поле **dwBufferLength**.

Теоретически размер блока памяти для воспроизводимых данных может иметь очень большой размер (определяется двойным словом), однако существует практическое ограничение на размер блока памяти - он должен помещаться целиком в оперативную память (иначе его невозможно зафиксировать). Поэтому в некоторых случаях (при проигрывании *очень* длинных WAV-файлов) имеет смысл создать два или более блоков, попеременно заполняя их из файла с аудиоданными и передавая драйверу для проигрывания в асинхронном режиме.

Структура **WAVEHDR** используется не только для воспроизведения, но и для записи; в этом случае после завершения записи блока в поле **dwBytesRecorded** будет находиться количество записанных байт данных (при воспроизведении это поле не используется).

Через поле **dwUser** приложение может передать функции обратного вызова или обработчику сообщения для данного устройства любую дополнительную информацию.

Поле **dwFlag** после прихода сообщения о событии или передачи управления функции обратного вызова будет содержать информацию о состоянии блока; в этом случае могут быть установлены следующие флаги

Идентификатор флага	*Описание*
WHDR_DONE	Работа с буфером данных закончена; данные из буфера были успешно проиграны или записаны, после чего драйвер

	вернул буфер приложению
WHDR_BEGINLOOP	Данный буфер является первым в цикле (флаг используется только при воспроизведении). При желании проиграть в цикле только один блок он должен быть отмечен и флагом **WHDR_BEGINLOOP** и флагом **WHDR_ENDLOOP**
WHDR_PREPARED	Буфер подготовлен для воспроизведения функцией **waveOutPrepareHeader** или для записи функцией **waveIn-PrepareHeader**

Приложение может указать драйверу, что блок необходимо проиграть несколько раз подряд, для этого следует указать число повторов в поле **dwLoops.**

При нормальном завершении функция **waveOutPrepareHeader** возвращает нуль, в противном случае указанный в таблице код ошибки.

Код ошибки	*Описание ошибки*
MMSYSERR_INVALHANDLE	Указан неправильный идентификатор устройства
MMSYSERR_NOMEM	Недостаток памяти для выполнения операции

После обработки блока памяти функцией **waveOutPrepareHeader** его можно проиграть с помощью вызова функции **waveOutWrite**.

```
UINT
waveOutWrite(HWAVEOUT  hWaveOut,  // идентификатор устройства
             LPWAVEHDR  lpWaveOutHdr, // указатель на ст-ру WAVEHDR
             UINT  wSize);  // размер структуры WAVEHDR
```

При нормальном завершении функция **waveOutWrite** возвращает нуль, в противном случае нижеуказанный код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_INVALHANDLE	Указан неправильный идентификатор устройства
MMSYSERR_UNPREPARED	Переданный блок функции не был подготовлен функцией **waveOutPrepareHeader**

Реально проигрывание блока начинается после вызова функции **waveOutWrite**, в случае проигрывания блока до конца или остановки проигрывания определенная указанным при открытии устройства через параметр **dwCallback** идентификатором оконная функция получит сообщение **MM_WOM_DONE**.

После получения приложением сообщения **MM_WOM_DONE** оно должно передать блок функции **waveOutUnprepareHeader**, затем разблокировать его функцией **GlobalUnlock** и освободить (если этот блок памяти больше не нужен) функцией **GlobalFree**.

Формат вызова функции **waveOutUnprepareHeader** приведен ниже

```
UINT
waveOutUnprepareHeader(HWAVEOUT  hWaveOut,
                       LPWAVEHDR  lpWaveOutHdr,
                       UINT  wSize);
```

При нормальном завершении функция **waveOutUnprepareHeader** возвращает нуль, в противном случае нижеуказанный код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_INVALHANDLE	Указан неправильный идентификатор устройства
MMSYSERR_STILLPLAYING	Указанный блок все еще находится в очереди для проигрывания

После завершения работы с устройством его необходимо закрыть, вызвав функцию **waveOutClose**

```
UINT
waveOutClose(HWAVEOUT hWaveOut);  // идентификатор  устройства
```

При нормальном завершении функция **waveOutClose** возвращает нуль, в противном случае нижеуказанный код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_INVALHANDLE	Указан неправильный идентификатор устройства
MMSYSERR_STILLPLAYING	Очередь данного устройства еще содержит блоки для проигрывания

3.6.5.**Запись звука**

Процесс записи имеет много общего с процессом воспроизведения. В начале необходимо открыть устройство записи, вызвав функцию **waveInOpen**

```
UINT
waveInOpen(LPHWAVEIN  lpWaveIn,
           UINT  wDeviceID,
           LPWAVEFORMAT  lpFormat,
           DWORD  dwCallback,
           DWORD  dwFlags);
```

Здесь **lpWaveOut** - дальний указатель на переменную типа **HWAVEIN**. В эту переменную будет записан идентификатор устройства

вывода, который необходим для выполнения всех операций с устройством. Функция **waveInOpen** может быть использована для определения возможностей записи звуковых данных в заданном формате (возможно, нестандартном), в этом случае параметр **lphWaveIn** может иметь значение **NULL** (дополнительно в параметре **dwFlags** следует установить флаг **WAVE_FORMAT_QUERY**).

Через параметр **wDeviceID** приложение должно передать функции **waveInOpen** номер устройства ввода, которое оно собирается открыть или константу **WAVE_MAPERR** (в первом случае номер устройства может лежать от нуля до полученного с помощью функции **waveInGetNumDevs**, во втором случае функция **waveInOpen** пытается самостоятельно выбрать и открыть подходящее для записи звуковых данных указанного формата устройство вывода).

При нормальном завершении функция **waveInOpen** возвращает нуль, в противном случае - нижеприведенный код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_NODRIVER	В системе нет нужного для работы с устройством ввода драйвера
MMSYSERR_BADDEVICEID	Указан неправильный номер устройства
MMSYSERR_ALLOCATED	Это устройство уже открыто
MMSYSERR_NOMEM	Для выполнения операции недостаточно памяти
MMSYSERR_BADFORMAT	Указанный формат звуковых данных не поддерживается драйвером устройства
MMSYSERR_SYNC	Попытка открыть синхронное устройство ввода без использования флага **WAVE_ALLOWSYNC**

После открытия устройства необходимо подготовить один или несколько блоков памяти, в которые будет записана введенная звуковая информация (требования к блокам памяти при этом такие же, как и при воспроизведении).

Перед передачей блока драйверу его необходимо подготовить функцией **waveInPrepareHeader**, входные параметры и коды ошибок которой практически повторяют таковые функции **waveOutPrepareHeader**

```
UINT
waveInPrepareHeader(HWAVEIN hWaveIn,
                    LPWAVEHDR  lpWaveInHdr,
                    UINT  wSize);
```

Подготовленный таким образом блок памяти передается драйверу устройства ввода функцией **waveInAddBuffer**

```
UINT
```

```
waveInAddBuffer(HWAVEIN  hWaveIn,
                LPWAVEHDR  lpWaveInHdr,
                UINT  wSize);
```

При нормальном завершении функция **waveInAddBuffer** возвращает нуль, в противном случае - нижеприведенный код ошибки

Код ошибки	*Описание ошибки*
MMSYSERR_INVALHANDLE	Указан неправильный идентификатор устройства
MMSYSERR_UNPREPARED	Переданный блок данных не был подготовлен функцией **waveInPrepareHeader**

Для реального начала записи необходимо вызвать функцию **waveIn-Start**

```
UINT
waveInStart(HWAVEIN  hWaveIn);
```

В качестве входного параметра передается полученный ранее от функции **waveInOpen** идентификатор вводного устройства. При нормальном завершении функция возвращает нуль, иначе - код **MMSYSERR_INVALHANDLE**, означающий указание неправильного идентификатора устройства.

Запись будет производиться до тех пор, пока не будет записан весь буфер или пока устройство ввода не будет остановлено функцией **waveIn-Stop**

```
UINT
waveInStop(HWAVEIN  hWaveIn);
```

При нормальном завершении функция **waveInStop** возвращает нуль, иначе - код **MMSYSERR_INVALHANDLE**, означающий указание неправильного идентификатора устройства.

При записи блока до конца или при принудительной остановке записи вызывается оконная функция (как в случае функции **waveOutWrite**). Вызовы **waveUnprepareHeader**, **GlobalUnlock**, **GlobalFree** и **waveInClose** синтаксически подобны описанным выше.

3.6.6. **Дополнительные функции низкого уровня**

Библиотека **mmsystem.dll** содержит несколько весьма полезных функций, предназначенных для работы со звуком на низком уровне.

Функции **waveInGetErrorText** и **waveOutGetErrorText** (подобно функции **mciGetErrorString**) преобразуют код ошибки в текстовое описание ее в виде строки символов.

Функции **waveInGetID** и **waveInGetID** позволяют определить реальный номер устройства, выбранного функцией открытия устройства ввода/вывода с указанием константы **WAVE_MAPPER** (режим автоматического поиска подходящего из установленных в системе устройств).

Функции **waveInReset** и **waveOutReset** выполняют останов устройств ввода или вывода соответственно и сброс текущей позиции для устройства в нуль.

Для запуска устройства ввода используется ранее рассмотренная функция **waveInStart**, для продолжения работы ранее приостановленного устройства вывода используется функция **waveOutRestart**.

Останов устройства ввода используется ранее описанная функция **waveInStop**, временный останов работы устройства вывода следует воспользоваться функцией **waveOutPause**.

Определение текущей позиции в блоке при записи и воспроизведении производится функциями **waveInGetPosition** и **waveOutGetPosition** соответственно (при этом используется структура **MMTIME**, описанная в файле **mmsystem.h**)

```
typedef  struct  mmtime_tag
{
   UINT  wType;  // формат времени
   union
   {
    DWORD  ms;  // миллисекунды
    DWORD  sample;  // выборки сигнала
    DWORD  cb;  // счетчик байт
    struct  // формат  SMPTE
    {
     BYTE  hour;  // часы
     BYTE  min;  // минуты
     BYTE  sec;  // секунды
     BYTE  frame;  // фреймы
     BYTE  fps;  // фреймы в секунду
     BYTE  dummy;  // байт для выравнивания
    }  smpte;
    struct   // формат MIDI
    {
     DWORD  songptpos;  // указатель позиции в мелодии
    }  MIDI;
   }  u;
}  MMTIME;

typedef  MMTIME          *PMMTIME;
```

```
typedef  MMTIME  NEAR  *NPMMTIME;
typedef  MMTIME  FAR   *LPMMTIME;
```

Для установки громкости следует использовать функцию **waveOutSetVolume**, прочитать текущее значение громкости можно функцией **waveOutGetVolume**.

В работе [6] приведен полный С-код приложения **WAVE,** служащего для записи и воспроизведения WAV-файлов.

3.7. **Управление устройством CD-ROM**

Как было сказано, данные на компакт-диске записаны вдоль одной спирали; в первом приближении можно считать, что на музыкальных дисках эта спираль (логически) разбита на несколько участков (дорожек), каждая из которых содержит отдельную запись (например, музыкальное произведение).

Приложение имеет возможность устанавливать лазерное устройство чтения в произвольное место спирали, причем драйвер обеспечивает позиционирование в режиме прямого д**о**ступа как на начало любой дорожки, так и в произвольную позицию внутри дорожки (позиционирование может длиться до 0,5 сек).

Устройство чтения CD-ROM имеет два звуковых выхода. Один из них обычно расположен на лицевой панели и предназначен для подключения головных телефонов (там же находится регулятор громкости), второй выведен на заднюю панель и подключается кабелем к входу звукового адаптера. Приложения мультимедиа поэтому могут выполнять не только проигрывание звуковых компакт-дисков, но и синхронную запись WAV-файлов.

Работа с устройством CD-ROM возможна при помощи интерфейса управляющих строк MCI или интерфейса управляющих сообщений MCI.

4.7.1. **Интерфейс управляющих строк MCI**

Для передачи управляющей строки устройству чтения CD-ROM необходимо использовать функцию **mciSendString** (подробнее см. подраздел 2.4).

Перед началом работы с устройством необходимо его открыть, передав управляющую строку **open**, указав при этом имя устройства как **cdaudio** (допускается использование алиаса - альтернативного имени)

```
open  cdaudio  alias  cd  wait
```

Операция открытия устройства CD-ROM может выполняться несколько секунд, поэтому целесообразно перед продолжением работы приложения дождаться ее завершения, указав параметр **wait**. С целью достижения возможности использования данного устройства несколькими приложениями одновременно можно указать параметр **shareable** (этот параметр должны указать все приложения, желающие иметь одновременный доступ к устройству). Так как драйвер устройства CD-ROM не работает с файлами, путь к конкретному файлу в управляющей строке **open** не указывается.

Команда **close** служит для закрытия устройства, особенностей не имеет, в качестве параметра необходимо указать имя устройства **cdaudo** (или алиас, если устройство было открыто с использованием альтернативного имени)

close cd

Справочные команды. Команда **sysinfo** требует один параметр - строку **cdaudio** (даже если при открытии был использован алиас); для команды **info** можно указывать только параметр **product**.

Команда **capability** с параметром **can eject** позволяет определить, имеет ли устройство CD-ROM возможность автоматического извлечения компакт-дисков. Можно использовать иные параметры (с целью определения соответствующих возможностей устройства) - **can play**, **can record**, **can save**, **compound device**, **device type**, **has audio**, **has video**, **uses file**.

Для определения текущего состояния CD-ROM следует использовать команду **status**, можно указывать следующие параметры

Параметр команды **status**	*Что определяется*
current track	Номер текущей дорожки
length	Общая длина
length track *track_number*	Длина заданной дорожки
media present	При наличии в устройстве вставленного компакт-диска возвращается строка **true**
mode	Текущий режим работы - **not ready** (не готов), **playing** (проигрывание), **stopped** (останов), **recording** (запись), **seeking** (позиционирование)
number of track	Количество дорожек
position	Текущая позиция
position track *track_number*	Текущая позиция на заданной дорожке
ready	При готовности устройства возвращается строка **true**
start position	Начальная позиция
time formar	Текущий формат времени

Команды установки режимов работы. Команда **break** позволяет определить код виртуальной клавиши, предназначенный для прерывания процесса выполнения текущей команды (по умолчанию используется комбинация клавиш **<Control+Break>**).

Для команды **set** можно указывать следующие параметры

Параметр команды **set**	*Действие*
audio all off	Отключение звукового выхода
audio all on	Включение звукового выхода
audio left off	Отключение левого канала
audio left on	Включение левого канала
audio right off	Отключение правого канала
audio right on	Включение правого канала
door closed	Загрузка компакт-диска и фиксирование его в устройстве (поддерживается не всеми устройствами)
door open	Извлечение компакт-диска (поддерживается не всеми устройствами)
time format miiliseconds	В качестве единицы измерения при позиционировании используются миллисекунды (вместо **miiliseconds** можно использовать **ms**)
time format msf	В качестве единицы измерения при позиционировании используются минуты, секунды и фреймы (разделитель - двоеточие); этот формат используется по умолчанию
time format tmsf	В качестве единицы измерения при позиционировании используются дорожки, минуты, секунды и фреймы

Команды для воспроизведения, записи и позиционирования. Команда **play** предназначена для запуска проигрывания и имеет следующий формат (необязательные параметры отмечены квадратными скобками)

play *device_id* [from position [to position]] [notify] [wait]

Здесь и далее **device_id** - идентификатор устройства, использованный при открытии устройства командой **open** (например, если устройство было открыто командой **open c:\windows\ding.wav alias ding**, то в качестве параметра **device_id** можно использовать алиас **ding**) .

Если не указан параметр **from position**, проигрывание начинается с текущей позиции (сразу после открытия текущая позиция устанавливается в начало первой дорожки); параметр **to position** позволяет указать конечную позицию, при достижении которой проигрывание прекращается (перед использованием параметров **from ...** и **to ...** необходимо установить формат для позиционирования при помощи команды **set**).

В нижеследующем примере выполняется проигрывание 10-й дорожки звукового компакт-диска

```
open cdaudio alias cd wait
set cd time format tmsf wait
play cd from 10 to 11 wait
close cd
```

Нижеследующая команда проигрывает фрагмент 12-й дорожки, начало которого отстоит на 10 сек от начала дорожки, а длительность составляет 16 сек

```
play cd from 12:0:10 to 12:0:16 wait
```

Останов проигрывания реализуется командой **stop**, единственным параметром которой является идентификатор устройства ***device_id***.

Команда **pause** ***device_id*** для устройства чтения CD-ROM работает как команда полного останова **stop**, при этом команда продолжения работы после останова **resume** не поддерживается.

Позиционирование с последующим остановом реализуется командой **seek** (перед использованием этой команды необходимо задать формат времени командой **set time format**)

seek ***device_id*** **parameter [notify] [wait]**

Параметр **parameter** является необязательным и может принимать значение одной из нижеуказанных строк

Значение **parameter**	*Описание*
to position	Позиционирование в заданное место компакт-диска
to start	Позиционирование в начало
to end	Позиционирование в конец

3.7.2. **Интерфейс управляющих сообщений MCI**

Большинство разработанных с использованием языка C/C++ приложений используют для управления устройством CD-ROM интерфейс управляющих сообщений MCI (напомним, что управляющие сообщения при этом посылаются устройствам с помощью функции **mciSendCommand**, см. подраздел 2.5).

Команда **MCI_OPEN** открывает устройство, предварительно необходимо подготовить структуру **MCI_OPEN_PARMS** и передать ее адрес через четвертый параметр функции **mciSendCommand**.

Поле **lpstrDeviceType** структуры **MCI_OPEN_PARMS** содержит указатель на строку имени устройства или его идентификатор. Для устройства

чтения CD-ROM можно указывать имя "cdaudio" или константу **MCI_DEVTYPE_CD_AUDIO**. Параметр **lpstrElementName** не используется, так как устройство чтения компакт-дисков не работает с файлами.

Приведенный ниже фрагмент кода открывает устройство чтения компакт-дисков

```
MCIOpen.lpstrDeviceType=(LPSTR) MCI_DEVTYPE_CD_AUDIO;
dwrc=maciSendCommand(NULL,  MCI_OPEN,
                              MCI_OPEN_TYPE | MCI_OPEN_TYPE_ID,
                              (DWORD) (LPVOID) &MCIOpen);
```

После выполнения этого фрагмента кода в переменную **dwrc** будет записан код результата выполнения, при успешном выполнении в поле **wDeviceID** структуры **mciOpen** будет находиться идентификатор открытого устройства.

Команда **MCI_CLOSE** закрывает устройство, **MCI_PLAY** запускает проигрывание, **MCI_STOP** останавливает выполнение проигрывания компакт-диска, **MCI_PAUSE** действует аналогично **MCI_STOP**, **MCI_SEEK** выполняет позиционирование, **MCI_BREAK** устанавливает виртуальный код клавиши прерывания операции (по умолчанию **<Control+Break>**).

С помощью команды **MCI_GETDEVCAPS** можно определить возможности устройства чтения компакт-дисков. Для этого используется блок параметров в формате нижеописанной структуры **MCI_GETDEVCAPS_PARMS**

```
typedef  struct  tagMACI_GETDEVCAPS_PARMS
{
   DWORD  dwCallback;
   DWORD  dwReturn;
   DWORD  dwItem;
}  MCI_GETDEVCAPS_PARMS;

typedef  MCI_GETDEVCAPS_PARMS  FAR  \
         *LPMCI_GETDEVCAPS_PARMS;
```

В поле **dwReturn** после возврата из функции **mciSendCommand** будет записано значение требуемого параметра (код нужного параметра следует записать в поле **dwItem** перед вызовом функции **mciSendCommand**)

Значение параметра **dwItem**	*Описание*
MCI_GETDEVCAPS_CAN_EJECT	Если устройство может выталкивать компакт-диск, после возврата из функции **mciSendCommand** в поле **dwReturn** будет ненулевое значение **TRUE**

MCI_GETDEVCAPS_CAN_PLAY	Устройство может проигрывать
MCI_GETDEVCAPS_CAN_RECORD	Устройство может записывать
MCI_GETDEVCAPS_CAN_SAVE	Устройство может сохранять записанные данные в файле
MCI_GETDEVCAPS_COMPOUND_ DEVICE	Устройство может работать с файлами
MCI_GETDEVCAPS_DEVICE_TYPE	Требуется определить тип устройства
MCI_GETDEVCAPS_HAS_AUDIO	Устройство имеет звуковой выход
MCI_GETDEVCAPS_HAS_VIDEO	Устройство имеет видеовыход
MCI_GETDEVCAPS_USES_FILES	При открытии устройства требуется указывать имя файла

Команда **MCI_INFO** дает возможность получить информацию об устройстве чтения CD-ROM в виде текстовой строки, при этом используется блок параметров в виде структуры **MCI_INFO_PARMS**

```
typedef struct tagMCI_INFO_PARMS
{
   DWORD dwCallback;
   LPSTR lpstrReturn;
   DWORD dwRetSize;
} MCI_INFO_PARMS;

typedef MCI_INFO_PARMS FAR *LPMCI_INFO_PARMS;
```

Поле **lpstrReturn** должно содержать указатель на буфер, в который будет записана строка информации, размер этого буфера следует передать через поле **dwRetSize**. Ниже приведен набор флагов для команды **MCI_INFO**, допустимых к использованию при работе с устройством чтения компакт-дисков

Идентификатор флага	*Описание*
MCI_NOTIFY	Если установлен этот флаг, после завершения команды оконной функции, адрес которой передан через поле **dwCallback**, будет послано сообщение **MM_MCINOTIF**
MCI_WAIT	Функция **mciSendCommand** вернет управление только после завершения процесса
MCI_INFO_PRODUCT	Требуется получить описание аппаратуры устройства

Команда **MCI_SYSINFO** позволяет получить системную информацию об устройстве в виде текстовой строки, команда **MCI_STATUS** используется для определения текущего состояния устройства (формат соответствующего блока параметров описывается структурой **MCI_STATUS_PARMS**)

```
typedef struct tagMCI_STATUS_PARMS
{
   DWORD dwCallback;
   DWORD dwReturn;
   DWORD dwItem;
   DWORD dwTrack;
} MCI_STATUS_PARMS;

typedef MCI_STATUS_PARMS FAR *LPMCI_STATUS_PARMS;
```

Через поле **dwReturn** передается возвращаемая информация, вид запрашиваемой информации определяется содержимым поля **dwItem**. Для устройства чтения компакт-дисков в поле **dwTrack** можно указать размер или номер дорожки (ниже приведены возможные значения параметра **dwItem**)

Значение параметра **dwItem**	*Описание получаемой информации*
MCI_STATUS_CURRENT_ TRACK	Номер текущей дорожки
MCI_STATUS_LENGTH	Общий размер всех дорожек компакт-диска
MCI_STATUS_NUMBER_ OF_TRACKS	Общее количество дорожек, которые можно проиграть
MCI_STATUS_POSITION	Текущая позиция
MCI_STATUS_READY	При готовности устройства возвращается **TRUE**, иначе - **FALSE**
MCI_STATUS_MODE	Текущий режим устройства; может иметь следующие значения - **MCI_MODE_NOT_READY** (не готово), **MCI_MODE_PAUSE** (пауза), **MCI_MODE_PLAY** (проигрывание), **MCI_MODE_STOP** (останов), **MCI_MODE_OPEN** (открывание), **MCI_MODE_RECORD** (запись), **MCI_MODE_SEEK** (позиционирование)
MCI_STATUS_TIME_ FORMAT	Текущий формат времени; может иметь следующие значения - **MCI_FORMAT_MILLISECONDS**, **MCI_FORMAT_MSF**, **MCI_FORMAT_TMSF**
MCI_STATUS_START	Начальная позиция
MCI_STATUS_TRACK	В поле **dwTrack** записывается либо начальная позиция заданной дорожки (если дополнительно используется **MCI_STATUS_POSITION**), либо размер дорожки (если дополнительно используется **MCI_STSTUS_LENGTH**)
MCI_STATUS_MEDIA_ PRESENT	Если компакт-диск вставлен в устройство, возвращается **TRUE**

Команда **MCI_SET** предназначена для установки режима работы устройства, совместно с ней используется блок параметров в формате структуры **MCI_SET_PARMS**

```
typedef struct tagMCI_SET_PARMS
{
   DWORD dwCallback;
   DWORD dwTimeFormat;
   DWORD dwAudio;
} MCI_SET_PARMS;

typedef MCI_SET_PARMS FAR *LPMCI_SET_PARMS;
```

Поле **dwTimeFormat** определяет формат времени для устройства, поле **dwAudio** задает выходной канал. Совместно с командой **MCI_SET** используются следующие флаги

Идентификатор флага	*Описание*
MCI_NOTIFY	При установке этого флага после завершения команды оконной функции, адрес которой передан через поле **dwCallback**, будет послано сообщение **MM_MCINOTIFY**
MCI_WAIT	Функция **mciSendCommand** вернет управление только после завершения процесса
MCI_SET_AUDIO	Включение или выключение каналов; используется вместе с флагами **MCI_SET_ON** и **MCI_SET_OFF**. Поле **dwAudio** содержит номера канала, дополнительно можно указать следующие константы - **MCI_SET_AUDIO_ALL** (все каналы), **MCI_SET_AUDIO_LEFT** (левый канал), **MCI_SET_AUDIO_RIGHT** (правый канал)
MCI_SET_DOOR_CLOSED	Команда защелкивания компакт-диска устройством
MCI_SET_DOOR_OPEN	Освобождение носителя данных
MCI_SET_VIDEO	Включение или выключение видеосигнала, используется вместе с флагами **MCI_SET_ON** и **MCI_SET_OFF**
MCI_SET_TIME_FORMAT	Устанавливает формат времени, используется совместно со следующими константами - **MCI_FORMAT_MSF** (минуты, секунды, фреймы), **MCI_FORMAT_MILLSECONDS** (миллисекунды), **MCI_FORMAT_TMSF** (треки, минуты, секунды, фреймы)
MCI_SET_ON	Включение заданного канала
MCI_SET_OFF	Выключение заданного канала

При использовании формата времени **MCI_FORMAT_MSF** старший байт старшего слова поля **dwTimeFormat** не используется, а младший со-

держит номер фрейма. Старший байт младшего слова содержит секунды, младший - минуты. Формат времени **MCI_FORMAT_TMSF** аналогичен вышеприведенному за исключением того, что старший байт старшего слова содержит номер дорожки.

В работе [6] приведен полный С-код приложения **MCICDPL**, служащего для управления устройством чтения CD-ROM при помощи управляющих сообщений MCI.

3.8. **Проигрывание MIDI-файлов**

Файлы в стандарте MIDI (*Musical Instrument Digital Interface*, был разработан в 1982 г.) имеют расширения **MID** и содержат заголовок и информацию для музыкального синтезатора; используется также стандарт RIFF (содержащие сообщения MIDI и соответствующие стандарту RIFF файлы имеют обычно расширения RMI).

MIDI-файлы создаются с помощью музыкальной клавиатуры и соответствующего программного обеспечения (и, разумеется, некоторых музыкальных способностей).

При воспроизведении MIDI-файлов могут возникнуть трудности, связанные с тем, что не все синтезаторы имеют одинаковое распределение каналов и инструментов; поэтому рекомендуется воспроизводить в среде Windows только авторизованные для этой ОС файлы MIDI-формата (при отсутствии авторизации Windows выдает предупреждающее сообщение).

3.8.1. **Интерфейс управляющих строк MCI**

Для работы с входящим в комплект звукового адаптера музыкальным синтезатором используется драйвер **mciseq.drv** (название суть производная от слова *sequencer* - ‘устройство задания последовательности’ - именно так в терминологии мультимедиа называется предназначенное для работы с файлами в стандарте MIDI устройство).

При работе с MID-файлами на уровне управляющих строк MCI допустимо пользоваться практически всеми командами, рассмотренными ранее в подразделе 2.7.1; не поддерживаются команды **resume**, **record** и **save**. Например, следующая последовательность команд выполнит проигрывание файла **canion.mid**, входящего в состав дистрибутива ОС Windows

```
open  c:\windows\canyon.mid  alias  music  wait
play  music  wait
close  music
```

Драйвер **mciseq.drv** не поддерживает следующие параметры команды **set** - **audio all off**, **audio all on**, **audio left on**, **audio left off**, **audio right on**,

audio right off. Дополнительно можно использовать устанавливающий формат времени в единицах ‘одна шестнадцатая ноты’ параметр **time format song pointer**, позволяющий задать темп исполнения мелодии параметр **tempo** и некоторые другие.

3.8.2. **Интерфейс управляющих сообщений MCI**

Использование интерфейса управляющих сообщений для проигрывания mid-файлов аналогично использованию этого интерфейса для проигрывания WAV-файлов, см. подраздел 3.7.2; ниже кратко приведены коды управляющих сообщений и самые нужные параметры, являющиеся специфичными для драйвера **mciseq.drv**

Коды управляющих сообщений MCI для работы с MIDI-файлами	*Описание*
MCI_OPEN	Устройство *sequencer* открывается с использованием вышеописанной структуры **MCI_OPEN_PARMS**. Поле **lpstrDeviceType** этой структуры должно содержать указатель на строку имени устройства или константный идентификатор устройства. Для устройства *sequencer* можно указать имя "sequencer" или константу MCI_DEVTYPE_SEQUENCER
MCI_CLOSE	Команда закрытия устройства, выдается после завершения работы с устройством
MCI_PLAY	Проигрывает выбранный файл, функционирование не отличается от случая проигрывания WAV-файлов
MCI_PAUSE	Приостанавливает выполнение операции проигрывания
MCI_RESUME	Не поддерживается драйвером **mciseq.drv**, вместо нее для запуска проигрывания с текущей позиции можно использовать команду **MCI_PLAY** без указания позиции
MCI_STOP	Останавливает выполнение операции проигрывания
MCI_SEEK	Выполняет позиционирование в пределах mid-файла
MCI_BREAK	Устанавливает виртуальный код клавиши прерывания выполнения операции
MCI_GETDEVCAPS	Позволяет определить возможности устройства
MCI_INFO	Служит для получения информации об устройстве в виде текстовой строки
MCI_SYSINFO	Позволяет получить системную информацию в виде текстовой строки об устройстве
MCI_STATUS	Используется для определения текущего состояния устройства
MCI_SET	Предназначена для установки режима работы устройства
MCI_COPY	Позволяет копировать данных в универсальный буфер обмена (Clipboard), совместно с ней используется блок параметров в формате структуры **MCI_GENERIC_PARMS** (флаги

	MCI_NOTIFY и **MCI_WAIT**)
MCI_PASTE	Вставляет данные из Clipboard в текущий буфер устройства, использует блок параметров в формате структуры **MCI_GENERIC_PARMS** (флаги **MCI_NOTIFY** и **MCI_WAIT**)
MCI_CUT	Удаляет данные из текущего буфера устройства и копирует их в универсальный буфер обмена Clipboard, использует блок параметров в формате структуры **MCI_GENERIC_PARMS** (флаги **MCI_NOTIFY** и **MCI_WAIT**)
MCI_DELETE	Удаляет данные из текущего буфера устройства без копирования их в Clipboard
MCI_LOAD	Используется для загрузки файла

В работе [6] приведен полный С-код приложения **MIDIPL**, демонстрирующего способы использования функций MCI для проигрывания MIDI-файлов.

3.9. **Штатное программное обеспечение записи, редактирования и воспроизведения звука**

Программное обеспечение для работы со звуком можно разделить (по функциональному назначению) на следующие группы (зачастую функциональность групп перекрывается):

- Плееры (*players*) - ПО для воспроизведения звука.
- Грабберы (*grabbers*) - ПО для ‘захвата’ существующей звуковой информации и перекодировки в иной формат.
- Устройства для создания и редактирования звука - ПО для создания звука (с возможностями гармонизации, аранжировки, стилистической обработки, наложения голоса, добавления спецэффектов - реверберация, псевдостереофоничность и др.).

Весьма распространенным среди пользователей Windows является плеер **Winamp** (фирма *NullSoft, Inc.,* www.winamp.com), являющейся являющийся устройством с поддержкой неограниченного числа аудиоформатов (для их поддержки используются декодеры, поставляемые производителями). Например, в версиях 2.20÷2.22 роль встроенного MP3-декодера играет ‘родной’, разработанный вышеупомянутым **Fraunhofer IIS** декодер (однако начиная с версии 2.23 возвращен ‘старый’, менее качественный декодер); удобным является наличие встроенного броузера для д**о**ступа к сети InterNet (MS представляет возможность встр**а**ивания компонентов своего броузера в любое пользовательское приложение путем использования органа управления **Microsoft Web Browser Control** из сред**ы** программиро-

вания **MS Visual C++** или вызова функций из DLL-библиотек, где определены соответствующие объекты **ActiveX**; дополнительные возможности предоставляют размещенные в библиотеке **wininet.dll** функции интерфейса **WinInet**, см. www.microsoft.com/win32dev).

Часто используется разработанный MS еще в 1992 г. **Windows Media Player**, также поддерживающий (на сегодняшний момент) практически все известные аудиоформаты. Серьезным преимуществом вышеописанного ПО является удобный пользовательский интерфейс.

Специалисты отдают предпочтение плееру **NAD** (nad.inept.org, известны версии до 0.94, после чего проект **NAD** был выкуплен фирмой *Dimension Music*), для повышения эффективности работы использующий интересный алгоритм предсказания. На основе семейства функций **NAD** создан набор библиотек AE (*Audio Enlightenment*, позднее AE переименован в **STARDUST** и задействован в плеере **Sonique**), использованный при разработке плеера **NADDY** (ae.dmusic.com).

Известны плееры **WPlay**, **I3dec**, **K-Jofol** (поддерживающий VQF), **Apollo**, **Soritong**, **C-4** (полный список можно найти на www.mp3.com, полезны сайты www.mpeg.org, www.mp3tech.com и страница www.uic.nnov.ru/~fmm).

Современные InterNet-технологии потребовали добавления функции проигрывания аудиофайлов из Сети (с возможностью их последующего сохранения во внешней памяти клиентской машины). Наиболее известным ПО подобного назначения является **RealPlayer** (фирма *RealNetworks*, www.real.com), поддерживающий более 20 мультимедиа-форматов (в т.ч. SMIL, Liquid Audio, RichFX и MP3) и позволяющий с помощью модема или выделенной линии получить д**о**ступ к более чем 2500 Сетевых радиостанций. Новая версия программы **iQfx Basic 2.0** (www.real.com/accessories/iqfx/index.htm) добавляет музыке объемные стереоэффекты, улучшает звучание бас**о**в и др.

Многие плееры являются всего лишь front-end - программами (обладающие удобным и красивым интерфейсом оболочками над программами-декодерами - например, над классическим декодером разработки **Fraunhofer IIS**). Большинство декодеров могут выполнять функции плеерами (самостоятельно, посредством управления с помощью командной строки или в комплекте с оболочкой), однако не каждый плеер может перенаправлять свой вывод в файл (что является необходимой функцией декодера).

Программа для ‘захвата’ звука (например, с WAV-файлов CD и выдачи его в заданный файл в просторечии называется *граббером* (от англ. grabbing - считывать звук с CD). Граббер считывает дорожки (*tracks*) с CD-Audio (файловая система CD-Audio согласно ‘Красной книге’ не дает возможности прямого копирования информации на другие носители) *по от-*

дельности (что представляет проблемы для некоторых CD-пр*и*водов) и записывает их в формате несжатого WAV-файла. Примером граббера является программа **CD Worx** (www.tfh-berlin.de/~s570959/dworx.html), отличающаяся хранением всего считанного файла в оперативной памяти компьютера.

В дальнейшем полученный файл можно сжимать по технологии MP3, например, Windows-ориентированными компрессорами **WinDAC** (members.aol.com/schelmik/dac.html) или **MPEG Layer 3 Audio Producer Professional** (www.iis.fhg.de). В результате сжатия получатся файлы MP3 или WAV значительно меньшего объема, однако при их записи на CD это будут уже не звуковые диски, а обычные CD-ROM (прослушивать их можно только на компьютере с CD-ROM и звуковой картой). Некоторые программы записи на (одно- или многократно записываемые) CD имеют встроенные MP3-компрессоры.

Особый интерес представляет ‘грабительство’ звука из InterNet. Многие музыкальные сайты работают с пот**о**ковым аудио, не позволяя выгруж**а**ть аудиофайлы (что, по мнению разработчиков, позволяет защищать аудиоинформацию от несанкционированного копирования). В этом случае выходом может быть перехват звука (например, с помощью **Total Recorder**) во время проигрывания на клиентской машине сетевых файлов с помощью **Real Player**, **Real Jukebox** или **Windows Media Player** с последующим сохранением несжатого WAV-файла (объем WAV-файла ограничен 4 Гбайт, что соответствует 40 мин при записи с качеством CD); полученный файл может быть в последующем конвертирован в MP3 (например, посредством **MusicMatch Jukebox**).

Windows штатно включает подсистему ACM (*Audio Compression Manager*), управляющую упак**о**вкой и распак**о**вкой звуковых файлов. Под управлением этой системы работают различные преобразователи (кодеки), обрабатывающие считываемый из файла (при распаковке) или записываемый в файл (при упаковке) звук. Т.к. система ACM стандартизирована, ее может использовать любая Windows-программа - например, **Sound Recorder** (‘Фонограф’), **Media Player** (‘Универсальный Проигрыватель’), звуковые редакторы и др. При использовании ACM распаковка и упаковка данных в любом формате происходит прозрачно для целевой программы, сама программа даже ‘не знает’ о (возможной) перекодировке и работает со стандартным форматом аудиоданных.

Контрольные вопросы

1. Чем является согласно современным представлением звук ?

2. Что такое аналоговый и цифровой метод представления звуковых колебаний ? Чем определяется максимальная частота звука при цифровой записи ?
3. В чем смысл блочной записи звука ?
4. Какие методы сжатия используются при работе со звуком и на каких принципах они основаны ?
5. Какие методы генерации звука используются в современных компьютерных технологиях ?
6. В чем заключается технология MCI работы с мультимедиа-данными ?
7. Каким образом осуществляется управление CD ROM с использованием MCI ?

4. ЗАПИСЬ И ВОСПРОИЗВЕДЕНИЕ МУЛЬТИМЕДИА-ИНФОРМАЦИИ

Штатным средством Windows для записи звука является MS **Sound Recorder**, позволяющий записать с микрофона и сохранить без компрессии звук в формате WAV (при этом должны быть корректно установлены низкоуровневые драйвер**а** звуковой платы). Подобные файлы имеют очень большие размеры и их приходится в дальнейшем компрессировать с использованием вышеуказанных методов.

Для вывода звука на внешние устройства возможно использование (также штатного для Windows) **Media Player** (версия 8.0 для Windows'XP); при этом поддерживаются все установленные в системе кодеки сжатия аудиоинформации.

Указанное ПО является фактически надстройками над системными средствами поддержки записи и воспроизведения звука (драйверами физических аудиоустройств и кодеками) и не обладают развитыми средствами редактирования звука.

Этими возможностями наделены специализированные системы записи, редактирования и конвертации (с применением известных методов сжатия) между форматами аудиоданных. Ниже будут упомянуты лишь некоторые из множества систем цифровой обработки звука; часть из них использует собственноразработанные кодеки, иногда являющимися более эффективными стандартных.

Для потокового приема и воспроизведения аудиофайлов из InterNet наиболее часто применяют **RealPlayer** (фирма *RealNetworks*, www.real.com), см. подраздел 3.9.

Система **Cakewalk Pro Audio** (www.cakewalk.com) является программой профессиональной обработки звука, реализует огромное количество возможностей: многопоточность, микширование цифрового звука и

MIDI, возможность обработки звука различными фильтрами, создание различных имитаций и эффектов.

Известная фирма *Ulead Systems, Inc*. (www.ulead.com) предлагает пакет **Ulead MediaStudio**, в состав которого входит достаточно мощный **Ulead Audio Editor**.

Удобен (созданный на основе MP3 Fraunhofer-кодека, см. подраздел 3.2) редактор звуковых файлов **Cool Edit** (support.syntrillium.com) со встроенной поддержкой 24-битного цифрового аудио (частота дискретизации до 96 kГц) и большим количеством звуковых эффектов, имеются возможности для записи с ан**а**логовых и цифровых источников и редактирования с помощью набора фильтров.

Также распространены пакеты **Gold Wave** (www.goldwave.com), **Music Guru** (www.), **Steinberg WaveLab** (www.steinberg.net) и многие другие – обычно пользователь самостоятельно выбирает нужный пакет на основе анализа возможностей, интерфейса и доступности.

4.1. Основные стандарты записи и воспроизведения аудиовидеоинформации

Исторически первыми форматами аудиовидео явились разработанные для нужд телевидения стандарты **PAL** (*Phase Alteration Line*, принят в Германии, Великобритании, Южной Америке. Австралии и многих странах Западной Европы и Азии, предполагает размер изображения из 625 строк при 50 полукадрах/сек), **NTSC** (*National Television Systems Committee*, принят в США, Японии, Канаде, Мексике, размер изображения 535 строк при 60 полукадрах/сек) и **SECAM** (S*equence de Coulers Avec Memoire*, распространен во Франции, размер изображения 818 строк при 50 полукадрах/сек). Общим во всех известных системах является метод передачи последовательных статических изображений, разным является способ передачи информации о цвете.

Претензии высокого уровня к качеству видео пр**и**звана удовлетворить система **HDTV** (*High-Defenition Television*, разрешение 1150 строк по 2035 пикселов в строке, 50 полукадров/сек, формат кадра 16:9).

Даже стандарт HDTV (1150×2035=2’340’250 пиксел) недостаточен для современных цифровых фото- и видеокамер, обычным становится применение т.н. 4-х мегапиксельных камер.

Для записи видеоинформации на магнитную ленту (видеомагнитофоны) используются форматы **VHS** (*Video Home System*, не более 240 строк при ан**а**логовом способе записи), **S-VHS** (*Super VHS*, усовершенствованная VHS, разрешение до 400 строк; вариант **S-VHS C** предполагает более компактный вариант оборудования), **Video 8** (цифровая запись *методом импульсно-кодовой модуляции*, 260 строк) и **Hi-8** (усовершенствование

Video 8, до 400 строк). Современными цифровыми системами являются **DV**, **miniDV** и **Digital 8** (не менее 500 строк в кадре).

Распространенной цифровой системой профессионального уровня является **Betacam**, предложенная и активно поддерживаемая с 1982 г. фирмой *Sony* (www.sony.com). Betacam является не просто форматом видеозаписи (625 строк при 50 полукадрах/сек или 525/30), но цифровой системой, объединяющей различные носители и технологии (Betacam и Betacam SP для ан**а**логовой и Betacam XP для гибридной – ан**а**логовой или цифров**о**й записи):

- магнитную ленту (шириной 12,65 мм и толщиной $14{,}5\times10^{-3}$ мм, время записи 62/194 мин) для съемки и *линейного монтажа* в полевых условиях
- магнитный диск для *нелинейного* монтажа в студийных условиях
- цифровые интерфейсы для подключения к сет**я**м передачи данных и линиям телекоммуникаций, стандарты **SDI** (Serial Digital Interface) и **SDTI** (Serial Digital Transport Interface)
- компрессию (степени 10:1 по методу **MPEG-2**, см. подраздел 4.2) с целью повышения эффективности обмена данными.

Линейный монтаж предусматривает простейшие манипуляции с исходным видеоматериалом; нелинейный монтаж позволяет произвольным образом компоновать отдельные фрагменты видеоряда, снабжать их титрами и видеоэффектами, озвучивать и переозвучивать фрагменты и др. В случае нелинейного монтажа новый вариант видеофильма создается (*компилируется*) на основе существующего ролика и *файла проекта*, содержащего информацию о всех необходимых манипуляциях с видеоданными (к сожалению, стандартизация форматов *файлов проектов* в настоящее время только зарождается).

Для записи аудиовидео CD используются форматы **VCD** (*Video Compact Disc*, качество видео сравн**и**мо с VHS, используется сжатие по методу MPEG-1), **SVCD** (*Super VCD*, длительность записи 35-60 мин на стандартном CD) и **DVD** (*Digital Compact Disc*, сжатие по MPEG-2); подробнее см. следующий подраздел. В нижерасположенной таблице приведены распространенные форматы записи аудиовидео-информации на CD :

Тип CD	***Стандарт NTSC***	***Стандарт PAL***	***Способ сжатия и скорость передачи данных для видео***	***Способ сжатия и скорость передачи данных для аудио***
DVD	720×480	720×576	MPEG-2, переменная до 9,8 Мб/сек	MPEG-2 (AC-3), 768 кб/сек
Mini DVD	720×480	720×576	MPEG-2, переменная до 9,8 Мб/сек	MPEG-2 (AC-3), 768 кб/сек

SVCD	480×480	480×576	MPEG-2, переменная до 2,6 Мб/сек	MPEG-1, 384 кб/сек
VCD	352×240	352×288	MPEG-1, 1,15 Мб/сек	MPEG-1, 224 кб/сек

4.2. **Сжатие и распаковка видеоданных**

Согласно современным понятиям видео представляет собой последовательность статических изображений, сменяющих друг друга с частотой не менее 25÷30 раз в секунду, при этом поток данных для видео весьма велик. Например, для разобранного в подразделе 3.2 случая (изображение 640×480 пикселов, глубина цвета 16 бит) видеопоток при 25 кадрах/сек (FPS – Frame Rep Second) равен 50 Мбайт/сек ! Ясно, что в случае видео сжатие данных еще более важно, чем для статических изображений.

Концепция одновременной записи видео и звука была предложена еще MS в формате **AVI** (*Audio Video Interleave*), принципом является запись аудио- и видеосигналов в одном файле отдельно для каждого кадра (*frame*) – сначала звук первого видеокадра, затем изображение, затем звук следующего кадра и т.д. При этом информация внутри кадров или последовательности кадров может быть не сжата вообще (в прошлом применялось часто вследствие ограниченной мощности процессоров для сжатия в реальном времени) или скомпрессована любым известным методом (при сохранении расширения AVI). Одним из распространенных ПО конвертации несжатых AVI в MPEG-1 является **XingMPEG Encoder** (www.xingtech.com). Аппаратные методы компрессации видео в настоящее время применяются не столь часто (можно упомянуть, например, мощный аппаратно-программный комплекс **miroVideo DC1000** фирмы **Pinnacle Systems**, www.pinnaclesys.com, подробнее см. раздел 6). В настоящее время обычным является сохранение в файлах с расширением AVI сильно скомпрессированных видеофайлов DivX (см. ниже).

При сжатии файлов видео обычно применяются методы, подобные известным технологиям компрессии файлов изображений; часто учитывается особенность видео, заключающаяся в (относительно малом) изменении содержащейся в последовательных кадрах информации. Звук в аудиовидеофильмах записывается в виде отдельной дорожки (дорожек) и располагается вперемешку с кадрами видео, при этом используются специальные механизмы синхронизации видео и аудио. Мощность аудиопотока значительно (на порядки) ниже видеопотока, поэтому в случае ограничения пропускной способности канала передачи информации первым страдает видео; при всех системах кодирования принята идеология возможного пропуска видеокадров при обязательном сохранении аудиопотока (пропуск части аудиопоследовательности значительно заметнее для зрителя, нежели кадров видео вследствие значительной запоминающей способности глаза).

На сегодняшний день повсеместно распространенным является стандарт **MPEG** сжатия видеоинформации. Термин **MPEG** является аббревиатурой *Moving Picture Expert Group* – название международной экспертной группы, действующая в направлении разработки стандартов кодирования и сжатия видео- и аудио- данных. **MPEG** состоит из трех частей: *Audio*, *Video* и *System* (объединение и синхронизация двух других).

Алгоритм **MPEG**-компрессии принципиально ориентирован на обработку последовательностей кадров и использует высокую избыточность информации в изображениях, разделенных малым временн**ы**м интервалом. Между смежными изображениями обычно меняется только малая часть сцены – например, происходит плавное смещение небольшого объекта на фоне фиксированного заднего плана. В этом случае полную информацию о сцене достаточно сохранять только выборочно - для *опорных изображений*. Для остальных изображений достаточно передавать только разностную информацию: о положении объекта, направлении и величине смещения, о новых элементах фона (открывающихся за объектом по мере его движения). Эти разности можно формировать не только по сравнению с предыдущими изображениями, но и с последующими (поскольку именно в них по мере движения объекта открывается часть фона, ранее скрытая за объектом). Таким образом, в MPEG-кодировке принципиально формируются три типа кадров: **I** (*Intra*), выполняющие роль опорных и сохраняющие полный объем информации о структуре изображения; **P** (*Predictive*), несущие информацию об изменениях в структуре изображения по сравнению с предыдущим кадром (типов I или P); **B** (*Bi-directional*), сохраняющие только самую существенную часть информацию об отличиях от предыдущего и последующего изображений (только I или P). Последовательности I-, P- и B-кадров объединяются в фиксированные по длине и структуре группы кадров - **GOP** (*Group of Pictures*). Каждая GOP обязательно начинается с I-кадра и с определенной периодичностью содержит P-кадры. Ее структуру описывают как M/N, где M – общее число кадров в группе, а N – интервал между P-кадрами. Так, типичная для **Video CD** и **DVD** IPB группа 15/3 имеет следующий вид: IBBPBBPBBPBBPBB. Здесь каждый B-кадр восстанавливается по окружающим его P-кадрам (в начале и конце группы - по I и P), а в свою очередь каждый P-кадр – по предыдущему P- (или I) кадру. В то же время I-кадры самодостаточны и могут быть восстановлены независимо от других, однако являются опорными для всех P и тем более B- кадров группы. Соответственно у I наименьшая степень компрессии, у B - наибольшая. Установлено, что по размеру типичный P-кадр составляет 1/3 от I, а B – 1/8 часть. В результате MPEG последовательность IPPP (GOP 4/1) обеспечивает 2-кратное уменьшение требуемого потока данных (при том же качестве) по сравнению с последовательностью только из I кадров, а использование GOP 15/3 позволяет достичь 4-кратного сжатия.

Известны следующие стандарты (или *фазы*) общего стандарта: *MPEG-1*, *MPEG-2*, *MPEG-3*, *MPEG-4*, *MPEG-7*.

MPEG-1 предназначен для записи синхронизированных видео- и звукового сопровождения на CD ROM с учетом скорости считывания 1,5-3,5 Мбит/сек (при б**о**льшей пропускной способности канала желателен переход к MPEG-2).

При демонстрации такого фильма однотонные поверхности зачастую кажутся составленными из рассыпающихся квадратиков, явно заметны квадратики и в имеющих много действий сценах (результат принципа компрессии MPEG-1). Качество видео по стандарту MPEG-1 лишь немного лучше обычного VHS-видео, однако возможность практически неограниченного проигрывания CD ROM (качество медиа-ролика не снижается против ленты видеомагнитофона) поддерживают существование этого формата.

Форматы кодирования звука в MPEG разделяются на три части – *Layer I, Layer II* и *Layer III*. Прообразом для Layer I и Layer II явился стандарт *MUSICAM*, также иногда называют Layer II. Наибольшее сжатие достигается в Layer III, однако требует б**о**льших ресурсов на кодирование.

Принципы кодирования основаны на известном факте наличия в несжатом звуке множества избыточной информации. Принцип сжатия основан на эффектах маскировки некоторых звуков для человека (например, если идет сильный звук на частоте 1000 Гц, то более слабый звук на частоте 1100 Гц уже не будет слышен человеку, также будет ослаблена чувствительность человеческого уха на период в 100 *м*с после и 5 *м*с до возникновения сильного звука). Используемая в MPEG психоакустическая (*psycoacustic*) модель разбивает весь частотный спектр на части, в которых уровень звука считается одинаковым, а затем удаляет не воспринимаемые человеком звуки (согласно описанным выше эффектам).

Синхронизация и объединение звука и видео, осуществляется с помощью системных потоков (**system stream**, рис. 4.1), который включает в себя:

- Системный слой, содержащий временн**у**ю и иную информацию с целью разделения и синхронизации видео и аудио
- Компрессионный слой, содержащий видео и аудио потоки

Видео поток (рис. 4.2) содержит заголовок, несколько групп изображений (заголовок и несколько изображений необходимы для того, что бы обеспечить произвольный д**о**ступ к изображениям в группе в независимости от их порядка). Звуковой поток состоит из пакетов, каждый из которых включает заголовок и нескольких звуковых кадров (*audio-frame*).

Для синхронизации аудио и видео потоков в системный поток встраивается таймер, работающий с частотой 90 kГц (SCR, *System Clock Reference*), т.е. метка, по которой происходит увеличения временного счетчика в декодере) и метка начала воспроизведения (PDS, *Presentation Data Stamp*) вставляются в изображений или в звуковой кадр, чтобы ‘объяснитьэ декодеру, когда их воспроизводить. Размер PDS составляет 33 бита, что обеспечивает возможность представления любого временн**о**го цикла длиной до 24 часов).

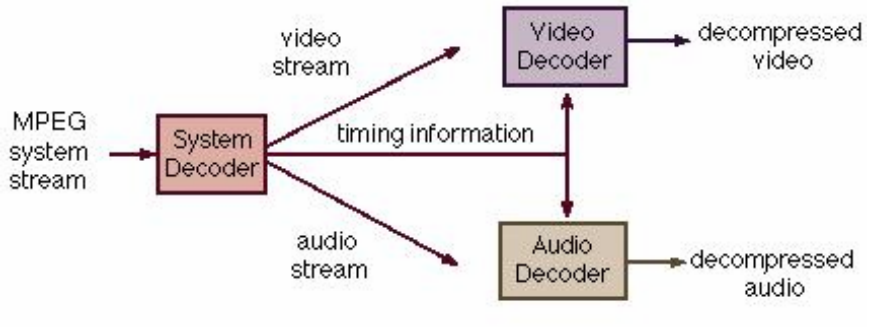

Рис. 4.1. Принцип синхронизации и объединения звука и видео

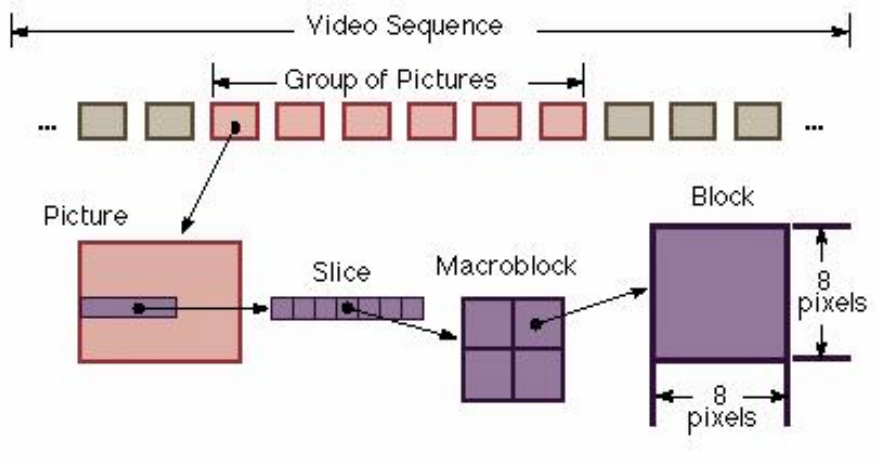

Рис. 4.2. Схема видеопотока при MPEG-компрессии

MPEG-2 предназначен для обработки видеоизображения, соизмеримого по качеству с телевизионным при пропускн**о**й способности системы передачи данных в пределах 3÷15 Мбит/сек (в профессиональной аппаратуре используют потоки скоростью до 50 Мбит/сек). На базирующиеся на стандарт MPEG-2 переходят многие телеканалы, сжатый в соответствии с этим стандартом сигнал транслируется через телевизионные спутники, используется для архивации больших объемов видеоматериала. Метод столь эффективен, что позволяет ‘убирать’ до 97% цифровых данных, являющихся избыточными вследствие дублирования.

По сравнению с MPEG-1 внесены изменение в аудиоканал:

1. Реализованы новые виды частот 16 kГц, 22,05 kГц, 24 kГц
2. Добавлена поддержка многоканальности (возможно иметь 5 полноценных каналов - *left, center, right, left surround, right surround* + 1 низкочастотный (*subwoofer*)
3. Появился стандарт **AAC** (*Advanced Audio Coding, прогрессивное кодирование звука*), обеспечивающий очень высокое качество звука со скоростью 64 Кбит/сек на канал; возможно использовать 48 основных каналов, 16 низкочастотных каналов для звуковых эффектов, 16 многоязыковых каналов и 16 каналов данных. Может быть использовано до 16 программ с использованием любого количества элементов звуковых и иных данных

Системный уровень MPEG-2 обеспечивает два уровня объединения данных:

1. PES (*Packetized Elementary Stream*) - разбивает звук и видео на пакеты.
2. Второй уровень делится на:
   - MPEG-2 *Program Stream* (совместим с MPEG-1 System) - для локальной передачи в среде с низким уровнем ошибок
   - MPEG-2 *Transport Stream* (рис. 5.3) - внешнее вещание в среде с высоким уровнем ошибок; передает транспортные пакеты (длиной 188 или 188+16 бит) двух типов (сжатые данные - PES - и сигнальную таблицу PSI - *Program Specific Information*)

MPEG-3 предназначался для использования в системах телевидения высокой четкости HDTV (*High-Defenition Television*) при разрешении 19201080 при 30 кадр/сек со скоростью потока данных 20÷40 Мбит/сек, но позже фактически стал частью стандарта MPEG-2 и отдельно теперь не упоминается. Формат **MP3** (который иногда путают с MPEG-3), предназначен только для сжатия аудиоинформации и полное название MP3 звучит как *MPEG-Audio Layer-3*.

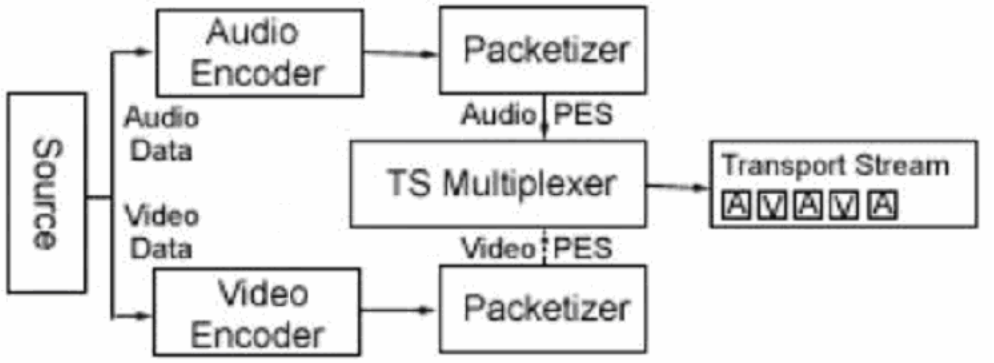

Рис. 4.3. Передача транспортных пакетов по стандарту MPEG-2 Transport Stream

MPEG-4 является стандартом для низкоскоростной передачи и задает принципы работы с цифровым представлением медиа-данных для трех областей: интерактивного мультимедиа (включая продукты, распространяемые на оптических дисках и через Сеть), графических приложений (синтетического контента) и цифрового телевидения. Существует MPEG-J, использующий элементы **JAVA**-технологии. Формат MPEG-4 дал толчок созданию популярного к**о**дека **DivX** (см. ниже).

При функционировании MPEG-4 позволяет:

1. Разделять картинку на различные элементы, называемые *media objects (медиа-объекты)*
2. Описывать структуру этих объектов и их взаимосвязи чтобы затем собрать их в видеозвуковую сцену (рис. 4.1)
3. Изменять сцену, обеспечивая этим высокий уровень интерактивности для конечного пользователя

Рис. 4.4 иллюстрирована видеозвуковая сцена, состоящая из объединенных иерархической структурой медиа-объектов; сцена состоит из:

1. Неподвижных изображений (например, фона сцены)
2. Видео-объектов (говорящий человек)
3. Аудио-объектов (связанный с этим человеком голос)
4. Связанного с данной сценой текста
5. Синтетических объектов – объектов, которых не было изначально в записываемой сцене, но которые туда добавляются при демонстрации конечному пользователю (например, синтезируется ‘говорящая голова’)
6. Связанного с ‘головой’ текста, на основе которого синтезируется голос

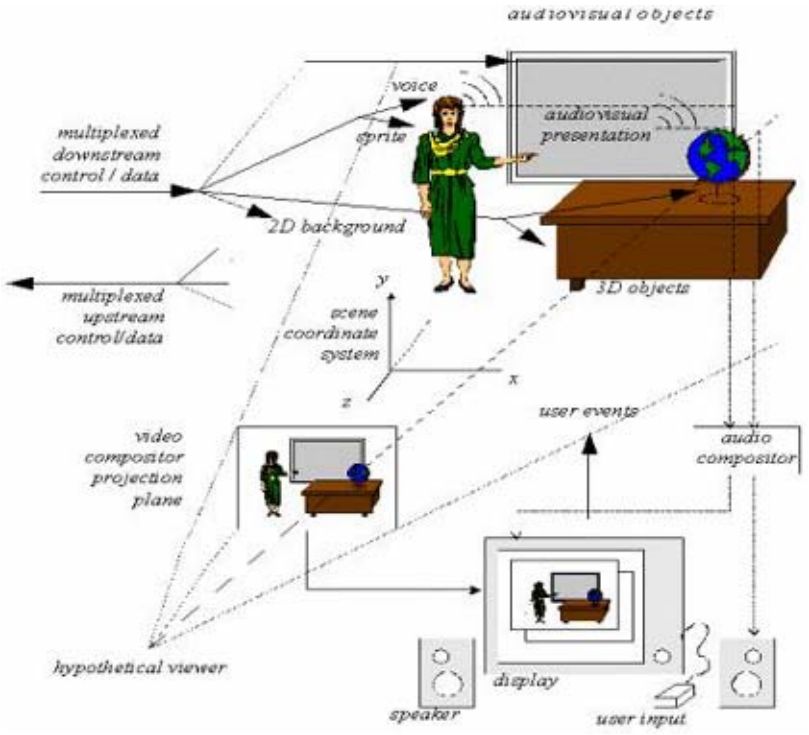

Рис. 4.4. Согласно MPEG-4 видеозвуковая сцена состоит из медиа-объектов, объединенных иерархической структурой

На рис. 4.5 схематически показана возможность интерактивного изменения видеозвуковой сцены.

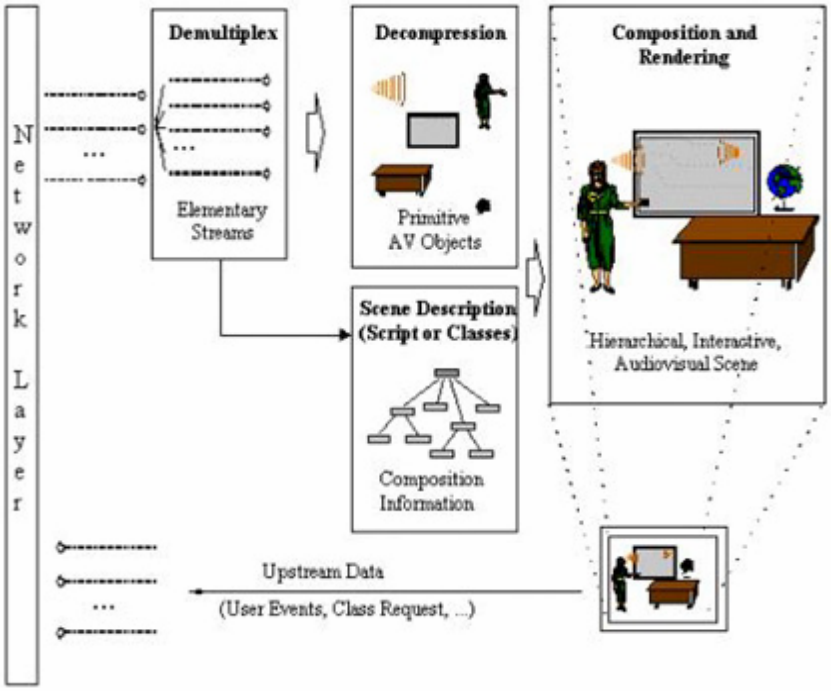

Рис. 4.5. Возможности изменения сцены конечным пользователем в интерактивном режиме

MPEG-7 не является продолжением ‘линейки MPEG’ и начал разрабатываться сравнительно недавно. MPEG-7 должен обеспечивать *стандарт для описания различных типов мультимедийной информации* (а не для ее кодирования), чтобы обеспечивать эффективный и быстрый ее поиск. MPEG-7 официально называют *Multimedia Content Description Interface* (*Интерфейс описания мультимедиа данных*). MPEG-7 определяет стандартный набор дискрипторов для различных типов мультимедиа информации, также он стандартизует способ определения своих дискрипторов и их взаимосвязи (*description schemes*). Для этой цели MPEG-7 вводит DDL (*Description Definition Language*, язык описания определений). Основная цель применения MPEG-7 - поиск мультимедиа информации (так же как сейчас возможно найти текст по определенному словосочетанию), например:

1. *Музыка*. Сыграв несколько нот на клавиатуре, можно получить список музыкальных произведений, которые содержат такую же последовательность
2. *Графика*. Нарисовав несколько линий на экране, получим набор рисунков, содержащих данный фрагмент
3. *Картины*. Определив объект (задав его форму и текстуру), получим список картин, содержащих такой объект
4. *Видео*. Задав объект и движение, получим набор видео или анимации
5. *Голос*. Задав фрагмент голоса певца, получим набор песен и видео роликов, где он является исполнителем

MHEG (*Multimedia & Hypermedia Expert Group*, экспертная группа по мультимедиа и гипермедиа) определяет стандарт для обмена мультимедийными объектами (видео, звук, текст и другие произвольные данные) между приложениями и передачи их разными способами (локальная сеть, сети телекоммуникаций и вещания) с использованием *MHEG object classes*. Он позволяет программным объектам включать в себя любую систему кодирования (например, MPEG), которая определена в базовом приложении. MHEG был принят **DAVIC** (*Digital Audio-Visual Council*, совет по цифровому видео и звуку). MHEG-объекты обрабатываются мультимедиа-приложениями с использованием *multimedia scripting languages*. Утверждается, что **MHEG** - будущий международный стандарт для интерактивного телевидения, так как он функционирует на любых платформах и его документация свободно распространяема.

Базовым объектом кодирования в стандарте **MPEG** является *кадр телевизионного изображения*. Поскольку в большинстве фрагментов фон изображения остается достаточно стабильным, а действие происходит только на переднем плане, сжатие начинается с создания *исходного кадра*. *Исходные* (*Intra*) кадры кодируются только с применением внутрикадрово-

го сжатия по алгоритмам, аналогичным описанным для формата *JPEG* (см. подраздел 2.3). Кадр разбивается на блоки 8×8 пикселов, над каждым блоком производится *дискретно-косинусное преобразование* (*ДКП*) с последующим квантованием полученных коэффициентов. Вследствии высокой пространственной корелляции яркости между соседними пикселами изображения *ДКП* приводит к концентрации сигнала в низкочастотной части спектра, который после квантования эффективно сжимается с использованием кодирования кодами переменной длины. Обработка *предсказуемых* (*Predicted*) кадров производится с использованием предсказания вперед по предшествующим исходным или предсказуемым кадрам. Кадр разбивается на макроблоки 16×16 пикселов, каждому макроблоку ставится в соответствие наиболее похожий участок изображения из опорного кадра, сдвинутый на *вектор перемещения*. Эта процедура называется *анализом и компенсацией движения*. Допуст**и**мая степень сжатия для предсказуемых кадров превышает возможную для исходных в 3 раза. В зависимости от характера видеоизображения, кадры *двунаправленной интерполяции* (*Bi-directional Interpolated*) кодируются одним из четырех способов: предсказание вперед, обратное предсказание с компенсацией движения (используется в том случае, когда в кодируемом кадре появляются новые объекты изображения), двунаправленное предсказание с компенсацией движения, внутрикадровое предсказание (применяется при резкой смене сюжета или при высокой скорости перемещения злементов изображения). С двунаправленными кадрами связано наиболее значительное сжатие видеоданных, но поскольку высокая степень сжатия снижает точность восстановления исходного изображения, *двунаправленные кадры не используются в качестве опорных*. В случае точной передачи коэффициентов ДКП восстановленное изображение полностью совпадало бы с исходным. Однако связанные с квантованием ошибки восстановления коэффициентов ДКП приводят к искажениям изображения. Чем грубее производится квантование, тем м**е**ньший объем занимают коэффициенты и тем сильнее сжатие сигнала, но и тем больше визуальных искажений.

Наиболее эффективным в настоящее время можно считать **DivX** (один из к**о**деков с таким названием разработан MS, технология сжатия является несколько измененным методом MPEG-4). Благодаря DivX стало возможным размещать видеоинформацию вместе со звуком при высоких разрешениях и с высоким качеством и при этом добиться приемлемых размеров файлов. Для просмотра таких фильмов достаточно установить в систему DivX к**о**дек, в**ы**грузить который можно с адреса www.3dnews.ru/download/dvd/divx-codec. В настоящее время действуют версии к**о**дека до 5.2; к**о**деки интенсивно используют процессорные технологии **MMX/SSE** (см. раздел 5). Процесс перекодировки в DivX весьма ресурсо**е**мок и (даже на мощных ПЭВМ) может длиться несколько часов

(именно поэтому DivX-кодирование в реальном времени в высшей степени затруднительно).

Реально качество DivX-фильмов хуже DVD, однако оно зачастую выше того, что встречается на видеокассетах. **Windows Media Player** легко детектирует присутствие DivX-кодека в системе и успешно применяет его (расширения файлов **AVI**, **DIVX**, **MP4**), из специализированных плееров можно рекомендовать **Playa** и **DivX Player** (оба производства фирмы DivXNetworks, Inc, www.divx.com).

Технологии сжатия изображений не ограничиваются вышеперечисленными; напр., фирма **AT&T** (www.att.com) продвигает метод эффективного сжатия **Wavelet transform**. Ряд фирм активно ведет разработку алгоритмов сжатия видеоинформации, стремясь достичь коэффициента сжатия порядка 200:1 и выше. В основе наиболее эффективных алгоритмов лежат различные адаптивные варианты: **DOT** *(Discrete Cosine Transform, дискретное косинус-преобразование),* **DPCM** *(Differential Pulse Code Modulation, разностная импульсно-кодовая модуляция),* а также фрактальные методы. Алгоритмы реализуются аппаратно — в виде специальных микросхем или записанной в ПЗУ программы, либо чисто программным путем.

4.3. **Программное обеспечение создания и воспроизведения видеофильмов**

Процесс создания видеофильмов фактически повторяет основные этапы технологии получения обычных фильмов – реальная съемка (или компьютерная генерация) первичных кадров, их монтаж (обрезка и вставка кадров в нужное место), генерация титров и спецэффектов, редактирование звука, сборка готового фильма и конвертирование его в нужный цифровой формат, запись в аналоговом или цифровом виде на внешние носители (магнитную ленту, CD или DVD). Все эти процедуры в высшей степени удобно выполняются в интерактивном режиме пользователя с ПЭВМ, причем вероятность необратимой порчи информации минимальна.

Сказанное относится к производству снятых ‘на натуре’ видеофильмов. При создании фильмов, содержащих элементы искусственно созданных изображений, применяется *технология анимации* (генерация последовательности кадров методом моделирования во времени определенных процессов). Анимация может быть как простейшей (реализуется посредством покадровой смены части изображений - *спрайтов*), так и весьма сложной - с использованием систем моделирования *виртуальной реальности* (см. раздел 6 данной работы) и *интерактивности*).

4.3.1. **Пакет VIDEO FOR WINDOWS и его основные возможности**

Появившийся в начале 90-х г.г. пакет Video For Windows (VFW) явился первым интегрированным пакетом поддержки обработки медиаинформации и предназначался для MS Windows'3.1; уже в Windows'95 отдельные его компоненты были переработаны и интегрированы в ОС. Тем не менее нижеприводимая информация важна с точки зрения истории развития мультимедиа.

В состав пакета входило 5 самостоятельных модулей для работы с видео и аудио:

1. **VidEdit -** ядро пакета VFW, позволяет загружать, сохранять, воспроизводить и обрабатывать (вырез**а**ть и синхронизировать части видеоклипа, вносить отдельные кадры, устанавливать частоту кадров при демонстрации и метод сжатия) видеоданные. Поддерживались форматы записи видеофайлов на диск **MS Video 1** (без сжатия), **MS RLE** (8-ми битовая глубина цвета и ограничение длины видеофайла) и **Intel Indeo** (наибольшая степень сжатие видеопоследовательности при глубине цвета 24 бита).
2. **VidCap** – программа для записи (захвата - *capture*) последовательности видеокадров от источника видеосигнала (видеокамеры, видеомагнитофона, телевизора) через видеоадаптер (специальную плату).
3. **BitEdit** – модуль обработки отдельных видеокадров, имеется возможность попиксельного изменения цвет**о**в, копирования фрагментов видеоизображения, сохранения изображения в BMP-формате в файле и внесение BMP-изображения в видеофайл.
4. **PalEdit** – программа изменения цветовой палитры видеофайла (сохранение, изменение и загрузка цветовых палитр).
5. **WaveEdit** – редактор звуковой информации, позволяет обрабатывать аудиоинформацию (аудиотрак) независимо от видеотрака. Возможна запись и обработка (вырезание, изменение громкости) последовательности звуков, поддерживаемый формат – WAV.

Демонстрация медиафайлов производилась с помощью **Media Player** (или **VidEdit**), упорядочивание медиафайлов (создание коллекций) осуществлялось программой **Media Browser**.

Т.о., несмотря на примитивную (с сегодняшней точки зрения) организацию модулей (отсутствие интегрированности), пакет VFW представлял практически идентичные современным системам возможности записи, редактирования и демонстрации медиаинформации.

4.3.2. **Современные пакеты работы с мультимедиа**

Ниже описываются проверенные практикой пакеты для создания, редактирования и воспроизведения мультимедиа для ПЭВМ класса IBM PC; системы высокопрофессиональной графики обычно эксплуатируются на аппаратной платформе **Silicon Graphics** (www.sgi.com).

Как было сказано, основным модулем воспроизведения медиаинформации в ОС Windows является **Media Player**; для Windows'XP известна версия 8.0, поддерживающий практически все известные форматы аудио- и видеоданных и впит**а**вший много полезного (в архитектуре построения) от известного аудиоплеера **WinAmp**, см. подраздел 3.9. Создание любительских видеофильмов поддерживает штатный (начиная с Windows'2000) пакет **Movie Maker** (к сожалению, указанное ПО не позволяет сохранять смонтированные ролики на видеомагнитофон – они сохраняются исключительно в разработанном MS специфичном формате **ASF**).

Для просмотра DVD часто используется **PowerDVD** (фирма CyberLink, www.gocyberlink.com, www.cli.co.jp, поддерживаются форматы MPG, VOB, ASF, M1V, M2V, AVI, WMV, DAT, VRO, WAV, MID, WM, WMA, MP2, MP3, RMI), удобен также пакет **WinDVD** (фирма InterVideo, www.intervideo.com).

Распространенным пакетом создания и редактирования аудиовидеофайлов является **Adobe Premiere** (www.adobe.com), позволяющий компановать аудиовидеофайлы как из отдельных изображений, так и готовых аудио- и видеоклипов, поддерживаются несколько аудио- и видеодорожек; пакет обладает также развитыми возможностями конвертации медиаформатов; близкими возможностями обладает **Ulead MediaStudio** (*Ulead Systems, Inc.*, www.ulead.com). Известен удобный пакет **Cinema** (*Avid Technology*, www.avid.com).

Современные пакеты создания и редактирования аудиовидеофайлов интегрированы со средствами полного цикла подготовки и выпуска DVD (включая 'захват' информации с ан**а**логовых и цифровых видеокамер, разбиение исходной оцифрованной ленты на клипы, редактирование аудио и видеоклипов, подготовку титров, эффекты плавного перехода между кадрами, привычные для DVD средства навигации внутри видеофильма) включая физическую запись DVD. Из известных систем такого рода можно назвать **PowerDirector** (фирма *CyberLink*, www.gocyberlink.com, поддерживаются входные форматы изображений BMP, GIF, JPG, TIF и видео Windows AVI, DV AVI, MPEG-1, MPEG-2, RealVideo, QuickTime, WMV) и WinDVD Creator (фирма *InterVideo, Inc.,* www.intervideo.com, функционирует только на MB фирмы **ASUSTek Computer Int.,** www.asus.com). Наряду с созданием DVD (объемом 4,7÷17 Гбайт на диск) указанные системы (часто) позволяют записывать смонтированный видеофильм на видеомагнитофон (форматы PAL и NTSC) и создавать 'обычные' компакт-диски

объемом 640÷700 Мбайт длительностью 74÷80 мин (VCD, *Video Compact Disc*).

4.3.3. **Особенности мультимедиа на DVD**

Технология **DVD** на сегодняшний день является наиболее мощной в деле сохранения и демонстрации мультимедиа. Большие объемы записываемой на стандартного размера CD, развитая система поиска и доступа к хран**и**мой информации, поддержка многоязычности в озвучивании и титрах потребовали серьезного развития технологии записи и ПО для обслуживания DVD. При подготовке материалов данного подраздела использованы данные с адрес**о**в www.dvd.da.ru, dvdsoft.online.fr/main.html и www.3dnews.ru; на указанных адрес**а**х приведено большое количество практически полезной информации в области записи, копирования, снятия региональной защиты DVD и др.

Аббревиатура DVD первоначально расшифровывалась как *Digital Video Disk,* соответствующий стандарт разработан в 1996 г. консорциумом компаний *Hitachi* (www.hitachipc.com)*, JVC* (www.jvc.com)*, Matsushita* (www.mei.co.jp)*, Mitsubishi* (www.mitsubishi.com)*, Phillips* (www.philips.com)*, Pioneer* (www.pioneerelectronics.com)*, Sony* (www.sony.com)*, Toshiba* (www.toshiba.com) и др. На DVD можно записывать любую информацию, поэтому современная расшифровка звучит как *Digital Versatile Disk (цифровой универсальный диск)*. Основное отличие DVD от CD-дисков заключается в объеме записываемой информации, который может достигать 17 Гбайт. Достигается это уменьшением длины волны считывающего лазера с 780 *н*м (инфракрасный) до 635/650 *н*м (красный), что позволяет существенно увеличить плотность записи (ширина дорожки 0,74 против 1,6 *н*м, размер пита 0,4 против 0,83 *н*м), использованием т.н. двуслойных дисков (один слой полупрозрачный, а второй читается ‘сквозь’ первый) и записью информации на обе стороны диска.

Основные виды DVD дисков (известны также диски диаметром 80 см):

1. DVD-5, диаметр 12 см, объем 4,7 Гбайт данных или свыше 2 часов видео, один слой на одной стороне
2. DVD-9, 12 см, 8,5 Гбайт данных или около 4 часов видео, два слоя на одной стороне
3. DVD-10, 12 см, 9,4 Гбайт данных или около 4,5 часов видео, на обоих сторонах по одному слою
4. DVD-14, 12 см, 13,24 Гбайт данных или около 6,5 часов видео, два слоя на одной стороне, один слой на другой

5. DVD-18, 12 см, 17 Гбайт данных или более 8 часов видео, на обоих сторонах по два слоя (в настоящее время слишком д**о**роги в производстве, поэтому применяются ограниченно).

DVD-формат представляет собой цифровое видео, сжатое по алгоритму MPEG-2 и записанное на DVD-диск. Формат - 25 кадров в секунду с разрешением 720×576 точек при глубине цвета 24-бит (PAL) или 30 кадров 720×480×24-бит (NTSC); из-за разницы в разрешении и числе кадров в секунду фактический поток информации одинаков в обоих случаях и равен 10’368’000 пикселов в секунду. В несжатом виде это поток 30 Мбайт в секунду, а двухчасовой фильм будет занимать более 100 Гбайт. Это огромный объем, и обеспечение необходимого для воспроизведения такого видео непрерывного потока данных на скорости 30 Мбайт/сек затруднительно. Поэтому используется сжатие по стандарту MPEG-2; кроме уменьшения размеров файлов это позволяет снизить поток данных до приемлемых 3÷4 Мбайт/сек. Чем сложнее сцена, тем хуже она поддается сжатию, и тем выше поток данных. Формат MPEG-2 первоначально разрабатывался для использования с высококлассной студийной аппаратурой и существуют варианты с записями в разрешениях гораздо более высоких чем 720×576 пиксел, но оказалось, что этот формат идеально подходит для записи фильмов на DVD-диски.

Алгоритм сжатия MPEG-2 весьма эффективен - удаляется примерно 97% избыточной информации практически без ущерба для качества картинки, благодаря чему на DVD-диске можно разместить до 4-х часов высококачественного видео плюс до 8 вариантов звукового сопровождения, плюс до 32 вариантов субтитров на разных языках. У DVD существует масса других интересных возможностей, таких, как возможность задания возрастн**ы**х ограничений на просмотр, интерактивность, быстрый переход в нужное место фильма, возможность наблюдать сцену с различных точек (этот режим получил название *multi-angle view* и весьма уважаем изготовителями продукции порнографического содержания).

Кроме этого, MPEG-2 отлично подходит как для сжатия киноматериала, так и для сжатия видео. Для работы с видео в MPEG-2 существует возможность хранить кадр не целиком, а в виде двух полей (как он записывается на видеокамеры, и как он показывается в большинстве телевизоров), что делает его идеальным выбором для этих целей. Не случайно именно в MPEG-2 транслируются современные спутниковые каналы, и именно MPEG-2 передается по сетям цифрового кабельного телевидения.

Отношение длины к высоте (*aspect ratio*) видеокадров DVD стандартизирован и равен телевизионному 4:3 (хотя встречаются от 1,55:1 до 2,55:1). При демонстрации таких кадров на реальных видеомониторах (с отл**и**чным от 4:3 соотношением сторон) используется два основных способа. Первый

(*Letterbox anamorph*) реализуется путем закрытия части изображения сверху и снизу черными полосками (чем дальше текущее отношение сторон экрана от 4:3, тем шире полосы), второй способ (*PAN&SCAN*) состоит в постановке специальных меток по ширине экрана на наиболее важных (в художественном смысле) частях изображения, в дальнейшем ширина кадра урезается по ширине при условии попадания метки в центр экрана.

Звук на DVD-дисках записывается в различных форматах. Это и PCM (для записей, где требуется точность передачи звуковой картины, например - музыкальное видео), и *Dolby ProLogic*, и *Dolby Digital* (от 2.0 до 5.1 и *Dolby Digital EX*), и даже *Audio MPEG 2.0* (этот стандарт для записи звука изначально предназначался для Европы, но не получил широкого распространения). Эти три формата объединяет одно - они воспроизводят несколько независимых каналов пространственного компрессированного звука, создавая тем самым реалистичную картину происходящего.

На DVD-Video диске в каталоге VIDEO_TS (AUDIO_TS обычно используется в DVD-Audio дисках) содержатся 3 типа файлов : *.IFO, *.BUP и *.VOB. Назначение этих файлов следующее :

- IFO (*InFOrmation*) - содержат в себе навигационную информацию для DVD-проигрывателей
- BUP (*BackUP*) - резервные копии *.IFO
- VOB (*Video OBjects*) - файлы, содержащие мультиплексированные (‘собранные’ из отдельных составляющих) субтитры, аудио-видео потоки (возможно, позволяющие реализовать мультиугловой - *multiangle* - просмотр) и служебную информацию

Современные мощные процессоры ПЭВМ без перенапряжения осуществляют программное декодирование DVD. Аппаратные декодеры обычно применяются лишь в аппаратуре домашних кинотеатров или при работе ПЭВМ с несколькими мониторами (один из которых является телевизором и предназначен для просмотра DVD в одно время с работой пользователя на ПЭВМ). Записывающие информацию в формате DVD устройства в настоящее время доступны большинству рядовых пользователей ПЭВМ. Необходимые скорости чтения для DVD-проигрывателя значительно (почти на порядок) выше, нежели обычных CD-дисков (однократная скорость для DVD равна 1,32 Мбайт/сек вместо 150 Кбайт/сек для CD).

Разногласия в стандартах записи и перезаписи DVD вызвали проблемы с совместимостью приводов DVD, в таблице ниже приведены (формальные) возможности соответствующим образом маркированных приводов; по горизонтали приведены типы DVD-приводов (DVD+RW и DVD+R представляют собой одно и то же), по вертикали – типы DVD-дисков.

	DVD-ROM	***DVD-R(G)***	***DVD-R(A)***	***DVD-RW***	***DVD-RAM***	***DVD+RW***
DVD-ROM	Читает	Читает	Читает	Читает	Читает	Читает
DVD-R (General)	Должен читать	Читает, пишет	Читает, не пишет	Читает, пишет	Читает	Читает
DVD-R (Authoring)	Должен читать	Читает, не пишет	Читает, пишет	Читает, не пишет	Читает	Читает
DVD-RW	Обычно читает	Читает	Читает	Читает, пишет	Обычно читает	Обычно читает
DVD-RAM	Редко, но читает	Не читает	Не читает	Не читает	Читает, пишет	Не читает
DVD+RW	Обычно читает	Обычно читает	Обычно читает	Должен читать	Обычно читает	Читает, пишет
DVD+R	Должен читать	Должен читать	Должен читать	Должен читать	Должен читать	Должен читать и писать

Энтузиастами DVD составляются списки совмест**и**мых и не совместимых с определенными стандартами привод**о**в. Наиболее полный из таких списков можно найти на адресах www.vcdhelp.com/dvdplayers.php и www.3dnews.ru. С целью стандартизации консорциум DVD ввел логотип **DVD multi**; получивший такой логотип привод должен писать и читать DVD-R, DVD-RW и DVD-RAM. Фирма *Sony* (www.sony.com) выпустила устройство DRU-500A, позволяющее записывать и считывать информацию с DVD независимо от типа носителя (кроме DVD-RAM). В очередной версии Windows (*Longhorn*) предполагается встроить штатную поддержку **DVD+RW** в качестве нового стандарта систем хранения данных для персональных компьютеров и бытовой электроники. Уже в начале 2003 г. консорциум компаний - ведущих производителей DVD лицензирует технологию **Blu-ray Disc**, позволяющая записывать на одностороннем 12 см диске 27 Гбайт данных (13 часов телетрансляции, используется сине-фиолетовый лазер с длиной волны 405 *н*м, планируется увеличить объем записываемой информации до 50 Гбайт).

С целью предотвращения нелегального распространения DVD-видео дисков, ассоциация разработчиков DVD-дисков ввела в спецификацию DVD несколько методов защиты. Наиболее распространённой является региональная защита. Суть ее такова - разработчиками поделен мир на несколько регионов:

1. Канада и США
2. Япония, Европа, Южная Африка, Ближний Восток (включая Египет)
3. Юго-Восточная Азия, Восточная Азия (включая Гонконг)
4. Австралия, Новая Зеландия, Тихоокеанские Острова, Центральная Америка, Южная Америка, Карибские острова

5. Бывший СССР, Индийский полуостров, Африка (также Северная Корея, Монголия)
6. Китай
7. Зарезервированный
8. Экстерриториальная зона (самолеты, круизные лайнеры и пр.), практически не используется

Любое устройство для воспроизведения DVD-дисков (в том числе и компьютерные DVD-приводы), произведенное в настоящее время, обязано поддерживать региональную защиту, подобную же защиту должны поддерживать и все программные средства, предназначенные для проигрывания DVD-видео. Такие устройства и программы при каждом проигрывании DVD-диска сравнивают код региона, записанный на диске со своим внутренним кодом, и если он не совпадает, то 'отказываются' проигрывать диск.

Для снятия защиты на программном уровне, существует несколько способов. Наиболее простой заключается в использование ПО www.3dnews.ru/download/tools/dvdgenie (для использования ее необходимо знать код применяемой программы-плеера) или www.3dnews.ru/download/dvd/dvdregionkiller (драйвер перехв**а**та потока информации с DVD и подм**е**ны данных регионального кода).

Для работы любой из этих утилит обязательно наличие DVD привода со *снятой региональной защитой*. Код региона на современных дисках, как правило, может быть изменен не более 5 раз, после чего он записывается навечно и стандартными средствами изменить его уже нельзя. Существуют методы замены (перезаписи) прош**и**вок DVD-плеера.

Кроме региональной защиты в DVD-видеодисках может использоваться несколько методов *защиты дисков от нелегального копирования*. Метод **Macrovision protection** (фирма *Macrovision*, www.macrovision.com) предназначен для предотвращения нелегального копирования с ан**а**логового (телевизионного) выхода на видеомагнитофон. Суть этой защиты в том, что в видеосигнал добавляются помехи, которые не дают сделать нормальную запись на неспособные отфильтровывать помехи устройства записи (при этом просмотр на телевизоре возможен). Для предотвращения копирования содержимого DVD-видео диска на жесткий диск компьютера используется **CSS** (*Content Scrambling System).* При этом содержимое DVD-диска шифруется и для расшифровки требуется ключ, состоящий из двух частей; одна часть представляет собой один из 400 хранящихся на каждом CSS DVD-видеодиске кодов, вторая часть ключа зависит от конкретного DVD-привода. Процесс расшифровки осуществляется программой-декодером, которая проигрывает DVD-видео. Таким образом, при простой перезаписи на жесткий диск содержимого защищенного с помощью

системы CSS DVD-диска декодер не сможет получить вторую половину ключа и данные не будут расшифрованы. 16-летний норвежский программист *Jon Johansen* сумел проанализировать работу дешифровального блока и выпустил утилиту **DeCSS** (в Сети официально не опубликована), которая расшифровывает данные и позволяет записывать содержимое защищённых DVD-видеодисков на жесткий диск (что послужило поводом к серии судебных разбирательств, не закончившихся до сих пор). Известен ан**а**логовый метод ограничения количества пиратских копий **CGMS/A** (*Copy Generation Management System/Analog*) и цифров**о**й метод **CGMS/D**. Практические рекомендации по клонированию DVD-дисков можно найти на www.3dnews.ru.

4.3.4. **Работа с мультимедиа в средах быстрой разработки приложений**

Современное программирование немыслимо без широкого использования *технологии объектно-ориентированного программирования*; наиболее последовательно этот метод реализован в **RAD** (*Rapid Application Design*, ср**е**дах быстрой разработки приложений) фирмы **Borland Int**. (www.inprise.com, www.borland.com.ru) – интегрированных системах **Delphi** и **C++Builder** [5].

Системы фирмы Borland включают **VCL** (*Visual Component Library*, библиотеки визуальных компонентов), инкапсулирующие модули для выполнения большинства возможных приложений; пользователь может разрабатывать собственные компоненты, включать их в библиотеку и распространять совместно с системами Delphi и C++Builder. Благодаря инкапсуляции многих сложных функций ОС в *компонентах* (функционально завершенных блоках объектно-ориентированного кода, обладающих интерфейсной частью для соединения с ОС и себе подобными) создание приложения упрощается (и убыстряется !) в десятки раз, при этом остается дост**у**пной работа непосредственно с API-функциями конкретной ОС.

Для визуализации статических изображений обычно применяется расположенный на форме (аналог Windows-окна в режиме *Design Time* - проектирования приложений) компонент типа **Image**, функция выбора файла и его визуализации приведена ниже (**Pascal**-вариант для Delphi)

```
procedure TForm1.BitBtn1Click(Sender: TObject);
{  выбирает и загружает файл изображения для просмотра  }
begin
  if OpenDialog1.Execute then { если файл выбран… }
    Image1.Picture.LoadFromFile(OpenDialog1.FileName); { визуализировать
                                                    его с помощью Image1 }
end;
```

Фактическое создание графической интерфейсной части достаточно сложного приложения занимает минуты (включая обеспечение возможности рисования, масштабирования, сохранения в файл изображения и его восстановления и др.); вариант подобной RAD-системы для **LINUX** получил название **Kylix** (www.borland.ru/kylix/index.html).

В состав штатного ПО включены процедуры поддержки изображений в формате **JPEG**, программист имеет возможность разработать новые классы поддерживающих вывод изображений объектов на основе объекта **TGraphic** (например, для работы с **PCX**-файлами и др.).

Наиболее мощным мультимедийным компонентом является **TMediaPlayer**, инкапсулирующий функции управления устройствами **MCI** (*Media Control Interface*), см. подраздел 4.7.1. Простейший приведенный ниже код служит для выбора нужного мультимедиа-файла и его демонстрации (минимально поддерживаются форматы **AVI**, **WAV** и **MID**)

```
procedure TForm1.BitBtn1Click(Sender: TObject);
{  выбирает и загружает мультимедиа-файл для просмотра  }
begin
  MediaPlayer1.Close; { закрыть ММ-устройство }
  if OpenDialog1.Execute then { если файл выбран… }
    MediaPlayer1.Filename:=OpenDialog1.FileName); { открыть его в
                                                    MediaPlayer1 }
    MediaPlayer1.Open; { начать демонстрацию мультимедиа-файла }
end;
```

Возможно выполнение всех допустимых MCI-команд и практически полный контроль над ММ-устройствами, форматы визуализируемых файлов определяются установленными в системе кодеками – при нужной комплектации ‘проигрываются’ любые типы файлов мультимедиа.

Контрольные вопросы

1. Какие распространенные стандарты записи и воспроизведения мультимедиа-информации известны ?
2. Какие программные пакеты для работы с мультимедиа применяются на современных ПЭВМ ?
3. На каких принципах основано сжатие видеоинформации по MPEG-методу ?
4. Что такое линейный и нелинейный монтаж ?
5. Какими технологическими способами реализуется возможность хранения указанного объема информации на DVD-дисках ?

5. АППАРАТНАЯ ПОДДЕРЖКА МУЛЬТИМЕДИА В ЭВМ

5.1. **Требования мультимедиа к пропускной способности системной шины и шин расширения ПЭВМ**

Как показано выше, обработка мультимедиа-информации требует значительных ресурсов ЭВМ. Несмотря на имеющуюся тенденцию переложения функций на исполняемое (мощным) центральным процессором специализированное ПО (кодеки, драйвер**а**), совершенствование аппаратной части мультимедиа не прекращается.

Важным направлением является увеличение пропускн**о**й способности как центральной шины, так и шин внешних устройств ЭВМ. Так как частоты шин стандартизирована, увеличение пропускн**о**й способности может быть достигнуто за счет увеличения ширины шины или за счет повышения ее ‘интеллектуальности’ (стандарты **PCI**, **AGP**, **USB**, **IEEE 1394** и др.). В настоящее время стандартом ширины шины является 32 бита, в ближайшие годы следует ожидать повсеместного перехода к 64-битовым системам; в августе 2003 г. появилось сообщение о подготовке фирмой **Advanced Micro Devices, Inc.** (*AMD*, www.amd.com) к выпуску процессора, исполняющего 64-битовые инструкции согласно набора **x86-64**. Обходн**ы**м путем является переложение несложных, но предельно часто исполняемых рутинных команд на интеллектуальные внешние устройства (при этом поток информации замыкается внутри этих устройств и не нагружает главную шину ЭВМ, в качестве примера ниже рассмотрена тенденция увеличения вычислительной мощности видеоплат).

Несмотря на (довольно успешные) попытки перенести б**о**льшую часть медиа-траффика с главной шины и шин расширения ПЭВМ внутрь видеоплаты требования (необходимые в особенности при отрисовке сложных трехмерных объектов) к росту пропускн**о**й способности обусловили модернизацию даже такой мощной шины как **PCI** (*Peripheral Component Interconnect*, пропускн**а**я способность 264 Мбайт/сек при разрядности 32 бит и частоте шины 66 МГц). Развитием **PCI** стал **AGP** (*Accelerated Graphic Port*) - новый 64-битный стандарт для подключения имеющих 3D-ускоритель графических адаптеров (потребляемая мощность для стандарта AGP Pro – до 110 Вт), при указанных выше условиях и *режиме сдвоенной передачи данных* пропускн**а**я способность 533 Мбайт/сек, в *режиме 4x* – до 1066 Мбайт/сек, в режиме AGP 8x – до 2132 Мбайт/сек [3, 4].

Современные цифровые фотокамеры и сканеры, ан**а**логовые и цифровые кинокамеры и видеомагнитофоны являются в настоящее время фактически внешними устройствами ПЭВМ; каждое подобное устройство имеет возможность ст**ы**ковки с ПЭВМ по быстродействующим каналам - **USB** (*Universe Serial Bus*, пропускн**а**я способность до 12 Мбит/сек, в версии USB 2.0 определена скорость 480 Мбит/сек) и **IEEE 1394** (*Hight Per-*

fomance Serial Bus, 100÷400 Мбит/сек) – специально разработанных для подключения мультимедиа-устройств высокого качества передачи [4].

Технологии распознавания изображений используются даже в ‘оптические мышах’, отслеживающих перемещение устройства без физического датчика перемещения; встроенный в ‘мышь’ процессор с частотой 100 Гц обрабатывает данные со встроенного в нижнюю часть ‘мыши’ микросканера и формирует сигналы перемещения манипулятора.

5.2. **Мультимедиа-расширения системы команд центрального процессора**

Важным этапом явилось расширение системы команд процессоров фирмы **Intel** (www.intel.com) - в конце 20 века в процессорах серии **Pentium** была введена серия ориентированных на поддержку мультимедиа команд обработки целых чисел **MMX** (*Multi Media eXtension*). Практически за год до этого компания **AMD** (www.amd.com) расширила систему команд процессора **K6-2** подобными по конечным целям инструкциями плавающей арифметики **3DNow!** ‘Голубой гигант’ не остался в долгу и оснастил свои процессоры набором команд плавающей арифметики **SSE** (*Streaming SIMD Extension*, где SIMD – *Single Instruction Multiply Data* означает принцип ‘одна инструкция над множественными данными’), а в процессоре **Pentium IV** ввел **SSE2** (144 дополнительные инструкции для ускорения работы криптосистем, программ сложной визуализации и математических приложений). Подобные (дорогостоящие и снижающие совместимость) расширения системы команд были вызваны не только необходимостью эффективной обработки мультимедиаданных. Особенностью устаревшей к тому времени схемотехники процессоров являлось неэффективное использование встроенного математического сопроцессора из-за задействования одних и тех же внутренних регистров для хранения данных разного типа; вследствие этого работа многозадачных ОС оказалась в высшей степени затрудненной из-за потери данных в регистрах при переключении задач.

‘Целочисленные’ команды **MMX** обеспечивают в первую очередь работу с цифровым звуком (микширование, регулировка громкости, преобразование форматов) и растровой графикой (вывод шрифтов, спрайтов, световые эффекты); ‘плавающие’ инструкции **3DNow!** и **SSE** предназначены для геометрических преобразований, необходимых при отображении трехмерной графики и поддержки сложных алгоритмов компрессии/декомпрессии данных с потерями. С успехом можно применять их в задачах, допускающих распараллеливание вычислений (нейронные сети, решение систем уравнений с ограниченной точностью и др.).

5.3. **Видеокарты современных ПЭВМ**

Идея перенести часть типов**ы**х вычислений на аппаратуру видеоплаты не дает покоя разработчикам практически с начала ‘бума ПЭВМ’; с момента взрывн**о**го развития компьютерных симуляторов и игр прогресс в этом направлении не останавливается. Принятая в настоящее время система машинной графики предполагает дискретизацию сложных линий и поверхностей простыми графическими примитивами, при этом отрисовка (*rendering*) сложных графических объектов сводится к многократному повторению рутинных операций (отрис**о**вка простых площадок-полигонов с *наложением текстур*, затенений и отражений света и др.) переносится на цифровой процессор видеоплаты. В результате современные видеоплаты фактически представляют собой мощный компьютер с работающим на частоте $200 \div 600$ МГц процессором, быстродействующим аналогоцифров**ы**м и цифроан**а**логовым преобразователями (возможность **VIVO** – *Video Input / Video Output* - для работы с ан**а**логовым и цифров**ы**м видео), оперативной памятью $64 \div 256$ Мбайт, собственным BIOS’ом, системой охлаждения и т.п.

В настоящее время наиболее известен графический процессор **GeForce4** фирмы *nVidia* (www.nvidia.com), на основе этого процессора множество фирм выпускают видеоплаты с широким спектром возможностей (www.asus.com, www.albatron.com.tw и др.). MS регулярно дорабатывает библиотеку функций **DirectX** (сейчас версия 9, в качестве составляющих фигурируют **DirectDraw**, **Direct3D** и **DirectSound**, www.microsoft.com/directx), позволяющих повысить скорость и функциональные возможности работы с медиа-информацией; известен разработанный признанным лидером в области мультимедиа фирмой **SGI** (*Silicon Graphics, Inc.*) API (*Application Programming Interface* – набор пользовательских функций*)* **OpenGL** (Open Graphic Library, www.sgi.com/software/opengl), примеры OpenGL-программирования приведены в [1]. Интерфейс OpenGL поддерживается операционными системами для разнообразных аппаратных средств – от ПЭВМ до супер-ЭВМ; работающий совместно с OpenGL видеоадаптер должен *аппаратно выполнять* все б**а**зовые функции OpenGL (преобразование координат, отрис**о**вка полигонов, расчеты освещения, наложение текстур, отсечение), для обеспечения совместной работы используются ICD-драйвер**а** (*Installable Client Driver*). Фирмой *3Dfx Interactive* (www.3dfx.com) разработан программный интерфейс **Glide**, используемый в распространенных в конце 20 века графических видеоадаптерах серии **Voodoo** (www.3dfx.km.ru/develop, www.3dnews.ru.reviews/video). Многие видеокарты аппаратно поддерживают *технологию Z-буфера* (трехмерный массив, в котором сохраняются значения расстояния до точки наблюдения для каждого пиксела р**а**строво-

го изображения, метод эффективен при удалении невидимых точек 3D-объектов).

Для профессионалов предназначены специальные платы обработки медиа-данных; распространенной является линейка плат **miroVIDEO** – от простейших **miroVIDEO** DC10-DC30 до более сложных моделей (например, плата двухпотоковой монтажной системы miroVIDEO DV500) фирмы **Pinnacle Systems** (www.pinnaclesys.com). Эти платы в комплекте с соответствующим ПО (называемые еще системами *нелинейного монтажа*) позволяют осуществлять захват (‘*capture*’) медиаданных с ан**а**логовых или цифровых источников (видеокамера, видеомагнитофон, TV-тюнер), компрессовать их в реальном масштабе времени и записывать на жесткий диск, осуществлять произвольный монтаж (вырезание и склейка кусков, подготовка титров, реализация визуальных эффектов, микширование звука и др.) с последующей записью скомпонованного фильма на видеомагнитофон, CD или DVD.

Таким образом ПЭВМ с платой видеоввода, соответствующим ПО и современными жесткими дисками является не только видеостудией, но и видеомагнитофоном с произвольным д**о**ступом к видеоклипам (на 120 Гбайтовом диске размещается около 200 часов видео при разрешающей способности 320×240 пиксел и глубине цвета 24 бит при сжатии по стандарту MPEG-1).

Заметим, что истинным пионером ‘мультимедиа на ПЭВМ’ явилась фирма **Applle Computer** (www.apple.com); до сих пор каждая ПЭВМ этой фирмы изначально снабжена комплектом устройств мультимедиа, для IBM PC эти устройства являются всего лишь возможными.

5.4. **Устройства аудиовизуального ввода и вывода информации**

К устройствам аудиовизуального ввода относятся микрофон и звуковая плата, сканер, устройства ввода аудиовидеоинформации и (с некоторым сомнением, модем, клавиатура и ‘мышь’), вывода - звуковые системы (‘*колонки*’) и звуковая плата, принтеры, графопостроители (*плоттеры*), дисплеи, устройства *виртуальной* (мнимой) *реальности* (специальные очки, шлемы, перчатки).

Микрофон подключается к разъему *Mic In* звуковой платы (рис. 4.8, в современных ПЭВМ звуковые платы часто интегрированы в состав системной платы - *matherboard*) с помощью малогабаритных разъемов (*‘миниджеков’, jack*) диаметром 3,5 мм. Чувствительность микрофонного входа $(3\div10)\times10^{-3}$ В, вход обычно монофонический (иногда встречается специальное трехконтактное гнездо, у которого находящийся на месте правого канала дополнительный контакт предназначен для подачи питания на *электретный микрофон*). На той же плате обычно находятся разъемы ана-

логовых сигналов - линейный вход (*Line In*, чувствительность 0,1-0,3 В), линейный выход (*Line Out*, уровень выходного сигнала такой же) и выход на акустические системы или наушники (*Speaker Out*); подключение внешнего усилителя акустических систем производится к *Line Out*.

Современные сканеры имеют разрешение до 1200×1200 DPI (точек на дюйм) и больше при глубине цвета $24 \div 36$ бит; физически подключаются к ПЭВМ с помощью интерфейса **USB**, менее быстродействующие устройства подключаются последовательно с принтером к двунаправленному параллельному порту (**LPT** – *Line PrinTer*, обычно используется протокол **ECP** – *Extended Capability Port*). Различают *планшетные* (сканируемый документ располагается на планшете сканера, линейка светочувствительных датчиков осуществляет при этом линейное перемещение вдоль документа) и *рулонные* (документ продольно перемещается цилиндрическим барабаном относительно неподвижной линейки фотодатчиков) сканеры. Универсальным стандартом для прикладного программного интерфейса таких периферийных устройств, как сканеры является **TWAIN** (*Technology Without An Interesting Name*), программное обеспечение для создания прикладных программ также называется TWAIN. Любая периферия, совместимая с TWAIN, может управляться программой, соответствующей стандарту TWAIN. С TWAIN совмест**и**мы большинство выпускаемых сканеров, цифровые фотокамеры и другие периферийные устройства, предназначенные для ввода информации.

К устройствам ввода медиа-информации относятся специализированные видеоплаты (подраздел 5.3), игров**ы**е устройства *(джойстики,* для подключения применяется интерфейс игрового адаптера *Game port*, поддерживающий 4 независимых ан**а**логовых и 4 дискретных входов) и устройства электромузыкальной техники (синтезаторы, записывающие и воспроизводящие устройства, микшеры, устройства специальных эффектов), м**о**гущие подключаться через цифровой интерфейс музыкальных инструментов **MIDI** (подраздел 4.1), представляющий собой последовательный асинхронный интерфейс с частотой передачи 31,25 kГц.

Широко предлагающиеся т.н. называемые ‘мультимедиа-клавиатуры’ мало отличаются от стандартных; отличия заключаются лишь в наличии нескольких дополнительных (‘*hot-keys*’) клавиш, предназначенных для быстрого в**ы**зова некоторых связанных с мультимедиа- и InterNet-возможностями работы.

Модем (*МОДулятор/ДЕМодулятор*) служит для связи ПЭВМ между собой с использованием (обычно) телефонных линий, реально интерес для целей мультимедиа представляют модемы со скоростью не менее 57,6 Кбит/сек. Интересна появившаяся в последнее время тенденция перен**о**са большинства функций модема на (мощный) центральный процессор ПЭВМ (т.н. *soft-модемы*).

Обычная ‘мышь’ (и ее разновидность – *трекболл*) являются 2D-устройствами (возможно управление по двум координатами). Некоторое приближений к пространственному управлению представляет *джойстик* (и его более сложные разновидности). Интересной моделью является беспроводная мышь **GiroMouse** (фирма *Dimond*), движущаяся по поверхности подобно обыкновенной ‘мыши’ и позволяющая моделировать трехмерное перемещение при поднятии и перемещении в воздухе (т.н. **3D-mouse**). В состав *шлема виртуальной реальности* VFX-1 (см. ниже) входит специальный манипулятор **Cyber Puck**, функционально являющийся трехмерным ан**а**логом привычного джойстика.

В настоящее время предлагаются тысячи моделей звуковых систем (звуковой) мощностью до нескольких сотен ватт, на смену стандартным системам их двух динамиков все чаще приходят системы и 5+1 (5 динамиков для создания квадро-эффекта плюс один мощный низкочастотный динамик – ‘сабвуфер’); при этом обычным является программное разделение обычного двухканального стереозвука на псевдоканалы (реально указанное число каналов возможно лишь при воспроизведении звука с DVD-дисков на ПЭВМ, оборудованных поддерживающими стандарты **Dolby Digital 5.1** или **DTS** (*Digital Theatre System*) **5.1** звуковыми картами).

Парк современных принтеров включает тысячи наименований, классификация разделяет все устройства на обеспечивающие черно-белую и цветную печать. Стандартное подключение к ПЭВМ осуществляется посредством параллельного порта LPT, все чаще встречаются USB-подключаемые устройства. Стандартное разрешение принтеров – $600 \div 1200$ DPI.

Подавляющее большинство ‘черно-белых’ принтеров используют *технологию лазерной печати* (применяется свойство фоточувствительных материалов изменять свой поверхностный заряд в зависимости от освещенности, световой узор формируется лазерным или светодиодным излучателем на поверхности фоточувствительного барабана с краской – ‘*тонером*’, в результате контакта барабана с бумагой на последней тонер остается в незасвеченных местах, далее изображение термически закрепляется), более дешевые модели используют *струйную технологию печати* (строго дозированное распыление красящего вещества формирует изображение на бумаге).

Цветные принтеры в большинстве являются струйными (в соответствии со стандартом **CMYK** используются 4 контейнера с краской, появляются сообщения о принтерах с 6 красками), см. подраздел 2.1. Лазерные устройства (многократная лазерная печать т**о**нерами разного цвета) обеспечивают прекрасное изображение, но д**о**роги.

Графопостроители (*плоттеры*) являются устройствами печати на бумаге больших форматов (до A1/A0 – $594 \times 841/1189 \times 841$ мм соответственно

- и выше), могут быть *планшетными* или *рулонными*. Большинство современных плоттеров используют струйный принцип печати (т.е. являются большеформатными принтерами), для подключения используют **LPT** и **USB**. Устройства *плоттерной резки* снабжены специальной режущей головкой и предназначены для изготовления в основном рекламных плакатов из разноцветной пластиковой пленки. Плоттеры д**о**роги вследствие необходимости обеспечения высокой механической точности позиционирования красящей головки при указанных линейных размерах носителя и обычно предназначены для совместного использования; в таких случаях они снабжаются встроенным процессором, буферным жестким диском большой емкости и возможностью подключения к сети. Трехмерный графопостроитель является скорее мечтой, идеи реализации (движение пипетки с красителем в ‘аквариуме’) слишком далек**и** от реализации (хотя имеются удачные примеры – создание объемных фигур посредством полимеризации специального компаунда под действием светового излучения, сфокусированного в определенной точке емкости).

Дисплей (монитор) является фактически высококачественным телевизором, работающим на принципе активизации слоя люминофора электронным лучом в вакууме (в ЭЛТ – *электронно-лучевой трубке*) или на принципе *твердотельного светоизлучения* (светодиоды, плазменные ячейки и т.д.). Качество изображения характеризуется разрешающей способностью (до 2048×1536 точек), яркостью, цветопередачей, частотой кадров. При малой *частоте кадров* наблюдается быстрая уставаемость пользователя компьютера, рекомендуется частота кадров не ниже $80 \div 100$ Гц при размере экрана дисплея по диагонали не менее 17 дюймов. Для подключения монитора к графическому адаптеру (видеокарте) используются специализированные интерфейсы, по которым передается информация о мгновенном значении основных цвет**о**в (*Red, Green* и *Blue*), сигналы строчной развертки и синхронизации по кадрам. При подключении чаще используют ан**а**логовый интерфейс **RGB** (глубина цвета 24 бит = 16’777’216 цвет**о**в), цифровой с ан**а**логовыми сигналами **DVI** (*Digital Visual Interface*, www.ddwg.org) и чисто цифровой интерфейс **DFP** (*Digital Flat Panel*, www.dfp-group.org). Интерфейсы мониторов в большинстве своем стандартизированы организацией **VESA** (www.vesa.org), [4].

В мультимедиа часто применяют *видеопроекторы*, функционально аналогичные дисплеям, но проецирующим изображение на экран размеров в несколько квадратных метров. Лучшие из них имеют USB-вход, световой поток до $1000 \div 3000$ люменов (до $10 \div 12$ тыс. люменов для кинотеатров) и контрастность изображения до 400:1. *Плазменные панели* с диагональю экрана $42 \div 50$ см и выше отличаются плоской конструкцией (толщина $7 \div 10$ см) и очень высокой контрастностью (до 2300:1) изображения.

Устройства виртуальной реальности (**VR**) призваны (в идеале) переносить пользователя в сконструированный с помощью компьютерных технологий мир, при этом возможно полное замещение реального мира на VR или ‘неполное’ погружение (обеспечиваемое, например, стандартной системой ‘дисплей – звуковые системы – игровой манипулятор’).

Особо важным является погружение в мир видео (исходя из получения человеком б**о**льшей части информации именно посредством зрения). Согласно подсчетам число различающих цв**е**тность фоторецепторов в сетчатке глаза около 6×10^6 (т.е. оценка разрешающей способности некоего охватывающего все поле зрения VR-супермонитора дает порядок 2400×2400 точек). В реальности приведенная оценка в высшей степени вульгарна, проблема много сложнее (в первую очередь вследствие весьма сложной психофизиологии процесса зрения).

Важным атрибутом человеческого зрения является *трехмерность* (*объемность*, ***3D***). Глаза человека воспринимают объекты под разными углами, в дальнейшем два независимых изображения анализируются мозгом и в результате их сопоставления формируется **о**браз предмета, его признаки и глубина изображения. Расстояние (*база*) между глазами человека составляет $6\div8$ см, мозг человека анализирует расстояние, основываясь на различии между изображениями, получаемыми левым и правым глазом (это различие называется *параллаксом зрения*); именно с помощью этого эффекта и создается представление об трехмерности (объемности) изображения.

Использующие стереослайды системы известны уже несколько десятилетий. Применялись также очки с цветными стеклами (красно-синие, красно-зеленые), в 70-е годы даже выпускались цветные телевизоры с вызывавшими смещение красного цвета на изображении линиями задержки, при просмотре телепрограмм на таком устройстве возникало некое подобие стереоскопичности (в горизонтальной плоскости). Подобные системы ушли в прошлое с развитием современных технологий.

Простой и дешевый способ формирования трехмерных изображений заключается в поочередном формировании изображения для левого и правого глаза. При этом оба изображения выводятся на единый экран монитора поочередно, а разделение изображений выполняется с помощью специальных скоростн**ы**х затв**о**ров, по очереди перекрывающих поток изображения от монитора; современные затворы основаны на принципе изменения прозрачности жидких кристаллов и выполнены в форме очков, через которые зритель смотрит на экран монитора. Линзы таких очков по очереди (синхронно с изменение изображения на мониторе для левого и правого глаз) становятся непрозрачными с частотой $150\div200$ Гц, на таком принципе формирования стереоизображений построены очки **CrystalEyes PC** (фирма *StereoGraphics Corp.*, www.stereographics.com), **3D Stereo Set**,

3D Max (*Kasan Electronics*, www.kasan.co.kr), **CyberMaxx 3D** (*VictorMaxx Technologies, Inc.*), **Holographic 3D**.

Второй способ более сложен и д**о**рог в реализации, однако обеспечивает значительно более глубокое ‘погружение’ в VR; способ заключается в использовании двух отдельных (для каждого глаза) устройств вывода видеоинформации, разделение изображений обеспечивается конструкцией системы. Первые разработки в этом направлении проведены еще в конце 60-х г.г. фирмой **IBM**, проверялись два принципа – использование световодов для передачи в каждый глаз отдельно видеоинформации, сформированной двумя ЭЛТ-дисплеями (схема ‘2 дисплея – 2 световода – 2 глаза’) и применение двух миниатюрных дисплеев, вмонтированных в специальный шлем и расположенных каждый напротив ‘своего’ глаза.

Первые способ не отличался ни простотой, ни компактностью, ни низкой стоимостью. Реализация второго способа стала возможной лишь с появлением миниатюрных цветных дисплеев на жидких кристаллах (LCD, *Liquid Cristal Display*), примерами реализации являются современные системы **I-Glasses** и **VFX-1**.

I-Glasses (фирма *i-O Display System*, www.i-glassesstore.com) представляет собой компактное устройство, обеспечивающее вывод видеоинформации через два малогабаритных дисплея, входящих в состав данного устройства, которое крепится на лоб пользователя и обеспечивает раздельную и независимую подачу видеоизображений с минидисплеев. Видеосигнал для i-Glasses подается в стандарте NTSC, что позволяет использовать это устройство не только в качестве компьютерного 3D-монитора, но и в качестве приставки к телевизору или видеомагнитофону для индивидуального просмотра видеопрограмм.

Одной из наиболее удачных моделей систем человеко-машинного интерфейса является **VFX-1** (*Forte Technologies, Inc*., www.fortevr.com). В основу VFX-1 положен шлемоподобный (*VR-helmes*) дизайн, обладающий многими преимуществами. Пользователь смотрит через регулируемую оптику, что создает иллюзию большого экрана. Для достижения эффекта трехмерности на каждый миниэкран шлема VFX-1 через специальный интерфейс посылается соответствующий видеосигнал от компьютера: отдельный для левого и отдельный для правого монитор. Датчик магнитного поля Земли контролирует положение головы пользователя и постоянно вводит эту информацию в компьютер (поворот головы в шлеме соответствует повороту головы в виртуальном мире), шлем оборудован встроенными высококачественными динамиками и микрофоном.

Системы, обеспечивающие вывод 3D-изображений - i-Glasses и VFX-1 -породили целый ряд устройств с более высокими параметрами. В качестве примера можно привести i-Glasses X2 и i-Glasses ProTec (фирма *i-O Display System*), VFX-3D представляет собой усовершенствованный вариант VFX-

1. Известны устройства **Virtuality Scuba PCT**, **Glasstron PLM-55** и **Glasstron PLM-S700E** (фирма *Sony*), **Philips Scuba** (www.scubafx.com), **Datavisors 10x** и **Datavisor 80** (*Virtual Research System*, www.virtualresearch.com), виртуальные бинокуляры **V8 Binoculars** (www.virtualresearch.com), **Virtual Binoculars** (*n-Visio*, www.nvis.com). Значительно б**о**льшими возможностями обладают системы класса **CyberTron**, позволяющие с помощью системы вращающихся металлических обручей изменять ориентацию тела реципиента в пространстве (‘Газонокосильщик’).

В современных системах подобного рода поддерживается разрешение до 1024×768 с различной частотой кадров, глубина цвета равна $16\div24$ бит, системы совмест**и**мы со всеми последними типами 3D-ускорителей, используют интерфейс USB для связи с ПЭВМ, используется более совершенная система ориентации, в некоторых случаях реализуется *полупрозрачность VR-изображения* (возможно частично видеть окружающий реальный мир).

Подобные системы реально применяются в промышленности. Например, компания *Virtual Research Systems* занимается разработкой и системной интеграцией виртуальных комплексов для решения более серьезных задач. В первую очередь это различные научные и промышленные проекты, требующие создания сложных объемных мир**о**в или визуализации физических процессов, которые невозможно (или очень дорого) реализовать ‘вживую’. Типичным примером может служить процесс разработки узлов и компонентов для машин класса ‘Формула-1’, практически полностью на первых этапах осуществляемый с помощью компьютерного моделирования. И только когда все детали доведены и состыкованы друг с другом, а виртуальные испытания подтверждают заданные аэродинамические и технические параметры, начинается физическое воплощение машины.

Несмотря на очевидный прогресс в области совершенствования систем 3D-изображений, некоторые эксперты считают, что будущее за лазерными устройствами, обеспечивающими проецирование цветного 3D-изображения *непосредственно на сетчатки обоих глаз*. Такие исследования уже ведутся, и утверждается их успешность. Близкий к идеальному вариант передачи мультимедиа-информации (в виде неких ‘мыслеобразов’) непосредственно в мозг (возможно, на зрительный и слуховой нервы) пациента пока скорее фантастичен (хотя первые примитивные эксперименты уже проведены).

Обеспечение раздельной подачи звука к ушам пользователя решается значительно проще, особенно в ‘шлемах виртуальной реальности’; остается проблема тактильных ощущений.

К обеспечивающим такт**и**льные ощущения устройствам относятся имитирующие осязание ‘мыши’ (например, мышь **FEELit Mouse**,

www.immerse.com, позволяющая почувствовать рельеф изображения, ‘выпуклость’ кнопки, ‘вес’ копируемого файла/каталога или ‘сопротивление’ изменению размеров окна), специальные тактильные перчатки (*gloves*) и ‘виртуальные костюмы’.

Перчатки *gloves* позволяют в соответствии с требованиями эксплуатируемых программ осуществлять ввод информации в компьютер с помощью различных манипуляций руками, кистями рук, пальцами. Например, указывая пальцем на объект в VR, можно вызвать определенные действия: перемещение объекта в VR-пространстве, изменение его размеров, цвета, формы и т.д. С объектом в VR может быть сопоставлен объект в реальном мире, таким образом можно осуществлять управление из VR. С помощью *gloves* может осуществляться вывод пользователю такт**и**льной информации (такт**и**льных ощущений) типа ‘твердый/мягкий ’, ’холодный/горячий’ и др. Такие устройства сложн**ы** и д**о**роги; предназначенные для профессионального применения *gloves* имеют несколько сотен датчиков: растяжения, сжатия, перемещения и т.д. Эти датчики фиксируют и измеряют пространственные перемещения как совместные – плеча, руки, кисти, так каждой выбранной точки в отдельности - каждого пальца и каждой фаланги. При этом необходимо учитывать, измерять и вводить в компьютер каждый изгиб и поворот.

Наиболее полным набором оборудования для VR является *виртуальный костюм*, состоящий из обтягивающего комбинезона с большим количеством магнитных с**е**нсоров, отслеживающих движения всех частей тела. К нему добавляется VR-шлем, перчатки *gloves* кисти и провода для присоединения всего этого к компьютеру. В результате получается полное ощущение погружения в виртуальную реальность. Датчики на костюме действуют лишь в магнитном поле, так что для функционирования такого костюма необходимо постоянно находиться на или внутри некоей системы внешних датчиков.

Для программирования обеспечивающих обратную тактильную связь устройств используется технология **Force Feedback** (*силовая обратная связь*, www.force-feedback.com, данная технология стандартизирована и встроена в DirectX 5.0, выпущен аппаратно поддерживающий Force Feedback специализированный процессор), особенно в ‘устройствах с силовой обратной связью’ преуспела фирма **Immersion Corp**. (www.immerse.com), специализирующейся на создании устройств для медицины.

Несмотря на указанные технические достижения до сих пор не удалось создать *дешевую и эффективную* систему для использования виртуальной реальности. Кроме технических недостатков, есть и другие факторы, влияющие на распространение VR-систем. Так, до сих пор не ясно, какое влияние оказывают эти системы на здоровье - в частности, *на зрение*. Дело

в том, что глазные мышцы не способны длительное время находиться в напряжении, а именно это и происходит во время сеанса виртуальной реальности. В противном случае глаза быстро устают, глазные мышцы ослабляются, в результате чего происходит быстрое ухудшение зрения. Однако чаще всего противниками виртуальной реальности высказываются опасения на счет *психического здоровья* при применении систем виртуальной реальности. Известно, что человеческая психика больше всего подвержена влиянию, когда человек на чем-то сосредоточен, а это и происходит во время сеанса виртуальной реальности.

При формировании информации данного раздела использованы материалы с сайта www.3dnews.ru и литературные источники [1, 3, 4].

Контрольные вопросы

1. Какие пути повышения пропускной способности общей шины считаются реальными на сегодняшний день ?
2. Для решения каких задач используются дополнительные (наборы MMX, SSE, 3DNow) команды центральных процессоров современных ПЭВМ ?
3. С какой целью разрабатываются современные видеокарты с мощными процессорами и большим объемом оперативной памяти ?
4. Каким образом человек ощущает эффект объемности звука и видео ?
5. Какие типы устройств вывода аудиовизуальной информации используются и каковы их основные технические характеристики ?
6. В чем достоинства и недостатки современных систем виртуальной реальности (очки, шлемы, перчатки *gloves*, специальные костюмы) ?

6. МУЛЬТИМЕДИА-СИСТЕМЫ МОДЕЛИРОВАНИЯ ОКРУЖАЮЩЕГО МИРА

Один из возможных путей познания окружающего мира является моделирование явлений и процессов мира. Эффективность применения мультимедиа в научно-технических задачах подтверждается системами визуализации результатов решения сложных (особенно характеризующихся огромным объемом выходных цифровых данных – например, *метод конечных элементов*) задач; см. пакеты **Maple** (фирма *MapleSoft*, www.maplesoft.com), **MathCad** (*MathSoft*, www.mathsoft.com), **MatLab** (*MathWorks*, *Inc*., www.mathworks.com), **Mathematica** (*Wolfram Research*, www.wolfram.com) и др.

ЭВМ оказались удачным механизмом создания *виртуальной реальности* (**VR**, *Virtual Reality*) – представленного в виде цифровых данных и операций над ними подобия окружающего мира; особенное развитие этот

метод получил в т.н. *компьютерных играх* (симуляторах взаимодействия пользователя с заданным виртуальным миром).

К средствам создания VR относятся пакеты **3D StudioMax** фирмы *Discreet*, www.discreet.com, **Maya** (*Alias Wavefront*, www.aliaswavefront.com/maya), **LightWave** (*NewTeck*, www.newtek.com/products/lightwave), **Softimage 3D** (www.avid.com, www.softimage.com) и др. Практически все они используют трехмерную (3D) модель представления объектов, основанную на задании (относительно ограниченного) количества р**е**перных точек (*вершин*) с последующим построением опирающихся на вершины плоских многоугольников – полигонов (*полигональная модель*), отрис**о**вкой текстур (*texture,* стиль закрашивания поверхности, создающий иллюзию объемности), расчетом освещения и др. Полигональная модель ориентирована на современные программные (технологии *рендеринга*, *трассировки лучей*, *методы Гуро* (Gourand) *и Фонг*а (Phong) для закрашивания поверхностей 3D-объектов) и аппаратные (отрисовка полигонов с наложением текстур в современных видеокартах) средства.

Кроме полигональной модели описания поверхностей используется аналитическая (описание в виде математических формул, зачастую с использованием сплайнов – обычно *кубических сплайнов Безье*) и воксельная (при этом трехмерные объекты образуют элементарные элементы объема – *вокселы*) модели [1]. Наиболее мощные пакеты такого рода поддерживают заданные пропорции между координатами вершин (обеспечивая ц**е**лостность объекта при трансформациях) и связь объектов в более сложные образования, существующие по заданным законам (‘*мир**ы***’, ‘*авата**а**ры*’).

Простейшим преобразованием координат объектов является *линейное преобразование* (зависимость координат каждой точки от исходных координат является линейной, математически преобразование сводится к определенным матричным операциям для случаев сдвига координат, растяжения-сжатия, поворота изображения), вариантом линейного является *афинное преобразование*). К *нелинейным преобразованиям* относятся, например, общего вида *проекции* (способы отображения объектов на графическом устройстве).

Часто применяемым эффектом является *морфинг* - плавное ‘превращение’ одного изображения в другое, во время которого конкретный элемент первого изображения ‘перетекает’ в элемент второго изображения (например, при ‘морфировании’ одного автомобиля в другой колесо первого превращается в колесо второго). ЭВМ не может выполнить морфинг двух изображений самостоятельно – оператору-художнику требуется задать соответствие элементов первого изображения элементам второго, задаются и другие параметры (используются специальные редакторы). Способ задания соответствия зависит от редактора - используются точки,

линии, полигоны. Процесс морфинга условно разбивается на три части: *warping* (преобразование изображения, при котором элементы изображения ‘пытаются’ принять положение и форму элементов второго изображения), *tweening* (интерполяция двух изображений для получения плавной анимации) и *dissolving* (слияние двух изображений).

При закрашивании поверхностей объектов (с учетом освещенности) используются модели *зеркального отражения* и *диффузного отражения* света.

С целью компьютерного представления формы реальных объектов (сложных деталей, человеческих лиц) используются системы лазерного сканирования объекта с оцифровкой в реальном масштабе времени, известны также механические системы подобного рода (напр., устройство **MicroScribe-3D** фирмы *Immersion Corp.*).

С конца 90-х годов важное значение в VR приобретают *системы оцифровки движений* (устройства, позволяющие преобразовывать движения в цифровой вид и вводить эти данные в ЭВМ для дальнейшей обработки), для телевидения важн**ы** системы датчиков оцифровки движения реальных актеров. Эти системы применяются для ‘оживления’ виртуальных персонажей - работы студий **Пилот** (pilot.cool.ru) и **BS Graphics** (www.bsgraphics.ru). Существуют три вида таких систем: механические, электромагнитные и оптические; обычно на теле актера закрепляется 16-20 датчиков.

VR-технологии активно используются в науке и технике (визуализация при моделировании сложных процессов, решение задач компоновки механизмов) и при создании художественных произведений (особенно после цифровой разработки визуальных эффектов ‘Терминатора’, ‘Звездных войн’ и ‘Газонокосильщика’).

Для описания виртуальных мир**о**в создан язык **VRML** (*Virtual Reality Modeling Language*) [7], формальная спецификация языка см. www.web3d.org/technicalinfo/specifications/specifications.html. Первые попытки разработки подобных языков относятся к концу 80-х г.г. (формат *Open Inventor*, проект *Labirinth*), в разработке VRML активно участвовала фирма **SGI** (*Silicon Graphics, Inc.*, www.sgi.com). VRML поддерживает иерархические преобразования 3D-объектов, источники света, возможность смены точки наблюдения, различные свойства материалов, наложение текстур, анимацию, интерактивность и является стандартом обмена 3D-данными между приложениями. Создавать сложные VRML-объекты возможно, например, в вышеупомянутом **3D StudioMax** (пакет поддерживает импорт и экспорт VRML-файлов) и использовать в дальнейшем н**у**жным образом.

На сегодняшний день VRML является наиболее мощным средством интеграции двух- и трехмерных объектов, текста и мультимедиа.

Контрольные вопросы

1. Каким образом мультимедиа-системы помогают познавать окружающий человека мир ?
2. На каких принципах основано представление объектов в системах виртуальной реальности ?
3. Что такое VRML и какими возможностями обладает этот метод представления трехмерных объектов ?

7. МУЛЬТИМЕДИА И ГЛОБАЛЬНАЯ СЕТЬ INTERNET

7.1. Программное обеспечение мультимедиа в сети INTERNET

Так как мультимедиа представляет интерес для широкого круга людей, использование его в сети InterNet весьма привлекательно. Возможности работы с мультимедиа были заложены уже в самом начале существования InterNet и в настоящее время являются штатными для всех распространенных броузеров.

Распространенные средства просмотра WEB-страниц (*броузеры*) поддерживают (путем использования коротких изменяющих состояние броузера предписаний - *тегов*) работу с неподвижными изображениями, звук и анимацию (интерактивность реализуется обычно с помощью языков *Java* или *JavaScript*).

Описание содержащегося в файле EXCLAIM.GIF изображения в формате GIF выглядит в тексте WEB-страницы на языке HTML следующим образом (текст квалификатора ALT ‘всплывает’ при задержке курсора ‘мыши’) в течении нескольких секунд на фоне изображения:

```
<img src="./gif_89/exclaim.gif" width=30 alt='Это красивая
картинка, согласны ?'>
```

Демонстрация видеоролика SKY_MOV.AVI может быть реализована следующим образом (квалификатор CONTROLS вызывает появление линейки управления демонстрацией в нижней части окна с видеофрагментом, start=”mousemove” задает режим начала демонстрации после помещения курсора ‘мыши’ на область предполагаемой демонстрации, src=”./gif/excuse.gif” объявляет появляющийся в окне при невозможности демонстрации видеофрагмента графический файл, loop=”3” и loopdelay=”1500” определяют демонстрацию видеофрагмента 3 раза с задержкой перед показом каждого в 1,500 сек):

```
<img dynsrc="./avi/sky_mov.avi" src=”./gif/excuse.gif”
```

**start=”mousemove” alt='Это красивый ролик, как думаете ?'
loop=”3” loopdelay=”1500” controls >**

Для однократного воспроизведения звукового файла nocturne.mid в броузере MS Internet Explorer (www.microsoft.com/windows/ie/default.asp) следует воспользоваться строкой предписаний

<bgsound src="./mid/nocturne.mid" loop=1>

а в броузере Netscape Navigator (home.netscape.com/computing/ download) строкой

<embed src="./mid/nocturne.mid" hidden="true">

Образец использования JavaScript для обеспечения интерактивности демонстрируется ниже (при ‘наведении’ курс**о**ра ‘мыши’ на область изображения файла RBAN_ANI.GIF активизируется функция rightMouseOver(), при ‘уходе’ курсора – функция rightMouseOut()):

**<img src="./gif_89/rban_ani.gif" align="right" border=0
height=50 hspace=3 name="pict_r"
onMouseOver="rightMouseOver()"
onMouseOut="rightMouseOut()"
alt="Это пример активизации JavaScript-функций, вызываемых в ответ
на события onMouseOver и onMouseOut”>**

Постоянно разрабатываются новые средства обеспечения мультимедиа в сети InterNet. Предложенная фирмой *Macromedia, Inc.* (www.macromedia.com) технология векторной графики **MacromediaFlash** позволяет передавать клиенту несложные анимированные изображения совместно со звуком (примером выполненного по данной технологии образца антиискусства является печальной памяти проект ‘Масяня’), поддержка *MacromediaFlash* встроена в современные броузеры; технология обеспечивает интерактивнсть (на чем основано множество несложных сетевых игр).

Описанный в предыдущем разделе язык описания VRML описания виртуальной реальности в сети InterNet является фактически трехмерным ан**а**логом **HTML**. Файл VRLM является текстовым и (подобно давно используемому методу генерации ответов WEB-сервера клиенту в формате HTML по стандарту *CGI*) может быть сгенерирован (или модифицирован по шаблону) стор**о**нней программой ‘на лету’,

Из известного ПО поддержки VRML можно рекомендовать модули расширения (*‘плагины’, plugins*) **Cosmo Player** (фирма *Cosmo Software*, www.cosmosoftware.com), **WorlView** (www.platinum.com), **Cortona**

(*ParallelGraphics*, www.paragraph.ru) и специализированный VRML-редактор фирмы ParallelGraphics.

Благодаря встроенности в InterNet (интерпретатор VRML выполнен в виде модуля расширения стандартных *броузеров*) достигается интерактивное существование пользователя среди '*аватаров*' – спроектированных виртуальных миров технического или художественного направлений.

Вполне определенным недостатком является невозможность поиска нужных мультимедиа-материалов в сети InterNet. В настоящее время достаточно эффективна разработана технология поиска только текстовых данных, количественные трудности 'поиска по образцам' изображений, видео и аудио-материалов заключаются в необходимости огромных ресурсов вычислительных систем с целью анализа и распознавания видео- и звукоряда.

7.2. **Обучение с использованием мультимедиа**

Согласно современным воззрениям, не менее 70% информации человек получает посредством зрения, поэтому использование мультимедиа при обучении должно быть эффективным

Использовать мультимедиа-технологии можно как для приобретения дополнительных знаний (*виртуальные музеи, обучающие пособия* и др.), так и в качестве основного курса обучения (в случаях сложности физического доступа обучаемых к центрам обучения). В случае использования *технологий удаленного доступа к информации* (например, посредством сети InterNet) говорят о *дистанционном обучении (образовании)*.

Однако использование мультимедиа как единственного средства обучения малоэффективно, так как традиционный процесс обучения основан на личном общении обучаемого с учителем; основным недостатком дистанционного обучения является как раз недостаток (вследствии низкого уровня интерактивности) личностных акцентов указанного взаимодействия (неразвитость технических средств связи здесь является вторичной).

В настоящее время в России имеющие значительный процент мультимедиа-технологий InterNet-обучающие системы нашли применение в областях знаний, основанных на простом копировании навыков без глубокого осмысления предмета (изучение иностранных языков, бухгалтерское дело, право, финансы и кредит и т.п.), причем получаемые знания обычно являются базой не основного, а дополнительного образования.

Известны системы дистанционного образования **Национального технологического университета** в США (*National Technological University, NTU*), объединяющего уже в начале 1990-х г.г. 40 университетских инженерных центров и более 1100 студентов. На рубеже 21 века в США методами дистанционного обучения получали образование более 1 млн студен-

тов посредством системы **PBS-TV** (*Public Brodcasting System*), функционирует программа обучения взрослых (*PBS Audit Learning Service*), армия США также интенсивно использует дистанционное образование.

В Европе дистанционное обучение ведут **Открытая школа бизнеса** *Британского открытого университета*, **Национальный университет дистанционного образования** (*Universidad National de Education a Distancia, UNED*) в Испании, **Национальный центр дистанционного обучения** (*Centre National D'Enseignement a Distance, CEND*) во Франции, **Балтийский университет** (*The Baltic University*) в Швеции.

В России идеи дистанционного образования активизирует **Московский Государственный университет Экономики, Статистики и Информатики** (*МЭСИ*), функционирует также **Евразийская ассоциация дистанционного образования** (*ЕАДО*, www.dist-edu.ru).

Прекрасно показывает себя мультимедиа при оформлении виртуальных (электронных, доступных через сеть InterNet или на CD) музеев. Известны виртуальные музеи Лувра (www.louvre.fr), Эрмитажа (www.hermitage.ru), Третьяковской галлереи (www.tretyakov.ru); общая информация о музеях России доступна на адресе www.museum.ru/defruss.html. Из технических представляют интерес Политехнический музей (www.poymus.ru/rus), виртуальный музей информатики (schools.keldysh.ru/sch444/MUSEUM), музей истории отечественных компьютеров (www.bashedu.ru/konkurs/tarhov/russian/index_r.htm), виртуальный компьютерный музей (www.computer-museum.ru/index.php). Удачная библиотека ссылок на сайты виртуальных компьютерных музеев мира расположена на адресе www.bashedu.ru/konkurs/tarhov/russian/anot_mus.htm.

Эффективно применение мультимедиа при:

- изучении иностранных языков – известны электронные версии экспресс-метода Илоны Давыдовой (ilonadavydova.com), мультимедийный курсы английского языка **Reward Intern@tive** (фирма *Young Digital Poland Multimedia*, www.reward.ru) и **Double English** (*Ньюком*, www.cd-rom.ru)
- обучении детей - мультимедиа-учебники **Открытая физика**, **Открытая математика**, **Открытая астрономия**, проект '**Открытый колледж**' www.college.ru (*ТОО НЦ 'Физикон'*, www.physicon.ru), **Биология**, **Химия** (*1С:Репетитор*, repetitor.1c.ru), интерактивная образовательная программа **Мир Алисы** (*1С/КомТех*, www.1c.ru), **Виртуальная школа** (*Кирилл и Мефордий*, www.km.ru, vschool.km.ru)
- повышения уровня интеллектуальности – CD **Династия Романовых**, **Художественная энциклопедия зарубежного классического исскуства** (*АО Коминфо*, www.cominf.ru)

- ознакомлении с явлениями природы - **The Way Things Work** (*DK Multimedia*, www.mammonth.net), **The Unexplained** (фирма *FlagTower*, www.flagtower.com)
- виртуальных прогулок по городам мира и истории – **Москвоведение**, **История России и ее ближайших соседей** (*Кордис&Медиа*, www.cordis.ru), **Киев и киевляне** (*Armitage's Studio*, www.virtual.kiev.ua)
- туризма - CD **Travel Market** (АМК 'Респект')

и десятков других приложений. К подобному применению близки грандиозные управляемые ЭВМ светомузыкальные представления '*Звук и свет*' (вблизи пирамид и Сфинкса в Гизе, храмов Луксора etc).

В настоящее время InterNet представляет хорошие возможности для получения односторонней мультимедиа-информации, однако интерактивные возможности явно малы и фактически ограничиваются посылкой клиентом коротких кодовых или текстовых сообщений.

Кодовые сообщения широко используются в сети InterNet (например, сообщения 'мыши' с целью выбора точки зрения в пространстве виртуальной реальности и т.д.) и придают сетевой работе некоторое подобие интерактивности.

Однако 'интерактивность' такого рода явно искусственна и не может заменить привычный (словесный, интонационный, мимический) контакт людей. В технологии дистанционного обучения введено понятие *тютор* (учитель), однако учителя в привычном смысле (как человека, передающего свои знания обучаемому привычным образом) в сети InterNet нет (обучение сводится к общению 'по переписке', что вряд ли эффективно).

Таким образом развитие мультимедиа в сети InterNet в ближайшие годы, вероятно, должно пойти по пути повышения интерактивности (и, в первую очередь, приближения качества этой интерактивности к привычному человеческому). Для решения этой проблемы придется разработать необходимые технологические приемы – от простого повышения пропускной способности InterNet-коммуникаций до серьезного психофизиологического анализа процесса общения с целью реализации его техническими способами. Автор данной работы не предвидит в обозримом будущем корректного решения указанной проблемы.

Контрольные вопросы

1. Каковы штатные возможности поддержки мультимедиа в распространенных броузерах ?

2. На чем основана идея применения сети InterNet при процессе обучения ? Какие InterNet-технологии используются в дистанционном обучении ? В чем ограниченность дистанционного образования ?
3. Какие пути развития мультимедиа в сети InterNet можно наметить ?

8. **БУДУЩЕЕ МУЛЬТИМЕДИА**

В настоящее время мультимедиа делает первые шаги и, как это всегда бывает в подобных случаях, излишне ‘техницировано’ (о чем говорят, в частности, содержание и объем данной работы) - существует множество ММ-стандартов, типов носителей информации и методов работы с ней, что зачастую затрудняет широкое применение мультимедиа.

8.1. **Программное обеспечение будущих поколений для поддержки мультимедиа**

8.1.1. **Специализированные проекты для работы с мультимедиа**

Ежемесячно возникают новые проекты применения мультимедиа в самых различных областях жизни, часто они ориентированы на широкий круг пользователей и реализованы с поддержкой сети InterNet.

В конце 90-х г.г. разработан InterNet-протокол **WAP** (*Wireless Application Protocol*) передачи информации из InterNet на мобильный аппарат (в конце 1999 г. Microsoft Corp. выпустила поддерживающий WAP броузер **Mobile Explorer**). Некоторые InterNet-компании заявили о поддержке этого (несовместимого с традиционным) стандарта (например, сайт **wap.infoart.ru** фирмы **Infoart Stars**, ‘увидеть’ который в начале внедрения технологии можно было лишь с помощью специализированного сотового телефона. Проблему несовместимости решила компании **Motorola** (создавшая первый основанный на Java телефон, обеспечивающей полноценный д**о**ступ к стандартным HTML-сайтам в InterNet) и другие. Почтовые услуги для владельцев телефонов с протоколом WAP предоставляет также сервер www.iname.ru и др.

В начале 21 века встраивание WEB-камер в сотовый телефон стало обычным, в связи с расширением пропускн**о**й способности линии можно ожидать возможность работы с видео. Таким образом, и ресурсы InterNet и возможность свободного обмена фото- видеоинформацией, становятся доступными пользователям в любой точке планеты.

К особо интересным разработкам можно отнести, например, моделирование эволюционного процесса **TechnoSphere** (www.technosphere.org.uk), предполагающий проектирование пользователем фантастических существ и отслеживание в дальнейшем их эволюции

в сообществе подобных, сконструированных другими пользователями (включая контакты, размножение, наследственность, борьбу за существование). С другими виртуальными мир**а**ми в InterNet можно ознакомиться по адрес**а**м www.cybertown.com, www.worlds.net, www.vr.org.au.

Другим примером является VR-моделирование реального мира с целью как упрощения выполнения рутинных операций в реальном мире (часто на расстоянии), так и обеспечение новых возможностей человека.

Например, упомянутая фирма **Immersion Corp**. ставит целью достижение технологии, при которой хирург или врач манипулирует не инструментами и пациентом, а работает с виртуальной моделью человека, используя для этого устройства с обратной связью; манипулируя объектами в виртуальном мире, врач тем не менее чувствует человека и его ткани. При этом всю ‘грязную’ работу делает робот, управляемый компьютером посредством устройства с обратной связью, которым, в свою очередь, управляет врач. Для медицины подобные устройства должны обеспечивать как можно более точные и реалистичные ощущения.

Подобные технологии имеют перспективное будущее. Например, проведение хирургических операций - хирург ‘погружается’ в VR, в которой он видит пациента в объеме и насквозь посредством изображения, получаемого с помощью сканеров (лазерные сканеры, ядерные томографы) и преобразованного в виртуальную модель с необходимой точностью. Хирург сможет увеличить изображение любой части тела и работать, не боясь случайного дрожания руки со скальпелем, при использовании микроробототов возможно провести операцию прямо внутри человека (например, внутри сердца).

Для достижения подобных близких к фантастике в настоящее время возможностей придется количественно и качественно развивать возможности виртуальной реальности. Количественные улучшения будут сводиться к увеличению графического разрешения, повышению качества звука и эргономичности конструкции.

Из качественных можно прогнозировать отказ от внешней аппаратуры (минидисплеев, шлемов, наушников и т.д.) с переходом на принципиально другой уровень формирования ощущений. Микролазер, посылающий импульсы определенной частоты прямо на нерв, сможет заменить ощущения внешнего мира искусственно сформированными.

Второе важное направление - интеграция и возможность сосуществования нескольких VR-аналогов физических лиц в одном виртуальном мире при обеспечении всего присущего человеку спектра ощущений. Это открывает широчайшие возможности для ‘виртуального общения’, решения сложных научно-технических задач, развлечений. Уже сейчас на базе сети InterNet идет работа над несколькими такими проектами.

Не стоит забывать, что мн**и**мая (виртуальная) реальность может стать наркотиком XXI века, который поработ**и**т людей быстрее, чем все дон**ы**не известные - ведь мир, представленный VR, обычно намного привлекательнее того, что в действительности окружает человека.

8.1.2. **Операционные системы, ориентированные на поддержку мультимедиа**

Мультимедиа является относительно новой отраслью компьютерных технологий, большинство создававшихся в конце 70-х – начале 80-х г.г. операционных систем не было рассчитано на специфические требования, свойственные мультимедиа-технологиям. Традиционные ОС оптимизировались в направлении повышения гибкости управления ресурсами (в том числе обеспечения мультизадачного и/или многопользовательского режимов), разработчики их архитектуры не могли представить в то время, с какими объемами мультимедиа-информации придется работать начиная с конца XX века.

В основу нового поколения специализированных ОС легла концепция **Media OS** - совокупность требований к операционной системе, предназначенной для работы с большими объемами цифровых данных. Описывающий концепцию Media OS официальный документ можно найти на сайте www.be.com/products/beos/mediaos.html.

Одним из основных свойств мультимедиа-ОС (ММОС) является встроенная поддержка симметричной многопроцессорной обработки - ОС изначально рассчитана на работу с несколькими процессорами одновременно - в отличие от большинства других операционных систем, в которых поддержка нескольких процессоров является лишь дополнительной функцией. При этом разработчикам приложений нет необходимости заботиться о распределении нагрузки между процессорами - это выполняет за них сама система. Помимо этого ММОС обязана обладать и другими качествами, свойственными современным ОС: *вытесняющая многозадачность* (позволяющая эффективно работать с несколькими приложениями одновременно), *многопотоковость* (возможность выполнения множества независимых процессов внутри одного приложения), *64-разрядная файловая система* (обеспечивающая улучшенные средства д**о**ступа к дискам большого объема), внутренняя *архитектура 'клиент-сервер'* и др. технологии, позволяющие увеличить производительность и эффективность работы ЭВМ (заметим, что большинство из указанных требований реализовано в современных неспециализированных ОС – например, в основанных на ядре NT версиях MS Windows).

К основанным на вышеприведенных концепциях ОС традиционно относится **BeOS**, подобным же образом рекламировалась **NextStep**; к ММОС

(с некоторым сомнением) можно отнести предельно упрощенные ОС т.н. мультимедиа-компьютеров (простейших ПЭВМ, ориентированных исключительно на работу в сети InterNet).

BeOS (компания *Be Inc.*, www.beincorporated.com) позиционируется как ‘операционная система для мультимедиа и InterNet’; первоначально разрабатывалась для компьютера **BeBox** (разработка аппаратной части не завершена), в 1996 г. к BeOS проявила интерес *Apple Computer*, в настоящее время доступна бесплатная версия этой ОС (на сайте free.be.com). Система весьма проста в установке и настройке, штатно обеспечивает все сетевые возможности (E-Mail, WEB, RealAudio, RealVideo), обладает оптимизированной параллельной архитектурой, поддерживающей работу с цифровым звуком, графикой и видео. Для BeOS разработан браузер **NetPositive**, ОС поддерживает одновременное использование нескольких рабочих областей - виртуальных экранов, каждый из которых может иметь свои собственные настройки (такие как разрешение, что позволяет избежать загромождения экрана окнами и в некоторых случаях существенно облегчает работу пользователя). BeOS изначально рассчитана на работу в режиме ‘двойной загрузки’ на одной машине с Windows или **MacOS** и поддерживает стандарт **POSIX** (поэтому написанные в соответствии с этим стандартом Unix-программы могут компилироваться под BeOS без или с незначительными изменениями).

Вместе с тем список поддерживаемых внешних устройств для BeOS невелик. Несколько поставщиков компьютеров (в том числе *AST Computers*, www.ast.com и *IDot.Computers*, www.idot.com) начали устанавливать BeOS на новые ЭВМ, компания *Microworkz.com* (www.microworkz.com) использует специализированную версию BeOS в своих недорогих устройствах **IToaster** - машинах с ограниченным набором функций, предназначенных для новичков в WEB. Компания *Be Inc*. пытается привлечь достоинствами своей ОС производителей других недорогих устройств (уровня телевизионных InterNet-приставок).

Одновременно компания *Be Inc*. рассчитывает на то, что распространение более быстрых InterNet-соединений подстегнет к принятию BeOS творческими личностями, создающие видео-, аудио- и анимационный контекст для WEB. Некоторые известные фирмы уже сейчас ориентируются на BeOS – известны вариант броузеров **Opera** (фирма *Opera Software*, www.opera.com) и **Netscape Communicator** (home.netscape.com), игры **Quake Arena** (фирма *ID Software*, www.idsoftware.com), видеодрайверов nVIDIA (*NVIDIA*, www.nvidia.com) и др. для BeOS.

История развития BeOS может повторить не слишком удачный проект ОС **NeXTStep** для компьютера **NeXT** (тем более, что начинали разрабатывать и BeOS и NeXTStep выходцы из *Apple Computer*); фактически мультимедийный компьютер NeXT выпускался с 1989 г. по 1993 г., но вследст-

вие несовместимости с иными моделями и дороговизн**ы** выпуск был прекращен, а фирма перепрофилирована в разработчика системного ПО.

Можно также говорить о применении мультимедиа в оформлении пользовательского интерфейса операционных систем и приемах его использования. Например, известный стандарт **GUI** (*Graphic User Interface*) является стандартом многопользовательского графического интерфейса Windows и многих других ОС (хотя и берет начало от разработки фирмы **Xerox** конца 70-х / начала 80-х г.г.). Устройство управления типа ‘мышь’ (разработка середины 60-х г.г. *Стемфордского исследовательского института* - **SRI**) является важнейшим элементом подобного пользовательского интерфейса.

Контрольные вопросы

1. Привести примеры наиболее перспективных направлений развития мультимедиа
2. В чем заключается идея об отказе от внешней аппаратуры мультимедиа и в чем может проявиться опасность подобного подхода ?
3. Какова концепция Media OS и в каких системах (и насколько п**о**лно) она реализована ?

ЗАКЛЮЧЕНИЕ

Мультимедиа является бурно развивающейся областью деятельности, фактически приближающей компьютерные технологии к человеку. Несмотря на большое количество известных технологий работы со звуком,

изображениями и видео ежемесячно разрабатываются новые, при этом существующие методы излишне техницированы (что как раз и говорит о ‘юности’ этой области).

Даже в обозримом будущем развитие мультимедиа-технологий предлагает близкие к фантастическим перспективы увеличение технологической мощности человечества. Однако приближающееся неизбежное вмешательство мультимедиа непосредственно в психофизическую сферу человека обоснованно представляется опасным и непредсказуемым по последствиям для человеческого сообщества.

СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ

1. Порев В.Н. Компьютерная графика. -С.Птб.: BHV, 2002. -432 с.
2. Загуменнов А.П. Компьютерная обработка звука. -М.: ДМК, 2000. -384 с.
3. Рудометов Е., Рудометов В. Аппаратные средства и мультимедиа (справочник). -С.Птб.: Питер, 2000. -416 с.
4. Гук М. Аппаратные интерфейсы ПК. -С.Птб.: Питер, 2002. -528 с.
5. Свиридов Ю., Тюкачев Н. Delphi 5 – создание мультимедийных приложений. -С.Птб.: BHV, 2001. -400 с.
6. Фролов А.В., Фролов Г.В. Мультимедиа для Windows (руководство для программиста, том 15). - М.: Диалог-МИФИ, 1995. - 284 с.
7. Авраамова О.Д. Язык VRML (практическое руководство). –М.: Диалог-МИФИ, 2000. - 288 с.

Учебное издание.

Баканов Валерий Михайлович

Программирование
мультимедиа-систем

Учебное пособие.

Подписано в печать __.__.2004 г.
Формат 60 × 84 1/16.
Объем 7,5 п.л. Тираж 100 экз. Заказ ___

Отпечатано в типографии Московской государственной академии
приборостроения и информатики.
107846, Москва, ул.Стромынка,20.