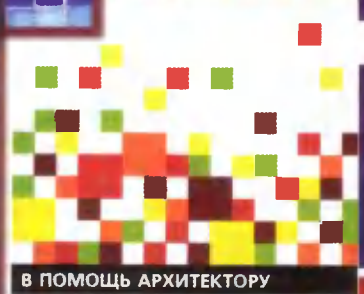
http://radio-mir.com

индексы: 48925, 45995

радиомир

6•2004

Ваш компьютер



В ПОМОЩЬ АРХИТЕКТОРУ

ИНТЕРЬЕР ЗА 5 МИНУТ

3D Home Architect Professional 5.0 Русская и Английс
Data Becker 3D Apartment and Condo Designer 3.0
Home Plan Pro 4.4.48
STRUCTURAL CONCRETE SOFTWARE FLOOR 1.01
Статьи

Компьютеры для дома и офиса

от компании СитиПринт

Интернет и правильно выбранный компьютер раскроют перед вашими детьми горизонты новых знаний.

Позаботьтесь об образовании ваших детей.

Предоставьте им возможность искать новую информацию в сети, работать с обучающим ПО и писать рефераты.

Приобретите компьютеры от компании "СитиПринт" на базе процессора Intel® Pentium® 4 с технологией HT уже сегодня.

СитиПринт
КОМПЬЮТЕРЫ И ОРГТЕХНИКА

220006, г.Минск, ул.Полевая 28-7А. Тел./факс (017) 2850134.
www.citiprint.by

Intel, логотип Intel Inside, Pentium и Celeron являются зарегистрированными товарными знаками компании Intel Corporation

Учредитель и издатель:
ООО "Радиомир Пресс"
Свидетельство о регистрации
ПИ №77-9841 от 17.09.2001

Главный редактор
Ольга СТРЫЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

Редколлегия:
Сергей ДРОЗДОВСКИЙ,
Алексей КОНДРАШОВ,
Анатолий КОРБИТ,
Владимир КУЦЕНКО,
Елена ЛЕВИТМАН,
Николай МЕДВЕДЬ,
Геннадий ПЕЧЕНЬ,
Геннадий ШУЛЬГИН

Контактные телефоны:
в Москве (095) 105-99-89,
в Минске (017) 249-41-47

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

Адреса для писем:
119454, РФ, г.Москва, а/я 37,
факс (095) 432-62-04;
220095, РБ, г.Минск-95, а/я 199

E-mail: ri@radiopage.by
rm@radio-mir.com
WWW: http://radio-mir.com

Наши платежные реквизиты:
получатель: ООО "Радиомир Пресс",
ИНН 7729404741, КПП 772901001,
р/с 40702810900010000084
в ООО КБ "Агропромкредит"
ф-л Центральный, г.Москва,
корр. счет 30101810500000000109
БИК 044525109
Адрес банка: 125315, г.Москва,
Ленинградский пр., дом 76, корп. 4

Материалы для публикации принимаются в рукописном, печатном и электронном вариантах

Требования к графическим матвриалам рекламного характера в электронном виде: CorelDRAW до 10.0, все шрифты в кривых; bitmaps 300 dpi; TIFF 300 dpi; CMYK. Приложить печатную копию.

За *достоверность рекламной и другой публикуемой информации несут ответственность рекламодатели и авторы. Мнение редакции не всегда совпадает с мнениями авторов*

Адрес редакции: 119454, РФ, г.Москва, ул.Коштоянца, 6 — 233, тел. (095) 105-99-89

Подписано к печати 23.04.2004 г.
Формат 60 x 84 1/8.
Печать офсетная. 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ № 1377.
© ООО "Радиомир Пресс""

радиомир
Ваш компьютер

ЕЖЕМЕСЯЧНЫЙ МАССОВЫЙ ЖУРНАЛ
С 1995 г. по 6/2001 г. издавался под названием "Радиолюбитель. Ваш компьютер"

№6/июнь/2004

ЧИТАЙТЕ В НОМЕРЕ

Полные листинги некоторых программ расположены по адресу
http://radio-mir.com

ПРЕСС ДЛЯ ФАЙЛОВ

В.Куц,
E-mail:victor_kootz@mail.ru

Хорошо известно правило, бытующее в компьютерном мире: емкости жесткого диска много никогда не бывает. Действительно, с этим трудно не согласиться — каким бы огромным ни казался винчестер при покупке, не пройдет и полгода, как вы будете удивляться, куда же делись те десятки гигабайтов, пустовавшие на совсем еще недавно казавшемся бездонным диске? Сколько всякой, нужной и не очень, информации, а то и просто обычного хлама скапливается в мириадах папок и файлов. Далеко не все из этого нужно, но, как обычно, выбрасывать жалко — а вдруг что-нибудь когда-нибудь пригодится? Значит, необходимо время от времени производить "складирование" всего этого добра в какое-нибудь хранилище, в архив. И тут-то нам не обойтись без помощи программ-упаковщиков (или архиваторов) — пожалуй, самого популярного класса утилит, широко используемых с самых первых дней массового применения персональных компьютеров. Кроме задач резервного копирования редко используемой информации, архиваторы широко применяются везде, где стоит задача размещения как можно большего объема данных на носителе ограниченной емкости, например, на съемных носителях, таких как дискеты, компакт-диски и пр., либо при пересылке файлов по каналам Интернета или по электронной почте. Эти программы позволяют помещать копии файлов в архив и извлекать файлы из архива, просматривать оглавление архива и тестировать его на предмет отсутствия повреждений, удалять файлы, находящиеся в архиве, и обновлять их, устанавливать пароль при извлечении файлов из архива, и многое другое. Разные архиваторы отличаются форматом архивных файлов, скоростью работы, степенью сжатия, набором дополнительных сервисных функций, удобством пользования, размером программы. Программ-архиваторов существует великое множество, но прежде чем рассматривать их, не помешает немного задержаться на особенностях наиболее популярных сегодня форматов архивов.

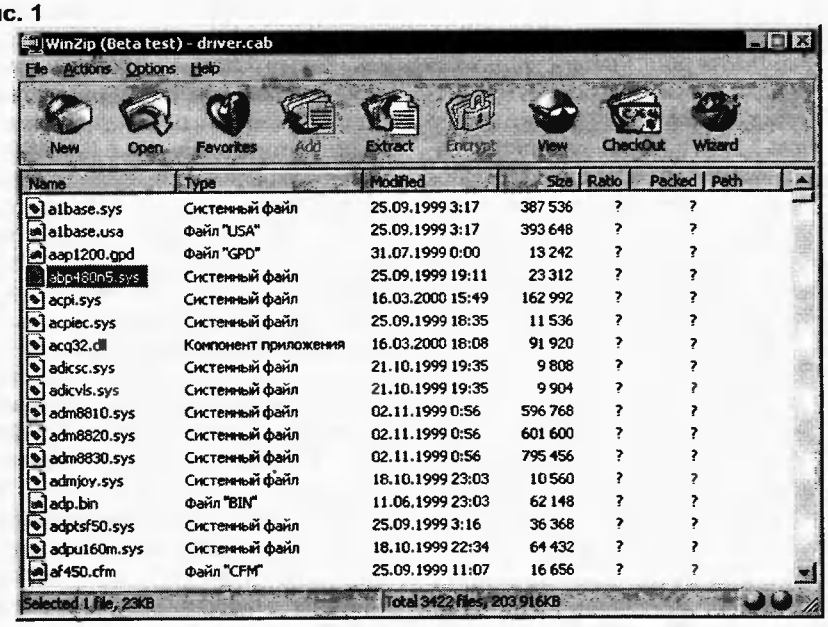
Еще со времен DOS одним из самых популярных и распространенных архивных форматов стал **ZIP**, основанный на алгоритмах сжатия, предложенных в 80-х годах прошлого столетия израильскими математиками Лемпелем и Зивом. Он отличается приемлемой степенью сжатия информации и достаточно высоким быстродействием. Сегодня ZIP является стандартом де-факто в Интернете, и его в обязательном порядке поддерживают практически все программы-архиваторы, претендующие хоть на мало-мальскую популярность. Но, в то же время, в связи со значительным ростом объемов архивов (особенно это касается мультимедийных приложений), вполне вероятна ситуация, когда формат ZIP уже нельзя будет использовать, поскольку размеры ZIP-файлов не могут превышать предел в 4 Гб (а емкость, к примеру, самого простого, одностороннего однослойного DVD-диска составляет 4,7 Гб). Поэтому в последнее время на ведущие позиции выдвинулся другой формат архивных файлов — **RAR**, всерьез претендующий на признание в качестве общепризнанного стандарта. Он разработан российским программистом Евгением Рошалем и позволяет получить размер сжатого файла гораздо меньший, чем ZIP, но ценой этому является более продолжительный процесс обработки архива. В целом формат RAR значительно лучше других оптимизирован для решения сложных задач с использованием большого количества файлов и гигабайтных дисковых пространств. Вообще говоря, эти два формата, один — быстрый, другой — обеспечивающий высокую степень сжатия файлов, прекрасно дополняют друг друга, и их вполне достаточно для реального использования при решении подавляющего большинства задач. Тем не менее, этими форматами список существующих далеко не исчерпывается. Совсем недавно сильную конкуренцию лидерам составлял формат **ARJ**, до сих пор отличающийся, пожалуй, наиболее широкими возможностями по настройке. Но с широким распространением Интернета этот формат стал потихоньку забываться, и теперь о нем вспоминают лишь те, кто испытывает ностальгию о временах командной строки DOS, когда ARJ был, по крайней мере на территории Советского Союза, самым популярным. Не мог не отметиться в славном деле создания новых форматов архивирования и всеми нами любимый "мелкомягкий" гигант. Формат **CAB** применяется в продуктах Microsoft как стандартный для упаковки файлов, причем его алгоритм, нигде не опубликованный и хранимый фирмой за семью печатями, представляет собой достаточно совершенный продукт, имеющий высокий коэффициент сжатия. Кроме этих форматов, гораздо реже можно встретить еще некоторые устаревшие, например, **HA**, применяющийся по большей части для сжатия текстовых файлов, или **LZH** — совсем древний формат, предшественник ARJ. В системах на базе Unix и ее самой популярной разновидности Linux, наибольшее распространение получили форматы **GZIP**, **TAR** и их всевозможные комбинации. И это только самые известные форматы архиваторов, которые, как говорится, у всех на слуху. А всего существует до сотни самых различных форматов, большая часть которых никому не известна, но это не особо большая потеря. Хотя популярных форматов никак не более десятка, но, тем не менее, достаточно остро стоит вопрос их совместимости, ведь мало кому нужен архиватор, пусть и великолепно сжимающий информацию, но чьи архивы никакая другая программа не сумеет прочесть. Поэтому большинство современных программ-архиваторов не зацикливаются на своих собственных форматах, и одним из их важнейших потребительских свойств становится максимально широкий охват всевозможных типов архивов. Во многих случаях удачным решением проблемы совместимости архивов различных типов является создание архивов в виде самораспаковывающихся программ (EXE-файлов), в состав которых входят все необходимые механизмы для извлечения информации из архива. Таким образом, отпадает необходимость иметь на компьютере соответствующую программу-распаковщик архива.

Итак, переходим к рассмотрению свежих версий наиболее интересных, на мой взгляд, программ-архиваторов.

WinZip 9.0 Beta 2

Это одна из самых популярных в Интернете программ, за годы своего существования собравшая значительное число наград самых различных компьютерных изданий **(рис.1)**. Сам ZIP-алгоритм свободно используется в десятках, если не в сотнях программ, тем не менее, для очень многих пользователей Windows именно WinZip является стандартной программой

Рис. 1



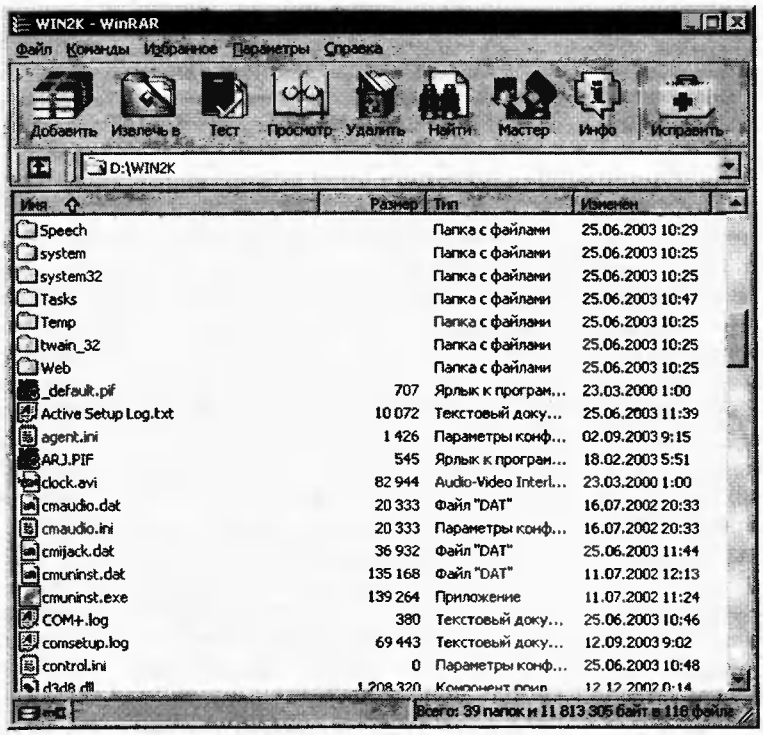
для работы с архивами. Встроенные средства обработки архивов WinZip позволяют упаковывать, просматривать и извлекать файлы из широко распространенных форматов архивов, таких как ZIP, CAB, Microsoft Compress, GZIP, TAR, UUencode, XXencode, BinHex, и MIME. Кроме того, установив дополнительно архиваторы типа ARC, LHA и ARJ и прописав в свойствах WinZIP путь к ним, можно полноценно работать с архивами и этих форматов. В данной версии архиватора появилась поддержка нового, 64-битного расширения формата ZIP, который способен преодолеть ограничения его оригинальной версии на максимальный размер архива (4 Гб). Все операции с архивами можно выполнять с помощью перетаскивания мышкой, длинные имена объектов (в том числе и кириллические) обрабатываются и отображаются правильно. WinZIP полностью интегрируется в интерфейс Windows (добавляются его основные команды в контекстные меню "Проводника" и "Моего Компьютера"), а новички, делающие первые шаги на нелегком компьютерном поприще, могут воспользоваться помощью "Мастера", который проведет их по всем этапам не очень сложной, но достаточно непонятной для "чайников" процедуры работы с архивами. Для более опытных гораздо проще будет работать с "классической" формой интерфейса архиватора, обеспечивающей быстрый и удобный доступ ко всем его функциям, среди которых можно выделить шифрование файлов по алгоритму AES (длина ключа может достигать 256 битов), проверку содержимого архивов внешним антивирусом, создание многотомных архивов. Утилита WinZip Self-Extractor, предназначенная для создания самораспаковывающихся EXE-файлов и ранее существовавшая самостоятельно, теперь входит в состав программы. Кроме того, предусмотрено создание виртуальной папки избранных архивов (Favorites), что позволяет быстрее находить архивы, независимо от их местоположения. К сожалению, в WinZIP не очень удобно реализована процедура создания нового архива, когда вначале нужно создать сам архив, и только после этого можно добавлять в него необходимые файлы.

Интерфейс WinZip — на английском языке, но в Интернете, не особо напрягаясь, можно найти большое количество русификаторов практически для всех его версий. Программа успешно работает под управлением всех современных разновидностей операционных систем Windows 95/98/Me/XP/NT 4.0/2000. А вот такие устаревшие как Windows 3.x или Windows NT 3.1/3.5, уже не поддерживаются (какая жалость!). В отличие от более ранних версий, отличавшихся компактностью, размеры программы WinZip 9.0 значительно выросли. Так, EXE-файл с дистрибутивом WinZip имеет размер около 2,2 Мб, а после инсталляции объем файлов программы уже более 5 Мб. Распространяется программа как shareware со сроком бесплатного использования 21 день. Стоимость последующей регистрации однопользовательской лицензии составляет 29 USD.

WinRar 3.20

По своей популярности архиватор WinRAR (**рис.2**), без сомнения, находится на первом месте в России и на одном из первых во всем остальном мире, стремительно оттесняя заслуженный WinZip с передовых позиций. Существует несколько версий RAR для разных операционных систем, в частности, RAR для DOS, OS/2, Windows, MacOS и почти всех разновидностей Unix, включая такие популярные ее варианты как Linux и BSD. Версия WinRAR для Windows имеет две разновидности: одна, для облегчения работы, имеет графический интерфейс пользователя (GUI), вторая — консольная, использующая командную строку для ввода команд в текстовом режиме. Программа полностью поддерживает работу с форматами ZIP и RAR, и ограниченно, позволяя только распаковы-

Рис. 2



вать и просматривать архивы форматов CAB, ARJ, LZH, TAR, GZ, ACE, UUE, BZ2, JAR и ISO. Пополнять архивы и извлекать из них объекты можно с помощью удобной технологии перетаскивания (drag&drop). Для просмотра упакованных файлов в программе имеется встроенный модуль визуализации, хотя можно подключать и внешний просмотрщик. Для обеспечения высокой степени сжатия архивируемой информации в WinRAR используется ряд оригинальных алгоритмов упаковки данных с поддержкой мультимедиа-сжатия, показывающих хорошие результаты при сжатии отдельных аудио- и графических форматов, а также алгоритм создания непрерывных (solid) архивов, оптимизирующий процесс упаковки большого количества небольших однотипных файлов. Кроме того, WinRAR может создавать самораспаковывающиеся (SFX) и многотомные архивы, восстанавливать поврежденные архивы, шифровать их по алгоритму AES-128, добавлять различные комментарии, протоколировать ошибки и т.д. Среди прочих сервисных функций, порой существенно облегчающих повседневную работу, могут оказаться очень полезными просмотр текстовых файлов из архивов или работа с содержимым упакованного файла как с обычным каталогом. Благодаря всем этим качествам, WinRAR пользуется большой популярностью. Однако следует иметь в виду, что, начиная с версии WinRAR 3.0, был серьезно изменен алгоритм создания архива. Это, с одной стороны, существенно повысило ее производительность, но с другой — появились проблемы с совместимостью форматов архивов, созданных разными версиями программы. Так, архивы, созданные WinRAR версии 3.0 и выше, не могут распаковываться более старыми версиями этого архиватора, а они до сих пор еще встречаются во многих старых программах и файловых оболочках.

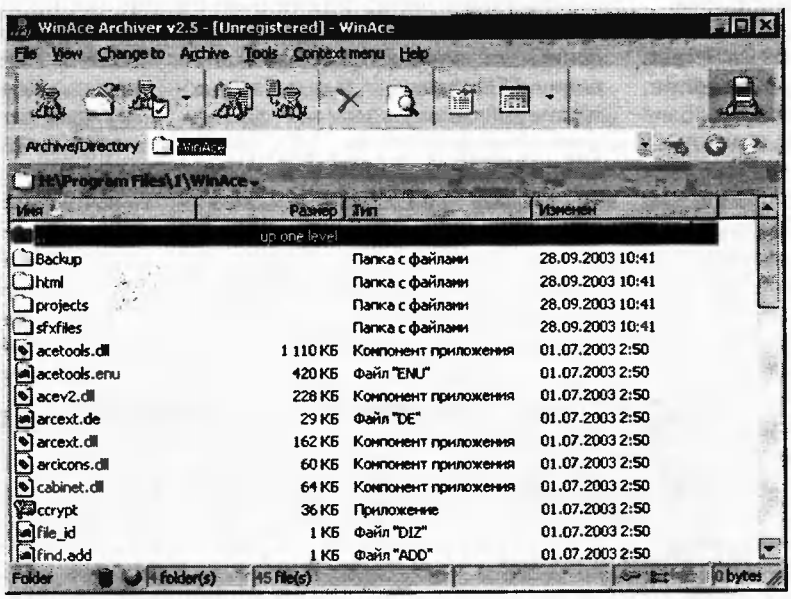
WinRAR имеет как русскую, так и английскую версии, причем внешний вид их графической оболочки очень напоминает WinZip. Впрочем, это в равной мере относится ко всем программам обзора и свидетельствует только о том, что в этой области все разработчики достигли определенного предела, близкого к оптимуму. Дистрибутив WinRAR 3.20 имеет объем около 1 Мб, а после инсталляции пакет занимает чуть меньше 3 Мб. Срок бесплатного использования программы WinRAR ее автор, Евгений Рошаль определил в целых 40 дней, одна же лицензия обойдется в "кругленькую сумму" в 35 USD.

WinAce 2.5

Один из самых агрессивных новичков, архиватор WinAce, без всякого почтения перед сединами ветеранов, активно оттесняет многих из них в борьбе за место под солнцем (**рис.3**). Программа может работать со многими популярными форматами архивов — создание и распаковка, кроме своего штатного формата ACE, поддерживаются еще и для ZIP, LHA, CAB и JAR; а вот архивы RAR (версий 2.0 и 3.0), ARC, ARJ, GZIP, TAR, ZOO, ISO и BZIP2 могут только распаковываться. Как и положено продвинутому архиватору, WinAce подерживает все необходимые для удобной и быстрой архивации файлов функции: он способен создавать многотомные (только для ACE, ZIP и CAB) и самораспаковывающиеся (ACE и ZIP) архивы; тестировать их и восстанавливать поврежденные; создавать и редактировать комментарии к архивам (не только в текстовом, но и в HTML-формате), поддерживает выполнение операций из командной строки. По умолчанию файлы сжимаются с нормальным качеством в т.н. solid-режиме (группа небольших файлов обрабатывается как один). Забавно, но для некоторых типов файлов (например, для изображений) этот режим наиболее приемлем, а использование максимальной степени сжатия лишь увеличивает размер итогового архива. Для обеспечения конфиденциальности архивированной информации предусматривается ее шифрование, а также блокировка модификации ACE-архивов. Возможен быстрый просмотр и редактирование графических файлов, текстовых ASCII-, HTML- и Word-документов с помощью внутреннего или внешнего просмотрщика, проверка файлов на наличие вирусов. Но все эти функции в той или иной мере встречаются и в других архиваторах, а WinAce имеет еще и некоторые уникальные возможности, и среди них:

- оптимизация (конвертирование в формат ACE 2.0) архивов устаревших типов;
- определение типов архивных файлов по их сигнатуре;
- запуск или открытие файлов из архивов, в том числе и инсталляция программ.

Рис. 3



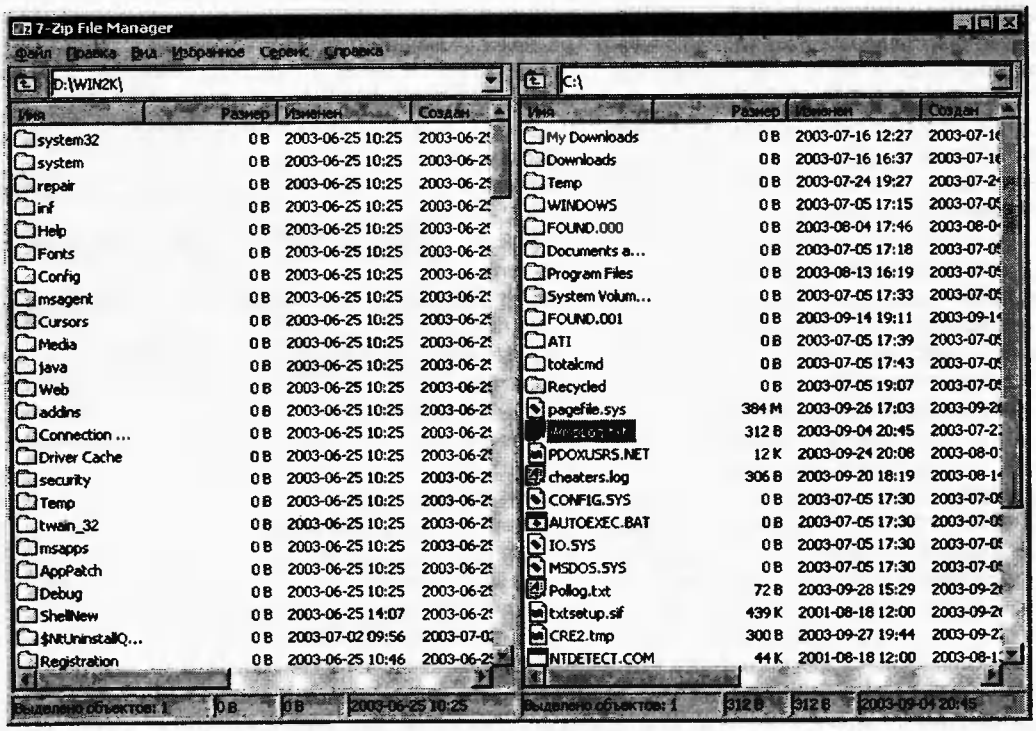
Однако за такие широкие возможности программы надо платить — она является "чемпионом" обзора как по размеру дистрибутива (3,5 Мб), так и по месту, занимаемому после установки (7,6 Мб). Что касается интерфейса WinAce, то он имеет отдельные черты файлового менеджера, полностью интегрируется в Windows (контекстные меню, закладки "Свойств"), и доступна возможность переключения языка (в оригинале — только английский и немецкий, но, поискав в Сети, можно найти и файлы поддержки русского языка). Программа работает со всеми версиями Windows. Как и большинство программ в этом обзоре, WinAce — продукт shareware, и по истечении 30-дневного оценочного периода разработчики мечтают получить от вас 29 USD за стандартную версию, в которой отсутствует режим работы из командной строки, и 39 USD — за полную WinAce plus.

7-Zip 3.10

Еще один самобытный архиватор, вышедший из-под пера отечественного программиста, отличается от всех остальных программ обзора тем, что он, помимо того, что совершенно бес-

платен, еще и отличается высокой степенью сжатия информации (**рис.4**). Выбор форматов, с которыми работает архиватор (ведь это, фактически, "визитная карточка" подобного рода программ), не столь богатый, как у конкурентов: помимо своего "фирменного" формата 7z, программа полностью поддерживает форматы ZIP, GZIP, TAR и BZIP2. Архивы форматов CAB, RAR (в том числе архивы 3-й версии), ARJ, CPIO, SPLIT и RPM и DEB (последние два — фирменные форматы упаковки бинарных файлов для RedHat Linux и Debian Linux соответственно) поддерживаются только на уровне просмотра и распаковки содержимого. А вот многотомные и непрерывные RAR-архивы пока еще не поддерживаются.

Рис. 4



Для форматов ZIP и GZIP, благодаря внутренней оптимизации, обеспечивается очень высокая степень сжатия. Наибольшая же степень сжатия достигается при использовании формата 7z, который, хотя пока еще довольно экзотичный, но по коэффициенту компрессии способен составить конкуренцию самому RAR'у, а в отдельных случаях даже может превосходить его (правда, во время тестирования такого замечено не было). Однако за хорошую компрессию приходится платить крайне низкой скоростью работы. Кроме того, данный формат, не накладывая никаких ограничений на размеры файлов, позволяет шифровать архивы по методу AES с ключом 256 битов, хранить имена файлов в формате Unicode, поддерживает самораспаковывающиеся и непрерывные solid-архивы.

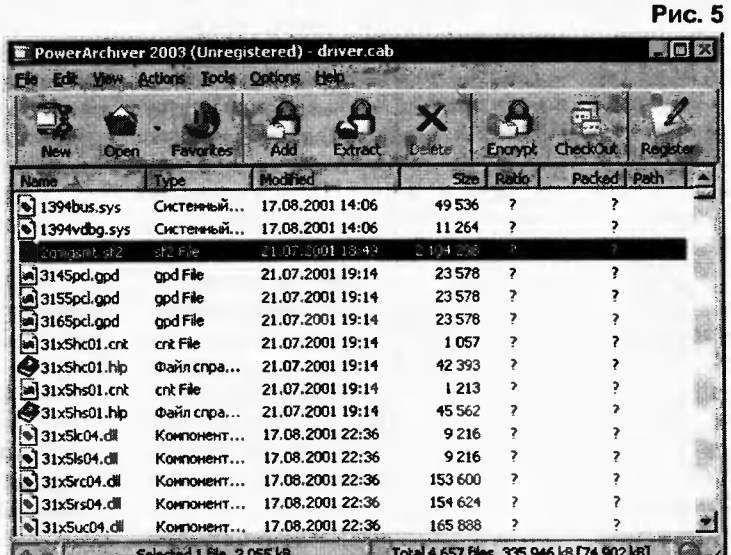
Программа 7-Zip достаточно хорошо интегрируется в "Проводник" Windows, добавляя в контекстное меню пункты, позволяющие управлять созданием архивов, их тестированием или распаковкой файлов поддерживаемых форматов, а также произвести некоторые другие операции. Графическая оболочка архиватора, простая до примитивизма, является своеобразным файловым менеджером, и может работать в одно- или двухоконном режимах. Основные операции в ней можно производить при помощи комбинации "горячих" клавиш или через меню — обычное текстовое или контекстное. Что радует в 7-Zip, так это ее многоязычность — интерфейс программы может быть на одном из более чем 40 языков, хотя вся справочная информация — только на английском. Помимо версии с графическим интерфейсом, существуют также версии архиватора с поддержкой командной строки, которые могут быть использованы как плагины для работы с архивами в менеджере файлов FAR. Программа успешно работает со всеми версиями Windows.

Дистрибутив 7-Zip 3.10 имеет объем менее 1 Мб, а после инсталляции пакет занимает уже около 4 Мб. Хотя архиватор распространяется бесплатно, его автор, Игорь Павлов, принимает добровольные пожертвования на развитие своей программы в размере 20 USD (от россиян принимаются пожертвования в размере 100 рублей).

PowerArchiver 2003 8.60

Еще один достаточно мощный архиватор PowerArchiver (**рис.5**), ранее известный как EasyZip, отличается от всех вышерассмотренных тем, что не имеет своего "фирменного" формата архивирования, а работает "по полной программе" с форматами ZIP, CAB, LHA, LZH, TAR, TAR.GZ, TAR.BZ2 и BH (Black Hole), а также позволяет просматривать и извлекать файлы из архивов RAR (включая 3-ю версию), ACE (включая 2-ю версию), ARJ, ARC, GZ, BZIP2, ZOO. Возможна также работа с форматами XXE и UUE, используемыми, по большей части, для кодирования двоичных файлов при их пересылке по электронной почте. Уже привычный стандартный набор функциональных возможностей архиватора включает создание многотомных и

Рис. 5



самораспаковывающихся архивов, разбивку архивов на тома, автоматическую инсталляцию из архивов (если в них включен файл setup.exe), поддержку технологии drag&drop, полную интеграцию с "Проводником" Windows. Кроме стандартного интерфейса, к сожалению, только на английском языке, программе можно придать и внешнее сходство с "Проводником" — такие же два окна с древовидной структурой папок-архивов в левом окне и их содержимым — в правом. Есть у PowerArchiver и такой символ "немереной крутизны" как возможность менять шкурки-скины, имитирующие облик некоторых популярных программ или стиль различных ОС (причем в дистрибутиве программы содержится всего 9 скинов, но на официальном сайте программы их гораздо больше). К числу вспомогательных, но порой очень полезных возможностей, можно отнести встроенную утилиту для просмотра содержимого графических и текстовых файлов форматов TXT, RTF, BMP, WMF, EMF, JPG (JPEG), GIF и ICO.

Программа PowerArchiver прекрасно себя чувствует во всех операционных системах Windows — от старушки 95-й и вплоть до XP. Размер ее дистрибутива — 2,5 Мб, а после установки она занимает на диске 6 Мб. Как и многие другие архиваторы, PowerArchiver имеет 30-дневное ограничение на период бесплатного использования.

Cabinet Manager 2003 4.1

Архиватор Cabinet Manager (**рис.6**), как можно понять из его названия, ориентирован на использование формата CAB, разработанного фирмой Microsoft в первую очередь для хранения дистрибутивов своих программ, отличающихся, как хорошо известно, весьма солидными габаритами. То есть главное требование к формату CAB — высокая степень сжатия. Плюс к этому не менее важна и возможность создания самораспаковывающихся архивов — ведь во многих случаях изделия Microsoft ставятся на абсолютно "голые" машины, где никакими архиваторами и не пахнет. Как и во многих других случаях, Microsoft удалось, и довольно успешно, воплотить свои задумки в жизнь, в чем мы сможем убедиться на примере одной из немногих утилит стороннего производителя, использующего формат CAB в качестве основного. Девизом Cabinet Manager можно с полным правом считать слово "сам". Считайте: основная задача программы — создание самораспаковывающихся архивов CAB, кроме того, возможно создание самоустанавливающихся после распаковки приложений, а также самоидентифицирующихся (во словечко-то, а?) и даже саморасшифровывающихся (использующих шифрование по довольно устаревшим алгоритмам RSA RC2, RC4 и DES) архивов. В общем, "сам, все сам" — совсем как бойскаут. А вот англоязычный интерфейс программной оболочки Cabinet Manager абсолютно ничем не выделяется по сравнению с другими программами обзора ни в лучшую, ни в худшую стороны — так, середнячок. Да, Cabinet Manager может интегрироваться в "Проводник" Windows, при установке добавляет в контекстное меню свою команду "Add to Cabinet", а для файлов с расширением exe еще и команду Inspect SFX, но это могут и все остальные, без исключений, архиваторы. Cabinet Manager работает также с файлами некоторых других форматов, в частности, ZIP и JAR. Архитектура программы позволяет дополнять ее с помощью плагинов новыми алгоритмами архивирования.

Размер и дистрибутива, и установленной программы не сильно отличается от 1 Мб, она самая компактная в обзоре. Условия использования Cabinet Manager довольно сносные — бесплатно в течение 40 дней и после этого регистрация за 19 USD. Многие аналогичные программы стоят гораздо дороже.

Рис. 6

Архиватор	Тип архива	Время сжатия, мин:сек	Время распаковки, мин:сек	Размер архива, битов
WinZip	ZIP	0:29	0:09	49775519
WinRAR	RAR	2:12	0:12	39415315
WinAce	ACE	2:41	0:15	44677629
7-Zip	7Z	3:50	0:18	43029483
PowerArchiver 2003	ZIP	0:46	0:14	49525601
PowerArchiver 2003	BH	0:56	0:13	49514794
Cabinet Manager 2003	CAB	2:07	0:17	45457960

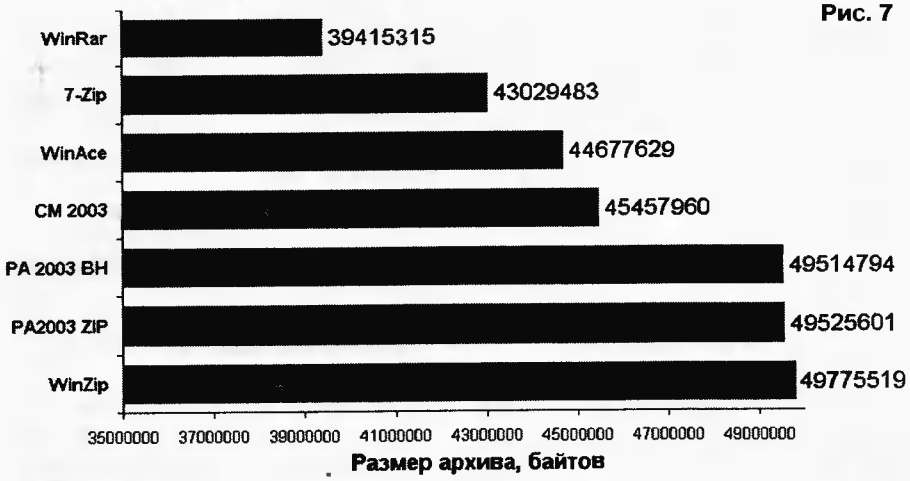
Тестирование

Для проверки возможностей рассмотренных архиваторов в условиях, максимально приближенных к "боевым", была создана подборка тестовых файлов общим размером 74363466 битов, включающая в себя 11 хорошо сжимаемых текстовых файлов общим размером 15 Мб, один исполняемый EXE-файл (13 Мб) и 2 библиотеки *.DLL (18 Мб). Для того чтобы проверяемым архиваторам "служба медом не казалась", туда же были добавлены практически несжимаемые 6 музыкальных MP3-файлов (25 Мб). Вот с такой "гремучей смесью" и пришлось мучиться нашим подопечным, последовательно ее упаковывая и распаковывая. Испытания проводились на не самом мощном, по нынешним меркам, но вполне типичном домашнем ПК с процессором AMD Athlon 1000 МГц и с 256 Мб SDRAM. Все это "добро" работало под управлением русской версии операционной системы Windows 2000SP4. Каждая из испытываемых программ создавала архив в своем штатном формате с установками по умолчанию, и лишь только PowerArchiver, исключительно любопытства ради, кроме основного ZIP'а, создал архив еще и в достаточно экзотическом формате Black Hole.

Результаты полученных измерений сведены в таблицу.

Если сравнивать размеры полученных архивов (**рис.7**), то все результаты делятся на две группы. Первая включает в себя самые "продвинутые" архиваторы WinRAR, 7-Zip, WinAce и Cabinet Manager, для получения наибольшей степени сжатия использующие самые современные алгоритмы. Тем не менее, эта группа неоднородна, и безоговорочное лидерство здесь принадлежит WinRAR. Во вторую группу аутсайдеров попали архиваторы, базирующиеся на уже устаревающем формате ZIP, и примкнувший к ним

Рис. 7

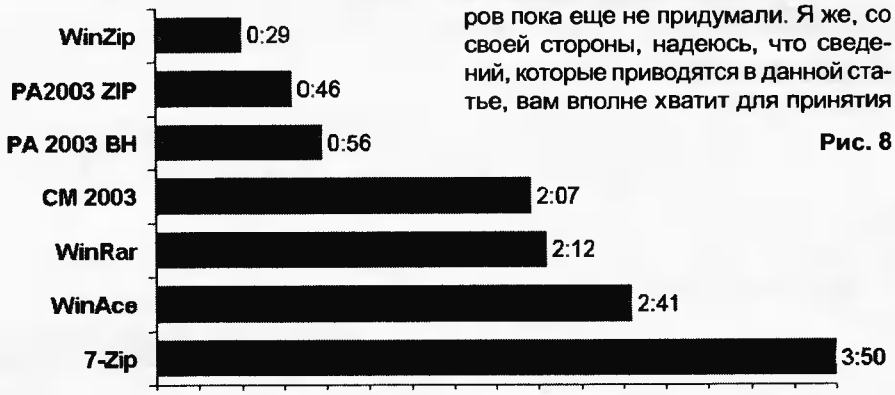


Black Hole. Что же, неужели заслуженный ветеран ZIP до такой степени ни на что не годен? Верится с трудом.

И в этом нас убеждает второй график (**рис.8**), на котором отображено время архивации тестового файла. Здесь ситуация повторяется с точностью до наоборот. Формат ZIP наиболее предпочтителен для самых нетерпеливых, а вот чемпионы по степени сжатия плетутся в самом конце. Причем WinRAR и в своей подгруппе показал себя очень достойно, чего не скажешь о молодом и рьяном его конкуренте 7-Zip. А вот претендующий на многое WinAce, впрочем, как и Cabinet Manager, со всех точек зрения оказались середнячками, хотя и вполне добротными.

График времени распаковки архивов не приводится по причине практического равенства для всех рассматриваемых программ времени выполнения этой задачи, ведь согласитесь, что 9 сек разницы между самым быстрым и самым медленным архиваторами при распаковке 70-мегабайтного архива в реальных условиях совсем не заметны.

Рис. 8



Заключение

Вот мы и добрались до подведения итогов. Что, думаете, сейчас скажу вам, какой архиватор хороший, какой плохой, какой нужно обязательно установить себе, а какой ни в коем случае? Ошибаетесь. В этом обзоре плохих архиваторов нет вообще, хотя каюсь, все хорошие охватить тоже не сумел. Да это, наверное, просто невозможно — уж очень их много. Так что вам придется думать самим, что для вас необходимо в первую очередь — высокая скорость работы, или максимальная степень сжатия, или может быть, симпатичный интерфейс со сменными "шкурками", или даже спокойная совесть при честном использовании бесплатного 7-Zip? При любом выборе чем-то придется пожертвовать, так как идеальных архиваторов пока еще не придумали. Я же, со своей стороны, надеюсь, что сведений, которые приводятся в данной статье, вам вполне хватит для принятия обоснованного решения. Замечу только, что какой бы вы архиватор ни выбрали, он в обязательном порядке должен поддерживать, как минимум, просмотр и распаковку самых распространенных типов форматов архивирования, а сегодня это, без сомнения, ZIP и RAR.

А.ГРИНЧУК,
E-mail: gav@nihe.niks.by

Анимация с Macromedia Flash MX

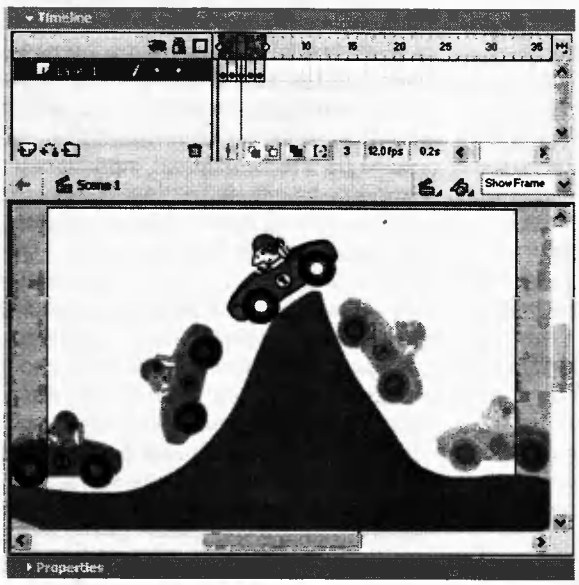
Экспорт и импорт

Любая хорошая программа (а Flash несомненно относится к их числу) не ограничивается работой с одним-единственным типом данных. И как всегда, это приводит не только к значительному увеличению возможностей программного продукта, но и расширяет область его применения. У Flash есть свои особенности, связанные с тем, что это средство создания векторной анимации для Web. И здесь важны все три термина: векторная, анимация, Web.

Для начала несколько слов о "родных" форматах. Рабочий файл программы имеет расширение FLA. Здесь хранится полная информация о слоях, импортированных изображениях и звуках, даже если они и не использовались при создании ролика. Естественно, в такой ситуации важнее полнота и качество информации. Но ситуация полностью меняется, когда заходит речь о публикации в Internet. Здесь желательно удалить все неиспользуемое, сжать растровые изображения и звуки. Во Flash для каждого типа импортированных файлов используется свой алгоритм сжатия. Также в ходу большое количество дополнительных трюков, например, запоминается не каждый новый кадр, а только изменения в кадрах. В результате Flash предлагает еще один формат — SWF, расшифровка которого претерпела эволюцию от Shockwave Format до Short Web Format (краткий Web-формат). По размерам SWF-файл в десятки раз меньше файлов FLA, да и для его просмотра достаточно небольшого по размерам проигрывателя (FlashPlayer, находится в папке Macromedia\Flash MX\Players и "весит" менее 1 Мб).

А вот все остальное приходится импортировать и экспортировать отдельно. Командой импорта мы уже неоднократно пользовались: **File/Import**. Она одинаково хорошо срабатывает как для векторных графических форматов (AI, EPS, EWF, WMF), так и для растровых (GIF, JPEG, PNG, BMP). Затруднения иногда возникают при использовании форматов фирмы

Рис. 1



Adobe (AI, EPS), особенно если они созданы приложениями не от Adobe (например, у автора регулярно возникают проблемы с EPS-файлами, сгенерированными Mathematica). Тогда вам придется просмотреть сообщение о нарушении адобовского соглашения о структуре документов (ADSC) вместе с уверениями, что такой формат продуктами Macromedia все-таки поддерживается.

Кроме статических картинок, возможно импортирование и анимационных файлов. Конечно же, возможно использование SWF-клипов. Так, на **рис.1** показан результат выполнения команды **File/Import** для нашего ролика с автомобилистом. Увы, каждый кадр импортируется отдельно и становится ключевым. К тому же, активные элементы, такие как управляющие кнопки, и вовсе теряют свои функции. Положение могут спасти только специальные программы-декомпиляторы, такие как Sothink SWF Decompiler MX. Их применение тоже потребует большого труда, но в этом можно найти и положительные моменты. По крайней мере, ваше творение не так уж просто будет позаимствовать без авторского согласия.

Версия Flash MX сильно порадовала расширением списка воспринимаемых видеоформатов (AVI, MPEG, ASF, WMV). На **рис.2** показан результат эксперимента со 117-секундным мультиком формата MPEG. Эксперимент оказался рискованным: Duron 900 около 15 минут обрабатывал мультик, но конечный файл получился в 9 раз меньше исходника. К тому же, сейчас мультик готов к добавлению кнопок остановки и запуска изображения, да и просто к вставке на Web-страницу. Не стоит забывать, что Flash создавалась не для обработки видеоизображений, поэтому более правильно такие файлы предварительно подправить и сократить хотя бы при помощи Windows Movie Maker.

Работу со звуками (WAV, MP3) мы рассмотрим несколько позже, а пока вернемся к традиционной графике. Следует сказать, что работа с GIF-анимацией сильно напоминает рассмотренную выше работу с видео. Автоматически создаются ключевые кадры, если захочется что-нибудь изменить, то вполне подойдут стандартные команды разгруппировки и рисования (**рис.3**).

Рис. 2

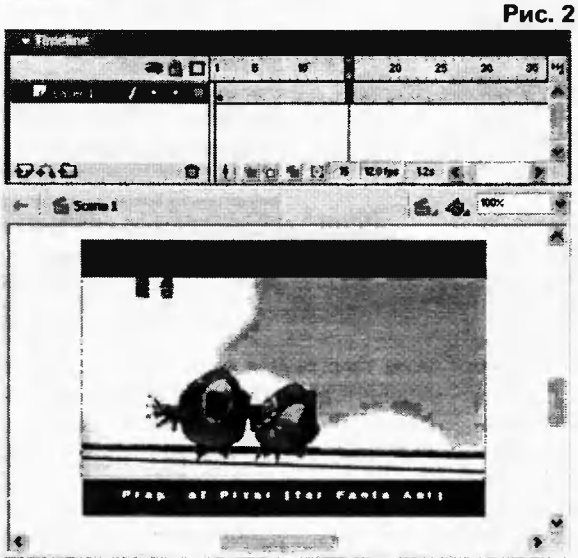
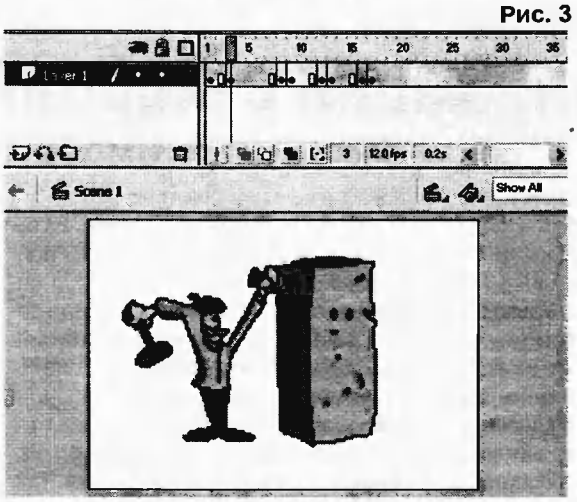


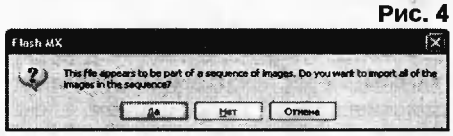
Рис. 3



Возможна и такая ситуация, когда в другом редакторе или другой программой была подготовлена последовательность изображений, и остается только объединить их в одну анимацию. Вариант с покадровым импортом каждого изображения, естественно, не рассматривается. Проще всего дать файлам одинаковые имена плюс номера кадров — что-то вроде frame01.gif, frame02.gif, ..., frame21.gif или anim001.eps, anim002.eps... Файлы должны размещаться в одной папке, тогда при импорте первого из них появится сообщение об обнаружении последовательности файлов с предложением импортировать всю группу файлов (**рис.4**).

Итак, изображение импортировано, и в большинстве случаев это изображение растровое. Недостатки этого формата — большой объем и ухудшение качества изображения при увеличении-уменьшении рисунка. Для графики, не насыщенной мелкими деталями, приемлемым вариантом может оказаться конвертация в векторный формат. Не так уж и много программ в состоянии достойно справиться с этой нелегкой задачей, "классика жанра" — Adobe Streamline. Flash MX, немного уступая Streamline в количестве настроек, в целом справляется с преобразованием примерно с тем же успехом. Для обработки импортированного изображения необходимо его выделить и выбрать команду **Modify/Trace Bitmap**. В диалоговом окне трассировки указываются параметры преобразования. **Color Theshold** (порог цвета) определяет, насколько близкими должны быть оттенки, чтобы восприниматься программой как один цвет. **Minimum Area** (минимальная площадь) задает в пикселах размер фрагментов, на которые может быть разбито исходное изображение. **Curve Fit** (подгонка кривой) указывает, насколько близко контуры конечного изображения будут следовать оригиналу, этот параметр принимает значения от **Pixels** (с точностью до пиксела) до **Very**

Рис. 4



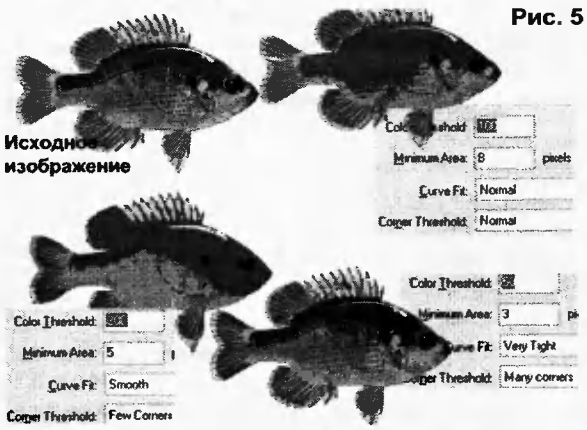


Рис. 5

Smooth (очень гладко). Наконец, **Corner Threshold** задает степень сглаживания углов. Практически для каждой картинки эти параметры приходится подбирать индивидуально для достижения оптимального соотношения размер/качество (**рис.5**). Общая картина напоминает оцифровку звука — при преобразовании происходит некоторая потеря качества, зато в дальнейшей работе новых искажений не возникает (**рис.6**).

Рис. 6

Исходный текст

Текст после обработки и увеличения

Исходный текст

После того как изображения импортированы и должным образом обработаны, ключевые кадры заданы, автозаполнение настроено, и анимация доведена до совершенства, возникает законный вопрос — а что дальше? Не смотреть же на все это только в окне Flash! Значит, настала пора публикации нашего фильма. Внутренний голос должен подсказать вам, что соответствующие настройки стоит поискать в меню **File/Publish Settings** (файл/настройки публикации — **рис.7**). В диалоговом окне предлагается выбрать типы файлов. Ограничимся только анимированными файлами, кстати, по умолчанию вам предлагается оставить исходные имена файлов и ограничиться только разными расширениями (флажок **Use default names**). Первым в списке идет уже описанный выше файл SWF, чуть ниже на основе данного файла предлагается создать HTML-документ. В этом случае полностью будет работать использованный программный код и все интерактивные эффекты. Для просмотра файла, как уже говорилось, должен быть установлен проигрыватель, причем желательно версии, не ниже использованной версии Flash. Можно вспомнить, что известная серия мультиков про Масяню, вроде бы, прекрасно обходится и без него. Это достигается за счет создания исполняемого EXE-файла. Он порой в десятки раз больше SWF, зато для проигрывания не требует ничего, кроме операционной системы. Примерно так же созданы многие заставки даже в Windows. Для создания "самоиграющего" ролика необходимо просто выбрать **Windows Projector**. Конечный продукт можно получить и в формате **QuickTime**. Из интерактивных элементов здесь будут поддерживаться только разве что простейшие кнопки (например, остановки анимации). Наконец, есть возможность пожертвовать и программным кодом, и качеством изображения, зато получить на выходе анимированный файл почти универсального GIF-формата. По мере того как выбираются форматы файлов в диалоговом окне, появляются корешки вкладок для настроек нужного формата (**рис.7**).

Рис. 7

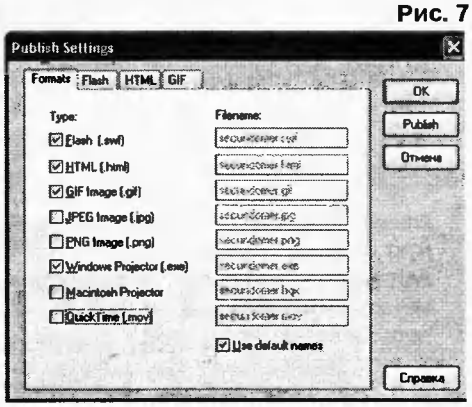
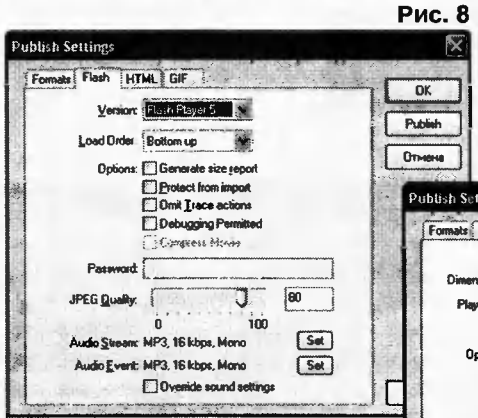


Рис. 8



В первую очередь рассмотрим SWF (**рис.8**). Здесь предлагается выбрать версию Flash-проигрывателя, для которой предназначен фильм, однако не все "навороты" Flash MX будут понятны более ранним (4-й и 5-й) версиям. Также можно указать порядок загрузки объектов (Load Order) — имеется в виду именно логический порядок элементов внутри сложной анимации. Для предотвращения скачивания анимации в Web-страницы можно установить флажок **Protect from Import** и защитить файл паролем. Если все-таки пришлось использовать растровую графику, качество ее передачи (зависит от степени сжатия) можно настроить выбором параметра **JPEG Quality**.

Что же касается GIF-файлов, то сильной их стороной является поддержка анимации и прозрачности, а слабой — возможность передачи только 256 цветов. Это крайне низкий показатель, поэтому львиная доля настроек связанна именно с цветопередачей (**рис.9**). Можно оптимизировать цветовую палитру (**Optimize Colors**) и указать предпочтительную палитру (**Palette Type**). Хуже всего передается градиентная заливка, поэтому предлагается либо убрать ее (**Remove Gradients**), либо преобразовать в смесь цветов (**Dither Solids**). В последнем случае есть три варианта действий, все они приводят к не очень хорошим результатам (**рис.10**). Если нужно получить именно качественную GIF-анимацию, то за качество нужно бороться не настройками публикации, а еще в процессе рисования, выбирая стандартные цвета и отказываясь по возможности от градиентных заливок. Рисунок будет анимационным, если не забыть установить переключатель в положение **Animated** и задать параметры воспроизведения (непрерывно — **Loop Continuously**, или повторить определенное количество циклов). Гото-

Рис. 9

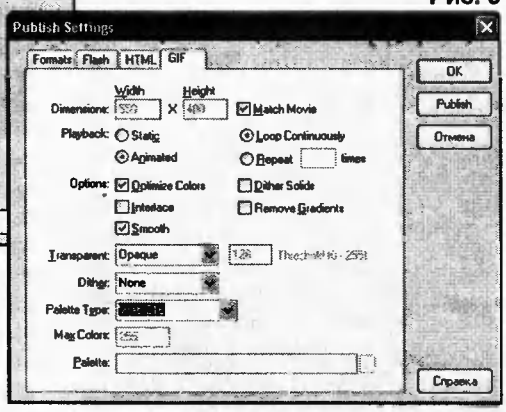


Рис. 10

вый файл можно использовать, например, в презентации PowerPoint, причем вставляется он стандартной командой **Вставка/Рисунок/Из файла**.

Преобразование в HTML-формат мы опустим по следующей причине. Flash в этом режиме создает страницу с одним-единственным элементом — внедренным объектом формата SWF. Дорабатывать ее до нужного вида довольно неудобно. Проще поступить иначе — создать HTML-страницу и вставить SWF-файл. Однако соответствующий код должен содержать информацию о версии файла и адресе загрузки обновлений проигрывателя. В результате код принимает довольно громоздкий вид (для файла MyBan.swf из папки img):

```
<object classid="clsid:D27CDB6E-AE6D-11cf-96B8-444553540000"
codebase="http://download.macromedia.com/pub/shockwave/cabs/ flash/swflash.cab#version=5,0,0,0"
width="180" height="320">
<param name="movie" value="img/MyBan.swf">
<param name="quality" value="high">
<embed src="img/MyBan.swf" quality="high" pluginspage="http://www.macromedia.com/go/getflashplayer"
type= "application/x-shockwave-flash" width="180" height="320"></embed>
</object>
```

Тот же эффект достигается автоматически при использовании редакторов Macromedia (DreamWeaver, HomeSite). Эти программы содержат команды вставки SWF-файлов и автоматически генерируют правильный код.

В итоге мы вплотную подошли к созданию Web-страниц и анимации для Web. На очереди оптимизация файлов и интерактивные элементы — как раз то, что необходимо в Internet. И одно, и другое достигается за счет использования символов. Именно о символах мы поговорим в следующей нашей статье.

Раз болванка, два компакт

Р.КАРПАЧ,
E-mail: Commander-Norton@tut.by
www.fdd5-25.narod.ru

Все знают программу для записи компакт-дисков Nero. И всем прекрасно известно, какой это замечательный продукт. Казалось бы, что еще нужно нормальному пользователю? Вроде бы, и ничего. Но все же есть одно маленькое "НО" — программа Nero версии 5.0 и выше не работает с операционной системой Windows 95. Многие скажут, что это пустяк, нужно лишь установить более раннюю версию пакета. Но в том-то и беда! Пакет Nero ниже версии 5.0 уже не работает со множеством современных приводов, например, мой **MITSUMI CDW58E** он вообще отказывается воспринимать **(рис.1)**.

Рис. 1

И вот, расстроенный и обескураженный таким результатом работы программы, я бросился в Internet на поиски альтернативы Nero. К моему глубокому удивлению, программ для записи компакт-дисков оказалось очень много. Но большинство из них или были неудобными в настройках, или вновь не распознавали привод CDW58E. Путем отсеивания нерабочих или неудобных пакетов, прямо как по Дарвину, были обнаружены две замечательные, полностью работоспособные программы. О них я и хочу сегодня поведать всем читателям. Сразу же следует отметить, что хотя описанные ниже программы работают под Windows 95, все они 2002 г. выпуска.

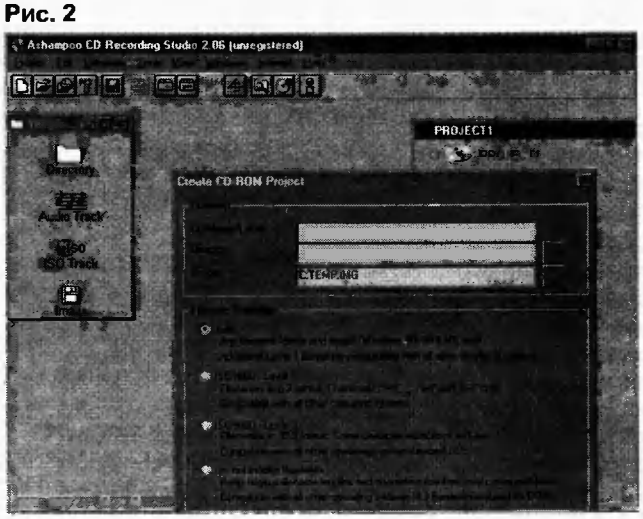
Программы для записи компакт-дисков

Ashampoo CD Recording Studio (shaware) — www.ashampoo.com

После первого запуска программы впечатление было шоковое, так как создатели Ashampoo отошли от стандартного вида, присущего Nero. Интерфейс напоминал что-то времен Windows 3.x. **(рис.2)**.

Но после беглого просмотра функций впечатление изменилось в положительную сторону. Самый большой плюс Ashampoo — это простота. В программе минимум функций, что позволяет пользователю не путаться в разных, иногда бесполезных настройках. Но каждая бочка меда не может быть без ложки дегтя — Ashampoo ни в каком виде не признает русские имена файлов. Вследствие этого создаются непонятные имена папок. Поэтому прежде чем записывать диск с помощью Ashampoo, следует все русские файлы переименовать в английские. Это же касается и каталогов. В целом же выбор функций для записи стандартен: запись CD-ROM с данными, запись аудио-диска, ну и, естественно, копирование. Также есть возможность импортирования play-листа и создания образов дисков. Перед записью можно протестировать матрицу на наличие физических повреждений.

Рис. 2



CD-Mate (shaware) — www.cd-mate.com

Как вы видите, программа очень напоминает знаменитую Nero **(рис.3)**.

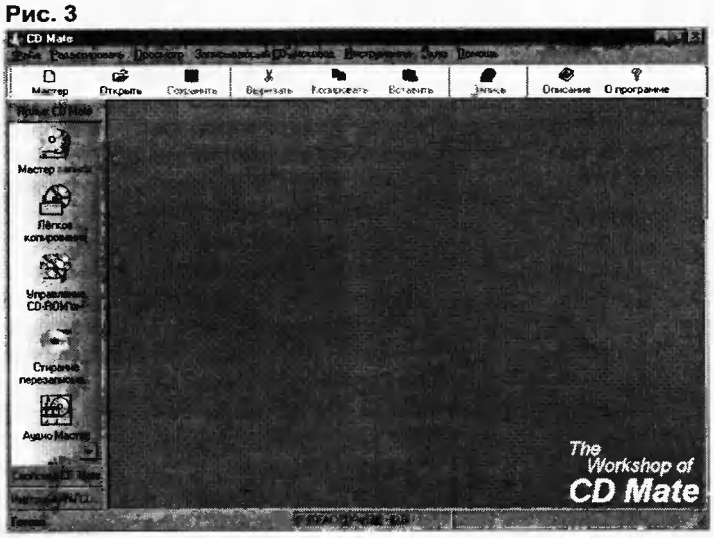
Но, в отличие от нее, CD-Mate не нуждается в русификации, так как на сайте имеется великолепно локализованная русская версия. В принципе, функции CD-Mate ничем не отличаются от предыдущей программы, кроме некоторых вещей, одна из которых — очень удобный и понятный wizard создания аудиодисков. Он позволяет без особых проблем перевести mp3-файлы в дорожки на CD-ROM и при этом выбрать нужное качество воспроизведения. Также пользователей должна порадовать опция "управление CD-ROM". Она дает полную информацию о функциональных возможностях вашего привода. Еще одна замечательная вещь — обновление программы через Internet. Поверьте, иногда это бывает очень нужно.

Что выбрать, CD-R или CD-RW?

Технологии создания компакт-дисков за последние годы прошли большой путь. От громадных

записывающих студий до маленьких домашних приводов CD-RW. Для достижения наилучших результатов записи осталось только два типа носителей — CD-R- и CD-RW-диски. Так что же выбрать для себя? Казалось бы, пустая матрица CD-R стоит дешевле, но, с другой стороны, диск CD-RW позволяет перезаписывать информацию несколько раз. Да еще эта вечная проблема нечитаемости! Ведь частенько бывает, что запишешь матрицу, отнесешь куда-нибудь, а она не читается. Почему? А дело вот в чем. Современные матрицы размером в 700 Мб, хотя и могут быть записаны на 650 Мб, не всегда читаются на старых приводах CD-ROM. Поэтому специалисты советуют: если вы знаете о том, что накопитель старый, лучше запишите для него диск размером 650 Мб.

Рис. 3



Использование перезаписываемых дисков CD-RW предполагает:

- использование компакт-диска, как если бы это был флоппи-диск, но большего объема;
- еженедельное копирование файлов с важной информацией;
- освобождение пространства на вашем жестком диске;
- передачу файлов с домашнего компьютера на рабочий и наоборот.
- создание индивидуализированных презентаций;
- хранение сообщений электронной почты.

Использование дисков CD-R (однократно записываемых) предполагает:

- постоянное хранение больших проектов;
- совместный доступ к нередактируемым презентациям;
- постоянное хранение домашних записей;
- обмен памятными событиями и фотографиями с друзьями и близкими;
- запись музыки на компакт-диск для личного пользования;
- постоянное хранение архивных файлов.

Некоторые устройства CD-ROM могут не считывать информацию с носителей CD-R. Если у вас возникли проблемы, отключите свойство оптимизации упреждающего чтения на вашем устройстве CD-ROM. Для этого щелкните правой кнопкой мыши по ярлыку "Мой компьютер" на рабочем столе Windows, выберите пункт "Свойства" и щелкните по закладке "Быстродействие". Далее щелкните по кнопке "Файловая система", затем по закладке "Компакт-диски" и выберите опцию "Без упреждающего чтения диска" в списке окна "Оптимизация доступа".

Как мне сказал один хороший знакомый: "Ты бы еще диски под Windows 3.11 записывать начал!" А что, хорошая идея...

Знакомство с Virtual Micro Lab

Е.ЛЕБЕДЕВ,
г.Череповец.

Микроконтроллеры — это сложные цифровые устройства, которые заменили "сердце" многим современным устройствам и приборам. Встретить их можно везде! Микроконтроллеры используются в автомобилях, персональных компьютерах, стиральных машинах, музыкальных центрах и т.д. Чем больше современный мир развивается, тем больше возникает необходимость использования в высоких технологиях этих устройств. Чтобы чуть более подробно ознакомиться с этим "чудом техники", мы вкратце рассмотрим структуру типичного современного микроконтроллера:

- центральный процессор (куда же без него?) — является мозгом контроллера и управляет его действиями;
- память программ — хранит в себе программу работы микроконтроллера (прямо как у Терминатора);
- цифровой порт ввода/вывода — с его помощью можно управлять одновременно несколькими линиями ввода/вывода;
- последовательный порт — позволяет обмениваться данными с внешними устройствами;
- тактовый генератор — задает (определяет) скорость работы микроконтроллера;
- оперативная память данных — содержит в себе переменные программ, включая стек;
- таймер — служит для отсчета временных интервалов;
- сторожевой таймер — специальный таймер, служащий для предохранения от сбоев программы.

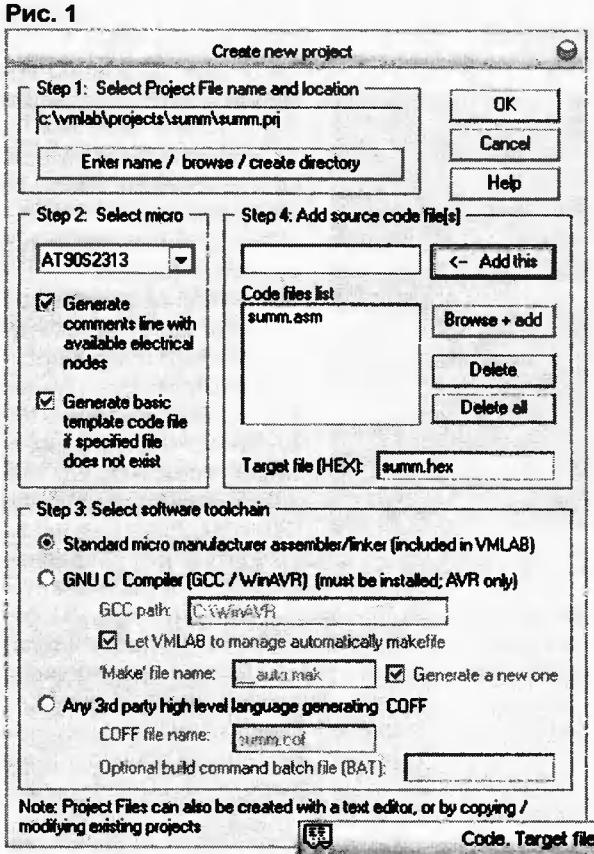
Судя по перечисленным блокам микроконтроллера — это очень универсальное и, следовательно, экономичное устройство. При использовании микроконтроллеров дополнительные схемы сопряжения не требуются, что позволяет снизить размеры конструкции и энергопотребление от источника питания. Как мы уже знаем, микроконтроллеры имеют собственную память, которая поддается программированию. Для программирования разработан ряд программ, разных по сложности и диапазону возможностей, но все они созданы с одной целью — "зашить" программу в микроконтроллер.

В мире программаторов устоялись два языка программирования — Си и Ассемблер. Безусловно, эти два языка имеют сходство с "настоящими" Си/Си++ и Ассемблером, но в нашем случае они "заточены" для работы с микроконтроллерами. Для обоих языков существуют специальные программные продукты. Например, для Си широкое распространение получил продукт CodeVision AVR, а для Ассемблера — Virtual MicroLab и AVReal. Я не стал приводить другие soft-программаторы, так как их обзор не является целью этой статьи. Сегодня мы рассмотрим очень популярный, простой в понимании, и, в то же время, мощный программатор фирмы Advanced Micro Tools (www.amtools.net) — Virtual Micro Lab (далее — VML).

Этот программаитор мне понравился своей простотой и, в то же время, большим набором функций, которые включает в себя каждый "уважающий себя" современный микроконтроллер :).

В VML работа с каждым микроконтрол-

Рис. 1



лером организуется в виде проекта. Чтобы создать новый проект, достаточно в меню выбрать — Project/New Project. Появится окошко, как показано на **рис.1**.

Здесь все предельно просто — нам предлагается по шагам создать каркас будущей программы (прошивки) для микроконтроллера.

На первом шаге требуется указать директорию, в которую будут "складываться" все файлы, относящиеся к проекту.

Шаг 2 — выбираем тип используемого контроллера. Выбор достаточно большой — нам предлагается 32 модели. Для начинающих вполне подойдет AT90S2313.

Третий шаг рассчитан на "продвинутых" пользователей, так что смело его пропускаем :).

Шаг 4 — здесь мы указываем файлы с ассемблерным кодом (*.asm), в которых находится наша программа для микроконтроллера. Если готовых файлов нет, то мы просто нажимаем кнопку "Add this" и добавляем пустой файл (*.asm) к нашему проекту. В нашем случае мы введем название "Summ.asm", так как чуть позже напишем программу для микроконтроллера, суммирующую два числа и выводящую результат в порт. Также требуется указать имя для hex-файла, который мы назовем "summ.hex". Этот файл будет содержать код программы для микроконтроллера, но в шестнадцатеричном представлении.

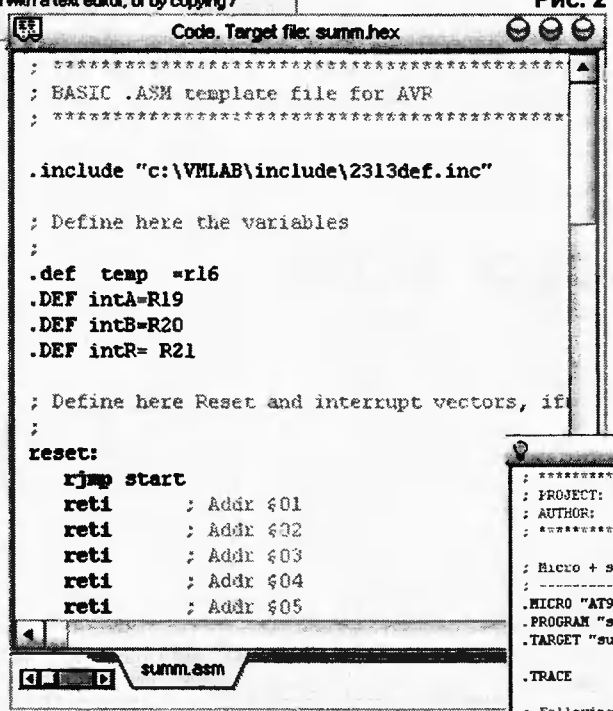
Жмем "OK". Итак, проект создан, и на экране мы видим три дочерних окна (по умолчанию): Code, Project и Messages.

Окно Code содержит в себе код программы для микроконтроллера. В нашем случае — это содержимое файла "summ.asm" (**рис.2**).

Окно Project содержит главный код проекта (**рис.3**).

Окно Messages — окно для выдачи сообщений во время работы над проектом.

Рис. 2



Для примера мы рассмотрим простую программку для микроконтроллера, выполняющую следующий алгоритм:

- в регистр R17 записывается шестнадцатеричное значение $17, а в регистр R18 — значение $3;
- числа суммируются;
- результат записывается в регистр R31 и выводится в порт D.

Рассматривая данный пример, я рассчитываю, что читатель имеет некоторое представление о структуре программ на языке Ассемблер.

В окне Code (рис.2) находится следующая структура:

- .include <имя файла> — здесь мы подключаем внешние файлы (заголовочные) с дополнительными функциями;
- .def <произвольная_метка>=<регистр> — ассоциация определенного регистра с произвольным именем;
- reset: — метка, начало обработчика сброса микроконтроллера в первоначальное состояние.

Первой командой в этой метке идет rjmp start. Эта команда осуществляет переход по коду программы к метке с именем "start". Что мы и сделаем. Идем к этой метке и видим следующий код:

```
start:
    Nop ;команда — "ничего не делать"
    …
    Nop
```

В теле этой метки код как таковой отсутствует — вместо этого стоят несколько команд NOP. Эта команда дает сигнал процессору микроконтроллера ничего не делать один такт. Следуя условию поставленной задачи, решаем первый пункт. Среди пор'ов пишем следующий код:

```
ldi R17, $17  ;Команда выполняет
              ;непосредственную
              ;запись константы ($17)
              ;в указанный регистр (R12)
```

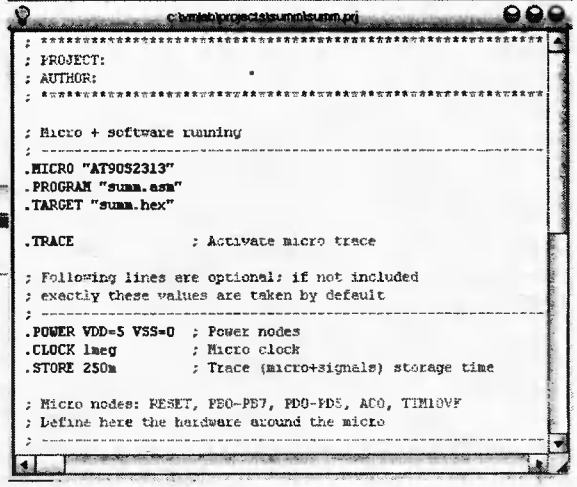
Так же поступаем и со вторым действием:

```
ldi R18, $3
```

Теперь нам требуется сложить эти числа:

```
add R17, R8   ;Если записать данное
              ;арифметическое действие
              ;на Си, то код будет
              ;выглядеть так: R12+=R15,
```

Рис. 3




```
                ;соответственно, результат
                ;операции будет помещен
                ;в регистр R12.
```

Помещаем (архивируем) результат в регистр R31:

```
mov R31, R17   ; R31=R31+R17
```

Рис. 4

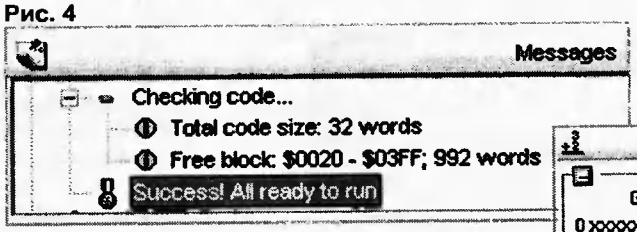
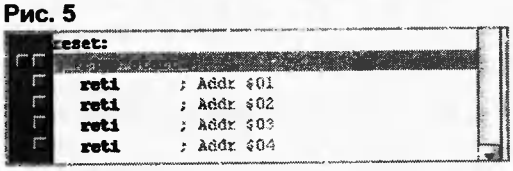


Рис. 5



И наконец, выполняем последнее действие:

```
out Portd, R31  ;Выводим результат
                ;в порт D
```

Маленькая программка готова, но мы не будем ее записывать в микроконтроллер, а просто наглядно просмотрим ее работу. В Virtual Micro Lab есть возможность отладки созданной программы и эмуляции ее выполнения, словно она уже находится в микроконтроллере.

Вначале скомпилируем набранный код (клавиша F9). Сигналом об успешной компиляции будет строка в окне "Messages" (**рис.4**) — "Success! All ready to run" (по-русски: "Успех! Все готово к запуску)". Если данной строки не наблюдается, вам следует проверить код программы еще раз.

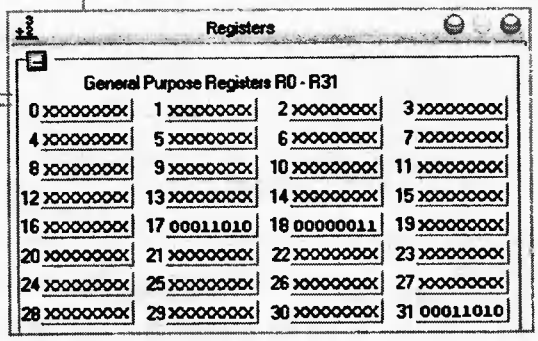
Теперь на панели инструментов нажимаем кнопку "Step Over" (пошаговое выполнение кода программы), тем самым запуская на выполнение первую строку программы:

```
rjmp start.
```

В окне Code можно увидеть, что данная строка подсвечивается (**рис.5**). Нажимаем еще раз кнопку "Step Over" (или используем "горячую" клавишу F6) и переходим к следующей строке программы:

```
nop.
```

Рис. 6



Заметим, что метки не являются командами, при трансляции программы в удобное для микроконтроллера представление все метки будут проигнорированы, и код будет записан линейно. Метки были введены для удобства написания программы программистом. Нажимаем F6 некоторое количество раз, пока не окажемся перед командой:

```
ldi R12, $17.
```

Теперь откроем дополнительное окно, которое показывает содержимое всех регистров. Для этого в меню нажимаем View/Registers/Flags. В результате появится окошко, как показано на **рис.6**.

Перед выполнением программы все регистры обнулены (XXXXXXXX). Теперь нажимаем два раза F6, выполняем команду

```
ldi R12, $17
```

и смотрим значение регистра R12:

```
00010111.
```

Нажимаем еще раз F6 — выполняется команда ldi R15, $3. Значение регистра 15 меняется на:

```
00000011.
```

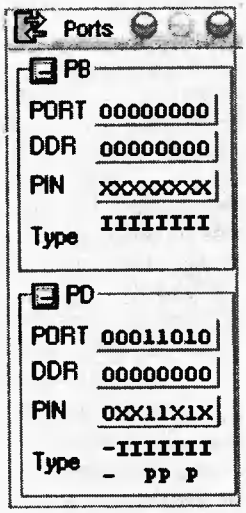
Выполняем команду

```
add R12, R15
```

и в регистр R12 записывается значение

```
00011010.
```

Рис. 7



Аналогично выполняется команда перемещения данных из регистра в регистр:

```
mov R31, R12.
```

И, наконец, команда вывода в порт D:

```
out PortD, R31.
```

Содержимое используемых в программе регистров к моменту завершения программы будет выглядеть, как показано на **рис.7**. Текущее значение, выведенное в порт D, можно посмотреть в окошке портов (View—I/O Ports), оно будет равно 00011010.

При выполнении последнего действия решаемой задачи мы вывели результат суммы в порт D. Следовательно, на выводах микросхемы AT90S2313 должны сформироваться три логические "1".

Литература

1. М.С.Голубцов. Микроконтроллеры AVR: От простого к сложному. — М.: СОЛОН-Пресс, Серия "Библиотека инженера", 2003.

2. www.avr123.by.ru

С.РЮМИК,
г.Чернигов.

Интернет-калейдоскоп, freeware #7

Не удивляй одеждой, а удивляй знаниями.
Монгольская пословица

По оценкам ученых, головной мозг человека имеет потенциальную информационную емкость до 10^{15} битов. И это при массе 1,2 кг, объеме 1,5 дм^3 и мощности потребления в электрическом эквиваленте всего 2,5 Вт! К сожалению, природные ограничения не позволяют нам в полной мере использовать возможности своей уникальной памяти. Человеку свойственно забывать.

С появлением Интернета проблема запоминания информации отошла на второй план. Действительно, что забыл — посмотри в Сети, начиная от школьных формул по физике и заканчивая кулинарным рецептом приготовления домашнего пирога. Главное — это выучить наизусть адрес хотя бы одного сервера-поисковика, например, http://www.google.com, корректно сформулировать в нем вопрос, а дальше — "язык до Киева доведет".

Поиск значения слова

Для расшифровки значения того или иного слова служат энциклопедии и толковые словари. Однако их книжные варианты занимают непомерно много места. Иное дело — электронный текст на экране компьютера, допускающий быструю навигацию и поиск. Например, на сайте http://www.yandex.ru имеется специальная закладка "Энциклопедии", при входе в которую обеспечивается поиск терминов из самых разных источников: БСЭ (Большой Советской Энциклопедии), "Брокгауза и Ефрона", "Москвы", справочников по экономике, военному делу и т.д.

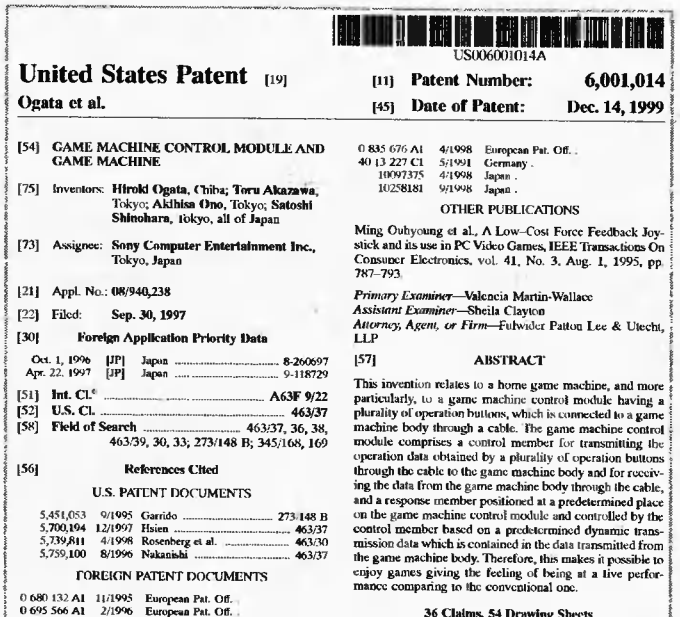
Существуют также специализированные сайты, посвященные электронным энциклопедиям "on-line": http://slovari.gramota.ru (поддержка Министерства по делам печати России), http://www.encyclopedia.ru, http://mega.km.ru (130 тысяч статей энциклопедии Кирилла и Мефодия), http://vidahl.agava.ru (Толковый словарь Даля), http://slang.od.ua (словарь сленга и жаргона), http://www.rubricon.com — официальные

НЕ ТОЛЬКО НОВИЧКУ

термины для всех желающих бесплатно, а углубленный уровень — по подписке или по системе баллов за участие в конкурсах эрудитов).

Те, у кто имеется возможность закачивать из Интернета файлы длиной 0,8...20 Мб, могут посетить сайт http://www.zipsites.ru/slovari. На нем размещены архивы 30 различных справочников "off-line" — от каталога имен и энциклопедии автомобильных гонок до статистического справочника о странах мира и

Рис. 1



US006001014A

United States Patent [19]
Ogata et al.

[11] **Patent Number: 6,001,014**
[45] **Date of Patent: Dec. 14, 1999**

[54] **GAME MACHINE CONTROL MODULE AND GAME MACHINE**

[75] Inventors: **Hiroki Ogata**, Chiba; **Toru Akazawa**, Tokyo; **Akihisa Ono**, Tokyo; **Satoshi Shinohara**, Tokyo, all of Japan

[73] Assignee: **Sony Computer Entertainment Inc.**, Tokyo, Japan

[21] Appl. No.: **08/940,238**

[22] Filed: **Sep. 30, 1997**

[30] **Foreign Application Priority Data**

Oct. 1, 1996 [JP] Japan 8-260697
Apr. 22, 1997 [JP] Japan 9-118729

[51] **Int. Cl.**[6] **A63F 9/22**
[52] **U.S. Cl.** **463/37**
[58] **Field of Search** ..... 463/37, 36, 38, 463/39, 30, 33; 273/148 B; 345/168, 169

[56] **References Cited**

U.S. PATENT DOCUMENTS

5,451,053	9/1995	Garrido	273/148 B
5,700,194	12/1997	Hsien	463/37
5,739,811	4/1998	Rosenberg et al.	463/30
5,759,100	8/1996	Nakanishi	463/37

FOREIGN PATENT DOCUMENTS

0 680 132 A1	11/1995	European Pat. Off.
0 695 566 A1	2/1996	European Pat. Off.
0 835 676 A1	4/1998	European Pat. Off.
40 13 227 C1	5/1991	Germany .
10097375	4/1998	Japan .
10258181	9/1998	Japan .

OTHER PUBLICATIONS

Ming Ouhyoung et al., A Low-Cost Force Feedback Joystick and its use in PC Video Games, IEEE Transactions On Consumer Electronics, vol. 41, No. 3, Aug. 1, 1995, pp. 787–793.

Primary Examiner—Valencia Martin-Wallace
Assistant Examiner—Sheila Clayton
Attorney, Agent, or Firm—Fulwider Patton Lee & Utecht, LLP

[57] **ABSTRACT**

This invention relates to a home game machine, and more particularly, to a game machine control module having a plurality of operation buttons, which is connected to a game machine body through a cable. The game machine control module comprises a control member for transmitting the operation data obtained by a plurality of operation buttons through the cable to the game machine body and for receiving the data from the game machine body through the cable, and a response member positioned at a predetermined place on the game machine control module and controlled by the control member based on a predetermined dynamic transmission data which is contained in the data transmitted from the game machine body. Therefore, this makes it possible to enjoy games giving the feeling of being at a live performance comparing to the conventional one.

36 Claims, 54 Drawing Sheets

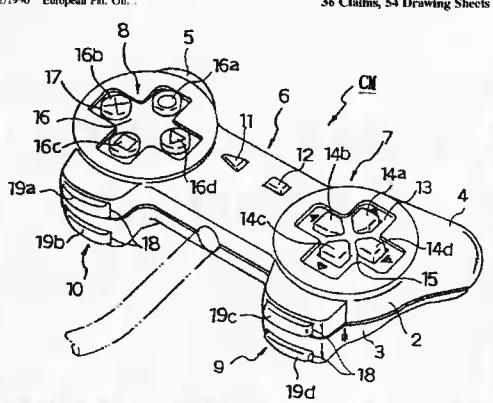


Рис. 2

толкового словаря Ожегова (**рис.1**).

Для удобства поиска энциклопедий в Сети можно ввести специальную надстройку в браузер — скачать файл http://dic.academic.ru/academicmenu.reg (329 Кб), закрыть Internet Explorer, щелкнуть два раза по файлу academicmenu.reg, в открывшемся диалоге нажать "Да", запустить Internet Explorer.

Ключевые слова для поиска: виртуальная энциклопедия, электронный словарь.

Поиск патентной информации

Легко было Архимеду, который воскликнул "Эврика!" и был абсолютно уверен, что никто до него еще не изобрел закон "плавучести тел". В наше время после "эврики" ставят не восклицательный, а вопросительный знак, после чего тщательно проверяют новшество на патентную чистоту, чтобы не прослыть "изобретателем велосипеда".

Специалист должен быть в курсе последних событий по интересующей его тематике. К сожалению, каталоги с описаниями отечественных открытий и изобретений прекратили свое обновление в начале 1990-х годов. Остается ориентироваться на зарубежные источники, многие из которых размещены в Интернете. В частности, крупнейшее собрание американских (US), европейских (EP) и японских (JP) патентов находится на сайте Delphion (http://www.delphion.com).

Строго говоря, доступ к полным текстам патентов является платным, однако предоставляется возможность свободного скачивания первой страницы формулы любого изобретения. Как правило, этого вполне достаточно, чтобы оценить суть идеи. Например, на **рис.2** представлен заглавный лист патента US6,001,014 на джойстик от игровой приставки "PlayStation 2", выданный 14.12.1999 г. четырем сотрудникам японской фирмы Sony Computer Entertainment Inc.

Поиск проводится как по номеру изобретения, так и по ключевым словам, разумеется, на английском языке. Небольшой нюанс. В США внутри номера патента обычно принято ставить разделительные запятые, но при поиске их вводить не надо.

Для работы на сайте Delphion требуется пройти бесплатную регистрацию, указав свой e-mail, и получить пароль. Это пригодится на случай оповещения вас о периодически проводимых акциях, когда в течение недели можно бесплатно скачивать полные тексты любых (!) патентов.

Рис. 3

Ключевые слова для поиска: Delphion, patent, патентный поиск, изобретение.

Поиск в электронных изданиях

Любитель компьютерной техники должен быть эрудитом (от лат. eruditio — ученость), т.е. человеком с глубокими всесторонними знаниями и широкой осведомленностью. Поддерживать эруди-

Свой сайт в Интернете

А.ГОРЯЧКИН,
г. Кыштым, Челябинской области.
E-mail: anatoliy_goryach@mail.ru

Интернет неуклонно продолжает вторгаться в нашу повседневную жизнь. Пользователям персональных компьютеров уже становится тесно в рамках виртуальных путешествий по Всемирной Паутине. Поэтому у них может возникнуть вполне естественное желание разместить в Интернете свою персональную Web-страницу. Однако недостаточно полная информация или ее отсутствие в отношении того, как это сделать, вызывает немало вопросов.

В эпоху бурного роста и стремительного развития Интернета все больше и больше пользователей пользуются разнообразными услугами, предлагаемыми специальными сервисными службами. Некоторые российские и зарубежные серверы предоставляют в бесплатное пользование дисковое пространство для размещения домашних Web-страниц. На этих серверах пользователи могут зарегистрировать доменное имя третьего уровня. В предлагаемой вниманию читателей статье приведены общие сведения и рассмотрены практические вопросы о бесплатном размещении в Интернете персональных Web-страниц.

Где в Интернете разместить свой сайт?

Для того чтобы иметь свой Web-сайт, предстоит решить две основных задачи — где и как разместить его в Интернете. В решении первой задачи нам поможет отечественная поисковая система "Апорт" (http://www.aport.ru). На ее главной странице выбираем рубрику каталога **Интернет**, и, поскольку мы намерены разместить Web-страницу бесплатно, то в содержимом этой рубрики "кликаем" по категории **Бесплатное в сети**. На открывшейся странице просматриваем список и выбираем из него те серверы, которые предоставляют пользователям доменные имена и место под Web-страницу (всего в этом списке около 200 серверов). Чтобы пользователям было удобнее выбирать, поисковая система "Апорт" указывает рейтинг каждого сервера.

Кстати, волнующий многих пользователей вопрос: почему все-таки пользоваться хостингом, т.е. размещением Web-страниц, можно абсолютно бесплатно? Как правило, серверы, предоставляющие какие-либо бесплатные услуги, всегда имеют высокую посещаемость. Компании, владеющие этими серверами, размещают рекламу не только на своих сайтах. Они оставляют за собой право размещать рекламу в

Имя сервера	Адрес сервера	Сервис	Дополнительная нформация
Кирилл и Мефодий	http://www.km.ru	Бесплатные сервисы — хостинг, электронная почта, открытки	
Новая почта	http://www.newmail.ru	Сервер бесплатного почтового доступа. Список услуг — предоставление электронного почтового ящика, доменного имени третьего уровня, 32 Мб под домашнюю страницу, рассылка новостей	Доменные имена: name.newmail.ru, name.hotmail.ru, name.nm.ru, name.nightmail.ru Зеркало сайта — http://www.hotmail.ru
Narod.ru	http://www.narod.ru	Предоставляется возможность: создать свой сайт с чатом, форумом, поиском, гостевой книгой, счетчиком, статистикой посещений; пользоваться хостингом; получить почтовый ящик	Надежный сервер. Высокая скорость доступа
Хобби.ру	http://www.hobby.ru	Регистрация и поддержка доменов третьего уровня, предоставление дискового пространства для некоммерческих домашних страниц	
Boom.ru	http://www.boom.ru	Бесплатный хостинг Web-страниц. Предложение дискового пространства — 50 Мб, предоставление шаблонов для построения сайта. Чат. Форум. Гостевая книга	Сравнительно новый сервер. Скорость доступа достаточно высока
LGG	http://www.lgg.ru	Для размещения Web-сайта выделяют 30 Мб.	Высокая скорость доступа
Агава	http://www.agava.ru	Объем дискового пространства выделяется по запросу пользователя. Получение почтового ящика. Включена поддержка CGI. Гостевая книга	Качественный и профессиональный хостинг
Chat.ru	http://www.chat.ru	Чат. Гостевая книга. Бесплатное место под Web-страницу	
HotBox	http://www.hotbox.ru	Предоставление бесплатной возможности зарегистрировать доменное имя третьего уровня и размещения Web-сайта объемом до 20 Мб. Предоставляется бесплатный почтовый ящик объемом 20 Мб	Доменные имена: name.hotbox.ru, name.fromru.com, name.pochtamt.ru, name.pisem.net, name.mailru.com, name.rbcmail.ru, name.krovatka.net
WebServis	http://webservis.ru	Неограниченное дисковое пространство, почтовый ящик. Имеется поддержка CGI. Широкий спектр услуг	Новый, надежный сервер с высокой скоростью доступа. Доменные имена: name.bos.ru, name.al.ru

цию на должном уровне позволяют электронные газеты и журналы, а также сборники статей и компьютерных обзоров.

На странице http://www.download.ru/russian/140.htm даны ссылки на архивы свободно распространяемых электронных изданий на русском языке, в частности, "Internet Zone" ("IZone"), "Компьютерные советы от CHIP", "Offline журнал EXE", "CompDocs" и др.

В качестве примера на **рис.3** изображена страничка из еженедельника "IZone" №585 (http://www.izcity.com, 472 Кб). Для читателей он бесплатный, можно заказать его рассылку по e-mail два раза в неделю (150 тысяч подписчиков). Финансово журнал держится на доходах от рекламы. Статьи к публикации на компьютерную тематику принимаются от любого желающего, но на некоммерческой основе. Для начинающих публицистов это хороший шанс заявить о себе, дать рекламу своему сайту или вновь созданной программе.

Для удобной навигации по электронным журналам разработана программа-оболочка FindJournals (автор — Вильнер Г.А., http://wiln.narod.ru/Files/SetupFindJournals_s.exe, 607 Кб). Она поддерживает 40 форматов разных журналов. Требования к компьютеру — Windows-9x/2000, Internet Explorer 5.0 и выше.

Ключевые слова для поиска: электронный журнал, компьютерное обозрение, IZone.

письмах и на Web-сайтах своих клиентов. Так что, если к вашим почтовым сообщениям или персональной странице в Интернете "приклеится" какой-нибудь "баннерный листок" — не удивляйтесь и не возмущайтесь. Это и будет ответом на ваш вопрос.

При выборе наиболее подходящего сервера следует учитывать не только размер дискового пространства, предоставляемого пользователю. Стоит обратить внимание на скорость доступа к серверу и на надежность его работы. Как правило, все инструкции по размещению и настройке на российских серверах приведены на русском языке. Выбор российского сервера внушает уверенность в том, что не возникнет никаких проблем с русской кодировкой. Да и вести деловую переписку с администрацией российского сервера проще на родном языке, а потребность в такой переписке может возникнуть у обеих сторон.

С помощью поисковых систем можно без особого труда найти серверы, предоставляющие разнообразный бесплатный сервис, в том числе место под Web-страницу и почтовый ящик. В **таблице** приводятся сведения о некоторых российских серверах, которые предоставляют возможность бесплатного хостинга персональных Web-страниц. Разумеется, этот список не претендует на полноту. Причиной тому, как я уже говорил выше, является стремительный рост и развитие Интернета, и поэтому оперативно отслеживать все изменения по хостингу не представляется возможным.

А теперь рассмотрим весь процесс размещения Web-страницы на конкретном примере. Я остановил свой выбор на сервере "Новая почта" (http://www.newmail.ru), принадлежащем компании ORC (Online Resource Center). Почему именно "Новая почта", а не какой-либо другой сервер? Причин для этого много.

Во-первых, это весьма быстрый и довольно надежный сервер. Недаром поисковая система "Апорт", из первой двадцатки претендентов, дала серверу newmail.ru самую высокую оценку.

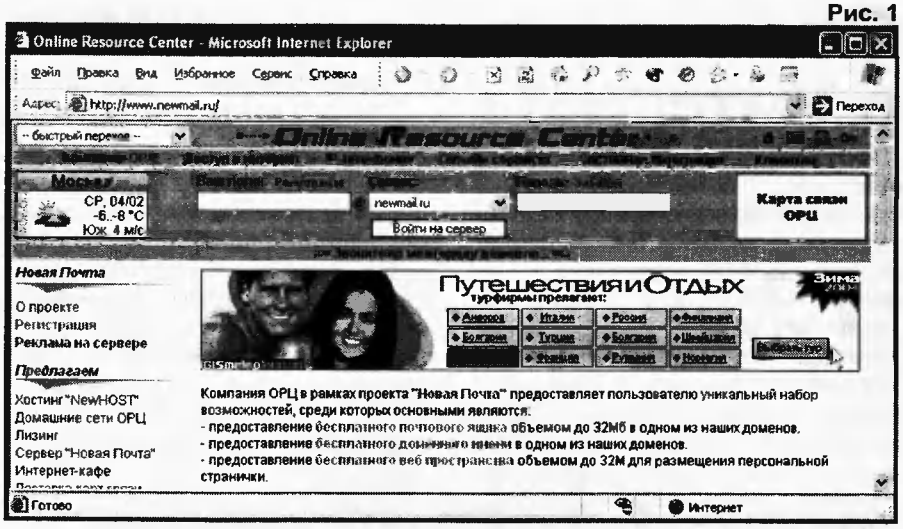
Во-вторых, "Новая почта" выделяет достаточно большое место под Web-страницу — можно разместить сразу целый корпоративный сайт.

В-третьих, привлекает возможность выбрать простое и короткое доменное имя nm.ru (см. таблицу), что, согласитесь, куда короче, чем, скажем, http://www.vasya.halyava.ru.

В-четвертых, имя сервера New Mail произносится благозвучнее, чем Boom.ru или Agava.ru.

И наконец, приятно удивляет тот факт, что практически сразу после окончания закачки файлов на сервер

Рис. 1



можно посетить свой "свежеиспеченный" сайт, без ожидания предварительной проверки и цензуры.

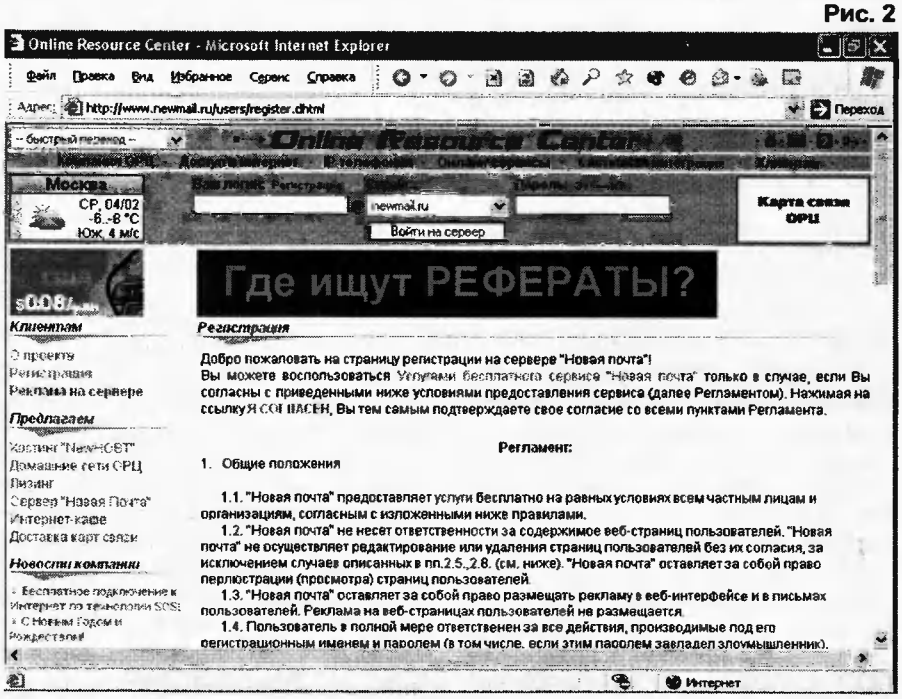
Итак, вкратце перечислю основные возможности, предоставляемые пользователю сервером "Новая почта": доменное имя третьего уровня, пространство объемом до 32 Мб для размещения персональной Web-страницы (**рис.1**), почтовый ящик объемом до 32 Мб. И все это удовольствие абсолютно бесплатно!

Регистрация

Для получения вышеперечисленного набора возможностей необходимо пройти обязательную регистрацию. Для этого на странице http://www.newmail.ru/users/register.dhtml следует ознакомиться со всеми приведенными условиями предоставления сервиса (**рис.2**). Не стоит опасаться слова "регистрация". Эта процедура — не что иное как упорядочивание отношений с сервером, который предоставляет место под размещение вашего сайта. Хотя регистрация и накладывает на пользователя определенную меру ответственности, но порядок должен быть в любом деле.

Воспользоваться услугами сервера "Новой почты" можно будет только в случае полного согласия со всеми этими условиями, что нужно обязательно подтвердить, щелкнув по ссылке "Я согласен", расположенной в нижней части страницы. После подтверждения автоматически происходит переход на страницу http://www.newmail.ru/users/reg.dhtml, где пользователю предлагают внести свои регистрационные данные в находящуюся там форму. Здесь следует указать свое регистрационное имя (логин) и выбрать доменное имя. Эти данные будут в дальнейшем использоваться для входа в систему, а

Рис. 2



также для подключения к FTP- и почтовому серверам. Логин может содержать от 2 до 20 символов и состоять из латинских букв, цифр и тире, например: "IvanSidorov". Регистр букв не имеет значения. Доменное имя должно быть выбрано из 4-х предложенных вариантов (**см. таблицу**). После того как будет указан логин и выбрано доменное имя, необходимо задать свой пароль. Пароль должен содержать от 5 до 16 символов и состоять из латинских букв, цифр и знаков подчеркивания.

Настоятельно не рекомендую использовать в качестве пароля свое имя, фамилию или любые другие данные, которые можно легко подобрать. Кроме того, он не должен совпадать с регистрационным именем. Наилучший пароль — случайный набор символов. После ввода пароля его нужно продублировать. На тот случай, если пользователь забудет свой пароль, вам предложат указать специальный вопрос и свой уникальный ответ на него. При желании контрольный вопрос можно выбрать из целого десятка готовых. Указывать вопрос и ответ не обязательно, но в этом случае процедура восстановления забытого пароля будет значительно затруднена. После окончания ввода вопроса и ответа необходимо "кликнуть" по полю "Продолжить регистрацию".

И вот, наконец, наступает заключительный этап регистрации. Если пользователь правильно указал все данные, он получает доменное имя третьего уровня и адрес электронной почты. Следует обратить внимание на то, что при получении необходимо установить "флажки" напротив адреса электронной почты и доменного имени, иначе почтовый ящик не будет активирован, а доступ на FTP будет закрыт.

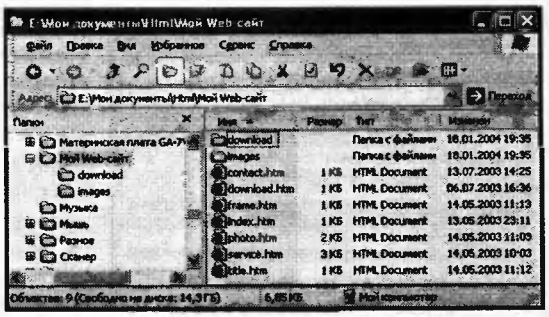
После окончания регистрации и получения доменного имени, а также адреса электронной почты можно приступать к непосредственному размещению Web-страницы в Интернете и настройке почтовой программы. В противном случае, если пользователь в течение 50 дней не воспользуется сервисами "Новой почты", т.е. не будет пользоваться Web-интерфейсом, FTP или почтовыми протоколами, его регистрационное имя удалят, о чем заблаговременно предупредят по электронной почте. Также удаляются все данные пользователя, а дисковое пространство освобождается.

Порядок регистрации на других бесплатных серверах примерно такой же. Главное при регистрации — внимательно ознакомиться с условиями предоставления сервиса. Эту информацию можно просмотреть и не находясь в Интернете, предварительно сохранив на своем компьютере нужные вам страницы.

Как разместить в Интернете свой сайт

Переходим ко второму основному вопросу: как разместить в Интернете свою персональную Web-страницу. Практически сразу после того, как пользователь зарегистрировался в системе "Новая почта" и получил доменное имя, он уже может разместить на сервере свои файлы или, говоря на сленге профессионалов, "залить" сайт. Разумеется, Web-страница (с графическими файлами, стилями и т.д.) к этому моменту уже должна быть создана и находиться на компьютере, с которого предполагается передача файлов на сервер (**рис.3**).

Рис. 3



При создании Web-сайта необходимо учесть, чтобы его заглавная страница имела имя index.htm или index.html, а в именах всех файлов (графических и т.д.) отсутствовали символы кириллицы. При этом имеет значение регистр символов.

Для размещения файлов Web-страницы можно воспользоваться любой специальной программой-клиентом, реализующей протокол FTP. Наиболее распространенные FTP-клиенты — CuteFTP, 3D-FTP, NetLoad, FTP Commander. Кроме того, допустимо использовать файловые менеджеры со встроенными в них FTP-клиентами. В их числе — широко известные Total Commander и Far Manager. С помощью FTP-клиента необходимо соединиться с сервером ftp.newmail.ru. В качестве имени пользователя следует указать свое регистрационное имя в системе "Новая почта" и домен (например, ivansidorov#nm.ru), а в качестве пароля — пароль, используемый для входа на страницу. Не забудьте между регистрационным именем и доменом указать символ "#"! В случае правильного ввода имени и пароля, у пользователя появляется возможность переслать файлы со своего компьютера на сервер ftp.newmail.ru. Кроме того, пользователь может переименовывать, а также удалять файлы и папки в директории, отведенной ему на этом сервере. Файлы можно размещать как в корневой директории, так и в любых создаваемых поддиректориях. Обычно в FTP-клиенте процесс передачи своих данных на сервер выглядит как простое копирование файлов с одной панели на другую.

Кроме специальных программ из числа FTP-клиентов и файл-менеджеров со встроенными в них FTP-клиентами, "заливку" сайта допустимо производить и с помощью браузера Internet Explorer. В этом случае браузер приобретает вид проводника Windows. На некоторых серверах для размещения файлов предлагают воспользоваться сервисом типа "Мои файлы". С помощью этого раздела вы можете управлять файлами на сервере по своему усмотрению, причем для этого не нужно никаких дополнительных специальных программ, достаточно возможностей интернет-браузера. Но все же FTP-клиенты с графическим интерфейсом и файловые менеджеры, реализующие протокол FTP, являются наиболее эффективным и удобным средством передачи файлов в Интернете.

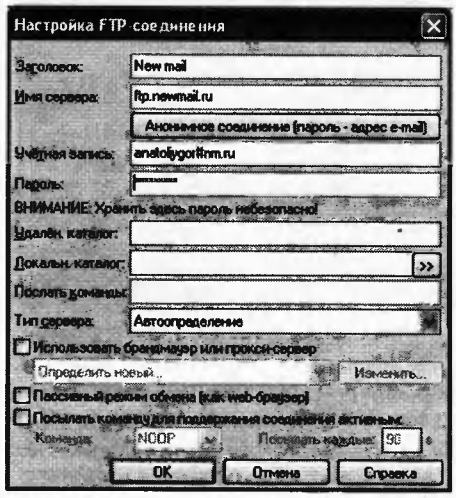
Total Commander в качестве FTP-клиента

Процесс передачи файлов Web-страницы на сервер заслуживает более подробного рассмотрения. В качестве FTP-клиента, как уже говорилось, можно использовать файловый менеджер Total Commander (желательно, русифицированную версию), поскольку эта программа довольно широко распространена. После загрузки файл-менеджера, в верхней части его окна на панели инструментов выбираем кнопку со значком FTP-соединения. После щелчка по этой кнопке откроется окно **Соединение с FTP-сервером**. Здесь и далее названия окон, меню, команды приведены на русском языке (версия 6.0).

Для того чтобы создать новое соединение, нажимаем на кнопку **Добавить**.... В **окне Настройка FTP-соединения** вверху в строке **Заголовок** указываем название FTP-сервера, например, New mail или Новая почта. В строке **Имя сервера** вводим адрес сервера **ftp.newmail.ru**. В строке **Учетная запись** указываем свое регистрационное имя (логин) и домен, а в строке **Пароль** — тот пароль, который был задан при регистрации. В остальных строках оставляем все без изменений (**рис.4**). Нажимаем на кнопку **ОК** и даем команду **Соединиться**.

После того как произойдет соединение, остается только выделить нужные файлы и скопировать их в свою директорию, отведенную на сервере, которая будет отображена на другой файловой панели менеджера. После окончания копирова-

Рис. 4



ния файлов набираем в адресной строке интернет-браузера полученное от сервера доменное имя и... наслаждаемся видом своего сайта в Интернете!

Если при размещении Web-сайта все же возникли какие-либо проблемы или неясные моменты, то в этих случаях следует обратиться за помощью. Как правило, на каждом сервере, предоставляющем бесплатный хостинг, существует разветвленная система помощи, организована техническая поддержка. На основе поступающих сообщений и писем составляются ответы на часто задаваемые вопросы (FAQ — Frequency Asked Questions). Так что пользователи не остаются один на один со своими проблемами.

Нет предела совершенству

В дальнейшем стоит заняться усовершенствованием и продвижением своего сайта. Хотите, например, узнать, сколько раз посещали вашу Web-страницу? Для этого вы можете разместить на ней простой счетчик посещений. О том, как это сделать, можно прочитать на странице http://www.newmail.ru/users/help/counter.dhtml. Но это еще не все. Кроме счетчиков посещений, на серверах в разделах помощи и технической поддержки указывают, где взять и как вставить в свою персональную Web-страницу другие стандартные дополнения: форумы, гостевые книги, средства для вывода результатов голосования.

При желании можно изменять и дополнять информацию, выложенную на сайте, периодически обновляя его с указанием даты последнего изменения. Причем, необязательно при обновлении копировать все файлы, достаточно — только новые и измененные.

Не забудьте и о "раскрутке" своей Web-страницы. Для этого нужно, не дожидаясь, когда роботы поисковых систем набредут на ваш сайт и проиндексируют его содержимое в своих базах данных, самим приложить некоторые усилия для индексирования сайта. На главных страницах практически всех поисковых систем и каталогов есть специальные ссылки **Добавить сайт** или **Добавить ресурс**. Щелкнув по этой ссылке, нужно ввести адрес своего сайта и дать ему краткую характеристику. А дальше рейтинг и популярность вашего сайта будет зависеть от того, насколько он окажется интересным и полезным для пользователей Интернета.

Заключение

Создание и размещение персональных Web-страниц — занятие весьма интересное и увлекательное. Тем более, что это не влечет за собой никаких финансовых расходов, кроме, разумеется, затраченного времени в сети Интернет. Так что дерзайте, изучайте язык HTML и основы Web-дизайна. Приобретайте полезные знания, которые помогут вам при создании персональных Home Pages (домашних страниц). Затем их можно будет разместить в Интернете и тем самым, с чувством собственного достоинства, заявить о себе и о своем существовании на весь мир. Не страшно?

Звук на вашей Home Page

Б.ШИЛЬНИКОВ,
Читинская обл., п.Дарасун.

Если у вас есть домашняя страничка, то несложно сделать так, чтобы при ее открытии начинала звучать какая-нибудь мелодия. Для этого потребуется добавить всего одну-две строчки в html-файл. Проще всего это сделать с помощью программы FrontPage Express. Это простой и удобный редактор web-страниц, он входит в состав некоторых версий Windows 98. Его можно добавить и в другие версии Windows.

Запустите FrontPage Express и откройте html-файл, который вы хотите озвучить ("Файл/Открыть", затем с помощью кнопки "Обзор" выберите нужный html-файл). Когда файл откроется в окне FrontPage Express, щелкните правой кнопкой мыши в поле окна. В открывшемся меню выберите "Свойства страницы". Это окно имеет несколько вкладок. Выберем вкладку "Общие". В параметрах "Фоновый звук" на этой вкладке с помощью кнопки "Обзор" нужно выбрать звуковой файл, который будет проигрываться при посещении вашей страницы. Например, C:\WINDOWS\MEDIA\canyon.mid. Сохраним измененную страницу. Если теперь загрузить страничку в MS Internet Explorer, мы услышим звучание выбранной мелодии.

При просмотре html-кода странички (Вид/Html) можно заметить, что в заголовке (часть документа между тегами <head> ... </head>) добавилась строчка

```
<bgsound src="file:///C:/WINDOWS/Media/CANYON.MID" loop="1">
```

Конечно, чтобы все работало, нужно в одном из каталогов сайта разместить midi-файл и указать в этой строчке правильный путь к нему. Например, `<bgsound src="pic/canyon.mid" loop="1">`, если файл находится в папке pic вашего сайта.

Лучше выбирать midi-файлы, т.к. при их небольшом объеме они звучат достаточно долго (файл длиной 20 Кб звучит около 2 минут). Wav-файл при большом объеме (десятки и сотни килобайт) звучит всего несколько секунд. Не всем может понравиться мелодия, поэтому в окошке "Цикл" лучше выставить значение 1 или 2 (число повторений мелодии). Можете установить флажок "Непрерывно", чтобы посетитель больше не возвращался на страничку с вашей любимой мелодией :). Если в папке C:\WINDOWS\MEDIA пусто, значит, вам нужно доустановить Windows. Откройте "Панель управления", щелкните по "Установка и удаление программ", выберите вкладку "Установка Windows", в ней компонент "Мультимедиа", отметьте галочкой "Образцы звуков" и нажмите "Ок".

Можно загрузить и запустить звуковой фрагмент по ссылке, например, так:

```
<a href="prezident.wav">Послушайте президента!</a>
```

Еще один способ запустить звуковой файл — поместить на вашу страничку изображение плеера с помощью следующей конструкции:

```
<embed src="canyon.mid" width=140 height=50 align=middle border=0 autostart=true>
```

Здесь:

- width — параметр, определяющий ширину плеера;
- height — параметр, определяющий высоту плеера;
- align — выравнивание по странице;
- border — ширина рамки плеера;
- autostart — признак автозапуска;
- true — запустить плеер сразу после того, как загрузится звуковой файл.

Плеер представляет собой элемент управления "Универсальный проигрыватель", имеет кнопки управления "Пуск", "Стоп" и др., что позволяет управлять воспроизведением звуковых модулей.

Литература.

1. Е.Якушина. Изучаем Интернет, создаем Web-страничку. — С.-Пб.: Питер, 2002, С.117.

2. Б.Леонтьев. Web-дизайн: тонкости, хитрости, секреты. — М.: Познавательная книга плюс, 1999.

3. http://www.microart.ru

ЛИСТАЯ СТРАНИЦЫ

Новый калькулятор известного производителя подобной техники — ф.Casio (www.casio.edu) с котором рас сказывается в статье "Casio Algebra FX: смерть на экзамене" **(Подводная Лодка, №2/2004, С.57)**, по своим возможностям приближается к миниатюрному ПК.

Тяжеловатый ящичек напоминает вычислительные машинки, которые были популярны в начале 80-х годов. Большой, прочный корпус, толстая крышка, 4 батарейки в качестве элементов питания. Правда, Algebra FX 2.0 Plus может похвастаться большим графическим дисплеем, прелести которого вы оцените, едва увидите пиктографическое меню — примерно такое, как в современных сотовых телефонах. Впрочем, из-за крупных точек монохромного дисплея, интерфейс калькулятора менее презентабелен, но за этой внешней неказистостью кроется недюжинный ум. Самый красивый сотовый телефон с цветным экраном в вычислительном плане может помочь разве что второкласснику, поскольку дальше умножения и деления его таланты не идут. А Algebra FX с легкость справится со сложнейшими уравнениями, графическим анализом функций, статистической обработкой и т.п.

Калькулятор оправдывает свое название при помощи встроенной технологии CAS (Computer Algebra System). Это означает, что он может решать математические задачи, не связанные с числами, то есть без численных методов. В специальном режиме можно без проблем интегрировать и дифференцировать функции, раскладывать выражение на множители, преобразовывать тригонометрические функции, упрощать выражения и сокращать дроби и т.п.

Впрочем, с числовыми задачками тоже не возникнет проблем, поскольку они решаются молниеносно, а ввод данных осуществляется в естественном виде. Не требуется никаких преобразований в уме, просто наберите на клавиатуре пример в том виде, в котором его обычно записывают на бумаге, проверьте на экране, что не допущено ошибки, с помощью курсора и клавиши стирания исправьте вкравшиеся опечатки, после чего нажмите клавишу EXE. Еще секунда — и результат на экране. Кстати, если пример использует обыкновенные дроби, они будут и в результате. То есть можно переписывать ответ с экрана на бумагу — опять без преобразований.

Калькулятор называется графическим, и вовсе не зря. Он прекрасно строит графики функций, хотя в первоначальном виде это не очень зрелищно — система часто неудачно выбирает масштаб. Но функция Zoom позволяет исправить этот недочет и проанализировать функцию, произвести численное интегрирование или поиск корней уравнения в графическом виде.

Ну а для тех, кто давно уже закончил школу и институт, все это станет инструментом решения более сложных задач, например, статистического анализа данных. Изобилие статистических инструментов и возможность работы с многомерными массивами данных впечатляют. Самое интересное, что массивы не обязательно набирать вручную — к специальному порту можно подключить датчики и регистрировать их показания автоматически. Можно также загружать данные с компьютера или другого калькулятора. А для их обработки никто не препятствует написать сложную вычислительную программу. Для удобства ее можно, опять же, написать на компьютере.

Технические характеристики калькулятора:

- число функций — более 1500, в том числе 619 научных (алгебра, тригонометрия, статистика, и т.п.);
- монохромный графический дисплей с разрешением 128x64 точек (21x8 символов);
- предельный объем загружаемых программ — 144 кб;
- максимальное число переменных — 28;
- коммуникации — последовательный порт для связи с ПК, другим калькулятором Algebra FX, внешними датчиками (до 38400 бит/с);
- габариты — 178x82x19,5 мм;
- масса — 213 г.

О новом комплекте для "общения" с компьютером рассказывает Е.Петров в заметке "Logitech diNovo Media Desktop: беспроводное "раздвоение личности" **(Подводная Лодка, №2/2004, С.40...41)**.

Это устройство включает беспроводные клавиатуру и мышь. Но клавиатура разделена на два блока — основную клавиатуру и цифровой блок, в обычных клавиатурах располагающийся справа и представляющий собой отдельное устройство. Причем этот "калькулятор" ф.Logitech снабдила LCD-дисплеем и превратила в самостоятельное устройство. Теперь он будет не только работать с числами, но и служить пультом дистанционного управления мультимедиа-содержимым жесткого диска (почему он и получил имя MediaPad), а также информировать о поступившей почте, грядущих мероприятиях и прочих событиях вашей жизни. Если же это вам не нужно, MediaPad просто можно убрать со стола.

Все это хозяйство связывается с компьютером через концентратор Bluetooth, который одновременно является блоком подзарядки аккумуляторов и средством подключения к компьютеру таких дополнительных устройств как мобильный телефон, карманный компьютер, ноутбук или принтер.

Для минимизации проблем с инсталляцией и настройкой, diNovo Media Desktop комплектуется специальным программным обеспечением, которое отвечает за быстрое подключение всех описанных выше устройств по привычной для Bluetooth схеме "спаривания". Приложение SetPoint — один из компонентов прилагаемого ПО — необходимо для тюнинга клавиатуры, мыши и мультимедиа-панели. Функционируя в фоновом режиме, оно следит за корректной интерпретацией пользовательских установок, в частности, за действием программируемых клавиш. Еще одна утилита — Mobile Phone Suite — предназначена для синхронизации компьютера и мобильного телефона. С ее помощью можно просматривать телефонную книгу, получать и отправлять SMS-сообщения, следить за входящими звонками (на экране выводится соответствующее оповещение). Ну а Logitech Media Desktop используется для управления (в основном, посредством панели MediaPad) мультимедийными функциями.

Системные требования:

- процессор Pentium III 500 МГц или выше;
- операционная система Windows 2000/XP;
- ОЗУ 128 Мб (рекомендуется 256 Мб);
- USB-порт;
- Windows Media Player 9.0 (содержится на прилагаемом CD);
- около 100 Мб свободного места на жестком диске.

Ориентировочная цена — 290 USD.

Все уже привыкли к тому, что современный ПК используется не только по своему прямому назначению — являться игровым центром для всех, кто имеет к нему доступ. Иногда на нем готовят курсовую или дипломную работу, или используют в качестве калькулятора. Но в последнее время все чаще **с помощью ПК прослушивают музыку, смотрят DVD-фильмы и даже телевизионные программы**. Эту особенность заметили производители и быстро отреагировали на нее. Об этом и рассказывается в статье "Мультимедийный компьютер Centro T730" **(Компьютер-Пресс, №2/2004, С.145)**.

По своему конструктивному дизайну развлекательно-компьютерный центр Centro T730, разработанный российской компанией Rover Computers (www.rovercomputers.ru), занимает промежуточное положение между ноутбуком и настольным ПК. Фактически, Centro T730 — это портативный моноблок (что, собственно, и роднит его с ноутбуком), но, в то же время, по своим характеристикам и размеру ЖК-экрана он не уступает полнофункциональным настольным ПК.

Компьютер имеет ЖК-экран с размером по диагонали 17 дюймов и рабочим разрешением 1280x1024, по бокам которого расположены две стереоколонки, что придает монитору стильный вид. Сам же системный блок ПК размещен непосредственно в корпусе и подставке ЖК-монитора.

ПК Centro T730 создан на базе процессора Intel Pentium 4 с тактовой частотой 3,06 ГГц и поддержкой технологии Hyper-Threading. Системная плата построена с использованием чипсета Intel 845PE с частотой FSB 533 МГц. Кроме того, ПК осна-

щен высокопроизводительной видеокартой ATI Radeon 9000 (AGP 4x) с DDR SDRAM-памятью объемом 64 Мб и жестким диском емкостью 120 Гб. Он снабжен сетевым адаптером Ethernet 10/100Base-TX и программным модемом с поддержкой протоколов K56Flex, V.90 и V.92.

Эта конфигурация характеризует Centre T730 именно как персональный компьютер. Фактически Centre T730 может работать в двух режимах: — PC и AV.

В PC-режиме система работает как мощный персональный компьютер, оснащенный современным процессором, высокопроизводительной видеоподсистемой, а также встроенным оптическим приводом, который дает возможность записывать собственные компакт-диски.

AV-режим превращает ПК в мультимедийный центр, который может работать как обычный телевизор, радиоприемник или DVD-проигрыватель.

Для того чтобы обеспечить широкие мультимедийные возможности, Centre T730 оснащен DVD/MP3-проигрывателем, TV-тюнером и FM-приемником. К тому же, в Centre T730 установлен комбопривод DVD-ROM/CD-RW (8xDVD/24xCD-ROM/24xCD-RW/16xCD-R), что позволяет не только прослушивать, но и записывать диски. Кроме того, Centre T730 снабжен картридером для карт форматов SM/MMC/SD/MS, PCMCIA-слотом, имеет два порта FireWire (стандартный и мини), четыре порта USB 2.0, разъем S-Video и композитный разъем для подключения внешнего видеосигнала, цифровой аудиовыход S/PDIF, разъем для микрофона и стандартные разъемы аудиовыхода и аудиовхода.

При всем многообразии функциональных возможностей, вес мультимедийного центра составляет всего 10,5 кг, а габариты — 483x361,5x88,5 мм. В общем, такой центр вполне мобилен, и его можно разместить в любом месте. Мобильность и комфортность центра подчеркивает наличие беспроводной клавиатуры и мыши, а также дистанционного пульта управления — обязательной детали любого домашнего телевизора или музыкального центра.

О том, что **ф.Intel предлагает новую технологию, позволяющую резко повысить качество изображения телевизоров**, сообщается в заметке "Новая технология Intel для телевизоров" **(КомпьютерПресс, №2/2004, С.165)**.

Эта технология, носящая кодовое название Cayley, построена на методе LCOS (Liquid Crystal on Silicon) и заключается в следующем: слой жидких кристаллов помещается между стеклянной пластиной и зеркальной полупроводниковой поверхностью с высокими отражающими свойствами, в которую вынесена вся схема управления пикселами. Из таких слоев состоят микродисплеи, которые могут использоваться в проекционных телевизорах с большим экраном. В LCOS-технологии современные процессы производства интегральных схем, разработанные корпорацией Intel, используются для создания высококачественной отражающей поверхности, которая обеспечивает высокую яркость изображения. Корпорация Intel разработала уникальный процесс производства микродисплеев LCOS, а увеличение количества транзисторов в таких устройствах по мере их развития, позволит разработчикам Intel интегрировать в них дополнительные функции, улучшающие такие характеристики экрана как яркость и качество изображения.

LCOS-технология корпорации Intel — полностью цифровая, поэтому обеспечивает воспроизведение значительно более четкого и точного изображения, чем в других архитектурах, основанных на аналоговых технологиях. Еще одним достоинством LCOS-технологии являлся возможность создавать одинаковые по размеру микродисплеи с разными уровнями разрешения. Однородная и полностью интегрируемая экранная область микродисплеев на базе LCOS-технологии корпорации Intel позволит OEM-компаниям использовать их в самых разнообразных видах продукции с разными разрешениями и размерами экрана, благодаря чему компании смогут снизить свои затраты на технические исследования.

Известная фирма Gigabyte представила новый, нетрадиционный для нее продукт — кулер 3D Cooler-PRO PCU21-VG. Об этом устройстве рассказывают *А.Кутехов* в заметке "3D Cooler-PRO: морозная свежесть" **(Подводная лодка, №2/2004, С.39)** и *С.Пахомов* в статье " Gigabyte 3D Cooler" **(КомпьютерПресс, №2/2004, С.136...137)**.

Основной особенностью кулера является то, что он построен на основе тепловых трубок (Heat Pipe). Тепловые трубки осуществляют эффективный отвод тепла от поверхности процессора. По коэффициенту теплопроводности они в сотни раз превосходят обычные медные трубки. По принципу действия тепловые трубки во многом схожи с термосифонами, в которых теплоотвод осуществляется за счет тепловой конвекции.

Простейший термосифон представляет собой полую, герметично закрытую трубку (например, из меди), внутри которой имеется небольшое количество рабочей жидкости. Термосифон располагается вертикально, а конец с жидкостью помещается в область повышенной температуры. При подводе тепла жидкость начинает превращаться в пар (зона испарения), который в результате конвекции движется вверх, то есть в зону с меньшей температурой. При остывании пар конденсируется и по стенкам термосифона под действием гравитационных сил стекает вниз. Для эффективного теплоотвода с помощью такого термосифона необходимо обеспечить постоянный отвод тепла от зоны конденсации, что можно сделать с помощью радиатора. То есть необходимо, чтобы всегда существовал градиент температуры и чтобы температура зоны конденсации была достаточно низкой для конденсации пара.

Термосифон может работать только тогда, когда зона испарения находится ниже зоны конденсации — в этом и заключается главный недостаток термосифона, ограничивающий его использование в системах охлаждения процессоров. Для построения универсальных систем охлаждения необходимо, чтобы теплоотвод осуществлялся при любом положении трубы, а не только при вертикальном. Однако для этого следует предусмотреть иной механизм достаки конденсата в зону испарения, то есть он должен происходить не под действием гравитационных сил, а возможно, и вопреки их действию. Таким механизмом возврата может служить капиллярный эффект в пористом материале. Именно данный механизм возврата жидкости в зону испарения и реализуется в тепловых трубках.

Тепловые трубы, используемые для системы охлаждения процессоров, обычно изготавливаются из меди, а на их внутреннюю поверхность наносят слой пористого материала. В качестве рабочей жидкости могут использоваться различные вещества, которые должны удовлетворять определенным условиям. Во-первых, рабочая жидкость должна иметь точку фазового перехода "жидкость-пар" в требуемом диапазоне рабочих температур. Во-вторых, жидкость должна обладать достаточно большой удельной теплотой парообразования, так как чем выше удельная теплота парообразования, тем меньше потребуется жидкости. В-третьих, жидкость должна обладать достаточно высокой теплопроводностью, чтобы свести к минимуму перепад температур между стенкой трубки и поверхностью испарения. В данном случае предпочтительнее использование жидкостей с высоким поверхностным натяжением, поскольку такая жидкость обладает ярко выраженным капиллярным эффектом.

Для охлаждения процессоров в качестве рабочей жидкости можно использовать воду (диапазон рабочих температур — от 30 до 200°C) или ацетон (диапазон рабочих температур — от 0 до 120°C).

Капиллярно-пористый материал, используемый в тепловых трубках, должен быть достаточно мелкопористым для улучшения капиллярного эффекта, однако слишком мелкопористая структура имеет очень низкую проницаемость и тем самым препятствует проникновению жидкости. Выбор капиллярно-пористого материала зависит как от рабочих температур, так и от общей длины тепловой трубки.

Зона контакта с поверхностью процессора (зона испарения для тепловых трубок) 3D Cooler выполнена из массивной медной пластины. Из этой зоны выходят вертикально вверх четыре тепловые трубки, на которые одет массивный листовой радиатор, образующий зону конденсации для тепловых трубок.

Для того чтобы обеспечить требуемый теплоотвод из зоны конденсации, внутри листового радиатора рвзмещен турбинный вентилятор, лопасти которого загнуты таким образом, чтобы выталкивать горячий воздух из радивторв наружу.

Достоинством этого вентилятора яаляется то, что в комплекте с ним поставляется VR-модуль, который можно крепить на лицевой панели корпуса в 3,5-дюймоаом отсеке, либо выводить за заднюю панель корпуса.

VR-модуль позволяет плавно регулироаать скорость врвщения вентилятора в пределах от 2000 до 4000 об./мин. Скорость вращения от 2000 до 2500 об./мин соответствует нормальной эксплуатации ПК, от 2500 до 3500 об./мин — игровому режиму с интенсивной нагрузкой нв процессор, а от 3500 до 4000 об./мин — режиму для оверклокинга. Конечно, в зависимости от скорости врвщения вентилятора меняется и уровень шума. Твк, согласно технической документации, при минимальной скорости вращения в 2000 об./мин уровень шума составляет 19,2 дБА, в при скорости вращения в 4000 об./мин — 37,2 дБА.

Для того чтобы оценить "шумность" рассматриваемого кулера, в редакции журнала "КомпьютерПресс" измерили посредством шумомерв по одной и той же схеме уровень шума, производимого тремя кулерами: 3D Cooler-PRO PCU21-VG, обычным боксоаым кулером, идущим в комплекте поставки с процессором Intel Pentium 4, и кулером Ajigo MF043-044A для процессоров AMD Opteron.

В соответствии с паспортными данными, последний имеет скорость вращения от 3050 до 6000 об./мин, а уровень шума — от 28 до 46 дБА. Согласно результатам измерений, уровень шума кулера 3D Cooler-PRO PCU21-VG составлял 46 дБА при скорости вращения 2000 об./мин, и 60 дБА при скорости вращения 4000 об./мин. В то же время, штатный кулер Intel производил шум в 49 дБА, а Ajigo MF043-044A — всего в 44 дБА.

Впрочем, уровень шума — это не единственная характеристика кулера. К несомненным достоинствам 3D Cooler-PRO PCU21-VG можно отнести его универсальность. Входящий в комплект поставки крепеж позволяет использовать его как для процессоров Intel Pentium 4, так и для процессоров AMD Athlon XP и Athlon 64. Другое несомненнов достоинство кулера — он отлично спревляется с возложенной на него задачей, обеспечивая эффектианый теплоотвод при значительной загрузке процессора.

Если вам приходится много работать за вашим ПК, рекомендуем ознакомиться со статьей *А.Юдова* "Взгляд на экран" **(CHIP, №2/2004, С.50...54).**

Большая часть опасений, связанных с длительным пребыванием перед экраном, связана с воздействием электромагнитного излучения. Его источниками являются монитор и системный блок компьютера. Наиболее вредное — ионизирующее излучение в рентгеновском диапазоне — возникает в процессе работы ЭЛТ при столкновении электронов со стеклянным экраном. В настоящее время экраны всех мониторов содержат некоторое количество свинца, что позволяет уменьшить уровень рентгеновского излучения до уровня естестаенного фона.

Другая опасность, имеющая ту же природу — электростатический заряд, возникающий на поверхности экрана. И эта проблема в наши дни достаточно легко решается: экраны мониторов двлают проводящими, и, благодаря слабым токам, потенциал поверхности всегда остается равным нулю.

В общепризнанном стандарте ТСО рекомендуется использовать заземленную клавиатуру. Это позволяет избежать возникновения электрического поля между экраном и пользователем, несущим на себе статический заряд. Особенно это актуально морозной зимой, когда воздух сухой.

Все вышесказанное относится лишь к ЭЛТ-мониторам. В силу конструктивных особенностей LCD-мониторов, в них не существует потока электроноа, бомбардирующих экран, следоватвльно, эти мониторы в рентгеновском диапазоне не излучают и статический заряд на себе не накапливают. Благодаря этому популяризация жидкокристаллических дисплеев долгое время проходила под девизом "абсолютно безопасный монитор для людей, заботящихся о своем здоровье". Конечно же, это не совсем верно, хотя доля истины а данном утаерждении есть,

Мониторы любого типа и системный блок компьютера являются источниками электромагнитного излучения низкой и сверхнизкой частоты. Достоаерно не известно, является такое излучение вредным для здоровья или нет, однако пока его влияние на человека не будет досконально изучено, стандарты обязывают производителей ограничивать его интенсивность. Таким образом, электромагнитный фон монитора (и системного блока с закрытой крышкой) не превышает некоего предела, который можно считать близким к уровню электромагнитного фона в городской черте.

Экраны современных мониторов имеют антибликовое покрытие, но оно практически не помогает, если вы будете сидеть спиной к окну, когда на экран падает прямой солнечный свет. С другой стороны, располагать монитор прямо перед окном тоже не стоит: слишком яркий фон не менее вреден, чем блики. Оптимальным является вариант, когда монитор установлен в нише компьютерного стола, отгороженного от окна. Этим вы гарантируете себе, что глаза не начнут уставать после пары часов работы, и тем самым сбережете свое здоровье. Некоторые профессиональные мониторы комплектуются специальными козырьками, защищающими экран от солнечных лучей. По опыту можно сказать, что это весьма полезное приспособление, и оно действительно облегчает работу с монитором.

Монитор желательно разместить на компьютерном столе. Компьютерный стол — это не просто конструкция с нишей для системного блока. Правильный компьютерный стол имеет полку для размещения клавиатуры, расположенную на рекомендованной для удобной работы высоте (порядка 75 см над уроанем пола), и специальную подставку, благодаря которой монитор находится выше уровня клавиатуры. Благодаря этому взгляд пользователя направляется чуть ниже верхнего края экрана. Таким образом, если ваш монитор будет слегка наклонен (угол не больше 15°), то не придется поднимать или наклонять голову, чтобы взглянуть на аерхнюю или нижнюю части экрана.

Если ааш рост сильно отличается от среднего, обязатвльно нужно купить кресло с регулируемой высотой. Использование компьютерной мебели позволит избежать хронической усталости шейных отделов позвоночника — весьма распространенной профессиональной болезни у людей, подолгу работающих с компьютером.

Для ЭЛТ-дисплееа одним из главных критериев безопасной работы является кедровая частота. Малое значение этой частоты влечет за собой мерцание экрана, что очень пагубно воздействует на зрение. Очевидно, что чем выше частота кадров, тем комфортнее будет смотреть на экран. Существует, однако, и обратная сторона медали. Не все графические адаптеры способны поддерживать достаточную четкость картинки при высоких частотах кадровой развертки. Особенно это касается работы при высоких разрешениях экрана (от 1280x1024 пикселоа). Если видеоадаптер плохо работает при высоких разрешениях, вы будете наблюдать размытость изображения. По своему воздействию на зрение это ничем не лучше мерцания — глаза постоянно пытаются сфокусироваться на нечетком изображении и быстро устают.

Поэкспериментируйте с различными частотами кадровой развертки. Если различие в качестве картинки для двух близких значений частоты будет заметным, лучше оставьте меньшую частоту. Однако частота меньше 75 Гц не рекомендуется.

После установки разрешения экрана и частоты кадровой развертки необходимо тщательно настроить линейность и ортогональность. Первое понятие означает отсутствие видимых искажений вертикальных и горизонтальных линий, а второе — их перпендикулярность друг другу. Это основные параметры, напрямую влияющие на то, насколько качественным будет изображение на экране. После настройки геометрических параметров экрана следует подобрать комфортные уроани яркости, контрастности и температуры цвета. Если вы не профессиональный художник или дизайнер, данные параметры можете выбирать, исходя из собстаенных предпочтений.

В силу конструктивных особенностей LCD-мониторов, из всего вышеперечисленного для них имеет значение лишь подбор комфортных яркости, контрастности и температуры цвета. Оптимальные линейность и ортогональность, так же как отсутствие размытия (при работе с аппаратным разрешением монитора) и мерцания, являются неотъемлемыми атрибутами LCD-дисплеев.

Кроме того, если ваш аидеоадаптер имеет цифровой выход, то, выбирая монитор, стоит отдать предпочтение модели с DVI-интерфейсом. Отсутствие двойного цифроаналогового преобразования в цепи видеокарта-монитор гарантирует максимально качественное изображение на экране.

Первым стандартом безопасности для компьютерной техники был стандарт MPR I, предложенный в начале 80-х годов Национальным департаментом стандартов Швеции. Однако более известным является его второй вариант — стандарт MPR II, утвержденный в 1990 г. Последний обязывает производителей следить за уровнем электромагнитного излучения, который не должен превышать допустимую величину и выдерживаться на расстоянии до 50 см от монитора. Стандарт также предъявляет требования к визуальной эргономичности: яркости, контрастности, равномерности яркости и контрастности, стабильности изображения.

Дальнейшее развитие стандартов безопасности во многом саязано с деятельностью Шведской федерации профсоюзоа — TCO. В 1991 г. она утвердила стандарт TCO'91 — более жесткий, чем MPR II. Согласно этому стандарту, требования к максимально допустимому уровню иэлучения должны выполняться на расстоянии 30 см от экрана и 50 см от остальной поверхности монитора. Годом позже стандарт был доработан и назван TCO'92. В него добавились требования, касающиеся электробезопасности и энергосбережения. Сейчас в европейских странах продажа мониторов, не соответствующих TCO'92, запрещена.

Новый стандарт Шведской федерации профсоюзов был принят в 1995 году и получил соотаетствующее название — TCO'95. От TCO'92 он отличался в основном наличием требований экологического плана и ужесточением требований по энергосбережению. Важным отличием от TCO'92 обладает лишь стандарт 1999 года, который предполагает, что требуемые параметры визуальной эргономики, аналогичные стандарту TCO'92, должны выполняться при более жестких условиях — большей яркости и частоте, а также при меньших уровнях электромагнитного излучения.

Последняя редакция стандарта — TCO'03. В ней ужесточены требования к визуальной эргономичности и экологическим характеристикам. По мнению экспертов, снижать допустимые нормы по излучению в ближайшие несколько лет не потребуется.

Стандарты для LCD-панелей и ЭЛТ-мониторов существуют в виде отдельных документоа, хотя требования к общим параметрам для обоих типов дисплееа, разумеется, совпадают. Стандарт подробно описывавт методику измерения и предельно допустимые значения либо отклонения от нормы для различных параметров. Оценивается размер пиксела, наличие или отсутствие мерцания, геометрические характеристики, яркость, контрастность и их равномерность. И это далеко не полный список требований к соаременным мониторам и процессу их производства.

Наиболее жестким стандартом является самый саежий — TCO'03, хотя в части требований по безопасности монитора для человека он не сильно отличается от TCO'99 или даже TCO'95. Даже если ваш монитор удовлетворяет лишь этому стандарту, его можно считать максимально безопасным. Однако специфика отечественного рынка такова, что наличие знаков иностранной сертификации не может а достаточной степени гарантировать, что вы покупаете безопасный и качественный монитор. Знаки могут быть подделаны, а сам товар может оказаться "некондицией". Степень доверия одним только лейблам TCO у нас не так высока, как а Европе.

Все официально поставляемые на наш рынок модели мониторов в обязатвльном порядке проходят сертификацию и должны нести на себе соответствующий логотип. Российские стандарты создавались и совершенствовались независимо от шведских, хотя во многом с ними совладают. Важное отличие состоит в том, что сертификация на соответствие производится в России, что значительно снижает вероятность обмана и подделки при импорте продукции. Кстати, список официально сертифицированных по стандартам TCO устройств доступен в Интернете — http://tco.networks.ru/index_publicsearch.htm,

Перед приобретением монитора прогоните на нем несколько раз утилиту Nokia Monitor Test, оцените визуальные параметры (если вы долгое время пользовались хорошим монитором, аозможные несоответствия сразу будут бросаться в глаза). Только тщательно изучив конкретный экземпляр, принимайте решение о покупке.

Новой технологии защиты компьютерных сетей от атак извне, разрабатываемой ф.Intel, посвящена статья *С.Голубева* "У IT-гигантов режутся зубы" **(CHIP, №2/2004, С.32...33)**.

Основными целями защиты, на которые направлена новая технология, назаанная LaGrande, являются клиентские конфиденциальные данные и средства связи, а также значимые коммерческие транзакции. Они защищаются от программных атак на систему или сеть, а том числе от непреднамеренного раскрытия информации из-за недостаточно защищенного ПО. Недостатки программных средста слишком часто и болезненно сказыаются на пользователях. Многочисленные ошибки, дыры и недоработки способствуют увеличению недоверия к ПО. Поэтому повсеместный переход от комбинации софт+железо к комбинации железо+софт является вполне очевидным выходом.

Многие атаки направлены просто на считывание информации из памяти, выделенной приложению, и поэтому необходимо защитить приложение от злоумышленника. Защищенное на уровне аппаратной поддержки исполнение по технологии LaGrande базируется на разделении доменов и не позволяет злоумышленнику получить доступ к ресурсам приложения.

Реализуется это асе в так называемой "кирпичной стене" (brick wall), где происходит осуществляемое аппаратными средствами разделение исполнения приложений, страниц памяти и устройств.

Для этого используется модуль TPM (Trusted Platform Module). TPM представляет собой некое аттестационное устройство, функционально аыполненное в виде генератора случайных чисел, совмещенного с хранилищем для идентификации. Этот же модуль обеспечивает уникальность сообщений, надежность хранения, а также не требующее питания опечатанное хранение.

В целом модуль TPM разработан с целью сохранения конфиденциальности информации пользователя. Вся информация о способе создания "кирпичной стены" хранится именно в нем. Модуль TPM позволяет хранить в защищенной области жесткого диска наиболее важные документы, и будет устанавливаться только на материнские платы Intel, предназначенные для поставок системным интеграторам и производителям брендовых систем. Следовательно, владельцам домашних систем небрендовой сборки придется искать для своей информации защиту иного рода. Впрочем, заинтересованные компании наверняка предстаяят массовые решения для этого сектора рынка.

Опечатанное хранение представляет собой сочетание идентификации и шифрования. Опечатанные данные доступны только в том случае, когда TPM дает добро. Технология гарантирует, что данные доступны только заведомо определенной среде, иными словами, опечатывание данных в "кирпичной стене" предполагает, что они будут доступны только той же "кирпичной стене". Изменение ее параметров изменяет идентификацию и приводит к недоступности данных.

Доверительный канал — безопасный канал между двумя комбинациями вычислительных устройств: "кирпичной стеной" и обычным устройством ввода-вывода, например, клавиатурой или монитором. Доверительный ввод создает канал между клавиатурой и устройством управления клавиатурой, а платформа LaGrande будет генерировать аппаратные ловушки, необходимые для создания доверительного канала. Это же касается и доверительного выаода, где создается канал между графическим управляющим устройством и видеоадаптером.

Предполагается, что при помощи новой технологии Intel будет гарантировать наиболее устойчивую из ранее существовавших защиту данных без ущерба для легкости использования и управляемости. Стоит отметить, что при этом гарантируется и персональная конфиденциальность, производительность системы, ее универсальность и обратная совместимость. Корпорация заявила, что реализует данную технологию защиты уже в следующем за Pentium 4 поколении процессоров, известном сейчас под названием Prescott. До реализации LaGrande в железе, если верить ее разработчикам, осталось не более двух...трех лет.

Основы OpenGL. Минимальное приложение

FatMoon /HI-TECH,
www: http://wasm.ru/

Сложно ли создать приложение под OpenGL? Да ничуть! И сейчас мы в этом убедимся. Вообще говоря, желание что-либо нарисовать на экране монитора вполне естественно, и библиотека opengl32.dll предоставляет один из самых простых способов это сделать. Для начала надо создать окно, потом сделать его контекстом вывода, потом вывести что-нибудь. На словах — это очень просто, впрочем, на деле тоже :).

1. Создание окна

Создадим обычное оконное приложение на ассемблере. Можно взять третий туториал Iczelion'a, например, с сайта wasm.ru или из пакета masm32 с этого же сайта. Да, именно так и делаются оконные приложения:

```
; ««««««««««««««««««««««««««««««««««««««««««««««««««««««««««
    .486
    .model flat, stdcall
    option casemap :none
;Это стандартное начало любой программы на ассемблере для
;компилятора masm32

    include \masm32\include\windows.inc
    include \masm32\include\gdi32.inc
    include \masm32\include\user32.inc
    include \masm32\include\kernel32.inc
    include \masm32\include\opengl32.inc
;В перечисленных файлах присутствуют почти все необходимые
;константы и объявления процедур. Если какие-то константы
;не определены, то нам придется выдрать их из заголовочных
;файлов к C++, а именно из gl.h, glu.h и, возможно,
;из wingdi.h. Я думаю, проблем с добавлением отсутствующих
;констант у читателя не возникнет

    includelib \masm32\lib\gdi32.lib
    includelib \masm32\lib\user32.lib
    includelib \masm32\lib\kernel32.lib
    includelib \masm32\lib\opengl32.lib
;А здесь все необходимые для компоновки библиотеки.
;В дальнейшем может понадобиться подключить библиотеку
;glu32.dll - это делается точно так же

WS_MY_OPENGL equ
WS_MINIMIZEBOX+WS_SYSMENU+WS_CLIPCHILDREN+WS_CLIPSIBLINGS+WS_VISIBLE

;Свойства окна - просто комбинация основных свойств. Ничто не
;мешает нам придумать что-то свое.Назовем его как хотим,
;определим все, нам необходимое, и используем это в дальнейшем.
;Окно, в которое будет выводиться что-либо с помощью OpenGL,
;должно иметь свойства WS_CLIPCHILDREN+WS_CLIPSIBLINGS. Это
;связано с правильным обновлением окна. Кроме того, хочется
;видеть системное меню и хотя бы кнопки, позволяющие закрыть окно
;и свернуть его - поэтому добавляем WS_MINIMIZEBOX+WS_SYSMENU.
;Для того чтобы сделать окно видимым, есть несколько способов,
;например, функция ShowWindow.
;Для уменьшения программы я использую WS_VISIBLE, и получаю окно,
;которое будет видно сразу после создания, без дополнительных
;телодвижений.

; ««««««««««««««««««««««««««««««««««««««««««««««««««««««««««
    .data
    szTitle db «Mini OpenGL Win32 program»,0
    szClass db «MyClass»,0
    .code
; ««««««««««««««««««««««««««««««««««««««««««««««««««««««««««
  start proc
    LOCAL wc:WNDCLASS
    LOCAL mess:MSG
;На самом деле, использование локальных переменных — весьма
;сложная задача. Но masm32 делает ее простой даже для начинающих
; ««««««««««««««««««««««««««««««««««««««««««««««««««««««««««

;Первое, что необходимо сделать - заполнить структуру WNDCLASS
;(не стоит искать глубокий смысл в значениях полей этой
;структуры - все достаточно очевидно).
        invoke GetModuleHandle, NULL
        mov [wc.hInstance], eax
;Используем стандартную системную иконку.
        invoke LoadIcon, NULL, IDI_APPLICATION
        mov [wc.hIcon], eax
;...и разрешим стандартный курсор.
        invoke LoadCursor, NULL, IDC_ARROW
        mov [wc.hCursor], eax
        mov [wc.style], CS_HREDRAW+CS_VREDRAW
;В принципе, здесь могло стоять NULL...
        mov [wc.lpfnWndProc], offset WndProc
        mov [wc.cbClsExtra], NULL
        mov [wc.cbWndExtra], NULL
        mov [wc.hbrBackground], COLOR_WINDOW
;Здесь тоже могло стоять NULL. Задание кисти для фона нужно
;в единственном случае - если обновление окна мы собираемся
;доверить системе.
        mov [wc.lpszMenuName], NULL
        mov [wc.lpszClassName], offset szClass
;Зарегистрируем новый класс окна.
        invoke RegisterClass, addr wc
;Теперь на основе этого класса мы можем создать свое окно.
;Окно будет иметь размеры 400 на 400 точек, и свойства,
;ранее определенные как WS_MY_OPENGL.
        invoke CreateWindowEx, NULL, offset szClass, \
             offset szTitle, WS_MY_OPENGL, 32, 32, 400, 400, \
                NULL, NULL, [wc.hInstance],NULL

;Вполне разумным шагом будет сохранить хендл созданного окна.
;Я не делаю этого только потому, что в дальнейшем не использую его ;).

   @MainLoop:
;Входим в (почти) бесконечный цикл обработки событий. Каждое
;событие (нажатие клавиши,движение мыши, минимизация, изменение
;размеров окна и т.д.) заставляет систему посылать нашей
;программе определенные сообщения. Остается только получить их
;и передать оконной процедуре.
;Если функция GetMessage вернула 0, значит, больше сообщений не
;предвидится, окно закрыли. Вот тогда-то и можно выйти.

        invoke GetMessage, addr mess, NULL,NULL,NULL
        or eax, eax
        jz @ExitLoop
        invoke TranslateMessage, addr mess
        invoke DispatchMessage, addr mess
        jmp @MainLoop
   @ExitLoop:
        invoke ExitProcess, NULL
;Мне пришлось оформить этот код как процедуру, поскольку я
;использовал локальные переменные
   start endp
; ««««««««««««««««««««««««««««««««««««««««««««««««««««««««««
```

Да, здесь чего-то не хватает. Нет оконной процедуры WndProc. Но сначала определимся, какие из сообщений нам нужны. Необходимо сделать окно контекстом вывода OpenGL — этот код инициализации вполне разумно выполнить при создании окна. При завершении работы надо выполнить обратные действия — отключить контекст OpenGL от нашего окна и разрушить окно. Также хотелось бы что-то вывести, иначе мы увидим просто обычный прямоугольник. Если читатель еще не догадался сам, скажу прямо — надо обработать сообщения WM_CREATE, WM_DESTROY и WM_PAINT. Сейчас этим и займемся.

2. Инициализация окна как устройства вывода для OpenGL

При создании окна, оно получает сообщение WM_CREATE, при закрытии — WM_DESTROY. Собственно, это минимум событий, которые должны быть обработаны. Причем сейчас будет один из главных моментов — надо сделать наше окно контекстом воспроизведения OpenGL. Вполне логично поместить процедуру инициализации в обработку создания окна, и более того, ни в какое другое место ее помещать не рекомендуется! Любое другое событие может быть вызвано

неоднократно, сообщение WM_DESTROY должно закрывать окно, и мы не успеем ничего увидеть. А вне оконной процедуры код инициализации совершенно не хочет работать :) !

Каждому окну OpenGL должен быть поставлен в соответствие некий формат пикселов. На самом деле он уже есть, и не один. В системе существует несколько форматов пикселов, и наша задача — выбрать наиболее подходящий. Для этой цели служат функции ChoosePixelFormat и SetPixelFormat. А сам формат пиксела запрашивается с помощью структуры PIXELFORMATDESCRIPTOR. Эта структура содержит много полей, нам же потребуются лишь некоторые, поэтому остальные надо заполнить нулями.

```
PIXELFORMATDESCRIPTOR struc
   nSize            dw  ?  ;Размер структуры,
                           ;sizeof PIXELFORMATDESCRIPTOR
   nVersion         dw  ?  ;Номер версии. Требует установки в 1
   dwFlags          dd  ?  ;Флаги. Очень важное поле, см. ниже
   iPixelType       db  ?  ;Режим отображения цвета
   cColorBits       db  ?  ;Число битовых плоскостей
                           ;в каждом буфере
   cRedBits         db  ?  ;Число битов красного
   cRedShift        db  ?  ;Смещение красного от начала
   cGreenBits       db  ?  ;Число битов зеленого
   cGreenShift      db  ?  ;Смещение зеленого
   cBlueBits        db  ?  ;Число битов синего
   cBlueShift       db  ?  ;Смещение синего
   cAlphaBits       db  ?  ;Число битов альфа - не
                           ;поддерживается
   cAlphaShift      db  ?  ;Смещение альфа - не поддерживается
   cAccumBits       db  ?  ;Общее число битовых плоскостей
                           ;в аккумуляторе
   cAccumRedBits    db  ?
   cAccumGreenBits  db  ?
   cAccumBlueBits   db  ?
   cAccumAlphaBits  db  ?
   cDepthBits       db  ?  ;Размер буфера глубины
   cStencilBits     db  ?  ;Размер буфера трафарета
   cAuxBuffers      db  ?  ;Число вспомогательных буферов - не
                           ;поддерживается
   iLayerType       db  ?  ;Вообще игнорируется
   bReserved        db  ?  ;Число плоскостей переднего и заднего
                           ;плана
   dwLayerMask      dd  ?  ;Вообще игнорируется
   dwVisibleMask    dd  ?  ;Цвет прозрачности нижней плоскости
   dwDamageMask     dd  ?  ;Вообще игнорируется
PIXELFORMATDESCRIPTOR ends
```

От устанавливаемых флагов зависит многое. Возможные значения:

```
PFD_DOUBLEBUFFER            equ  00000001h
PFD_STEREO                  equ  00000002h
PFD_DRAW_TO_WINDOW          equ  00000004h
PFD_DRAW_TO_BITMAP          equ  00000008h
PFD_SUPPORT_GDI             equ  00000010h
PFD_SUPPORT_OPENGL          equ  00000020h
PFD_GENERIC_FORMAT          equ  00000040h
PFD_NEED_PALETTE            equ  00000080h
PFD_NEED_SYSTEM_PALETTE     equ  00000100h
PFD_SWAP_EXCHANGE           equ  00000200h
PFD_SWAP_COPY               equ  00000400h
PFD_SWAP_LAYER_BUFFERS      equ  00000800h
PFD_GENERIC_ACCELERATED     equ  00001000h
PFD_SUPPORT_DIRECTDRAW      equ  00002000h

;PIXELFORMATDESCRIPTOR flags for use in ChoosePixelFormat only
PFD_DEPTH_DONTCARE          equ  20000000h
PFD_DOUBLEBUFFER_DONTCARE   equ  40000000h
PFD_STEREO_DONTCARE         equ  80000000h
```

Представляется разумной комбинация **PFD_DRAW_TO_WINDOW+PFD_SUPPORT_OPENGL**. Это означает, что мы собираемся что-то выводить в окно, используя функции OpenGL. Флаг PFD_DOUBLEBUFFER, как можно догадаться по названию, используется для двойного буфера. Его используют многие демонстрационные программы. Единственное отличие дальнейшего программирования при установленном PFD_DOUBLEBUFFER — необходимость переключать буфер на окно функцией SwapBuffers.

Установка типа пикселов дает нам либо возможность задавать все компоненты цвета, либо использовать индексные цвета. Если мы выбрали все компоненты цвета, то в дальнейшем сможем задавать цвет с помощью функций glColor*. Иначе придется работать с индексным цветом, устанавливая его через glIndex*. Допустимые значения:

```
PFD_TYPE_RGBA          equ  0
PFD_TYPE_COLORINDEX    equ  1
```

Также мельком отмечу, что если мы используем индексный цвет, то надо задать количество битов на каждый канал, а если RGBA — то надо запросить количество битов на пиксел.

Вот что из структуры нужно заполнить:

```
nSize        = sizeof PIXELFORMATDESCRIPTOR
nVersion     = 1
dwFlags      = PFD_DRAW_TO_WINDOW+PFD_SUPPORT_OPENGL
iPixelType   = PFD_TYPE_RGBA
cColorBits   = 16
```

Остальные поля можно установить в 0. Я запрашиваю 16-битный цвет, поскольку моя видеокарта не тянет большего. Но ничто не запрещает задать здесь хоть 64. Другое дело, что столько нам никто не даст :)). Дело в том, что задать тип пикселов невозможно. Установка нужного нам формата заключается в выборе из существующих, наиболее близких к требуемому. Неясно? Если вы запросите 64-битный цвет, а ваша видеокарта с трудом выдает 256, то рисовать вы все равно будете в 256-цветном окне. В системе всегда присутствует несколько форматов пикселов для цвета RGBA и для индексного, с двойным буфером и без него, и обычно — для каждого из поддерживаемых картой режимов. Выбор осуществляется, как можно догадаться, функцией

```
invoke ChoosePixelFormat, hdc, addr pixfd
```

Эта функция возвращает номер формата, наиболее подходящего под заполненную структуру и устройство. После этого остается только назначить нашему окну именно этот формат:

```
invoke SetPixelFormat, hdc, format, addr pixfd
```

Теперь осталось создать в окне контекст воспроизведения OpenGL и сделать его текущим. После этого назначить порт просмотра размером во все окно — и можно рисовать.

Всякий раз, когда окно должно быть перерисовано, оно получает сообщение WM_PAINT. И именно в обработку этого события мы и поместим код, который что-то будет выводить на экран.

При закрытии окна оно должно вызвать функцию PostQuitMessage — это требование Windows. А для правильного закрытия устройства вывода надо сделать текущим нулевой контекст, удалить наш, освободить окно и только потом выйти. Ниже приводится листинг оконной процедуры:

```
   .data?
;Поскольку локальные переменные теряются после выхода из
;процедуры, используем глобальные.
    hdc  dd ?
    hglc dd ?
   .code
; «««««««««««««««««««««««««««««««««««««««««««««««««««««««««
  WndProc proc hWnd:DWORD, uMsg:DWORD, wParam:DWORD, lParam:DWORD
   LOCAL pixfd:PIXELFORMATDESCRIPTOR
    LOCAL rc:RECT
;Использование локальных переменных в оконных процедурах -
;НЕ САМЫЙ РАЗУМНЫЙ поступок, однако я буду использовать эти
;переменные только один раз - при инициализации.
    cmp uMsg, WM_DESTROY
    jz @Destroy
    cmp uMsg, WM_CREATE
    jz @Create
    cmp uMsg, WM_PAINT
    jz @Paint
;Все сообщения, которые мы не обрабатываем,надо передать системе
    invoke DefWindowProc,hWnd,uMsg,wParam,lParam
    ret
;==================
;=== Уничтожение ==
;==================
  @Destroy:
;Делаем текущим нулевой контекст, чтобы можно было удалить наш.
```

```
    invoke wglMakeCurrent,NULL,NULL
;Отключаем наш. Если бы я не сохранил хендл контекста в
;глобальной переменной, мы бы могли сейчас получить его с
;помощью wglGetCurrentContext.
    invoke wglDeleteContext,[hglc]
;Освобождаем контекст устройства,перед этим мы отключили от него
;OpenGL. Теперь же отключаем и системный графический интерфейс
    invoke ReleaseDC,hWnd,[hdc]
;и посылаем системе сообщение о выходе
    invoke PostQuitMessage, NULL
;================
;== Завершение ==
;================
  @Finish:
;Все события, которые мы обрабатываем, должны вернуть 0
    xor eax,eax
    ret
;====================
;== Инициализация ==
;====================
  @Create:
;Получим контекст устройства, соответствующий нашему окну
    invoke GetDC,[hWnd]
    mov [hdc],eax
;Заполним структуру нулями. Длина структуры - 36 байтов,
;поэтому следующий фрагмент должен работать вполне корректно.
    lea edi, pixfd
    mov ecx, sizeof PIXELFORMATDESCRIPTOR shr 2
    xor eax,eax
    rep stosd
;Странно, но одно из полей структуры - ее размер
    mov [pixfd.nSize], sizeof PIXELFORMATDESCRIPTOR
;Номер версии должен быть всегда 1. Вам кажется, что эти два поля
;совершенно лишние? Мне тоже так кажется, но заполнять их придется. :)
    mov [pixfd.nVersion],1
;Устанавливаем флаги
    mov [pixfd.dwFlags],PFD_SUPPORT_OPENGL+PFD_DRAW_TO_WINDOW
;и тип цветового режима
    mov [pixfd.iPixelType],PFD_TYPE_RGBA
;Глубина цвета - 16 битов на пиксел.
    mov [pixfd.cColorBits],16
;Выбираем из существующих форматов наиболее подходящий.
    invoke ChoosePixelFormat,[hdc],addr pixfd
;Приходится где-то сохранить этот формат, поскольку иначе
;eax будет уничтожен при использовании addr.
    mov ecx,eax
;Устанавливаем формат пикселов.
    invoke SetPixelFormat,[hdc],ecx,addr pixfd
;Создаем новый контекст воспроизведения в нашем окне
    invoke wglCreateContext,[hdc]
;И делаем его текущим.
    mov [hglc], eax
    invoke wglMakeCurrent,[hdc],eax
    invoke GetClientRect,[hWnd],addr rc
;Назначаем область просмотра размером во все окно.
    invoke glViewport,0,0,[rc.right],[rc.bottom]
    jmp @Finish
;====================
;== Вывод в окно ==
;====================
  @Paint:
;То, что идет ниже, пока объясняется поверхностно - подробное
;описание я приведу позже.
    invoke glClear,GL_COLOR_BUFFER_BIT
;Можете считать, что только что сделали CLS. Собственно,
именно
;это мы и сделали.
    invoke glBegin,0
;Сообщили OpenGL, что мы собираемся что-то выводить.
    invoke glVertex2f,0,0
;Вывели точку по координатам 0,0
    invoke glEnd
;Сообщили OpenGL, что мы уже вывели все, что хотели.
    invoke glFlush
;И хотим отправить все на экран. И побыстрее.
    jmp @Finish
  WndProc endp
; «««««««««««««««««««««««««««««««««««««««««««««««««««««««««««««
 end start
```

Используемая константа легко находится в заголовочном файле из Си:

```
GL_COLOR_BUFFER_BIT    equ    00004000h
```

Если все скомпилировано без ошибок, то мы должны увидеть ма-аленькую белую точку в центре черного квадрата. Малевич отдыхает!

Исходный текст и исполняемый файл примера создания минимального приложения — http://radio-mir/com/files/wk_2004/wk06/sample1.zip

А.ИВАНЧИКОВ,
E-mail: iaa@rbcmail.ru

COM — это не сон

(Продолжение. Начало в №№4-5/2004)

Реализация IUnknown

При помощи функции QueryInterface клиент определяет, поддерживается ли компонентом тот или иной интерфейс. Если компонент поддерживает некоторый интерфейс, то QueryInterface возвращает указатель на этот интерфейс, в противном случае QueryInterface возвращает код ошибки (в этом случае клиент может запросить указатель на другой интерфейс или выгрузить компонент). У QueryInterface два параметра:

```
virtual HRESULT __stdcall QueryInterface(
        REFIID iid,
        void** ppvObject);
```

Первый параметр — идентификатор интерфейса, второй — адрес, по которому QueryInterface помещает указатель на искомый интерфейс. QueryInterface может возвратить либо S_OK, либо E_NOINTERFACE.

Следующий фрагмент кода реализует QueryInterface для класса CCardsService из приведенного выше исходного кода файла CardsService.cpp.

```
HRESULT __stdcall CCardsService::QueryInterface(
    REFIID iid,
    void** ppvObject)
{
    if ( !ppvObject )
        return E_INVALIDARG;

    *ppvObject = NULL;

    // запрос интерфейса IUnknown
    if ( ::IsEqualIID(iid, IID_IUnknown) )
        *ppvObject =
            static_cast<ICardsService*>(this);

    // запрос интерфейса ICardsService
    else if (::IsEqualIID(iid, IID_CardsService))
        *ppvObject =
            static_cast<ICardsService*>(this);

    // запрашиваемый интерфейс не поддерживается
    else
        return E_NOINTERFACE;

    // увеличение счетчика ссылок на единицу
    static_cast<IUnknown*>(*ppvObject)->AddRef();
    return S_OK;
}
```

В начале метода указатель на возвращаемый интерфейс устанавливается в NULL. Это значение будет возвращено для интерфейса, который компонентом не поддерживается. Возврат из функции инициализированного NULL-указателя в случае возникновения ошибки в работе метода является не только требованием COM, придерживаться данного правила полезно вообще. NULL вызовет фатальную ошибку в клиентах, которые не проверяют возвращаемые значения. Но это менее опасно, чем позволить клиенту выполнять произвольный код, содержащийся по неинициализированному указателю.

Для сравнения идентификаторов интерфейсов на равенство используется макрос IsEqualIID, что синонимично IsEqualGUID.

В зависимости от запрашиваемого интерфейса выполняет-

ся необходимое приведение указателя this к определенному типу, прежде чем он сохранится в *ppvObject. Если для компонента, реализующего один интерфейс, это и не важно, т.е. приведение типов в QueryInterface такого компонента может и отсутствовать, то для компонента, реализующего более чем один интерфейс, данное требование принципиально. Для пояснения ситуации рассмотрим следующий простой пример.

```
#include <stdio.h>
struct I
{
    virtual void f() = 0;
);
class I1 : public I
{
];
class I2 : public I
{
);
class CA : public I1,
           public I2
{
    void f() { };
};
void main(void)
{
    I* pI1,
     * pI2;
    CA* pA = new CA;

    pI1 = static_cast<I1*>(pA);
    pI2 = static_cast<I2*>(pA);
    printf("pA =   %p\npI1 =  %p\npI2 =  %p\n",
           pA, pI1, pI2);
    delete pA;
)
```

Результат выполнения данной программы:

```
pA =   002F0FE0
pI1 =  002F0FE0
pI2 =  002F0FE4
```

Разумеется, конкретные значения pI1 и pI2 при разных запусках могут меняться, но разность между ними всегда будет равна 4 (размер указателя (I1*)pA).

Из **рис.3**, иллюстрирующего этот пример, видно, что указатель на CA указывает на указатель таблицы виртуальных функций I1, так как I1 — первый интерфейс в списке наследования. Таким образом, можно без приведения типов использовать указатель на CA вместо указателя на I1. Однако очевидно, что указатель на CA не указывает на vtbl I2, как раз в этом случае его необходимо привести к нужному типу. Чтобы не держать в голове список наследования, достаточно помнить, что перед возвратом из QueryInterface всегда необходимо приводить указатель this к требуемому типу.

Выражение

```
pI1 = static_cast<I1*>(pA);
```

является синонимичным выражению

```
pI1 = (I1*)pA;
```

т.е. оператор static_cast<>() является новым стилем для привычного приведения статических типов.

Все реализации QueryInterface должны следовать определенным правилам [1]. Если их выполнять, клиент сможет узнать о компоненте достаточно, чтобы управлять им и использовать его полноценно. Без соблюдения этих правил поведение QueryInterface было бы неопределенным, и использовать такие компоненты было бы невозможно.

Вот эти правила:

- QueryInterface всегда возвращает один и тот же IUnknown. У данного экземпляра компонента есть только один интерфейс IUnknown. Всегда, когда у компонента запрашивается IUnknown (неважно через какой интерфейс), в ответ получается всегда одно и то же значение указателя. Таким образом, запросив у каждого из используемых интерфейсов IUnknown и сравнив результаты, можно определить, указывают ли эти интерфейсы на один компонент, или нет;

- интерфейс можно получить вновь, если можно было получить его ранее. Если QueryInterface однажды успешно отработала запрос на некоторый интерфейс, то все последующие запросы для того же компонента также будут успешными. Если же запрос был неудачным, то для этого интерфейса QueryInterface всегда будет возвращать ошибку. Это правило относится только к конкретному экземпляру компонента. Ко вновь созданному экземпляру оно неприменимо;

- можно снова получить интерфейс, который уже имеется. Если имеется интерфейс I1, то через него можно запросить интерфейс I1, и получить в ответ указатель на I1;

- всегда можно вернуться туда, откуда начали. Если имеется указатель на интерфейс I1, и с его помощью можно получить интерфейс I2, то можно выполнить и обратную операцию, получив интерфейс I1 по указателю на I2. Иными словами, независимо от того, какой интерфейс имеется в данный момент, можно снова получить интерфейс, с которого все началось;

- если можно попасть куда-то хоть откуда-нибудь, то можно попасть туда откуда угодно. Если можно получить у компонента некоторый интерфейс, то его можно получить с помощью любого из интерфейсов, поддерживаемых компонентом. Если можно получить I2 через I1, а I3 через I2, то I3 можно получить и через I1.

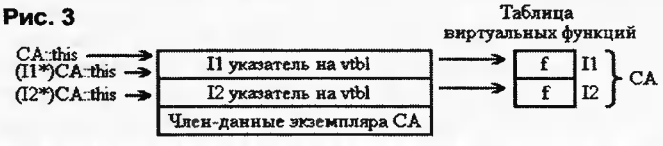
Общая задача всех приведенных правил — сделать использование QueryInterface простым, логичным, последовательным и определенным. По счастью, при реализации QueryInterface для компонента следовать правилам нетрудно.

QueryInterface — это одна из самых важных составляющих COM, поскольку она определяет компонент. Интерфейсы, поддерживаемые компонентом — это те интерфейсы, указатели на которые возвращает QueryInterface. Их определяет QueryInterface, а не заголовочный файл для класса C++, реализующего компонент. Компонент не определяется и иерархией наследования этого класса C++. Его определяет исключительно реализация QueryInterface.

Перед завершением работы QueryInterface увеличивает на единицу счетчик ссылок на интерфейс следующим вызовом:

```
static_cast<IUnknown*>(*ppvObject)->AddRef();
```

Рис. 3



Подсчет ссылок

Подсчет ссылок напрямую связан с управлением временем жизни компонента. Стратегия COM в этом вопросе такова — вместо того чтобы удалять компоненты напрямую, компоненту сообщается, что работа с интерфейсом завершена. Точно известно, когда интерфейс начинает использоваться, и в большинстве случаев известно, когда интерфейс следует прекратить использовать. Однако, чаще всего невозможно указать момент, когда завершено использование компонента вообще. Поэтому имеет смысл ограничиться сообщением об окончании работы с данным интерфейсом — и пусть компонент сам отслеживает, когда клиенты перестают использовать все его интерфейсы.

Именно для реализации этой стратегии и предназначены еще две функции-члена IUnknown — AddRef и Release. AddRef и Release реализуют технику управления памятью, известную как подсчет ссылок. Когда клиент получает некоторый интерфейс, значение некоторого внутреннего счетчика компонента увеличивается на единицу. Когда клиент заканчивает работу с интерфейсом, значение счетчика уменьшается на единицу. Когда оно доходит до нуля, компонент удаляет себя из памяти. Клиент также увеличивает счетчик ссылок, когда создает

новую ссылку на уже имеющийся у него интерфейс. Увеличивается счетчик вызовом AddRef, а уменьшается — вызовом Release. Для того чтобы пользоваться подсчетом ссылок, необходимо соблюдать лишь три простых правила:

- функции, возвращающие интерфейсы, перед возвратом всегда должны вызывать AddRef для соответствующего указателя. Таким образом, нет необходимости вызывать AddRef в клиенте после получения (от функции) указателя на интерфейс;
- когда работа с интерфейсом завершена, для него необходимо вызвать Release;
- когда один указатель на интерфейс присваивается другому, следует вызывать AddRef.

Реализация AddRef и Release относительно проста. AddRef увеличивает значение переменной m_IReferance — счетчика ссылок. Release уменьшает m_IReferance и удаляет компонент, если значение переменной становится равным нулю. При реализации этих функций, как правило, используются функции API Win32 InterlockedIncrement и InterlockedDecrement, которые гарантируют, что значение переменной изменяет в каждый момент времени только один поток управления.

Описание/определение GUID

Для того чтобы при сборке компонента GUID интерфейса и CLSID компонента были определены, необходимо в будущем проекте иметь файл CardsGuids.cpp следующего содержания:

```
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//  CardsGuids.cpp
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//-------------------------------------
#include <objbase.h>
#include <initguid.h>
#include "ICardsService.h"
```

В файле ICardsService.h GUID и CLSID описаны. В файле CardsGuids.cpp, благодаря заголовочному файлу initguid.h, включенному после objbase.h, GUID и CLSID определяются. Все это успешно работает, так как макрос

```
DEFINE_GUID(IID_CardsService,
0xf345c351, 0x28fd, 0x4ee3, 0xab, 0xfa, 0x9d,
0xbf, 0x9b, 0xe9, 0xa3, 0x8a);
```

при отсутствии файла initguid.h раскрывается в выражение:

```
EXTERN_C const GUID FAR IID_CardsService;
```

что является описанием, а при наличии файла initguid.h, макрос DEFINE_GUID раскрывается в выражение:

```
EXTERN_C const GUID IID_CardsService =
{0xf345c351, 0x28fd, 0x4ee3,
{0xab, 0xfa, 0x9d, 0xbf, 0x9b, 0xe9, 0xa3, 0x8a }};
```

что является определением.

Фабрика классов

Создание объектов — очень важная операция в любой объектно-ориентированной системе. Прежде чем использовать объекты, их необходимо создать. Если каждый объект создается по-своему, то будет трудно использовать разные объекты полиморфно. Следовательно, желательно, чтобы создание объектов было как можно более гибким, а все компоненты, в свою очередь, создавались сходным образом. Решение проблемы заключается в создании унифицированных компонентов, которые предназначены для порождения необходимых компонентов. Эти унифицированные компоненты называются фабриками классов. Фабрика класса — это компонент, единственной задачей которого является создание других компонентов. Конкретная фабрика класса создает компоненты, соответствующие только одному CLSID. Клиент использует поддерживаемые фабрикой класса интерфейсы для управления процессом создания каждого компонента. Стандартный интерфейс создания компонентов — IClassFactory. Самый простой способ создания компонентов — при помощи функции CoCreateInstance. Объявление CoCreateInstance имеет вид:

```
HRESULT __stdcall CoCreateInstance(
    REFCLSID rclsid,
    LPUNKNOWN pUnkOuter,
    DWORD dwClsContext,
    REFIID riid,
    LPVOID* ppv);
```

У функции четыре входных параметра и один выходной. Первый параметр — CLSID создаваемого компонента. Второй параметр используется для агрегирования компонента (тема агрегирования компонентов, равно как и многие иные приемы программирования, выходит за рамки данной статьи). Третий параметр — dwClsContext — ограничивает контекст исполнения компонента, с которым данный клиент может работать, т.е. управляет тем контекстом, где компонент может исполняться: в том же процессе, что и клиент, в другом процессе или на другой машине. Значение параметра может быть выражено комбинацией флагов, перечисленных в **табл.2**.

Четвертый параметр функции CoCreateInstance — riid — является IID интерфейса, который планируется использовать для работы с компонентом. Указатель на этот интерфейс возвращается через последний параметр — ppv.

Компоненты, порождаемые CoCreateInstance, наследуются от интерфейса IClassFactory.

Ниже приведен заголовочный файл фабрики класса для создания компонента CCardsService.

```
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//  CardsFactory.h
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//-----------------------------------------------

#ifndef __CARDSFACTORY_H__
#define __CARDSFACTORY_H__

#include "ICardsService.h"
#include "CardsInside.h"

//-----------------------------------------------
class CCardsFactory : public IClassFactory
{
public:
    // определение IClassFactory
    virtual HRESULT __stdcall CreateInstance(
        IUnknown* pIUnknown,
        REFIID    iid,
        void**    ppvObject);

    virtual HRESULT __stdcall LockServer(
        BOOL      bLock);

    // определение IUnknown
    virtual HRESULT __stdcall QueryInterface(
        REFIID    iid,
```

Табл. 2

Флаг	Описание
CLSCTX_INPROC_SERVER	Клиент принимает только компоненты, которые исполняются в одном с ним процессе. Подобные компоненты должны быть реализованы в DLL
CLSCTX_INPROC_HANDLER	Клиент будет работать с обработчиками в процессе. Обработчик в процессе — это компонент внутри процесса, который реализует только часть компонента. Другие части реализуются компонентом вне процесса — локальным или удаленным сервером
CLSCTX_LOCAL_SERVER	Клиент будет работать с компонентами, которые выполняются в другом процессе, но на той же самой машине. Локальные серверы реализуются в .EXE
CLSCTX_REMOTE_SERVER	Клиент допускает компоненты, выполняющиеся на другой машине. Использование данного флага требует задействования DCOM (Distributed COM)


```
        void**    ppvObject);
    virtual ULONG __stdcall AddRef() ;
    virtual ULONG __stdcall Release();

public:
    CCardsFactory();
    ~CCardsFactory();

private:
    long         m_lReferance;
);

#endif //__CARDSFACTORY_H__
```

Как указывалось выше, стандартный интерфейс, который поддерживают фабрики класса для создания компонентов — это IClassFactory. Объявление данного интерфейса приводится в файле unknwn.h:

```
IClassFactory : public IUnknown
[
public:
    virtual HRESULT STDMETHODCALLTYPE CreateInstance(
        IUnknown *pUnkOuter,
        REFIID riid,
        void **ppvObject) = 0;
    virtual HRESULT STDMETHODCALLTYPE LockServer(
        BOOL fLock) = 0;
  };
```

В IClassFactory описаны две чисто виртуальные функции.

Функция CreateInstance возвращает указатель на интерфейс компонента с одновременным созданием последнего.

Метод LockServer предназначен для управления удержанием компонента в памяти до завершения работы с ним. Реализуется LockServer простым увеличением и уменьшением счетчика блокировок.

Фабрики класса в большинстве своем характеризуются следующим:

- данный экземпляр фабрики класса создает компоненты, соответствующие только одному CLSID. Это очевидно, поскольку CoCreateInstance имеет в качестве параметра CLSID, а CreateInstance — нет;

- фабрика класса для данного CLSID создается тем же разработчиком, который реализует соответствующий компонент. Компонент-фабрика класса чаще всего содержится в той же DLL, что и создаваемый компонент.

Конкретный экземпляр фабрики класса соответствует одному конкретному CLSID, и как фабрика, так и создаваемый ею компонент реализуются одним и тем же программистом. Поэтому фабрика класса может иметь и имеет специфические знания о создаваемом компоненте. Реализация фабрики класса с использованием специфической информации о создаваемом компоненте — это преимущество. Задача фабрики класса состоит в том, чтобы знать, как создается компонент, и инкапсулировать это знание так, чтобы максимально изолировать клиента от требований компонента.

Реализация фабрики класса компонента CCardsService, приведенная в файле CardsFactory.cpp, не должна вызвать сложности при анализе.

```
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//  CardsFactory.cpp
//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//~\\~//
//-------------------------------------------------

#include "CardsFactory.h"
#include "CardsService.h"

// счетчик активных блокировок
extern long g_lServerLocks;

// конструктор и деструктор фабрики класса
//-------------------------------------------
CCardsFactory::CCardsFactory() : m_lReferance(1)
{
}

//-------------------------------------------
CCardsFactory::~CCardsFactory()
{
}

// реализация IUnknown
//-------------------------------------------
HRESULT __stdcall CCardsFactory::QueryInterface(
    REFIID iid,
    void** ppvObject)
{
    if ( !ppvObject )
        return E_INVALIDARG;

    *ppvObject = NULL;

    // запрос интерфейса IUnknown или IClassFactory
    if ( ::IsEqualIID(iid, IID_IUnknown) ||
         ::IsEqualIID(iid, IID_IClassFactory) )
    (
        *ppvObject =
            static_cast<IClassFactory*>(this);
    }

    // запрашиваемый интерфейс не поддерживается
    else
        return E_NOINTERFACE;

    // необходимо увеличить значение ссылки
    static_cast<IUnknown*>(*ppvObject)->AddRef();
    return S_OK;
}

//-------------------------------------------------
ULONG __stdcall CCardsFactory::AddRef()
[
    return ::InterlockedIncrement(&m_lReferance);
}

//-------------------------------------------------
ULONG __stdcall CCardsFactory::Release()
(
    if ( !::InterlockedDecrement(&m_lReferance) )
       {
        delete this;
              return 0;
       }
    return m_lReferance;
)

// реализация компонента фабрики класса
//-------------------------------------------------
HRESULT __stdcall CCardsFactory::CreateInstance(
    IUnknown* pIUnknown,
    REFIID    iid,
    void**    ppvObject)
{
    // агрегирование не поддерживается
    if ( pIUnknown )
        return CLASS_E_NOAGGREGATION;

    // создать компонент
    ICardsService* pCardsService =
        new CCardsService;
    if ( !pCardsService )
        return E_OUTOFMEMORY;

    HRESULT hr = pCardsService->QueryInterface(
        iid, ppvObject);
    pCardsService->Release();

    return hr;
}

//-------------------------------------------------
HRESULT __stdcall CCardsFactory::LockServer(
    BOOL bLock)
(
    if ( bLock )
        ::InterlockedIncrement(&::g_lServerLocks);
    else
        ::InterlockedDecrement(&::g_lServerLocks);

    return S_OK;
}
```

(Продолжение следует)

ИССЛЕДОВАНИЕ ПРОГРАММ

CyberManiac /HI-TECH,
www: http://wasm.ru/

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ КРЭКИНГА

(Продолжение. Начало в №№8-12/2003, 1-5/2004)

Практически в любом nag screen'е изначально заложен способ от него избавиться. Этим способом может быть клик по кнопке в диалоге, нажатие "горячей клавиши" или же истечение определенного времени с момента появления окна. Все эти случаи объединяет одно — nag screen ждет некоего события, наступление которого станет для него сигналом к исчезновению. "Приемник" этого сигнала, который и выполняет все действия по убиранию nag screen'а с экрана, чаще всего прячется во все той же оконной процедуре, о которой уже столько раз говорилось и еще не раз будет сказано. Для тех, кто не особо интересовался программированием с использованием "чистого" WinAPI, дам некоторые пояснения относительно традиционного устройства оконной процедуры.

"Сердцем" оконной процедуры, несомненно, является обработка сообщений, поступающих от окна, которая на ассемблере выглядит приблизительно так:

```
.IF uMsg==WM_DESTROY
  invoke PostQuitMessage,NULL
.ELSEIF uMsg==WM_INITDIALOG
 ...
.ELSEIF uMsg==WM_COMMAND
  mov eax,wParam
  .IF ax==ID1
   ...
  .ELSEIF ax==ID2
   ...
  .ENDIF
.ELSE
  invoke DefWindowProc,hWnd,uMsg,wParam,lParam
  ret
.ENDIF
```

Не вдаваясь в подробности (которые вы можете найти в документации и в книгах по программированию под Windows), скажу, что обычно все самое интересное — обработка нажатий на кнопки или реакция на выбор пунктов меню — скрывается в блоке, анализирующем сообщение WM_COMMAND. Внутри этого блока обычно выполняется последовательное сравнение параметров сообщения wParam и lParam с идентификаторами и хэндлами управляющих элементов (кнопок, элементов меню, тулбаров и т.п.). Если значения хэндла и идентификатора указывают, что пользователь нажал на соответствующий элемент — программа выполняет соответствующие действия. Допустим, что мы знаем, где располагается оконная процедура интересующего нас окна. Что мы можем сделать с нашей находкой? Самое простое — это, конечно, возможность менять местами функции управляющих элементов: если в приведенном выше примере поменять местами значения ID1 и ID2, соответственно изменятся и функции элементов с идентификаторами ID1 и ID2. Менее очевиден, но тоже достаточно прост для понимания тот факт, что можно "переключить" реакцию программы с одного сообщения на другое. К примеру, заменив в нашем примере WM_INITDIALOG на WM_PAINT, мы заставим процедуру инициализации выполняться не единожды при создании диалога, а при поступлении каждой команды на перерисовку содержимого окна. Зачем такое может понадобиться? С точки зрения среднего программиста, это действо совершенно нелогично, нефункционально, и даже более того — опасно. Но крэкинг — это искусство, существующее по ту сторону обычного программирования. И именно благодаря своей "потусторонней" сущности даже во внешне бессмысленных операциях крэкер способен узреть потаенные возможности и раскрыть их к своей пользе.

Итак, представим также, что кнопка ID1 закрывает nag screen (на самом деле представлять придется только вам — у меня код этого примера сейчас перед глазами). Наша задача — убрать nag screen любыми доступными средствами. Как можно решить эту задачу? Да очень просто — базовую идею я описал парой строчек выше. Перво-наперво заменим в оконной процедуре константу WM_COMMAND на WM_PAINT. Следующее, что нам нужно — это чтобы оконная процедура реагировала на сообщение WM_PAINT как на нажатие кнопки с идентификатором ID1. Добиться этого можно, подправив условие IF ax==ID1 таким образом, чтобы оно выполнялось в любом случае, независимо от величины wParam. Думаю, проделать такой фокус не составит никакого труда даже для самого начинающего крэкера, только-только научившегося заменять условный переход на безусловный. Если в оконной процедуре присутствует "настоящий" обработчик WM_PAINT, есть смысл при помощи безусловного перехода выполнить и его — ради соблюдения принципа минимального вмешательства. На этом нашу миссию можно считать завершенной — после внесенных исправлений nag screen будет закрываться сам собой сразу же после появления. Главная хитрость заключается в том, что как только nag screen пытается стать видимым (а вы когда-нибудь видели невидимые nag screen'ы?), окно этого screen'а автоматически получит команду на перерисовку содержимого клиентской области — сообщение WM_PAINT. Мы же, движимые возвышенными эстетическими чувствами, протестующими против созерцания рекламных банальностей, преобразовали это безобидное сообщение в приказ немедленно убрать раздражающее окно с экрана.

Все, о чем я говорил выше, относится к классическим средствам работы с окнами и диалогами Windows и к объектно-ориентированным "оберткам" для WinAPI. Однако фирма Borland внесла в такую консервативную область как графические интерфейсы Windows-программ заметное оживление. Я имею в виду прежде всего фирменную борландовскую разработку — Visual Component Library (она же VCL) — объектно-ориентированную библиотеку, предназначенную для создания графического интерфейса. Впервые эта библиотека появилась в Delphi 1, и с тех пор стала неотъемлемой частью всех версий Delphi и C++ Builder. В VCL широко используются такие нетрадиционные для Windows вещи как "безоконные" управляющие элементы (non-windowed controls), по сути представляющие собой картинки на экране, и потому напрямую не подчиняющиеся функциям WinAPI. Другим новшеством, внедренным Borland, является специфический формат хранения шаблонов форм — они хранятся как данные типа RCData, причем в качестве идентификаторов используются названия классов этих форм: к примеру, шаблон формы класса TMyForm хранится в ресурсах под именем TMYFORM. Ну и, наконец, VCL отличается от большинства библиотек аналогичной направленности обширной и довольно сложной иерархией классов.

Давайте возьмем какую-нибудь программу на Delphi (я взял одну из своих разработок) и попробуем повторить эксперимент с удалением шаблона какого-нибудь диалога. Открываем файл программы в Resource Hacker, выбираем любую понравившуюся форму — и видим, что Resource Hacker нещадно глючит, пытаясь перевести шаблон из внутреннего борландовского формата в более удобочитаемое **нечто**. На самом деле Resource Hacker довольно успешно понимает формат Borland'овских шаблонов — просто я использовал слишком новую версию Delphi, о которой наш редактор ресурсов, увы, ничего не знает. Кстати, если взглянуть на тот же ресурс в шестнадцатеричном редакторе, вы увидите перед собой малопонятную кучу байтов, в которую вкраплены имена компонентов и их свойств, а также текстовые значения. Но мы пока не будем пытаться дешифровать шаблон, а просто

удалим его. Удаляем, запускаем, пытаемся вызвать соответствующее окно — и получаем сообщение Resource <имя удаленного нами ресурса> not found. Да, это определенно не то, о чем мы мечтали — nag screen, конечно, исчез, но окошко с сообщением об ошибке смотрится ничуть не лучше. Увы, так просто от nag screen'ов в программах на Delphi не избавиться.

Поскольку Resource Hacker показал себя не с лучшей стороны, внесем коррективы в наш инструментарий и возьмем eXeScope 6.41 (последняя версия на момент написания данной главы) — эта программа более или менее успешно переваривает шаблоны Delphi 7 с надписями в кодировке Unicode. Если вы не хотите нарушать ничьих авторских прав хотя бы в процессе обучения, раздобудьте для экспериментов мою программку InqSoft Window Scanner — лицензионное соглашение дает вам полный карт-бланш на ее ломание в учебных целях. Распакуйте ее UPX'ом, откройте полученный исполняемый файл в eXeScope и найдите шаблон формы TAboutForm. Вы увидите что-то вроде:

```
object AboutForm: TAboutForm
  Left = 265
  Top = 185
  BorderIcons = [biSystemMenu]
  BorderStyle = bsSingle
  Caption = #1054' '#1087#1088#1086#1075#1088#1072#1084#1084#1077"...'
  ClientHeight = 192
  ClientWidth = 318
  ...
end
...
```

Если вы раньше программировали на Delphi или C++ Builder, то наверняка уже догадались, что это — текст типичного dfm-файла, в котором описан шаблон формы вместе со всеми ее компонентами, как визуальными, так и не очень. Если же вы не программировали ни на Delphi, ни на C++ Builder — вам будет непросто разобраться в ломании программ, написанных при помощи этих средств разработки. Однако в этой главе мы будем разбирать достаточно простые вещи, для понимания которых достаточно знания английского языка, основ ООП и знакомства с любым визуальным средством разработки. В конце концов, тот, кто умеет помещать управляющие элементы на шаблоны диалогов в RadAsm'е, вряд ли испытает какие-либо трудности в понимании того, как бросать точно такие же компоненты на формы в Delphi. Но в общем случае работает совершенно обратное правило: **чтобы эффективно исследовать программу, необходимо иметь представление об инструментах, с использованием которых эта программа была написана**. То есть, чтобы взломать программу, написанную на C++ Builder, нужно знать характерные особенности C++ Builder.

Одной из таких характерных особенностей, к примеру, является то, что имена классов окон (понятие "класс окна" здесь используется в том же смысле, что и в документации по Windows API) в Delphi/C++ Builder начинаются с буквы T и совпадают с именами классов форм. Проще говоря, если программа создает окно как экземпляр класса TMyForm, то имя класса окна, возвращаемое WinAPI'шной функцией GetClassName, тоже будет TMyForm. Собственно, InqSoft Window Scanner в качестве объекта для экспериментов я посоветовал не случайно — эта программа, помимо прочих функций, как раз умеет определять имена классов окон. Как вы помните, имена классов форм в программе совпадают со строковыми идентификаторами шаблонов этих форм в секции ресурсов, поэтому "подглядев" средствами WinAPI имя класса окна, вы смело можете шерстить секцию ресурсов на предмет наличия ресурса типа RCDATA с соответствующим идентификатором, и, скорее всего, вы такой ресурс обнаружите.

Строго говоря, использование механизмов наследования в Delphi/C++ Builder позволяет создавать формы, у которых имя класса окна отличается от символьного идентификатора шаблона, но а большинстве программ эта возможность не используется. Да и распознать форму-наследника по dfm-описанию родителя обычно бывает не так уж сложно, поэтому я не буду заострять анимание на этом достаточно специфическом случае.

Итак, что нам показал eXeScope? А показал он описание формы и свойства всех ее компонентов, причем в самом что ни на есть текстовом виде. Вы наверняка уже поняли, что слово **object** начинает описание объекта (или компонента, если быть до конца точным), а слово **end** это описание завершает. И что строки вроде **Left = 265**, **BorderIcons = [biSystemMenu]** или **BorderStyle = bsSingle** представляют собой не что иное как имена свойств компонентов и значения этих свойств, как они были заданы во время разработки приложения. И что самое приятное, вы можете не только смотреть на эти свойства, но и редактировать их. Исправить числовые или текстовые значения — дело нехитрое, стираем старое значение, вписываем новое, сохраняем результат, и можно любоваться эффектом. Но вот если понадобится исправить свойство вроде BorderStyle — скорее всего, придется заглянуть в документацию по Delphi, чтобы узнать, что такое bsSingle, и какие еще значения может принимать это свойство. Пока что все довольно просто, и чем-то напоминает наши эксперименты по редактированию шаблонов диалогов в Калькуляторе.

Впрочем, сказав, что изменить значение свойства — дело нехитрое, я был не совсем прав. Состояние современного инструментария таково, что если пользоваться специализированными программами (тем же eXeScope, например), то эта операция не всегда выполняется корректно. Поэтому на практике мне нередко приходилось пользоваться следующим приемом — оригинальное значение я читал при помощи eXeScope, но вот новые данные вписывал уже в шестнадцатеричном редакторе; как я определял, какие байты и где надо было менять, я думаю, в очередной раз повторяться не стоит. В самых тяжелых случаях можно даже экспортировать шаблон формы в dfm-файл, а затем попытаться при помощи той же версии Delphi/C++ Builder впихнуть этот dfm-файл в тестовый проект, откомпилировать и затем перенести шаблон формы из тестового проекта на его "родное" место в подопытной программе. При этом, разумеется, придется обеспечить совпадение версий компилятора и, если на форме имеются нестандартные компоненты, раздобыть и установить точно такие же. Как вы догадываетесь, весь этот процесс весьма хлопотный, и такая игра вряд ли стоит свеч — я подобную "хирургическую операцию" по пересадке ресурсов проделывал лишь единожды, и то в основном в порядке эксперимента. Но вот изучение тестовых примеров перед тем как начинать править свойства — вещь очень и очень полезная, поскольку, помимо чисто практической пользы, это помогает лучше понять внутреннюю логику инструментальных средств, при помощи которых создана программа.

После того как мы убедились в возможности редактировать отдельные свойства компонентов, было бы нелишним узнать, что произойдет в том случае, если удалить описание какого-либо свойства. Как ни странно, если выполнить удаление корректно — то обычно ничего страшного не случается. Хотя, справедливости ради, надо отметить, что в некоторых случаях удаление свойства приводит к катастрофическим последствиям — прежде всего, когда удаляемое свойство ссылается на другой компонент. Но, к счастью, такие специфические компоненты чаще всего не имеют никакого отношения в nag screen'ам и баннерам.

Причина "нечувствительности" программ к исчезновению свойств следующая — в Delphi и C++ Builder создание объекта на основе шаблона производится в два этапа. На первом этапе собственно создается объект, а все его свойства и внутренние структуры инициализируются значениями по умолчанию. И лишь на втором этапе новые значения свойств, которые хранятся в шаблоне диалога, помещаются на место значений по умолчанию. Неудивительно и то, что в eXeScope вы видите у каждого компонента намного меньше свойств, чем видит разработчик, редактируя форму в IDE — для уменьшения объема исполняемого файла в ресурсах сохраняются не все значения свойства, а только те, которые отличаются от значений по умолчанию. Например, в приведенном выше отрывке описания формы AboutForm вы не увидите свойство AutoSize, несмотря на то что такое свойство у формы имеется. Причина этого проста — значение свойства AutoSize по умолча-

нию было равно false, и аатору (то есть мне) не пришлось его менять, поскольку оно аполне меня устраивало.

Итак, теперь мы знаем, что описания свойств удалять можно. Осталось лишь разобраться, зачем это нам может понадобиться. В этот раз я отступлю от принципа "сначала — теория, потом — практика" и начну с описания небольшого эксперимента. Возьмите все тот же InqSoft Window Scanner, запустите и нажмите в нем кнопку "О программе...". Вы увидите окно с информацией о программе, а левой части которого будет находиться логотип. А теперь представьте себе, что это не безобидная картинка, которая никому не мешает, а ужасный черно-зелено-оранжевый баннер размером в пол-экрана, от которого мы хотели бы избавиться.

Следующим шагом будет определение идентификатора шаблона. Запустите вторую копию Window Scanner'а и "просканируйте" окно "О программе...". В очередной раз убедившись, что класс окна носит имя TAboutForm (да, я знаю, что написал про это несчастное окно уже как минимум пять килобайт — но мы будем действовать так, как будто видим программу впервые), лезем в секцию ресурсов и ищем там шаблон формы, спрятавшийся за идентификатором TABOUTFORM. Теперь нам нужно найти в шаблоне формы описание нашего "баннера". Нетрудно догадаться, что в этой роли выступает объект Image1 (поскольку картинка на форме одна, догадаться, что Image1 и есть наш "баннер", нетрудно). Дешифрованное описание этого объекта выглядит следующим образом:

```
object Image1: TImage
  Left = 0
  Top = 0
  Width = 57
  Height = 192
  AutoSize = True
  Picture.Data = {
    0A544A504547496D616765F5260000FFD8FFE000104A4649460001020000100
    ............
    3EEBFFD9}
end
```

Теперь можно сделать то, ради чего мы и затевали все эти поиски — удалите свойство Picture.Data, запустите программу и посмотрите, что случилось с нашей формой. А случилось именно то, чего мы и хотели добиться — логотип исчез, как будто его и не было. Сам объект, разумеется, никуда не делся — мы лишь удалили связанное с ним изображение, "подчистив" свойство Picture. Вспомните, что я говорил о двух этапах создания форм в Delphi, и вы легко поймете, каким образом изменилась логика работы программы — стереа значение, которое было назначено свойству Picture.Data программистом, вы не остааили программе иного выхода, кроме как использовать значение этого свойства по умолчанию. А значением по умолчанию в нашем случае оказалось "никакое" изображение (т.е. отсутствие изображения), которое программа успешно отобразила. Более того, удалять можно не только свойства объектов, но и объекты целиком. Однако такое действо является куда более грубым вмешательством в код программы, да и ограничений в этом случае намного больше.

А теперь давайте посмотрим, как простым редактированием ресурсов можно избавиться от настоящего, вполне реального баннера. Для этого нам потребуется программа Ghost Installer Free Edition. XML-редактор, входящий в состав этой программы, как раз содержит баннер. И этот баннер, кроме того, что занимает довольно много места внизу экрана и сокращает поле редактирования, еще и постоянно мерцает, раздражая пользователя. Так что, если вы активно пользуетесь бесплатной редакцией Ghost Installer'а, вам придется либо проявлять чудеса терпеливости, либо спасти огромное количество нервных клеток, избавиашись от созерцания зловредного баннера.

Сначала при помощи InqSoft Window Scanner определим имя класса окна той формы, которую мы собираемся править — TfrmProjectSource. Заодно будет нелишним просканировать и сам баннер — вдруг это позволит нам узнать хотя бы тип компонента, на основе которого этот баннер был создан. А если нам удастся узнать тип компонента, это заметно облегчит задачу поиска самого компонента в дешифрованном шаблоне формы. Пробуем просканировать окно баннера и узнаем, что на самом деле наш баннер — не что иное как кусок Internet Explorer'а, засунутый в программу при помощи технологии ActiveX. Что ж, с первого захода мы получили яано недостаточно полезной информации. Поэтому попробуем копнуть глубже и просканировать всю ветку дерева окон, имеющую отношение к нашему баннеру. InqSoft Window Scanner выдал следующий результат:

```
+[00050356] {TPanel}
 *[00010414] {TElSplitter}
 +[00050370] {TPanel}
  +[0001040E] {TPanel}
   +[00010410] Panel1 {TPanel}
    +[0043036A] {Shell Embedding}
     +[000103D4] {Shell DocObject View}
      *[00010412] {Internet Explorer_Server}
```

Что это значит? Всего лишь то, что наш ошметок Internet Explorer'а лежит на компоненте Panel1 типа TPanel (тут программист явно поленился стереть свойство Caption у объекта Panel1, чем сильно облегчил нам задачу поиска нужного объекта). Сам объект Panel1, в свою очередь, лежит на другой панели, которая лежит на панели, которая... В общем, всяких панелей там много, и разбираться в них можно долго. Поэтому я предлагаю начать наше расследование с той панели, которую мы идентифицировали однозначно. Откроем файл GIEditor.exe в Restorator 2.52 (да, вам опять придется обновить инструментарий — как показала практика, eXeScope с редактированием шаблона в данной программе не справился, а вот Restorator сработал как нельзя лучше), выберем шаблон нашей формы и в этом шаблоне найдем следующее:

```
object panMainBanner: TPanel
  Left = 0
  Top = 287
  Width = 553
  Height = 126
  Align = alBottom
  BevelOuter = bvNone
  TabOrder = 1
  object Panel1: TPanel
    Left = 0
    Top = 4
    Width = 553
    Height = 122
    Align = alBottom
    BevelOuter = bvLowered
    Caption = 'Panel1'
    TabOrder = 0
    object webMainBanner: TEmbeddedWB
      Left = 1
      Top = 1
      Width = 551
      Height = 120
      Align = alClient
      ...
      end
    end
end
```

Как видно из приведенного куска текста, это и есть описание нашей Panel1, баннера, который на этой панели находится (как оказалось, он представлен объектом webMainBanner типа TEmbeddedWB), и некой панели panMainBanner, на которой лежит Panel1. Было бы логично предположить, что имя panMainBanner расшифровывается как panel for Main Banner ("панель для главного баннера"), и что если ликвидировать эту панель вместе со всем ее содержимым, то назойливое мерцание баннера более не будет смущать наш взор. Сказано — сделано, переходим в режим редактирования и удаляем вышеприведенный блок текста из ресурса. Сохраняем, запускаем — и получаем GPF. Как я и предупреждал, безоглядное удаление компонентов — операция отнюдь не безобидная. Поэтому давайте поищем другой путь.

В самом начале главы я выделил четыре подхода к проблеме, и мы только что испробовали тот из них, который стоит под номером два. Это, конечно, было несколько непоследовательно с моей стороны, но иначе вы бы не смогли поэкспериментировать с уда-

лением объектов и увидеть, к чему это иногда приводит. И вот пришло время пойти правильным путем. Действительно — это ведь всего лишь баннер, он нв выпрыгивает в самый неподходящий момент на передний план, не блокирует работу с программой, даже не требует, чтобы по нему кликали. Так что нам нет принципиальной необходимости совершенно изничтожать этот баннер — достаточно просто его не видеть. В нашем случае, кстати, придется убрать не только баннер, но и панели, на которых он расположен, чтобы они не занимали место, и самый простой способ сделать это — уменьшить высоту панелей до нуля. Если быть до конца точным — нам надо спрятать лишь панель panMainBanner, поскольку компоненты webMainBanner и Panel1 находятся внутри этой панели, и после "исчезновения" panMainBanner тоже не будут видны (в общем случае этому могло бы помешать свойство AutoSize=true, но у наших подопытных компонентов это свойство равно false). В Restorator'е исправляем строчку Height=126 на Height=0, сохраняем результат и запускаем многострадальную программу. Баннера больше нет!

Читая эту главу, вы, наверное, отметили, что я меньше чем обычно говорил про общие подходы к проблемв рекламных окон. Действительно, чвм более частные вопросы крэкинга мы рассматриваем, тем сильнее приходится "привязываться" к особенностям операционных систем и средств разработки. Но даже в этом случае "за бортом" моего повествования осталось очень многое: визуальные средства для разработки под ДОС (да, были и такие — и некоторые из них, вроде старых версий FoxPro, все еще используются), использование шаблонов диалогов в Visual Basic/VBA и т.д. Вы сами, при желании, сможете узнать об этих вещах через самостоятельные эксперименты ничуть не меньше, чем я мог бы вам сообщить. А поскольку сама идея хранить шаблоны интерфейсов программ в унифицированном аиде давно и прочно вошла в практику программирования, было бы наивным предполагать, что новые средства разработки не принесут с собой новых форматов хранения этих шаблонов. Но я надеюсь, что и в этом случае вы сможете извлечь пользу из информации, изложенной в этой главе — разумеется, не как из руководства по конкретным инструментам, а как из источника идей.

(Продолжение следует)

ИССЛЕДОВАНИЕ ПРОГРАММ

sars //HI-TECH,
sars@ukrtop.com

Основные методы заражения PE EXE

Введение

Существует множество методов заражения (т.е. записи кода вируса в код программы без потери работоспособности последнего) файлов формата Portable Executable. Есть простые методы, которые с легкостью отлавливаются антивирусами, есть очень навороченные, о которых я вам когда-нибудь, может быть, расскажу... Вполне может случиться так, что вы изобретете свой собственный оригинальный способ, и успеет пройти дня этак три, прежде чем сотрудники Kaspersky Lab смогут его проанализировать и пополнить свои avc'шки парой-тройкой новых контрольных сумм... Чтобы иметь хотя бы теоретический шанс сделать это, вы должны:

- знать структуру PE;
- уметь находить адреса API'шных функций по их имени;
- assembler, assembler и еще раз assembler, что бы там ни говорили о нем ортодоксы из RSDN'а;
- разобраться с основными методами заражения PE, чем мы с вами сейчас и займемся под моим руководством.

Приведенные в статье исходники "заточены" под TASM32. Замечу жирным, что они **не являются вирусами**; и вообще, материал статьи предназначен исключительно для самообразования, и в первую очередь ориентирован на любопытных программистов, интересующихся местом жительства этих "зверушек". Те несколько строк, которые могут превратить вас из законопослушных исследователей программ в вирмейкеров, остаются на совести читателя, и в данной статье не рассматриваются.

Хочу выразить благодарность людям, которые оказали неоценимую помощь при подготовке данного материала: Serrgio, 90210, Dmitri Alexeenko; а также группам: HI-TECH, WASM, TNT.

Однако, ближе к делу. Существует три основных метода заражения PE EXE:

- внедрение в заголовок;
- расширение последней секции;
- добавление новой секции.

На этой оптимистической ноте одеваем белый халат, берем в руки скальпель и, вооружиашись бинокулярным микроскопом, осторожно приоткрываем чашку Петри с копошащейся там заразой.

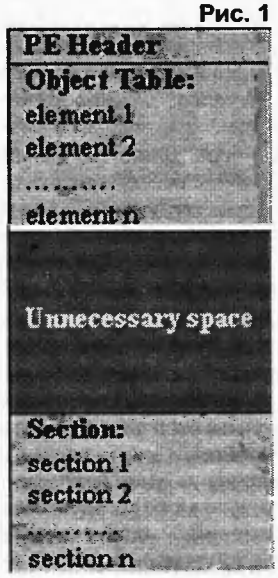
1. Внедрение в заголовок

Итак, рассмотрим первый метод заражения PE EXE-файлов — запись в заголовок "жертвы". Этот метод — не самый простой, но довольно интересный, с него и начнем.

Если кому интересно, то небезызвестный вирус win95.CIH (Чернобыль) записывал свою загрузочную процедуру именно в то место, которое будет рассмотрено ниже. Мы пока что пойдем простым путем и запишем туда все тело нашего "вируса". Я надеюсь, у вас под рукой есть описание структуры PE-заголовка, рекомендую почаще заглядывать в него с целью лучшего усвоения материала. Довольно неплохой труд у Hard'a Wisdom'a, хотя его статьи и имеют ряд неточностей. Для знающих английский язык, рекомендую описание PE от некого Luevelsmeyer.

Прежде всего отмечу, что "запись в заголовок" — это не совсем верное утверждение, так как рассматриваемое пространство находится не в самом заголовке. Но для простоты изложения я позволю себе это допущение.

Рис. 1



На **рис.1** изображена примерная структура PE EXE-файла.

Выделенный участок — это и есть "дыра", в которую мы запишем тело "вируса". Как видно из рисунка, она находится между последним элементом (element n) таблицы объектов и физическим смещением первой секции.

Откуда же эта дыра взялась? Дело в том, что существует такое понятие как **выравнивание**. Т.к. нас интересует физическое положение секции в файле, то за его величину отвечает поле File Align в PE Header по смещению 3Ch,

имеющее тип DWORD. В большинстве случаев оно равно 200h — это означает, что секции в файле должны быть расположены по адресам, кратным данному значению. Большинство линкеров выставляет физический адрес первой секции равным 600h, следовательно, первая секция располагается в файле по смещению 600h. Заголовок файла содержит намного меньше информации, чем 600h байтов, вот в результате и получается "пробел" — чистый клочок бумаги, на котором мы можем написать все, что пожелаем ;).

Вот вам реальный пример (**рис.2**).

Полагаю, пояснений не требуется, из рисунка видно, что до смещения 600h есть свободное место.

Для того чтобы найти это свободное место, необходимо:

- узнать некоторые значения из PE Header (**табл.1**);
- узнать некоторые значения из Object Table (**табл.2**);
- произвести парочку нехитрых арифметических операций.

Рис. 2

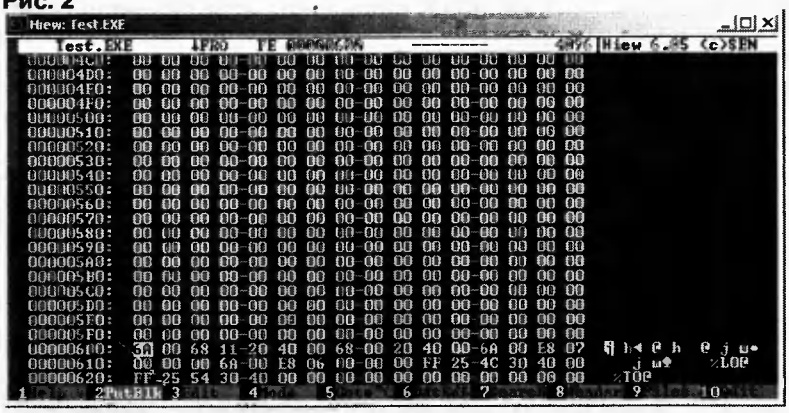


Табл. 1. PE Header

RVA	Size	Name	Description
06h	Word	Num of Objects	Число элементов в Object Table
14h	Word	NT Header Size	Размер заголовка PE файла -18h
54h	DWord	Header Size	Общий размер всех заголовков

Табл. 2. Object Table

RVA	Size	Name	Description
14h	DWord	Physical Offset	Физическое смещение секции относительно начала EXE-файла

Почему в поле NT Header Size присутствует ("минус"18h)? Дело в том, что это поле показывает размер PE-заголовка, начиная с поля Magic, которое находится по смещению 18h, следовательно, чтобы найти размер всего заголовка, необходимо прибавить смещение поля Magic.

Теперь давайте подробнее рассмотрим действия, которые нам необходимо выполнить для поиска свободного места, в скобках даны пояснения, откуда берутся значения — из PE Header или Object Table:

1. Находим размер PE заголовка (PE Header).
2. Находим размер таблицы объектов, для этого:
- берем количество элементов Object Table (PE Header);
- умножаем на размер одного элемента, т.е. на 40.
3. К виртуальному адресу (далее VA) файла-"жертвы" прибавим сумму результатов вычислений первого и второго пунктов, получим VA искомой области.
4. Находим физическое смещение первой секции:
- находим смещение первого элемента таблицы объектов — это конец заголовка PE и начало Object Table;
- получаем из этого элемента физическое смещение первой секции и добавляем к нему VA файла-"жертвы".
5. Вычисляем разность результатов четвертого и третьего пунктов, получаем размер искомой области.

А вот и сам код, который находит нужную "дыру" в заголовке:

```
;------------Из примера Example1-1.asm---------------------

_GetFreeSpace:
        mov edi,[esp]   ;Начало прочитанного файла
        mov esi,[edi].mz_neptr ;Offset PE Header
        movzx eax,word ptr [edi+esi].pe_numofobjects  ;В EAX кол-во
                                                      ;элементов
        imul ebx,eax,28h        ;Размер элемента 40 байт,
                                ;получили размер Object Table в EBX
        movzx eax,word ptr [edi+esi].pe_ntheadersize
        add esi,edi
        lea esi,[eax+esi+18h]  ;VA первого элемента в Object Table

        mov eax,[esi].oe_phys_offs     ;Физ. смещение первой секции
        add eax,edi     ;Прибавим VA файла в памяти
        add esi,ebx     ;VA свободного места в файле
        sub eax, esi    ;Размер свободного места
;-------------------------------------------------
```

***(От редакции:** архив sources.zip с исходными текстами примеров выложен на web-страницу журнала.)*

После выполнения этого кода в eax будет размер свободного места, а в esi его виртуальный адрес.

Увидели? Все просто и ясно. Теперь вам понятно, как можно "заразить" файл, не увеличивая при этом его размера?

Сейчас немного охлажу ваш пыл. Не все так просто, вот явные минусы этого метода:

- "дыра" не резиновая, а следовательно, и размер вируса ограничен;
- если вы поставите Entry Point на эту область, антивирусные программы это сразу заметят, т.к. они не любят, когда точка входа в программу указывает на заголовок. Напомню, что Entry Point (точка входа в программу) находится в PE Header и имеет смещение 28h.

С первым минусом бороться просто: возьмите любую статью, в которой рассказывается, как оптимизировать код по размеру, и не поленитесь пробежаться по своему исходнику. Есть мнение, что ресурсы современных компьютеров позволяют не принимать во внимание такую "мелочь" как размер программы, но наука под названием "вирмейкинг" придерживается на этот счет несколько иного мнения.

Для борьбы со вторым минусом можете по оригинальной Entry Point вставить jmp на ваш код, сохранив затертые байты (байты, поверх которых запишется jmp на код "вируса"), и антивирусы не смогут обнаружить вас по этому признаку, т.к. Entry Point не изменится.

Ну и еще пара моментов, которые могут доставить вам несколько "приятных" часов.

Запись в данную область запрещена, так что ваш код (в заголовке "жертвы") не сможет сохранять данные в свое тело, следовательно, можно забыть про глобальные переменные.

Есть три выхода, которые позволят обойти это ограничение, рассмотрим вкратце, в чем они заключаются.

1. Используйте VirtualProtect на эту страницу. В SDK написано, что функция VirtualProtect изменяет режим доступа к требуемой области памяти для текущего процесса. Вот простенький пример:

```
push esp            ;адрес переменной, в нее возвращается
                    ;"старый" режим доступа
push 40h            ;режим доступа (нам нужен 40h)
push 1000h          ;размер области памяти в байтах
push 400000h        ;адрес области памяти, чьи атрибуты страниц
                    ;нужно изменить
call VirtualProtect
```

Более детальное описание функции смотрите в Platform SDK.

2. Можно воспользоваться оригинальным методом Z0MBiE, сущность которого заключается в изменении атрибутов страниц в таблице страниц. Этот метод позволяет производить запись даже в kernel из ring3! Однако у него

есть один очень существенный недостаток — он работает только под win9X.

3. Чтобы не повторяться, я пошел другим путем — отказался от всех глобальных переменных в своем "вирусе" и использовал только локальные (т.е. стековые) переменные. Кто читал мою статью по поиску API (доступна на www.wasm.ru), должен был заметить, что в исходнике нет глобальных переменных, все операции я проводил через стек.

Вы думаете, это все? Напрасно, жизнь низкоуровневого программиста тяжела, давайте поглядим на сам код. Он хоть и будет работать, но возможны некоторые баги. Почему?

Мы берем первый элемент в таблице объектов и считаем, что он описывает первую секцию. Это утверждение основано на том, что секции идут в порядке возрастания, и первый элемент описывает первую секцию. Хорошо, если бы это всегда было так.

Однако в жизни все намного сложнее — элементы и секции PE-файла могут идти в любом порядке. И если алгоритм "вируса" не предусматривает подобную ситуацию, то он может запросто уничтожить такой файл, что не есть хорошо.

Выход здесь такой — нужно найти элемент с наименьшим физическим смещением секции, это и будет смещение первой секции.

Это было наше первое "упущение". Второе заключается в том, что после последнего элемента Object Table и до rva=SizeOfHeaders могут находиться так называемые Bound-импорты. На **рис.3** видно, что после последнего элемента Object Table есть еще "что-то". Вот это и есть Bound-импорты. Что это такое, мы рассмотрим более подробно в третьей части статьи, а пока просто посмотрите в исходнике, как происходит проверка на их наличие, и как их присутствие отражается на алгоритме поиска свободного места.

Размер и RVA Bound-импортов можно узнать из заголовка PE (**табл.3**).

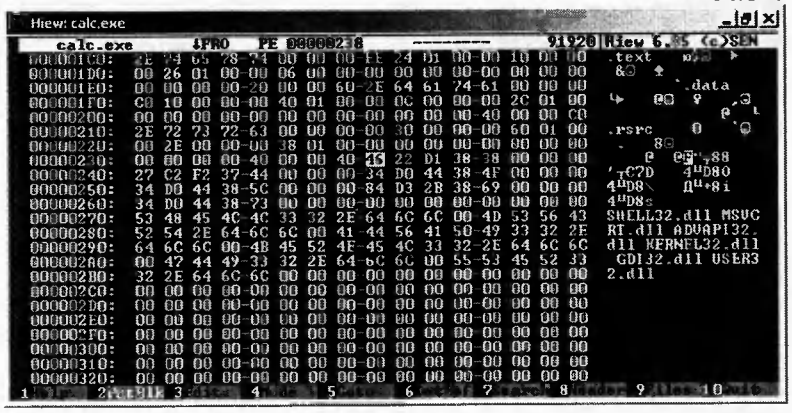
Табл. 3. PE Header:

RVA	Size	Name	Description
D0h	DWord	Bound Import RVA	RVA Bound импортов
D4h	DWord	Bound Import Size	Размер Bound импортов

```
;-------------Из примера Example1-2.asm-------------------------

_GetFreeSpace:
        mov edi,[esp]                   ;Начало прочитанного файла
        add edi,[edi].mz_neptr;Offset PE Header
        movzx ecx,word ptr [edi].pe_numofobjects ;В ECX кол-во
                                                 ;элементов (счетчик)
        push ecx                                 ;Сохраним
        movzx esi,word ptr [edi].pe_ntheadersize;NT Header size
        lea eax,[edi+esi+18h] ;VA первого элемента в Object
Table
        push eax        ;Сохраним
        mov ebx,[eax].oe_phys_offs  ;Наименьшее физ. смещение секции

_SearchLowhOffset:
        mov edx,[eax].oe_phys_offs    ;Физ. смещение секции
        cmp ebx, edx
        jb _BigOffset  ;Если больше, то смотрим следующий элемент
        mov ebx,edx    ;Иначе, примем его за наименьший

_BigOffset:
        add     eax, 28h                ;Следующий элемент
        loop _SearchLowhOffset

;ebx - физическое смещение первой секции

_CheckBounds:
        pop eax ;VA первого элемента в Object Table
        pop ecx ;Кол-во элементов
        imul ecx,ecx,28h        ;Размер элемента 40 байтов,
                                ;получили размер Object Table
        mov edx,[edi].pe_boundimportrva
;Присутствуют Bound-импорты?
        or edx,edx
        jz _NoBounds    ;Нет, в расчет не берем
        add ecx,[edi].pe_boundimportsize  ;Да, добавим их размер
                                          ;к Object Table

_NoBounds:
        add eax,ecx     ;VA "свободного места" в файле
        add ebx,[esp]
        sub ebx,eax     ;Размер "свободного места"
;-------------------------------------------------------------
```

Я думаю, не имеет смысла все подробно объяснять, ничего сверхъестественного в этом коде нет, просто мы находим первый элемент в таблице объектов и смотрим на физическое смещение секции, которую он описывает. Для начала мы принимаем это смещение за наименьшее. Далее находим второй элемент и сравниваем физическое смещение "его" секции с предыдущим. Если оно меньше, чем в первом элементе, то примем текущее за наименьшее, и таким образом работаем в цикле столько раз, сколько у нас элементов. В итоге мы находим элемент, в котором физическое смещение секции является наименьшим по отношению к другим элементам. Теперь можно утверждать, что этот элемент описывает первую секцию. Затем мы проверяем наличие Bound-импортов. Если обнаружено их присутствие, то добавим их размер к размеру PE Header и Object Table. Дальше все так же, как и в первом примере.

Рис. 3



Хочу сразу заметить, что данные примеры написаны не для того, чтобы их тупо передирали, а для объяснения самого принципа. Поэтому они не сохраняют содержимое регистров, и их алгоритм не оптимален. Я надеюсь, что вы разберетесь с этим и сделаете лучше, т.к. оптимизация кода выходит за рамки данной статьи.

Напоследок еще одна тонкость. В PE Header по смещению 54h находится DWord, который показывает общий размер всех заголовков, т.е. DOS Stub + PE Header + Object Table + Bound Imports. Это значение нужно сделать равным физическому смещению первой секции, т.е. мы как бы растянули заголовок до первой секции. Если этого не сделать, то загрузчик забьет нулями все, что находится между физическим смещением первой секции и общим размером заголовков. Таким образом мы потеряем часть "вируса".

Запись кода в заголовок показана в примере Example1-3.asm

(Продолжение следует)

Возвращаясь к напечатанному
("РМ. ВК" №5-6/2002)

Автоматическое определение кодировки текста — 2

И.РОЩИН,
г.Москва,
ZXNet: 500:95/462.53
E-mail: bestview@mtu-net.ru
WWW: http://www.ivr.da.ru

В статье я рассматривал алгоритм автоматического определения кодировки текста для случая, когда текст может быть в одной из трех кодировок: альтернативная кодировка DOS, Windows-1251 или KOI8-R. Там же приводилась процедура определения кодировки, написанная на ассемблере Z80 (в журнале она не была напечатана из-за нехватки места, но была выложена на сайте журнала и на моей web-странице).

В то же время, очевидно, что кому-то может потребоваться определять кодировку текста на компьютерной платформе, не основанной на процессоре Z80. Чтобы помочь в решении этой задачи, я написал библиотеку на Си, которую вы можете использовать в своих программах.

Определение кодировки в ней происходит по алгоритму, описанному в вышеупомянутой статье (напомню, принцип его работы основан на том, что не все возможные сочетания букв встречаются в словах русского языка). Как и в первоначальной процедуре на ассемблере Z80, повторяющиеся сочетания (т.е. уже встречавшиеся в тексте) отбрасываются, а определение кодировки происходит по табл.10 (см. вышеупомянутую статью).

Кстати, приведу еще один довод в пользу того, что отбрасывание повторяющихся сочетаний улучшает качество определения кодировки текста. Если в тексте несколько раз встречается какое-нибудь редкое слово или специальный термин, в котором содержится недопустимое сочетание букв, или обладающая тем же свойством аббревиатура (в которых возможны любые сочетания букв), то это недопустимое сочетание будет учтено лишь один раз, независимо от того, сколько раз оно встречается в тексте. А это, в свою очередь, уменьшит вероятность того, что при определении кодировки правильный вариант будет отвергнут из-за большого количества недопустимых сочетаний в нем.

Вернемся к рассмотрению библиотеки. Она содержит две доступные извне функции, определяющие кодировку текста. Функция m_def_code предназначена для случая, когда текст находится в памяти, а функция f_def_code — для случая, когда текст находится в файле. Кстати, предназначение каждой функции легко запомнить по первой букве ее названия: "m" — от "memory" (память), "f" — от "file" (файл).

При вызове функции m_def_code ей передается адрес текста в памяти, длина текста и количество различных сочетаний русских букв, которого достаточно для определения кодировки. Текст просматривается до тех пор, пока в каком-либо из трех вариантов не наберется указанное количество сочетаний (если, конечно, текст не закончится раньше).

При вызове функции f_def_code ей передается указатель на структуру, связанную с файлом, и уже упомянутое выше количество сочетаний.

Как видим, если в процедуре на ассемблере Z80 количество сочетаний, достаточное для определения кодировки, не указывалось при вызове, а было всегда равно 128, то здесь это изменяемый параметр. Он может принимать значения от 1 до 255 включительно — чем больше, тем надежнее определяется кодировка, но тем больше времени уходит на ее определение. На практике 128 — вполне достаточное значение для надежного определения кодировки, но вы, если хотите, можете его увеличить.

И функция m_def_code, и функция f_def_code возвращают номер кодировки текста: 0 — альтернативная кодировка DOS, 1 — Windows-1251, 2 — KOI8-R.

Ниже приведен текст библиотеки.

```
/* ============================================================
Файл: def_code.c
Компилятор: Turbo C 2.0
Описание: библиотека для автоматического определения кодировки
          текста (ALT, WIN, KOI).
          Функция m_def_code - для случая, когда текст в памяти,
          функция f_def_code - когда текст в файле.
Описание алгоритма: http://ivr.webzone.ru/articles/defcod_2/
                    (c) Иван Рощин, Москва, 2004.
============================================================ */
#include <stdio.h>

/* Глобальные переменные */

static int len_;
static unsigned char *p_;
static FILE *f_;

/* Таблица сочетаний */

static unsigned char table_2s[128]={0xFF,0xFF,0xFF,0xC7,0xFE,
 0xBE,0xF7,0xFB,0xFD,0xBF,0xF7,0xF9,0xFC,0xBE,0xF1,0x80,0xFF,
 0xFF,0xF7,0xBB,0xFF,0xFF,0xFF,0xCF,0xDE,0xBF,0xD1,0x08,0xFF,
 0xBF,0xF1,0xBF,0xFF,0xFF,0xFF,0xC7,0x1D,0x3F,0x7F,0x81,0xA7,
 0xB6,0xF2,0x82,0xFF,0xFF,0x75,0xDB,0xFC,0xBF,0xD7,0x9D,0xFF,
 0xAE,0xFB,0xDF,0xFF,0xFF,0xFF,0xC7,0x84,0xB7,0xF3,0x9F,0xFF,
 0xFF,0xFF,0xDB,0xFF,0xBF,0xFF,0xFF,0xFD,0xBF,0xFF,0xFF,0xFF,
 0xFF,0xE7,0xC7,0x84,0x9E,0xF0,0x12,0xBC,0xBF,0xF0,0x84,0xA4,
 0xBA,0x10,0x10,0xA4,0xBE,0xB8,0x88,0xAC,0xBF,0xF7,0x0A,0x84,
 0x86,0x90,0x08,0x04,0x00,0x00,0x03,0x7F,0xFD,0xF7,0xC1,0x7D,
 0xAE,0x6F,0xCB,0x15,0x3D,0xFC,0x00,0x7F,0x7D,0xE7,0xC2,0x7F,
 0xFD,0xF7,0xC3};
/============================================================
   Вспомогательная функция alt2num.
   Вход: a - код русской буквы в кодировке ALT.
   Выход: порядковый номер этой буквы (0...31).
 ============================================================ */

static int alt2num (int a)
{
 if (a>=0xE0) a-=0x30;
 return (a&31);
}

/* ============================================================
   Вспомогательная функция koi2num.
   Вход: a - код русской буквы в кодировке KOI.
   Выход: порядковый номер этой буквы (0...31).
 ============================================================ */

static int koi2num (int a)
[
 static unsigned char t[32]={30,0,1,22,4,5,20,3,21,8,9,10,11,12,
  13,14,15,31,16,17,18,19,6,2,28,27,7,24,29,25,23,26};

 return (t[a&31]);
}

/* ============================================================
Вспомогательная функция work_2s - обработка двухбуквенного
                                  сочетания.
Вход: c1 - порядковый номер первой буквы (0...31),
      c2 - порядковый номер второй буквы (0. ..31),
      check - надо ли проверять, встречалось ли сочетание
              раньше (1 - да, 0 - нет),
      buf - адрес массива с информацией о встреченных сочетаниях.
Выход: 0 - указанное сочетание уже встречалось раньше,
       1 - сочетание не встречалось раньше и является допустимым,
       2 - сочетание не встречалось раньше и является недопустимым.
============================================================ */

static int work_2s (int c1, int c2, int check, unsigned char
buf[128])
{
 int i=(c1<<2)+(c2>>3); /* Номер байта в массиве. */
 int mask=0x80>>(c2&7); /* Маска, соответствующая номеру
                           бита в байте. */
/* Если check=1, проверяем: если соответствующий бит массива
buf равен 0, значит, указанное сочетание уже встречалось раньше.
```


```
Тогда выходим из функции, возвращая 0. Если же сочетание не
встречалось, то помечаем, что оно встретилось (обнуляем
соответствующий бит массива buf). */

 if (check==1)
 {
  if ((buf[i]&mask)==0) return (0);
  buf[i]&=~mask;
 }

 /* Проверяем, допустимо сочетание или нет. */

 if ((table_2s[i]&mask)!=0) return (1); /* Допустимо. */
 return (2);                            /* Недопустимо. */
}

/* ============================================================
Вспомогательная функция def_code - определение кодировки текста.
Функции m_def_code и f_def_code - лишь надстройки над этой функцией.
Вход: get_char - указатель на функцию, которую надо вызывать
      для получения очередного символа текста. Функция должна
      возвращать либо код символа, либо -1 при достижении конца
      текста.
      n - количество различных сочетаний русских букв (1-255),
      которого достаточно для определения кодировки.
Выход: 0 - текст в кодировке ALT, 1 - WIN, 2 - KOI.
============================================================ */

static int def_code (int (*get_char)(), int n)
{
/* В массиве buf_1 хранится информация о том, какие сочетания
руских букв уже встречались в варианте ALT, а в массиве
buf_2 - в варианте WIN. */

 unsigned char buf_1 [128];
 unsigned char buf_2 [128];

 int bad_1=0;
 int bad_2=0;
 int bad_3=0;
 int all_1=0;
 int all_3=0;  /* all_2=all_3 */

 int c1; int c2=0;
/* Символы текущего обрабатываемого сочетания. */
 int i;

 /* Инициализация buf_1 и buf_2. */

 for (i=0;i<128;i++) buf_1[i]=0xFF;
 for (i=0;i<128;i++) buf_2[i]=0xFF;

/* Главный цикл - обработка сочетаний для каждого из трех
вариантов. Цикл выполняется, пока не кончится текст или в
каком-либо из вариантов не встретится n сочетаний. */

 while (((c1=c2,c2=(*get_char)())!=-1)&&(all_1<n)&&(all_3<n))
 (
/* Вариант ALT. Вначале проверяем, являются ли символы
текущего сочетания кодами русских букв в кодировке ALT. */

  if ((((c1>=0x80)&&(c1<0xB0))||((c1>=0xE0)&&(c1<0xF0)))&&
      (((c2>=0x80)&&(c2<0xB0))||((c2>=0xE0)&&(c2<0xF0))))
  [
   switch (work_2s(alt2num(c1),alt2num(c2),1,buf_1)) /* Обработали. */
   (
    case 2: bad_1++;
    case 1: all_1++;
   }
  }
/* Варианты WIN и KOI. Вначале проверяем, являются ли символы
текущего сочетания кодами русских букв в этих кодировках
(в обеих кодировках диапазоны кодов русских букв совпадают). */

  if ((c1&c2)>=0xC0)
/* Эквивалентно условию (c1>=0xC0)&&(c2>=0xC0). */
  {
   switch (work_2s(c1&31,c2&31,1,buf_2)) /* Обработали. */
   [
    case 0: continue;
/* Если сочетание букв уже встречалось в варианте WIN, то оно
уже встречалось и в варианте KOI, так что  пропускаем обработку
варианта KOI и переходим к следующей итерации главного цикла. */
    case 2: bad_2++;
   }

/* Если сочетание букв еще не встречалось в варианте WIN, то
оно заведомо не встречалось и в варианте KOI, поэтому
специально проверять это не надо - значит, функцию work_2s
вызываем с параметром check, равным 0. */

   switch (work_2s(koi2num(c1),koi2num(c2),0,NULL))
/* Обработали. */
   {
    case 2: bad_3++;
    case 1: all_3++;
   }
  }
 }

/* Данные собраны. Теперь, если в каком-либо из вариантов
недопустимых сочетаний не больше 1/32 от общего их числа,
то считаем, что их и не было. */

if (bad_1<=(all_1>>5)) bad_1=0;
if (bad_2<=(all_3>>5)) bad_2=0;
if (bad_3<=(all_3>>5)) bad_3=0;


 /* Получаем результат. */

 (
  unsigned int a=((255-bad_1)<<8)+all_1;
  unsigned int b=((255-bad_2)<<8)+all_3;
  unsigned int c=((255-bad_3)<<8)+all_3;

  if ((a>=b)&&(a>=c)) return (0);
  if (b>=c) return (1); else return (2);
 )
)

/* ============================================================
Вспомогательная функция m_get_char вызывается из функции
def_code, когда та вызвана из m_def_code.
Выход: очередной символ текста или -1, если текст кончился.
============================================================ */

static int m_get_char()
{
 if (len_==0) return (-1);
 len_--;
 return (*(p_++));
)

/* ============================================================
Функция m_def_code - определение кодировки находящегося в
памяти текста.
Вход:  p - адрес текста,
       len - длина текста,
       n - количество различных сочетаний русских букв (1-255),
           которого достаточно для определения кодировки.
Выход: 0 - текст в кодировке ALT, 1 - WIN, 2 - KOI.
============================================================ */

int m_def_code (unsigned char *p, int len, int n)
{
/* Присваиваем значения глобальным переменным len_ и p_, которые
будут доступны из функции m_get_char. */
 len_=len;
 p_=p;
 /* Получаем результат. */
 return (def_code(&m_get_char,n));
}

/* ============================================================
Вспомогательная функция f_get_char вызывается из
функции def_code, когда та вызвана из f_def_code.
Выход: очередной символ текста или -1, если текст кончился.
============================================================ */

static int f_get_char()
{
 int a=fgetc(f_);
 if (a==EOF) return (-1);
 return (a);
}

/*============================================================
```

```
Функция f_def_code - определение кодировки находящегося в файле текста.
Вход: f - указатель на структуру, связанную с файлом (указатель
      текущей позиции в файле должен указывать на начало файла),
      n - количество различных сочетаний русских букв (1-255),
      которого достаточно для определения кодировки.
Выход: 0 - текст в кодировке ALT, 1 - WIN, 2 - KOI.
============================================================ */

int f_def_code (FILE *f, int n)
{
/* Присваиваем значение глобальной переменной f_, которая будет
доступна из функции f_get_char. */
 f_=f;
 /* Получаем результат. */
 return (def_code(&f_get_char,n));
}
```

После написания библиотеки на Си я решил оптимизировать первоначальную процедуру на ассемблере Z80. Если раньше эта процедура во время работы использовала три 128-байтных массива (BUF_1...BUF_3), то теперь — только два. Уменьшилась и длина самой процедуры, без учета длины этих массивов — с 469 байтов до 393, т.е. на 76 байтов (значения приведены для худшего случая, они могут быть немного уменьшены в случае расположения массивов, используемых в этих процедурах, по "удобным" адресам). К тому же новый вариант процедуры работает в несколько раз быстрее.

На то, что в журнале найдется место для листинга оптимизированного варианта процедуры, я даже и не надеюсь :), но вы можете скачать его с сайта журнала или с моей web-страницы.

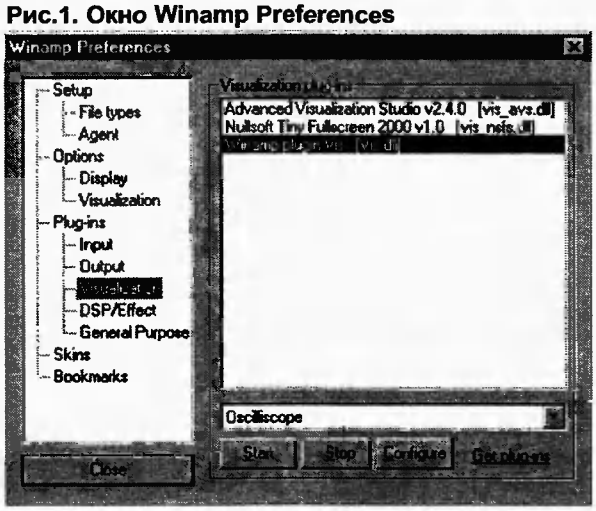
Разработка плагинов визуализации для Winamp

И.СОШИН,
г.Лысьва,
Пермской области,
E-mail: s1w17@permonline.ru

Современные музыкальные проигрыватели, кроме непосредственно воспроизведения звука, имеют большое количество дополнительных возможностей. Среди них можно отметить широкий набор настроек, возможности дистанционного управления и подключения внешних модулей — плагинов. Среди последних наиболее часто встречаются плагины изменения внешнего вида — скины, модули обработки звукового потока и плагины визуализации. Для самой известной программы-проигрывателя Winamp существует множество разнообразных модулей, но самые яркие впечатления остаются от модулей визуализации. Эти модули преобразуют информацию о звуковом потоке в разнообразные визуальные эффекты на экране компьютера. Хорошо известны творения фирмы WildTanget для Winamp. Они выводят на экран трехмерные модели, танцующие под музыку.

Создать собственный модуль визуализации несложно. В этой статье приведены основные принципы программирования визуализации для Winamp в среде Delphi.

Рис.1. Окно Winamp Preferences



Плагин визуализации для Winamp представляет собой обычную библиотеку типа **dll**, в которой описываются как необходимые действия по работе с проигрывателем, так и непосредственно процедуры визуализации. Данная библиотека помещается в подкаталог плагинов проигрывателя, по умолчанию имеющий путь C:\Program Files\Winamp\plugins. В этом подкаталоге также находятся модули обработки звука.

Запуск нужного плагина производится из специального меню визуализации проигрывателя. Вызов данного меню можно осуществить путем нажатия на правую кнопку мыши над окном графической информации. При выборе команды Select Plug-in появляется окно Winamp Preferences. В этом окне, на панели Visualization Plug-ins, отображаются имена доступных библиотек визуализации и выполняемые ими функции (**рис.1**). Нажатием кнопки Start можно запустить предварительно выделенный модуль. Кнопка Stop останавливает работу, а кнопка Configure вызывает процедуру настройки модуля.

При старте плагина визуализации, проигрывателем из **dll** вызывается процедура инициализации, назначение которой — подготовка системы к работе плагина.

Во время работы проигрыватель через определенные промежутки времени передает модулю необходимые данные о звуковом потоке и вызывает функцию рендеринга, которая осуществляет основные действия по построению изображения. При этом проигрыватель Winamp может передавать информацию о звуковом потоке в виде осциллограммы и в виде анализатора спектра. Оба формата представляют собой двумерный массив, содержащий данные левого и правого каналов стереосигнала. Глубина массива составляет 575 элементов, каждый из которых может принимать значение от 0 до 255. Эти данные могут быть использованы для непосредственного отображения.

При вызове окна Winamp Preferences, проигрыватель производит поиск доступных внешних модулей. Чтобы модуль мог быть обнаружен Winamp, он должен содержать заголовок следующей структуры:

```
TWinampVisHeader = record
  version:     Integer;
  description: PChar;
  getModule:
function(i:Integer):PWinampVisModule;cdecl;
end;
```

Здесь: version — версия плагина; description — название плагина, отображаемое в окне выбора модулей визуализации; getModule — функция, возвращающая указатель рабочей переменной типа TWinampVisModule, через которую происходит обмен данными.

Структура TWinampVisModule содержит следующие поля:

```
TWinampVisModule = record
    description:  PChar;
    hwndParent:   HWND;
    hDllInstance: HINST;
    sRate,nCh, latencyMs, delayMs, spectrumNch, waveformNch:  Integer;
    spectrumData: array[0..1,0..575]of char;
    waveformData: array[0..1,0..575]of char;
    Config:       procedure(this_mod:PWinampVisModule);cdecl;
    Init:           function(this_mod:PWinampVisModule):Integer;cdecl;
    Render:       function(this_mod:PWinampVisModule):Integer;cdecl;
    Quit:           procedure(this_mod:PWinampVisModule);cdecl;
    userData:     Pointer;
end;
```

Поля структуры имеют следующие значения: description — описание функции модуля; hwndParent — дескриптор родительского окна; hDllInstance — указатель на данную DLL; sRate, nCh — количество каналов; latencyMs — задержка между командой визуализации и непосредственно визуализацией в миллисекундах; delayMs — задержка по времени между вызовами процедуры визуализации в миллисекундах; spectrumNch, waveformNch — переменные, определяющие режим, в котором будет работать

плагин (анализатор спектра или осциллограф); spectrumData — данные анализатора спектра сигнала (используются в режиме анализатора спектра); waveformData — данные по огибающей сигнала (используются в режиме осциллографа); userData — данные пользователя.

В состав структуры входит описание основных функций и процедур, необходимых для работы совместно с Winamp:

- Init — пользовательская функция инициализации. Вызывается при активации модуля. В ней необходимо описать действия, которые должны производиться при запуске модуля. Должна возвращать "1", если инициализация прошла успешно;
- Config — пользовательская процедура конфигурации. Вызывается при нажатии кнопки Configure на панели Visualization Plug-ins.
- Render — пользовательская функция визуализации. В ней описываются действия, необходимые для построения изображения. Winamp вызывает данную функцию через отрезки времени заданные переменной delayMs. Процедура должна возвращать "1", пока производится отрисовка изображения, и "0" — если она закончена;
- Quit — пользовательская процедура выхода, описывающая действия при деактивации плагина.

Ниже приведен пример простого плагина визуализации, написанного без применения VCL. Данная программа осуществляет вывод сигналов анализатора спектра или осциллографа в отдельном окне. Для отрисовки изображения используются функции WinAPI.

Создание плагина начинается с создания нового проекта динамической библиотеки — DLL. Имя проекта должно соответствовать имени будущего плагина. В описание подключаемых модулей нужно включить только модуль Windows и будущий программный модуль. В описании процедур необходима только процедура передачи заголовка библиотеки.

```
{Динамическая библиотека визуализации для Winamp}
library Vis;
uses
  plug in 'plug.pas', {Основной модуль}
  Windows;
Exports
  WinampVisGetHeader; {Функция передачи Winamp заголовка модуля}
end.
```

Затем создается основная часть плагина — программный модуль, не содержащий формы, в котором описываются все действия по работе с проигрывателем Winamp. Его нужно сохранить с именем, указанным в заголовочном файле библиотеки.

В разделе описания типов плагина указываются структуры для работы с проигрывателем и указатели на соответствующие структуры.

Раздел **implementation** начинается с определения основных констант модуля. Первая константа имеет тип TWinampVisHeader и однозначно описывает заголовок модуля. Две константы типа TWinampVisModule описывают режимы осциллографа и анализатора спектра. Каждая из констант имеет индивидуальную процедуру визуализации. Переключение между данными константами, а соответственно, и режимами визуализации, осуществляется в окне Visualization Plug-ins. При этом проигрыватель вызывает функцию **getModule**, которая передает указатель на нужную переменную.

Функция инициализации модуля создает методами WinAPI новое окно, в котором и будут отображаться результаты визуализации.

Процедура конфигурации Config в данном примере осуществляет только вывод информации о плагине.

Функция визуализации render1 выводит на экран осциллограмму. Данные осциллограммы, получаемые от Winamp, разбиты на две части, лежащие по разные стороны от нулевой оси. Отрицательные значения идут от 0 в сторону увеличения, положительные — от 255 в сторону уменьшения. Для нормального отображения осциллограммы все данные приводятся к одной оси. Результат работы функции приведен на **рис.2**.

Функция render2 осуществляет построение графика данных анализатора спектра. Результат работы функции приведен на **рис.3**.

При компиляции плагина Delphi выводит сообщение об ошибке, заключающейся в невозможности отладки программы, так как получаемая dll не является исполняемым файлом. Готовый модуль необходимо скопировать в подкаталог Plugins каталога Winamp. Далее открывается окно Winamp Preferences с панелью Visualization Plug-ins, где выбирается нужный плагин и нажимается кнопка Start. Работоспособность модуля проверялась под операционными системами Windows 98/2000 и версиями проигрывателя 2.61,2.73 и 5.00

```
{Модуль визуализации для Winamp}
unit plug;

interface

uses
  Windows, Messages;

type
  PWinampVisModule =^TWinampVisModule;
  TWinampVisModule = record
    description:  PChar;
    hwndParent:   HWND;
    hDllInstance: HINST;
    sRate,nCh,latencyMs,delayMs,spectrumNch,waveformNch:
Integer;
    spectrumData: array[0..1,0..575]of char;
    waveformData: array[0..1,0..575]of char;
    Config:       procedure(this_mod:PWinampVisModule);cdecl;
    Init:
function(this_mod:PWinampVisModule):Integer;cdecl;
    Render:
function(this_mod:PWinampVisModule):Integer;cdecl;
    Quit:         procedure(this_mod:PWinampVisModule);cdecl;
    userData:     Pointer;
  end;
  PWinampVisHeader =^TWinampVisHeader;
  TWinampVisHeader = record
    version:     Integer;
    description: PChar;
    getModule:   function(i:Integer):PWinampVisModule;cdecl;
  end;
  WinampVisGetHeaderType=function:PWinampVisHeader;

const ver=$100;    {Версия плагина (1.00)}

var
  WC      : TWndClassEx; {Класс окна}
  MainWnd : HWND;  {Хэндл окна}
```

Рис.2. Кривая на экране осциллографа

Рис.3. Анализатор спектра.

```
  Msg     : TMsg;  {Сообщения Windows}
  hDC1    : HDC;   {Дескриптор окна}

function  WinampVisGetHeader:PWinampVisHeader;cdecl;export;
function  GetModule(which:Integer):PWinampVisModule;cdecl;
procedure config(this_mod:PWinampVisModule);cdecl;
function  init(this_mod:PWinampVisModule):Integer;cdecl;
function  render1(this_mod:PWinampVisModule):Integer;cdecl;
function  render2(this_mod:PWinampVisModule):Integer;cdecl;
procedure quit(this_mod:PWinampVisModule);cdecl;
function  WindowProc(Wnd:THandle;Msg:Integer;wParam:Word;
                  lParam:Longint):Integer;stdcall;export;

implementation

const
{Заголовок модуля}
hdr: TWinampVisHeader =(version:ver;
 description:'Winamp plugin Vis'; {Название модуля в Winamp
Preferences}
 getModule:getModule);

{Данные для Осциллографа}
fmod1: TWinampVisModule =
  (description:'Oscilliscope';
   hwndParent:0;
   hDllInstance:0;
   sRate:0;
   nCh:0;
   latencyMs:25;
   delayMS:25;
   spectrumNch:0;
   waveformNch:2;
   Config:config;
   Init:init;
   Render:render1;
   Quit:quit);

{Данные для анализатора спектра}
fmod2: TWinampVisModule =
  (description:'SpectrAnalizer';
   hwndParent:0;
   hDllInstance:0;
   sRate:0;
   nCh:0;
   latencyMs:25;
   delayMS:25;
   spectrumNch:2;
   waveformNch:0;
   Config:config;
   Init:init;
   Render:render2;
   Quit:quit);

{Функция передачи Winamp заголовка модуля}
function WinampVisGetHeader:PWinampVisHeader;
begin
  Result:=@hdr;
end;

{Функция реакции на сообщения операционной системы}
function WindowProc(Wnd: THandle; Msg: Integer; wParam : Word;
                    lParam: Longint): Integer; stdcall;
export;
begin
  Result:= 0;
  case Msg of
    WM_DESTROY :
    begin
      PostQuitMessage(0);
    end;
  end;
  Result:= DefWindowProc(Wnd, Msg, WParam, LParam);
end;

{Определение нужной константы данных}
function GetModule(which:Integer):PWinampVisModule;
begin
  Case which of
  0 : Result:=@fmod1; {Для осциллографа}
  1 : Result:=@fmod2; {Для анализатора спектра}
  else Result:=nil;
  end;
end;

{Процедура конфигурации плагина}
{Запускается из Winamp кнопкой Configure}
{Здесь можно произвести настройку работы модуля}
procedure config(this_mod:PWinampVisModule);
begin
  MessageBox(0,'Модуль визуализации для Winamp',
  'WinampVisConfig',MB_ICONINFORMATION);
end;

{Функция инициализации плагина}
{Запускается из Winamp кнопкой Start}
function init(this_mod:PWinampVisModule):Integer;
begin
  {Создание окна визуализации}
  WC.cbSize           := SizeOf(WC);
  WC.style            := CS_HREDRAW or CS_VREDRAW;
  WC.lpfnWndProc      := @WindowProc;
  WC.cbClsExtra       := 0;
  WC.cbWndExtra       := 0;
  WC.hInstance := this_mod.hDllInstance;
  WC.hIcon            := LoadIcon(0, IDI_APPLICATION);
  WC.hCursor   := LoadCursor(0, IDC_ARROW);
  WC.hbrBackground:= GetStockObject(BLACK_BRUSH);
  WC.lpszMenuName     := '';
  WC.lpszClassName:= 'WinampVis';
  if RegisterClassEx(WC) <> 0 then
   begin
     MainWnd := CreateWindowEx(0,'WinampVis','Визуализация для Winamp',
                     WS_OVERLAPPED or WS_SYSMENU or WS_CAPTION,
                     CW_USEDEFAULT, CW_USEDEFAULT,300, 255,
            this_mod.hwndParent, 0, this_mod.hDllInstance, NIL);
     if MainWnd <> 0 then
      begin
        ShowWindow(MainWnd, CmdShow); {Показать окно}
        hDC1:=GetDC(MainWnd); {Получение дескриптора окна}
        Rectangle(hDC1,0,0,300,230);
        Result:=0;
      end
       else Result:=1;
   end
    else Result:=1;
end;

{Функция отрисовки осциллографа}
function render1(this_mod:PWinampVisModule):Integer;
var
  y,i: integer;
begin
  i:=1;
  Rectangle(hDC1,0,0,300,230); {Очистка окна}
  MoveToEx(hDC1,300,100,nil);
  LineTO(hDC1,0,100);
  While i<300 do
  begin
  y:=ord(this_mod^.waveformData[1,i]);  {Получение данных}
  if y>125 then y:=y-255;               {Приведение данных}
  LineTo(hDC1,i,y+100);                 {Отрисовка кривой}
  i:=i+1;
  end;
  Result:=0;
end;

{Функция отрисовки анализатора спектра}
function render2(this_mod:PWinampVisModule):Integer;
var
  y,i,x: integer;

begin
  Rectangle(hDC1,0,0,300,230);
  i:=1; x:=1;
  While i<570 do
  begin
  y:=abs(ord(this_mod^.spectrumData[0,i])); {Получение данных}
   {Отрисовка анализатора спектра}
  Rectangle(hDC1,x,255,x+1,255-(y*2));
  i:=i+2; x:=x+1;
  end;
  Result:=0;
end;

{Процедура завершения работы модуля}
procedure quit(this_mod:PWinampVisModule);
begin
  ReleaseDC(MainWnd,hDC1);
  SendMessage(MainWnd, WM_CLOSE, 0, 0);
end;

end.
```

Светомузыка на клавиатуре

В.РУБАШКА,
г.Лисичанск,
Луганской обл.

Знакомство с удивительно интересным направлением современной компьютерной периферии — компьютерными светоэффектами — можно начать с простейшей светомузыки на клавиатуре. Для ее изготовления в маломощном демонстрационном варианте не понадобится даже паяльник! Три светодиода-индикатора (Num Lock, Caps Lock, Scroll Lock), расположенные в правом верхнем углу клавиатуры, будут выполнять роль мини-экрана нашей светомузыки. Управлять свечением этих светодиодов можно не только соответствующими клавишами клавиатуры, но и программным путем [1]. Для этого необходимо в порт с адресом 60H (H указывает, что адрес представлен в шестнадцатеричной форме) записать код команды ED. Затем, с интервалом в несколько миллисекунд, необходимых для обработки и передачи кода контроллером клавиатуры, записать в этот же порт восьмиразрядный двоичный код, три младших разряда которого управляют светодиодами клавиатуры. Логическая "1" в соответствующем разряде зажигает светодиод, а логический "0" — гасит. Первый разряд управляет светодиодом Scroll Lock, второй — Num Lock, а третий — Caps Lock. Значение остальных разрядов (4...8) должны быть равны "0". В таблице указаны соответствия между двоичным кодом состояния светодиодов клавиатуры и его десятичным эквивалентом.

Разряд 8	Разряд 7	Разряд 6	Разряд 5	Разряд 4	Разряд 3	Разряд 2	Разряд 1	Десятичное
					Caps	Num	Scroll	значение
0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1	1
0	0	0	0	0	0	1	0	2
0	0	0	0	0	0	1	1	3
0	0	0	0	0	1	0	0	4
0	0	0	0	0	1	0	1	5
0	0	0	0	0	1	1	0	6
0	0	0	0	0	1	1	1	7

Программа "DEMO KEYBOARD LIGHT (RANDOMIZE)", написанная на языке QBASIC 4.5, демонстрирует работу светодиодов клавиатуры в режиме случайного включения — выключения.

```
10  REM "DEMO KEYBOARD LIGHT (RANDOMIZE)"

20  TIMER ON: ON TIMER(1) GOSUB 40
30  WHILE 1: j = j + 1: WEND
40  np = INT(j / 50): vp = INT(j / 2): dp = INT((vp - np) / 50)
50  tempo = vp / 2

60  OUT &H60, &HED
70  x = INT(RND * 8)
80  OUT &H60, x
90  FOR T = 0 TO tempo: NEXT T:

100 k$ = INKEY$
110 IF k$ = CHR$(27) THEN STOP
120 IF k$ = CHR$(46) THEN tempo = tempo - dp:
130 IF tempo <= np THEN tempo = np:
140 IF k$ = CHR$(44) THEN tempo = tempo + dp:
150 IF tempo >= vp THEN tempo = vp:
160 GOTO 60
```

Чтобы набрать эту программку, достаточно всего нескольких минут. Нажимаем "START" и любуемся плодами своей работы — светодиоды на клавиатуре начинают весело перемигиваться. Причем скорость их переключения (tempo) можно регулировать клавишами "<" или ">" клавиатуры, а выйти из программы можно по нажатию клавиши "ESC". Эти функции обслуживают строки программы с номерами 100...150. В строках 20...50 определяется быстродействие компьютера (количество простейших операций за 1 с) и вычисляются константы, необходимые для регулировки скорости переключения. Вывод очередных случайных данных в порт 60H, определяющих, какие из светодиодов будут гореть, и задержка переключения на время tempo организованы в строках 60...90.

Налюбовавшись необычной светомузыкой, можно задаться вопросом — как ее усовершенствовать? Есть несколько вариантов. Можно переписать программу так, чтобы она эмулировала "бегущий огонь", "бегущую тень", их реверс, накопление-денакопление, изменяла скорость и автоматически перебирала все многообразие используемых эффектов. Можно привязать скорость переключения к музыке, а смену эффектов — ко времени. Здесь все зависит от вашей фантазии и открывает неограниченные творческие возможности. Тот, кто собирал автоматы светоэффектов в "железе", знает, как нелегко перестроить их на другой алгоритм работы. Это и коренное изменение в схеме, и ввод дополнительных узлов, и перепрограммирование ПЗУ. На компьютере все значительно проще и намного эффективнее — достаточно дописать несколько строчек кода, и работа светоэффекта может в корне измениться! Начинает сбываться мечта, когда вместо паяльника и осциллографа на первый план выходит творческая мысль, воплощаемая с помощью новой компьютерной программы. Следующая программа "DEMO KEYBOARD LIGHT (10 EFFEKT)" реализует десять эффектов, которые выбирают рядом клавиш "1"..."0". Количество эффектов можно увеличить, добавив их в базу данных DATA (строки 110...200), не забывая при этом изменить размерность массива в строках 60 и 70 (в данном примере он равен 60). Также следует добавить количество управляющих клавиш для выбора эффектов по примеру строк 310...400.

```
10  REM "DEMO KEYBOARD LIGHT (10 EFFEKT)"

20  TIMER ON: ON TIMER(1) GOSUB 40
30  WHILE 1: j = j + 1: WEND
40  np = INT(j / 50): vp = INT(j / 2): dp = INT((vp - np) / 50)
50  tempo = vp / 3

60  DIM a(60)
70  FOR S = 1 TO 60
80  READ F
90  a(S) = F
100 NEXT S

110 DATA 2,4,1,2,4,1
120 DATA 1,4,2,1,4,2
130 DATA 2,0,4,0,1,0
140 DATA 1,0,4,0,2,0
150 DATA 2,6,7,5,1,0
160 DATA 1,5,7,6,2,0
170 DATA 2,6,7,6,2,0
180 DATA 1,5,7,5,1,0
```

```
  190 DATA 3,4,3,4,3,4
  200 DATA 0,7,4,3,4,7

  210 a.d = &H60: y = 1
  220 x = y * 6 - 5
  230 OUT a.d, &HED: FOR z = 0 TO 100: NEXT z: OUT a.d, a(x):
PRINT a(x)
  240 FOR t = 0 TO tempo: NEXT t: x = x + 1: IF x = (y * 6) + 1
THEN x = y * 6 - 5:

  250 k$ = INKEY$
  260 IF k$ = CHR$(27) THEN STOP
  270 IF k$ = CHR$(46) THEN tempo = tempo - dp: SOUND 3000, .2:
  280 IF tempo <= np THEN tempo = np:
  290 IF k$ = CHR$(44) THEN tempo = tempo + dp: SOUND 1500, .2:
  300 IF tempo >= vp THEN tempo = vp:
  310 IF k$ = CHR$(49) THEN y = 1: GOTO 220
  320 IF k$ = CHR$(50) THEN y = 2: GOTO 220
  330 IF k$ = CHR$(51) THEN y = 3: GOTO 220
  340 IF k$ = CHR$(52) THEN y = 4: GOTO 220
  350 IF k$ = CHR$(53) THEN y = 5: GOTO 220
  360 IF k$ = CHR$(54) THEN y = 6: GOTO 220
  370 IF k$ = CHR$(55) THEN y = 7: GOTO 220
  380 IF k$ = CHR$(56) THEN y = 8: GOTO 220
  390 IF k$ = CHR$(57) THEN y = 9: GOTO 220
  400 IF k$ = CHR$(48) THEN y = 10: GOTO 220

  410 GOTO 230
```

Следующий этап — замена светодиодов клавиатуры на разноцветные сверхъяркие. Здесь необходимо быть аккуратным и соблюдать полярность включения светодиодов. Результат стоит затраченного времени.

Кстати, в бездонных сетях Интернет есть готовая программа управления светодиодами клавиатуры — "LedSwitcher 1.2", которая работает совместно с Winamp — наиболее популярным музыкальным проигрывателем. Программа сконфигурирована как плагин для Winamp любых версий, после ее установки ваши светодиоды Num, Caps и Scroll Lock будут мигать под музыку. LedSwitcher имеет настройки типа (громкость/эквалайзер/баланс) и частоты (от 0 до 127) мерцания, что, в свою очередь, позволяет настроить его под любую песню. Для данного плагина версия Windows, Winamp и модель клавиатуры не имеют значения, хотя все же для Windows NT\2000\XP лучше использовать аналогичный плагин — Keyboard Lights, имеющий более расширенные настройки. LedSwitcher использует очень мало вычислительных ресурсов, что позволяет установить его на любой (!) компьютер. Автор программы — Alexei Team, а скачать ее из Интернета можно по ссылке http://noenemy.by.ru/LedSwitcher.zip. Программа бесплатная, язык интерфейса — русский.

Теперь остается только привязать более мощную и высоковольтную нагрузку (например, лампы накаливания или дискотечные софиты) к маломощным и низковольтным светодиодам клавиатуры с помощью оптронов и симисторов — так называемого силового блока, чтобы получить полноценную светомузыкальную установку.

Схема силового блока (**рис.1**) подключается, к клавиатуре, и обеспечивает с примененными в ней оптотранзисторами напряжение оптоизоляции 1500 В, что гарантирует безопасное совместное использование блока с компьютером.

Доработка клавиатуры состоит в установке на ней в любом подходящем месте малогабаритного разъема X1 и распайке его согласно схеме. Светодиоды оптронов подключаются параллельно светодиодам клавиатуры. Нагрузка на контроллер клавиатуры возрастает незначительно, и поэтому даже круглосуточное ее использование совместно с силовым блоком не нарушает ее нормальной работы (это проверено на практике на разных моделях клавиатуры). Когда загорается любой из светодиодов клавиатуры, включается соответствующий ему оптотранзистор DA1, который открывает аналог тринистора на VT1, VT2, тем самым активизируя VS1 и включая в работу нагрузку (схема данного оптоблока построена по мотивам статьи [2]). Нагрузку до 200 Вт можно использовать без радиатора.

Печатная плата одного из идентичных блоков (OPTOBLOCK 1...3) представлена на **рис.2**. Если симистор предполагается эксплуатировать с мощной нагрузкой, радиатор к нему можно изготовить из подходящего по размерам алюминиевого корпуса электролитического конденсатора. При желании все платы можно объединить в одну

Рис. 1

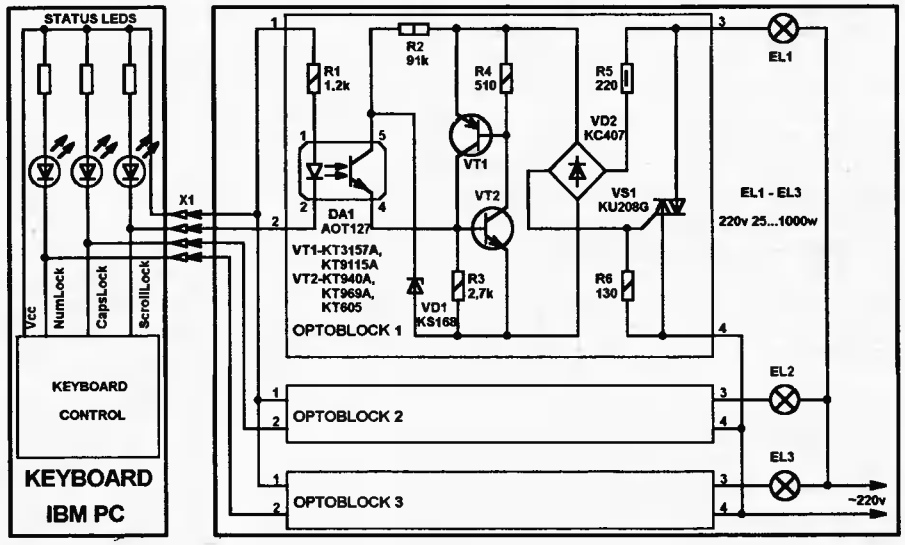
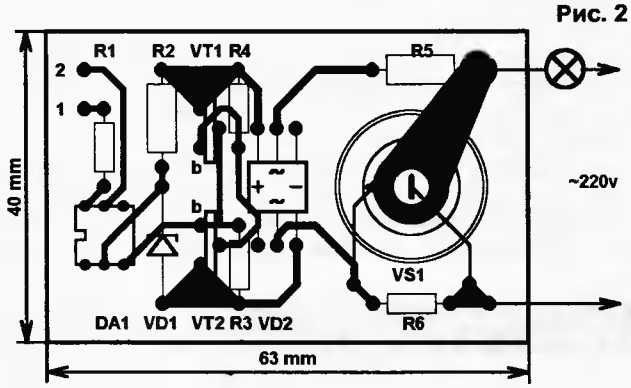


Рис. 2



законченную конструкцию. Следует помнить, что на плате присутствует опасное для жизни напряжение сети 220 В, и неукоснительно соблюдать правила ТБ при сборке и наладке конструкции.

Литература

1. А.Долгий. Клавиатура IBM PC. — Радио, 1997, №№4, 6.
2. А.Бутов. Аналог оптосимистора. — Радиоаматор, 2002, №7, С.39.

Проверка корректности файловой структуры дисков TR-DOS

И.РОЩИН,
г.Москва,
ZXNet: 500:95/462.53
E-mail: bestview@mtu-net.ru
WWW: http://www.ivr.da.ru

Введение

По различным причинам (прерывание дисковых операций, связанных с записью на диск; использование программ, неправильно работающих с диском; ошибочные действия пользователя при ручном редактировании каталога и т.д.) файловая структура диска TR-DOS может оказаться некорректной.

Если работающая с диском программа не проверяет правильность файловой структуры (а в общем-то, и сама TR-DOS этого не делает — проверяется лишь специальный байт в служебном секторе, указывающий на принадлежность диска TR-DOS), то работа с "неправильным" диском может привести, вообще говоря, к непредсказуемым последствиям. Возможно и разрушение программы в памяти, и потеря информации на диске. А если, скажем, в каталоге для какого-то файла неправильно указан его размер или номер начального трека/сектора, то попытка работы с этим файлом, теоретически, может привести даже к повреждению дисковода при попытке позиционирования на несуществующий трек.

Ясно, что своевременное обнаружение ошибок в файловой структуре было бы очень кстати. Значит, нужна процедура, проверяющая правильность каталога диска. Такая процедура может быть использована как в специальной программе проверки дисков, так и вообще в любой программе, работающей с диском — если перед дисковыми операциями прочитать каталог и проверить его корректность, это поможет избежать неприятных последствий. А если ваша программа выполняет операции, связанные с записью на диск, работая непосредственно с каталогом, то вызов этой процедуры будет полезен и перед записью измененного каталога на диск — если из-за ошибки в программе записываемый каталог не является корректным, это будет сразу же замечено. Ваша программа, таким образом, будет проверять саму себя!

Теория

Кратко расскажу о формате дисков TR-DOS — эта информация пригодится для понимания логики работы программы.

На каждом треке (они нумеруются с 0) находится 16 секторов (также нумеруются с 0). Длина каждого сектора — 256 (#100) байтов.

Для системных нужд выделен нулевой трек, а область с первого трека и до конца диска используется для хранения содержимого файлов.

В секторах 0...7 нулевого трека находятся элементы каталога, содержащие информацию о записанных на диске файлах. Каждый элемент каталога занимает 16 байтов, его структура описана в **табл.1**.

Всего каталог содержит #800/16=128 элементов, т.е. на диске может быть максимум 128 файлов. Используемые элементы каталога (соответствующие реальным файлам и удаленным файлам) расположены в его начале, неиспользуемые — в конце.

Табл. 1

Смещение от начала	Длина	Комментарий
0	8	Имя файла. Если первый байт равен 0 — значит, элемент каталога не используется; если равен 1 — элемент соответствует ствртому файлу
8	1	Расширение файла (или первый символ расширения, если используется 3-символьное расширение)
9	2	Стертовый адрес фейла (или последние две символа расширения, если используется 3-символьное расширение). Для бейсик-файлов — длина программы и переменных
11	2	Длинв файла в байтах. Для бейсик-файлов — длина программы без учета длины переменных
13	1	Длина файла в секторах
14	2	Дисковый адрес начала файла (первый бейт — номер свктора, второй байт — номер трека)

Каждый файл занимает целое число секторов. Файлы расположены друг за другом без промежутков, именно в том порядке, в котором они перечислены в каталоге. Первый файл начинается с трека 1, сектора 0.

Табл. 2

Смещение от начала	Длина	Комментарий
#00	1	Этот байт должен быть равен 0 для правильного определения конца каталога, когда на диске 128 файлов
#E1	2	Дисковый адрес первого свободного сектора (первый бвйт — номвр сектора, второй байт — номер трека)
#E3	1	Тип дискеты: - #16 — 80 трвков, 2 стороны; - #17 — 40 треков, 2 стороны; - #18 — 80 трвков, 1 сторона; - #19 — 40 трвков, 1 сторона. (Здесь имеются в виду физические треки, а не логические)
#E4	1	Общее количество файлов (включая удаленныв)
#E5	2	Количество свободных секторов (сначала младший байт, потом — старший)
#E7	1	Идвнтификвционный код TR-DOS (байт #10)
#F4	1	Количество удаленных файлов
#F5	8	Имя диска (вообще говоря, может занимать и все 11 байтов до конца сектора)

Сектор 8 каталога (служебный сектор) содержит информацию о диске в целом (**табл.2**). Не все байты этого сектора используются.

Как видим, информация, представленная в каталоге, избыточна. Например, в описателе файла можно было бы обойтись без указания дискового адреса начала файла (так как для первого файла он известен — трек 1, сектор 0, а для любого другого файла может быть вычислен, исходя из суммы длин в секторах всех файлов до него). Многие параметры, находящиеся в служебном секторе, также можно было бы не хранить там, а вычислять, исходя из содержимого элементов каталога.

С одной стороны, избыточность — это плохо, так как данные занимают больше места. Но, с другой стороны, эта же избыточность упрощает работу с данными (скажем, когда при записи нового файла надо узнать адрес первого свободного сектора, то вместо того чтобы вычислять его, складывая длины всех файлов, можно взять уже готовое число).

Именно за счет избыточности удается обнаруживать ошибки, сравнивая записанные в каталоге значения параметров с вычисленными значениями, а также проверяя, чтобы значения параметров находились в правильных диапазонах (номер сектора — в диапазоне 0...15, номер типа диска — в диапазоне #16...#19, и т.д.). Подробную информацию о производимых проверках вы можете узнать из комментариев в тексте программы.

Программная реализация

Сначала я написал функцию, проверяющую корректность каталога, на языке С (используя компилятор Turbo C 2.0), а потом уже перенес ее на ассемблер Z80. С одной стороны, это облегчило мне тестирование и отладку, а с другой стороны — наличие функции на С поможет тем, кому потребуется добавить проверку корректности каталога в программу для работы с дисками TR-DOS (или с файлами-образами таких дисков) не на ZX Spectrum, а, например, на AMiGA или PC.

При вызове функции ей передаются два параметра: адрес каталога в памяти (т.е. адрес содержимого первых девяти секторов нулевого трека) и максимально допустимое количество треков на диске (в функции производится проверка, чтобы количество занятых секторов диска, сложенное с количеством свободных секторов, указанным в служебном секторе, не превышало количества секторов на ука-

занном при вызове функции количестве треков).

В принципе, можно было бы считать второй параметр равным 40, 80 или 160 трекам, в зависимости от типа диска (табл.2), и определять его в самой функции. Решение о передаче этого параметра при вызове функции было принято с учетом того, чтобы можно было проверять диски, отформатированные на большее количество треков, чем следует из значения типа диска, а также RAM-диски, количество треков на которых определяется объемом выделенной под них памяти.

Функция возвращает двухбайтное значение, где младший байт либо 0, если ошибок не обнаружено, либо код ошибки (**табл.3**), а старший байт — номер элемента каталога (нумеруются с 0), при рассмотрении которого ошибка была обнаружена (для тех кодов ошибок, которые помечены "*" в первом столбце табл.3). Если в каталоге несколько ошибок, то возвращаемое функцией значение будет соответствовать первой обнаруженной ошибке.

Табл. 3

Код ошибки	Комментарий
1	Неправильный идентификационный код TR-DOS
2	Первый байт служебного сектора не равен 0
3	Неправильный тип дискеты (не #16..#19)
4	Номер первого свободного сектора не является числом в диапазоне 0...15
5*	Обнаружен файл (возможно, удаленный), номер начального сектора которого не является числом в диапазоне 0...15
6	Фактическое общее количество файлов не совпадает с указанным в служебном секторе
7	Фактическое количество удаленных файлов не совпадает с указанным в служебном секторе
8*	Обнаружен используемый элемент каталога, следующий после неиспользуемого
9	Первый файл начинается не с трека 1, сектора 0
10*	Обнаружен файл (возможно, удаленный), дисковый адрес которого не равен дисковому адресу предыдущего файла, увеличенному на количество секторов в предыдущем файле
11	Указанный в служебном секторе дисковый адрес первого свободного сектора не равен дисковому адресу последнего файла, увеличенному на количество секторов в последнем файле
12	Файлов нет, но указанный в служебном секторе дисковый адрес первого свободного сектора — не "трек 1, сектор 0"
13	Количество занятых секторов диска, сложенное с количеством свободных секторов, указанным в служебном секторе, оказалось больше, чем содержится секторов на указанном при вызове функции максимальном количестве треков

Ниже приведен листинг функции.

```
#define byte unsigned char     /* 1 байт.  */
#define word unsigned int      /* 2 байта. */

/* ============================================================
       Функция test_cat - проверка правильности каталога TR-DOS.
Вход:  cat - адрес каталога в памяти,
       tracks - максимальное количество треков на диске.
Выход: 2-байтное значение, где младший байт - код ошибки или 0,
       если нет ошибок, а старший байт для некоторых кодов
       ошибок - номер элемента каталога, при рассмотрении
       которого была обнаружена ошибка.
============================================================ */

word test_cat (byte cat[0x900], byte tracks)
{
 byte files=0;       /* Общее количество файлов.          */
 byte del_files=0;   /* Количество удаленных файлов.      */
 byte element=0;     /* Номер текущего элемента каталога. */

 /* Проверка идентификационного кода TR-DOS. */
 if (cat[0x8E7]!=0x10) return (1);

 /* Первый байт служебного сектора должен быть равен 0. */
 if (cat[0x800]!=0) return (2);

 /* Тип дискеты должен быть числом в диапазоне 0x16-0x19. */
 if ((cat[0x8E3]<0x16)||(cat[0x8E3]>0x19)) return (3);

 /* Номер первого свободного сектора должен быть числом
    в диапазоне 0-15. */
 if (cat[0x8E1]>15) return (4);

 /* Подсчитываем общее количество файлов и количество удаленных
    файлов, одновременно проверяя, чтобы номер начального сектора
    каждого файла был числом в диапазоне 0...15. */

 while (cat[element*16]!=0)
 { /* Пока не обнаружим неиспользуемый элемент или
        не просмотрим весь каталог. */
  files++;
  if (cat[element*16]==1) del_files++;
  if (cat[element*16+0x0E]>15) return (5+(element<<8));
  element++;
 }

 /* Проверяем, правильно ли в служебном секторе указано
    количество файлов и количество удаленных файлов. */

 if (cat[0x8E4]!=files) return (6);
 if (cat[0x8F4]!=del_files) return (7);

 /* Если сейчас element=128, то просмотрен весь каталог, иначе
    остались неиспользуемые элементы, и надо проверить, чтобы
    все они были помечены как неиспользуемые. */

 while (element<128)
 (
  if (cat[element*16]!=0) return (8+(element<<8));
  element++;
 )

 if (files>0)
 (/* Файлы есть. */

  /* Первый файл должен начинаться с трека 1, сектора 0. */
  if ((cat[0x0F]!=1)||(cat[0x0E]!=0)) return (9);

  /* Дисковый адрес (т.е. начальный трек-сектор) каждого файла,
     начиная со второго, должен быть равен дисковому адресу
     предыдущего файла, увеличенному на количество секторов в
     предыдущем файле. */

  for (element=1;element<files;element++)
  (
   if ((cat[(element-1)*16+0x0E]+cat[(element-
              1)*16+0x0F]*16+cat[(element-1)*16+0x0D])!=
       (cat[element*16+0x0E]+cat[element*16+0x0F]*16))
   return (10+(element<<8));
  )

  /* Указанный в служебном секторе дисковый адрес первого
     свободного сектора должен быть равен дисковому адресу
     последнего файла, увеличенному на количество секторов в
     последнем файле. */

  if ((cat[(files-1)*16+0x0E]+cat[(files-1)*16+0x0F]*16+
       cat[(files-1)*16+0x0D])!=
      (cat[0x8E1]+cat[0x8E2]*16)) return (11);
 }
 else
 {/* Файлов нет. */

  /* Указанный в служебном секторе дисковый адрес первого
     свободного сектора должен быть "трек 1, сектор 0". */
  if ((cat[0x8E2]!=1)||(cat[0x8E1]!=0)) return (12);
 )

 /* Количество занятых секторов диска, сложенное с количеством
    свободных секторов, указанным в служебном секторе, не должно
    быть больше, чем содержится секторов на указанном при вызове
    функции максимальном количестве треков. */

 /* Преобразование к типу long нужно, чтобы избежать возможного
    переполнения при сравнении (для этого разрядность long должна
    быть больше двух байтов). */

 if
(cat[0x8E1]+cat[0x8E2]*16+cat[0x8E5]+(((long)(cat[0x8E6]<<8))&
     0xFFFF))>(tracks*16)) return (13);

/* Все проверки успешно пройдены - делаем вывод, что ошибок нет */

 return (0);
)
```

А вот аналогичная процедура на ассемблере Z80:

```
;-------------------------------------------------------------
;TEST_CAT – проверка правильности каталога TR-DOS.
;
;Вход:  каталог (#900 байтов) находится в памяти с адреса CAT,
;       A – максимальное количество треков на диске.
;
;Выход: A – код ошибки или 0, если нет ошибок,
;       D – для некоторых кодов ошибок – номер элемента каталога,
;           при рассмотрении которого была обнаружена ошибка.

TEST_CAT   EX   AF,AF'  ;Сохраняем количество треков.

;Проверка идентификационного кода TR-DOS.

           LD   A,(CAT+#8E7)
           CP   #10
           LD   A,1        ;*
           RET  NZ

;Первый байт служебного сектора должен быть равен 0.

           LD   A,(CAT+#800)
           AND  A
           LD   A,2        ;*
           RET  NZ

;Тип дискеты должен быть числом в диапазоне #16-#19.

           LD   A,(CAT+#8E3)
           SUB  #16
           AND  %11111100
           LD   A,3        ;*
           RET  NZ

;Номер первого свободного сектора должен быть числом в
;диапазоне 0...15.

           LD   A,(CAT+#8E1)
           AND  #F0
           LD   A,4        ;*
           RET  NZ

;Подсчитываем общее количество файлов и количество удаленных
;файлов, одновременно проверяя, чтобы номер начального сектора
;каждого файла был числом в диапазоне 0...15.

           LD   IX,CAT
           LD   BC,16
           LD   D,B        ;Общее количество файлов = 0.
           LD   E,B        ;Количество удаленных файлов = 0.

;Цикл будет выполняться, пока не обнаружим неиспользуемый
;элемент или не просмотрим весь каталог.

TEST_CAT_1 LD   A,(IX)
           AND  A
           JR   Z,TEST_CAT_3

           DEC  A
           JR   NZ,TEST_CAT_2
           INC  E
TEST_CAT_2 LD   A,(IX+14)
           AND  #F0
           LD   A,5        ;*
           RET  NZ

           INC  D
           ADD  IX,BC
           JR   TEST_CAT_1

;Проверяем, правильно ли в служебном секторе указано количество
;файлов и количество удаленных файлов.

TEST_CAT_3 LD   A,(CAT+#8E4)
           CP   D
           LD   A,6        ;*
           RET  NZ

           LD   A,(CAT+#8F4)
           CP   E
           LD   A,7        ;*
           RET  NZ

;Если сейчас D=128, то просмотрен весь каталог, иначе остались
;неиспользуемые элементы, и надо проверить, чтобы все они были
;помечены как неиспользуемые.

           LD   E,D        ;E – общее количество файлов.

TEST_CAT_4 BIT  7,D        ;D=128?
           JR   NZ,TEST_CAT_5

           LD   A,(IX)
           AND  A
           LD   A,8        ;*
           RET  NZ

           INC  D
           ADD  IX,BC
           JR   TEST_CAT_4

TEST_CAT_5 INC  E
           DEC  E          ;E=0?
           JR   Z,TEST_CAT_9

;Файлы есть.
;Первый файл должен начинаться с трека 1, сектора 0.

           LD   HL,(CAT+#0E)
           DEC  H
           LD   A,H
           OR   L
           LD   A,9        ;*
           RET  NZ

;Дисковый адрес (т.е. начальный трек-сектор) каждого файла,
;начиная со второго, должен быть равен дисковому адресу предыдущего
;файла, увеличенному на количество секторов в предыдущем файле.

           LD   IX,CAT+16
           LD   D,1        ;N текущего элемента каталога.

TEST_CAT_6 LD   B,(IX-1)   ;N трека предыдущего файла.
           LD   C,(IX-2)   ;N сектора предыдущего файла.
           CALL MAKE_CDA
           LD   A,(IX-3)   ;Кол-во секторов в предыдущем файле.
           ADD  A,C
           LD   C,A
           JR   NC,TEST_CAT_7
           INC  B

TEST_CAT_7 LD   H,B
           LD   L,C        ;HL=BC.
           LD   A,D
           CP   E          ;Все файлы обработаны?
           JR   Z,TEST_CAT_8

           LD   B,(IX+15) ;N трека текущего файла.
           LD   C,(IX+14) ;N сектора текущего файла.
           CALL MAKE_CDA
           SBC  HL,BC      ;Флаг C сброшен после MAKE_CDA.
           LD   A,10       ;*
           RET  NZ

           LD   BC,16
           ADD  IX,BC
           INC  D
           JR   NZ,TEST_CAT_6

;Указанный в служебном секторе дисковый адрес первого свободного
;сектора должен быть равен дисковому адресу последнего файла,
;увеличенному на количество секторов в последнем файле (т.е.
;должен быть равен HL).

TEST_CAT_8 LD   BC,(CAT+#8E1)
           CALL MAKE_CDA
           SBC  HL,BC      ;Флаг C сброшен после MAKE_CDA.
           LD   A,11       ;*
           RET  NZ

           JR   TEST_CAT_A

;Файлов нет.
;Указанный в служебном секторе дисковый адрес первого свободного
;сектора должен быть "трек 1, сектор 0".

TEST_CAT_9 LD   HL,(CAT+#8E1)
           DEC  H
           LD   A,H
           OR   L
           LD   A,12       ;*
           RET  NZ

;Количество занятых секторов диска, сложенное с количеством
```

```
;свободных секторов, указанным в служебном секторе, не должно
;быть больше, чем содержится секторов на указанном при вызове
;функции максимальном количестве треков.

TEST_CAT_A LD   HL,(CAT+#8E5)
           LD   BC,(CAT+#8E1)
           CALL MAKE_CDA
           ADD  HL,BC
           LD   A,13                    ;**
           RET  C      ;Переполнение!   ;**

           EX   AF,AF' ;Максимальное количество треков.
           LD   B,A
           LD   C,0
           CALL MAKE_CDA

;HL должно быть меньше BC+1.

           SCF
           SBC  HL,BC ;HL=HL-(BC+1)
           LD   A,13                    ;***
           RET  NC                      ;***

;Все проверки успешно пройдены - делаем вывод, что ошибок нет.

           XOR  A
           RET

;MAKE_CDA - подпрограмма преобразования дискового адреса
;в компактный вид, более удобный для вычислений.
;
;Вход:  B - N трека, C - N сектора.
;Выход: BC - N трека*16+N сектора.

MAKE_CDA   LD   A,B
           RRCA
           RRCA
           RRCA
           RRCA
           LD   B,A
           AND  #F0
           OR   C
           LD   C,A
           LD   A,B
           AND  #0F
           LD   B,A
           RET
```

Длина этой процедуры — 228 байтов. Если в программе нужно лишь выяснить, есть ошибка или нет, а какая именно ошибка — значения не имеет, то процедуру можно упростить. Для этого надо исключить из листинга двенадцать строк, помеченных "*", заменить фрагмент из двух строк, помеченных "***", на следующие строки:

```
           SBC  A,A
           RET  NZ
```

и заменить фрагмент из двух строк, помеченных "****", на следующие строки:

```
           CCF
           SBC  A,A
           RET  NZ
```

В результате длина процедуры сократится на 25 байтов и составит 203 байта. Входные параметры останутся теми же, а выходные параметры будут такими: если флаг Z установлен, то все в порядке, а если этот флаг сброшен — значит, обнаружена ошибка.

В процедуре не используется самомодификация кода, так что при необходимости она может быть помещена в ПЗУ.

Литература

1. А.Ларченко, Н.Родионов. ZX Spectrum и TR-DOS для пользователей и программистов. — СПб.: Питер, 1994.

КОМПЬЮТЕРНЫЕ ХРОНИКИ

BestView 2.19

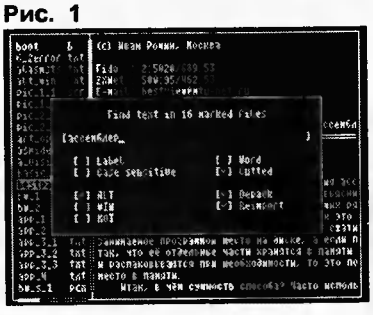
Итак, очередная версия известной программы для просмотра и запуска файлов на "ZX-Spectrum". Что нового?

Реализован поиск строки в указанных файлах (клавиша "F"), при этом можно использовать те же параметры, что и при поиске в просматриваемом тексте: искать метку в тексте ассемблерной программы, отличать прописные буквы от строчных, искать отдельное слово, искать строку, даже если слова в ней разбиты переносами.

Параметры "ALT", "WIN" и "KOI" (хотя бы один из них должен быть установлен) указывают, в каких кодировках искать строку.

Параметр "Depack" указывает, надо ли при поиске распаковывать упакованные файлы (обратите внимание — если упакованный файл по какой-либо причине не распаковывается при просмотре, то он не будет распакован и при поиске!). Параметр "Reimport" указывает, надо ли при поиске преобразовывать BASIC-, ASM-, HTML-файлы в текст, или же следует производить поиск непосредственно в содержимом этих файлов (**рис.1**).

Рис. 1

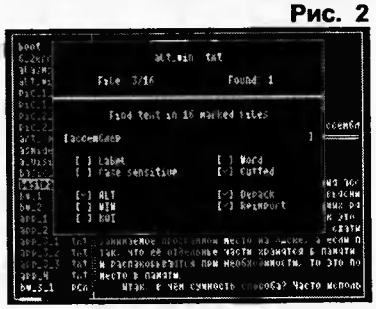


Как и при поиске в просматриваемом тексте, буквы "е" и "ё" не различаются (это удобно, когда неизвестно, используется ли в тексте буква "ё"), а также учитывается, что вместо русских букв "Н", "р", "у" в тексте могут быть соответствующие латинские (например, если текст получен из Fido или ZXNet).

Возможно, вы обратите внимание на то, что при поиске метки в файлах ZX ASM преобразование их в текст не производится, независимо от значения параметра "Reimport". Так сделано потому, что формат файлов ZX ASM позволяет искать в них метку непосредственно, без предварительного преобразования в текст (что гораздо быстрее).

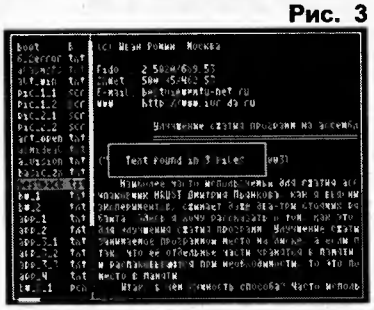
Файлы обрабатываются в том порядке, в котором они были помечены. В процессе поиска вы можете видеть имя текущего обрабатываемого файла, его порядковый номер, общее количество файлов, в которых производится поиск, и количество файлов, в которых был найден искомый текст (**рис.2**). Поиск можно досрочно прервать нажатием "Break".

Рис. 2



После окончания или досрочного прерывания поиска появляется окно с информацией о количестве файлов, в которых был найден искомый текст (**рис. 3**). Эти файлы остаются помеченными, а с остальных файлов пометки снимаются. Если изначально помеченных файлов не было, то есть поиск происходил в файле, на который указывал курсор, этот файл становится помеченным, если в нем был найден искомый текст.

Рис. 3



Клавиши "S" и "X" позволяют быстро перемещать курсор к ближайшему помеченному файлу в указанном направлении.

Кратко перечислю другие улучшения программы. В окне с просьбой подтвердить удаление одиночного (не помеченного) файла теперь выводится имя этого файла. В окне с просьбой подтвердить удаление помеченных файлов теперь выводится количество этих файлов. При вертикальном скроллинге каталога курсор не мерцает. Распознаются HRIP-архивы с расширением "hmp" (файлы-проекты Quick HyperText).

Также исправлены замеченные ошибки. Размер исполняемого файла увеличился на один килобайт (т.е. на четыре сектора).

Как обычно, последнюю версию BestView можно скачать с моей web-страницы: http://www.ivr.da.ru.

И.РОЩИН,
г.Москва,
ZXNet: 500:95/462.53
E-mail: bestview@mtu-net.ru
WWW: http://www.ivr.da.ru

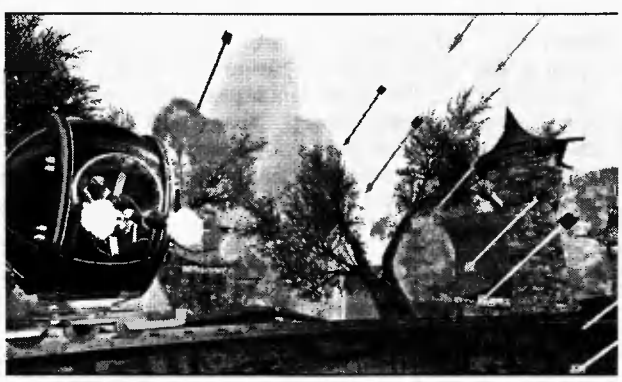
А.ВЕНДИЛОВСКИЙ,
г.Минск.

Armed and Dangerous

Автор не несет ответственности за достоверность описанного ниже. Хотя Дедушка Геймер, с чьих слов было записано это повествование, как и шесть санитаров психиатрической больницы, сопровождавших его, клялись и божились, что все это истинная правда от первого до последнего слова.

Давным-давно, в далекой Галактике... Стоп! Я сказал СТОП! Уберите эти титры, уплывающие с экрана вдаль! Для разнообразия сегодня мы поговорим об игре, выпущенной студией "LA" не по теме "Звездных войн". Вы удивлены? Вы в шоке? Не лезьте ко мне с вашим транквилизатором, я не сошел с ума. И господин Лукас тоже. Просто за последние года четыре эксплуатация этого бренда привела к тому, что большая часть игр просто ужасала своей третьесортностью и невероятно стойким подташниванием, при самом мало-мальском знакомстве. *(В этом месте санитары, спрятав свои лазерные мечи, печально покачали головами в знак согласия, а самый главный из них даже стянул со своей лысой головы шлем Дарта Вейдера — здесь и далее прим. Автора*) Предполагаю, что и большинству разработчиков этой некогда славной компании это тоже порядком надоело, а у некоторых упоминание о "Звездных войнах" вызывало еще большое желание, как выразились бы персонажи "A&D", положить на стол генерального большой болт, но терять место у кормушки они не желали. И решило руководство, чтобы разрядить атмосферу, создать игру-пародию на все свои предыдущие фильмы и разработки. В срочном порядке к созданию привлекли праздношатающуюся команду разработчиков "МДК" и разрешили всем оставшимся внести посильный вклад в этот "Звездновойный стеб". И закипела работа в компании, как никогда раньше...

Посмотрите на звездную карту, видите чуть левее пролетающий объект, чем-то напоминающий имперский эсминец? Это моя крыша, сорвало беднягу от буйного хохота. Думаю, в скором времени туда потянется не один косяк таких крыш... Ах да, не будем отвлекаться. Давным-давно, в далекой, но соседней галактике, существовало небольшое такое королевство с абсолютно неадекватными жителями. (*Санитары и подошедший главврач больницы еще раз утвердительно покачали головами, причем для убедительности главврач размахивал справочником по психиатрическим болезням*). И была у королевской семьи огромная и волшебная "Книга правил", в которой, как утверждалось, были записаны великие пророчества и указания, что нужно делать и когда, ну и еще, что было и что будет... Хотя некоторые утверждают, что это полная фигня, и на самом деле там был просто нарисован



план дворца и полинявшие за века анекдоты неприличного содержания в картинках. Выяснить это толком никому так и не удалось, так как по традиции в эту страну вторгся тиран (поправка — это у тирана была такая традиция, куда-нибудь вторгаться, а не у страны), который захватил власть. Тиран так себе — средней руки отморозок с весьма отмороженными подчиненными. Стал он тиранить себе народ потихоньку, но один обломчик покоя не давал — не мог он подчинить себе книгу правил, по какой-то причине она превратилась в огромное пособие по плетению корзин! Ну какой тиран потерпит подобное? Отдал он книгу монахам, строго-настрого приказав расколдовать ее. Но эффект вышел... несколько иной — книга как была пособием по плетению корзинок, так и осталась, а вот у самих монахов появилось непреодолимое желание плести корзинки! Любых форм и размеров. (*Без комментариев*). Тогда тиран решил, что во всем виноват старый маг Рексус, что это он стоит за подобным безобразием

(*Трудно не догадаться, если на обложке книги написано "Принадлежит Рексусу", и на каждой странице примечание — "Тиран — идиот. Рексус"!*), и послал он своих гвардейцев на его поиски. В прошлом Рексус был велик — он был мастером Йода, Брома и Кальция одновременно. Но во время вторжения тирана с ним произошел несчастный случай, и, вследствие воздействия постороннего предмета, двигающегося с большой скоростью, на его голову, он: а) начисто забыл, что он сделал с книгой; б) стал вести себя несколько неадекватно (*почему просто не сказать, что на него с крепостной стены уронили кирпич, и он стал вести себя как последний шизик?)*

И вот, когда черные-пречерные замыслы тирана Форджа уже вовсю сгустились над бедной страной, а отряды тирана подбирались к убежищу Рексуса, в игру вступает не менее клиническая команда по борьбе за справедливость, проходящая в полицейских хрониках как банда "Львиные Сердца".

Из досье полиции.

Банда "Львиные сердца" — состоит из трех человек.

Главарь — некто по имени Роман. Любит стрелять из всего, что может стрелять. Харизматично-сексуален (*здесь уже поддакивать и кивать головами в знак согласия стали весьма симпатичные медсестрички...*).

Любит бороться с несправедливостью, поэтому грабит награбленное и передает его несправедливо обиженным — то есть себе.

Большой Крот по кличке .... — знает о взрывчатке все, любит взрывать все, простоват и прямолинеен, не блещет умственными способностями, и вечный брюзга.

Третий — еще более харизматичная личность, огромный боевой робот с манерами английского аристократа, считающий, что единственный способ, обрести нирванну — это научиться понимать всю полноту вкуса чая. Пьет чай в любой свободный момент времени, встроил внутрь себя чайник.

Решили "Львиные сердца", что пора им подняться в рейтинге популярности народонаселения, исполнить пророчества, освободить родину от тирана, стянув у него Книгу Правил (а заодно и себя обезопасить, а то, не ровен час, тиран расколдует Книгу и покончит с ее помощью с ними). Много веселых и, на первый взгляд, бесполезных подвигов придется им совершить, прежде чем им удастся добиться своей цели...

На первый взгляд, мы имеем экшен от третьего лица, выполненный в аркадном стиле, и пока головной мозг отдыхает, его спинному коллеге придется вовсю поработать. Правда, серому веществу в черепной коробке тоже найдется развлечение — игра ОЧЕНЬ смешная. То враги, умирая, так и норовят отмочить что-нибудь такое, что вызывает улыбку. То ваши сопартийцы такой выдадут прикольчик, что на пару минут приходиться нажимать на "Паузу", потому что от хохота играть становится невозможно. А ролики между уровнями... Со времен "Worms 2" я не видел ничего более забойного. Все пропитано сочным юмором и иронией над собственными проектами, которые весьма узнаваемы и так спародированы, что кажутся маленькими шедеврами. Графика несколько подкачала — полигонов маловато, время от времени они выпадают, и можно заглянуть за внезапно ставшую прозрачной стену. Но это можно простить, так как есть одна потрясающая возможность — практически все на уровнях можно взорвать! Геймплей так и нашептывает, мол, порви их всех, как Тузик грелку! Зачем долго изгаляться в снайперской стрельбе по противнику, который засел в окне второго этажа? Да просто взорви к чертям это здание двумя минами-липучками, благо, их всегда много, и просто проводи взглядом улетающего вдаль врага. Деревья тоже рушатся, а самое главное — это бочки с порохом! Они всегда имеются на уровне — один точный выстрел — и можешь любоваться далеко в небесах пролетающими бочками, пламенем и весьма неосторожным противником, для которого этот полет станет последним. А как взрывается казарма... Для нетерпеливых есть возможность поработать с захваченными вражескими пулеметами и пушками, превратив пасторальный деревенский вид в лунный ландшафт. Да, чуть не забыл, если в миссии вам рекомендуют действовать как можно более скрытно, это означает, что ничего целого на уровне остаться просто не должно — в каждом доме прячутся враги, поэтому, не жалейте взрывчатки и патронов! Кстати, об оружии. Его не очень много, винтовку с автоматом еще можно узнать, а вот остальные образчики... Переделанный в гранатомет тромбон — страшная штука. Но страшнее всего акуломат — под землей к врагу плывет огромная акула, которая неожиданно выскакивает и пожирает его, славно чавкая и причмокивая, а потом сразу начинает искать следующую жертву. Но если врагов слишком много, а бомбы-липучки закончились, в дело идут их старшие сестры: микро-черная дыра — засасывает все живое не хуже пылесоса и самая "веселая" бомба-перевертыш. Как огромный штопор, она вкручивается в землю и... все переворачивается, там где была земля, теперь небо, куда сломя голову начинают падать ваши враги, пока все не возвращается на круги своя, земля становится на свое место, и улетающие в небесную даль враги не менее быстро начинают падать вниз. Это очень умиротворяющее зрелище, поверьте мне!

Хотя в бой вы вступаете вместе со своими коллегами, о партийцах можете особо не заботиться, т.к. любую миссию можно пройти абсолютно самостоятельно, и если даже кого-то убьют, то он автоматически воскреснет в начале следующего уровня. Уровни не очень большие, и хорошо чувствуется, что делали их те же товарищи, что и "МДК" — один ранцевый двигатель просто выбивает слезу ностальгии. В остальном бежать можно только туда, куда надо по сюжету, правда, всегда есть парочка маршрутов специально для тех, кто считает, что нормальные герои всегда идут в обход!

Это стоящая игрушка, в которую рекомендую поиграть всем — получите массу удовольствия. Только не верьте разработчикам, что игру можно пройти, расстреляв всего двадцать две тысячи патронов, я проверил и пересчитал — двадцать семь тысяч триста восемьдесят три штуки! И ни патроном меньше!

Тут Дедушка Геймер стал сильно волноваться, подскочившие санитары скрутили его, главврач прочитал вслух системные требования к игре и курс лечения, Дедушку увезли, и все разошлись... играть в **Armed and Dangerous.**

Игра предоставлена интернет-магазином "MediaCraft" www.mediacraft.shop.by

КУПЛЮ, ПРОДАМ, ОБМЕНЯЮ

Для публикации бесплатных объявлений ***некоммерческого характера*** *о покупке и продаже компьютерных комплектующих и программ, их текст можно присылать в письме по адресам:*

119454, г.Москва, а/я 37 или
220095, г.Минск-95, а/я 199,
передавать по тел. в Минске ***(017) 249-41-47****, либо через E-mail:*
rl@radiopage.by
rm@radio-mir.com
WWW: ***http://radio-mir.com***

Куплю книги М.Гука:
Интерфейсы ПК: справочник — С.-Пб: Питер-Ком, 1999.
Аппаратные средства IBM PC. Энциклопедия. — С.-Пб: Питер, 2000.
93120, Украина, Луганская обл., г. Лисичанск, ул. К. Маркса, 267, Рубашка В.Н.
E-mail: radan1970@mail.ru

Меняю 20 кг радиодеталей на ПК.
Тел. в г.Казани (8432) 98-58-53.

Клуб RU-QRP **примет в дар** ноутбук уровня "Пентиум-1".
E-mail: master72@lipetsk.ru

Куплю ПК "Орион-128/256" или печатную плату для его сборки, программное обеспечение для него на дискетах 5,25I.
425568, Республика Марий Эл,
Куженерский р-н, д.Шудумарь, Нагаав М.А.

Продаю коллекцию CD для "Sega Dreamcast" и "Panasonic 3DO" (различные эмуляторы и программы, много игр и фильмов); кабель для подключения "Dreamcast" к компьютерному монитору. Вышлю по почте. Могу выслать список имеющихся дисков (вложите в письмо конверт с обратным адресом).
442150, Россия, Пензенская обл.,
г.Нижний Ломов, а/я 58, Перепелкин Сергей.

Куплю шлейфы (можно б/у) к принтеру "Электроника МС6312"; программы системные, игровые для "Профи 3+".
231800, Гродненская обл., г.Слоним,
ул.Шоссейная, 18 — 69, Синельников М.В.
Тел. 8-10375-156-24-74-81.

Продаю коллекцию CD-дисков для "Sega Dreamcast" и "Panasonic 3DO" (различные эмуляторы и программы, много игр и фильмов), кабель для подключения "Dreamcast" к компьютерному монитору. Вышлю по почте. Могу выслать список имеющихся дисков (вложите в письмо конверт с обратным адресом).
442150, Пензенская обл., г. Нижний Ломов, а/я 58, Сергей.

Продаю:
- микросхемы новые (серий от 134 до 1810);
- диоды, стабилитроны, транзисторы, (большой выбор);
- автомобильный телевизор "Мара-2", новый (270x140x70 мм, размер по диагонали — 7 см);
- дисковод MF580 (Венгрия) с дискетами 5,25I;
- ПЭВМ "Сантака-001";
- блок питания импульсный +5 В/17,5 Вт и +12 В/48 Вт;
- блок питания +5 В/10 Вт.
Тел. (017) 243-02-97, Саша.

Куплю процессор ATHLON только с разъемом SLOT-A.
E-mail: santinal@yandex.ru

Школьник **примет в дар** дистрибутив Delphi версии 5 или более поздней. Дмитрий.
E-mail: svidinskii@iren.ru

Продам CD-диски. Список дисков:
www.m0l.nm.ru/shop_cd.htm

Подписные индексы:

Радиомир

- для жителей России и стран СНГ (кроме Беларуси): 48996 — подписка по каталогу Агентства "Роспечать" (**72370** — годовая), **25785** — подписка по Объединенному каталогу "Пресса России" с адресной рассылкой (электронный адрес подписки в INTERNET - **www.presscafe.ru**);
- для жителей Беларуси: 48996, 489962 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямым договорам").

Радиомир. КВ и УКВ

- для жителей России и стран СНГ (кроме Беларуси): 48924 — подписка по каталогу Агентства "Роспечать", (**71545** — годовая);
- для жителей Беларуси: 48924, 489242 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямым договорам").

Радиомир. Ваш компьютер

- для жителей России и стран СНГ (кроме Беларуси): 48925 — подписка по каталогу Агентства "Роспечать" (**45995** — годовая);
- для жителей Беларуси: 48925, 489252 (для организаций) по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямым договорам").

Кроме того, **предприятия** региона России, а также ближнего и дальнего зарубежья могут оформить подписку на журналы "Радиомир", "Радиомир. КВ и УКВ", "Радиомир. Ваш компьютер" через ООО "Корпоратиная Почта" по телефонам: (095) 953-92-62, 953-92-02, 953-93-20.

Получить подписку или заказать имеющиеся в наличии отдельные номера журналов можно из редакции. Текущие цены приведены в таблице.

Год	Можно заказать следующие номера журналов			Расценки на 1 экз. (с учетом пересылки)		
	Радиолюбитель	Радиолюбитель. Ваш компьютер	Радиолюбитель. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
1999	11	1 — 6, 10, 11, 12	6	20	700	5
2000	3 — 6, 8 — 12	2 — 8, 10 — 12	6, 7, 9, 11, 12	20	800	5
2001	4 — 6	2, 5, 6	2, 3, 5, 6	25	900	5
	Радиомир	Радиомир. Ваш компьютер	Радиомир. КВ и УКВ	По России и СНГ (рос. руб)	По Беларуси (бел. руб.)	По Украине (гривны)
2001	7 — 10, 12	7, 9 — 12	7, 8, 10, 11	30	1000	6
2002	1 — 12	1 — 12	1 — 3, 5 — 12	35	1500	7
2003	1 — 12	1 — 12	1, 5 — 8, 10 — 12	40	1800	8
2004	1 — 12	1 — 12	1 — 12	45	2100	9

Наши платежные реквизиты:
- для жителей России и стран СНГ (кроме Беларуси) —
получатель: ООО "Радиомир Пресс", ИНН 7729404741, КПП 772901001,
р/с 40702810900010000084 в ООО КБ "Агропромкредит", ф-л Центральный, г.Москва,
корр. счет 30101810500000000109, БИК 044525109
Адрес банка: 125315, г.Москва, Ленинградский пр-т, дом 76, корп. 4;
- для жителей Беларуси —
получатель: УП "РЛД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" в г.Минске код 121.
Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.

При оплате через Сбербанк в графе "назначение платежа" необходимо написать свой почтовый индекс, полный адрес, фамилию, имя и отчество полностью и точно перечислить, какие конкретно номера какого из журналов Вы заказываете.

При оплате почтовым переводом все сведения о заказе необходимо указать в графе "Для письма".

Для ускорения процесса получения журналов заказ можно продублироаать по *E-mail:* ***rm-sales@radio-mir.com.*** *Вся информация — там же или по тел. в г.Минске* ***(017) 221-01-10, (017) 505-13-65.***

Приобретение отдельных номеров журналов

В РОССИИ:

В ЗАО "Интерпресс — Экспресс": тел. в Москве (095) 208-65-50, 208-67-55, e-mail: inter@aif.ru, http://www.interpress-express.com.

В магазинах радиодетелей **"ЧИП и ДИП"** (единая справочная — тел. (095) 780-95-09):
- г.Москва, ул.Гиляровского, д.39 (ст. метро "Проспект Мира" — радиальная);
- г.Москва, ул.Ивана Франко, д.40, к.1, стр. 2 (платф.Рабочий поселок, 15 мин. от Белорусского вокзала);
- г.Москва, ул.Беговая, д.2;
- г.Ярославль, ул.Нахимсона, 12, тел. (0852) 27-57-15.

На радиорынках в Москве: Митинском (места R4, S8, K52, E50, G56), Царицынском (место 121).

В ООО "ТРЭНТЭКС": 111024, г.Москва, 4-я Кабельная, 2А (ст. метро "Авиамоторная"), тел. (095) 258-91-94, 258-91-95. E-mail: abook@mail.ru, abook@inbox.ru
На радиорынках: "Царицыно" (место 13/А), "Митино" (ряд 1, контейнер 17 и место Т-8); в ТК "Савеловский" (ст. метро "Савеловская", места А4, А5); на книжной ярмарке на "Тульской" (ст. метро "Тульская", место 515-19).

На УКРАИНЕ:

В Киеве в фирме **"Торм"**, тел. (044) 227-72-73.

В КАЗАХСТАНЕ:

В фирме **ТОО СП "Аргументы и факты. Казахстан"**: тел. в Алма-Ате (3272) 50-22-60, 50-21-64.

В БЕЛАРУСИ:

В Минске в магазинах **"Книга XXI век"**, пр.Ф.Скорины, д.92, тел. (017) 264-27-97 (ст.метро "Москоеская") и **"Глобус"**, ул.Володарского, д.16, тал. (017) 227-30-67 (ст.метро "Площадь Независимости").

Компьютеры для дома и офиса

от компании СитиПринт

Нужен ключ к успеху?
Добейтесь одновременного повышения эффективности и конкурентоспособности.

Intel, логотип Intel Inside, Pentium и Celeron являются зарегистрированными товарными знаками компании Intel Corporation

Начните использовать компьютеры от компании "СитиПринт" на базе процессора Intel® Pentium® 4 с технологией HT уже сегодня, и Вы получите возможность выполнять больше работы за меньшее время.

Подарите Вашему бухгалтеру точность высоких технологий.

СитиПринт
КОМПЬЮТЕРЫ И ОРГТЕХНИКА

220006, г.Минск, ул.Полевая 28-7А. Тел./факс (017) 2850134.
www.citiprint.by

RADIO РАДИО PAGE ПЭЙДЖ

БелРадиоПэйдж 233-00-00

МЕЖРЕГИОНАЛЬНАЯ ПЕЙДЖИНГОВАЯ СЕТЬ

Адрес: г.Минск, ул. Козлова 3; тел.: (017) 284-75-44, sales@radiopage.by

www.radiopage.by

- Ремонт и восстановление неисправных пейджеров
- Перепрограммирование пейджеров
- Большой выбор новых пейджеров (от 35000 р.)

7 способов отправки сообщений!

Стационарные охранные системы для дома и офиса

Получение на пейджер полной информации о состоянии охранной системы, установленной в квартире, офисе или котедже

АтлантМоторс

Спутниковые автомобильные охранные системы

Осуществление мониторинга за автомобилем при угоне: определение его географических координат по всему миру

Лицензия №55[illegible] от 23.03.06 МС