

радиомир

6 • 2003

Ваш компьютер



В Помощь Студенту

PDF-библиотека+софт



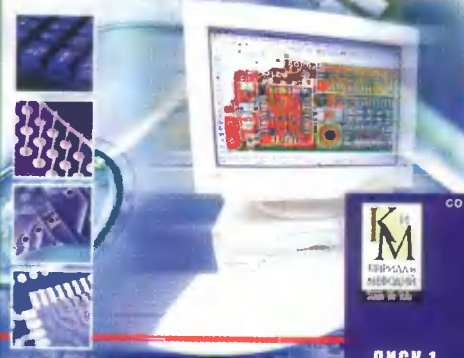
**БАЗОВЫЕ УЧЕБНИКИ
по высшей математике**



СПРАВОЧНАЯ СИСТЕМА ПО ВЫСШЕЙ МАТЕМАТИКЕ

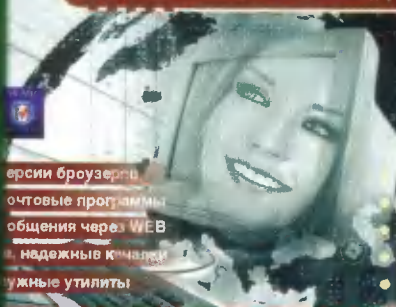
p-cad 2002

Full CD+ ДОПОЛНИТЕЛЬНЫЕ КОМПОНЕНТЫ



ВСЕЛЬ МИР ЧЕРЕЗ ИНТЕРНЕТ!

ОБЩЕНИЕ ЧЕРЕЗ ИНТЕРНЕТ

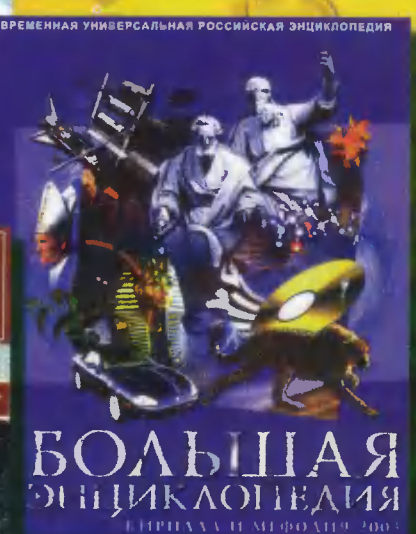


версии браузеров
почтовые программы
общения через WEB
надежные каналы
полезные утилиты

ALEX SOFT

Norton 2003 AntiVirus

лидер рынка антивирусного программного обеспечения



БОЛЬШАЯ ЭНЦИКЛОПЕДИЯ

СТЕПАНОВА И МЕЛЮХИНА 2003

Linux

НОВЫЕ LINUX-ТЕХНОЛОГИИ



- Corel Draw 9
- IBM DB2 Enterprise v7
- IPPanel Directory Server v4.12
- Unitree Central File Manager v2.3
- WebSphere Site Analyzer v3.5

ALEX SOFT



Microsoft Windows XP

шаг за шагом

загрузочный диск

Самоучитель по Windows XP
Microsoft Windows XP Professional
Microsoft Windows XP HotFixes SP1
Microsoft Windows XP Professional CD Boot Floppies
Norton Utilities 2002 Русская версия

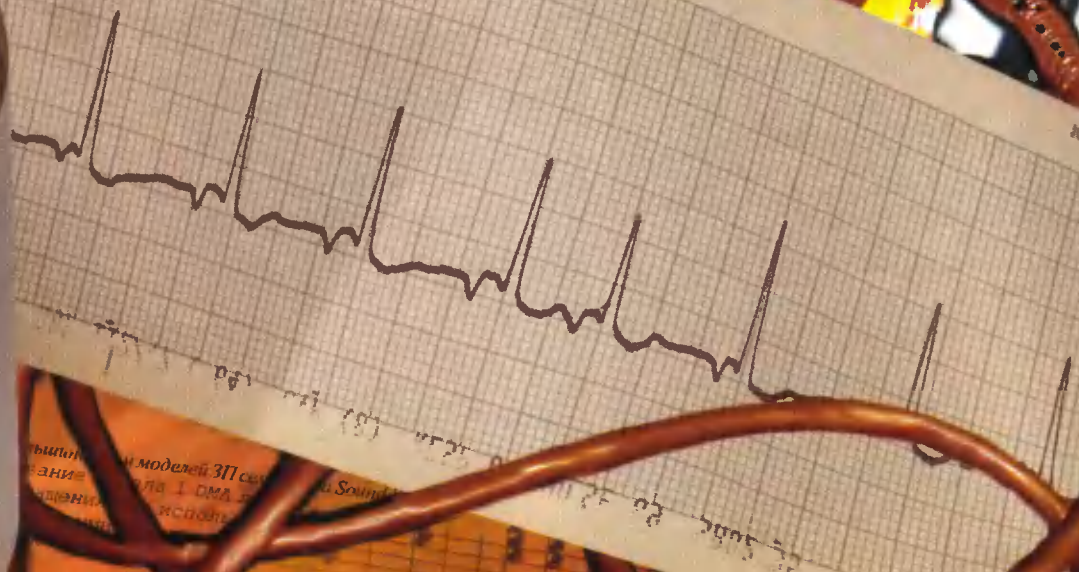


NEW CLIPART

ОТДЫХ

1000 ФОТО

Исследование УЗЧ аудиокарты



Учредитель и издатель: ООО "Радиомир Пресс"

Свидетельство о регистрации
ПИ №77-9841 от 17.09.2001

Главный редактор
Ольга СТРИЖАНКОВА
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) 131-97-17;
220095, РБ, г.Минск-95, а/я 199

E-mail: rl@radiopage.by
rm@radio-mir.com
WWW: <http://radio-mir.com>

Наши платежные реквизиты:
получатель: ООО "Радиомир Пресс",
ИНН 7729404741,
р/с 40702810900010000084
в ООО КБ "Агропромкредит"
ф-л Центральный, г.Москва,
корр. счет 30101810500000000109
БИК 044525109
Адрес банка: 113054, г.Москва,
ул.Зацепы, 43Б

Материалы для публикации принимаются
в рукописном, печатном и электронном
вариантах

Требования к графическим материалам
рекламного характера в электронном виде:
CorelDRAW до 9.0, все шрифты в кривых;
bitmaps 300 dpi; TIFF 300 dpi; CMYK.
Приложить печатную копию.

За достоверность рекламной и другой
публикуемой информации несут ответ-
ственность рекламодатели и авторы.
Мнение редакции не всегда совпадает
с мнениями авторов

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

Подписано к печати 25.04.2003 г.
Формат 60 × 84 1/8.
Печать офсетная, 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ № 1138.

радиомир Ваш компьютер

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

№6/июнь/2003

ЧИТАЙТЕ В НОМЕРЕ

ВОКРУГ ПК

НЕ ТОЛЬКО НОВИЧКУ

События и факты	2
НАШИ АВТОРЫ. СНИЛ кафедры системного анализа БГУ	3
А.ГРИНЧУК. Работа с редактором деловой графики Visio 2002.	
Оформление и печать диаграммы	4
Б.КИСЕЛЕВ. MATLAB. Работа в командном режиме	6
А.ГОРЯЧКИН. "Переключатель скоростей" для привода CD-ROM	8

ДАЙДЖЕСТ 10

ПРОГРАММЫ И АЛГОРИТМЫ

У ШКОЛЬНОЙ ДОСКИ

Олимпиадные задачи	14
--------------------------	----

УРОКИ ПРОГРАММИРОВАНИЯ

А.КОРБИТ, А.ЩЕРБАКОВ. Программирование в среде Visual C++ 6.0.	
Элементы управления Radio Button и Check Box	16
М.ДОЛИНСКИЙ. Решение задач на графах построением	
минимального остовного дерева	17
Sergio /HI-TECH. Низкоуровневое программирование	22

ДИАЛОГ ПРОГРАММИСТОВ

В.ГРИЦАН /HI-TECH. Современные профилировщики — что выбрать?	24
И.РОЩИН. Преобразование псеудографических	
таблиц в формат HTML	28
О.ВАЛЬПА. Электронная переписка	30

РЕЦЕПТЫ

С.РЮМИК. Чип-"невидимка" в "PlayStation"	33
И.ДЗЮБА. Исследование УЗЧ аудиокарты	36
Г.ПОЛОЗНЕВ. Элементы 316 вместо "больного" аккумулятора	38
Е.ЗАРЕЦКИЙ. Кое-что о Windows 9x, и не только	39

МИР 8 БИТ

Е.ИЛАСОВ. Конвертирование графики в iS-DOS на Sinclair	40
И.РОЩИН. Процедура сравнения строк на ассемблере Z80	42
Я.ОЧАКОВСКИЙ. Какие бывают ZX	43
Е.ЗАРЕЦКИЙ. Дела околоСпектрумовские	44

ИГРОТЕКА

А.ВЕНДИЛОВСКИЙ. Inquisition	45
Е.КУРИЛОВИЧ. Коды, читы	45

ДОСКА ОБЪЯВЛЕНИЙ

Куплю, продам, обменяю	48
------------------------------	----

Дорогие друзья!

Страницы журнала "Радиомир. Ваш компьютер" открыты для всех, кто хочет и умеет сделать общение с компьютером лучше, интереснее и полезнее. Поделитесь с коллегами своими идеями, схемными решениями, программами и советами.

Присылайте нам и свои вопросы — возможно, кто-то уже знает ответ. Многочисленному форуму наших читателей под силу найти решения самых сложных проблем. В общем, ждем ваших писем.

Редакция.

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



События и факты

Intel расстается с BIOS

Ни для кого не секрет, что на персональном компьютере могут быть установлены не только различные наборы программ, но даже и разные операционные системы. И только одна программа остается неизменной и присутствует на каждой машине — BIOS. Пользователи уже настолько привыкли к этому неизменному атрибуту компьютера, что кто-то уже успел забыть, а кто-то и не подозревает, что изначально BIOS планировалось использовать только на самых первых машинах, а впоследствии заменить ее чем-нибудь более совершенным.

Компания Intel уже заявила, что планирует отказаться от этого средства. На смену BIOS придет расширяемый фирменный интерфейс (Extensible Firmware Interface — EFI). Достоинство EFI в том, что он будет написан на языке C и представлять собой что-то вроде простейшей операционной системы, которая даже позволит выходить в сеть.

Microsoft хочет отказаться от традиционной файловой системы

По заявлениям Microsoft, "можно и даже нужно всю информацию хранить в одной громадной базе данных". Специалисты компании считают, что настало время отказаться от традиционной файловой системы и заменить ее базой данных, в которой могла бы храниться информация всех типов. Это значительно облегчило бы задачу поиска необходимых данных и увеличило бы надежность всей системы в целом.

Первый раз такая крамольная идея прозвучала из уст Джима Оллина еще в 1992 г., но внедрение единой БД с тех пор постоянно откладывалось. Сейчас о необходимости кардинального преобразования принципов хранения данных заговорил и CEO Microsoft Стив Баллмер. По его словам, дебют новой технологии (по крайней мере, ее части) состоится в очередной версии Windows под кодовым именем Longhorn, тестовая версия которой должна выйти в следующем году.

Главная проблема, встающая перед разработчиками — необходимость коренной переработки всех ныне существующих приложений для Windows, чтобы они могли работать с новой файловой системой.

Служебные программы должны быть встроенными

Компания Phoenix Technologies — известный разработчик BIOS для большинства материнских плат — сообщила о новой концепции обслуживания операционных систем.

Созданная этой компанией технология

получила название Core Managed Environment. "Изюминка" новинки заключается в том, что пользователю предлагается отказаться от специальных утилит для восстановления системы после ее краха — этим должна заниматься специальная встроенная программа. Для внедрения предлагаемой технологии производители компьютеров должны разместить на малоиспользуемых секторах винчестеров отдельный код, который не будет зависеть от установленных ОС. При падении системы пользователь сможет загрузить этот код и восстановить разрушенное.

Рутинному труду программистов приходит конец?

Процесс автоматизации программирования на сегодняшний день — самый актуальный. Ведь действительно, уже автоматизировали все и вся, а сам процесс написания программы продолжает оставаться рутинным занятием, которое выполняется "вручную".

Английская компания Appligenics заявила, что ей удалось создать программу, которая может самостоятельно писать приложения на стандартном языке Java. Разумеется, скорость написания текста этим виртуальным программистом значительно выше, чем у программиста из плоти и крови. Реальные цифры кажутся фантастическими — программа в состоянии сгенерировать дневную норму программиста-человека всего за четверть секунды. Вопрос оптимальности кода остается открытым.

CDR-ROM — новое слово в индустрии хранения информации

Как известно, оптические диски бывают трех типов — предварительно записанные, записываемые и перезаписываемые. Но уже несколько лет Optical Disc Corporation (ODC) ведет исследования в области создания технологии, позволяющей производить диски смешанного типа, часть полезной площади которых является предварительно записанной, а остальная площадь отдана в распоряжение пользователю. Теперь дело, наконец, дошло до опытного производства.

Диск CDR-ROM представляет собой обычный 120-миллиметровый диск, произвольная часть которого предварительно отпечатана на заводе-изготовителе и содержит некую информацию, которую нельзя изменить. Другая часть диска, емкость которой также произвольна, покрыта дорожкой, пригодной для однократной записи при помощи стандартного CD-R/RW-привода.

Вся сложность производства таких дисков заключается в изменяемой гео-

метрии профиля дорожки. Такой принцип устройства комбинированной болванки очень хорошо подходит для реализации всяких защитных решений. Интерес к разработке уже проявили различные правозащитные организации.

Fujitsu признала проблему

Инженеры компании Fujitsu признали, наконец, существование проблемы с жесткими накопителями серии MPG3, уже набившей оскомину и подорвавшей в сердцах пользователей доверие к компании. Как оказалось, при стечении ряда обстоятельств, влага, проникшая через защитную крышку винчестера, соединившись со смолой, дает в результате реакции фосфорную кислоту. Что происходит дальше, нетрудно представить — это агрессивное соединение поражает электрические проводники, окисляя их до полной потери проводимости. В конце концов, жесткий диск становится непригодным для использования, унося с собой в небытие "все, что найдено непосильным трудом". О методах спасения дисков Fujitsu умалчивает, но можно предположить, что помочь может дополнительная внештатная герметизация защитной крышки. Однако, возможно, наиболее правильное решение — срочно сохранить информацию с диска и заменить привод.

Новаяз наступает

Учителя Великобритании проявляют серьезную озабоченность в связи с тем, что традиционный английский язык начинает потихоньку терять популярность среди школьников.

Привычка общаться по ICQ и отправлять SMS-сообщения по любому поводу и без такового заставляет детей использовать сокращения при печатании текста. Потихоньку они к этому привыкают, и тогда нормальный язык кажется им очень длинным и неудобным.

В качестве примера происходящего агентство Reuters приводит отрывок из сочинения одного школьника, которое так и не смог прочитать его учитель. В частности, фраза "Я люблю Нью-Йорк" трансформировалась в "ILNY".

Русский язык официально признали в Сети

Английский язык является родным только для трети всех жителей Сети. Именно поэтому заинтересованные организации уделяют повышенное внимание созданию ресурсов для неанглоязычных пользователей. Ведь ни для кого не является секретом, что потенциальный покупатель не сделает покупку в интернет-магазине, языка которого он не понимает или понимает недостаточно хорошо для того, чтобы рисковать своими деньгами.



Компания VeriSign официально объявила, что к английскому, японскому, корейскому и китайскому языкам, на которых уже есть возможность регистрировать домены, добавились французский, немецкий, испанский и русский.

Файервол — не преграда

Канадский программист Оливер Лавери (Oliver Lavery) опубликовал работу, в которой утверждается, что существует возможность обхода правил персональных файерволов. Автором была написана программа, который позволяет внедрять код прямо в запущенный процесс без использования сторонних DLL и других библиотек.

Предложенный автором код как бы внедряется в установленное при помощи IE соединение на уровне сокета и активизирует сетевую деятельность, абсолютно невидимую для "датчиков" большинства современных персональных файерволов.

Согласно утверждениям самого автора, работа кода была проверена в ZoneAlarm Pro 3.5, Zero-Knowledge Freedom Firewall, Look'n'Stop 2.04, Sygate Personal Firewall PRO и Norton Personal Firewall 2003. Тестирование проводилось на машине с установленной операционной системой Windows XP Professional SP1.

Яндекс расширяет список форматов
До недавнего времени поисковая сис-

тема Яндекс могла работать только с документами в формате гипертекста, то есть с веб-страницами. Однако сеть состоит не только из html-файлов. Поэтому одна из популярнейших в Рунете поисковых машин начала осуществлять поиск и по файлам формата RTF и PDF. В результатах поиска найденные документы нестандартных форматов помечаются после названия (rtf и pdf соответственно). Найденный документ можно будет просмотреть в виде текста. Разумеется, форматирование при этом будет потеряно, однако смысл абсолютно не пострадает.

Подготовил Н.ПОТАПОВ,
E-mail: nic_potapov@mail.ru

НАШИ АВТОРЫ



Студенческая научно-исследовательская лаборатория (СНИЛ) кафедры системного анализа БГУ

На страницах журнала в течение последних пяти лет было опубликовано немало статей, подготовленных в этой лаборатории [1...19].

Все начиналось в 1996 г., когда на кафедру, где уже работала группа студентов старших курсов (О.Барановский, А.Гордеев, Т.Лашина), пришли энергичные и целеустремленные второкурсники А.Дедков, Р.Макарский, С.Нечаев. Активную работу талантливых молодых людей поддержали на кафедре. Научным руководителем этой команды стал заведующий кафедрой, д.ф-м.н., профессор В.В.Апанасович, а "играющим тренером" — старший преподаватель В.М.Лутковский. В апреле 1998 г. эта группа получила официальный статус студенческой лаборатории.

В лаборатории работали десятки студентов. В настоящее время многие из них уже работают на предприятиях Республики Беларусь. Другие продолжают учебу в европейских университетах или аспирантуре БГУ. Некоторые уже окончили магистратуру Эйндховенского технического университета (Нидерланды).

Главное направление исследований СНИЛ — искусственные нейронные сети и их применение в системах обработки информации. Очень быстро от уровня курсовых и обзорных работ сотрудники

СНИЛ поднялись на уровень международных конференций, таких как Вторая международная конференция по нейронным сетям и искусственному интеллекту ICNNAI-2001 (Минск, октябрь 2001 г.) и Шестая международная конференция по нейронным сетям и мягким вычислениям ICNNSC-2002 (Закопане, Польша, июнь 2002 г.). Достигнуто это было в значительной

степени благодаря усилиям П.Назарова и А.Балахонцева, работающих в лаборатории до настоящего времени.

На снимке (слева направо): А.Балахонцев, Р.Макарский, С.Нечаев, А.Дедков, П.Назаров, В.Большаков.

Список публикаций

1. Т.Лашина. Выбор метода защиты текстовых файлов. — Радиолучитель. Ваш компьютер, 1997, №6, С.16.
2. А.Дедков. Эти неслучайные случайные числа. — Радиомир. Ваш компьютер, 1998, №2, С.14.
3. А.Дедков, Р.Макарский. Электропитание: сюрпризы и тайны. — Радиолучитель, 1998, №10, С.10.
3. С.Нечаев С. MATLAB: Что может быть проще векторов и матриц? — Радиолучитель. Ваш компьютер, 2001, №9, С.10.
4. С.Нечаев, MATLAB: Базовые конструкции. — Радиомир. Ваш компьютер, 2001, №10, С.14.
6. П.Назаров. Распознавание символов с помощью персептрона. — Радиолучитель. Ваш компьютер, 1998, №8, С.2.
7. А.Балахонцев. Азбука нейронных сетей: методы обучения. — Радиолучитель. Ваш компьютер, 1998, №9, С.2.
8. А.Дедков, Р.Макарский. Что общего между телефоном и нейронной бомбой? — Радиолучитель. Ваш компьютер, 1999, №5, С.8.

9. В.Большаков, В.Лутковский. Мультимедийные технологии в обучающих программах. — Радиолучитель. Ваш компьютер, 1999, №6, С.34.

10. А.Дедков. Маленькое может стать большим. — Радиолучитель. Ваш компьютер, 2000, №2, С.11.

11. В.Лутковский, Т.Лашина, П.Назаров. Проектирование нейронной сети: задачи классификации и распознавания. — Радиолучитель. Ваш компьютер, 2001, №1, С.15.

12. В.Лутковский, П.Назаров. Проектирование нейронной сети: однослойный персептрон. — Радиолучитель. Ваш компьютер, 2001, №2, С.6.

13. В.Лутковский, П.Назаров. Проектирование нейронной сети: обучение однослойного персептрона. — Радиолучитель. Ваш компьютер, 2001, №3, С.6.

14. В.Лутковский, П.Назаров. Проектирование нейронной сети: достаточно ли одного слоя? — Радиолучитель. Ваш компьютер, 2001, №4, С.8.

15. В.Лутковский, П.Назаров. Проектирование нейронной сети: многослойный персептрон. — Радиолучитель. Ваш компьютер, 2001, №5, С.8.

16. В.Лутковский, П.Назаров. Проектирование нейронной сети: программные средства. — Радиолучитель. Ваш компьютер, 2001, №6, С.5.

17. Е.Лутковская, В.Лутковский, П.Назаров. Проектирование нейронной сети: от исходных данных до тестирования. — Радиомир. Ваш компьютер, 2001, №7, С.5.

18. В.Лутковский, Д.Миронов, П.Назаров. Проектирование нейронной сети для аппроксимации данных. — Радиомир. Ваш компьютер, 2001, №10, С.11.

19. П.Назаров, В.Лутковский. Проектирование нейронной сети Кохонена. — Радиомир. Ваш компьютер, 2001, №11, С.7.

Адрес СНИЛ:

220116, Минск-116, а/я 202,
E-mail: SRLSA@bsu.by

Информацию подготовила
Е. Лутковская.

Работа с редактором деловой графики Visio 2002

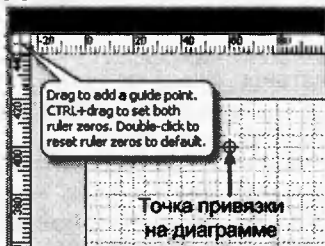
(Окончание. Начало в №№12/2002, 1-5/2003)

Оформление и печать диаграммы

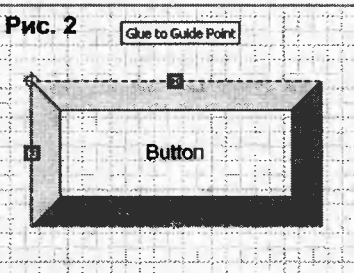
После изучения основных приемов работы в Visio 2002, нам осталось рассмотреть окончательную "доводку" готового документа. Условно здесь можно выделить три основные группы команд:

- команды точного позиционирования элементов диаграммы;
- команды выравнивания диаграммы;
- печать больших диаграмм как заключительный этап работы.

Рис. 1



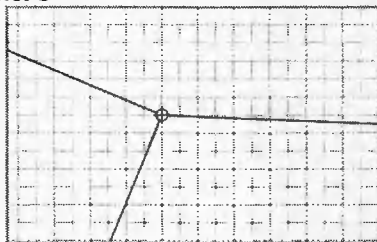
Точка привязки на диаграмме



Потребность в точном размещении фигур часто возникает при рисовании схем помещений, электрических схем и при создании иллюстраций к математическим текстам. Здесь основными элементами являются точка привязки (**Guide point**) и окно размеров (**Size & Position Window**). Точка привязки находится в левом верхнем углу окна рабочего листа, на пересечении вертикальной и горизонтальной линеек (рис.1). Достаточно навести на нее курсор мыши, нажать левую клавишу и перетянуть точку на рабочий лист. При печати или предварительном просмотре точка привязки не отображается, но позволяет "приклеивать" к ней другие фигуры (рис.2). Перетаскивание этого элемента с нажатой клавишей **Ctrl** позволяет изменить начало координат на диаграмме. Позиционирование фигур в Visio обладает одной особенностью — фигуры привязываются к координатной сет-

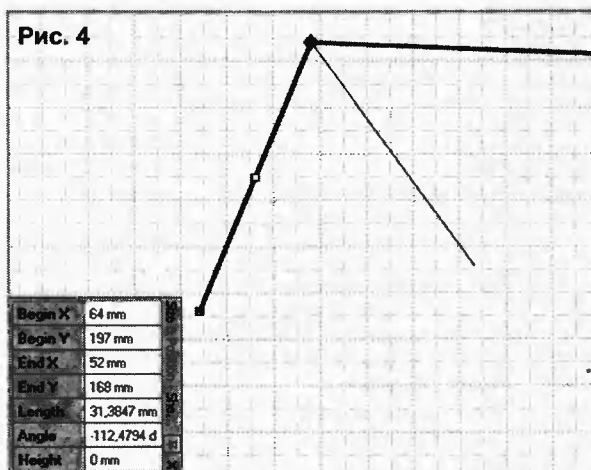
ке, и при малых масштабах изображения (25...75%) начинаются слишком резкие "прыжки". Для преодоления этого "препятствия" и предлагается выставлять (при масштабах 200...400%) точку привязки, а затем с ней связывать фигуры. Впрочем, есть еще один вариант ее использования. В Visio не предусмотрена возможность склейки двух или более

Рис. 3



линий, но зато легко делается склейка линий с **Guide point** (рис.3), и после этого можно перемещать концы всех линий одновременно.

Рис. 4



Однако описанные выше манипуляции с масштабами изображения — не лучший выход, когда вам нужно поместить фигуру в точку с заранее заданными координатами, и уж тем более, когда речь идет о повороте на заданный угол. Для

этих целей используется окно размеров и положения **View/Size & Position Window**. В этом окне (рис.4) указываются координаты центра выделенной фигуры (либо координаты начальных и конечных точек линий и стрелок), размеры фигуры (**Width** — ширина, **Height** — высота, **Length** — длина) и угол поворота (**Angle**). Это окно не только информационное, но и активное — данные можно изменять для перемещения или поворота объекта. В качестве примера разделим пополам угол на рис.4. С помощью окна размеров и положения выясняем, что угол наклона одной кривой равен 112,4794 deg (или, в привычных обозначениях — 112,4794°). Выделяем вторую линию и снова находим угол наклона (-8,7328°). Затем для третьей линии в поле **Angle** вводим полусумму этих углов: $(-112,4794 - 8,7328)/2 = -60,6061$ — рис.5. По аналогии можно задавать точное положение точки привязки, раз-

меры прямоугольника и т.д. При изменении размеров следует помнить, что у некоторых фигур установлена защита отношения длины к ширине, тогда изменение одного из этих параметров приведет к автоматическому изменению другого. Можно, правда, и снять защиту — **Format/Protection/снять флажок Aspect Ratio**, а затем изменять размеры независимо. Для дву-

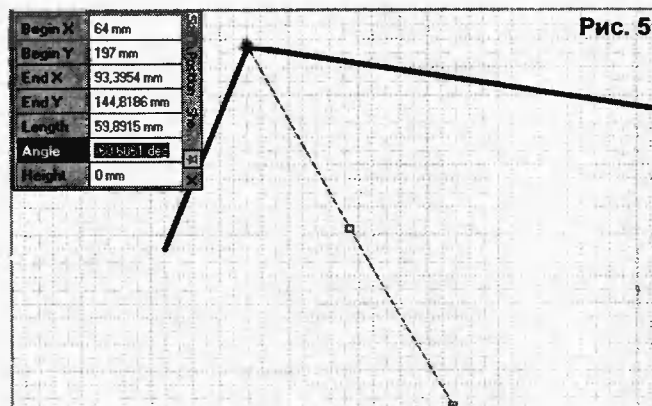


Рис. 5



мерных фигур в окне **Size & Position Window** появляется дополнительное поле **Pin Pos** — центр фигуры, т.е. уточняется, какую именно точку считать центром. По умолчанию — это **Center-Center** (геометрический центр фигуры).

Но в том случае, когда пользователя интересует не точное расположение элементов диаграммы, а их общее размещение, на помощь придут команды меню **Shape**. Для размещения диаграммы в центре листа применяется команда **Shape/Center Drawing**. Для автоматического выравнивания — **Shape/Layout Shapes**. Выбор этой

Рис. 6

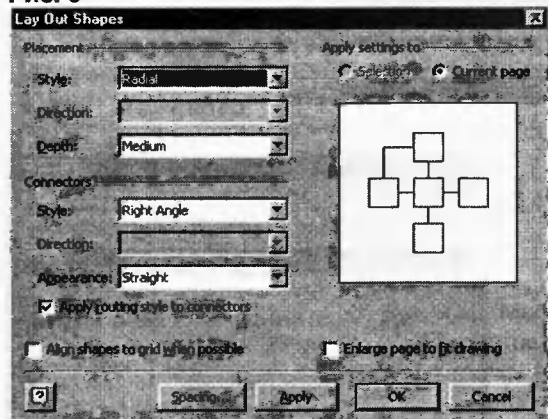
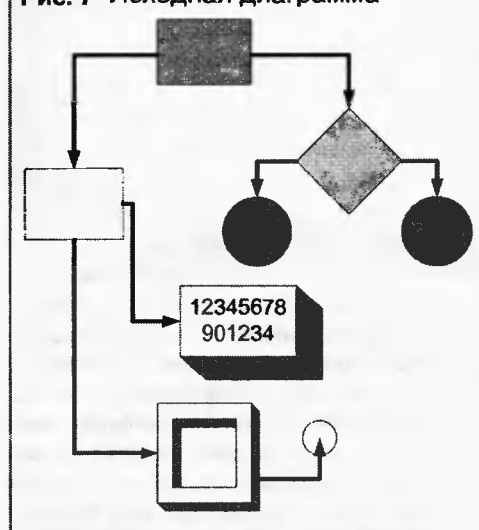


Рис. 7 Исходная диаграмма

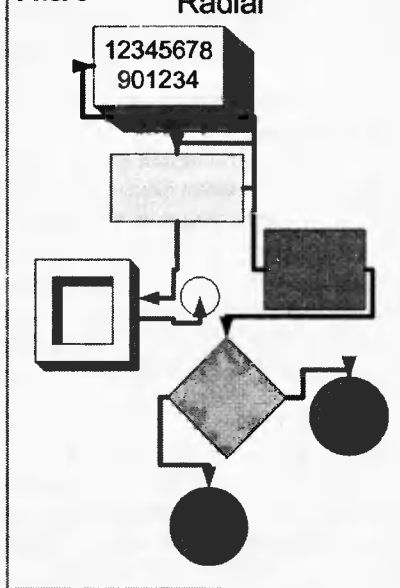


команды сопровождается появлением диалогового окна (рис.6). Здесь в разделе **Placement** (размещение) указывается общий стиль диаграммы (**Style**), направление (**Direction**) и глубина (**Depth**). Стиль определяет относительное размещение связанных фигур — здесь предлагается три варианта: радиальное (**Radial**), древовидное (**Flowchart/Tree**) и круговое (**Circular**). Направление диаграммы (слева направо, сверху вниз) — **Direction** — исполь-

зуется только при создании древовидной диаграммы. Параметр **Depth** определяет, сколько свободного пространства оставлять между фигурами. Выбор из списка **Shallow** приводит к выигрышу качества за счет неэкономного использования рабочей площади листа, **Deep** — наоборот, экономия за счет качества, и **Medium** — промежуточный вариант. В разделе **Connectors** можно настроить общий вид соединительных линий. На рис.7...9 показан результат манипуляций с готовой диаграммой. Не оставлено без внимания также и цветовое оформление — команда **Shape/Actions/Color Schemes** позволяет в единой манере изменить раскраску фигур и соединительных линий, за исключением тех фигур, для которых "вручную" применялась заливка.

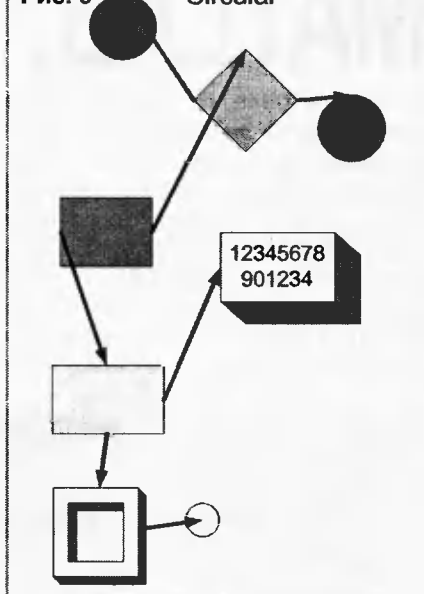
Заключительный этап работы — печать диаграммы. Сложности возникают только в той ситуации, когда размер диаграммы больше размера стандартной страницы. Но Visio 2002 достаточно умело обходит эти трудности. В режиме предварительного про-

Рис. 8



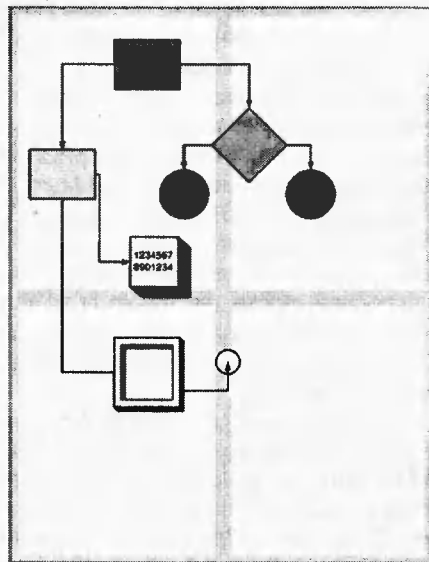
смотра (**File/Print Preview** — рис.10) на стыках страниц видны области серого цвета. Эти области печатаются на обеих смежных страницах. Поэтому после обрезания краев страниц они легко собираются и склеиваются в большую диаграмму, и создать большой плакат из листов формата A4 в Visio проще, чем, например, в Word (это проверено автором лично).

Рис. 9



Итак, мы рассмотрели основные приемы работы с Visio, начиная с создания простейших диаграмм и заканчивая публикацией результатов работы в Интернет и печатью диаграмм. Если вас заинтересовала эта программа, можно упомянуть те возможности, которые оказались за пределами нашего рассмотрения. Это использование

Рис. 10



математических выражений в свойствах фигур и рабочих страниц, а также большие, по сравнению с приложениями MS Office, возможности при создании макросов на языке Visual Basic. Но для большинства практических приложений можно обойтись описанными в данном цикле статей средствами (это, опять-таки, проверено автором на личном опыте).



MATLAB. РАБОТА В КОМАНДНОМ РЕЖИМЕ

Б.КИСЕЛЕВ,
г.Минск, БГАТУ, каф.ВТ.

Работа в среде MATLAB может осуществляться в двух режимах [1, 2]:

- командном (калькулятора), т.е. вычисления производятся непосредственно после набора очередного оператора;
- путем вызова программы, записанной на языке MATLAB.

В обоих режимах пользователю доступны практически все вычислительные возможности системы.

Командное окно

После запуска программы MATLAB, на дисплее компьютера появляется ее главное окно (рис.1), и система готова к проведению вычислений в командном режиме (режиме калькулятора). Окно содержит меню, инструментальную линейку с кнопками, указание текущей папки (Current Directory) и ряд подокон, набор которых можно изменять по составу и количеству с помощью меню View (Вид). Рассмотрим основное из подокон — командное окно (Command Window).

В нем отображаются символы команд, набираемые на клавиатуре, результаты выполнения этих команд, информация об ошибках выполнения программ и т.п. Признаком того, что программа MATLAB готова к восприятию и выполнению очередной команды, является наличие в последней строке текстового поля командного окна знака приглашения (>>), после которого стоит мигающая вертикальная черта.

В верхней части окна MATLAB размещена строка меню: File, Edit, View, Web, Window, Help. Назначение пунктов меню будет поясняться по мере необходимости. При этом последовательность действий будет указываться перечислением пунктов меню через тире. Выход из системы MATLAB осуществляется закрытием главного окна, или выбором File Exit MATLAB.

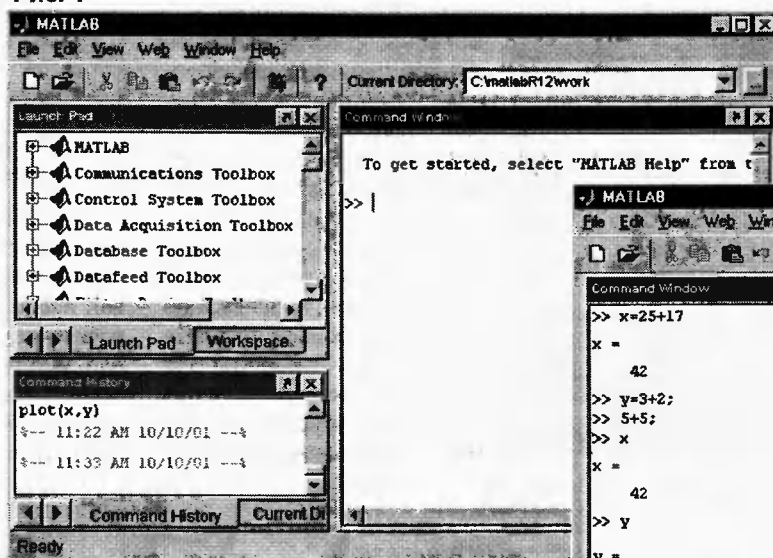
Пункт меню Help позволяет воспользоваться подробной справочной системой и демонстрационными примерами. На эти примеры можно также выйти, набрав в командном окне — DEMO. Для быстрой справки по какому-либо объекту надо в командном окне ввес-

ти — help имя объекта.

Если нет уверенности в правильном написании имени объекта, или оно неизвестно, можно использовать поиск по ключевому слову или по последовательности слов. Для этого можно использовать команду lookfor **Ключевое слово** или lookfor **'Ключевые слова'**

Существуют еще несколько команд для "быстрых" справок [1, 2].

Рис. 1



Простейшие действия в командном окне

Для арифметических операций используются обычные знаки: "+", "-", "*", "/", для возведения в степень принят символ "^", а для деления справа налево "\". Например, если ввести выражение

>> R = 4.5^2*7.23 - pi*10.4

и нажать Enter, получим ответ:
R =
113.7349

Здесь pi — это константа π. Видно, что введенное значение присвоилось переменной R. Кстати, имена переменных и других элементов должны начинаться с буквы, они могут иметь любое число символов (запоминаются первые 31), заглавные и строчные буквы системой **различаются**. Если формат вывода вас не устраивает, его можно установить через меню File — Preferences.

Введенная в окно информация сохраняется в стеке и снова может быть вызвана клавишей "↑", что удобно при корректировке выражений или повторных вычислениях.

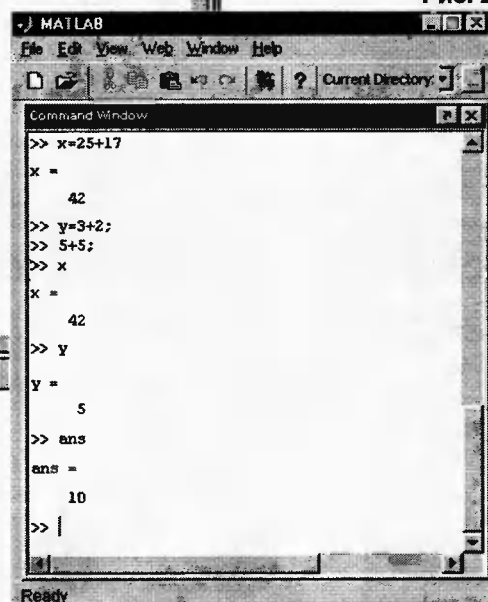
В общем случае форма вывода результата в командное окно имеет вид:

имя переменной = выражение

При этом вывод в командное окно подчиняется следующим правилам:

- если запись оператора **не** заканчивается символом ";", то результат его действия сразу отображается в командном окне;
- если запись оператора заканчивается символом ";", то результат его действия **не** отображается в командном окне;
- оператор может содержать только

Рис. 2



<выражение>, тогда значение результата присваивается специальной системной переменной ans;

- полученное значение можно использовать в последующих вычислениях под именем ans, но следует помнить, что значение этой переменной изменится при первом же встретившемся операторе без знака присваивания.

Примеры (рис.2) поясняют сказанное. Напомним, что вводимые пользователем выражения находятся в строках, которые начинаются с символа ">>", а в остальных строках — результаты.

Далее информация будет представляться не в форме копий экрана (как на приведенных рисунках), а просто в виде полужирного текста и без пропуска строк внутри одного примера. Так, действия, изображенные на рис.2, мож-

но продолжить следующим образом:

```
>> sin(pi/2)
ans =
    1
>>
```

Кроме уже известных имен **ans** и **pi**, используются и другие системные имена:

- **i**, **j** — мнимая единица (корень квадратный из -1);
- **eps** — погрешность для операций над числами с плавающей точкой (2^{-52});
- **realmin** — наименьшее число с плавающей точкой (2^{-1022});
- **realmax** — наибольшее число с плавающей точкой (2^{1023});
- **Inf** — машинная бесконечность;
- **NaN** — указание на нечисловой характер данных (Not-a-Number).

Примеры:

```
>> sqrt(-2)
ans =
    0 + 1.4142i
```

```
>> eps
ans =
    2.2204e-016
```

```
>> realmin
ans =
    2.2251e-308
```

```
>> realmax
ans =
    1.7977e+308
```

```
>> 2/0
Warning: Divide by zero.
ans =
    Inf
```

```
>> 0/0
Warning: Divide by zero.
ans =
    NaN
```

Общая форма вызова функции:
имя результата = имя функции
(список имен аргументов или их значений)

Имя результата может отсутствовать, тогда результат присваивается переменной **ans**. Применение как элементарных, так и других встроенных функций будет поясняться по ходу их использования.

Массивы

Рекомендуется внимательно разобраться эту тему, т.к. массивы играют центральную роль в языке MATLAB. Все данные представлены в виде массивов, например, даже простая переменная представляется как массив размером 1×1 .

Как и в других языках программирования, именованные наборы чисел называются **массивами**. Все массиву присваивается одно имя, а доступ к отдельным элементам массива осуществляется по целочисленному индексу, т.е. по номеру элемента в массиве. Правила

использования массивов аналогичны другим языкам. Основное отличие — массив заранее не определяется, а при вводе элементов допускается значительная свобода действий. Признаком массива являются прямые скобки, а элемента массива — имя с индексом в круглых скобках. Если массив представляет вектор-строку, то символы разделяются пробелами или запятыми, а если вектор-столбец — символами “;” либо переводами строк. Примеры:

```
>> a1 = [1 2 3]
a1 =
    1    2    3

>> a2 = [3;2;1]
a2 =
    3
    2
    1

>> [3;7:8]
ans =
    3
    7
    8
```

```
>> d = [4
        3]
d =
    4
    3
```

Массивы можно вводить частями, которые потом объединяются. Например, объединим определенные выше массивы **a1** и **a2**:

```
>> b = [a1 a2]
b =
    1    2    3    3    2    1
    3    2    1
```

Можно определять массив, прописывая каждый элемент:

```
>> A(1) = 2;
>> A(2) = 7;
>> A(3) = 5
A =
    2    7    5

Или еще “интересней”:
>> B(4) = 3
B =
    0    0    0    3
```

Если значения элементов изменяются с постоянным шагом, то для формирования массива удобно использовать оператор двоеточие — “:”, например:

```
>> x = 1:0.1:2
x =
Columns 1 through 7
    1.0000    1.1000    1.2000    1.3000
    1.4000    1.5000    1.6000
Columns 8 through 11
    1.7000    1.8000    1.9000    2.0000
```

Можно строить массивы, используя специальные встроенные функции. Например, получить массив из пяти единиц:

```
>> E=ones(1,5)
E =
    1    1    1    1    1
```

Для определения двумерных и многомерных массивов используются все те же приемы, а строки отделяются друг от друга записью с новой строки или символом “;”. Примеры:

```
>> [9 8 7
```

```
    6 5 4
    3 2 1]
ans =
    9    8    7
    6    5    4
    3    2    1
```

```
>> [1 2 3;4 5 6;7 8 9]
ans =
    1    2    3
    4    5    6
    7    8    9
```

Определим массив из единиц размером 3×5 :

```
>> E=ones(3,5)
E =
    1    1    1    1    1
    1    1    1    1    1
    1    1    1    1    1
```

По умолчанию элементы двумерного массива просматриваются по строкам, т.е. сначала изменяется второй индекс. В случае обработки многомерных массивов, сначала изменяется крайний правый индекс, затем второй справа и т.д.

Упомянутый выше оператор двоеточие (“:”) используется и для других целей. Он как бы осуществляет перечисление всех элементов по данному индексу. Поясняют это примеры формирования массивов **v**, **w**, **u** из определенного выше двумерного массива **ans**:

```
>> v = ans(:)
v =
```

```
    1
    4
    7
    2
    5
    8
    3
    6
    9
```

```
>> w = ans(:,1)
w =
```

```
    1
    4
    7
```

```
>> u = ans(2,:)
u =
```

```
    4    5    6
```

Пустой массив задается двумя квадратными скобками без символов между ними. Пустой массив можно, в частности, использовать для удаления элементов массива, например, удалить второй столбец из массива **ans**:

```
>> ans(:,2) = []
ans =
```

```
    1    3
    4    6
    7    9
```

Удалить третью строку из массива **ans**:

```
>> ans(3,:) = []
ans =
```

```
    1    3
    4    6
```

Аналогично строятся и используются многомерные массивы.

Далее, наряду с термином “массив”, будем использовать термины “вектор”, “матрица” и т.п.

(Продолжение следует)



А.ГОРЯЧКИН,

г.Кыштым, Челябинской области.

E-mail: anatoliy_goryach@mail.ru

Приводы CD-ROM отличаются от остальных узлов и компонентов компьютера наличием в своей конструкции движущихся механических частей. В отличие от жестких дисков, где весь механизм герметично закрыт, в приводах CD-ROM — все наоборот. Из-за этого внутрь приводов попадает пыль, которая загрязняет механизм и усугубляет его износ. Поэтому приводы CD-ROM первыми попадают в число устройств, которые выходят из строя в результате механического износа.

Современные приводы CD-ROM выпускаются с 40-, 48-, 50-, 52-, 54-, 56-кратной скоростью передачи данных. При такой высокой скорости не всегда обеспечивается требуемый уровень качества считывания дисков, особенно "пиратского" производства. При чтении поцарапанных и загрязненных компакт-дисков также происходят ошибки. При этом скорость считывания данных падает, и происходит многократное, и, как правило, малоэффективное повторное чтение диска.

Уменьшить в какой-то степени механический износ привода можно снижением скорости вращения компакт-диска. В результате применения специально разработанных программ-утилит появляется возможность управлять скоростью приводов CD-ROM. Кроме того, после ограничения максимальной скорости, чтение компакт-дисков происходит без повышенного шума, издаваемого приводами, улучшается читаемость и предотвращается разрыв некачественных дисков в скоростных приводах. Ограничение скорости приводов не сказывается отрицательно на качестве воспроизведения видеоизображения. Приводы CD-ROM, например, с 6-кратной скоростью обеспечивают поток данных, достаточный для воспроизведения видеоинформации с частотой от 25 до 47 кадров/с.

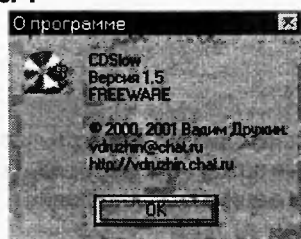
Среди многочисленных программ для ограничения скорости приводов CD-ROM хотелось бы отметить отечественную разработку — программу CDSlow версии 1.5 для управления скоростью приводов CD-ROM (рис.1). Разработчик этого программного продукта — Вадим Дружин, условия рас-

“Переключатель скоростей” для привода CD-ROM

пространения — freeware. Программа написана по мотивам утилиты фирмы ASUSTeK — CDSpeed (http://www.asus.com.tw/products/addon/vga/cdrom_drv.html), и использует тот же принцип. По сравнению с CDSpeed, утилита CDSlow имеет дополнительные функциональные возможности:

- корректно работает с приводами CD-ROM не только фирмы ASUSTeK, но и других фирм (во всяком случае, пытаются);
- работает не в окне, а в виде значка в System Tray, позволяя оперативно менять скорость привода CD-ROM;

Рис. 1



- может добавлять свой ярлык в папку “Автозагрузка” и устанавливать скорость сразу после старта Windows;

- может восстанавливать скорость после смены диска в приводе, при выходе из спящего режима и по таймеру;

- имеется функция открытия/закрытия привода (Eject/Close);

- существует встроенная возможность управления автоопределением диска.

Программа работает под Windows 95/98/ME, для работы под Windows NT/2000 необходимо установить драйвер ASPI (описание установки — в файле **nt2000.txt**). С некоторыми моделями приводов CD-ROM возможна некорректная работа. Это обусловлено тем, что различные модели приводов имеют разные возможности выбора скорости. В этом плане особо выделяются приводы ASUSTeK, которые позволяют выбирать любую скорость, от минимальной до

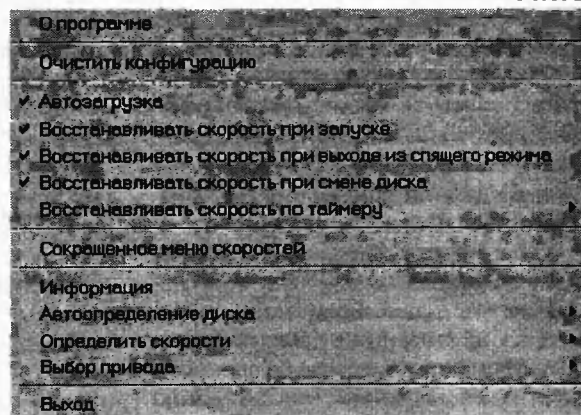
максимальной для этого привода с шагом 1. Некоторые модели приводов CD-ROM вообще не позволяют изменять скорость. Даже если в информации о приводе написано, что он позволяет устанавливать скорость, это говорит только о том, что привод CD-ROM принимает команду на установку скорости, но при этом реально может ее не менять.

Программа CDSlow очень компактна (размер exe-файла — 47 Кб, что в 8 раз меньше по сравнению с утилитой ASUSTeK) и состоит из одного файла **cdslow.exe**. Программа не требует инсталляции и допускает произвольное размещение в любом каталоге на жестком диске.

После запуска программы, в System Tray появляется значок с изображением компакт-диска. Затем можно приступить к настройке программы. Левая кнопка мыши позволяет установить ограничение скорости для выбранного привода CD-ROM или открыть/закрыть загрузочный лоток привода. Галочка в меню выбора скорости отмечает выбранную скорость, символ “>>” — текущую скорость привода. Если эти скорости совпадают, символ “>>” не появляется; если не выбрана скорость для привода — не появляется галочка. Значок “стрелка вверх” позволяет открыть лоток привода CD-ROM, значок “стрелка вниз” — закрыть. Перед использованием меню скоростей необходимо выбрать привод CD-ROM в меню настроек.

Правая кнопка мыши вызывает меню настроек (рис.2).

Рис. 2





Пункт меню "Выбор привода" позволяет задать привод CD-ROM, с которым будет работать программа. Даже если в компьютере имеется только один привод, его необходимо выбрать. Пункт меню "Определить скорости" позволяет заново произвести определение диапазона скоростей, поддерживаемого приводом CD-ROM, или задать список скоростей вручную. Эта опция может помочь, если автоматическое определение диапазона дало неверный результат при выборе привода в первый раз.

Режим "Быстрый перебор" повторяет процедуру определения скоростей, которая выполняется автоматически при первом выборе привода. Этот режим позволяет быстро определить скорости, поддерживаемые приводом при условии его корректной работы (в соответствии со стандартом MMC-1). Если привод обрабатывает команды установки скорости некорректно (но все-таки обрабатывает), может помочь режим "Полный перебор". Однако если возникает необходимость в использовании этого режима, на устойчивую работу программы с таким приводом рассчитывать не стоит.

Перед использованием меню "Определить скорости" рекомендуется вставить в привод компакт-диск с данными (не перезаписываемый), т.к. на некоторых приводах наличие диска влияет на результат определения скоростей. В случае, если пользователь точно знает, что привод CD-ROM поддерживает установку определенных скоростей, а программа эти скорости определить не может, можно отредактировать список скоростей, используя пункт меню "Задать вручную". Диалог редактирования скоростей показывает список скоростей в две колонки — скорость и команда, которую нужно дать приводу CD-ROM для установки этой скорости (у "нормальных" приводов эти колонки совпадают).

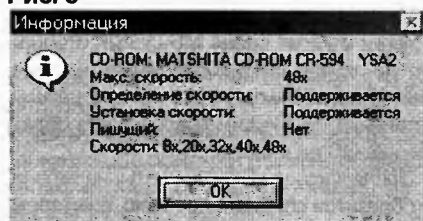
Меню "Автоопределение диска" позволяет управлять режимом автоопределения диска в Windows. Пункт "Автоматическое распознавание диска" позволяет изменить эту настройку в Windows, в свойствах привода CD-ROM. Если эта опция отключена, ни автозапуск, ни автовоспроизведение для этого привода выполняться

не будут, также для него не будет работать опция "Восстанавливать скорость при смене диска" в программе CDSlow. По возможности, следует изменять эту настройку стандартными средствами Windows, так как программа CDSlow может неправильно определять, для какого из приводов изменять настройку, если в компьютере установлено несколько одинаковых приводов CD-ROM. Если программа CDSlow установлена под Windows NT/2000, эта опция недоступна.

Пункт "Автозапуск программ" позволяет включать и отключать автозапуск для дисков, содержащих файл **autorun.inf**. Эта настройка влияет сразу на все приводы в системе. Под Windows NT/2000 для изменения этой настройки необходимо иметь права администратора.

Пункт "Автовоспроизведение музыки" работает аналогично предыдущему, но для музыкальных дисков. Пункт меню "Информация" показывает информацию о возможностях привода CD-ROM (рис.3).

Рис. 3



Пункт "Сокращенное меню скоростей" позволяет переключать меню скоростей в режим двухуровневого отображения, скорости отображаются в подменю группами по 10. Эта опция будет особенно полезна владельцам приводов фирмы ASUSTeK.

Пункт меню "Восстанавливать скорость по таймеру" позволяет установить интервал времени, по истечении которого программа будет восстанавливать скорость. Эта опция предназначена для приводов, которые самопроизвольно меняют скорость во время работы с диском. **Внимание!** Использование этой опции может вызывать нежелательные побочные эффекты, например, прерывание воспроизведения или перескакивание на другую дорожку при проигрывании музыкальных дисков.

Пункт "Восстанавливать скорость при смене диска" позволяет устанавли-

вать скорость после вставки диска в привод. Эта опция предназначена для приводов CD-ROM, которые сбрасывают скорость на максимум после смены диска (например, ASUSTeK). Для работы этой функции в настройках приводов CD-ROM в Windows должна быть включена опция "Автоматическое распознавание диска".

Опция "Восстанавливать скорость при выходе из спящего режима" позволяет устанавливать выбранную скорость после выхода из режима Suspend Mode, при котором все устройства, кроме CPU, выключаются.

Опция "Восстанавливать скорость при запуске" позволяет установить нужную скорость сразу после запуска программы.

Пункт меню "Автозагрузка" позволяет создать/удалить ярлык для программы в соответствующей папке и запускать программу при старте Windows.

Пункт меню "Очистить конфигурацию" удаляет все настройки программы из реестра Windows.

После окончания настройки программы при наведении на значок курсора мыши всплывает подсказка с указанием установленной скорости выбранного привода CD-ROM.

Значок программы в System Tray может изменяться в следующих ситуациях:

- "?" — не выбран привод CD-ROM;
- "!" — текущая скорость привода CD-ROM не совпадает с выбранной в настройках скоростью, либо скорость привода вообще не была выбрана;

- "песочные часы" — производится определение диапазона скоростей, поддерживаемого выбранным приводом CD-ROM, или выполняется какая-либо команда.

Если у пользователей возникнут какие-нибудь конструктивные предложения, они могут написать автору программы по адресу vdruzhin@chat.ru. При появлении новых версий программы CDSlow, они выкладываются автором программы на сайте <http://vdruzhin.chat.ru>.

Как видите, программа CDSlow имеет развитое меню настроек. Русскоязычный интерфейс делает ее простой и удобной в настройке и использовании.



ЛИСТАЯ СТРАНИЦЫ

Финляндия — родина операционных систем. Еще один талантливый финн написал еще одну ОС. Причем эта система, имея оконный графический интерфейс, уместается на одной дискете. Билл Гейтс, наверное, лишился сна. **О подвигах нового оосостроителя** рассказывает В.Холмогоров в статье "Маленький гигант" (CHIP, №2/2003, С.108...111).

По словам автора системы Вилле Турьянмаа (Ville Turjanmaa), идея заняться разработкой собственной ОС родилась в тот момент, когда он заметил, что изначально созданные на C или C++ компоненты Linux, будучи переписаны на ассемблере, увеличивают быстродействие системы в среднем на 10...40%. Решив построить принципиально новую платформу, основными критериями для которой являлись бы компактность и быстродействие, Турьянмаа взялся за дело, и уже 16 мая 2000 года он опробовал систему, которую назвал Menuet OS.

В настоящее время Menuet OS является вполне современной 32-битной многозадачной операционной системой с простым и удобным оконным интерфейсом, использующей низкоуровневый API и опирающейся на таблицу размещения файлов FAT32. Подобный подход весьма удобен со всех точек зрения, отмечает автор статьи, поскольку не требует от пользователя проводить повторное разбиение дисковых разделов и позволяет обращаться к файлам и папкам жесткого диска на компьютере, работающем, как правило, под управлением Microsoft Windows, без использования каких-либо специальных утилит.

Для запуска Menuet OS необходим компьютер с процессором не ниже Intel 80386, рекомендуются наличие 32 Мб оперативной памяти (нижний предел — 4 Мбайт), также необходима EGA-, VGA- или SVGA-совместимая видеокарта. В случае использования SVGA-адаптера рекомендуется плата, имеющая минимум 1 Мб памяти и поддерживающая стандарты VESA 1.2, VESA 2.0 или VESA 3.0. При наличии соответствующего монитора, такой видеоадаптер позволяет получить на экране изображение с разрешением 1280x1024 точки и передачей 16,7 млн. оттенков.

Menuet OS работает с жесткими дисками, использующими протокол LBA, распознает любую модель мыши, совместимую с Microsoft или Logitech Mouse и подключаемую к COM-порту, либо порту PS/2 (с поддержкой аппаратуры, использующей интерфейс USB, в Menuet OS, пока еще наблюдаются определенные сложности). Что касается звука, операционная система прекрасно работает с SB-совместимыми звуковыми картами (в случае использования такого адаптера возможен вывод стереозвук с частотой до 44,1 КГц). В

Menuet OS можно подключить и сетевую плату, но пока только типа Realtek 8029 PCI Ethernet Adapter. Но, наверное, главным достоинством Menuet OS является то, что сам дистрибутив этой операционной системы вместе с оконным интерфейсом и базовым набором прикладных программ занимает всего 1,44 Мб и, будучи установлен на 3,5-дюймовую дискету, позволяет загружать Menuet OS непосредственно с дискета, что автоматически исключает какую бы то ни было необходимость устанавливать систему на жесткий диск с созданием для нее собственного раздела и настройкой многовариантной загрузки. Благодаря этому каждый желающий может попробовать Menuet OS на своем компьютере и оценить ее достоинства. Эта операционная система бесплатно распространяется вместе с написанным на ассемблере исходным кодом на основе общедоступной публичной лицензии GNU.

Процедура установки Menuet OS достаточно проста. Дистрибутив поставляется в виде исполняемого файла MSETUP.EXE, в котором содержится образ операционной системы. Для инсталляции понадобится отформатированная дискета емкостью 1,44 Мб. Если работа проводится на компьютере, работающем под Windows 9x/ME/2000/XP, необходимо поместить дискету в дискковод и просто запустить файл MSETUP.EXE, после чего система автоматически откроет сеанс MS-DOS и отобразит на экране все необходимые инструкции. Следуя рекомендациям программы установки, необходимо распаковать образ Menuet OS на дискету.

После загрузки ОС с дискеты, на экране отобразится специальное окно, в котором будет предложено выбрать видеорежим, используемый Menuet OS. Для большинства современных видеоадаптеров можно указать режим VESA 2.0 (пункты 1...4 для различного экранного разрешения). Если на компьютере установлена более старая видеокарта, следует выбрать VESA 1.2 (пункты 5...8). Для очень старых компьютеров предусмотрены режимы VGA с палитрой в 16 цветов (пункт 0) и EGA/VGA с максимальной палитрой в 256 цветов (пункт 9). Следует отметить, что если в качестве используемой по умолчанию конфигурации указан стандарт VESA 2.0, на финальном шаге настройки системы необходимо также указать тип обращения к видеопамяти адаптера (LFB), выбрав требуемую настройку из двух вариантов — прямого или постраничного. Далее следует указать порт, к которому подключена мышь, и задать режим кэширования видеоданных MTRR. Последний параметр необходим для обеспечения аппаратного ускорения вывода визуальных изображений на экран монитора. Здесь рекомендации автора статьи следующие: если в компьютере установлен процессор класса 80386, 80486, Pentium или AMD K5, следует выбрать ре-

жим "2" (кэширование отключено), если же используется компьютер с процессором Pentium PRO/MMX, Celeron, Pentium II, AMD K6 и выше, нужно выбрать вариант "1" (кэширование включено). И, наконец, следует указать, откуда Menuet OS должна читать данные виртуального диска (ramdisk): при первом запуске системы рекомендуется выбрать режим "1 - floppy".

В отличие от других систем, Menuet OS не умеет самостоятельно определять подключенные к компьютеру оборудование, поэтому после окончания загрузки системы следует открыть окно настройки устройств и вручную указать некоторые параметры аппаратной конфигурации машины — в частности, тип подключения жестких дисков и привода CD-ROM (master/slave), характеристики звуковой карты (каналы DMA и запрос на прерывание), а также ряд других данных.

Обсуждая достоинства и недостатки ОС, автор отмечает, что главными достоинствами Menuet OS по сравнению с другими системами подобного класса являются ее бесплатность, компактность, быстродействие и нетребовательность к системным ресурсам. Это идеальная операционная система для малобюджетных школ, колледжей и иных учебных заведений, оснащенных устаревшей техникой. Поддержка этой системой кириллицы значительно упрощает настройку и конфигурирование компьютера, а многозадачность и феноменальное быстродействие Menuet OS позволяет загружать ее даже на машинах сочень "медленной" конфигурацией. К тому же, при помощи встроенного в оболочку операционной системы компилятора ассемблера можно превратить работающий под управлением Menuet OS компьютер в незаменимый "полигон" для обучения школьников основам программирования.

Среди очевидных недостатков ОС можно отметить пока еще чрезвычайно малый список поддерживаемого платформой оборудования, отсутствие прикладного программного обеспечения (за исключением приложений, входящих в комплект поставки), невозможность работы в Интернете, значительные ограничения в использовании локальной сети. Файловый менеджер Menuet OS неудобен, встроенный текстовый редактор более всего напоминает примитивную программу Edit для MS-DOS. В процессе работы с приложениями возникают проблемы при попытке масштабирования окон — в случае, если видеоподсистема настроена на высокое экранное разрешение, границы активного окна практически невозможно захватить мышью, а границы окон некоторых программ вообще масштабируются как угодно, но только не так, как это нужно пользователю.

Операционная система действительно позволяет обращаться к дисковым разделам, имеющим таблицу размещения файлов FAT, но, вместе с тем, файловый ме-



нджер почему-то демонстрирует содержимое только Primary DOS Partition, а куда исчезают все остальные логические диски — остается решительно непонятным. Возможно, дополнительные дисковые разделы требуют отдельного монтирования, но отсутствие толковой документации по Menuet OS не позволило автору определить механику этого процесса, к тому же, несмотря на продолжительные поиски, автор не нашел в системе привычную пользователям Linux полнофункциональную командную консоль.

В заключение автор отмечает, что Menuet OS пока еще находится в стадии разработ-

ки и предварительного тестирования, а потому она определенно будет развиваться и совершенствоваться. Еще рано предъявлять какие-либо серьезные претензии к создаваемой энтузиастом любительской разработке, которой, к тому же, едва исполнилось два с половиной года. Вполне возможно, что в следующих версиях Menuet OS будут полностью устранены все очевидные недостатки и добавится множество новых функциональных возможностей.

Для тех, кто желает опробовать новую ОС, предлагаем ссылки по теме, приведенные во врезке в статью:

- www.menuetos.org — официальный

сайт проекта;

- <http://menuet.narod.ru> — русскоязычный сайт Menuet OS: документация, статьи, ссылки;

- <http://sourceforge.net/projects/menuetos> — Menuet OS на nortane SourceForge;

- www.adcodex.com/embedded_html — сайт разработчиков TCP/IP для Menuet OS;

- <http://strony.wp.pl/wp/menuetos> — библиотеки, драйверы, утилиты для Menuet OS;

- <http://geocities.com/kirkalx/menquake> — проект, посвященный портированию игры Quake для Menuet OS;

- <http://fasm.sourceforge.net> — компилятор ассемблера для Menuet OS.

Если вы являетесь владельцем **устаревшего ноутбука**, и есть желание и финансовые возможности провести его апгрейд, рекомендуем прочитать статью "Операция "Ноутбук" (**CHIP**, №2/2003, С.152...158). Но имейте в виду, что заменить процессор в вашем компьютере не удастся.

Главная причина этого — путаница с форм-факторами. Ноутбуки со старыми процессорами Pentium II и Celeron (до 466 МГц) устанавливались в разъемы MMCX1 и MMCX2 (Mobile Module Connector). На них, кроме внешнего кэша второго уровня, были размещены даже чипсеты, так что замена такого процессора не имеет смысла.

Некоторые ноутбуки имели процессоры типа Mini Cartridge, которые сегодня уже не

могут подвергаться апгрейду, так как их максимальная частота — 400 МГц. Мобильные процессоры Pentium III с интегрированным кэшем L2 имеют формат BGA-1 и BGA-2 (Ball Grid Array — матрица шариковых выводов), но заменить их невозможно, поскольку они припаяны к материнской плате. Некоторым производителям ноутбуков это показалось неудобным, и они создали формат MicroPGA-1 и -2 (Pin Grid Array — матрица штырьковых выводов). Это, по сути, был тот же BGA, но только процессор соединялся с платой при помощи привычных штырьковых контактов.

В свободной продаже очень редко удается найти процессор для ноутбука. Причина проста — каждый ноутбук очень точ-

но "привязан" к тепловым характеристикам определенной группы процессоров. Если все же попытаться установить новый процессор, то моментально появится проблема перегрева компьютера. Это, в свою очередь, станет причиной многих неприятностей — от нестабильной работы системы до пожара. Каких-либо шагов по исправлению существующего положения Intel не предпринимает, что позволяет ей и производителям ноутбуков экономить тем самым на технической поддержке. Процессоры Pentium 4 имеют удобную контактную платформу. В принципе, их можно менять, только подобная замена вряд ли будет кому-то нужна в ближайшее время.

Одним из важных вопросов, возникающих при установке новой ОС, является **выбор типа файловой системы**. То, что не следует использовать FAT16 — это ясно. Ну а какая разница между FAT32 и NTFS? На этот вопрос пытается ответить *К.Данилов* в статье "FAT32 vs. NTFS. Что предпочтительнее?" (**Мир ПК**, №1/2003, С.60...65).

Файловая система — это методы и структуры данных, которые используются операционной системой для хранения информации на диске или в его разделе. При этом важными являются такие понятия как кластер и сектор. Сектор — минимальная единица пользовательских данных, которую могут использовать программы, работающие с жестким диском напрямую, в обход операционной системы. Кластер — минимальная единица пользовательских данных, которую могут использовать программы, работающие с жестким диском посредством операционной системы. В некоторых случаях они могут совпадать.

FAT32 разработана компанией Microsoft в конце 1996 г. Ее основой является 32-битовая таблица размещения файлов, что позволило уменьшить размер кластера и добиться поддержки больших по объему разделов, а также дало возможность сохранять в одном разделе свыше 65 тыс. файлов. FAT32 поддерживают все ОС семейства Windows начиная с Windows 95 OSR2, кроме Windows NT4.

Основными составляющими FAT32 являются:

- загрузочная запись (Boot Record), где хранится информация о размере раздела, количестве свободного места, размере кластера и т.д.;

- таблица размещения файлов (File Allocation Table, FAT). В этой таблице содержится информация о том, как располагаются данные файлов в разделе диска;

- корневой каталог (Root Directory). Это самый "главный" каталог на логическом диске, все остальные каталоги и файлы располагаются по иерархии ниже его. В FAT32 корневая папка (каталог) имеет некоторые особенности. Во-первых, она не имеет фиксированного адреса (хотя чаще всего начинается со второго кластера). Во-вторых, ее размер не ограничен, и под свои нужды она использует область данных пользователя.

Теоретически размер логического диска FAT32 ограничен 8 Тб. Но на практике встроенные в Windows 2000/XP средства администрирования дисков не позволяют создавать раздел размером более 32 Гб. В зависимости от размера создаваемого логического диска, кластер может иметь разный объем. Но чем больше кластер, тем менее эффективно используется место на жестком диске. Например, файл размером 10 байт реально займет один кластер. Следовательно, при большом размере кластера и значительном количестве неболь-

ших файлов потери могут стать существенными. С другой стороны, слишком маленький объем кластера приведет к увеличению размера таблицы размещения файлов до десятков мегабайт, а их все надо держать в оперативной памяти. Поэтому, если нет уверенности в том, какой размер кластера будет оптимальным, следует выбрать вариант, предлагаемый программой форматирования раздела.

Имя файла может содержать до 255 символов, с ISO-кодами (цифровым представлением символа) — от 33 до 255, за исключением специальных символов — "?", "!", " " и некоторых других. Максимально возможный размер файла составляет 4 Гб.

Прежде чем начать разговор о стабильности файловой системы, необходимо дать несколько определений. Логическими ошибками файловой системы являются неправильные сведения о размере файла, ошибочные данные в файле или некорректные имена файлов. Стабильность файловой системы определяется частотой, условиями появления и характером ошибок.

Еще один термин — журналирование. Так называется ведение журнала всех изменений и операций, производимых над данными. Журналирование помогает установить тип возможной ошибки, произошедшей при сбое в работе операционной системы, и используется при самовосстановлении файловой системы.

FAT32, в отличие от NTFS, не является



журналируемой, так что не стоит пренебрегать проверкой диска при перезапуске ОС, если до этого произошел сбой, повлекший некорректное выключение и перезагрузку операционной системы. Наиболее распространенная ошибка — неправильно записанные данные о количестве свободного места. Возникает эта ошибка в основном из-за того, что данные о количестве свободного места записываются в специальную отведенное место на диске, а не вычисляются, как это происходит в FAT16. Например, при копировании файла происходит сбой, ОС не успевает записать обновленные данные о свободном месте в разделе, хотя на самом деле оно уменьшилось. Журнал операций отсутствует, в результате чего невозможно определить последнюю выполнявшуюся процедуру. Остается один выход — полная проверка диска на наличие логических ошибок.

В целях профилактики полезно каждые два месяца проводить дефрагментацию, так как кластеры, в которых находятся данные файла, могут фрагментироваться (т. е. несколько кластеров, на которых записан длинный файл, располагаются не строго последовательно друг за другом, а разбросаны по диску). Со временем поиск следующего кусочка файла происходит все дольше и дольше. Ликвидацией последствий фрагментации занимаются специальные программы-дефрагментаторы, собирающие части разбросанных по диску файлов в один массив.

В апреле 1987 г. Microsoft и IBM начали совместную разработку новой операционной системы OS/2. Под эту ОС специально была разработана файловая система, призванная обеспечить стабильную и быструю работу с диском и облегчить труд администраторов. Но разногласия компаний привели к тому, что в сентябре 1990 г. сотрудничество было прекращено, и каждый пошел своей дорогой. В результате были выпущены OS/2 и файловая система HPFS (High Performance File System) от IBM и Windows NT с файловой системой NTFS (New Technology File System) от Microsoft. У файловых систем было много общего, и до версии Windows NT 3.51 включительно Microsoft обеспечивала в своих операционных системах поддержку HPFS.

На сегодняшний день из семейства Windows файловую систему NTFS поддерживают только те операционные системы, которые базируются на ядре NT. Это Windows NT 3.xx, Windows NT 4.0, Windows 2000 и Windows XP.

Базисом NTFS является главная таблица файлов (Master File Table, MFT). MFT изначально резервирует под себя одну восьмую часть раздела (примерно 12%). Если место на разделе заканчивается, MFT сокращается в два раза, освобождая для файлов пользователя свободное пространство. Процедура может повторяться несколько раз. При появлении незанятого

места MFT снова резервирует под себя 12% от объема раздела, что приводит к нежелательному эффекту — фрагментации MFT. При этом эффективность работы с NTFS-дискон падает.

В NTFS любой элемент является файлом, включая каталоги и главную таблицу файлов. Элемент состоит из двух частей: обязательной записи о нем в MFT и опциональных параметров, называемых потоками. Все данные файла представляют собой опциональные параметры (его содержимое, версия, дата последней модификации, автор и т.д.). Но наиболее известные файловые менеджеры дают пользователю информацию только об ограниченном и заранее определенном наборе потоков. А размер файла, показываемый пользователю, является объемом только одного потока, который, собственно, и представляет собой то, что мы привыкли традиционно называть данными файла. Получается, что текстовый документ, состоящий всего из нескольких страниц текста, может занимать не один гигабайт, скрытый в другом потоке.

Характерно также то, что файл, содержащий небольшие по объему данные, может иметь только ссылку на запись в MFT. В этом случае данные хранятся в главной таблице файлов вместе с информацией о самом элементе. Грубо говоря, весь файл находится внутри MFT. Каталоги в NTFS представляют собой ссылки на другие директории и файлы. Имя элемента данной файловой системы может содержать до 255 символов в кодировке Unicode. Размер логического диска практически неограничен, так как он может занимать до 2000000 Гб. Величина кластера не зависит от объема раздела и равна 4 Кб.

Одним из важных преимуществ NTFS является разграничение прав пользователей на файлы и каталоги. Правильное применение этой возможности повышает стабильность системы. Следует, однако, отметить, что разделение прав пользователей привязано к используемой операционной системе, и в другом семействе ОС права могут и не соблюдаться. В таком случае поможет шифрование.

Возможность шифрования была введена в Windows 2000. Если вы опасаетесь за конфиденциальность своих данных, вы можете зашифровать любой элемент файловой системы. Даже если жесткий диск попадет в руки людей, для которых информация не предназначалась, они не смогут ее извлечь. Взломать шифр, как известно, можно, и здесь уже вопрос лишь в том, насколько желанна ваша зашифрованная информация для взломщика.

Однако главное достоинство NTFS — журналирование и методы, которыми файловая система проводит операции с данными. Любое действие в разделе NTFS выполняется транзакцией. Транзакция — это пакет операций, который или выпол-

няется полностью, или не выполняется совсем. Любое действие с данными записывается в журнал; из него в случае какого-либо сбоя в дальнейшем можно узнать, какая транзакция не смогла успешно завершиться и почему. Основные объекты NTFS, ко всему прочему, зеркалируются, т. е. делается резервная копия загрузочной записи и некоторых элементов MFT. Такая логика операций с данными приводит к высокой стабильности файловой системы. Сбой во время дефрагментации, скорее всего, будет просто незаметен для пользователя, в то время как для FAT32 такая ошибка стала бы в большинстве случаев фатальной.

Несколько хуже у NTFS обстоят дела с фрагментацией, особенно когда диск заполнен более чем на 88%. Выход — в дефрагментации, но здесь есть проблема. Практически ни одна из созданных для этого программ не способна провести нормальную оптимизацию, поскольку возможности используемых ими стандартных функций ОС очень ограничены. В результате этот процесс придется повторять чуть ли не каждый месяц. Один из немногих, а может быть и единственный дефрагментатор, который способен исправить ситуацию — Speed Disk из пакета программ Norton Utilities. Его методы работы позволяют обойти ограничения, наложенные функциями ОС. Так что приходится выбирать: или не проводить дефрагментацию вообще, потому что оптимизацию раздела не этой утилитой можно назвать вредной, или использовать Speed Disk. Хотя падение производительности на NTFS из-за фрагментации гораздо менее заметно, чем в случае FAT32.

Итак, если у вас есть желание выбрать NTFS, подумайте, сможете ли вы в полной мере насладиться ее возможностями, предупреждает автор. Целиком использовать преимущества этой файловой системы под силу только опытному пользователю, да и на домашнем компьютере обычно нет необходимости в шифровании данных, разделении прав и выделении квот. Конечно, в качестве главного критерия может выступать надежность, но в этом плане ничто не сравнится с резервным копированием данных.

Существует еще одна хитрость, подстерегающая любителей поэкспериментировать над своим жестким диском. Дело в том, что если преобразование FAT32 в NTFS легко осуществимо средствами, например, Windows NT/2000/XP, то обратное преобразование без потери данных не сделаешь. Поэтому, прежде чем решиться на настойчивое предложение Windows переформатировать FAT32 в NTFS, следует хорошо подумать, так как единственным методом обратного преобразования будет копирование содержимого NTFS-раздела диска в другой раздел (или на другой жесткий диск) с последующим форматированием NTFS-раздела и его преобразованием в FAT32.





Олимпиадные задачи

Витебская областная олимпиада по информатике 2002-2003 гг. среди школьников

Задачи для 10-11 классов

Тур 1

Задача 1. "Икосаэдр"

Каждому жучку-паучку в возрасте одного года предстоит выдержать испытание: совершить наикратчайшее путешествие по граням икосаэдра — правильного многогранника с 20-ю гранями, пронумерованными от 1 до 20. Известно, что путешественник должен проходить через каждую грань икосаэдра только один раз. На каждой грани можно сделать паузу для отдыха. Общее время прохождения грани (в секундах) вычисляется путем умножения номера этой грани на ее порядковый номер в последовательности маршрута. Переход с одной грани на другую, соседнюю, занимает 1 с. Например, для путешествия по маршруту 12, 2, 18, 5, 15, 7, 17, 10, 8, 20, 14, 4, 11, 13, 1, 19, 3, 16, 6, 9 жучку-паучку понадобится 2176 с. Действительно, $2176 = 12 \cdot 1 + 2 \cdot 2 + 18 \cdot 3 + 5 \cdot 4 + 15 \cdot 5 + \dots + 16 \cdot 18 + 6 \cdot 19 + 9 \cdot 20 + 19$ (19 с добавлены для переходов с одной боковой грани на другую).

Две грани называются соседними, если они имеют общее ребро.

Напишите программу, которая определит номер наиболее благоприятной грани для начала маршрута и сам маршрут прохождения всех граней. Если таких маршрутов несколько, то получите любой из них.

Формат ввода:

Ваша программа должна прочитать из файла IKOSA.IN информацию о заданной нумерации граней икосаэдра в следующем виде:

1-я строка: $n[1,1]$ $n[1,2]$ $n[1,3]$ (через пробел), где $n[1,1]$, $n[1,2]$, $n[1,3]$ — номера граней, соседних с гранью под номером 1;

2-я строка: $n[2,1]$ $n[2,2]$ $n[2,3]$, где $n[2,1]$, $n[2,2]$, $n[2,3]$ — номера граней, соседних с гранью под номером 2;

...

20-я строка: $n[20,1]$ $n[20,2]$ $n[20,3]$, где $n[20,1]$, $n[20,2]$, $n[20,3]$ — номера граней, соседних с гранью под номером 20.

Формат вывода:

Ваша программа должна вывести в файл IKOSA.OUT результат решения задачи в следующем виде: в первой строке (через пробел) — номера граней в порядке их прохождения по кратчайшему маршруту, во второй строке — время прохождения кратчайшего маршрута.

Пример файла входных данных:

```
7 13 19 — номера граней, соседних с гранью под номером 1;
12 18 20 — номера граней, соседних с гранью под номером 2;
16 17 19
и т.д., всего 20 строк.
```

Задача 2. "Арифметическая прогрессия"

Напишите программу построения арифметической прогрессии, состоящей из N простых чисел.

Если таких прогрессий несколько, выведите любую из них.

Формат ввода:

Ваша программа должна ввести с клавиатуры значение N (N — натуральное число, $3 \leq N \leq 10$).

Формат вывода:

Ваша программа должна вывести на экран найденные N чисел по одному в строке.

Пример входных данных:

3

Пример выходных данных:

3

5

7

Задача 3. "Мотоциклисты"

Среди мотоциклистов города G популярно следующее соревнование. На одной из улиц города выбирается прямой участок, на котором установлено N ($1 \leq N \leq 100$) светофоров, имеющих только красный и зеленый цвета. Мотоциклист проезжает выбранный участок на постоянной скорости, причем нарушать правила дорожного движения, т.е. проезжать перекресток на красный свет, категорически запрещается, кроме случаев, когда мотоциклист пересекает перекресток в момент переключения светофоров. Мотоциклист, изменивший скорость движения, либо проехавший на красный свет, либо выбравший скорость, меньшую 5 м/с, дисквалифицируется. Максимальная скорость движения мотоциклистов — 60 м/с. Побеждает мотоциклист, проехавший трассу за минимальное время.

Напишите программу определения оптимальной скорости движения по трассе, если известно, что в начальный момент времени все сигналы светофоров зеленые.

Формат ввода:

Ваша программа должна прочитать из файла MOTO.IN следующую информацию:

- в первой строке — N (число светофоров);
- в следующей строке — $N+1$ чисел (через пробел). Первое — расстояние от точки старта до первого светофора, второе — расстояние между первым и вторым светофорами, ..., последнее число — расстояние от последнего светофора до точки финиша (числа из интервала от 1 до 100000);
- в третьей строке — N чисел: время работы каждого светофора в красном и зеленом режимах (оно одинаково), где время — число из интервала от 1 до 300.

Формат вывода:

Ваша программа должна вывести в файл MOTO.OUT одно число (с точностью до четырех знаков) — искомую скорость или сообщение "NO", если решения нет.

Пример файла входных данных:

```
3
200 100 200 200
2 1 9
```

Пример файла выходных данных:

55.5556

Тур 2

Задача 1. "Правильная дробь"

Правильная дробь вида P/Q (P и Q — натуральные числа и $P < Q$) задана последовательностью не более чем 1000 цифр после запятой в ее десятичной записи. Если десятичное представление дроби является конечным, незначащие правые нули также записываются, и в этом случае будем считать, что период такой дроби состоит из цифры 0. Известно, что точность представления дроби достаточно велика, чтобы период повторился не менее двух раз.

Напишите программу, которая восстановит значения P и Q по десятичной записи дроби. Если в вашем ответе значения P и Q окажутся взаимно простыми, то соответствующий тест оценивается полным баллом, в противном случае — половинным баллом.

Входные данные: текстовый файл FRACTION.IN содержит единственную строку с десятичным представлением дроби (целая часть числа и запятая не записываются).

Выходные данные помещаются в текстовый файл FRACTION.OUT, состоящий из двух строк. Первая строка файла содержит искомое значение P , вторая — значение Q .

Пример входных данных:

333333333333333333

Пример выходных данных:

1

3

**Задача 2. "Фишки"**

Имеются разноцветные фишки K цветов ($K \leq 25$), на каждой из которых записано любое целое неотрицательное число, выбранное из N заданных чисел ($N \leq 101$). Набор из K фишек разных цветов назовем комплектом. Два комплекта считаются одинаковыми, если на любой паре фишек одинакового цвета числа совпадают. В противном случае комплекты считаются разными. Ценой комплекта назовем сумму чисел на фишках, входящих в его состав.

Напишите программу, которая определит число различных комплектов, цена которых равна S ($S \leq 300$).

Исходные данные вводятся из текстового файла FISHC.IN.

Структура файла:

- 1-я строка — число K ;
- 2-я строка — число S ;
- 3-я строка — число N .

Следующие N строк содержат значения чисел, которые можно писать на фишках.

Результат: число, равное количеству всевозможных комплектов, записывается в текстовый файл FISHC.OUT.

Пример:

Исходные данные:

```
6
2
2
0
1
```

Результат: 15, т.к. всего существует 15 возможностей разместить две 1 и четыре 0 на шести разноцветных фишках.

Задача 3. "Преступление и наказание"

Однажды в полночь директора банка Макса Крейзи поднял с постели телефонный звонок. Звонил начальник охраны банка. Он сказал, что произошло крупное ограбление, но, к счастью, охране удалось задержать N подозреваемых. Видимо, грабители каким-то образом избавились от похищенных денег, потому что при обыске ни у кого из подозреваемых не было обнаружено крупной суммы денег. Теперь перед Максом Крейзи стоит задача узнать, кто из подозреваемых — грабитель, а кто — нет? Для этого Макс Крейзи может устраивать подозреваемым очные ставки. В каждой очной ставке участвуют два человека. Первого из них спрашивают, является ли второй грабителем, а второго — является ли грабителем первый. Известно, что если человек не является грабителем, то он всегда правдиво отвечает на задаваемые ему вопросы. Если же человек — грабитель, то он может ответить как правду, так и неправду. Также известно, что количество грабителей меньше $N/2$. Каждый подозреваемый знает, кто является грабителем, а кто — нет. Макс Крейзи необходимо определить список всех грабителей как можно скорее.

Для решения этого задания вам предоставляется библиотека Crime.tpi для работающих на Паскале и Crime.obj для работающих на языке C. Она содержит 3 процедуры:

```
Procedure Init(var N: LongInt);
Procedure Test(i, j: LongInt; var res1, res2: LongInt);
Procedure Result(i: LongInt);
void Init(long &N)
void Test(long i, long j, long &res1, long &res2)
void Result(long i)
```

Процедура **Init** должна вызываться один раз в самом начале программы. Она возвращает значение N — количество подозреваемых. Процедура **Test** устраивает очную ставку i -му и j -му подозреваемым. Она возвращает $res1$ — ответ i -го подозреваемого (0 — грабитель, 1 — не грабитель) и $res2$ — ответ j -го подозреваемого (0 — грабитель, 1 — не грабитель). При вызове процедуры **Test** значение i должно быть не равно значению j . Процедура **Result** должна использоваться для возвращения результатов. Вызов процедуры **Result** означает, что i -й подозреваемый является грабителем. Процедура **Result** должна вызывать-

ся ровно один раз для каждого грабителя. После вызова процедуры **Result** запрещается вызывать процедуры **Test** и **Init**.

Работа с библиотекой

Для корректной работы вашей программы включите в ее текст строку **"uses Crime;"** для программирующих на Паскале или **"#include <crime.h>"** для программирующих на C.

При отладке своей программы вы можете экспериментировать с библиотекой, создавая файл "Input.txt". Первая строка этого файла должна содержать число N . Следующие N строк должны содержать по одному числу каждая: 0 — если очередной подозреваемый является грабителем, 1 — в противном случае. Диалог вашей программы и библиотеки записывается в файл "Output.txt". Результат работы (тест прошел/тест не прошел) записывается в файл "Result.txt".

Пример

Для файла Input.txt:

```
3
1
1
0
```

Если в вашей программе на Паскале есть, например, следующие операторы:

```
Var N, res1, res2: longint;
...
begin
...
    Init(N);
...
    Test(1, 2, res1, res2);
...
    Test(1, 3, res1, res2);
...
    Result(3);
...
end.
```

или если в вашей программе на C есть, например, следующие операторы:

```
main()
{
    long N, res1, res2;
...
    Init(N);
...
    Test(1, 2, res1, res2);
...
    Test(1, 3, res1, res2);
...
    Result(3);
...
}
```

то диалог вашей программы и библиотеки (в файле Output.txt) может иметь вид:

```
Чтение входных данных...
Ok!
Init(3)
Test(1, 2, 1, 1)
Test(1, 3, 0, 1)
Result(3)
```

Ограничения: $1 \leq N \leq 10000$; время на тест — 3 с; гарантируется, что суммарное время работы вызываемых вами процедур библиотеки не превысит 2 с.

Оценка: если ваша программа вызывает **Test** более чем 20000 раз, или совершает какие-либо некорректные действия (например, вызывает процедуру **Init** два раза или вызывает процедуру **Test** после процедуры **Result** и т.д.), или результат ее работы не является правильным, она получает 0 баллов. Иначе, она получает полный балл за тест.

Замечания: ваша программа не должна содержать операторов чтения данных из файлов или записи данных в файлы. Если ваша программа пытается сделать это, она будет снята с тестирования; как только ваша программа совершает некорректное действие, ее работа сразу же будет прервана.

А.КОРБИТ, А.ЩЕРБАКОВ,
г.Минск, БГУИР.

Программирование в среде Visual C++ 6.0

Элементы управления Radio Button и Check Box

Цель этой статьи — изучить принципы работы элементов управления Radio Button и Check Box.

1. Элементы управления Radio Button и Check Box

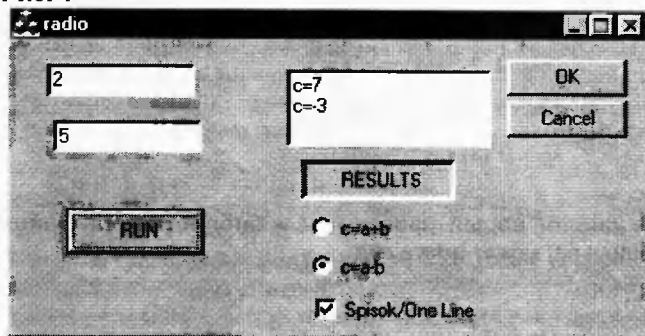
Элемент **Check Box** организует кнопку независимого переключателя, с помощью которой пользователь может указать свое решение типа да/нет. В программе состояние данной кнопки может быть связано со значением булевской переменной, которая проверяется с помощью оператора if.

Элемент **Radio Button** организует группу кнопок — зависимых переключателей. При нажатии одной из кнопок группы все остальные кнопки отключаются. В программе элемент Radio Button связывается с переменной типа int, при этом в программу передается номер включенной кнопки, который анализируется с помощью оператора switch или if. Номер первой кнопки равен 0, второй — 1 и т.д.

2. Пример написания программы

Необходимо написать программу, соответствующую интерфейсу, приведенному на рис.1. В поля ввода должны вводиться два целых числа. “Радиокнопка” (Radio Button) выбирает выполняемое действие — сложение (“c=a+b”) или вычитание (“c=a-b”). “Переключатель” (“Spisok/On Line”), когда он не активен (отсутствует флажок), устанавливает вывод в одну строку при многократном выполнении приложения, при его активизации (установка флажка) результаты выводятся списком.

Рис. 1



Для решения задачи сделайте следующие действия.

1. Создайте при помощи MFCAppWizard(exe) новый проект (предварительно выбрав для него путь и имя — например, L4 или Radio) типа Dialog based.

2. Используя панель Controls, нанесите на Dialog четыре элемента типа Edit Box, три элемента типа Static Text, один элемент типа Button, два элемента типа Radio Button и один элемент типа Check Box. Установите для всех элементов нужные свойства и надписи (для Radio Button с идентификатором IDC_RADIO1 нужно установить свойство Group).

3. При помощи ClassWizard через закладку Member Variables свяжите поле Edit1 с переменной m_a типа int, поле Edit2 — с переменной m_b типа int, поле Edit3 — с переменной m_c типа CString. Для поля Edit3 установите в закладке Styles свойство Multiline.

4. При помощи ClassWizard через закладку Member Variables свяжите идентификатор “переключателя” IDC_CHECK1 с переменной m_s типа BOOL, а “радиокнопки” IDC_RADIO1 — с переменной m_r типа int. Кроме того, в самом начале метода BOOL CL4Dlg::OnInitDialog() нужно установить значение переменной m_r в ноль, т.е. m_r=0;

5. Теперь “оживим” кнопку “RUN”. Для этого необходимо сделать щелчок правой клавишей мыши в любом месте диалоговой панели, и в выпадающем меню выбрать ClassWizard. Затем выбираем закладку Message Maps, увязываем идентификатор IDC_Button1 с событием BN_CLICKED (один щелчок на кнопке) и добавляем обработчик на данное событие — Add Function/OK/Edit Code. Среда добавит в конец файла L4Dlg.cpp:implementation file пустой обработчик:

```
void CL4Dlg::OnButton1()
{
    // TODO: Add your control notification handler code here
}

Теперь вместо фразы
// TODO: ДОБАВЬТЕ ваш код обработчика сообщения здесь
добавляем код решение поставленной задачи:
{
    int r; CString str;

    UpdateData(TRUE);
    if (m_r==0)    r=m_a+m_b;
                  // если выбрана первая радиокнопка — суммирование
    if (m_r==1)    r=m_a-m_b;
                  // если выбрана вторая радиокнопка — вычитание
    if (m_s)
                  // если переключатель активен — вывод списком
    {
        str.Format("c=%d%c",r,13,10);
        m_c=m_c+str;
    }
    else
                  // если переключатель не активен — вывод в одну строку
        m_c.Format("c=%d",r);
    UpdateData(FALSE);
}
```

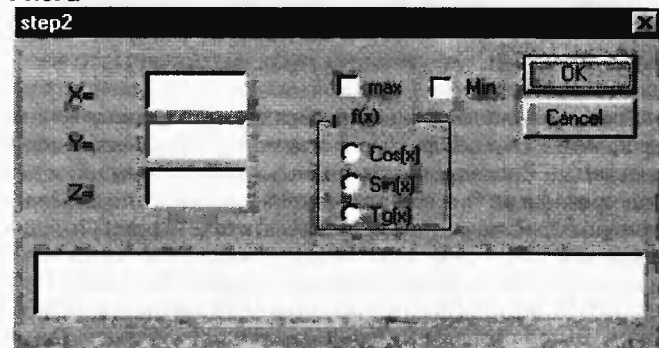
3. Индивидуальные задания

Необходимо написать программу для вычисления составной функции в соответствии с предложенными ниже вариантами. Требуется самостоятельно выбрать необходимое количество исходных данных, для того чтобы в программе выполнялись все возможные ветви алгоритма.

Перед выводом полученного результата программа должна сообщать о ветви, для которой он получен. “Радиокнопками” Cos(x), Sin(x) и Tg(x) должен производиться выбор функции, используемой в качестве f(x). Программа должна через “переключатели” запоминать или не запоминать минимальное и максимальное найденные значения.

Вариант интерфейса программы приведен на рис.2.

Рис. 2



$$\begin{aligned}
 1) \ a &= \begin{cases} (f(x)+y)^2 - \sqrt{f(x)y}, & xy > 0 \\ (f(x)+y)^2 + \sqrt{f(x)y}, & xy < 0 \\ (f(x)+y)^2 + 1, & xy = 0. \end{cases} \\
 2) \ b &= \begin{cases} \ln(f(x)) + (f(x)^2 + y)^3, & x/y > 0 \\ \ln|f(x)/y| + (f(x)+y)^3, & x/y < 0 \\ 0, & x = 0 \\ & y = 0. \end{cases} \\
 3) \ c &= \begin{cases} f(x)^2 + y^2 + \sin(y), & x - y = 0 \\ (f(x) - y)^2 + \cos(y), & x - y > 0 \\ (y - f(x))^2 + \operatorname{tg}(y), & x - y < 0. \end{cases} \\
 4) \ d &= \begin{cases} (f(x) - y)^3 + \operatorname{arctg}(f(x)), & x > y \\ (y - f(x))^3 + \operatorname{arctg}(f(x)), & y > x \\ (y + f(x))^3 + 0.5, & y = x. \end{cases} \\
 5) \ e &= \begin{cases} i\sqrt{f(x)}, & i - \text{нечетное}, x > 0 \\ i / 2\sqrt{f(x)}, & i - \text{четное}, x < 0 \\ \sqrt{|f(x)|}, & \text{иначе.} \end{cases} \\
 6) \ g &= \begin{cases} e^{f(x)-|b|}, & 0.5 < xb < 10 \\ \sqrt{|f(x)+b|}, & 0.1 < xb < 0.5 \\ 2f(x)^2, & \text{иначе.} \end{cases} \\
 7) \ s &= \begin{cases} e^{f(x)}, & 1 < xb < 10 \\ \sqrt{|f(x)+4 \cdot b|}, & 12 < xb < 40 \\ bf(x)^2, & \text{иначе.} \end{cases} \\
 8) \ j &= \begin{cases} \sin(5f(x) + 3m|f(x)|), & -1 < m < x \\ \cos(3f(x) + 5m|f(x)|), & x < m \\ (f(x) + m)^2, & x = m \end{cases} \\
 9) \ l &= \begin{cases} 2f(x)^3 + 3p^2, & x > |p| \\ |f(x) - p|, & 3 < x < |p| \\ (f(x) - p)^2, & x = |p|. \end{cases} \\
 10) \ k &= \begin{cases} \ln(|f(x)| + |q|), & |xq| > 10 \\ e^{f(x)+q}, & |xq| < 10 \\ (f(x) + q), & |xq| = 10. \end{cases} \\
 11) \ m &= \frac{\max(f(x), y, z)}{\min(f(x), y)} + 5. \\
 12) \ n &= \frac{\min(f(x) + y, y - z)}{\max(f(x), y)}. \\
 13) \ p &= \frac{|\min(f(x), y) - \max(y, z)|}{2}. \\
 14) \ q &= \frac{\max(f(x) + y + z, xyz)}{\min(f(x) + y + z, xyz)}. \\
 15) \ r &= \max(\min(f(x), y)z).
 \end{aligned}$$

Литература

1. С.Холзнер. Visual C++ 6: учебный курс. — СПб.: Питер, 1999. — 576 с.
2. Н.Ю.Секунов. Самоучитель Visual C++ 6 (+ дискета). — СПб.: BHV-Санкт-Петербург, 1999. — 960 с.
3. К.Грегори. Использование Visual C++ 6. Специальное издание. — СПб.: Вильямс, 1999. — 864 с.
4. Ю.В.Тихомиров. Visual C++ 6 (+ дискета). — СПб.: BHV-Санкт-Петербург, 1999. — 496 с.
5. Майкл Дж. Янг. Visual C++ 6. Полное руководство (+ CD-ROM). — BHV-Киев, 2000. — 1056 с.
6. С.Гилберт, Б.Маккарти. Самоучитель Visual C++ 6 (+ CD-ROM). — М.: Диалог-МИФИ, 2001. — 416 с.
7. К.Паппас, У.Мюррей. Visual C++ 6 для пользователя. — BHV-Киев, 2000. — 336 с.
8. К.Паппас, У.Мюррей. Visual C++ 6: руководство разработчика. — BHV-Киев, 2000. — 624 с.
9. И.Баженова. Visual C++ 6. 0. VISUAL STUDIO 98: Уроки программирования. — М.: Диалог-МИФИ, 2001. — 416 с.
10. Лэйси Дж. Visual C++ 6 Distributed. Сертификационный экзамен — экстерном. — СПб.: Питер, 2001. — 624 с.
11. А.Черносвитов. Visual C++ 7: учебный курс (с дискетой). — СПб.: Питер, 2001. — 528 с.
12. А.Корбит, А.Щербаков. Программирование в среде Visual C++ 6.0. Диалоговые окна и простейшие элементы управления. — Радиомир. Ваш компьютер, 2003, №4, С.23.

Решение задач на графах построением минимального остовного дерева

М.ДОЛИНСКИЙ,
г.Гомель.

(Продолжение. Начало в №5/2003)

3. Решение задачи "Веревочный телеграф" методом Крускала

Для решения приведенных далее задач "Secret Milk Pipes" (USACO Open 2002) и "Метро" (Белорусская республиканская олимпиада 2002 г.) удобнее применять метод Крускала. Поскольку его придется модифицировать, то вначале рассмотрим его применение в "чистом виде" для решения задачи "Веревочный телеграф".

В тело главной программы необходимо внести существенное изменение:

```

begin
  InputData;
  QuickSortR(1, N*(N-1) div 2);
  MinTree_Kruskal;
  OutputData;
end.

```

Алгоритм Крускала работает на множестве упорядоченных по возрастанию весов ребер $R[k]$. Поэтому перед началом его работы необходимо произвести переупорядочивание ребер по возрастанию весов. Эту работу и выполняет рекурсивная процедура QuickSortR, которой при начальном вызове передаются номера начального (1) и конечного ($N*(N-1) \div 2$) ребер.

В данном случае рекомендуется применять алгоритм быстрой сортировки, который ввиду своей важности будет рассматриваться отдельно.

Ввод данных тоже приходится немного изменить:

```

procedure InputData;
begin

```

```

  assign(input, 'input.txt'); reset(input);
  readln(N);
  for i:=1 to N do readln(x[i], y[i]);
  close(input);
  k:=0;
  for i:=1 to N-1 do
    for j:=i+1 to N do
      begin
        inc(k);
        R[k]:=sqrt(sqr(x[i]-x[j])+sqr(y[i]-y[j]));
        B[k]:=i; E[k]:=j;
      end;
  end;

```

Здесь граф представлен как набор из $N*(N-1)/2$ ребер, где:

- $R[k]$ — вес ребра (длина веревки) с номером k ;
- $B[k]$ — начальная вершина ребра;
- $E[k]$ — конечная вершина ребра.

Ответ будет накапливаться в переменной Answer непосредственно в процессе добавления ребер в минимальное остовное дерево, поэтому процедура вывода результата предельно упростится:

```

procedure OutputData;
begin
  assign(output, 'output.txt'); rewrite(output);
  writeln(Answer:0:2);
  close(output);
end;

```

Процедура же построения минимального остовного дерева методом Крускала выглядит следующим образом:

```

procedure MinTree_Kruskal;

```



```

begin
  for i:=1 to N do Makeset(i);
  Answer:=0;
  for i:=1 to k do
    if Findset(B[i]) <> Findset(E[i])
    then
      begin
        Answer:=Answer+R[i];
        Union(B[i],E[i]);
      end;
  end;
end;

```

Алгоритм Крускала работает следующим образом:

```

for i:=1 to N do Makeset(i);

```

Вначале строятся N (по числу вершин) непересекающихся множеств, каждое из которых хранит номера вершин, в него входящих (вначале каждое такое множество хранит только одну вершину — с номером вершины, соответствующим номеру множества).

Далее осуществляется цикл по всем ребрам $(B[i], E[i])$ с весом $R[i]$ (в порядке возрастания весов $R[i]$). Если ребро $(B[i], E[i])$ соединяет два разных множества:

```

Findset(B[i]) <> Findset(E[i]),

```

то это ребро включается в минимальный остовный граф, его вес суммируется с ответом:

```

Answer:=Answer+R[i];

```

а сами множества объединяются (как связывающиеся данным ребром):

```

Union(B[i],E[i]);

```

Рассмотрим более подробно процедуры, которые реализуют работу с множеством непересекающихся подмножеств вершин в процессе построения минимального остовного дерева.

Все они работают с двумя массивами — $Pred[1..N]$ и $Rank[1..N]$. Массив $Pred[i]$ содержит номер вершины-предка. Массив $Rank[i]$ содержит ранг вершины. Понятие ранг вводится для ускорения обработки.

Как уже говорилось, процедура $MakeSet$ осуществляет начальное построение системы независимых множеств вершин:

```

procedure MakeSet(x:longint);
begin
  Pred[x]:=x; {вначале вершина ссылается сама на себя}
  rank[x]:=0; {ее ранг равен 0}
end;

```

Рекурсивная Функция $Findset(x)$ отвечает на вопрос: "Какому множеству из построенной системы подмножеств принадлежит вершина x?" Каждое из подмножеств фактически представляет собой поддерево, и функция $FindSet$ возвращает номер вершины, которая является корнем этого поддерева:

```

function FindSet(x:longint):longint;
begin
  if x <> pred[x]
  then Pred[x]:=FindSet(pred[x]);
  FindSet:=Pred[x];
end;

```

Процедура $Union(x,y)$ вызывается в том случае, если вершины x и y лежат в разных поддеревьях, и служит для объединения этих поддеревьев. При этом она находит корни соответствующих поддеревьев, два раза вызывая рекурсивную функцию $FindSet$ ($Findset(x)$ и $Findset(y)$), а затем вызывает процедуру $Link$ для объединения деревьев с заданными корнями:

```

Procedure Union(x,y:longint);
begin
  Link (FindSet(x),FindSet(y));
end;

```

И наконец, процедура $Link(x,y)$ осуществляет объединение деревьев с корнями x и y и перестроение рангов вершин, входящих в эти деревья:

```

Procedure Link(x,y:longint);
begin
  if rank[x]>rank[y]
  then pred[y]:=x
  else
    begin
      pred[x]:=y;

```

```

    if rank[x]=rank[y]
    then inc(rank[y]);
  end;
end;

```

Напомним, что ранги вершин обеспечивают сокращение количества операций (и соответственно времени), затрачиваемых на обработку системы непересекающихся поддеревьев.

Полное решение задачи "Веревочный телеграф" методом Крускала имеет вид:

```

program gg99dlt2;

```

```

Const

```

```

  MaxN = 100;

```

```

var

```

```

  N,i,j      : longint;

```

```

  X,Y        : array [1..MaxN] of longint;

```

```

  Pred,Rank  : array [1..MaxN] of longint;

```

```

  k,t,m,p,rr : longint;

```

```

  R          : array [0..4950] of single;

```

```

  B,E        : array [1..4950] of 0..MaxN;

```

```

  Answer,Min  : single;

```

```

procedure InputData;

```

```

begin

```

```

  assign (input,'input.txt'); reset(input);

```

```

  readln(N);

```

```

  for i:=1 to N do readln(x[i],y[i]);

```

```

  close(input);

```

```

  k:=0;

```

```

  for i:=1 to N-1 do

```

```

    for j:=i+1 to N do

```

```

      begin

```

```

        inc(k);

```

```

        R[k]:=sqrt(sqr(x[i]-x[j])+sqr(y[i]-y[j]));

```

```

        B[k]:=i; E[k]:=j;

```

```

      end;

```

```

  end;

```

```

procedure OutputData;

```

```

begin

```

```

  assign (output,'output.txt'); rewrite(output);

```

```

  writeln(Answer:0:2);

```

```

  close(output);

```

```

end;

```

```

procedure MakeSet(x:longint);

```

```

begin

```

```

  Pred[x]:=x;

```

```

  rank[x]:=0;

```

```

end;

```

```

function FindSet(x:longint):longint;

```

```

begin

```

```

  if x <> pred[x]

```

```

  then Pred[x]:=FindSet(pred[x]);

```

```

  FindSet:=Pred[x];

```

```

end;

```

```

Procedure Link(x,y:longint);

```

```

begin

```

```

  if rank[x]>rank[y]

```

```

  then pred[y]:=x

```

```

  else

```

```

    begin

```

```

      pred[x]:=y;

```

```

      if rank[x]=rank[y]

```

```

      then inc(rank[y]);

```

```

    end;

```

```

  end;

```

```

Procedure Union(x,y:longint);

```

```

begin

```

```

  Link (FindSet(x),FindSet(y));

```

```

end;

```

```

procedure MinTree_Kruskal;

```

```

begin

```

```

  for i:=1 to N do Makeset(i);

```

```

  Answer:=0;

```

```

  for i:=1 to k do

```

```

    if Findset(B[i]) <> Findset(E[i])

```

```

    then

```



```
begin
  Answer:=Answer+R[i];
  Union(B[i],E[i]);
end;
end;
```

```
function Partition(p,rr:longint):longint;
var
  i,j,z : longint;
  x,t : single;
begin
  x := R[p];
  i:=p-1;
  j:=rr+1;
  while i<j do
    begin
      repeat j:=j-1 until R[j]<=x;
      repeat i:=i+1 until R[i]>=x;
      if i<j
      then
        begin
          t:=R[i]; R[i]:=R[j]; R[j]:=t;
          z:=B[i]; B[i]:=B[j]; B[j]:=z;
          z:=E[i]; E[i]:=E[j]; E[j]:=z;
        end;
      end;
    end;
  Partition:=j;
end;
```

```
procedure QuicksortR(p,rr:longint);
var
  q : longint;
begin
  if p<rr
  then
    begin
      q:= Partition(p,rr);
      QuicksortR(p,q);
      QuicksortR(q+1,rr);
    end;
end;
```

```
begin
  InputData;
  QuicksortR(1,N*(N-1) div 2);
  MinTree_Kruskal;
  OutputData;
end.
```

4. Быстрая сортировка

Быстрая сортировка участка массива R от p до rr происходит следующим образом.

Элементы массива R переставляются так, чтобы любой из элементов R[p], ..., R[q] был не больше любого из элементов R[q+1], ..., R[rr]. Эта операция называется разделением (Partition). Здесь q — некоторое число в интервале p≤q<rr. Процедура сортировки рекурсивно вызывается для массивов R[p..q] и R[q+1..rr]:

```
procedure QuicksortR(p,rr:longint);
var
  q : longint;
begin
  if p<rr
  then
    begin
      q:= Partition(p,rr);
      QuicksortR(p,q);
      QuicksortR(q+1,rr);
    end;
end;
```

Теперь подробнее рассмотрим реализацию функции Partition(p,rr):

```
function Partition(p,rr:longint):longint;
var
  i,j,z : longint;
  x,t : single;
begin
  x := R[p];
  i:=p-1;
  j:=rr+1;
  while i<j do
    begin
```

```
      repeat j:=j-1 until R[j]<=x;
      repeat i:=i+1 until R[i]>=x;
      if i<j
      then
        begin
          t:=R[i]; R[i]:=R[j]; R[j]:=t;
          z:=B[i]; B[i]:=B[j]; B[j]:=z;
          z:=E[i]; E[i]:=E[j]; E[j]:=z;
        end;
      end;
```

```
    end;
  Partition:=j;
end;
```

В качестве "граничного элемента" принимается X=R[p]. Далее заводим два указателя:

- i — при просмотре элементов с начала сортируемого участка;

- j — при просмотре элементов с конца сортируемого участка. И, пока i<j, выполняем следующее:

- находим R[j] — первый с конца элемент меньше X;

- находим R[i] — первый с начала элемент больше X.

Если i<j, то меняем местами элементы R[i] и R[j]:

```
t:=R[i]; R[i]:=R[j]; R[j]:=t;
```

Поскольку в данной задаче нужно синхронно с перемещением весов ребер R[k] перемещать также вершины их начала и конца, выполняются также соответствующие операции:

```
z:=B[i]; B[i]:=B[j]; B[j]:=z;
z:=E[i]; E[i]:=E[j]; E[j]:=z;
```

По завершении цикла "пока i<j" мы и получаем границу "разделения":

```
Partition:=j;
```

Понятно, что вызывать алгоритм быстрой сортировки нужно передавая ему начальные номера начального и конечного элементов массива, например, для задачи "Веревочный телеграф":

```
QuicksortR(1,N*(N-1) div 2);.
```

5. Задача "Secret Milk Pipes"

(USA Computing Olympiad (USACO) Open 2002, Green Division)

Фермер Джон (ФД) хочет с как можно меньшими затратами создать свою систему распределения воды, но он не хочет, чтобы его конкурент — фермер Snidely — мог предсказать маршруты, которые он выберет. ФД знает, что подобная задача обычно требует самого дешевого способа соединять трубы, поэтому он решил использовать другой дешевый способ.

Дан список всех двунаправленных труб, которые могут соединять множество из W ($3 \leq W \leq 2000$) станций с водой (каждая из которых может быть встроена в колодец). Ваша задача — найти второй из самых дешевых способов соединить водные станции, используя не более чем P ($P \leq 20000$) труб с заданной стоимостью каждой трубы. Не должно быть трубы, соединяющей станцию саму с собой. Не должно быть двух труб, соединяющих дважды одну и ту же пару станций.

Гарантируется, что есть только один самый дешевый способ распределить воду, и что всего существует как минимум два способа. Все стоимости — положительные числа, помещающиеся в 16-битное целое. Водная станция идентифицируется своим номером — целым числом в диапазоне 1...W.

PROBLEM NAME: secret

Формат ввода:

Строка 1 — два разделенных пробелом целых числа W и P.

Строки 2..P+1 — каждая строка описывает одну трубу и содержит три числа, разделенные пробелом: номера станции начала и конца трубы и стоимость этой трубы.

Пример ввода (файл secret.in):

```
5 7
1 2 3
2 3 4
1 4 7
2 4 11
2 5 9
5 4 5
3 5 8
```

**Формат вывода:**

Одна строка, содержащая целое число — вторая минимальная стоимость конструирования системы распределения воды.

Пример вывода (файл secret.out):

20

В данной задаче мы можем рассматривать водные станции как вершины; трубы, соединяющие водные станции — как ребра, а стоимость труб — как веса ребер.

Тогда по условию задачи от нас требуется построить второе по весу из остовных деревьев.

Прежде всего обеспечим ввод исходных данных:

```
procedure InputData;
begin
  assign (input, 'secret.in'); reset(input);
  readln(N, P);
  for i:=1 to P do readln(B[i], E[i], R[i]);
  close(input);
end;
```

Теперь решение задачи можно обеспечить следующим образом:

1. Найти минимальное остовное дерево.

2. В цикле "удалять" по очереди одно из найденных ребер минимального остовного дерева и достраивать оставшийся нецельным остов до минимального остовного дерева. Попутно вычислять стоимости полученных деревьев и запоминать меньшую из них.

Для нахождения минимального остовного дерева используем алгоритм Крускала:

```
procedure MinTree_Kruskal_1;
begin
  Answer:=0; j:=0;
  for i:=1 to N do Makeset(i);
  for i:=1 to P do
    if (Findset(B[i]) <> Findset(E[i]))
    then
      begin
        inc(j); Key[j]:=i;
        Union(B[i], E[i]);
      end;
  end;
```

end;

Легко заметить, что в процедуру поиска минимального остовного дерева вместо строки подсуммирования ответа

```
Answer:=Answer+R[i];
```

введена строка

```
inc(j); Key[j]:=i;
```

которая сохраняет в массиве Key номера всех ребер, включенных в минимальное остовное дерево.

Тогда тело главной программы может выглядеть следующим образом:

```
begin
  InputData;
  QuickSortR(1, P);
  MinTree_Kruskal_1;
  for i:=1 to P do NotMasked[i]:=true;
  CurAnswer:=maxint;
  for v:=1 to N-1 do
    begin
      NotMasked[key[v]]:=false;
      MinTree_Kruskal_2;
      if Answer < CurAnswer then CurAnswer:=Answer;
      NotMasked[key[v]]:=true;
    end;
  assign (output, 'secret.out'); rewrite(output);
  writeln(CurAnswer);
  close(output);
end.
```

Здесь первые три строки:

```
InputData;
QuickSortR(1, P);
MinTree_Kruskal_1;
```

обеспечивают построение минимального остовного дерева, где Key[i] (для i от 1 до N-1) — номера ребер остовного дерева.

Затем отмечаем все ребра как свободные:

```
for i:=1 to P do NotMasked[i]:=true;
```

и инициализируем поиск минимального ответа:

```
CurAnswer:=maxint;
```

Далее в цикле по всем ребрам, включенным в минимальное остовное дерево:

```
for v:=1 to N-1 do ,
```

помечаем текущее ребро минимального остовного дерева как занятое:

```
NotMasked[key[v]]:=false;
```

Вызываем алгоритм построения минимального остовного дерева, который учитывает, можно ли использовать при построении текущего дерева данное ребро:

```
MinTree_Kruskal_2;
```

Среди получаемых ответов Answer ищется минимальный:

```
if Answer < CurAnswer then CurAnswer:=Answer;
```

В конце цикла возвращаем "исключенное" ребро в список:

```
NotMasked[key[v]]:=true;
```

Модифицированный алгоритм Крускала для поиска минимального остова на подмножестве ребер выглядит следующим образом:

```
procedure MinTree_Kruskal_2;
begin
  Answer:=0; j:=0;
  for i:=1 to N do Makeset(i);
  for i:=1 to P do
    if (Findset(B[i]) <> Findset(E[i])) and
    (NotMasked[i])
    then
      begin
        Answer:=Answer+R[i];
        Union(B[i], E[i]);
      end;
  end;
```

end;

Как легко заметить, отличие заключается в операторе сравнения:

```
if (Findset(B[i]) <> Findset(E[i])) and
(NotMasked[i])
```

т.е. если вершины находятся в разных поддеревьях, и текущее ребро не маскировано.

Однако такое решение на тестах максимальной размерности (2000 вершин, 20000 ребер) не проходит по времени исполнения.

Задумавшись, а можно ли сократить объем работы, выполняемой в процедурах MinTree_Kruskal_1 и MinTree_Kruskal_2?

Да, можно — из следующих соображений.

1) В обеих процедурах мы осуществляем перебор по всем ребрам:

```
for i:=1 to P do
```

в то время как достаточно вести перебор до нахождения N-1-го ребра минимального остовного дерева:

```
while (J < N-1) do,
```

где J — номер последнего ребра, найденного для минимального остовного дерева.

2) В процедуре MinTree_Kruskal_2 можно сделать несколько дополнительных оптимизаций: а) можно начинать с остовного дерева из N-2 ребер. Для этого достаточно без перебора в главной программе включить в минимальное остовное дерево N-2 найденных в процедуре MinTree_Kruskal_1 ребер (по очереди исключая из рассмотрения одно из найденных ребер);

б) можно не вызывать процедуру Union;

в) просмотр ребер из множества заданных можно начинать с ребра, следующего за исключенным из просмотра ребром минимального остовного дерева;

г) оператор if можно разбить на два оператора, чтобы для помеченного ребра не вызывалась рекурсивная процедура Findset.

Полный текст программы, решающей задачу "Secret Milk Pipes", имеет вид:

```
{ $E+, N+, R+, O+ }
```



```
program us02ope2;
Const
  MaxN = 2000 ;
  MaxP = 20000 ;
  Free =0;
  Used =1;
var
  N          : 1..MaxN;
  P          : 1..MaxP;
  i,j        : integer;

  Pred,Rank  : array [1..MaxN] of integer;
  B, E       : array [1..MaxP] of integer;
  NotMasked  : array [1..MaxP] of boolean;
  Key        : array [1..MaxN-1] of integer;

  Answer,MinAnswer,CurAnswer : longint;
  u,v        : integer;

  R : array[1..MaxP] of integer;

procedure InputData;
begin
  assign (input,'secret.in'); reset(input);
  readln(N,P);
  for i:=1 to P do readln(B[i],E[i],R[i]);
  close(input);
end;

function Partition(p,rr:longint):longint;
var
  i,j,z : longint;
  x,y   : integer;
begin
  x := R[p];
  i:=p-1;
  j:=rr+1;
  while i<j do
    begin
      repeat j:=j-1 until R[j]<=x;
      repeat i:=i+1 until R[i]>=x;
      if i<j
      then
        begin
          y:=R[i]; R[i]:=R[j]; R[j]:=y;
          z:=B[i]; B[i]:=B[j]; B[j]:=z;
          z:=E[i]; E[i]:=E[j]; E[j]:=z;
        end;
      end;
      Partition:=j;
    end;
end;

procedure QuicksortR(p,rr:longint);
var
  q : longint;
begin
  if p<rr
  then
    begin
      q:= Partition(p,rr);
      QuicksortR(p,q);
      QuicksortR(q+1,rr);
    end;
end;

procedure MakeSet(x:longint);
begin
  Pred[x]:=x;
  rank[x]:=0;
end;

function FindSet(x:longint):longint;
begin
  if x<>pred[x]
  then Pred[x]:=FindSet(pred[x]);
  FindSet:=Pred[x];
end;

Procedure Link(x,y:longint);
```

```
begin
  if rank[x]>rank[y]
  then pred[y]:=x
  else
    begin
      pred[x]:=y;
      if rank[x]=rank[y]
      then inc(rank[y]);
    end;
end;

Procedure Union(x,y:longint);
begin
  Link (FindSet(x),FindSet(y));
end;

procedure MinTree_Kruskal_1;
begin
  Answer:=0; j:=0;
  for i:=1 to N do Makeset(i);
  i:=1;
  While J<N-1 do
    begin
      if (FindSet(B[i])<>FindSet(E[i]))
      then
        begin
          Answer:=Answer+R[i];
          inc(j); Key[j]:=i;
          Union(B[i],E[i]);
        end;
      inc(i);
    end;
end;

procedure MinTree_Kruskal_2;
var k : longint;
  FindSetConst:array [1..MaxN] of integer;
begin
  While (J<N-1) do
    begin
      if (NotMasked[i])
      then
        if (FindSet(B[i])<>FindSet(E[i]))
        then
          begin
            Answer:=Answer+R[i];
            inc(j);
          end;
        inc(i)
      end;
    end;
end;

begin
  InputData;
  QuicksortR(1,P);
  MinTree_Kruskal_1;
  for i:=1 to P do NotMasked[i]:=true;
  CurAnswer:=maxlongint; MinAnswer:=Answer;
  for v:=1 to N-1 do
    begin
      NotMasked[key[v]]:=false; Answer:=MinAnswer-
      R[key[v]];
      for i:=1 to N do Makeset(i);
      for u:=1 to N-1 do
        if NotMasked[key[u]]
        then
          begin
            Union(B[key[u]],E[key[u]]);
            NotMasked[key[u]]:=false;
          end;
        i:=key[v]+1; j:=N-2;
        MinTree_Kruskal_2;
        if Answer<CurAnswer then CurAnswer:=Answer;
        for u:=1 to N-1 do NotMasked[key[u]]:=true;
      end;
      assign (output,'secret.out'); rewrite(output);
      writeln(CurAnswer);
      close(output);
    end;
end.
```

(Продолжение следует)



Serrgio /HI-TECH,
E-mail: serrgio@gorki.unibel.by
http://wasm.ru/

НИЗКОУРОВНЕВОЕ ПРОГРАММИРОВАНИЕ

(Продолжение. Начало в №№9-12/2001, №№1-10,
12/2002, №№1-3,5/2003)

Подсистема Windows GDI

#1. Если помните, под DOS и в ранних версиях Windows у программиста была замечательная возможность записывать изображения прямо в видеопамять. Под win32 не стоит даже замахиваться на такое (это могут делать только драйвера; и это правильно, иначе невозможно было бы организовать правильную совместную работу множества программ). За отображение графики в Windows отвечает подсистема **GDI** (Graphics Device Interface), которая обеспечивает *аппаратно-независимую* графику за счет использования драйверов (правда, есть еще DirectX, который специально разработан для того, чтобы сделать железо "ближе" к программам, но это уже совсем другая история). То есть прикладные программы независимы от конкретных характеристик и механизмов работы "устройств отображения информации" (мониторов, принтеров, плоттеров). Они работают абстрагированно, в *аппаратно-независимых* образах, на уровне *отрезков, прямоугольников, полигонов, сплайнов* и тому подобных объектов. А GDI, в свою очередь, передает информацию *драйверу устройства*, который работает с соответствующим железом уже на низком уровне (или переводит эту информацию в другую систему команд, например, принтера).

Следует также понимать, что подсистема GDI ориентирована на статическое отображение информации. И, несмотря на то что определенная поддержка анимации все-таки имеется (например, перемещение графических объектов по экрану), она является ограниченной — нет трехмерной системы координат, возможности вращения объектов и прочих графических "наворотов". Все это — следствие того, что из соображений производительности, для представления координат объектов используются исключительно целые числа.

Таким образом, если вы хотите программировать 3D-графику, то нужно "направить лыжи" совершенно в другом направлении. С использованием описанной в данной статье техники вы сможете написать, разве что, очередной клон Тетриса — анимация более высокого уровня с использованием GDI невозможна.

#2. Начнем традиционно — с исходника. Для экономии места я приведу только *процедуру окна*. Остальное все равно является частью шаблона, который практически без изменений повторяется в любой оконной программе.

... ..

.data?

```
cxClient dd ?
cyClient dd ?
```

... ..

```
;-----
; процедура окна
;-----
```

```
WndProc proc hWnd :HWND, uMsg :UINT, wParam :WPARAM,
lParam :LPARAM
```

```
local hDC :HDC
```

```
local ps :PAINTSTRUCT
local x1 :dword, y1 :dword, x2 :dword, y2 :dword
```

```
mov eax, [uMsg]
```

```
.if eax==WM_PAINT
```

```
invoke BeginPaint, hWnd, addr ps
mov [hDC], eax
```

```
;----- прямоугольник
```

```
mov eax, [cxClient]
lea ecx, [eax*8]
sub ecx, eax
shr eax, 3 ; = cxClient/8
shr ecx, 3 ; = 7*cxClient/8
mov [x1], eax
mov [x2], ecx
```

```
mov eax, [cyClient]
lea edx, [eax*8]
sub edx, eax
shr eax, 3 ; = cyClient/8
shr edx, 3 ; = 7*cyClient/8
mov [y1], eax
mov [y2], edx
```

```
invoke Rectangle, hDC, x1, eax, ecx, edx
```

```
;----- две диагональные линии
```

```
invoke MoveToEx, hDC, 0, 0, NULL
invoke LineTo, hDC, cxClient, cyClient
```

```
invoke MoveToEx, hDC, 0, cyClient, NULL
invoke LineTo, hDC, cxClient, 0
```

```
;----- эллипс
```

```
invoke Ellipse, hDC, x1, y1, x2, y2
```

```
;----- прямоугольник со скругленными углами
```

```
mov ecx, [cxClient]
lea eax, [2*ecx+ecx]
shr ecx, 2 ; = cxClient/4
shr eax, 2 ; = 3*cxClient/4
mov [x2], eax
```

```
mov eax, [cyClient]
lea edx, [2*eax+eax]
shr eax, 2 ; = cyClient/4
shr edx, 2 ; = 3*cyClient/4
```

```
invoke RoundRect, hDC, ecx, eax, x2, edx, ecx, eax
```

```
;----- конец рисования
```

```
invoke EndPaint, hWnd, addr ps
```

```
.elseif eax==WM_SIZE
```

```
;----- координаты клиентской области
```

```
movzx eax, word ptr lParam[0] ; low word
mov [cxClient], eax
movzx eax, word ptr lParam[2] ; hi word
mov [cyClient], eax
```

```
.elseif eax==WM_DESTROY
```

```
invoke PostQuitMessage, 0
```

```
.else
```

```
invoke DefWindowProc, hWnd, eax, wParam, lParam
ret
```

```
.endif
xor eax, eax
ret
```

```
WndProc endp
```



Пожалуй, стоит оговориться, что эта программа является ассемблерным вариантом сишного "сырца" под названием "LINEDEMO" из книги Ч.Петзольда "Программирование под Windows 95". Оба варианта доступны для скачивания по адресу <http://radio-mir.com/>

#3. Первое, что нам нужно сделать — получить хэндл **контекста устройства** (device context), через который выполняется доступ к устройству. Именно этот хэндл является обязательным параметром для всех функций GDI. Существует несколько способов его получения, в зависимости от того, для каких целей мы собираемся его использовать.

1. В обработчике сообщения **WM_PAINT** чаще всего используется функция **BeginPaint**. Ее параметры — хэндл окна и указатель на структуру типа **PAINTSTRUCT**, которая в результате выполнения этой функции заполняется полезной информацией. Например, ее элемент **rcPaint** — это структура **RECT**, содержащая координаты **недействительного** региона окна (т.е. региона, требующего перерисовки). Когда необходимость в контексте устройства отпадает, его необходимо освободить при помощи функции **EndPaint**.

2. В обработчиках, отличных от **WM_PAINT**, т.е. предназначенных не для рисования, а, например, для получения какой-либо информации об устройстве, с той же целью принято использовать функцию **GetDC**. Полученный таким образом контекст устройства относится ко всей клиентской области окна, и не имеет ни малейшего понятия о действительности/недействительности регионов. Попробуйте использовать его для рисования, и вы поймете, почему это не стоит делать :). Этот контекст устройства освобождается при помощи функции **ReleaseDC**.

3. Хэндл, полученный при помощи функции **GetWindowDC**, относится уже ко всему окну. Т.е. в дополнение к клиентской области он включает в себя заголовок окна, меню, полосы прокрутки и даже рамку окна. При желании можно рисовать и на этих элементах :). Сообщение, которое генерируется системой для перерисовки неклиентской части окна, называется **WM_NCPAINT** (nonclient paint). Контекст устройства освобождается функцией **ReleaseDC**.

4. И наконец, при помощи функции **CreateDC** мы можем получить хэндл, описывающий "устройство целиком". Например, хэндл, полученный вызовом **invoke CreateDC, addr Driver, 0, 0, 0**, при условии, что **Driver** db "DISPLAY",0, позволит рисовать на **всей** видимой части дисплея. Контекст устройства освобождается функцией **DeleteDC**.

Как нетрудно заметить, в нашем примере использована пара **BeginPaint / EndPaint**, т.е. мы получаем хэндл контекста устройства для рисования в пределах клиентской области.

#4. Теперь посмотрим на глобальные переменные **cxClient** и **cyClient**. В этих переменных содержатся ширина и высота клиентской области. При поступлении сообщения **WM_SIZE** (смотрим на соответствующий обработчик) младшее слово переменной **lParam** (она декларирована вместе с процедурой окна) содержит ширину, а старшее — высоту клиентской области. При помощи пары нехитрых операций

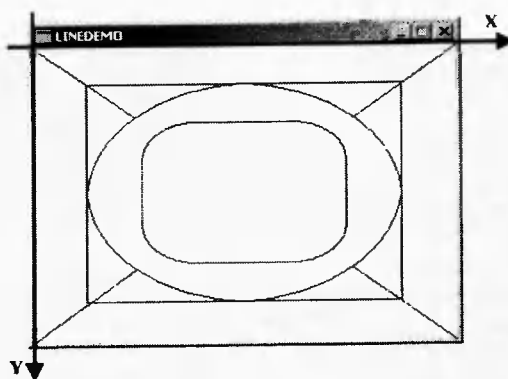
```
movzx eax, word ptr lParam[0] ; low word
mov [cxClient], eax
movzx eax, word ptr lParam[2] ; hi word
mov [cyClient], eax
```

мы извлекаем из **lParam** старшее и младшее слова и сохраняем их в глобальных переменных **cxClient** и **cyClient**. Подобная информация нам понадобится при рисовании для масштабирования отрисованных элементов относительно размеров окна.

Что ж, теперь самое время заняться рисованием. Скриншот того, что мы нарисуем, выглядит так, как показано на **рис.1**.

Я добавил к нему две стрелки, чтобы отметить систему координат. Windows поддерживает несколько режимов отображения (впоследствии мы еще рассмотрим их), однако именно режим "координаты сверху вниз" является режимом по умолчанию.

Рис. 1



#5. Для большинства задач наиболее простым графическим примитивом является линия (хотя с точки зрения геометрии правильнее будет назвать этот примитив отрезком). Рисуются она с помощью функции **LineTo**:

```
BOOL LineTo(
    HDC hdc, // device context handle
    int nXEnd, // x-coordinate of ending point
    int nYEnd // y-coordinate of ending point
);
```

Как нетрудно заметить, эта функция получает отнюдь не все координаты отрезка. В качестве параметров ей передаются только координаты конечной точки. А где, спрашивается, исходная?

Координаты исходной точки определяются при помощи функции **MoveToEx**:

```
BOOL MoveToEx(
    HDC hdc, // handle to device context
    int X, // x-coordinate of new current position
    int Y, // y-coordinate of new current position
    LPPOINT lpPoint // old current position
);
```

Т.е. трижды вызвав **LineTo** с разными параметрами, мы начертим три отрезка из одной и той же исходной точки. Этот странный механизм называется "пером", и придуман он, наверное, для того, чтобы оптимизировать рисование детьми солнышек на уроках информатики :).

В нашем примере мы чертим две линии. Сначала перемещаем перо в начало координат и чертим линию до правого нижнего угла клиентской области. А затем перемещаем перо в правый верхний угол и чертим линию до нижнего левого. Получаются две диагональные линии. Все это осуществляет следующий код:

```
invoke MoveToEx, hdc, 0, 0, NULL
invoke LineTo, hdc, cxClient, cyClient

invoke MoveToEx, hdc, 0, cyClient, NULL
invoke LineTo, hdc, cxClient, 0
```

Последний параметр **NULL** в **MoveToEx** означает, что мы не нуждаемся в информации о предыдущем состоянии. По умолчанию перо установлено в начало координат (впрочем, это можно проверить при помощи функции **GetCurrentPositionEx**). Точка, определенная как конец отрезка, в отрезок не включается.

#6. Прямоугольник рисуется при помощи функции **Rectangle** по двум координатам — левого верхнего и нижнего правого углов:

```
BOOL Rectangle(
    HDC hdc, // handle to DC
    int nLeftRect, // x-coord of upper-left corner of rectangle
    int nTopRect, // y-coord of upper-left corner of rectangle
    int nRightRect, // x-coord of lower-right corner of rectangle
    int nBottomRect // y-coord of lower-right corner of rectangle
);
```

Здесь, как и в случае с линией, Windows рисует прямоугольник до правой нижней координаты, **не включая ее (рис.2)**.

В нашем исходнике этому предшествует "извращенное" вычисление координат, таких, чтобы рисунок был более или менее симметричен. Мы просто разбиваем клиентскую область на 8 частей по вертикали и горизонтали и делаем отступ размером в одну часть от границ клиентской области. Формулы для этого — более чем простые:

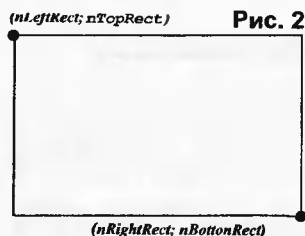


Рис. 2

```
x1 = xClient/8
y1 = yClient/8
x2 = 7*xClient/8
y2 = 7*yClient/8
```

(на самом деле эти формулы прямо соответствуют формулам из сишного варианта, однако для вычисления отступов нам больше подошли бы формулы $x2=xClient-x1$ и $y2=yClient-y1$. Но оставим это на совести Ч.Петзоляда).

Также обратите внимание на необычное использование инструкции *lea*. Воистину, нетривиальная инструкция ;). Впервые увидев *такой* вариант ее использования, я подумал, что глючу.

#7. Эллипс рисуется при помощи функции *Ellipse*:

```
BOOL Ellipse(
    HDC hdc,           // handle to DC
    int nLeftRect,     // x-coord of upper-left corner of rectangle
    int nTopRect,      // y-coord of upper-left corner of rectangle
    int nRightRect,    // x-coord of lower-right corner of rectangle
    int nBottomRect    // y-coord of lower-right corner of rectangle
);
```

Как нетрудно заметить, ее прототип — почти такой же, как и у функции рисования прямоугольника. Дело в том, что эта функция рисует эллипс с использованием "ограничивающего прямоугольника". Для того чтобы описать его, мы задаем координаты

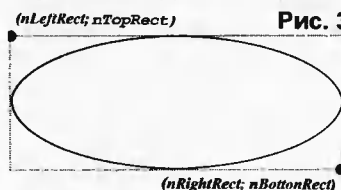


Рис. 3

этого гипотетического прямоугольника, в который будет *вписан* наш эллипс (рис.3).

В нашем примере это те же координаты, что и для рисования прямоугольника,

мы "достаем" их из переменных $x1, y1, x2, y2$.

#8. Прямоугольник со скругленными углами рисуется при помощи функции *RoundRect*:

```
BOOL RoundRect(
    HDC hdc,           // handle to DC
    int nLeftRect,     // x-coord of upper-left corner of rectangle
    int nTopRect,      // y-coord of upper-left corner of rectangle
    int nRightRect,    // x-coord of lower-right corner of rectangle
    int nBottomRect,   // y-coord of lower-right corner of rectangle
    int nWidth,        // width of ellipse
    int nHeight         // height of ellipse
);
```

Для описания этой фигуры нужно шесть параметров. Первые четыре — это координаты ограничивающего прямоугольника. А вот само скругление углов определяется длиной и шириной эллипсов, вписанных по углам этого прямоугольника. Все это проиллюстрировано на рис.4. Что касается нашего исходника, то там координаты ограничивающего прямоугольника высчитываются по формулам:

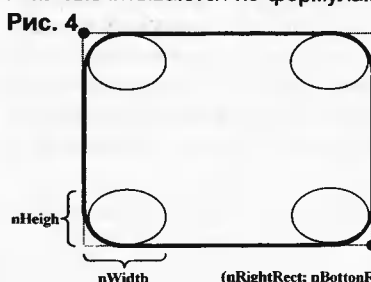


Рис. 4

```
x1 = xClient/4
y1 = yClient/4
x2 = 3*xClient/4
y2 = 3*yClient/4
```

Здесь $x1$ и $y1$ — это "по совместительству" еще длина и ширина эллипсов, определяющих степень закругления углов прямоугольника.

#9. Вас, должно быть, несколько озадачило то, что ни одна из этих функций не имеет такого параметра как цвет. Линии получаются черными на белом. Фигуры имеют черную рамку и залиты тем же цветом, что и фон клиентской области... Надеюсь, вы не думаете, что черный — это единственный цвет, которым можно рисовать в Windows?

В следующий раз мы познакомимся с подсистемой GDI поближе ;).

(Продолжение следует)

Современные профилировщики — что выбрать?

В.ГРИЦАН /ИИ-ТЕСН,

г.Торонто, Канада,

WWW: <http://wasrn.ru/>

Лирическое отступление (или вступление?)

Материал для этой статьи был собран, когда мне потребовалось разобраться в профилировке программы — что это такое и зачем нужно. В русскоязычном Интернете практически отсутствует информация на эту тему (за единственным, правда, исключением — статьи Стивена Е.Сайпа "Профилировщики программ Си++", уже капитально устаревшей). В англоязычной сети ситуация получше, но тоже далека от идеала. Именно поэтому данная статья и появилась.

Итак, перед началом я предполагаю, что читатель имеет твердое представление об ассемблере, неплохо владеет Си, разбирается, как компилировать и линковать программы. Также, очевидно, необходимо уметь пользоваться Visual Studio.

Существует множество профилировщиков, т.е. утилит, которые позволяют определить (запуская "подопытную" программу), где в ней присутствуют узкие места. Под словом "узкие" здесь понимаются "вызывающие значительное снижение скорости выполнения". В общем-то, вполне очевидно, что можно определить это все и по исходному коду, однако если размер исходного кода программы переходит некоторый порог, скажем, 100 Кб, то без профилировщика здесь уже не обойтись, т.к. в голове все уже не удержать. Можно, конечно, профилировщиком и не пользоваться, но тогда появляются о-очень уродливые и медленные шедевры.

Итак, программа должна быть специально подготовлена для анализа (хотя в некоторых случаях в этом нет необходимости, см. ниже) и запущена под профилировщиком, который выдаст кучу цифирь о времени выполнения различных функций. Затем все это прилежно анализируется, и код переписывается так, чтобы обеспечить более приемлемое быстродействие.

VTune

Это безусловный лидер среди профилировщиков. Он имеет очень много полезных функций и возможностей. Правда, за это приходится платить быстродействием и дисковым пространством. Краткая характеристика профилировщика VTune приведена в табл.1.

Поскольку я не системный администратор, то использую VTune лишь для анализа конкретных приложений, не интересуясь быстродействием системы. Если машина тормозит — надо удалить все лишнее и вставить новый процессор, вот и вся недолга. Поэтому весь дальнейший анализ будет исходить именно из этой предпосылки. Особо обратите внимание на выделенные графы таблицы — это самые главные опции при анализе приложения.

Работа ведется в так называемом "проекте". Принцип работы — такой же, как и в Visual Studio. Сначала создается проект, а потом уже производятся определенные действия. Проект надо должным образом при-



Табл. 1

CHARACTERISTICS	SAMPLING	CALL GRAPH	COUNTER MONITOR
Intrusiveness (вставки в код)	Non-intrusive. Вставок в код не происходит, периодически просматриваются активные в данный момент инструкции	Intrusive. Делает вставки в код (в пролог и эпилог каждой функции) для определения производительности	Non-intrusive. Используются счетчики производительности, встроенные в современные процессоры
System-wide performance data (данные о производительности системы в целом)	Выдает данные о всей системе в целом	Применим только к данному конкретному приложению	Выдает данные о всей системе в целом
Parent and child function relationship (отношение родительских и дочерних функций)	Учитывает лишь время, проведенное в конкретной функции	Учитывается все	Соотнесения информации, полученной со счетчиков, с кодом/функциями исследуемого приложения не происходит
Program flow (ход программы)	Выдает статистический анализ вызовов функций приложения. Очень полезно!	Совместно с Sampling дает множество очень полезной информации об исследуемом приложении в виде дерева вызова функций	Не определяет ничего, т.к. не соотнесено ни с каким приложением
System and microarchitecture-level performance data	Информация на уровне микроархитектуры — например, количество кеш-промахов (т.е. когда запрошенные CPU данные в кеше не находились), количество времени, потраченное на выполнение конкретных инструкций и т.п.	Не сообщает информацию о проблемах на уровне микроархитектуры	Информация на уровне микроархитектуры

готовить к использованию. Т.к. call graph осуществляет вставки в исполняемый код, то создатели VTune рекомендуют линковать программу с ключом /FIXED:NO, независимо от того, exe это файл, или dll.

FIXED:NO

Прочитайте спецификацию на PE-файл (MSDN). Здесь я обращаю ваше внимание всего на одно понятие — image base. Воспользуемся неизменным HIEW. Выберите любой файл в директории Windows (убедившись, что присутствует сигнатура PE00) и вызовите просмотр заголовка (F8):

```
Image base      00400000 | Subsystem
Console
```

Image base означает "базовый (иногда говорят "предпочтительный") адрес загрузки" — HIEW прибавляет число в поле Image base к обычным внутрифайловым смещениям для получения того адреса, по которому данная строка будет располагаться в оперативной памяти (этой операцией управляет строка в ini-файле ShowOffset = Local). IDA/Wdasm используют тот же механизм. Вы будете ставить bpx address в Soft-ICE именно по этим адресам. Однако значение этого поля не исчерпывается прибавлением к внутрифайловым смещениям определенного значения (для Win-EXE-PE — это 0x400000 по умолчанию, для DLL-PE — 0x10000000). Как вы знаете, файл загружается в собственное адресное пространство размером в 2 Гб, и начальный адрес загрузки определяется image base (но не ниже 0x400000). Relocations (fixups) под Windows куда не исчезли, они были, есть и будут. А теперь представьте, что два файла, принадлежащие одному процессу, имеют один и тот же базовый адрес загрузки (допустим, приложение имеет две dll, и обе они имеют один и тот же адрес по умолчанию). Что же произойдет в этом случае? Возникнет коллизия, и WinLoader будет вынужден производить перерасчет адресов, что неизбежно приведет к увеличению времени на загрузку. Перерасчет адресов определенных инструкций осуществляется с помощью информации из секции .reloc. Секция, в норме, не присутствует в exe-файлах, т.к., в основном, базовый адрес загрузки для них является и единственно возможным (dll же, как правило, очень редко бывают загружены по базовому адресу, по-

этому практически во всех dll есть reloc-секция). Если VTune требует переопределения базового адреса загрузки, то это означает, что программа будет перемещена в памяти, а без секции .reloc это невозможно. Подробнее смотрите статьи Мэтта Питрека на msdn или на его собственном сайте <http://www.wheaty.net/>.

Опция /fixed:no добавляется в опциях проекта Properties/Linker/Command line/Additional options. В версиях Visual Studio до .NET существовала опция линкера /PROFILE. Теперь /fixed:no стала ей заменой. Хотя говорить об исключении профилировщика из Visual Studio .NET все же не стоит. Вместе с VS.NET поставляется приложение ACT (Application Center Test), позволяющее оценивать производительность Web-страниц на основе VBScript. Лично я не могу сказать что-нибудь хорошее об этом языке (уж лучше JavaScript, по многим причинам, например, по соображениям кросс-платформенности), поэтому ACT здесь не рассматривается. У MS также существует профилировщик под SQL-сервер (он поставляется вместе с SQL Server) и анализатор производительности под IIS для ASP-страниц.

Также VTune требуется база данных о символах программы — pdb-файл. Как известно, pdb/dbg/мар-файлы используются для символьной отладки программы.

Символьная отладка

Существуют так называемые *символьные* отладчики. Они позволяют видеть инструкции, подобные call 1234567, как, например, call _time! Since является как раз таким отладчиком. Что нужно отладчику для символьной отладки (т.е. такой отладки, когда мы видим осмысленные имена функций, а если есть родной файл-исходник, то и строки исходного текста)? Ответ — файл с отладочной информацией. На данный момент времени существует, как минимум, три MS-формата таких файлов (Borland, Watcom и т.п., скорее всего, используют собственные разработки).

1. мар-файл — простой текстовый файл, с документированным и абсолютно понятным форматом. Сначала идет адрес строки, а потом — имя функции. Данный формат немного устарел, однако по-прежнему генерируется Visual Studio (о конкурентах не знаю). Существуют утили-



ты для перевода этого файла в формат, понятный отладчикам или дизассемблерам, например, такой как .sym.

2. dbg-файл — файл, имеющий внутренний формат (уже не текстовый). Формат полностью документирован Microsoft, однако Visual Studio непосредственно в отдельный файл его генерировать не умеет (не хочет?). Однако ей можно приказывать записать эту отладочную информацию непосредственно в исполняемый файл, откуда ее можно потом извлечь с помощью утилиты rebase. Полученный dbg-файл понимает IDA, его можно перевести в тар-формат и т.п.

3. pdb-файл — самая новая разновидность файла с отладочной информацией. Формат не полностью документирован. Недавно Microsoft выпустила DIA SDK, входящий в состав Visual Studio .NET (... \Visual Studio SDKs \DIA SDK), где описывается интерфейс взаимодействия с pdb-файлом (до этого приходилось использовать dbghelp.dll). Много полезной информации по данному вопросу можно найти в последних статьях Джона Роббинса и Мэтта Питрека. Также очень полезной является книга Свена Шрайбера "Недокументированные возможности Windows 2000", где этот формат неплохо освещен. К сожалению, даже IDA 4.30 не понимает последние версии pdb-формата, говоря что-то вроде "Unknown codeview Information format: RSIDS". Так что имеет смысл разобраться с DIA SDK и, кто знает, может, написать свой плагин к IDA.

Не помешает, кстати, повнимательнее присмотреться к содержимому каталогов \Util16 из дистрибутива Soft-ics. Может, что интересное углядите.

Эти установки в VS.NET задаются в свойствах линкера. Не забудьте это сделать!

Подопытным кроликом у нас будет мой собственный фильтр текстовых строк для бинарников — filter.c. Здесь я буду рассматривать только ключки исходного кода.

Уже говорилось, что можно обойтись и без профилировщика. Вот, например, так:

```
#include <time.h>
```

```
....;
```

```
startTime = clock ();
...; //функция, время выполнения которой необходимо знать
endTime = clock ();
```

```
printf ("\n\nProgram completed.\n\n");
printf ("Execution time in seconds: %f\n\n",
(double) (endTime - startTime) /CLOCKS_PER_SEC);
```

Использование этого метода на больших проектах превращается в головную боль. Целесообразнее использовать профилировщик.

Итак, компилируем и линкуем filter.c (помните о ключе /fixed:по и pdb-файлах). VTune встраивается в VC.NET, поэтому можно "позвать" его и работать оттуда, но лучше работать отдельно — приложение ресурсоемкое.

Общая философия работы проста — выделить функции приложения, занимающие больше всего процессорного времени, и попытаться их как-то оптимизировать. В этой

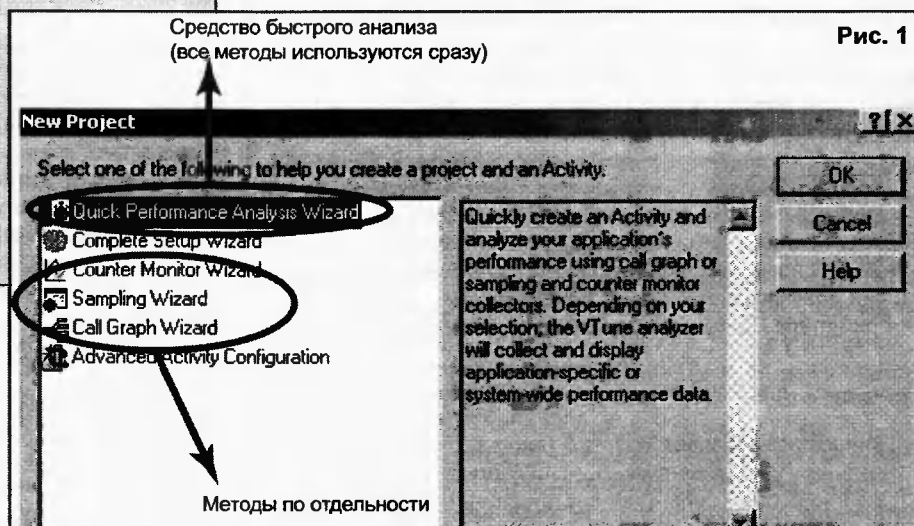


Рис. 1

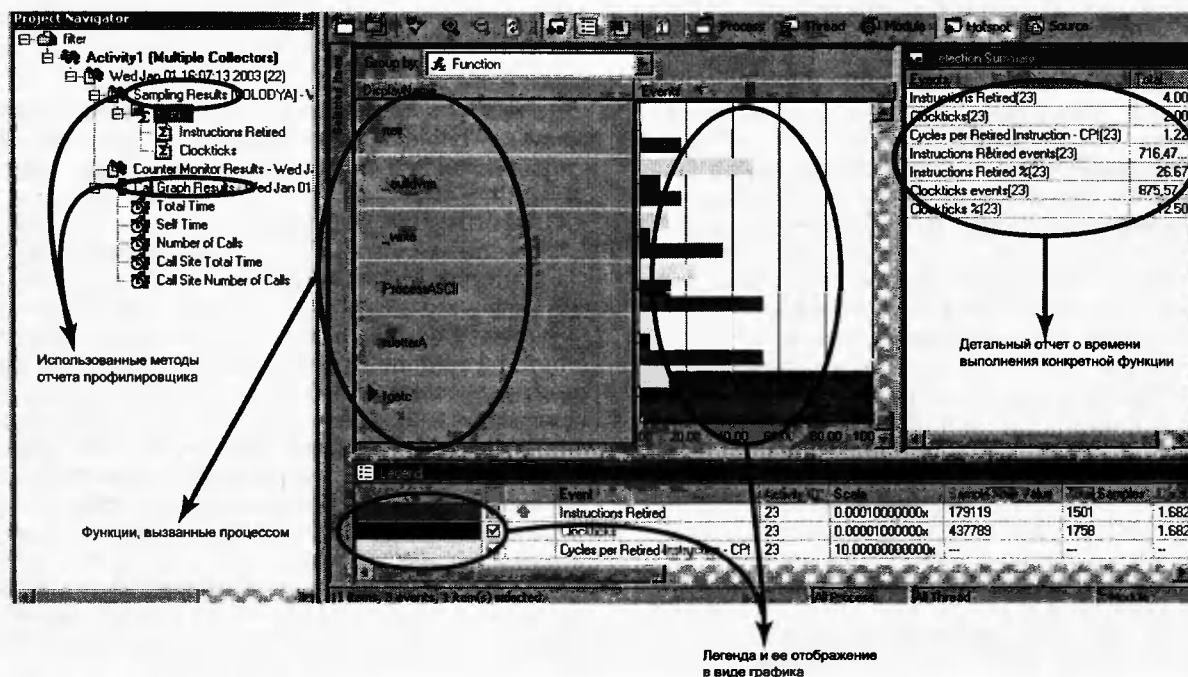


Рис. 2



статье рассказывается только о нахождении таковых. Оптимизация — тема, требующая отдельной статьи, а может, и книги.

Для того чтобы проанализировать быстродействие программы, будем использовать опции "Sampling" и "Call graph". При запуске VTune предложит выбрать нужный тип проекта (рис.1). Обычно я выбираю более универсальное решение, которое позволит собрать максимум информации о программе — Quick Performance Analysis Wizard. Учтите, что процесс этот очень ресурсоемкий, профилировщик сильно тормозит, поэтому нужна сильная машина, которая сможет вытянуть такую работу.

Выдается следующее диалоговое окошко, в котором предлагается задать аргументы командной строки (если таковые имеются). Мой filter — простой консольник, и аргументы командной строки в нем активно используются. Задаем, например, -a 4 C:\WINNT\notepad.exe — это выведет на экран все ASCII-строки, содержащиеся в notepad.exe, длиной не менее четырех символов. Терпеливо подождем, пока выполнятся многочисленные прогоны программы для сбора всех необходимых данных. Наконец-то (рис.2)!

Заметьте, что метод Sampling выводит информацию обо всех активных в данный момент времени на данной машине процессах. В списке всех процессов надо найти подопытную программу, и только тогда можно будет получить отчет о времени выполнения различных ее функций. Отчет может быть представлен в виде графика или таблицы. График нагляднее, таблица полезнее. Думаю, сначала стоит внимательно изучить график, а затем уже переходить к таблице. Для этого разберемся, что показано на графике, и каков смысл легенды.

Создатели VTune рекомендуют начинать с показателя CPI (cycles per instruction). Этот показатель отражает количество тактов процессора, затраченных на обработку инструкций конкретной функции. Ассемблерщики знают, что, скажем, операции умножения, деления или операции, использующие сложные модели адресации памяти, выполняются дольше, условные переходы — вообще напасть для современных процессоров, хорошо оптимизированный код старается обходиться без них, и т.д., и т.п. Так вот, диапазон значений по CPI в Vtune составляет:

ОК	Плохо
0.75	4

Если CPI высокий, это означает, что код надо переписать с использованием оптимизирующих техник, например, заменить возведение в степень двойки битовым сдвигом, использовать хитрые математические алгоритмы для замены умножения и деления и т.п. В принципе, хороший компилятор (т.е. компилятор от MS или от Intel) способен сделать это все за вас (только активируйте все опции оптимизации для Release-конфигурации), однако есть пара интересных приемов, облегчающих жизнь компилятора.

Оптимизация

Умение оптимально писать код приходит не сразу. Это долгий процесс, нет, вероятно, это бесконечный процесс. Я лишь дам пару-тройку отправных точек. У Intel на сайте есть великолепнейшие пособия по оптимизации, не поленитесь, загляните — эти посо-

бия перекрывают многие "самопальные" рукописи. Есть сайт x86.org — там тоже много чего интересного лежит. Ну и, наконец, just to give you a flavour, нечто вроде немного поправленной таблички от Paul Hsieh — одного из очень неплохих специалистов по оптимизации программ:

Так делать НЕ надо	А вот так — надо (только комментируйте такие места)
<code>x = y % 32</code>	<code>x = y & 31</code>
<code>x = y * 8</code>	<code>x = y << 3</code>
<code>if (x>=0 && x<8 && y>=0 && y<8)</code> { ... }	<code>if ((unsigned) (x y)) <8)</code> { ... }
<code>char array[z];</code> <code>for (i=0; i<z; i++)</code> { array[i]=2; }	<code>char array[z];</code> <code>char * p = &(array[0]);</code> <code>for (i=0; i<z; i++)</code> { *p+=2; }
Используйте алгоритм быстрой сортировки	Используйте сортировку объединением или лучевую сортировку (www.nr.com)
Кодить, кодить, кодить	Иногда перед этим не мешает подумать :)

Ну и, наконец, если показатели нормальные, а время выполнения все равно не удовлетворяет, есть смысл подумать об алгоритме. Может быть, следует как-нибудь переписать программу. Вообще-то, по правилам хорошего тона, программа должна быть основательно рассмотрена минимум дважды. Есть замечательная книга Алена Голуба "Правила программирования на C/C++". Она доступна для скачивания где-то в русской зоне Интернета. Очень рекомендую. Загляните также на сайт <http://www.kalinin.ru>.

Забыл сказать, что VTune поддерживает и мультипроцессорность — полезная черта. Современные ОС, типа Win2000, полностью поддерживают мультипроцессорность, и от программ, скомпилированных с ключом /MT или /MD, на мультипроцессорных платформах можно ждать сильного повышения производительности. Говоря обобщенно, чтобы не вступать в долгие дискуссии (тема-то действительно спорная), мультизадачные приложения на единственном процессоре выполняются медленнее, поэтому для простых приложений на единственном процессоре мультиточечность ни к чему. Однако, с точки зрения пользователя, если приложение выполняет какую-нибудь ресурсоемкую операцию, имеет смысл вынести ее в отдельный поток, чтобы имелась возможность прекратить этот поток, если надоест. Скажем, если бы проверка грамматики в Word выполнялась не в фоновом режиме, то компакт-диск с этим продуктом расколотили бы на голове у создателя.

Замечу, что спускаться можно и еще ниже. Так, если программа подготовлена как я говорил, то можно изучить исходный код функции, посмотреть его ассемблерное представление, получить совет по поводу потенциальных задержек и т.п.

(Окончание следует)



Преобразование псевдографических таблиц в формат HTML

(Продолжение. Начало в №5/2003)

Третий этап

Пусть в анализируемом тексте имеются, например, две такие таблицы (рис.4). Номера в ячейках показывают, в каком порядке они хранятся в списке ячеек, который был сформирован на втором этапе.

Рис. 4

0	1
4	5
8	9

2	3
6	7

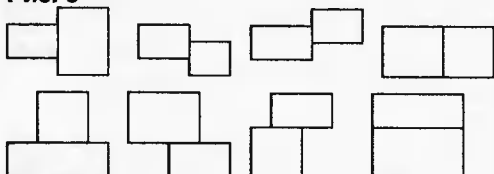
Это нам с вами при взгляде на рисунок сразу становится понятно, что там две таблицы, и что ячейки с номерами 0, 1, 4, 5, 8 и 9 принадлежат первой таблице, а ячейки с номерами 2, 3, 6 и 7 — второй таблице. Но компьютер этого пока еще «не знает». В его распоряжении — только список ячеек, в котором для каждой ячейки указаны ее координаты.

На третьем этапе как раз и определяется, какая ячейка какой таблице принадлежит, и выясняется общее количество таблиц в тексте (какой бы сложной формы они ни были).

Так как таблица представляет собой связанную совокупность ячеек, то от любой ячейки таблицы можно перейти к любой другой через смежные ячейки. Две ячейки считаются смежными, если они имеют общую сторону или ее часть (рис.5).

В программе имеется специальная функция (смежны), определяющая, являются ли две ячейки смежными, с помощью сравнения их координат.

Рис. 5



Таким образом, зная любую ячейку, принадлежащую таблице, можно, переходя от одной ячейки

к другой и пометая ячейки, в которых уже побывали, выявить все ячейки, принадлежащие этой таблице.

Тогда логика действий такова — берем из списка ячеек первую ячейку и считаем, что она принадлежит первой таблице. Выявляем все ячейки, принадлежащие первой таблице. Затем берем из списка ячеек первую из оставшихся ячеек и считаем, что она принадлежит второй таблице. Выявляем все ячейки, принадлежащие второй таблице. Эти шаги повторяются до тех пор, пока для всех ячеек не будет установлена принадлежность к какой-либо таблице.

На основе примера, изображенного на рис.4, рассмотрим, как эти действия осуществляются в программе.

В списке ячеек для каждой ячейки, кроме координат, хранится номер таблицы, которой эта ячейка принадлежит. Вначале для всех ячеек там число 255 — оно обозначает, что номер таблицы неизвестен (а если известен — там будет число от 0 до 254, т.е. всего в тексте может быть не более 255 таблиц).

В функции find_all_tables берется первая ячейка из списка (т.е. с номером 0 — и ячейки, и таблицы нумеруются с 0) и считается, что она принадлежит таблице 0. Чтобы установить для всех ячеек, принадлежащих этой таблице, соответствующий номер, вызывается функция find_table (на вход подается номер таблицы и номер ячейки, принадлежащей этой таблице).

Рассмотрим, как работает функция find_table. Вначале для ячейки, указанной при вызове (в данном случае 0) устанавливается соответствующий номер таблицы (тоже 0), и номер этой ячейки заносится в стек. Возникшая при этом ситуация изображена на рис.6, штриховкой на этом и следующих рисунках помечаются ячейки, для которых уже установлена принадлежность к таблице.

Далее начинается цикл, который будет выполняться, пока стек не опустеет.



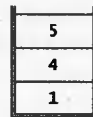
0	1
4	5
8	9

2	3
6	7

Рис. 6

Из стека извлекается верхнее значение (в данном случае 0), и находятся все ячейки, для которых еще не установлена принадлежность к какой-либо таблице, и которые одновременно являются смежными с той ячейкой, номер которой был извлечен из стека. В данном случае это ячейки 1, 4 и 5. Для них также устанавливается принадлежность к таблице 0, и их номера заносятся в стек (рис.7).

Стеком организован с помощью массива с количеством элементов, равным максимальному количеству ячеек (MAX_CELLS), так что он не может переполниться.



0	1
4	5
8	9

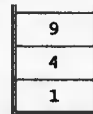
2	3
6	7

Рис. 7

При следующем выполнении цикла картина будет соответствовать рис.8.

И далее — рис.9.

При дальнейших выполнениях цикла со стека будут сниматься номера ячеек 8, 4, 1, но, т.к. для всех смежных с ними ячеек уже установлена принадлежность к



0	1
4	5
8	9

2	3
6	7

Рис. 8

таблице, новые значения в стек помещаться не будут. В результате стек опустеет, и цикл завершится. На этом работа функции find_table закончится.

Происходит возврат в функцию find_all_tables. Переменная all_tables (количество найденных таблиц), изначально равная 0, увеличивается на 1. Далее делается попытка найти в списке ячеек ячейку, которая не принадлежит никакой таблице. В данном случае это ячейка 2.

Опять вызывается функция find_table со следующими параметрами: номер таблицы — 1, номер ячейки, которая принадлежит этой таблице — 2. Подробно рассматривать то, что происходит при втором вызове этой функции, я здесь не буду.

После завершения работы функции find_table опять выполняется увеличение all_tables на 1 и проверяется, не найдется ли в списке ячейка, не принадлежащая никакой таблице. В данном случае — нет, следовательно, задача выполнена. На этом работа функции find_all_tables завершается.



Найденные таблицы оказываются пронумерованы, исходя из следующего принципа — слева направо и сверху вниз. Так происходит потому, что при каждом поиске ячейки, не принадлежащей никакой таблице, список ячеек просматривается от начала к концу, а в этом списке ячейки расположены как раз в порядке “слева направо, сверху вниз”.

Ошибочные ситуации, которые могут возникнуть на третьем этапе — обнаружено слишком много таблиц (больше 255).

Если третий этап завершился успешно, количество обнаруженных таблиц печатается на экране.

Четвертый этап

Это финальный этап, на котором происходит построение HTML-текста, описывающего таблицы. Количество таблиц уже известно (`all_tables`), какие ячейки принадлежат каждой таблице — тоже. И в функции `out_all_tables` в цикле последовательно выводятся все таблицы с номерами от 0 до `all_tables-1`.

Рассмотрим действия, которые выполняются при выводе каждой таблицы.

Сначала происходит переход от физических координат ячеек таблицы (рис.10а) к логическим координатам (рис.10б). Для этого на таблицу накладывается координатная сетка, так чтобы каждый угол каждой ячейки лежал на пересечении координатных линий, и, вместе с тем, не было “лишних” координатных линий.

Рис. 10а

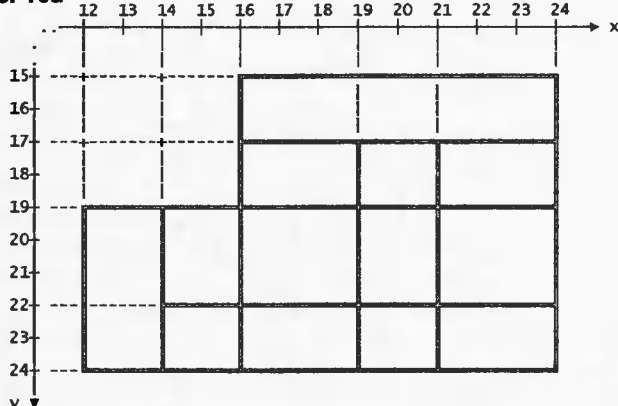
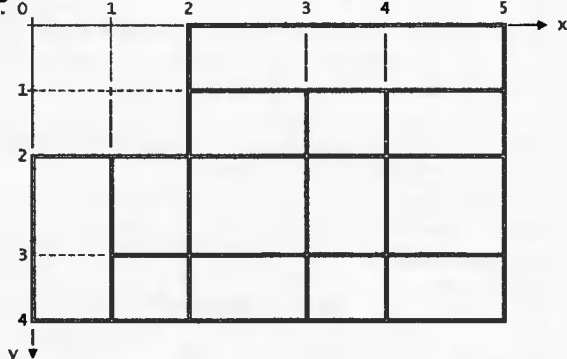


Рис. 10б



В программе этим занимается функция `make_log_xy`. Она заносит физические x-координаты углов каждой ячейки, относящейся к выводимой таблице, в массив `log_x`, а y-координаты — в массив `log_y`. Повторяющиеся значения при этом игнорируются. Далее массивы `log_x` и `log_y` сортируются по возрастанию. Для приведенного выше примера их содержимое будет следующим:

`log_x`: 12, 14, 16, 19, 21, 24.

`log_y`: 15, 17, 19, 22, 24.

Эти массивы как раз и используются для перехода от физических координат к логическим и наоборот. Чтобы определить логическую координату по физической, достаточно отыскать в массиве элемент, значение которого равно физической координате, тогда индекс этого элемента будет логической координатой. Обратный переход еще проще — значение элемента с индексом, равным логической координате, и будет равно искомой физической координате. Элементы массивов, напомним, нумеруются с 0.

Количество элементов массива `log_x` (в данном случае 6) — это количество различных x-координат (хранится в переменной `log_l`). Соответственно, количество элементов массива `log_y` (в данном случае 5) — это количество различных y-координат (хранится в переменной `log_h`). Тогда логическая ширина таблицы будет равна `log_l-1`, а высота — `log_h-1` (в данном случае, соответственно, 5 и 4).

Итак, физические координаты ячеек таблицы преобразованы в логические. Что происходит дальше? Язык HTML рассчитан на описание прямоугольных таблиц, поэтому, если таблица не является прямоугольной, к ней добавляются пустые ячейки, чтобы дополнить ее до прямоугольной. В программе этим занимается функция `make_space_cells`. Она формирует список добавляемых пустых ячеек (`space_cells`) и определяет их количество (`all_space`).

Ячейки добавляются так, чтобы по возможности минимизировать их количество (а значит, и размер выходного HTML-файла). Используется следующий алгоритм — в прямоугольнике, содержащем таблицу, последовательно слева направо и сверху вниз просматриваются все клетки 1x1. С помощью функции `test_def` проверяется, описана ли где-нибудь эта клетка, т.е. принадлежит ли она исходной таблице или какой-либо ячейке из списка добавленных пустых ячеек. Если нет, значит, надо добавить в список новую ячейку, содержащую эту клетку. Определяем логические размеры ячейки. Сначала определяем максимальную ширину — проверяем клетки справа от исходной, пока не наткнемся на уже где-то описанную клетку или на границу таблицы. Определив ширину, определяем максимальную высоту, проверяя ячейки снизу, пока не наткнемся на уже описанную клетку или на нижнюю границу таблицы. После того как ячейка занесена в список, просмотр клеток продолжается.

В данном случае к таблице будет добавлена одна пустая ячейка с логическими координатами левого верхнего и правого нижнего углов, соответственно — (0,0) и (2,2).

И, наконец, с помощью этих данных происходит формирование HTML-кода, соответствующего таблице (функция `out_table`).

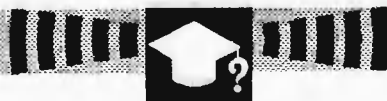
Таблица выводится по строкам сверху вниз (количество строк уже известно — `log_h-1`). При этом используется список ячеек исходной таблицы (`cells`) и добавленных пустых ячеек (`space_cells`). В каждой строке ячейки выводятся слева направо.

Рассмотрим вывод одной строки.

В указанных двух списках с помощью функции `find_cell` отыскивается ячейка с минимальной x-координатой левого верхнего угла (здесь и далее, если не оговорено особо, имеются в виду логические координаты), находящаяся в выводимой строке (то есть с y-координатой левого верхнего угла, равной номеру выводимой строки).

Если логическая ширина ячейки больше 1, то к тегу `<td>` (или `<th>`) добавляется атрибут `colspan`, значение которого равно логической ширине ячейки. Аналогично, если логическая высота ячейки больше 1, добавляется атрибут `rowspan`.

Содержимое ячейки выводится в выходной файл (чтобы добраться до содержимого, придется перейти от логических координат ячейки обратно к физическим).



Далее ищется и выводится ячейка со следующей по величине х-координатой, находящаяся в выводимой строке, и так далее, пока уже не найдется ни одной ячейки с х-координатой, большей чем у ячейки, выведенной в последний раз — это значит, что все ячейки, находящиеся в выводимой строке, уже выведены.

Ошибочные ситуации, которые могут возникнуть на четвертом этапе:

- количество пустых ячеек, добавляемых к какой-либо таблице прямоугольной формы, оказалось больше максимального (100 — определяется константой MAX_SPACE);
- не удастся записать информацию в выходной файл по каким-либо причинам (скажем, нет места на диске).

Электронная переписка

О.ВАЛЬПА,
E-mail: sandh@narod.ru

Электронная почта вошла в нашу жизнь, но тот, кто стал ее постоянным пользователем, уже наверняка настолько привык к ее преимуществам, что вряд ли вернется к обычному способу переписки с помощью бумаги и конвертов.

И в самом деле, какие преимущества имеет электронная почта?

Во-первых, скорость доставки сообщений — письмо может достичь адресата, находящегося на другом конце света, за считанные секунды!

Во-вторых, способ хранения и просмотра писем как в базе данных, позволяет сортировать их по различным критериям и очень быстро отыскивать необходимое письмо.

В-третьих, удобство рассылки по многим адресам одновременно, используя адресную книгу, хранящуюся как база данных.

В-четвертых, отправляемые и получаемые данные могут быть не только текстом, но и файлом-рисунком, звуковым файлом (живое голосовое письмо) и даже видеофайлом (живое изображение со звуком). Конечно, в последнем случае необходимо отправлять файлы большой длины, но сегодня существуют способы сильного сжатия файлов, что позволяет сократить их размеры в десятки раз.

На этом преимущества не кончаются, и многие читатели дополнили бы этот перечень. Однако поговорим теперь о недостатках.

На мой взгляд, одним из основных недостатков электронной почты является нарушение национальной кодировки письма при пересылке его через многие почтовые серверы или по причине несовместимости текста письма с кодировкой компьютера адресата, и, как результат — невозможность прочесть полученное письмо. Скорее, это даже не недостаток электронной почты, а неудобство, доставшееся нам в наследство от многоязыковости народов, населяющих нашу планету.

Дело в том, что русский (как, впрочем, и любой другой национальный, не англоязычный) алфавит изначально не присутствовал в таблицах кодировки букв, а появился лишь в новых типах кодировок, таких как CP866, ISO8859-5, KOI-8R, WIN1251, ГОСТ и т.п. В результате, при получении письма с русскими буквами в тексте файла, пользователь, не имеющий на своем компьютере поддержки русской кодировки символов, увидит вместо осмысленного текста набор бессмысленных символов из своего национального алфавита.

Первоначально существовала лишь кодировка ASCII, которая состояла из 128 символов и требовала для кодирования всего 7 битов (рис.1).

Листинг программы

Программа написана на Turbo C 2.0. Параметры компиляции должны быть следующими: модель памяти — huge, параметр Jump Optimize должен быть отключен, иначе могут быть ошибки (у меня это проявлялось в том, что при отладке некоторые строки программы не выполнялись).

Длина листинга программы в распечатанном виде — более 8 метров, так что привести его в журнале нет возможности. Размер исполняемого файла программы — около 20 Кб, так что его дампы тоже не получится привести. Но вы можете скачать эти файлы (а также пример ipi-файла) с сайта журнала или с моей web-страницы.

(Окончание следует)

Здесь H/L означает старшую/младшую тетраду кода символа. Например, для буквы "А" код — "41h".

Впоследствии появились национальные кодировки, которые расширили

таблицу до 256 символов и потребовали для кодирования уже 8 битов. Например, на рис.2 приведена кодировка CP866, содержащая символы русского алфавита (кириллицы).

При пересылке писем по электронной почте некоторые серверы могут "обрезать" передаваемые коды до 7 битов, в результате до адресата дойдет искаженное письмо, где русские буквы примут вид спецсимволов или других знаков. Именно по этой причине рекомендуется использовать при составлении и отправке письма кодировку

КОИ-8, где русские буквы располагаются в таблице кодировки так, что даже при усечении передаваемых байтов до 7 битов буквы русского алфавита заместятся на созвучные им буквы латинского алфавита, и письмо, с небольшим трудом, можно будет прочесть (рис.3).

Рассмотрим это на конкретном примере. Например, я написал письмо, состоящее из фразы: "ПРИВЕТ АЛЕКСАНДР!", и отправил его в кодировке КОИ-8. Допустим, что при пересылке письма произошло усечение пересылаемых байтов символов до семи битов. Тогда адресат получит

Рис. 1

Таблица кодировки символов ASCII

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

Рис. 2

Таблица кодировки символов CP866

H/L	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	неотображаемые спецсимволы															
1	[]															
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
7	q	r	s	t	u	v	w	x	y	z	{		}	~	^	^
8	A	B	V	G	D	E	J	З	И	Й	К	Л	М	Н	О	П
9	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Э	Ю	Я	
A	a	b	v	g	d	e	j	з	и	й	к	л	м	н	о	п
B	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	э	ю	я	
C	А	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	
D	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Э	Ю	Я	
E	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	э	ю	я	

Рис. 3

Таблица кодировки символов КОИ-8

H/L	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	неотображаемые спецсимволы															
1	[]															
2		!	"	#	\$	%	&	'	(	)	*	+	,	-	.	/
3	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
7	q	r	s	t	u	v	w	x	y	z	{		}	~	^	^
8	А	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	
9	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Э	Ю	Я	
A	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
B	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	э	ю	я	
C	А	В	Г	Д	Е	Ж	З	И	Й	К	Л	М	Н	О	П	
D	Р	С	Т	У	Ф	Х	Ц	Ч	Ш	Щ	Ъ	Ы	Э	Ю	Я	
E	а	б	в	г	д	е	ж	з	и	й	к	л	м	н	о	п
F	р	с	т	у	ф	х	ц	ч	ш	щ	ъ	ы	э	ю	я	


```

c2 = ' ';
do
{
if((c=getc(file_inp))==EOF) break;

if (c==0x96) c2 = 'S';
if (c==0x97) c2 = 'H';
if (c==0x98) c2 = 'H';
if (c==0x99) c2 = 'C';
if (c==0x9e) c2 = 'U';
if (c==0x9f) c2 = 'A';
if (c==0xe6) c2 = 's';
if (c==0xe7) c2 = 'h';
if (c==0xe8) c2 = 'h';
if (c==0xe9) c2 = 'c';
if (c==0xee) c2 = 'u';
if (c==0xef) c2 = 'a';
c1 = tabl(c);
putc(c1,file_out);
if (c2 != ' ') putc(c2,file_out);
c2 = ' ';
}
while(1);
}
tabl(cin)
{
int cout[256]={
0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,
0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,
0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,
0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3a,0x3b,0x3c,0x3d,0x3e,0x3f,
0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4a,0x4b,0x4c,0x4d,0x4e,0x4f,
0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5a,0x5b,0x5c,0x5d,0x5e,0x5f,
0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,
0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x7f,
'A', 'B', 'V', 'G', 'D', 'E', 'G', 'Z', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
'R', 'S', 'T', 'U', 'F', 'H', 'T', 'C', 'S', 'S', 'Y', 'Y', 'E', 'J', 'J',
'a', 'b', 'v', 'g', 'd', 'e', 'g', 'z', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
0xb0,0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7,0xb8,0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,
0xc0,0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xca,0xcb,0xcc,0xcd,0xce,0xcf,
0xd0,0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,0xd9,0xda,0xdb,0xdc,0xdd,0xde,0xdf,
'r', 's', 't', 'u', 'f', 'h', 't', 'c', 's', 's', 'y', 'y', 'e', 'j', 'j',
'E', 'e', 0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0xfc,0xfd,0xfe,0xff
};
return (cout[cin]);
}

/* ===== */
/* Программа win lat преобразует буквы русского алфавита */
/* кодировки WIN1251 из исходного файла, в созвучные буквы */
/* латинского алфавита и формирует новый файл с расширением 'lat' */
/*
/* Автор: Valpa Oleg */
/* Дата: 27.11.1999 */
/* ===== */

#include <stdio.h>

int tabl (int);

main(argc,argv)

int argc;
char **argv;
{
int c,c1,c2;
int n=0;
char name[13];
FILE *file_inp, *file_out;
register int i;
register char **p;
if(argc == 1)
{
printf("\nПрограмма win lat преобразует буквы русского алфавита ");
printf("\nкодировки WIN1251 из исходного файла, в созвучные буквы");
printf("\nлатинского алфавита и формирует новый файл ");
printf("\ncs расширением 'lat' \n ");

printf("\nПример использования программы: win_lat [имя исходного файла]");
printf("\n\nВведите имя исходного файла : ");
scanf("%s",argv[1]);
}

```

```

while((file_inp=fopen(argv[1],"rb"))==NULL)
{
printf("\nНе могу открыть файл с таким именем !!!");
printf("\nВведите имя исходного файла > ");
scanf("%s",argv[1]);
}

```

```

do
{
if( {argv[1][n]=='.'} || (argv[1][n]==0) ) break;
name[n]=argv[1][n];
n++;
}
while(1);

name[n+0]='.';
name[n+1]='l';
name[n+2]='a';
name[n+3]='t';
name[n+4]=0;

```

```

file_out=fopen(name,"w+b");
c2 = ' ';
do
{
if((c=getc(file_inp))==EOF) break;

```

```

if (c==0xd6) c2 = 'S';
if (c==0xd7) c2 = 'H';
if (c==0xd8) c2 = 'H';
if (c==0xd9) c2 = 'C';
if (c==0xde) c2 = 'U';
if (c==0xdf) c2 = 'A';
if (c==0xf6) c2 = 's';
if (c==0xf7) c2 = 'h';
if (c==0xf8) c2 = 'h';
if (c==0xf9) c2 = 'c';
if (c==0xfe) c2 = 'u';
if (c==0xff) c2 = 'a';
c1 = tabl(c);
putc(c1,file_out);
if (c2 != ' ') putc(c2,file_out);
c2 = ' ';
}
while(1);

```

```

tabl(cin)
{
int cout[256]={
0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,
0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,
0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,
0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3a,0x3b,0x3c,0x3d,0x3e,0x3f,
0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4a,0x4b,0x4c,0x4d,0x4e,0x4f,
0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5a,0x5b,0x5c,0x5d,0x5e,0x5f,
0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,
0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x7f,
0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x8a,0x8b,0x8c,0x8d,0x8e,0x8f,
0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9a,0x9b,0x9c,0x9d,0x9e,0x9f,
0xa0,0xa1,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7, 'E', 0xa9,0xaa,0xab,0xac,0xad,0xae,0xaf,
0xb0,0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7, 'e', 0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,
'A', 'B', 'V', 'G', 'D', 'E', 'G', 'Z', 'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
'R', 'S', 'T', 'U', 'F', 'H', 'T', 'C', 'S', 'S', 'Y', 'Y', 'E', 'J', 'J',
'a', 'b', 'v', 'g', 'd', 'e', 'g', 'z', 'i', 'j', 'k', 'l', 'm', 'n', 'o', 'p',
'r', 's', 't', 'u', 'f', 'h', 't', 'c', 's', 's', 'y', 'y', 'e', 'j', 'j'
};
return (cout[cin]);
}

```

Желаю всем приятной переписки.



Рисунок
О. Попова



С.РЮМИК,
г.Черногов.

Чип-“невидимка” в “PlayStation”

*“К любому замку ключ подобрать можно”.
Татарская поговорка*

Казалось бы, что общего между американским военным самолетом “F-117A” и японской игровой приставкой “Sony PlayStation”? Ответ парадоксален — и в том, и в другом случае используется технология “стелс” (от англ. “stealth” — хитрость, уловка). Самолет она делает невидимым для большинства типов радаров, а игровую приставку — неуязвимой для ультрасовременных игр, проверяющих наличие в ней мод-чипа (МЧ, ModChip).

Так сложилось, что на территории стран СНГ фирменных приставок “PlayStation” (PSX), на которые бы распространялась гарантия фирмы “Sony”, практически нет. Несмотря на красивые пломбы, все имеющие хождение приставки хотя бы раз да вскрывались. Зачем? Для установки МЧ. Это позволяет запускать на PSX диски любых регионов, их CD-R-копии, а также игры, переведенные на русский язык.

Юридическая законность подобных действий в разных странах до сих пор является предметом дискуссий. Например, раздел 47 закона об авторском праве Австралии предусматривает возможность копирования программного обеспечения, если имеется его оригинал. На основании этого Положения юристы ставят под сомнение правомерность ввода фирмами различных региональных защит в свои программные продукты. Как бы там ни было, но в любом случае никто не запрещает пользователю модернизировать свою приставку установкой МЧ с превращением ее из фирменной в PlayStation-совместимую. С точки зрения ремонтника, выход МЧ из строя является отказом обычного радиоэлемента. О том, как его восстановить, как правильно подключить и эксплуатировать, пойдет речь дальше.

Существует много разновидностей МЧ, отличающихся типом примененного контроллера, кодом прошивки, схемой подключения. До недавнего времени это было не принципиально, поскольку базовый алгоритм обхода защиты не изменялся, и МЧ от “старых” моделей PSX с незначительными поправками переходили на “новые” модели.

Однако время не стоит на месте. Изготовители программного и аппаратного обеспечения приняли меры, затрудняющие эксплуатацию нефирменных устройств. Появились новые

игры и новые версии приставок, поддерживающие дополнительную защиту. Чтобы лучше уяснить ситуацию, обратимся к истории.

Историческая справка

Днем рождения PSX считается 3 декабря 1994 г., когда ее первые экземпляры появились в продаже в магазинах Японии [1]. В течение следующего года, вслед за “японскими”, появились “американские” и “европейские” модели PSX. Разделение по географическому признаку коснулось не только приставок, но и лазерных компакт-дисков к ним. Соответственно, диски с отметкой “PAL” могли запускаться только на “европейских”, диски “NTSC_U/C” — только на “американских”, а диски “NTSC/J” — только на “японских” версиях приставки. Это, по мнению руководства фирмы “Sony”, должно было упорядочить финансовые потоки трех ее подразделений: SCEI (“Sony Computer Entertainment Inc.”, Япония), SCEA (“Sony Computer Entertainment America”, США, Канада), SCEE (“Sony Computer Entertainment Europe”, Европа, Австралия).

Система региональной защиты построена по принципу “ключ — замок”, то есть в ПЗУ приставки и на треке диска содержатся кодовые слова, при совпадении которых разрешается дальнейшая работа. Первоначальный вариант защиты оказался малоэффективным из-за неточностей в ПЗУ BIOS. Пользователи быстро сообразили, что запустить диск любого региона можно, не вскрывая приставки, при помощи специального загрузочного диска (“swar boot-disk”).

Технология получила название “дисковая подкачка” и заключалась в следующем. Устанавливается загрузочный диск “родного” формата. В приставке острым предметом (пружиной, иглой) блокируется кнопка поднятия крышки доступа к диску. Включается питание и ожидается, пока пройдет начальная проверка кода региона. После этого диск останавливается и меняется игровым другого региона.

Такая процедура была возможна в моделях SCPH-100x образца 1994-95 гг. Здесь и дальше буква “х” в названии модели означает одну из четырех разновидностей: “0” — “японская”, “1” —

“американская”, “2” — “европейская”, “3” — “азиатская”. В последующем в ПЗУ PSX были внесены соответствующие коррекции, и “swar”-метод перестал приносить результат.

Для пользователей закончилось “розовое” время, и наступил черед “хирургических” решений, связанных с внесением изменений в электрическую схему приставки. В начале 1996 г. один из западных инженеров, работавший в то время по контракту в Гонконге (его имя история умалчивает), разработал первый МЧ, получивший название “Hong Kong”. Чип содержал плату с контроллером, который встраивался внутрь приставки и позволял обходить региональную защиту без загрузочного диска. Стоимость установки первого МЧ варьировалась от 40 до 80 USD.

В середине 1996 г. в Нидерландах параллельно (и независимо) был разработан другой вариант МЧ стоимостью 25 USD. Однако вплоть до начала 1997 г. эта продукция считалась элитарной и практически невозможной для повторения в домашних условиях.

1997 г. стал поворотным, благодаря стараниям Скотта Райдера (Scott Rider, псевдоним — “Old Crow”), который не только смог разобраться в работе МЧ “Hong Kong”, но и разместить в Интернете исходный код улучшенной версии его прошивки.

Чип Райдера получил название “стандартный” (“Standard”, “Std”) и имел стоимость 10...20 USD. С переходом на более дешевый тип микросхемы контроллера, цена снизилась до 5...15 USD. Известно несколько модификаций стандартного МЧ для разных версий приставок — SCPH-100x, SCPH-550x, SCPH-700x.

Фирма “Sony” в долгу не осталась. Ее новые модели SCPH-750x, SCPH-900x, выпущенные в 1998-99 гг., использовали улучшенную схемотехнику контроллера CD-ROM. Прежние уязвимые точки подключения МЧ теперь оказались спрятанными внутри СБИС, и для стандартного чипа сделались недоступными. Тем не менее, были найдены аналогичные по функциям точки подключения в других местах схемы PSX. Прошивку чипа Райдера пришлось модифицировать путем усложнения структуры выходного информационного сигнала.

Не успели пользователи побороть аппаратные проблемы, как появились программные. В конце 1998 г. пришли первые сообщения о выпуске в Японии, а затем и в США новых игр, которые обнаруживали присутствие МЧ в приставке и отказывались дальше загружаться. Их окрестили "антимод"-играми ("anti-mod games"). В табл. 1 в хронологическом порядке перечислены десять первых защищенных японских игр.

Табл. 1

Название игры	Фирма	Дата
I.Q. Final Trial	SCEI	(1998-11)
Poporogue	SCEI	1998-11-26
Pocket Muu Muu	SCEI	1999-02-04
Final Fantasy VIII	SQUARE	1999-02-11
Silent Hill	KONAMI	1999-03-04
Global Force	SCEI	1999-03-11
Chocobo Racing	SQUARE	1999-03-18
UmJammer Lammy	SCEI	1999-03-18
Katte ni Momotenshi II	KONAMI	1999-03-18
Magical Tetris Charange	CAPCOM	1999-03-18

Особый интерес вызвала игра "Dino Crisis", вокруг которой развернулись целые баталии. Именно она стала полигоном для отработки методов обхода защиты. Вначале проблему решали ручным подбором задержек. Для этого в МЧ, в дополнение к стандартному режиму, добавили специальный "дино-режим" ("dino-mode"). Пользователь, нажимая определенное количество раз кнопку сброса "RESET" приставки, мог подобрать такую величину задержки, при которой игра начинала устойчиво запускаться.

Чипы, имеющие несколько режимов, получили название "мультистандартные" ("multi-mode chip", "MM-chip"). Их было выпущено несколько версий, но наибольшую популярность получила та, которая использует стелс-технологии ("stealth-technology") для автоматического обхода защиты "антимод"-игр. Стелс-режим подобен "шапке-невидимке". Он делает работу МЧ прозрачной для игровой программы. Как только программа начинает поиск импульсов, исходящих от МЧ, его функционирование останавливается, и наоборот. К слову сказать, на многих игровых дисках, прошедших через руки домашних хакеров, "антимод"-защита снята программным путем, поэтому стелс-режим для них не нужен.

"Мяч" в очередной раз оказался на стороне фирмы "Sony", и ответ не заставил себя долго ждать. В новую модель приставки SCPH-10x "PSone"

образца 2000 г. была включена двойная защита проверки регионального кода на уровне BIOS. Стандартный МЧ успешно преодолевал первую часть защиты, но был бессиль перед второй. Не помогал тут и стелс-режим. Долгое время единственным выходом из создавшегося положения считалась физическая замена микросхемы ПЗУ BIOS в "PSone". Однако эта операция технологически весьма сложная, и рекомендовать ее для широкого повторения было нельзя, да и гарантировать абсолютную программную совместимость — тоже.

Чуть позже техническая мысль электронщиков и программистов помогла найти оригинальное решение без замены ПЗУ. После дизассемблирования начального участка программы BIOS "PSone" был вычислен точный момент времени, в который нужно забло-

кировать (видоизменить) информацию на шине данных центрального процессора (ЦП), чтобы пройти вторую часть защиты.

Оказалось, что с блокировкой данных успешно может справиться все тот же контроллер МЧ. Модернизированный таким образом чип вновь стал выполнять свои прямые функции, получив уважительное название "ONEchip". По функциональному назначению он относится к мультистандартным, поскольку содержит "дино-режим" (как запасной) и режим отключения МЧ.

Структура лазерного диска PSX

Фирма "Sony" многое сделала в PSX для обеспечения защиты авторских прав своей продукции. Например, фирменные лазерные игровые диски имеют черный цвет покрытия. Если оставить в стороне эстетические нюансы, то налицо первая ступень защиты от копирования.

На естественный вопрос: "А как же луч лазера отражается от черной поверхности, и где находятся микровыступы и микроуглубления?" следует ответ: "Черное поликарбонатное покрытие поглощает видимый свет, но отлично пропускает лучи инфракрасного диапазона, в котором работает лазерный диод. Отражение происходит от тонкой алюминиевой матрицы (рис. 1), расположенной на внешней стороне диска. Технология получения алюминиевого сплава матрицы является производственным секретом.



Рис. 1

Поверх матрицы наносится трафаретный рисунок с названием игры".

Идея цветных дисков не нова. В начале 90-х годов фирма "Tandy" разработала перезаписываемые оптические диски THOR на основе технологии "красящий полимер". Тем не менее, изготовить в кустарных условиях лазерный диск черного цвета совсем не просто. Расчет инженеров фирмы "Sony" оказался верным — подделок нет, а нефирменные диски легко отличить от лицензионных по серебристому цвету поверхности. Такая практика была продолжена в новой игровой приставке "PlayStation 2", где игровые диски окрашены в синий цвет.

"Черные" и "синие" диски по физическим параметрам ничем не лучше и ничем не хуже обычных CD аналогичного качества. С легкой руки фирмы "Sony" цветовая окраска стала применяться и другими изготовителями, например, известны CD-R 700 фирмы "Dysan Universal" черного цвета.

Вторая ступень защиты заключается в форматировании диска по стандарту CD-ROM/XA (CD-XA). Этот стандарт был создан в 1990 г. по инициативе трех фирм — "Philips", "Microsoft", "Sony". Предполагалось, что в будущем он сможет стать основным для зарождающегося направления мультимедиа.

Буквы "XA", которые дополняют "CD-ROM" в названии, означают "расширенную архитектуру" ("eXtended Architecture"). Расширение, по сравнению с обычным компьютерным диском ("Yellow Book"), означает ввод некоторых компонентов из стандарта CD-I ("Green Book").

В частности, формат CD-XA позволяет располагать сжатые компьютерные, аудио- и видеоданные на диске вперемежку ("interleaving"). Например, за первым видеофрагментом может следовать сегмент с его звуковым сопровождением, после которого располагается следующий видеофрагмент, и т.д. Все это способствует повышению синхронности воспроизведения звука и видео, что важ-



но в играх. Компрессия аудиоданных по методу ADPCM позволяет вместо обычных 74 минут хранить на диске CD-XA до нескольких часов качественной музыки и речи. Плата за удовольствие — необходимость разделения потоков специальными аппаратными средствами в контроллере CD-ROM.

Согласно стандарту CD-XA, на треке могут располагаться сектора со сжатием информации (Mode2) двух типов: Form1 длиной 2048 байтов — для компьютерных данных, и Form2 длиной 2324 байта — для ADPCM аудио- и видеоданных. Файловая система соответствует международному стандарту ISO-9660.

Во времена появления PSX не каждый CD-ROM мог читать диски CD-XA и, тем более, их переписывать. Это вселяло определенный оптимизм в части затруднения нелегального копирования игровых дисков.

Практика не оправдала надежд разработчиков. Сейчас любой CD-ROM, CD-R, CD-RW при соответствующей программной поддержке уверенно читает диски CD-XA. Более того, сам стандарт не получил широкого распространения. Причина банальная — технический прогресс резко повысил быстродействие устройств чтения информации. Современные CD-ROM имеют в 20...30 раз большую скорость передачи данных, чем у PSX, и на порядок более высокий объем внутреннего буферного ОЗУ. Следовательно, можно обойтись без чередования данных на диске и без введения дополнительной аппаратуры распаковки в контроллере CD-ROM.

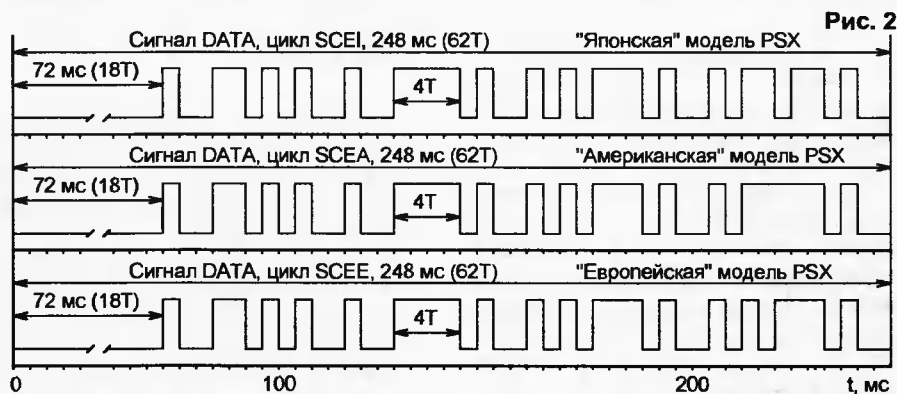
Третья ступень защиты — это физическая структура первых 16 секторов диска. Известно, что они содержат данные в формате CD-XA/Mode2 Form1. Каждый из секторов имеет длину 2048 байтов и заканчивается записью кода исправления ошибки ECC (Error Correction Code) и кода обнаружения ошибки EDC (Error Detection Code). На фирменных дисках PSX значения ECC и EDC для начальных 16 секторов специально обнулены. Именно по ним ЦП приставки распознает музыкальные audio-CD (ECC и EDC не равны нулю) и PSX-диски (ECC и EDC равны нулю).

Для пользователей это создает определенные трудности при копировании фирменных игр на CD-R, поскольку многие копировщики автоматически восстанавливают правильные коды ECC и EDC. При перезаписи CD-R требуются специальные про-

граммы, которые умеют записывать диски в режиме DAO ("Disk At Once") с физическим копированием информации секторов "как есть".

Заметим, что получение копии игры на CD-R для личных целей не является противозаконным актом. Общеизвестным считается, что пользователь имеет право сделать для себя одну копию игры, оригинал которой у него имеется в наличии. Созданные копии нельзя отправлять по почте, передавать через Интернет, продавать. За соблюдением такого порядка во многих странах следит организация IDSA (International Digital Software Association), опытные адвокаты которой могут затруднить жизненные условия любому незадачливому коммерсанту.

Четвертая ступень защиты — это логическая организация данных в первых пяти секторах диска, где располагается информация о его географической принадлежности. Там записаны коды аббревиатуры одной из региональных организаций, соответственно "SCEI" (Япония), "SCEA" (США), "SCEE" (Европа). Когда ЦП приставки при чтении диска получает эти коды, он сравнивает их с зашитыми в ПЗУ BIOS и выносит вердикт — продолжить работу с диском или вывести на экран телевизора сообщение об ошибке.



Для преодоления этой защиты и служит МЧ. Генеральная идея заключается в подмене кодов, поступающих в ЦП с первых секторов диска, кодами от МЧ, которые соответствуют версии приставки.

Структура кодов обхода защиты

На рис.2 приведены временные диаграммы трех циклов информационного сигнала обхода защиты, которые поступают в ЦП PSX [2]. В качестве примера рассмотрим цикл SCEE для "европейской" модели приставки. Если принять в качестве одного такта отрезок времени 4 мс, то формула кода

сигнала будет выглядеть так — 18T-T-2T-2T-T-T-T-T-2T-2T-4T-T-T-2T-T-T-T-3T-T-T-2T-T-T-T-T-3T-T-T-2T.

Начальный нулевой интервал времени 18T можно считать маркером цикла и отбросить как неинформативный. Далее, представим оставшийся код в виде последовательности единиц и нулей с тактом 4 мс — 1-00-11-0-1-0-1-00-1-00-1111-0-1-00-1-0-1-0-111-0-1-00-1-0-1-0-111-0-1-00. Итого — 44 бита, передаваемых в последовательном формате со скоростью 250 битов в секунду. Разделим их на 4 группы по 11 битов в каждой — 10011010100; 10011110100; 10101110100; 10101110100.

Если приглядеться, то каждая группа начинается с одного стартового единичного бита и заканчивается двумя стоповыми нулевыми битами. Внутри них располагается ровно 8 битов информации, то есть в каждой группе по одному "полезному" байту — 00110101; 00111101; 01011101; 01011101. Запишем их по-арабски, справа налево — 10101100; 10111100; 10111010; 10111010. Инvertируем каждый бит и переведем в шестнадцатеричную форму — 01010011=53h; 01000011=43h; 01000101=45h; 01000101=45h. Если каждому байту сопоставить одну букву английского алфавита ASCII, то получается слово "SCEE" — аббревиатура названия европейского филиала

фирмы "Sony"!

Аналогичные рассуждения для двух других разновидностей циклов приводят к названиям "SCEI" и "SCEA". Эту четкую логику, казалось бы, в хаотичном наборе "единиц" и "нулей" впервые разглядел Скотт Райдер.

Поскольку в PSX соблюдается сквозной принцип преемственности, то указанная процедура действительна для всех без исключения нынешних и всех последующих моделей приставки. Разница будет заключаться лишь в схемотехнических особенностях ввода указанной информации в схему PSX.

(Продолжение следует)



И.ДЗЮБА,
г.Великий Новгород.

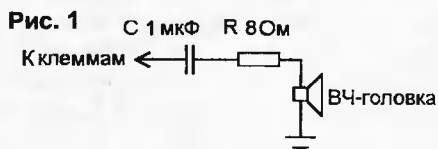
Исследование УЗЧ аудиокарты

Мне понравилась статья В.Федорова из г.Екатеренбурга [1]. Во-первых, очень интересный подход, во-вторых, очень много полезного материала к размышлению. И еще какого материала! После прочтения этой статьи мне самому стало интересно поэкспериментировать. Думаю, осциллограф есть, программа "Audio Test", генерирующая синусоиду и меандр на выходе аудиокарты компьютера, тоже есть, почему бы и нет!

Компьютер генерировал меандр и подавал его на акустику "Радиотехника S-30" (сопротивление — 4 Ом) с выхода "Speaker" аудиокарты. Звуковая карта у меня, вроде бы, не самая плохая, лучшая из всех, что у меня были (Gravis UltraSound MAX), на "звук" которой у меня никогда нареканий не было.

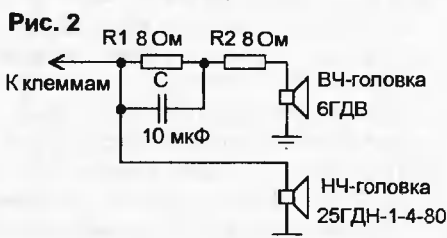
Выставил частоту 1 кГц, "посмотрел" — меандр нормальный. Стало интересно, а что же доходит до ВЧ-динамика, если весь фильтр в колонке состоял из последовательно включенных конденсатора 1 мкФ и резистора 8 Ом (рис.1)?

На выходе фильтра, как и предпо-



лагалось, все очень мало напоминало меандр, скорее, все что угодно, но не меандр. Было решено доработать фильтр, как указывалось в вышеупомянутой статье.

После доработки разница чувствовалась просто буквально. Появилась, так сказать, четкость, верность воспроизведения, и меандр соответственно "исправился". Музыку просто хотелось слушать снова и снова. Вот тогда я понял, что все-таки, как ни "крути" усилитель, колонки тоже надо дорабатывать. Особенно с таким простым фильтром. Иначе результаты не почувствуешь. Номиналы, правда, у меня получились немного другие (рис.2).

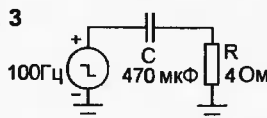


Затем мне стало интересно "заглянуть глубже", и я решил смоделировать схему колонки в программе "Micro-Cap V".

Вот здесь как раз и проявились некоторые интересные моменты. Дело в том, что в тот период меня интересовали такие вещи как постоянные времени. Сопротивление акустической системы (далее АС) у меня было 4 Ом. А вот что и как зависит от емкости разделительного конденсатора, стоящего в усилителе? Было, конечно, понятно, что чем она больше, тем лучше, но что там при этом происходит?

Для проведения анализа я взял только НЧ-динамик АС, считая, что его сопротивление чисто активное (4 Ом). В данном случае это вполне допустимо. Модель получается крайне простой (рис.3).

Рис. 3



Имеется источник меандра, допустим, частотой 100 Гц, разделительный конденсатор (соответственно для усилителей, в которых он есть на выходе) и динамик.

Для такой модели получаем следующие АЧХ и ФЧХ (рис.4, ФЧХ — верхняя кривая, АЧХ — нижняя).

Вертикальная линия-метка находится примерно на нижней граничной частоте, т.е. около -3 дБ по АЧХ. При этом хорошо

Рис. 4

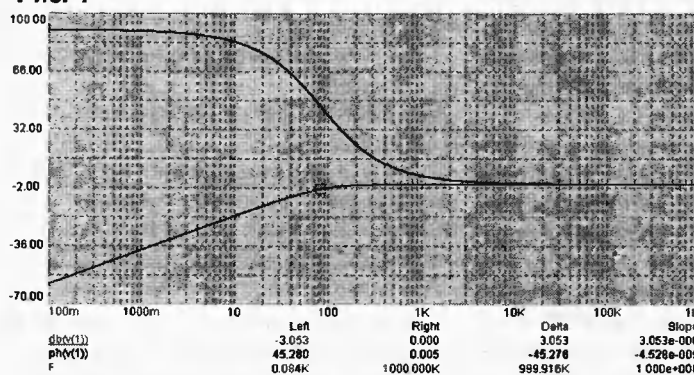
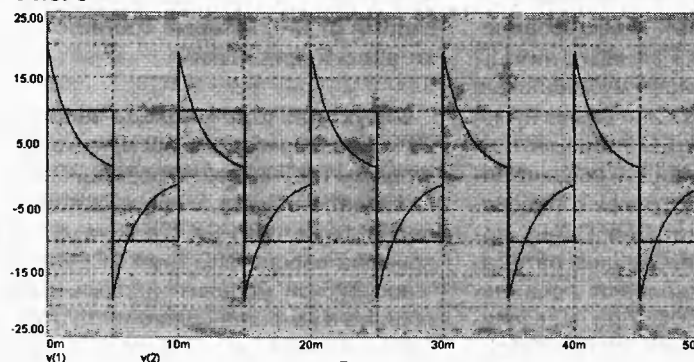


Рис. 5



видно, что нижняя граничная частота составляет примерно 83 Гц, и при этом фаза поворачивается на 45 градусов.

Вот теперь — то самое, интересное. Если через такую цепочку пропустить синусоидальный сигнал частотой, допустим, 100 Гц, то он немного ослабится по амплитуде, изменит фазу на 45 градусов, что, в принципе, здесь эквивалентно задержке на определенное время, но пройдет на выход и останется все тем же синусоидальным. Наше ухо, в принципе, разницы может даже и не заметить. Но что произойдет, если пропустить меандр? Картина разительно меняется. Меандр, как известно, содержит большой набор гармоник (1-ю, 3-ю, 5-ю, 7-ю и т.д.), и все они внесут свой вклад в сигнал, который мы получим на выходе, поразному усилятся и повернутся по фазе на разные углы. В итоге, на выходе гармоники сложатся, так сказать, не в той последовательности, в которой мы бы хотели. На рис.5 представлены входной и выходной сигналы. Амплитуда сигнала генератора изменяется от -10 В до +10 В, частота — 100 Гц.

Как видно, сигнал-меандр по форме исказился очень даже здорово. Да, перепады напряжений остались такими же, как и были, но форма изменилась. Как уже говорилось в статье В.Федорова, усилитель не должен менять форму сигнала.

"Закон сохранения формы сигнала" — так у него это называлось, т.е. что на усилитель входит, то должно и выходить. Усилитель не должен вносить что-то "свое".

А теперь давайте узнаем, а какие же "меандры" проходят через эту цепь без искажений, начиная с какой частоты.

Об искажениях меандра говорить сложно, для этого надо предварительно принять какую-то условную



“меру”, когда будем считать, что меандр остался неискаженным. Если посмотреть на рисунок с АЧХ и ФЧХ (рис.4), то можно увидеть, что искажения по фазе отсутствуют начиная с частоты 10 кГц, как это ни странно. На самом деле на этой частоте поворот фазы составляет где-то 0,5 градуса. А если еще учесть, что у большинства устройств АЧХ простирается до 20 кГц, и это верхняя граничная частота, на которой спад составляет -3 дБ, то получается, что линейных участков вообще нет.

А есть ли какое-нибудь оптимальное значение? Я считаю, что да. Если частота сигнала (меандра) в 10 раз больше нижней граничной частоты, то меандр получается уже вполне приемлемым, и искажения его формы не такие уж большие. Приведу пример. На рис.6 изображен сигнал-меандр частотой 1 кГц, пропущенный через все ту же цепь. То есть его частота в 10 раз больше нижней граничной частоты.

Искажения, конечно, есть, они составляют примерно 12...14% — это поворот фазы примерно на 5 градусов. Возможно, это много, но я думаю, что человеческое ухо их уже почти не будет замечать, так как при воспроизведении через АС возникают искажения в динамиках (инерционность). Но вот разницу между формами сигналов 100 Гц и 1000 Гц, пропущенных через одну и ту же цепь, оно заметит.

Вот один пример, возможно, слишком примитивный, но доступный для понимания.

Во время грозы, когда гром гремит в нескольких километрах от вас, вы чувствуете, как до вашего уха начинают доходить сначала высокочастотные составляющие раската, потом среднечастотные, а уж затем появляются низкие частоты, которые проявляются как ударная волна. В момент, когда до вас дойдут высокочастотные составляющие, ваш мозг уже успевает проанализировать, что это за звук, и из какого он источника. Когда придут среднечастотные составляющие, мозг лишь подтвердит правильность выбора и уже будет “знать”, что сейчас подойдут низкочастотные составляющие. Не замечали? Вы даже успеваете повернуть голову в нужном направлении. Поэто-

Рис. 6

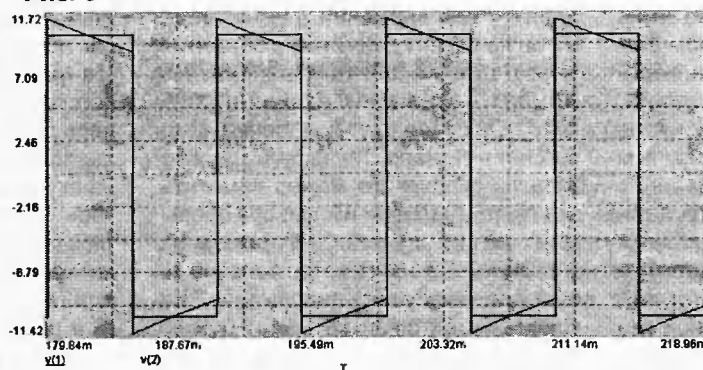
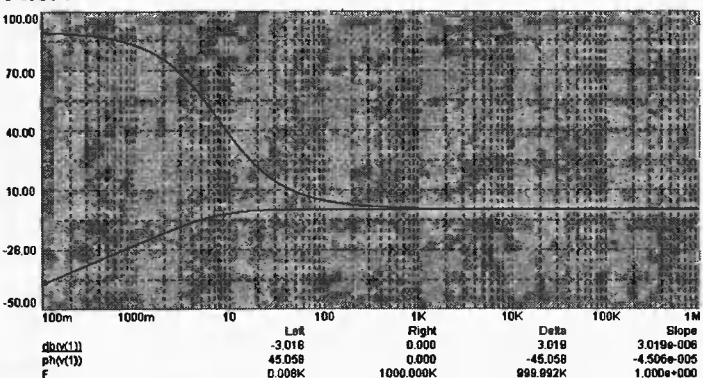


Рис. 7



му, когда приходит ударная волна, вы ее не пугаетесь, так как о ее возникновении вы уже “знаете”, и подготовились. Поэтому люди не очень пугаются грома. Это происходит как раз потому, что, как и в нашей цепи, есть какой-то поворот фазы, какая-то задержка распространения. И скорость прихода разных частот неодинакова. А теперь представьте, что молния ударит метрах в 5 за вашей спиной. Каковы будут ваши ощущения? Я думаю, слово “испуг” будет просто уменьшительно-ласкательным. Вот как раз в этом случае все составляющие звука раската грома придут к вашему уху одновременно, и мозг не успеет их “опознать”, вы просто испугаетесь. То же самое происходит и в нашей цепи.

Как помочь меандру доходить до динамиков без искажений? Конечно же, увеличить емкость конденсатора! В принципе, чем больше, тем, возможно, лучше. Вот здесь я и предлагаю воспользоваться небольшим правилом — выбирайте нижнюю граничную частоту и емкость конденсатора, соответственно, в 10 раз ниже частоты, которую бы вы хотели слышать условно без искажений. Это оптимальное правило. Или в 100 раз ниже, если хотите получить бескомпромиссный звук.

Таким образом, я решил “добратся” до своей звуковой карты в компьютере. Там стояли разделительные

конденсаторы номиналом 470 мкФ, как раз для этого случая и была составлена математическая модель цепи, которую мы только что исследовали. Так что, те, кто имеет компьютер со звуковой картой (а таких, я думаю, большинство), может “посмотреть”, что же там происходит. Посмотреть, какие емкости стоят у вас, если вы, конечно, прослушиваете музыку через пассивные АС, подключенные к выходу “Speaker”. Если у вас стоит усилитель с колонками или активные колонки, то, наверное, стоит просмотреть все цепи.

И не забудьте про разделительные фильтры в колонках. Им тоже стоит уделить внимание.

Так как нижняя граничная частота зависит не только от емкости конденсатора, но и от сопротивления резистора нагрузки (в нашем случае это сопротивление динамика), то лучше будет оперировать значением постоянной времени $\tau=RC$.

Чаще всего приходится изменять именно емкость конденсатора, так как сопротивление АС и входные сопротивления усилителей изменить не удастся.

После описанных исследований я решил доработать свою звуковую плату. Так как она у меня не встроена, а для ISA-слота, и под ней не было других плат, то я просто допаял дополнительные конденсаторы снизу, параллельно “родным”. Поставить их вместо “родных” не позволяла их высота. Конденсаторы допаял номиналом 4700 мкФ. Общая емкость, соответственно, стала 5170 мкФ. Подставил эти данные в модель и получил характеристики, показанные на рис.7.

Как видно, АЧХ начинается уже с частоты 8 Гц, неискаженный сигнал начинается с 1 кГц, а, так сказать, условно неискаженный — где-то со 100 Гц. Что, в принципе, уже лучше, чем было ранее. Графики форм сигналов я приводить не буду, так как для частот 10 Гц и 100 Гц они будут такими же, как и для частот 100 Гц и 1000 Гц в предыдущем случае.

А теперь попытаюсь описать, что же изменилось в звуке.

В принципе, я думаю, и так понятно, что “количество” НЧ-составля-



щей немного увеличилось, точнее, встало на место. Но и качество звука не заставило себя "ждать". Басы стали собранными, более четкими, перестали "бубнить". Разница ощущалась, даже если к выходу звуковой карты подключался усилитель с килоомным входным сопротивлением.

Хотелось бы привести еще один пример. Для этого снова понадобится график формы сигнала на динамике для нижней граничной частоты (рис. 4).

На графике видно, что входной меандр изменился и стал чем-то напоминать серию треугольных импульсов. Также можно предположить, что динамик из всех сил пытается воспроизвести сигнал, который на него поступает, а поступает совсем не меандр. Диффузор динамика старается воспроизвести все изменения формы сигнала. А так как диффузор динамика — вещь инерционная, то он следует за сигналом не точно. Представьте, каково приходится динамику, когда его заставляют воспроизвести эти острые пики сигнала. Диффузор еще не успел отклониться в одну сторону, а магнитное поле уже тянет его обратно. Как следствие, в этот момент появляется расхлябанность, динамик предостав-

лен сам себе. И пока он сам не остановится при движении в одну сторону, он не начнет движение в обратную. И вот здесь возникает неприятный момент — динамик начинает "бубнить". Он совершает колебания, которых нет в сигнале. Также, если посмотреть на график, то можно заметить, что амплитуда этих импульсов превышает амплитуду меандра примерно в 2 раза. Как вы считаете, полезно ли будет динамику перенапряжение? Каково приходится его катушке? Думаю, что хорошего мало. К тому же, при таком перенапряжении диффузор динамика пытается отклониться в 2 раза дальше от центрального положения, при этом он выходит из линейного режима работы. Причем, как из линейного (однородного) магнитного поля в зазоре катушки, так и из линейного диапазона перемещений диффузора. Все это, конечно, справедливо, если на динамик подается мощность, близкая к его номинальной мощности. Динамик начинает компрессировать звук, занижать амплитуду. В итоге вы все это слышите в звуке. Как считаете, вы достойны такого звука?

Если сомнения остались, приведу еще один пример. Разделительный

конденсатор у вас — 470 мкФ. Вы слушаете рок, любите такой инструмент как гитара. Естественный звук гитары (синусоидальной формы) в музыкальных композициях можно услышать теперь достаточно редко, разве что барды воспроизводят его в чистом виде на акустических гитарах. Сигнал с гитары, как правило, пропускают через "искажитель", или, как его официально называют, процессор эффектов, или гитарный процессор. После этого в сигнале появляются гармоники, сигнал на выходе становится в какой-то степени прямоугольным и чем-то напоминает тот самый меандр. Частоты звучания гитар, а особенно бас-гитар, в основном, лежат в диапазоне 100 Гц...1 кГц и выше. А как искажаются сигналы этого диапазона частот при подходе к динамику, вы уже знаете. Так что задумайтесь! Возможно, вы сможете различить, на нейлоновых или металлических струнах играет исполнитель! И слышать все оттенки его игры!

Удачи!

Литература

В. Федоров. Оценка качества УЗЧ. — Радиомир, 2002, №6, С.3.

Г. ПОЛОЗНЕВ,
г. Бердск.

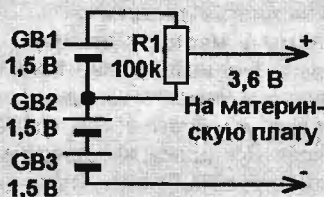
Элементы 316 вместо "больного" аккумулятора

В настоящее время происходит постепенный выход из строя аккумуляторов (синих "бочоночков") на материнских платах сохранившихся еще кое-где компьютеров семейств 286 и 386. Заменить их практически невозможно, так как аккумуляторы такого типа уже давно не выпускаются. В статье [1] предлагается в помощь аккумулятору подсоединить отдельный блок питания, постоянно подключенный к сети. Я попытался найти решение, не требующее такого подключения.

Первоначально предполагалось к батарее напряжением 4,5 В параллельно подключить высокоомный (номиналом несколько мегаом) делитель напряжения, и с него снимать нужные 3,6 В. Однако такая схема оказалась непрактичной из-за трудностей измерения напряжения в высокоомных цепях. Более удобна схема, приве-

денная на рисунке.

К верхнему по схеме элементу GB1 параллельно подсоединен переменный резистор. С движка резистора снимается "вольдобавка" и суммируется с напряжением двух других элементов. На выходе устройства можно получить напряжение от 3 до 4,5 В.



Элементы R6 (316) размещены внутри системного блока в пластиковом пенале. Резистор установлен на плате размером 40x40 мм. Плата закреплена внутри блока винтами, креплениями заглушки задней стенки блока. Резистор снабжен простейшей шкалой и указателем. На плату вы-

ведены контакты для проверки напряжения. Устройство подсоединено к материнской плате вместо удаленного аккумулятора.

На корпусе аккумулятора указана его емкость — 60 мА·час. Так как срок его службы — три-четыре года, можно прикинуть, что микросхемы компьютера потребляют от аккумулятора (в нашем случае — от гальванических элементов) ток порядка 2 мкА. Верхний (по схеме) элемент GB1 разряжается через резистор током порядка 15 мкА. Ожидается, что срок его службы будет не менее полугода. Описанное устройство работает в компьютере модели 386 уже в течение месяца.

Литература

1. С. Дубовой. Блок питания для "больного" аккумулятора. — Радиомир. Ваш компьютер, 2002, №1, С.36.

Кое-что о Windows 9x, и не только

(Окончание. Начало в №4-5/2003)

Е. ЗАРЕЦКИЙ,
г. Чаусы, Могилевской обл.

Для получения доступа к некоторым настройкам видеокарты Riva TNT/TNT2 (управлению Vsync, включению мультитекстурирования в OpenGL) нужно ввести в реестр следующие строки (выполнить данный рег-файл):

```
REGEDIT4
[HKKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\Class\Display\
0000\NVIDIA\Direct3D]
"LODBIASADJUST"=dword:000000fc
[HKKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA
Corporation\Global\NVtweak]
"Coolbits"=dword:00000003
[HKKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA
Corporation\Global\NVtweak]
"PowerUser"=dword:00000001
[HKKEY_LOCAL_MACHINE\SOFTWARE\NVIDIA Corporation\Global\OpenGL]
"ForceMultiTexture"=dword:00000001
*****
```

Чтобы поместить свою информацию на закладку General пункта System Properties панели управления, нужно проделать следующее:

1. Подготовить картинку размером 172x117 пикселей в формате bmp (поддерживается RLE-компрессия, лучше сохранять в 8-битном формате без дизеринга).

2. Сохранить картинку в папке %SystemRoot%\System под именем "oemlogo.bmp".

3. В той же папке создать файл "oeminfo.ini" следующего содержания (все строки без кавычек):

```
[General]
Manufacturer=Я, любимый, сам собрал
Model=Super Rulez Computer
SupportURL=http://www.microsoft.com
LocalFile=%windir%\web\support.htm
[OEMSpecific]
SubModel=необязательная строка
SerialNo=необязательная строка
OEM1=необязательная строка
OEM2=необязательная строка
[ICW]
Product=Computer
[Support Information]
Line1=эти строки будут показаны в окне,
Line2=появляющемся при нажатии на
Line3=кнопку "Поддержка" ("Support")
```

Перезагружать систему не нужно, изменения вступают в силу при первом вызове свойств системы.

В контекстное меню для папок можно добавить команду "Сеанс MS-DOS", которая будет вызывать окно MS-DOS в текущей папке. Для этого нужно создать рег-файл, содержащий следующие строки:

```
REGEDIT4
[HKKEY_CLASSES_ROOT\Folder\Shell\DosPrompt]
@="&Сеанс MS-DOS"
[HKKEY_CLASSES_ROOT\Folder\Shell\DosPrompt\Command]
@="c:\windows\command.com /k cd %1"
```

Аналогично можно поступить и для открытия папки в FAR'е:

```
REGEDIT4
[HKKEY_CLASSES_ROOT\Folder\Shell\FarOpen]
@="&Открыть FAR-ом"
[HKKEY_CLASSES_ROOT\Folder\Shell\FarOpen\Command]
@="c:\progra~1\far\far.exe %1"
*****
```

В командной строке вместо "del *.*" можно использовать более короткий вариант — "del..".

Если удерживать клавишу [Shift] во время щелчка по ссылке в Internet Explorer, то она откроется в новом окне.

Чтобы при наведении курсора мыши на часы в "трее", перед датой отображался день недели, необходимо изменить формат даты. Для этого пройти по пути "Панель управления\Язык

и стандарты\Дата" и в графе "Полное представление даты" вписать "дддд, д ММММ гггг гггг".

Если адрес сайта, на который вы хотите попасть (в Internet Explorer), начинается на "WWW" и заканчивается на ".com", то будет достаточно набрать только среднюю часть адреса и одновременно нажать [Ctrl]+[Enter]. Т.е. вместо "http://www.super_puper.com" можно набрать "super_puper".

Чтобы увидеть содержимое открытого в "Блокноте" файла в DOS-кодировке, необходимо выделить весь текст и выбрать в меню "Формат" команду "Шрифт". В появившемся окне — изменить шрифт на "Terminal" и нажать ОК.

В Internet Explorer все-таки есть возможность "докачки" файлов по протоколам HTTP и FTP. Если при "сливании" какого-либо файла связь оборвалась, восстановите соединение и дайте команду на сохранение файла в тот же каталог. Загрузка данных начнется с места обрыва.

Можно отменить некоторые действия, произведенные в Windows (копирование, перенос, удаление и др.), нажав одновременно [Ctrl]+[Z].

Если в первой строке текстового файла написать ".LOG", то всегда при его открытии в "Блокноте" курсор будет устанавливаться в конце файла, а перед ним будет автоматически вводиться текущее время и дата (это бывает удобно для ведения каких-либо дневников или журналов).

В "Проводнике" можно переименовать сразу группу файлов. Для этого необходимо выделить нужные файлы, щелкнув правой кнопкой мыши по одному из них, и затем переименовать. Все выделенные файлы получают то же имя, но с индивидуальными номерами в скобках.

Для проверки качества записи файлов на диске воспользуйтесь командой "a:*. * nul/s/e/h/r". Этим же способом можно проверить на "читабельность" какой-нибудь заезженный компакт-диск.

Чтобы просмотреть содержимое диска или просто некоторых папок в виде удобного каскадного меню (типа меню "Пуск"), щелкните по панели задач правой кнопкой мыши, перейдите в пункт "Панели инструментов" и выставьте флажок напротив элемента "Рабочий стол". Смотрите, что получилось.

В Windows XP есть встроенная поддержка IE шестой версии. Для активизации нужно набрать в командной строке "iev6 install".

Windows XP считает архивы zip-папками. Но это не всегда бывает удобно, да и добавляет лишние "тормоза". Есть возможность отключить сей "наворот". Достаточно в командной строке набрать "regsvr32/u zipfldr.dll". Все можно вернуть, набрав "regsvr32 zipfldr.dll".

Для анализа сетевого графика (в локальной сети или Internet) в Windows XP можно использовать индикатор работы сети. Активизировать его можно через свойства "Сетевого окружения" — здесь нужно выбрать уже свойства вашего dialup- или сетевого соединения, и в появившемся окне поставить флажок напротив пункта "При подключении вывести значок в области уведомлений".

Для получения различной полезной информации о системе (Windows XP), в командной строке наберите systeminfo.



Е.ИЛЯСОВ,
412302, г.Балашов,
ул.Красина, д.82.

Конвертирование графики в iS-DOS на Sinclair

На страницах любительских компьютерных изданий не раз встречались публикации на тему конвертирования файлов графических изображений из стандартного синклеровского формата в файлы графического формата BMP, широко используемого на компьютерных платформах IBM PC [1...4]. Все перечисленные статьи основаны на более ранней публикации [5], содержащей ассемблерный листинг исполняемого блока машинных кодов, который, собственно, и осуществляет такое конвертирование.

Особенностью этого "популярного" машинно-кодowego фрагмента является то, что он предназначен, в основном, для работы программы-конвертора в среде дисковой операционной системы TR-DOS. Но при многих своих положительных качествах, обеспечивших широчайшее распространение системы TR-DOS на просторах СНГ, она не является оптимальной для деловых, профессиональных (хотя бы полупрофессиональных) применений. Для успешного решения задач, относящихся к подобным категориям, требуются как легко доступное для использования непрерывное операционное пространство значительного объема, так и массивированные файловые операции над структурированными и сортированными данными. Всем этим и многим другим требованиям в гораздо большей степени удовлетворяет дисковая операционная система iS-DOS (IskraSoft, г. С.-Петербург, 1992...2000).

Не будет преувеличением сказать, что при регулярной работе с графическими, текстовыми и иными архивами (наборами данных) преимущества, предоставляемые и обеспечиваемые системой iS-DOS, становятся более чем очевидными. Это относится, в частности, к подготовке полиграфических вариантов каких-либо документов средней степени сложности, где распечатка документа в окончательном виде должна выполняться с хорошим качеством на высококлассных печатающих устройствах под управлением компьютеров офисных платформ, чаще всего — IBM PC [4]. Если учесть, что графические файлы-иллюстрации в формате BMP в таких случаях приходится "заготавливать" в системе TR-DOS, а уже затем переносить и хранить в iS-DOS, то становится наглядной необходимость иметь инструмент для конвертирования графики из фор-

мата SCR в BMP, работающий непосредственно в среде iS-DOS и использующий все предоставляемые для этого системой возможности.

За основу для создания прикладной утилиты scr_bmp.com была взята все та же процедура, описанная в [5] и творчески переработанная с учетом значительных отличий в файловой структуре и в самой концепции системы iS-DOS. Следует особо отметить, что поскольку приведенная версия утилиты проектировалась не как законченный программный продукт, а как примитивный инструмент для решения узкого, очень ограниченного круга задач, то в ней отсутствуют даже намеки на какой бы то ни было сервис, настройки, инструкции, копирайты, условия распространения и другие непереносимые атрибуты, сопутствующие понятию программного продукта. Более того, даже стандартные возможности iS-DOS здесь не задействованы полностью. Например, при создании bmp-файла в него не заносится системная дата и не подсчитывается контрольная сумма для возможных последующих проверок целостности информации. Все, что "умеет" делать эта утилита — это, будучи запущенной одним из стандартных для iS-DOS способов (по какой-нибудь "горячей" клавише, из командной строки монитора или из пользовательского меню), "взять" из-под курсора файл, и если он имеет длину 6912 байтов и расширение имени scr, то записать 16-цветный конвертированный экран в файл длиной 24694 байта с расширением bmp в текущий каталог, пользуясь при этом системным обработчиком ошибок. Таким образом, несмотря на свою "полуфабрикатность", этот программный макет является вполне работоспособным и "дуракоустойчивым".

Публикацию такой "сырой" и неоптимальной программы при желании можно рассматривать как своеобразную "провокацию", призванную всколыхнуть пользовательскую (желательно, от слова "польза") читающую и программирующую массу и дать повод если уж не самостоятельно "довести до ума" конвертор графики, так хотя бы обозначить и выразить свое отношение к iS-DOS'ным вопросам и проблемам (почтовый адрес — в начале статьи). Самые интересные и дельные отклики могут войти в полную версию информационного сборника "IS-files", посвящен-

ного обмену опытом между пользователями системы iS-DOS, выход первого выпуска которого планируется в течение текущего, 2003 года.

Итак, исходный текст утилиты scr_bmp.com в формате iS-DOS Assembler v2.3.:

```

BUF S      EQU 16384
;scr=>bmp by P.A.A.
;ver 0.1 alpha, 16.02.02
;mode - cursor
          ORG 24000
;ini printline
          LD A,0
          LD C,#0B
          RST 16
          LD HL,0
          LD C,#0C
          RST 16
          RET C
BL1       LD B,40
          PUSH BC
          LD A,32
          LD C,#0A
          RST 16
          RET C
          POP BC
          DJNZ BL1
          LD A,0
          LD C,#0B
          RST 16
          LD HL,0
          LD C,#0C
          RST 16
          RET C
;test memory
          LD C,$g_cnfg
          RST 16
          EXX
          LD BC,5
          ADD HL,BC
          LD E,(HL)
          INC HL
          LD D,(HL)
          PUSH DE
          EXX
          POP HL
          AND A
          LD DE,32000
          SBC HL,DE
          LD A,1
          JP C,ERROR
;take cursor
          LD C,$g_curs
          RST 16
          LD E,A
;test filename
;take name
          LD C,$gname
          RST 16
          RET C
          EXX
          PUSH HL
          EXX
          POP IX
;test for lenght
          LD A,(IX+14)
          LD DE,6912
          CP E
          JR Z,G1
          LD A,2
          JP ERROR
G1        LD A,(IX+15)

```



```
CP D
JR Z,G2
LD A,2
JP ERROR

;OK
; "Please wait"
G2
; LD C,$mwait
; RST 16
;open source file
PUSH IX
POP HL
LD C,$fopen
RST 16
RET C

;LDIR fname
EXX
PUSH HL
EXX
POP HL
LD DE,DNAME
LD BC,19
LDIR

;load s_file in screen
LD HL,BUF_S
LD DE,0
LD B,27
LD C,$rifle
RST 16
RET C

;close source file
LD C,$putf
RST 16
RET C

;make dest-file
LD IX,DNAME
LD (IX+8),"b"
LD (IX+9),"m"
LD (IX+10),"p"
LD (IX+11),%00000001
LD (IX+12),0
LD (IX+13),0
LD (IX+14),#76
LD (IX+15),#60
LD (IX+16),0
LD HL,DNAME
LD C,$scrfil
RST 16
RET C

;translate
;save header
LD A,0
LD HL,0
LD IX,HEADER
LD DE,120
LD C,$wpart
RST 16
RET C

LD A,192
LD (YS),A
LD HL,BUF_S
LD (SCADR),HL
LD HL,24566
LD (VECTOR),HL
MAIN LD HL,BUF_D
LD (LINE),HL
LD B,32
CYCLE LD HL,(SCADR)
LD C,(HL)
LD A,H
SRL A
SRL A
SRL A
OR 88
LD H,A
LD A,(HL)
LD (ATR),A
LD D,4
SLA C
PUSH DE
```

```
CALL PSET
POP DE
LD E,A
SLA E
SLA E
SLA E
SLA C
PUSH DE
CALL PSET
POP DE
OR E
LD HL,(LINE)
LD (HL),A
INC HL
LD (LINE),HL
DEC D
JR NZ,LOOP
LD HL,(SCADR)
INC HL
LD (SCADR),HL
DJNZ CYCLE

LD HL,(LINE)
LD DE,256
OR A
SBC HL,DE
LD (LINE),HL
LD HL,(SCADR)
LD DE,32
OR A
SBC HL,DE
INC H
LD A,H
AND 7
JR NZ,DCONT
LD A,L
ADD A,32
LD L,A
JR C,DCONT
LD A,H
SUB 8
LD H,A

DCONT
;save destination... (256b)
PUSH AF
PUSH BC
PUSH HL
PUSH DE
LD A,0
LD HL,(VECTOR)
LD IX,BUF_D
LD DE,128
LD C,$wpart
RST 16
LD A,0
JP C,ERROR
LD HL,(VECTOR)
LD DE,128
OR A
SBC HL,DE
LD (VECTOR),HL
POP DE
POP HL
POP BC
POP AF
LD (SCADR),HL
LD A,(YS)
DEC A
LD (YS),A
JP NZ,MAIN

;CLOSE
LD C,$putf
RST 16
RET C
LD C,$flush
RST 16
XOR A
LD A,#F4
RET

;ERROR
```

```
;0 - ВНУТРЕННЯЯ ОШИБКА
;1 - no memory
;2 - it isn't right source file
ERROR
;ini proc. print
CP 0
JR Z,ER0
CP 1
JR Z,ER1
CP 2
JR Z,ER2
XOR A
LD A,0
ER0 LD HL,ERM0
JR ER_PRT
ER1 LD HL,ERM1
JR ER_PRT
ER2 LD HL,ERM2
JR ER_PRT
ER_PRT LD C,$str
RST 16
XOR A
LD A,0
RET

PSET JR NC,PAPER
LD A,(ATR)
LD E,A
AND 7
LD D,A
LD A,E
AND 64
RRCA
RRCA
RRCA
OR D
LD H,0
LD L,A
LD DE,TABLE
ADD HL,DE
LD A,(HL)
RET

PAPER LD A,(ATR)
AND 120
RRCA
RRCA
RRCA
LD H,0
LD L,A
LD DE,TABLE
ADD HL,DE
LD A,(HL)
RET

;destination file
DNAME DEFW "12345678"
DEFW "123"
DEFB 0;FSTAT
DEFW 0;FLSA
DEFW 0;FLENG
DEFB 0
DEFS 3

;LIST OF ERROR MESSAGES
ERM0 DEFW "Ошибка программы"
DEFB 13
ERM1 DEFW "Не хватает памяти"
DEFB 13
ERM2 DEFW "Исходный файл неверного формата"
DEFB 13

LINE DEFW 0
SCADR DEFW 0
YS DEFB 0
ATR DEFB 0
TABLE DEFW 0,4,1,5,2,6,3,7,0
DEFB 12,9,14,10,14,11,15

VECTOR DEFW 0
;128
BUF_D DEFS 8
```



```

HEADER  DEFW #4D42, #6076, #0000, #0000
        DEFW #0000, #0076, #0000, #0028
        DEFW #0000, #0100, #0000, #00C0
        DEFW #0000, #0001, #0004, #0000
        DEFW #0000, #6000, #0000, #0ECE
        DEFW #0000, #0EC4, #0000, #0000
        DEFW #0000, #0000, #0000, #0000
        DEFW #0000, #0000, #0080, #8000
        DEFW #0000, #8000, #0080, #0080
        DEFW #0000, #0080, #0080, #8080
        DEFW #0000, #C0C0, #00C0, #8080
        DEFW #0080, #0000, #00FF, #FF00
        DEFW #0000, #FF00, #00FF, #00FF
        DEFW #0000, #00FF, #00FF, #FFFF
        DEFW #0000, #FFFF, #00FF, #0000
;BUF_S  DEFS 1

```

Для пользователей iS-DOS, мало знакомых с Ассемблером, приводится дамп — шестнадцатеричная распечатка машинного кода программы. Для того чтобы получить с его помощью работоспособную программу, следует очень аккуратно выполнить несложную последовательность действий. Сначала любым способом требуется создать файл с любым содержанием и с любым именем длиной 705 байтов, а затем заполнить его значениями из прилагаемого дампа, например, программой `demon.com`. После этого нужно будет сосчитать контрольную сумму полученного файла (это можно сделать утилитой `calc.com` или `ch+2.com`). Теперь останется только нанести завершающий штрих, переименовав с помощью системной утилиты `rename.com` почти готовую программу в `scr_bmp.com` и установив "правильную" дату полученного конвертора:

16.02.02.

LSA:24000 (#5DC0), Size: 705 bytes

```

#5DC0 3E 00 0E 0B D7 21 00 00
#5DC8 0E 0C D7 D8 06 28 C5 3E
#5DD0 20 0E 0A D7 D8 C1 10 F6
#5DD8 3E 00 0E 0B D7 21 00 00
#5DE0 0E 0C D7 D8 0E 10 D7 D9
#5DE8 01 05 00 09 5E 23 56 D5
#5DF0 D9 E1 A7 11 00 7D ED 52
#5DF8 3E 01 DA 43 5F 0E 8A D7

```

```

#5E00 5F 0E 27 D7 D8 D9 E5 D9
#5E08 DD E1 DD 7E 0E 11 00 1B
#5E10 BB 28 05 3E 02 C3 43 5F
#5E18 DD 7E 0F BA 28 05 3E 02
#5E20 C3 43 5F DD E5 E1 0E 25
#5E28 D7 D8 D9 E5 D9 E1 11 92
#5E30 5F 01 13 00 ED B0 21 00
#5E38 40 11 00 00 06 1B 0E 2B

```

```

#5E40 D7 D8 0E 28 D7 D8 DD 21
#5E48 92 5F DD 36 08 62 DD 36
#5E50 09 6D DD 36 0A 70 DD 36
#5E58 0B 01 DD 36 0C 00 DD 36
#5E60 0D 00 DD 36 0E 76 DD 36
#5E68 0F 60 DD 36 10 00 21 92
#5E70 5F 0E 23 D7 D8 3E 00 21
#5E78 00 00 DD 21 09 60 11 78

```

```

#5E80 00 0E 2A D7 D8 3E C0 32
#5E88 ED 5F 21 00 40 22 EB 5F
#5E90 21 F6 5F 22 FF 5F 21 01
#5E98 60 22 E9 5F 06 20 2A EB
#5EA0 5F 4E 7C CB 3F CB 3F CB
#5EA8 3F F6 58 67 7E 32 EE 5F
#5EB0 16 04 CB 21 D5 CD 68 5F
#5EB8 D1 5F CB 23 CB 23 CB 23

```

```

#5EC0 CB 23 CB 21 D5 CD 68 5F
#5EC8 D1 B3 2A E9 5F 77 23 22
#5ED0 E9 5F 15 20 DD 2A EB 5F
#5ED8 23 22 EB 5F 10 C0 2A E9
#5EE0 5F 11 00 01 B7 ED 52 22
#5EE8 E9 5F 2A EB 5F 11 20 00
#5EF0 B7 ED 52 24 7C E6 07 20
#5EF8 0A 7D C6 20 6F 38 04 7C

```

```

#5F00 D6 08 67 F5 C5 E5 D5 3E
#5F08 00 2A FF 5F DD 21 01 60
#5F10 11 80 00 0E 2A D7 3E 00
#5F18 DA 43 5F 2A FF 5F 11 80
#5F20 00 B7 ED 52 22 FF 5F D1
#5F28 E1 C1 F1 22 EB 5F 3A ED
#5F30 5F 3D 32 ED 5F C2 96 5E
#5F38 0E 28 D7 D8 0E 02 D7 AF

```

```

#5F40 3E F4 C9 FE 00 28 0B FE
#5F48 01 28 0C FE 02 28 0D AF
#5F50 3E 00 21 A6 5F 18 0A 21
#5F58 B7 5F 18 05 21 C9 5F 18
#5F60 00 0E 6C D7 AF 3E 00 C9
#5F68 30 17 3A EE 5F 5F E6 07
#5F70 57 7B E6 40 0F 0F 0F B2
#5F78 26 00 6F 11 EF 5F 19 7E

```

```

#5F80 C9 3A EE 5F E6 78 0F 0F
#5F88 0F 26 00 6F 11 EF 5F 19
#5F90 7E C9 31 32 33 34 35 36

```

```

#5F98 37 38 31 32 33 00 00 00
#5FA0 00 00 00 00 00 00 8E E8
#5FA8 A8 A1 AA A0 20 AF E0 AE
#5FB0 A3 E0 A0 AC AC EB 0D 8D
#5FB8 A5 20 E5 A2 A0 E2 A0 A5

```

```

#5FC0 E2 20 AF A0 AC EF E2 A8
#5FC8 0D 88 E1 E5 AE A4 AD EB
#5FD0 A9 20 E4 A0 A9 AB 20 AD
#5FD8 A5 A2 A5 E0 AD AE A3 AE
#5FE0 20 E4 AE E0 AC A0 E2 A0
#5FE8 0D 00 00 00 00 00 00 00
#5FF0 04 01 05 02 06 03 07 00
#5FF8 0C 09 0E 0A 0E 0B 0F 00

```

```

#6000 00 00 00 00 00 00 00 00
#6008 00 42 4D 76 60 00 00 00
#6010 00 00 00 76 00 00 00 28
#6018 00 00 00 00 01 00 00 C0
#6020 00 00 00 01 00 04 00 00
#6028 00 00 00 00 60 00 00 CE
#6030 0E 00 00 C4 0E 00 00 00
#6038 00 00 00 00 00 00 00 00

```

```

#6040 00 00 00 00 00 80 00 00
#6048 80 00 00 00 80 80 00 80
#6050 00 00 00 80 00 80 00 80
#6058 80 00 00 C0 C0 C0 00 80
#6060 80 80 00 00 00 FF 00 80
#6068 FF 00 00 00 FF FF 00 FF
#6070 00 00 00 FF 00 FF 00 FF
#6078 FF 00 00 FF FF FF 00 00

```

#6080 00

Литература

1. Зарецкий Е. Конвертация ZX-картинки в формат BMP. — Радиолюбитель. Ваш компьютер, 2000, №12, С.35.
2. Рошин И. Конвертация ZX-картинки в формат BMP. — Радиолюбитель. Ваш компьютер, 2001, №5, С.39.
3. Ильясов Е. Конвертирование графических файлов на Spectrum. — Радиомир. Ваш компьютер, 2001, №8, С.35.
4. Ильясов Е. Полиграфия на Sinclair. — Радиолюбитель, 2002, №4, С.22.
5. Max & Alex from "CYBERAX Soft". Конверсия цветной спектральной картинки на IBM. — Электронный журнал "ZX-Ревю", №11-12, Москва, 1997 г.

И.РОШИН,

г.Москва,

Fido: 2:5020/689.53

ZXNet: 500:95/462.53

E-mail: bestview@mtu-net.ru

WWW: http://www.ivr.da.ru

Процедура сравнения строк на ассемблере Z80

При написании программы BestView мне понадобилась процедура сравнения двух строк со следующими входными и выходными параметрами: в HL задается адрес первой строки, в DE — адрес второй строки, в BC — длина строки (не равная 0); на выходе флаг Z установлен, если строки совпали, и сброшен в противном случае.

Алгоритм сравнения строк очень простой. Берем байт по адресу, указанному в DE, сравниваем его с байтом по адресу, указанному в HL. В случае не-

равенства — выходим из процедуры (флаг Z при этом сброшен), в случае равенства — увеличиваем HL и DE на 1 (таким образом, они будут теперь указывать на следующие байты сравниваемых строк) и уменьшаем BC на 1 (т.е. уменьшаем счетчик). Если после уменьшения BC < 0, переходим к сравнению следующих байтов, иначе (BC = 0) — строки совпали, тогда выходим из процедуры (флаг Z при этом установлен).

Соответственно, я написал такую процедуру, которая использовалась в

BestView до версии 2.14 включительно:

```

COMP_ST LDA, (DE)
        CP      (HL)
        RET     NZ      ; Не совпали.

        INC     HL      ; !
        INC     DE      ; !
        DEC     BC      ; !
        LD      A, B    ; !
        OR      C        ; !
        JR      NZ, COMP_ST

```

RET ; Совпали.

И вот недавно, просматривая ее, я обратил внимание вот на что: увеличе-



ние HL и DE на 1, уменьшение BC на 1, проверка BC на равенство нулю — да ведь в системе команд процессора Z80 есть команда, которая делает все это!

Команда LDI копирует байт из (HL) в (DE), увеличивает HL и DE на 1, уменьшает BC на 1, сбрасывает флаг P/V, если после уменьшения BC=0, и устанавливает флаг P/V в противном случае. То есть эта команда делает практически то же, что и помеченный "!" фрагмент процедуры. Только, во-первых, она еще копирует байт из (HL) в (DE), и во-вторых, помещает информацию о равенстве BC нулю во флаг P/V, а не во флаг Z.

Копирование байта из (HL) в (DE) в рассматриваемой процедуре совершенно ни к чему. Однако, приглядевшись внимательнее, можно заметить, что после выполнения команды RET NZ, байт по адресу, указанному в HL, заведомо равен байту по адресу, указанному в DE (действительно, если бы они не были равны, то после их сравнения произошел бы выход из процедуры).

А если байты равны, значит, копирование из (HL) в (DE) не изменит (DE), т.е. окажется безвредным побочным эффектом. Тогда весь участок процедуры, помеченный "!", можно заменить на одну команду LDI, одновременно заменив команду условного перехода к началу цикла (JR NZ, COMP_ST) на команду JP PE, COMP_ST — ведь теперь именно флаг P/V указывает на выполнение или невыполнение условия BC=0.

Но остается еще одна проблема — при выходе из процедуры в случае совпадения строк флаг Z должен быть установлен. И если в первоначальном варианте процедуры он автоматически оказывался установлен при проверке условия выхода из цикла (BC=0), то теперь проверка этого условия вообще не связана с флагом Z.

Сначала я хотел добавить перед выходом из процедуры в случае совпадения строк (т.е. перед последней командой RET) команду XOR A, устанавливающую флаг Z. Но, посмотрев в справоч-

ник, я увидел, что команда LDI не изменяет флаг Z, а посмотрев на процедуру, увидел, что после сравнения очередных байтов строк флаг Z заведомо будет установлен (ведь строки совпадают!). Таким образом, ничего специально добавлять не потребовалось.

Итак, оптимизированный вариант процедуры:

```
COMP_ST LD A, (DE)
CP (HL)
RET NZ ; Не совпали.

LDI ; На Z не влияет!
JP PE, COMP_ST

RET ; Совпали, Z=1!
```

Длина процедуры уменьшилась на 2 байта, а время выполнения сократилось следующим образом — в случае, когда сравниваемые строки не совпадают (причем их первые n байтов совпадают), выигрыш составляет $12n$ тактов, а в случае, когда строки совпадают (n — их длина), выигрыш составляет $12(n-1)+7$ тактов.

Какие бывают

ZX

Я.ОЧАКОВСКИЙ,
Алтайский край,
р.ц. Петропавловское.

Компаньон-2

Позиционируется как бытовой компьютер, модификации — 2.00, 2.01, 2.02, 2.03.

Производитель — Государственное предприятие "Ижевский механический завод", г.Ижевск. Время появления — конец 1992...начало 1993 г.

"Компаньон-2" — это широко распространенный Spectrum-совместимый компьютер. Все его модификации различаются лишь количеством разъемов для подключения периферии. Но мы остановимся на компьютере "Компаньон-2.01", полностью совместимом на аппаратном уровне с фирменным "ZX-Spectrum". На мой взгляд, он является самым "гибким" из всего семейства "Компаньонов". "Компаньон-2.01" имеет:

- встроенный кодек SECAM, обеспечивающий работу с обычным телевизором;
- параллельный интерфейс Centronics и последовательный интерфейс RS-232;
- интерфейсы Kempston- и Sinclair-джойстиков;
- выход RGB;
- расширенную клавиатуру (88 клавиш, включая клавиши ESC, набор F-клавиш, а также дополнительные функциональные, цифровые и установочные клавиши);
- факультативно установленное проф. ПЗУ (27128, одна микросхема);
- два гнезда для сопряжения компьютера с магнитофоном;
- ЦП — Z80A (аналог отечественного производства);
- интерпретатор языка программиро-

вания Бейсик и встроенный транслятор Ассемблера;

Потребляемая мощность "Компаньона-2.01" — не более 75 Вт.

Плата "Компаньона" имеет малые размеры. Поэтому она расположена в одном корпусе с клавиатурой. Функции, которые ранее использовались в сочетании с клавишами SHIFT, имеют отдельные клавиши — это курсоры G, L, C, D (рус.), E, inv video, true video. Также "Компаньон" "знает" русский алфавит. С помощью клавиатуры имеется возможность регулировать громкость звучания внутреннего динамика.

На мой взгляд, этот компьютер является наиболее совершенным компьютером в классе "ZX-Spectrum 48".

Микрон-3

Позиционируется как учебный компьютер. Поставлялся радиолюбительскими кооперативами и ДОСААФ в виде набора радиоэлементов или готовых плат для самостоятельной сборки.

Время выпуска — 1989...92 гг.

Технические характеристики "Микрона":
- имеет фирменную плату компьютера "ZX-Spectrum" (на печатной плате имеется трассированная надпись: ZX-Spectrum 1989 The Sinclair Research co Ltd (c));

- проф. ПЗУ объемом 32 Кб.;
- ОЗУ объемом 48 Кб.;
- встроенный динамик.

Недостаток этого компьютера в том, что он не имеет разъема интерфейса, а снабжен только минимальным набором разъемов для подключения "желе-

за" — "Монитор", "Kempston-джойстик", "Загр.", "Сохранение" (для магнитофона). Но у "Микрона" есть и плюсы. В первую очередь — это полная программная и аппаратная совместимость с фирменным "ZX-Spectrum".

Sintez-2

Позиционируется как игровой компьютер. Производитель — Производственное объединение "Сигнал", г.Кишинев, 1993 г.

Параметры компьютера "Sintez-2":

- имеет фирменный ЦП — Z80 от ф. "Zilog";
- построен на широко распространенной плате "Terminal";
- объем ОЗУ — 128 Кб.;
- проф. ПЗУ (27128, один корпус);
- стандартная клавиатура;
- параллельный интерфейс "Centronics".

На мой взгляд, "Sintez-2" является прекрасной машиной для развлечений, но, к великому сожалению, он полностью несовместим с фирменным "Spectrum" на аппаратном уровне.

Недавно в мои руки попал компьютер "Toshiba", название которого я поставил под сомнение. Дело в том, что он полностью смонтирован из деталей отечественного производства на плате "Terminal" и является точной пиратской копией "Sintez-2". Все надписи на корпусе оказались обычными переводными наклейками (табуировками, как в жвачке).

Юниор

Позиционируется как учебный компьютер. Разработчик — фирма "ELWRO", Польша, 1990 г.

В состав "Юниора" входят следующие функциональные блоки:

- ЦП — Z80A;
- страничное ОЗУ емкостью 64 Кб.;



- ПЗУ емкостью 24 Кб.

"Юниор" имеет открытую структуру, допускающую расширение дополнительными функциональными блоками (модулями графики высокой разрешающей способности, сложным звукоинтезаторами и т.п.).

Интерпретатор языка Бейсик и программа управления кассетным магнитофоном полностью совместимы с соответствующими программами компьютера "ZX-Spectrum".

Дисплей обеспечивает два стандартных режима вывода символьной информации: 24x32 и 24x64.

Устройство управления позволяет работать с двумя дисковыми на дисках 133 мм объемом 175...800 Кб.

Этот компьютер разрабатывался как учебный, но он неплохо подходит и для других целей.

Микроша

Позиционируется как учебный компьютер. Поставлялся теми же организациями, что и "Микрон-3". Дата выпуска — 1986 г.

Представляет собой одноплатный компьютер с ОЗУ емкостью 32 Кб и ПЗУ емкостью 2 Кб, без каких-либо внешних устройств, но, в принципе, позволяющий подключение любого бытового телевизора и принтера.

Scorpion Z5 256 Turbo

Производитель — фирма "Скорпион", г.Санкт-Петербург, разработка 1992...94 гг. (производился до конца 2001 г.).

Основные технические и программные характеристики компьютера:

- полная программная совместимость с компьютерами "Spectrum 48", "Spectrum 128";

- возможность работы в операционных системах TR-DOS (в ПЗУ), IS-DOS, CP/M. При работе в ОС CP/M поддерживается электронный диск объемом 160 Кб. Допустимо использование дискет от компь-

ютеров "Robotron 1715", "Хоббит", "Prof, AMT-Турбо", "Вектор", "Корвет" и других;

- уникальная встроенная система — теневой сервис-монитор — дает широкие возможности для изменения аппаратных и программных характеристик в любой момент работы компьютера (конвертация программ для сохранения на диске программ, загруженных с магнитофона; распечатка экранных картинок на принтере и запись их в файл; резидентные дисковые утилиты для форматирования и проверки дисков в форматах TR-DOS, CP/M, IS-DOS);

- теневой монитор-отладчик с анализатором на базе форт-системы является великолепным средством для отладки программ. С его помощью программу можно не только посмотреть изнутри, но и изменить по своему вкусу. Полезен при программировании в машинных кодах;

- обеспечивается режим "турбо" — 7 МГц. Включение/выключение турбо-режима производится из меню теневого сервис-монитора. В режиме "турбо" в два раза повышается скорость работы программ и дисководов;

- объем ОЗУ — 256 Кб (микросхема 565PY7), ПЗУ — от 64 Кб (микросхема 27512) до 256 Кб (микросхема 27020), центральный процессор — Z80B (частота 7 МГц). Потребляемый ток — переменный 0,8...0,9 А (5 В), 20 мА (12 В);

- интегрирован звуковой процессор AY-3-8910/12 для формирования трехканального звука;

- "турбированный" контроллер дисковода с цифровой ФАПЧ значительно повышает надежность и скорость считывания информации. Он расположен непосредственно на плате. Микросхема 1818BF93 защищена от выхода из строя по питанию;

- интерфейсы принтера — параллельный (Centronics) и последовательный (RS232C);

- введена поддержка порта атрибутов экрана 0FFh, "оживляющего" неко-

торые программы, не работающие на других вариантах Spectrum-совместимых компьютеров;

- обеспечена привязка к уровню "черного";

- имеется буферизированная клавиатура;

- все подключения к плате сделаны через три разъема: разъем периферии (питание, TV, клавиатура, принтер, джойстики, магнитофон), разъем дисковода (полностью совпадает с разъемом дисководов 5305, 5311, 5313, 5323, Teac и т.п.), системный разъем (полностью совпадает с разъемом фирменного "ZX-Spectrum 48/128");

- встроенный в плату индивидуальный номер обеспечивает гарантированную защиту от компьютерного пиратства. Каждый компьютер имеет метку-сигнатуру, хранимую в ПЗУ и отображаемую на экране при входе в меню теневого сервис-монитора по кнопке "Magic". Этот же номер указан в фирменном паспорте-сертификате и подтвержден круглой печатью фирмы "Скорпион". Дата продажи соответствует дате выпуска ПЗУ.

Для "Scorpion" разработан контроллер SMUC, обеспечивающий подключение к нему жесткого диска, а также программатор УФ ППЗУ, контроллер XT/AT IBM-клавиатуры, плата согласования с IBM (Hayes)-модемами, графический манипулятор Kempston-Mouse, световой пистолет. Все разработки подержаны соответствующим ПО.

К особым разработкам следует отнести плату GMX (Graphics Memory Expander), повышающую разрешение графики с 192x256 до 640x200 и выше, а также музыкальную плату General Sound.

На мой взгляд, "Scorpion" является одним из самых "продвинутых" Spectrum-совместимых компьютеров и имеет все для нормальной работы.

Дела околоСпектрумовские

Компания Amstrad представила новую модель телефона (если можно так сказать) — устройство Em@iler Plus. В одном корпусе находятся телефон, факс, автоответчик, система для работы с электронной почтой, веб-браузер, считыватель смарт-карт и встроенный "ZX-Spectrum"! Для развлечения и работы можно использовать программы с платформы "ZX-Spectrum". Пользователь может (за отдельную плату) скачать игру с сайта провайдера услуг для Em@iler Plus. Стоимость этого агрегата (или чудо-комбайна) всего 100 английских фунтов. Но вот пользоваться таким чудом техники можно только в Великобритании, т.к. только на ее территории действуют услуги системы Amsurf, к которой подключается чудо-телефон.

Все так привыкли к старине Z80, что лишь немногие знают о других процессорах — более совершенных его прототипах, выпускаемых компаниями Zilog. И по сей день эта линейка процессоров весьма популярна.

64180/Z180 построен на архитектуре, совместимой с Z80, что дает возможность использовать все "море" программ, написанных для Z80. Новая система имеет более "продвинутый" набор команд и меньшее энергопотребление. Все внутренние периферийные устройства 64180 для программиста являются портами ввода/вывода, доступен динамический выбор параметров шины, включая количество циклов ожидания памяти и устройств ввода/вывода и скорости регенерации. Процессор имеет два независимых 16-разрядных таймера, каждый из которых может быть запрограммирован для ге-

Е. ЗАРЕЦКИЙ,

г. Чаусы, Могилевской обл.

нерации прерывания. 64180 имеет аналогичный Z80 набор команд, но выполняет их гораздо быстрее, добавлено также несколько новых команд. Процессор адресует значительно больший объем памяти.

Корпорация Sharp сообщила о разработке новой технологии производства дисплеев со встроенной непосредственно в стекло микропроцессорной схемой CG Silicon Technology, что превращает дисплеи в своеобразные "компьютеры на стекле". В качестве наглядной демонстрации был представлен прототип чипа Z80, выполненный с использованием CSG с внедренным в структуру подложки 8-битным процессором, состоящим из 13 тысяч транзисторов, который компания Zilog разработала в 1976 г., а Sharp начала производить в 1977 г.



Inquisition

Из дневника неизвестного автора

Каюсь, сам напросился на этот обзор. Шеф, со странной искоркой в глазах, что делало его чем-то похожим на преподобного отца-инквизитора, передал мне диск, стараясь поаккуратнее выбирать выражения о том, какое впечатление произвела на него эта игра. Подсознание взвыло, ощущая подвох, но быстро заткнулось, так как поддержки у сознания не нашло. Уже через несколько часов диск стоял в приводе компьютера. Так начались хроники одного обзора...

День первый

Инсталляция, и та пыталась пойти кривь и вкось, синий экран смерти стал настолько привычен, что я уже, грешным делом, стал подумывать, а не установить ли мне его как скринсервер. Но я упрямый, умные люди посоветовали переустановить "Окна", драйвера...

День шестой

... а также поменять память, процессор и материнскую плату вместе с видюхой. Подсознание с самодовольной ухмылкой типа "а я вас всех предупреждало" отстраненно наблюдает за процессом, появляется ощущение легкой шизоидности. В сотый раз прочитал описание на диске об увлекательном сюжете и прелестной графике. Помогло, но ненадолго. Поймал себя на мысли, что прикидываю, смогу ли я попасть этим компактом в окно соседа.

День девятый

Она запустилась! Мне стало нехорошо на скриптовой части ролика, дает знать о себе переутомление. Хотя фон и проработан симпатично, но персонажи больше напоминают первые бета-версии Буратин. В их походке есть что-то ужасающее, словно они — плохо склеенные марионетки в неумелых руках. Вид от третьего лица в adventure с

А.ВЕНДИЛОВСКИЙ,
г.Минск.

элементами action — не лучшая находка. Кстати, кто писал скрипты для камереры?! Очень хотелось бы лично познакомиться с ним, желательно, в темном переулке и без свидетелей. Зато управление очень порадовало бы... отцов-инквизиторов, если бы они еще существовали в наши дни. Выхнул пальцы. Подзреваю, что игру, в основном, создавали для игровой приставки — такой интерфейс обычно передается по наследству. Ночью снились костры инквизиции и вопли: "Раскайся, грешник!"

День двенадцатый

Шеф сказал, что после удаления цензурщины из статьи, там остались только системные требования к игре да название статьи, хотя в общем контексте и они выглядят несколько... Порекомендовал больше внимания уделить сюжету. А что тут говорить? Время действия — XIV век. Главный персонаж — разорившийся мелкий дворянин, живущий экспроприацией, циничный и жадный (короче говоря, подался в воры). А тут подоспели и Столетняя война (в те времена "Санта Барбары" еще не было, и люди развлекались как могли), и поветрие Черной Смерти, и добрые священники с кострами из раскаявшихся или не раскаявшихся (что, правда, случилось намного реже, обычно аргументы святых отцов были убедительны) грешников. А кругом — заговоры дьяволопоклонников. Под сводами церкви тоже тлеют заговоры — тень их готова поглотить Европу... И только главный герой, пройдя через все испытания и ловушки, может показать миру, откуда у проблем рога растут... Хотя, на самом деле его интересует лишь одно — сколько он сможет на этом заработать.

День пятнадцатый

Кто сказал, что по геймплею игра напоминает "Thief"? Вы когда-нибудь про-

бавали прокрасться персонажем, которого шатает как пьяного конюха? А боевая система! Бедная "клава" не выдержала — пока выбьешь нужную комбинацию, да и пальцы пришлось в гипс упаковать. Кто вообще придумал этот жуткий гибрид жанров? От action достались тупые схватки (AI тут и не пахло, любопытно, почему любой таракан умнее любого стражника?), а от adventure — утомительные забеги с поиском предметов, что в данном случае напоминает нечто мазохистское. После отчаянных выкрутасов на "клаве", я плюнул на все и, как танк, с голыми руками ломанул на толпу монахов с алебардами. Глядя, как эти толстопузые несчастные погибают под ударами кулаков главного героя, начинаю понимать, что если мой дохляк еще и меч в руки возьмет, то Дункан Маклаут рядом с ним — просто ребенок. Баланс в игре витает в районе абсолютного нуля. Шеф снова на сюжет намекал... Ну как ему объяснить, что сюжет, может быть, и интересный, только понять это трудно — каждый час по чайной ложке, и когда тебя пичкают очередной порцией информации, ты уже благополучно успеваешь забыть предыдущую. Ночью опять снились инквизиторы с испанскими сапожками, дыбой и Железной Девой. Кажется, начинаю привыкать.

День двадцать первый

Я все-таки сделал это! Попал диском в окно соседа! Доктор говорит, что мои дела идут на поправку, хотя я слишком буйный, но они уже к таким привыкли. Встретил несколько своих ранее пропавших коллег. Бедняга, который описывал квесты, по-прежнему пежит в коме, а "железячник" ищет драйвера для своего нового матраса. Приходил шеф, незаметно пытался пощупать у меня пульс, говорит, что у него есть на примете одна работенка. В глаза старался не смотреть. Подсознание взвыло...

Коды, читы... Е.КУРИЛОВИЧ.

Age of Mythology

Во время игры — нажать "Enter" и ввести любой из следующих кодов, а потом еще раз нажать "Enter":

- JUNK FOOD NIGHT — получить 100 единиц еды;
- TROJAN HORSE FOR SALE — получить 100 единиц дерева;
- ATM OF EREBUS Add — получить 1000 единиц золота;
- LAY OF THE LAND — открыть карту;
- UNCERTAINTY AND DOUBT — скрыть карту;
- THRILL OF VICTORY — выиграть сценарий;
- IN DARKEST NIGHT — вызвать ночь;
- MOUNT OLYMPUS — улучшить ситуацию;
- SET ASCENDANT — показать на карте всех животных;
- CONSIDER THE INTERNET — замедлить все юниты;
- PANDORAS BOX — получить новую силу "God Power";
- DIVINE INTERVENTION — "включить" уже использованную силу "God Power";

- GOATUNHEIM — превратить врагов в козчиков;
- FEAR THE FORAGE — получить силу "Walking Berry Bushes Power";
- BAWK BAWK BOOM — получить силу "Chicken Metero God Power";
- I WANT TEH MONKEYS!!!! — получить много обезьян;
- WRATH OF THE GODS — получить катаклизм (землетрясение, упавший метеорит, торнадо и т.п.);
- L33T SUPA H4X0R — строить быстро;
- CHANNEL SURFING — дать юнитам возможность ходить по воде;
- RED TIDE — превратить воду в кровь;
- ISIS HEAR MY PLEA — герои из кампании;
- O CANADA — "Lazer Bear";
- TINES OF POWER — "ForkBoy";
- WUV WOO — "Flying Purple Hippo".

Демиурги (Etherlords)

Вызвать консоль клавишей "~". Находясь в консольном режиме,



напечатать "EtherRevelation", нажать клавишу "Enter" и ввести коды.

Стратегический режим:

- open_fog — убрать туман войны;
- hide_fog — вернуть туман войны;
- lose — проиграть миссию;
- win — выиграть миссию;
- save — быстрое сохранение;
- load — быстрая загрузка игры;
- player — информация об игроках;
- give all — получить по 15 единиц всех ресурсов, кроме эфира;
- view resources — информация о ресурсах.

Боевой режим:

- lose — проиграть битву;
- win — выиграть битву;
- view_hand — заглянуть в руку противника;
- hide_hand — убрать информацию о руке противника;
- swap — поменяться местами с противником;
- view army — получить информацию о своей армии;
- view hand — получить информацию о своей руке;
- view players — получить информацию о сражающихся;
- view spells — вывести список всех заклинаний и существ в виде 4-знаковых кодов;

- add spell "код" — добавить заклинание;
- add creature "код" — добавить существо;
- change health "число" — задать здоровье героя (кроме числа 100).
- change mana "число" — задать количество маны героя (кроме числа 100).

Данные коды полностью действительны только для версии игры от фирмы "1C".

Duke Nukem: Manhattan Project

Нажать клавишу "~" (тильда), чтобы вызвать режим консоли, а затем ввести exes cheats.cfg. Теперь во время игры можно использовать одну из указанных ниже клавиш, чтобы активировать нужный код:

- G — получить все оружие и предметы;
- H — получить все оружие;
- J — получить Jetpack;
- F — получить Force Field;
- L — получить дополнительную жизнь;
- K — самоубийство;
- I — режим Бога;
- P — пауза в стиле "The Matrix".

Heroes of Might and Magic 4: The Gathering Storm

Во время игры нажмите клавишу "Tab" для входа в консольный режим и вводите коды:

- nwcambrosia — получить свободные материалы;
- nwcares — победа;
- nwcachilles — проигрыш;
- nwcathena — получить навык выгоды;
- nwcthoth — повысить уровень;
- nwcragnarok — проигрыш сценария;
- nwcvalhalla — победа в сценарии;
- nwchermes — неограниченное движение;
- nwcspinx — показать пазл-карту;
- nwcselfcetothe gods — получить максимум удачи;
- nwscrap — получить максимум морали;
- nwccityoftroy — все структуры;
- nwcimagod — доступ к меню кодов;
- nwcgosolo — авто-игра;
- nwcprometheus — получить саван;
- nwcettubute — получить кинжал отчаяния;
- nwcsexcalibur — получить кольцо большего отрицания;
- nwcnielungenlied — меч Богов;
- nwcstrisram — участники общественной кампании;
- nwcancelot — чемпионы;
- nwcstmichael — ангелы;
- nwcsevenlittleguys — карлики;
- nwcmerlin — маги;
- nwcronus — титаны;

- nwcblablab — вампиры;
- nwchades — дьяволы;
- nwcunderthebridge — тролли;
- nwckingminos — минотавры;
- nwcxanthus — кошмары;
- nwcxfafnir — черные драконы;
- nwcdoyou smellbrownies — эльфы;
- nwcferriir — волки;
- nwcfixmyshoes — эльфы;
- nwcthelast — единороги;
- nwcvalkyries — маги-людоеды;
- nwcposeidon — морские монстры;
- nwcoldman — старик Джак.

Mafia: City of Lost Angels

Чтобы легко заработать деньги, сделайте в казино "отрицательную" ставку (значение ставки — отрицательное число). Теперь вы будете получать деньги каждый раз, когда проигрываете.

После прохождения игры откроется новый секретный режим — "Free Ride Extreme", в котором появятся новые машины и исчезнет полиция.

Neverwinter Nights

Для получения доступа к кодам в любом текстовом редакторе откройте файл nwn.ini, который находится в директории с игрой, и найдите в нем секцию "Game Options". Добавьте в эту секцию строку Debug Mode=1, сохраните изменения и закройте файл. Теперь начните игру и нажмите клавишу "~". В появившейся консоли напечатайте DebugMode 1. Вам доступны следующие коды:

- dm_god — неуязвимость;
- SetCHA # — изменить уровень параметра Charisma на число #;
- SetSTR # — изменить уровень параметра Strength на число #;
- SetINT # — изменить уровень параметра Intelligenc на число #;
- SetWIS # — изменить уровень параметра Wisdom на число #;
- SetCON # — изменить уровень параметра Constitution на число #;
- dm_givegold # — получить # единиц золота;
- GiveXP # — получить # единиц экспы;
- SetAge # — изменить возраст персонажа на #;
- SetAttackBase # — изменить уровень навыков боя персонажа на #;
- SetAppearance elf — изменить расу персонажа на эльфийскую;
- SetAppearance human — изменить расу персонажа на человеческую.

Если после ввода соответствующего чита появилась надпись "Entered Target Mode" — нужно кликнуть на персонаже, к которому применяется данный код. Если код введен верно, появляется надпись "Success". При изменении своего персонажа, в файл nwnplayer.ini, который находится в папке с игрой, нужно добавить строки:

Single Player Enforce Legal Characters=0
Single Player ItemLevelRestrictions=0.

RollerCoaster Tycoon II

Чтобы воспользоваться кодами, нужно присвоить посетителю парка одно из следующих имен:

- Chris Sawyer — посетитель превратится в фотографа;
- Simon Foster — посетитель превратится в артиста;
- Melanie Warn — повысить уровень счастья посетителей и увеличить рейтинг парка;
- Katie Brayshaw — общая паника;
- Tony Day — посетитель с таким именем будет, как правило, очень голоден;
- John Wardley — с посетителем произойдет что-нибудь неожиданное;
- John Mace — при входе в парк посетитель будет платить двойную цену;
- Damon Hill — увеличить скорость картинга;
- Michael Schumacher — еще больше увеличить скорость картинга;



- Mr Bean — замедлить картинг.

Нескольким посетителям можно присвоить одно и то же имя, для этого нужно написать его буквами в разных регистрах (прописными или строчными). Например: melanie warn, MELANIE WARN, MELANIE warn.

Soldier of Fortune II: Double Helix

Нужно найти в директории игры в папке /base файл myconfig.cfg, открыть его в любом текстовом редакторе и найти в нем строку "toggleconsole". Привяжите эту команду к любой свободной клавише. Например, для привязки к клавише "U" строка должна иметь такой вид: bind U "toggleconsole".

Теперь, после запуска игры, с помощью клавиши "U" (в данном случае), вызывается режим консоли. После этого следует загрузить нужный уровень, используя команды map <имя карты> или devmap <имя карты>.

Названия карт: air1; air2; air3; air4; arm1; arm2; cem1; col1; col10; col2; col3; col4; col6; col7; col8; col9; finca1; finca2; finca3; finca4; hk1; hk2; hk3; hk4; hk5; hk6; hk7; hos1; hos2; hos3; hos4; kam1; kam10; kam11; kam12; kam2; kam3; kam4; kam5; kam6; kam7; kam8; kam9; liner1; liner2; liner3; pra1; pra2; pra3; pra4; pra5; pra6; shop1; shop2; shop3; shop4; shop5; shop6; shop7; shop8; tut1

После загрузки уровня, в консоли вводится команда sv_cheats 1 для активации читов. Теперь доступны нижеприведенные коды:

- god — режим Бога;
- noclip — получить возможность проходить сквозь стены;
- give all — получить все оружие и полный боекомплект;
- notarget — отключить AI противников;
- raven 1 — выбор уровня из главного меню;
- give ammo — получить полный боекомплект для имеющегося оружия;
- give health — восполнить здоровье;
- cmdlist — увидеть список всех консольных команд;
- mapname — увидеть название текущей карты;
- dir maps — увидеть список всех карт;
- quit — выйти из игры.

Star Wars: Galactic Battlegrounds — Clone Campaigns

Во время игры — нажать клавишу "Enter", чтобы вызвать консольный режим. Теперь можно вводить коды:

- FORCEFOOD — получить 1000 единиц еды;
- FORCENOVA — получить 1000 единиц Nova;
- FORCECARBON — получить 1000 единиц углерода;
- FORCEORE — получить 1000 единиц руды;
- FORCEBUILD — ускоренная постройка;
- FORCEEXPLORE — открыть всю карту;
- FORCESIGHT — отключить "туман войны";
- DARKSIDE# — уничтожить противника (# — номер врага от 1 до 8);
- TARKIN — уничтожить всех врагов;
- SKYWALKER — выиграть миссию;
- 1...58 — прослушать цитату из Star Wars.

Unreal Tournament 2003

Нажмите клавишу "Tab" и введите коды:

- god — режим Бога;
- loaded — получить все оружие и 999 патронов;
- allammo — получить по 999 патронов ко всему имеющемуся у вас оружию;
- fly — получить возможность летать;
- ghost — получить возможность проходить сквозь стены;
- walk — отмена режимов полета и прохождения сквозь стены.

Warcraft 3: Reign of Chaos

Нажать клавишу "Enter", ввести код и снова нажать "Enter" (на экране должна появиться надпись "Cheat enabled"):

- lseedeadeople — открыть всю карту;
- allyourbasearebelongtous — мгновенно выиграть миссию;
- somebodysetupsthebomb — мгновенно проиграть миссию;
- thereisnospoon — получить бесконечную ману;
- whosyourdaddy — получить неуязвимость.

The Elder Scrolls 3: Morrowind

Во время игры нажмите клавишу "~", чтобы вызвать консоль. Наберите "player->код" и нажмите клавишу "Enter". Вместо "код" подставляются следующие значения:

- TMG — вкл./выкл. режим Бога;
- Help — показать список чит-кодов;
- SetFlying 1 — получить возможность летать;
- SetSuperJump 1 — включить суперпрыжки;
- SetWaterWalking 1 — ходить по воде (0 — отключить);
- SetWaterBreathing 1 — дышать под водой (0 — отключить);
- SetLevel — установить уровень игрока;
- AddItem "Gold_100" — получить золото;
- SetFatigue — установить значение fatigue;
- SetMagicka — установить значение magic;
- SetHealth — установить значение health;
- COMMAND — полный список команд.

FIFA 2003

Зайти в директорию с игрой, любым текстовым редактором открыть файл soccer.ini и добавить в него следующие строки:

- ONE_ON_ONE=1 — включить режим игры "один на один";
- PRACTICE_MODE=1 — включить режим тренировки;
- WINDOWED=1 — запускать игру в окне;
- ABSOLUTELY_PERFECT_GOALIES=1 — получить очень хорошего вратаря;
- AGGRESSIVE_TACKLE_CHEAT=1 — включить режим "жесткой игры" (подножки, удары и пр.);
- CHEAT_EQUAL_TEAM_STATS=1 — уравнивать статистики команд.

Gothic

В любом текстовом редакторе — открыть файл gothic.ini и заменить "test mode 0" на "test mode 1". Затем войти в консольный режим (нажать F2) и набрать следующие коды:

- cheat — режим Бога;
- cheat full — пополнить здоровье.

Arthur's Quest: Battle for the Kingdom

Во время игры — вызвать консоль клавишей "~" и ввести следующие коды:

- god — режим Бога;
- ExitLevel — пропустить уровень;
- kfa — получить все оружие и патроны;
- FragSelf — совершить самоубийство;
- heal — восстановить здоровье.

SimCity 4

Для перехода в консольный режим, во время игры — одновременно нажать "Ctrl" + "X". После введения нужного кода — нажать "Enter":

- FightThePower — убрать потребление энергии;
- You don't deserve it — получить все награды;
- WeaknessPays — получить 1000 USD;
- HowDryAm — убрать потребление воды;
- Stopwatch — выключить/включить время;
- HelloMyNameIs # — изменить ваше имя на #;
- WhatTimeIsIt # — установить время на #;
- WhereRUFrom # — изменить название города на #;
- SizeOf # — увеличить до размера #;
- TastyZots — убрать предупреждения;
- Recorder — начать запись демо.

Если код введен правильно, консоль автоматически закроется.

Impossible Creatures

Во время игры — нажать клавишу "~" для вызова режима консоли и вводить следующие коды:

- cheat_buildings — получить больше зданий;
- cheat_coal(9999) — получить 9999 единиц угля;
- cheat_electricity(9999) — получить 9999 единиц электричества;
- cheat_killself — совершить самоубийство;
- cheat_rank — повысить свое звание.



КУПЛЮ, ПРОДАМ, ОБМЕНЯЮ

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



119454, а.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

Продаю: C1-55, C1-112, ГЗ-36А, ЧЗ-32, M1109, M2051, Ц4313, Ц4353, P40114, K78177/K7819G (бу), программы для ZX-Spectrum и др.

Куплю документацию на ЧЗ-32, ГЗ-36А.
Возможен обмен на тонер ("Пионер", "Техникс" и др.).
Тел. (095) 944-53-13, с 19.00 до 21.00, Михаил.

Продаю:

- компьютер "Scorpion 256" (расширенная клавиатура для "Scorpion"; контроллер для подключения IBM-клавиатуры и мыши; 2 дисковод "Robotron");
- монитор "Электроника 32ВТЦ-202";
- два принтера (без головок) "Электроника СМ 6312";

- 160 дискет 5,25S (системные программы, электронные журналы и газеты);
- черно-белый монитор "Электроника MC6105.01";
- два дисководы TEAC на запчастях;
- дисковод "Электроника 5305";
- дисковод "Epson SD-600".
165300, Архангельская обл., г.Котлас,
ул.Мартыновская, 44 — 7, Ятченко Н.Ф.

Ищу возможные схемы модернизации игровой приставки SUBOR (подключение к ПК, перепроектирование ПЗУ и т.д.). Приму в дар любой ЭЛТ-монитор.
Тел. в Минске 257-91-46.

Куплю: любые выпуски сборника "Радиодизайн", журналы "Телеспутник" за 2002-2003 гг.; панель к лампе ГУ-34Б: диод 6Д13Д; ПК, совместимый с "ZX-Spectrum" с контроллером дисковода и пакет программ к нему.
40000, Украина, г.Сумы, а/я 9, Александр.

Ученик 11-го класса с благодарностью примет в дар компьютер.
Беларусь, г.Гродно, ул.Калючинская, 6 — 11, Олег.

Куплю запрограммированные ПЗУ для контроллера TR-DOS (версия 5.03 или 5.04) или готовый контроллер со схемой.
385765, Р. Адыгея, Майкопский р-н, ст.Кужорская, ул.Чапаева, 1, Ветров И.П.

Ищу контроллер дисковода к ПК "Байт" (можно без ВГ-93) с дисководом 5,25" и принципиальную схему к ПК "Балтик".
223082, Минская обл, п/о Бровки, 31,
Кривошеков Б.Г.
Тел. 504-35-16, Борис.

Куплю контроллер жесткого диска для ПК "Искра 4816.03" (1991 г.) и программное обеспечение для ПК "Искра-1030М".
453500, Башкортостан, г.Белорецк, а/я 21,
Сафонов С.Н.
Тел. (34792) 4-45-67.

Ищу владельцев "Вектор-06L".
Куплю АОН-модем, вичестер и его контроллер в рабочем состоянии.

Продам "Мощный музыкальный автомобильный сигнал своими руками" — подробное описание сборки, платы, подключение к любому типу автомобиля. Приложить конверт с обратным адресом.
353866, Краснодарский край, г.Приморско-Ах-

тарск, Авиатородок, 4/24, Тыщенко Р.В.
Тел. (861-43) 2-76-89.

Приглашаются к сотрудничеству пользователи и программисты в iS-DOS с целью создания информационного сборника по обмену опытом работы в этой ОС. Подробности в газете "Абз@ц" N13 или в конверте с обратным адресом.
412302, Саратовская обл., г. Балашов,
ул. Красина, д.82., Ильясов Е. В.

Продаю ПК "Электроника БК 0011" с блоком питания.
E-mail: mdmennev@yandex.ru. Михаил.

Продам ПК "KAY-1024" (с документацией) + 5,25" FDD "Robotron" + кучу дисков.
230001, г.Гродно, ул.Суворова, 276 — 19.
Тел. 8-029-6-20-82-86.

Ищу любое программное обеспечение для "ZX-Spectrum 48 (128)K" на дискетах 5,25" (TR-DOS 5.03); ищу схемы подключения периферии (программатор, принтер и т.д.) с программным обеспечением; книгу Ларченко, Родионов. "ZX-Spectrum и TR-DOS для пользователей и программистов".
385765, Р.Адыгея, Майкопский р-н, ст.Кужорская, ул.Чапаева, 1, Ветров И.П.

Уважаемые читатели!

Подписаться на наши журналы (индексы: 48996 — "Радиомир", 48924 — "Радиомир. КВ и УКВ", 48925 — "Радиомир. Ваш компьютер") во II полугодии 2003 г. можно по каталогу Агентства "Роспечать", стр. 264 или по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямым договорам").

Те, у кого возникли проблемы с подпиской, могут получить их из редакции. Там же можно заказать имеющиеся в наличии отдельные номера журналов за предыдущие годы.

Год	Можно заказать следующие номера журналов			Расценки на 1 экз. (с учетом пересылки)		
	Радиолобитель	Радиолобитель. Ваш компьютер	Радиолобитель. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
1998	4, 5	1 — 5, 8, 9, 11, 12	1	20	700	5
1999	3, 9, 10, 11	1 — 6, 10, 11, 12	3, 5, 6	25	800	5
2000	2 — 6, 8 — 12	2 — 8, 10 — 12	4, 6, 7, 9, 11, 12	25	900	6
2001	2 — 6	1, 2, 4 — 6	2 — 6	30	1000	6
	Радиомир	Радиомир. Ваш компьютер	Радиомир. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
2001	7 — 12	7, 9 — 12	7 — 12	35	1300	7
2002	1 — 12	1 — 12	1 — 12	40	1400	8
2003	1 — 12	1 — 12	1, 2, 4 — 12	45	1700	9

Наши платежные реквизиты:

- для жителей России и стран СНГ (кроме Беларуси) —
получатель: ООО "Радиомир Пресс", ИНН 7729404741,
р/с 40702810900010000084 в ООО КБ "Агропромкредит", ф-л Центральный, г.Москва,
корр. счет 30101810500000000109, БИК 044525109
Адрес банка: 113054, г.Москва, ул.Зацепа, 43Б;

- для жителей Беларуси —
получатель: УП "РПД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" в г.Минске код 121.
Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.

На бланке почтового перевода необходимо очень четко написать свой почтовый индекс, полный адрес, фамилию, имя и отчество полностью. В графе "Для письма" необходимо точно перечислить, какие конкретно номера какого из журналов Вы заказываете. При оплате платежным поручением эти сведения необходимо указать в графе "назначение платежа".

Вся информация по E-mail: rm-sales@radio-mir.com
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) 208-85-50, 208-67-55,
e-mail: inter@aif.ru

В магазинах радиодеталей "ЧИП и ДИП":

- г.Москва, ул.Гилыровского, 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 "ТРЭНТЭК": 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
"Тулеской" (ст. метро "Тулеская", место 412-38).

На УКРАИНЕ:

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

В БЕЛАРУСИ:

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

Inquisition



ENGLISH VERSION

SIMULATION

Microsoft
Combat Flight Simulator 3
Battle For Europe



1C ИГРОТЕКА NIVAL

КА-52 против КОМАНЧА

РАЗОРВАННОЕ НЕБО

ЖАЖДА СКОРОСТИ
НО РУССКИ

Вот это скорость! Вот это драйв! В русском языке! Впечатляет!



ПОЛНОСТЬЮ НА РУССКОМ ЯЗЫКЕ **Русский Проект**

SIM CITY 3000
UNLIMITED

empire

ИГРЫ ОТ «1C» И NIVAL

OPERATION

BLOCKADE

ПОЛНАЯ РУССКАЯ ВЕРСИЯ

HARD TRUCK
18 WHEELS OF STEEL

ПОЛНОСТЬЮ НА РУССКОМ ЯЗЫКЕ **Русский Проект**

ПРИКЛЮЧЕНИЯ МУМИ-ТРОЛЛЕЙ
БОЛЬШАЯ ВЕЧЕРИНКА

СТРАТЕГИИ НА РУССКОМ ЯЗЫКЕ **Русский Проект**

МИР АНТИЧНЫХ ИМПЕРИЙ

4 в 1
ZEUS: Master Of Olympus
CAESAR 3
PHAROAH
CLEOPATRA

ПОЛНОСТЬЮ НА **ФАРГУС** РУССКОМ ЯЗЫКЕ

GREEN BERETS
POWERED BY
DUTH II