http://radio-mir.com

индексы: 48925, 45995

радиомир

Ваш компьютер

10•2004

Компьютеры для дома и офиса

от компании СитиПринт

Домашний компьютер: обучение как развлечение!

С новым компьютером Ваши дети смогут быстрее освоить любой предмет, ведь у них будет максимум возможностей для обучения. Компьютер от компании "СитиПринт" на базе процессора Intel® Pentium® 4 с технологией HT станет для них настоящим образовательным центром.

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

СитиПринт

КОМПЬЮТЕРЫ И ОРГТЕХНИКА

220006, г.Минск, ул.Полевая 28-7А.

Тел./факс (017) 2850134 www.citiprint.by

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

Главный редактор
Ольга СТРЫЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

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

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

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

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

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

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

Материалы для публикации принимаются в рукописном, печатном и электронном вариантах

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

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

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

Подписано к печати 24.08.2004 г.
Формат 60 x 84 1/8.
Печать офсетная. 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ № 2649.

радиомир **Ваш компьютер**

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

№10/октябрь/2004

ЧИТАЙТЕ В НОМЕРЕ

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

В.КУЦ,
E-mail: victor_kootz@mail.ru

Я тебя вижу, а ты меня?

Активное внедрение мультимедийных технологий в современные компьютеры никак не могло обойти такую важную область как связь и Интернет. И далеко не самое последнее место в составе бывших "писишек", превратившихся уже в настоящие мультимедийные комплексы, занимают web-камеры (**рис.1**), популярность которых, благодаря их невысоким ценам и широким функциональным возможностям, в последнее время заметно возросла. Но на просторах нашей Родины такие изделия пока не очень востребованы, может быть потому, что пользователи еще не "распробовали" в полной мере этот продукт и не совсем представляют себе, что интересного или полезного с ним можно делать.

Рис.1. Logitech QuickCam Pro 4000 — классический образец web-камеры.

Основным предназначением web-камеры (как явствует из названия) является передача видеоданных с относительно низкой скоростью по каналам локальной (в том числе и беспроводной) сети, а иногда и по модемным коммутируемым соединениям. Разумеется, ожидать высокого качества записываемых видеороликов или цифрового фото в данном случае не приходится — не тот класс устройств. Да и не имеет особого смысла снабжать такие камеры высококачественными, а значит, и очень дорогими объективами и сенсорами (и прибавлять к цене сотню-другую долларов), если передача видеоданных по существующим сегодня линиям в режиме реального времени требует значительного их сжатия, неизбежно приводящего к заметной потере качества изображения. Именно поэтому к web-камере, по крайней мере на современном этапе, следует отнестись как к устройству, у которого при максимально доступной цене аппаратные возможности более или менее сбалансированы с пропускной способностью низкоскоростных каналов связи.

Для чего может использоваться web-камера? Во-первых, как уже упоминалось выше, с ее помощью можно участвовать в видеоконференциях. Не вызывает сомнения, что при этом общение через Интернет становится гораздо более легким и приятным, чем стандартный чат или даже "беседы голосом" с использованием протокола TCP/IP.

На западе видеоконференции уже давно нашли самое широкое применение в крупных компаниях, юридических фирмах, в органах власти, в сферах здравоохранения, образования и во многих других областях. Существует масса ситуаций, когда по ходу разговора собеседнику нужно показать нечто (фрагмент чертежа, текст документа, внешний вид установки и т.д.), вместо того чтобы тратить десятки и сотни слов на описание того, как "это" выглядит. С web-камерой сделать это очень просто, и собеседник увидит все, что вы пожелаете ему показать. Конечно, участие в видеоконференциях никогда не заменит личного общения, но они позволяют добиться принципиально нового уровня общения людей, подчас разделенных многими тысячами километров. Ведь согласно известным исследованиям, при телефонном разговоре можно передать только десятую часть транслируемой информации. Использование телефонной связи в совокупности с факсимильной позволяет увеличить объем передаваемой информации примерно до 25%. В случае же, когда есть возможность в процессе разговора следить за жестикуляцией и мимикой собеседника, КПД передачи информации достигает 60%. Но не только сухая статистика убеждает нас в том, что видеоконференции позволяют добиться качественно нового уровня связи. Менеджеры компаний, уже использующих видеоконференции, считают, что системы видеоконференций резко сокращают временные и финансовые затраты на командировки их сотрудников и делают проводимые совещания более продуктивными. Помимо высокого комфорта аудио- и видеосвязи, использование видеоконференций при деловом общении зачастую позволяет сэкономить значительные средства на телефонных переговорах, особенно в условиях большого расстояния между собеседниками.

Очевидно, что для работы в режиме видеоконференции необходима довольно высокая скорость соединения, но это уже от web-камеры не зависит. Обычные телефонные каналы вполне подходят для передачи аудиосигнала, но качественную передачу видеопотока они не обеспечивают. Эта проблема медленно, постепенно (у нас, в России — ну очень медленно и слишком уж постепенно!), но решается. Вспомним хотя бы, какой экзотикой были локальные вычислительные сети еще лет пять тому назад. А сейчас в редком офисе компьютеры не объединяются в локальную сеть, которая великолепно подходит для организации высококачественной видеоконференции. Да и среди пользователей Интернета (в том числе и частных), доля счастливых обладателей высокоскоростного доступа увеличивается.

Кроме участия в видеоконференциях, с помощью установленной в помещении web-камеры через определенные промежутки времени можно делать отдельные снимки и автоматически отправлять их в Интернет для размещения на соответствующих сайтах. Конечно, при использовании ее в качестве стационарной видеокамеры, web-камера проигрывает обычным как по функциональной насыщенности, так и по качеству получаемой "картинки", но бесспорными плюсами web-камер является их более низкая цена и запись полученного видеосигнала непосредственно в компьютер. Таким образом, практически любую web-камеру можно без труда превратить в элемент охранной системы помещения, начинающей захват изображения при обнаружении каких-либо изменений в кадре. Можно настроить камеру так, чтобы компьютер подавал звуковой сигнал при движении объекта в кадре. Кроме того, сигнал тревоги и изображение можно в автоматическом режиме передавать по e-mail или транслировать видео на web-сайт.

И наконец, большинство моделей web-камер, за исключением совсем уж дешевых, можно использовать как обычный цифровой фотоаппарат. Разумеется, получается стационарный фотоаппарат, ограничивающий свободу передвижения с ним длиной имеющегося кабеля, да и разрешающая способность у такого аппарата не ахти какая. Однако для создания домашнего цифрового фотоальбома этих возможностей может оказаться вполне достаточно. Хотя отдельные, наиболее дорогие модели web-камер, сумели избавиться и от "хвоста" — для этого они имеют встроенную флэш-память и батарейное питание.

Но все вышеперечисленные возможности web-камер сегодня мало кого могут удивить — все они давно стали для всех нас привычными. Однако компании-производители компьютерного "железа" в извечной погоне за благосклонностью покупателей постоянно изобретают для своей продукции новые, все более и более экзотические функции. Так, компания Intel комплектует свою Pocket PC Camera оригинальной программой Digimarc Mediabridge. Необычность этой программы заключается в том, что если пользователь (счастливый обладатель комплекта из Intel Pocket PC Camera и современного компьютера с установленной Digimarc Mediabridge), читая журнал или газету, заинтересуется определенной картинкой или статьей, то, как утверждают авторы программы, он, всего лишь поднеся данную картинку или статью к камере, может получить о ней дополнительную информацию. Каким образом? Просто перейдя на web-страничку, располагающую дополнительной информацией о том, что изображено на картинке, или по теме заинтересовавшей статьи. К сожалению, указанной возможностью сравнительно трудно воспользоваться отечественному пользователю, так как Digimarc Mediabridge "работает" лишь с ограниченным числом журналов, естественно, англоязычных. Такие журналы имеют на "поддерживаемых" страницах эмблему "D". Программа считывает служебную информацию со страницы журнала и, пользуясь специальной базой данных, переходит на соответствующую страницу в Интернете. Просто и изящно, не правда ли?

"Сердцем" любой web-камеры является сенсор. Большинство устройств среднего и верхнего ценовых диапазонов используют CCD-сенсоры (Charge Coupled Device — приборы с зарядовой связью). Они производятся небольшим числом компаний, в частности, Sony, Philips, Kodak, Matsushita, Fuji и Sharp, и обеспечивают гораздо лучшее качество изображения, чем CMOS-сенсор, но и стоят при этом существенно дороже. Низкая стоимость CMOS-сенсоров обусловлена тем, что в них обработка изображений и преобразование аналог-код осуществляются непосредственно в чипе. Кроме того, CMOS-камеры требуют меньше сопутствующей электроники и печатных плат и могут быть размером с ноготь или еще меньше. В подавляющем большинстве web-камер используются простейшие CMOS-сенсоры, чаще всего содержащие порядка 300 тыс. пикселов, что обеспечивает захват изображения с разрешением не более 640x480 точек.

Такие камеры позволяют захватывать видео с частотой от 15 до 30 кадров/с при типичном разрешении 320x240. При этом для данных видеоформата RGB, как правило, обеспечивается 24-битная глубина цвета. Большее же разрешение — 640x480 точек — вряд ли будет целесообразным для проведения видеоконференций, поскольку в этом случае скорость потока данных (только видео!) не превысит величину 5...10 кадров/с. Для нормальной передачи видеопотока по каналам связи (в первую очередь — модемным) обычно осуществляется его оптимизация под заданную полосу пропускания, при этом можно выбирать между четкими картинками с маленькой частотой кадров или динамичными, но менее четкими. Для локальной сети или скоростного канала (скорость от 80 Кбит/с и более) вполне реально одновременно получить и скорость, и качество.

Если же web-камера работает в режиме фотоаппарата и не ограничена пропускной способностью линии, то сделанные с ее помощью фотографии, чаще всего с разрешением 640x480, а в самых последних моделях и больше (**рис.2**), получаются вполне четкими, хотя в большинстве случаев по ним все-таки явно видно, что снимки сделаны

Рис.2. Mustek gSmart Mini 3 — почти настоящий фотоаппарат с 2,1-мегапиксельной CMOS-матрицей.

отнюдь не "настоящим" цифровым фотоаппаратом. Ведь для снижения стоимости, web-камеры, имеющие возможность делать автономные снимки, имеют минимальную функциональность — их объективы чаще всего имеют малую светосилу (а фотовспышка у них, естественно, отсутствует), не предусмотрена и возможность автофокусировки. Первое означает, что приемлемое качество съемки достижимо только при хорошем освещении объекта, а второе — что изображение почти всегда будет нерезким (если оно расположено на расстоянии, отличном от установленного производителем камеры).

Еще одна проблема недорогих web-камер — это проблема скорости обработки аудио- и видеопотока, т.е. кодирования передаваемых и декодирования получаемых данных. Дело в том, что в видеоконференциях используются специальные и весьма эффективные алгоритмы сжатия потока в десятки (а подчас и сотни!) раз. А все операции по кодированию/декодированию видеосигнала осуществляются программным способом. Если компьютер не успевает обрабатывать поток, то появляются пропущенные кадры, сбои в речевом канале и т.п.

Простейшая web-камера подключается к компьютеру с помощью интерфейса USB (через него она обычно и получает питание). Однако все USB-камеры имеют и характерный недостаток — сравнительно небольшую частоту смены кадров (изображение "тормозит"), связанную с ограниченной скоростью передачи данных по шине USB 1.1 — не более 12 Мбит/с (из которых web-камере "достается" в лучшем случае 5...9 Мбит/с). Впрочем, со своим основным назначением — видеосвязью через Интернет или телефонную сеть — такие видеокамеры справляются нормально, поскольку передать по телефонной линии видео с более высокой частой кадров или более высоким разрешением все равно невозможно.

Практически все web-камеры, объективы которых ничем не защищены от влаги и прочих внешних воздействий, предназначаются для использования исключительно в закрытых помещениях, они комплектуются пластиковой платформой для установки на монитор либо специальной скобой для закрепления на корпусе ЖК-дисплея.

Очень часто производители стараются снабдить свои web-камеры дополнительными функциями, предоставив в распоряжение пользователя некое универсальное устройство типа "все в одном". При этом очень часто ни одну из заявленных функций они не выполняют на должном уровне. В итоге получается довольно неплохая электронная игрушка, которая может принести много радости детям, тем более, что часто такие устройства комплектуются отличными программами, обеспечивающими интересные забавы с использованием технологий цифровой фотографии.

Хотя многие настольные web-камеры комплектуются встроенными микрофонами, как показывает практика, эффективность их использования достаточно низка, поскольку обычно камера контактирует с твердой поверхностью стола или панелью монитора и "ловит" огромное количество паразитных шумов и вибраций.

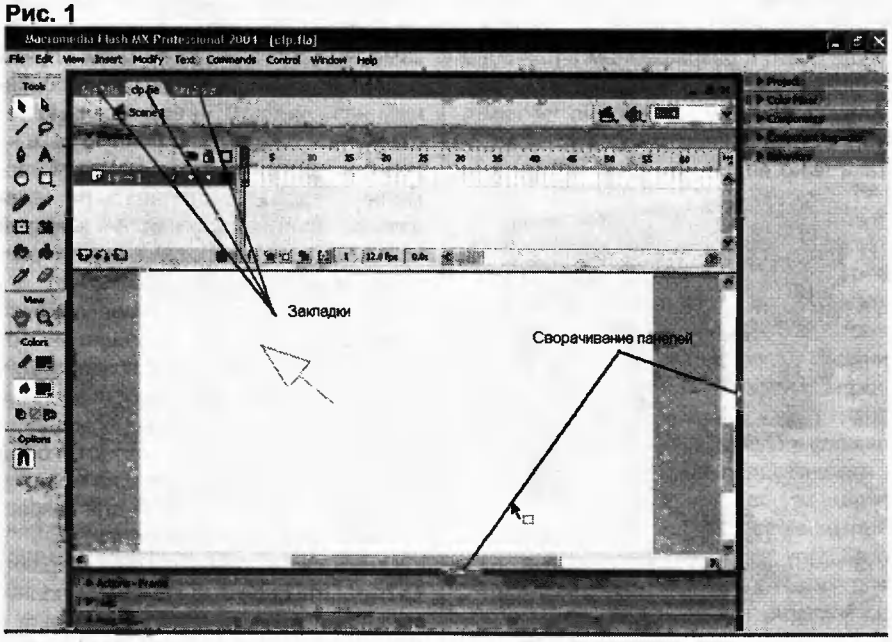
А.ГРИНЧУК,
E-mail: gav@nihe.niks.by

Анимация с Macromedia Flash MX

Flash в Internet

Мы подошли к концу нашего знакомства с основами Flash MX. С момента публикации, а уж тем более, написания первой статьи данной серии утекло немало воды. И это время компанией Macromedia было использовано весьма продуктивно. Появилось следующее поколение популярных инструментов для web-дизайна. Как и ранее, Flash, Dreamweaver, Fireworks и FreeHand объединены в общий пакет Macromedia Studio. Только название немного изменилось, сейчас это Studio MX 2004, соответственно, новая версия Flash теперь носит название Flash MX 2004. Более того, появились две подверсии — обычная и с приставкой Professional.

Рис. 1

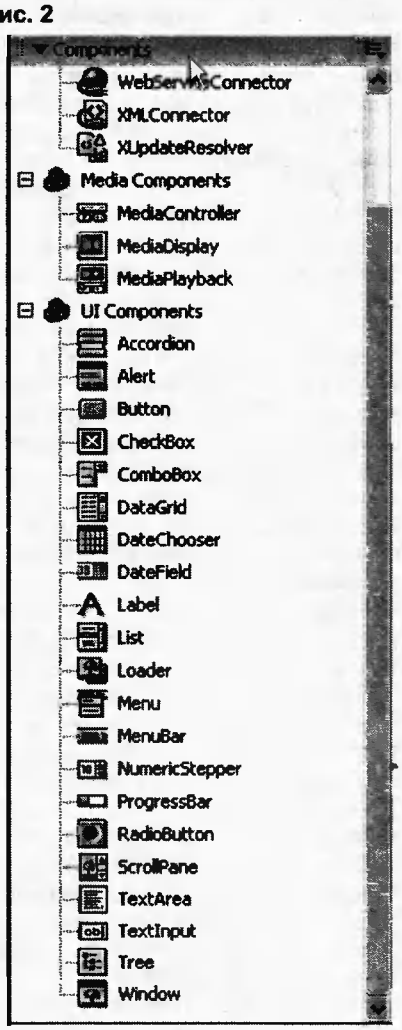


Первые впечатления оказались весьма благоприятными. Значительно выросла степень интеграции между программами. И если остается спорным вопрос о том, кто же лучше обрабатывает растровую графику, Fireworks или PhotoShop, то совершенно очевидно более выигрышное положение Fireworks, если нужно обработать JPEG- или GIF-файл для последующего использования во Flash. О самой программе Flash MX 2004: работать стало удобнее, особенно когда приходится иметь дело с 15-дюймовым монитором. Связано это с тем, что наборы панелей можно свернуть-развернуть, освобождая тем самым место для работы. Каждый открытый файл теперь снабжается ярлычком, поэтому уже нет необходимости постоянно обращаться к меню Window (**рис.1**). Что касается создания анимации и рисования, то и здесь есть небольшие приятные новинки. Упростилось создание простейших типов анимации (**Insert/Timeline Effects**) и скриптов (панель **Behaviors**). Но все же возникает ощущение, что развитие программы в качестве средства создания анимации если не остановилось, то уж точно резко замедлилось. Зато появилось большое число компонентов (**Components — рис.2**), в числе которых есть как стандартные заготовки для форм (переключатели, прокручивающиеся списки, меню), так и объекты, облегчающие обмен информацией с сервером. Есть готовые текстовые области с прокруткой и предзагрузчик. Появилась новая версия языка программирования — ActionScript 2.0. Похоже, что компания Macromedia решила окончательно доказать, что Flash — средство создания серьезных приложений, которое мимоходом умеет еще и мультики рисовать.

Обо всех новинках и изменениях лучше узнавать из самого оперативного источника — сети Internet. В первую очередь, стоит посетить сайт разработчика www.macromedia.com. Здесь можно найти самую оперативную и точную информацию не только о Flash, но и о сопутствующих программных продуктах. Единственный минус этого сайта — англоязычность. Массу полезной информации, переводы справки и учебники на русском можно найти по адресу www.zona5.al.ru. Старейшим и крупнейшим flash-порталом в русскоязычной части всемирной паутины по-прежнему считается www.flasher.ru, здесь собрана огромная коллекция трюков, уроков с исходниками, игрушек и идей. Особое внимание стоит обратить на ссылки "сайт недели" и "архив победителей". Они приведут вас на сайты, которые заставят с уважением относиться к возможностям Flash.

Если не терпится посмотреть на живые примеры реализации flash-технологий, то любая поисковая система после ввода ключевых слов "лучшие flash-сайты" выдаст вам десятки рейтингов и обзоров. Мы тоже устроим небольшой обзор, ни в коей мере не претендующий на объективность. Просто сайты сумели обратить на себя внимание автора этих строк.

Рис. 2



www.flahoo.com. Из названия понятно, о чем идет речь — flahoo = flash + yahoo. Это небольшой, но вполне практичный каталог flash-ресурсов, даже по внешнему виду напоминающий поисковик yahoo. Само намерение создать такой ресурс вызывает уважение, хотя в связи с бумом flash-технологий авторам сайта придется в ближайшее время изрядно попотеть.

www.ahouse.ru. Студия ahouse design. Небольшое окно flash-сайта появляется на фоне основной html-страницы и выглядит как ее часть. Эффектная иллюзия, дополненная экономным и неярким дизайном. Благодаря этому сайт выглядит очень стильно, особенно с удачно подобранным звуковым сопровождением.

www.2methods.com/secondmethod. Персональное пространство Виталия Онищенко. Интересная идея со всплывающей панелью Customize. После загрузки сайта можно поменять его цветовую гамму и дизайн оформления. В общем, не нравится сайт — сделай так, как нравится.

www.2advanced.com. Рекламный сайт хостинговой компании www.2advanced.net. Поражает обилие мелких деталей. Сайт на самом деле состоит из целого набора роликов, что заметно при загрузке сайта — работает целая команда предзагрузчиков. Судя по информации из обзоров Internet, в свое время этот сайт открыл новую страницу в истории flash-дизайна и вызвал целую лавину подражаний.

www.relevare.com. Стоит обратить внимание на навигационную панель в виде системы вложенных квадратов. Такое сложно повторить без Flash. Пример flash-сайта, в котором упор сделан на функциональность, а не на внешние эффекты.

www.avestadesign.com. В какой-то мере противоположность предыдущему образцу. Набор красочных проектов: египетский оракул, гадание по И-Цзин и Другой Мир, полезной информации немного, но зато как сделано! Судя по описанию, использовалась еще Flash 3.0. Маленькое замечание — перед закрытием сайта не забудьте нажать кнопку отключения звука, иначе придется долго слушать поначалу приятную мелодию.

www.canubia.com. Один из недавних победителей конкурса на flasher.ru. Здесь все сбалансировано: эффектность и функциональность, оформление и содержание. Плюс хорошее озвучивание. Это сайт студии web-дизайна, причем, судя по сайту, достаточно профессиональной.

Рис. 3

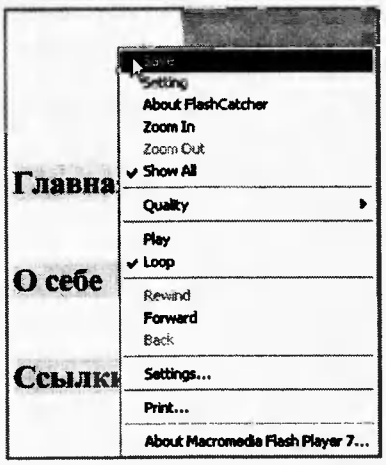
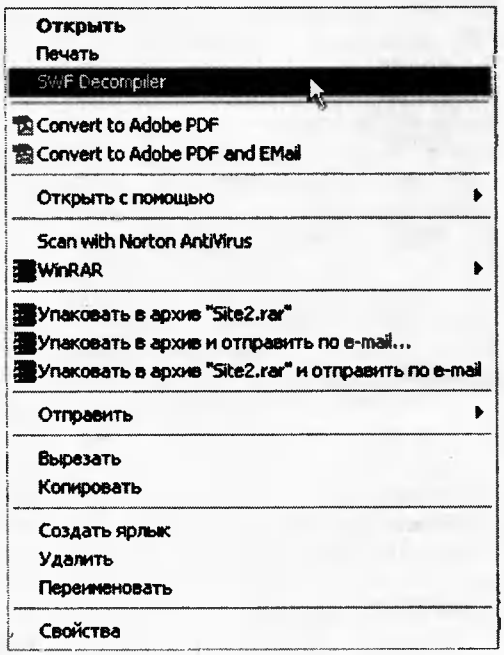
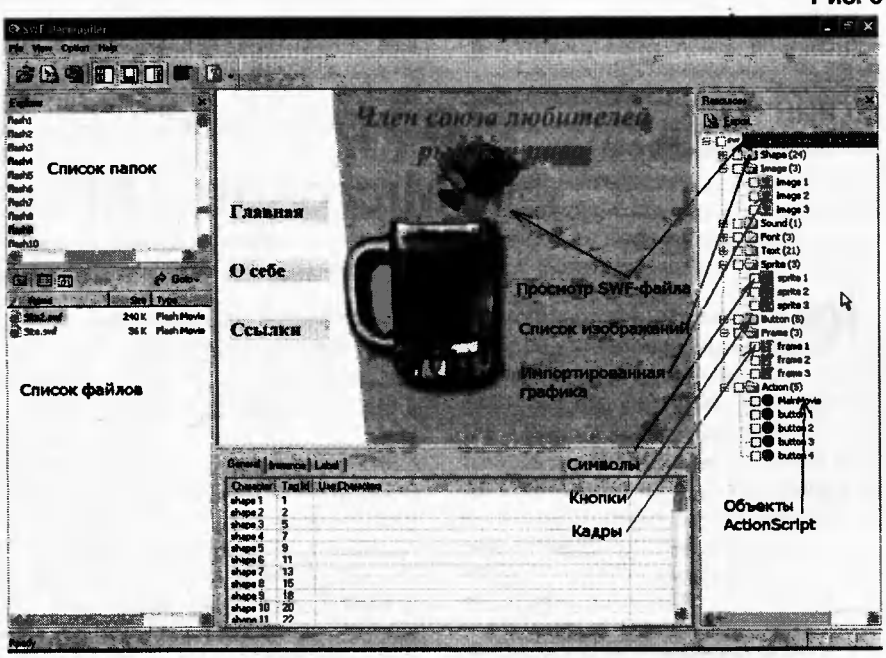


Рис. 4



Наверняка при путешествии по всемирной паутине вас заинтересуют эти и некоторые другие сайты. Однако у flash-технологий есть и обратная сторона — не так просто сохранить увиденное на экране. Что же можно сделать в этой ситуации? В некоторых случаях, увы, ничего. Попытайтесь просмотреть папку Temporary Internet Files, там часто кешируются просмотренные SWF-файлы. Иногда помогает использование программ вроде FlashCatcher (дистрибутив можно найти на winall.ru/soft.shtml). После ее установки в контекстном меню (после щелчка правой клавишей мыши по flash-вставке) появляется команда **Save** (Сохранить — **рис.3**).

Рис. 5



Сохраняется, естественно, файл формата SWF. Разобраться с его устройством поможет другая программа — SWFDecompiler (сайт разработчика — www.sothink.com, см. также winall.ru/soft.shtml). Программа тоже может быть вызвана из контекстного меню в проводнике (**рис.4**). В открывшемся окне можно увидеть разобранную на составные части анимацию (**рис.5**). Удобно, что в списке присутствуют внедренные символы (**Sprite** — **рис.6**) или звуки (**Sound**), и в наличии имеется команда Export. Более того, можно посмотреть, в каких объектах какой программный код был использован (**рис.7**). Главное в этих экспериментах — не забывать о существовании закона об охране авторских прав. Поэтому лучше смотреть и изучать, но делать все по-своему. Почти

Рис. 6

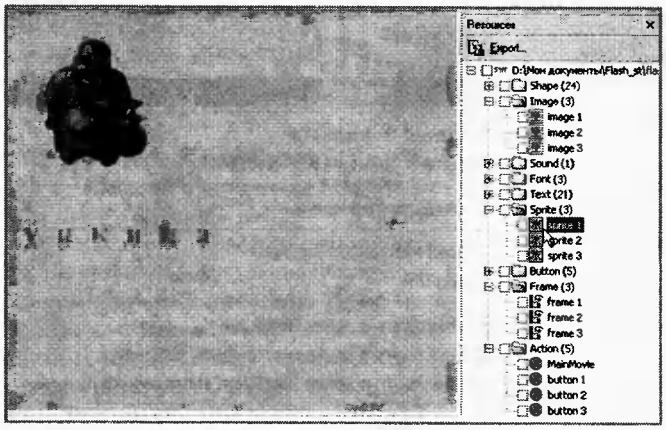
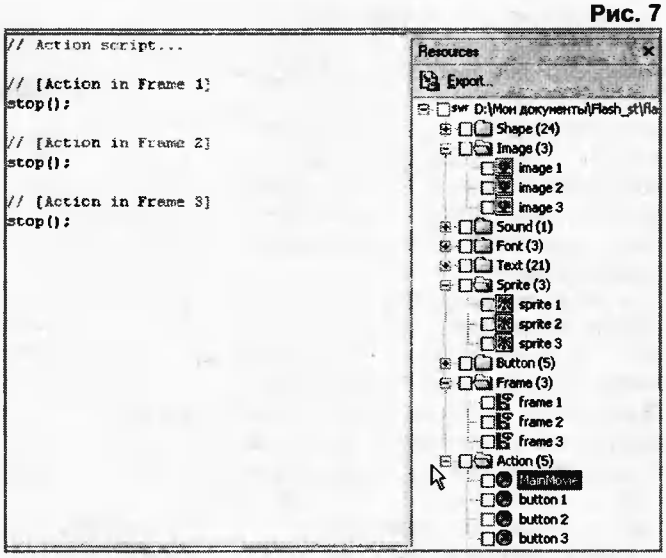


Рис. 7



все, что нужно для этого, мы изучили. А именно:

- рисование;
- импорт графики и звуков;
- создание различных типов анимации;
- работу с символами;
- добавление действий с использованием ActionScript.

Если поначалу у вас ничего не получается, к вашим услугам в Internet целые наборы бесплатных заготовок, например, на www.comteche.com.

Программа Macromedia Flash давно доказала свою эффективность при создании красочных мультимедийных и интерактивных сайтов. С другой стороны, заметна тенденция к увеличению функциональных возможностей создаваемых flash-сайтов. Так что, последнее слово в развитии flash-технологий еще не сказано. Дерзайте, и пусть вас не смущает тот факт, что сайт самой Macromedia не целиком сделан во Flash. Кто знает, возможно, это просто вопрос времени.

С.ГРИНЧУК,
E-mail: grinchuk@nihe.niks.by

Разработка web-сайтов в Microsoft Office FrontPage 2003

(Продолжение. Начало в №№8-9/2004)

Вставка и редактирование гиперссылок

Как вы помните, на прошлом занятии мы начали работу над персональным сайтом, который к концу занятия уже содержал три web-страницы: **index.htm** — начальную страницу сайта; **biography.htm** — страницу с краткими биографическими сведениями и **hobby.htm** — страницу, содержащую информацию об увлечениях. Сегодня мы поговорим о том, как эти страницы связать с помощью гиперссылок. Для начала выполните

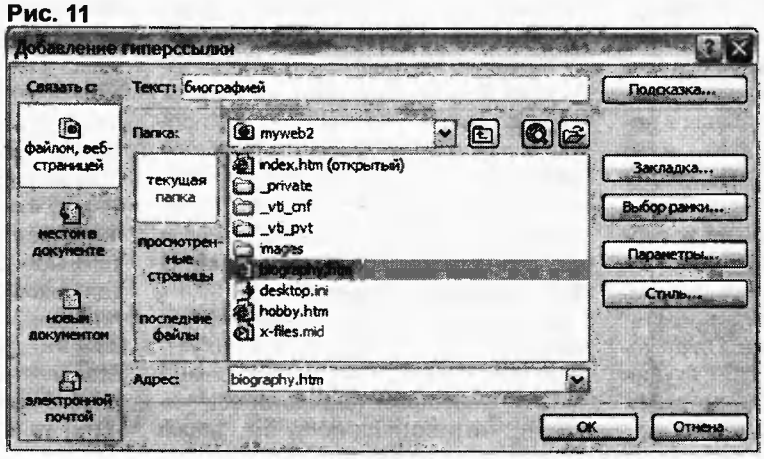
1. Запуск Microsoft Office FrontPage 2003 с помощью команды **Пуск/Программы/Microsoft Office/Microsoft Office FrontPage 2003**.

Естественно, для дальнейшей работы нам понадобится сайт **myweb2**, созданный на прошлом занятии. Если при запуске Microsoft FrontPage этот сайт автоматически открылся как последний использовавшийся, тогда вы сразу можете приступать к п.3. В противном случае вам дополнительно придется выполнить

2. Открытие web-сайта, созданного на прошлом занятии

Чтобы открыть ранее созданный сайт, проще всего воспользоваться разделом **Открыть** области задач **Приступая к работе**. В этом разделе перечислены последние использовавшиеся сайты, и для открытия любого из них достаточно выполнить щелчок мышью по его имени. В случае, если в этом разделе отсутствует название нужного сайта, воспользуйтесь командой **Дополнительно...**, после чего в появившемся диалоговом окне **Открытие веб-узла** откройте свою рабочую папку, выделите корневую папку сайта **myweb2** и подтвердите открытие сайта с помощью кнопки **Открыть**. Вызвать диалоговое окно **Открытие веб-узла** можно также с помощью команды **Файл/Открыть узел...** или же открыв список инструмента панели **Стандартная** и выбрав команду **Открыть узел...**.

Рис. 11



3. Вставка и редактирование гиперссылок

Итак, сегодня мы поговорим о создании гиперссылок. Любая гиперссылка включает в себя так называемый указатель гиперссылки (фрагмент текста или графический объект, которые станут ссылкой) и адресную часть (адрес ресурса назначения, на который указывает гиперссылка). В результате, общий алгоритм создания гиперссылки следующий: необходимо выделить фрагмент текста

или графический объект, которые будут использоваться в качестве ссылки, затем выполнить команду **Вставка/Гиперссылка...** или воспользоваться кнопкой панели инструментов **Стандартная** и в диалоговом окне **Добавление гиперссылки (рис.11)** указать ресурс назначения.

Рассмотрим более подробно элементы окна **Добавление гиперссылки**. В поле **Текст** указывается текст, который должен отображаться в качестве гиперссылки (при назначении гиперссылки рисунку этот параметр становится недоступным). Кнопка **Подсказка...** позволяет создать для гиперссылки текстовую подсказку, отображаемую в браузере при наведении курсора мыши на ссылку. В разделе **Связать с** необходимо определить тип создаваемой гиперссылки. Обратите внимание, что ссылка может указывать на любой существующий файл, на определенное место в документе (для чего используются предварительно созданные закладки), на новый, пока еще не существующий документ (FrontPage предложит вам его создать) и на адрес электронной почты. Для создания гиперссылки на какой-либо внешний ресурс сети Интернет адрес этого ресурса (URL) необходимо указать в поле **Адрес**.

Вооружившись нужными знаниями, попробуем применить их на практике. Двойным щелчком мыши откройте в режиме редактирования начальную страницу сайта **index.htm**. Создадим ссылку для перехода к документу **biography.htm**. Для этого выделите в тексте нашей web-страницы слово **"биографией"**, которое будет использоваться в качестве указателя ссылки, и выполните команду **Вставка/Гиперссылка...** или же воспользуйтесь кнопкой панели **Стандартная**. Далее в диалоговом окне **Добавление гиперссылки** (рис.11) проверьте, что в разделе **Связать с** выбран вариант **файлом, веб-страницей**. Обратите внимание, что при этом вам предлагается выбрать требуемый документ из папки сайта **myweb2**, что нам и нужно (для создания гиперссылки на любой другой существующий файл, расположенный вне папки нашего сайта, используется кнопка **Поиск файла**, которая и позволяет выбрать документ назначения). Щелчком мыши выберите документ **biography.htm**. Обратите внимание, что после выбора требуемого файла в поле **Адрес** отобразится относительный путь к указанному файлу. Не забудьте подтвердить создание ссылки щелчком по кнопке **ОК**.

Итак, наша первая ссылка готова. Проверим ее работоспособность. Сохраните изменения начальной страницы сайта (кнопка на панели **Стандартная**) и включите просмотр web-страницы в браузере (кнопка на панели **Стандартная**). Выполните щелчок по созданной ссылке и убедитесь в том, что она приводит нас к странице с краткими биографическими сведениями. Не забудьте после этого завершить работу с браузером и вернуться в Microsoft FrontPage.

Аналогичным образом создайте гиперссылку на документ **hobby.htm**, используя в качестве указателя ссылки слово **"увлечениями"**. Сохраните web-страницу и протестируйте новую ссылку в браузере, после чего вернитесь в Microsoft FrontPage.

Таким нехитрым способом мы связали три web-страницы нашего сайта. А теперь поговорим о других типах создаваемых гиперссылок. Добавим на начальную страницу сайта ссылку на адрес электронной почты, чтобы поддерживать обратную связь с будущими посетителями нашего сайта. Выделите свою фамилию и инициалы (после символа авторского права ©) и снова выполните команду **Вставка/Гиперссылка...** или же воспользуйтесь кнопкой панели **Стандартная**. В диалоговом окне **Добавление гиперссылки** в разделе **Связать с** выберите вариант **электронной почтой** и в поле **Адрес эл. почты** введите свой адрес электронной почты (например, ivanov@mail.ru). Заметьте, что перед адресом электронной почты FrontPage автоматически добавит протокол **mailto** с двоеточием. В поле **Тема** при желании можно указать тему сообщения электронной почты. Подтвердите создание гиперссылки с помощью кнопки **ОК**. Сохраните web-страницу и просмотрите ее в браузере. Выполните щелчок мышью по своей фамилии и убедитесь в том, что при этом будет создано новое почтовое сообщение, а ваш адрес электронной почты автоматически появится в поле **Кому** соответствующего письма. Закройте электронное письмо и завершите работу с браузером.

Рассмотрим, каким образом создаются гиперссылки, указывающие не просто на другой документ, а на определенное место в этом документе. Как уже упоминалось выше, для создания подобной гиперссылки используются предварительно созданные закладки. Вначале в документе назначения с помощью команды **Вставка/Закладка...** требуемое место помечается именной закладкой, а затем создается гиперссылка на данную закладку. Продемонстрируем это на примере. Добавим на начальную страницу сайта ссылку, реализующую переход в начало документа, т.е. к заголовку. Итак, установите текстовый курсор перед заголовком **Личный сайт Фамилия Имя Отчество** и выполните команду **Вставка/Закладка...**, в поле **Имя закладки** введите **head** и щелкните по кнопке **ОК**. Обратите внимание, что после создания закладки в указанном месте документа появится пиктограмма в виде флажка. После этого перед информацией об авторских правах создайте новый текстовый абзац и введите: **Вверх**. А сейчас нам осталось превратить это введенное слово в ссылку на только что созданную закладку **head**. Для этого выделите слово **"Вверх"** и выполните команду **Вставка/Гиперссылка...** или щелкните по кнопке на панели **Стандартная**. В диалоговом окне **Добавление гиперссылки** в разделе **Связать с** выберите вариант **местом в документе**, щелчком мыши выберите закладку **head** и подтвердите создание гиперссылки с помощью кнопки **ОК**. Сохраните web-страницу и просмотрите ее в браузере. Т.к. начальная страница нашего сайта содержит мало информации, вполне вероятно, что для тестирования созданной ссылки вам придется предварительно уменьшить размеры рабочего окна браузера так, чтобы при просмотре последних строк документа заголовок скрывался. Вот тогда наша ссылка покажет себя во всей красе — вместо того чтобы приводить нас к другому документу, она вернет нас в начало нашего же документа. Завершите работу с браузером и вернитесь в Microsoft FrontPage.

В заключение поговорим о том, каким образом создаются гиперссылки на внешние ресурсы сети Интернет. Чтобы продемонстрировать это на примере, создадим новую web-страницу, которая будет содержать список адресов интересных ресурсов Интернета.

Для этого откройте список инструмента панели **Стандартная**, выберите команду **Страница...** и в диалоговом окне **Шаблоны страниц** выберите шаблон **Обычная страница**. Как вы помните, вначале необходимо настроить общие параметры новой страницы, для чего выполните щелчок правой кнопкой мыши в любом месте на странице и в появившемся контекстном меню выберите **Свойства страницы...**. Определите для нового документа в качестве названия текст **Фамилия Имя Отчество | Избранные ссылки**. Задайте цветовое оформление новой страницы в соответствии с оформлением начальной страницы сайта, т.е. определите такой же фоновый рисунок, как и для начальной страницы (фоновый рисунок хранится

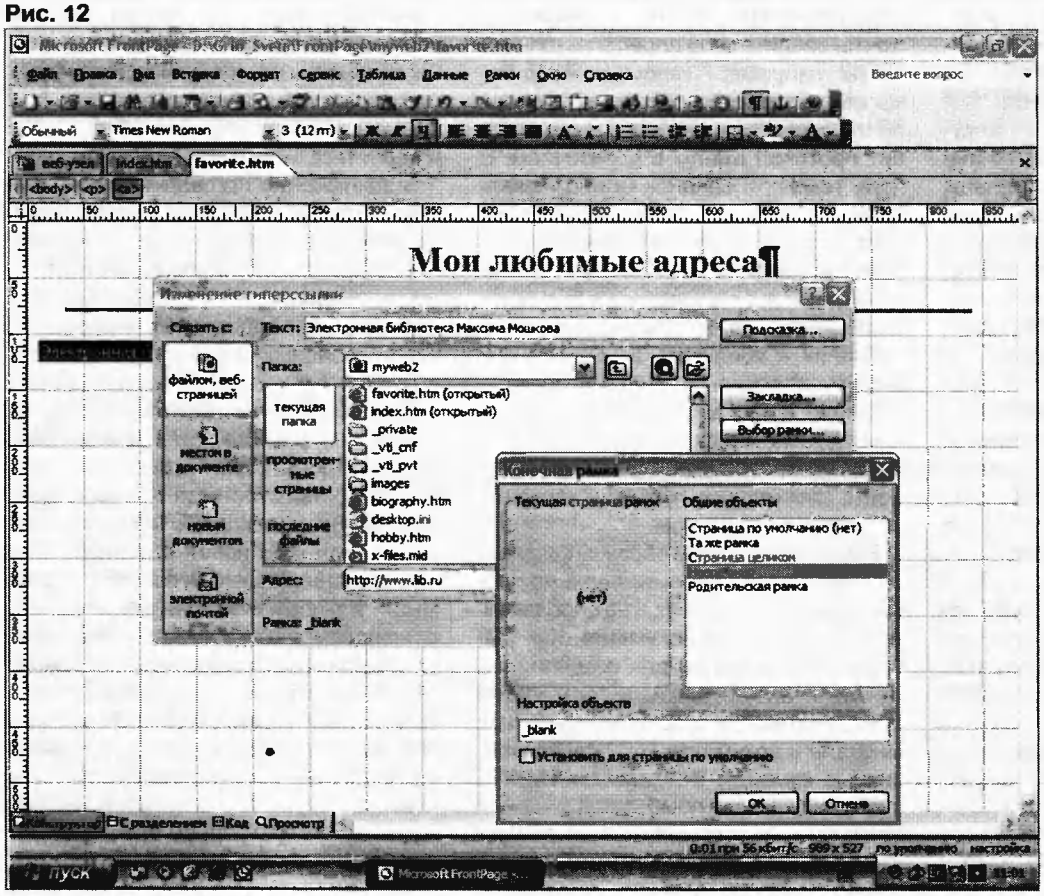
в папке **images** нашего сайта **myweb2**) и те же цвета фона и текста. Определите в качестве языка документа русский, и кириллицу в качестве кодировки, используемой при сохранении web-страницы. Далее на web-страницу поместите заголовок **Мои любимые адреса**. Примените к заголовку страницы стиль **Заголовок 1**, выровняйте его по центру, измените цвет заголовка. Под заголовком вставьте горизонтальную линию и настройте ее параметры.

Вот теперь можно приступать к созданию гиперссылок на ваши любимые интернет-ресурсы. Самый простой способ создания гиперссылки на какой-либо внешний ресурс сети Интернет — это ввод адреса этого ресурса непосредственно в документ. Попробуйте, например, после горизонтальной линии ввести адрес www.lib.ru и нажать на клавишу Enter или просто на пробел. Обратите внимание, что Microsoft FrontPage автоматически создаст соответствующую гиперссылку. Точно так же можно создать гиперссылку и на адрес электронной почты — необходимо лишь ввести нужный адрес в документ. Однако такой способ создания ссылок плох тем, что вашим посетителям будет совершенно непонятно, куда их приведет эта ссылка. Гораздо приятнее видеть в качестве ссылки название ресурса. Организуем ссылку на библиотеку Максима Мошкова по всем правилам. Удалите созданную Microsoft FrontPage ссылку www.lib.ru и введите название ресурса — Электронная библиотека Максима Мошкова. Далее выделите слова "Электронная библиотека Максима Мошкова", выполните команду **Вставка/Гиперссылка...** или щелкните по кнопке на панели **Стандартная** и в диалоговом окне **Добавление гиперссылки** в разделе **Связать с** выберите вариант **файлом, веб-страницей**. После этого в поле **Адрес** введите полный интернет-адрес ресурса — **http://www.lib.ru** — и щелкните по кнопке **ОК**, чтобы завершить создание ссылки.

С помощью инструмента панели **Стандартная** сохраните новую web-страницу под именем **favorite.htm** в папке сайта **myweb2** и, воспользовавшись инструментом , проверьте работоспособность ссылки в браузере. Обратите внимание, что по щелчку мыши новый ресурс будет открываться в том же самом окне браузера, сменяя собой наш сайт. А это означает, что таким образом мы можем растерять своих посетителей. Хотелось бы, чтобы внешние ресурсы, уводящие с нашего сайта, открывались в отдельных окнах.

Чтобы изменить нашу гиперссылку, вернитесь в Microsoft FrontPage и, выполнив щелчок правой кнопкой мыши по ссылке, выберите команду **Свойства гиперссылки...**. На экране появится диалоговое окно **Изменение гиперссылки**, практически полностью идентичное уже знакомому нам окну создания ссылки. Это окно вы можете использовать для изменения адреса ресурса назначения, для удаления гиперссылки (без удаления указателя ссылки) или для настройки параметров существующей ссылки. Нам необходимо, чтобы существующая ссылка открывалась в новом окне браузера. Чтобы это установить, щелкните по кнопке **Выбор рамки...**, в списке **Общие объекты** выберите **Новое окно (рис.12)** и щелкните по кнопке **ОК**. Подтвердите изменение текущей ссылки с помощью кнопки **ОК**. Еще раз сохраните web-страницу и включите просмотр страницы в браузере. Убедитесь в том, что сейчас библиотека Мошкова действительно открывается в отдельном окне. Завершите работу с браузером и, вернувшись в Microsoft FrontPage, аналогичным образом добавьте в документ **favorite.htm** еще несколько ссылок на интересные интернет-ресурсы, настроив для них загрузку в новом окне. Оформите перечень интернет-ресурсов в виде нумерованного или маркированного списка с помощью инструментов и панели **Форматирование**. Не забудьте сохранить изменения в документе и протестировать новые ссылки в браузере. После завершения работы с документом закройте его с помощью команды **Файл/Закрыть** или воспользовавшись кнопкой в заголовке окна документа.

Рис. 12



В заключение, поскольку к нашему сайту добавилась новая страница, хотелось бы в начальную страницу сайта **index.htm** добавить гиперссылку на новый документ. Создайте гиперссылку на документ **favorite.htm**, используя в качестве указателя ссылки текст "список адресов интересных ресурсов Интернета". Сохраните web-страницу и проверьте работоспособность ссылки в браузере.

На этом наше сегодняшнее занятие мы заканчиваем. В следующий раз мы поговорим об использовании графических изображений на web-страницах, а пока не забудьте выполнить

4. Завершение работы с Microsoft Office FrontPage 2003 с помощью команды **Файл/Выход** или кнопки строки заголовка программы.

(Продолжение следует)

MATLAB. ЛИНЕЙНОЕ ПРОГРАММИРОВАНИЕ

Б.КИСЕЛЕВ,
Ю.СИЛКОВИЧ,
г.Минск, БГАТУ.

(Окончание. Начало в №9/2004)

Для читателя, знакомого с линейным программированием, коротко представим полную форму обращения к функции **linprog**:

[x,Sval,exitflag,output,lambda] =
linprog(C,A,b,Aeq,beq,ib,ub,x0,options),

которая позволяет получить информацию для анализа решения и управлять процессом решения. Перечислим лишь не упоминавшиеся выше параметры:

- **ub** — вектор правых границ значений переменных;
- **x0** — вектор, задающий начальные значения переменных, от которых будет производиться поиск решения. Эти начальные значения используются в алгоритме среднемасштабной оптимизации и игнорируются в алгоритме крупномасштабной оптимизации (см. ниже).

Параметр **options** позволяет задавать опции для решения задачи при помощи функции **optimset**, которая имеет вид:

optimset('param1',value1,'param2',value2,...),

где:

- **param1, param2** — имена параметров;
- **valuel, value2** — значения этих параметров.

Рассмотрим возможные параметры.

• Параметр **LargeScale** может иметь значения **'on'** (по умолчанию) или **'off'**. При значении **'on'** используется алгоритм крупномасштабной оптимизации (Large-Scale Algorithm), при значении **'off'** — алгоритм среднемасштабной оптимизации (Medium-Scale Algorithm). Среднемасштабный алгоритм использует метод проекций (разновидность симплексного метода), крупномасштабный алгоритм — LIPSOL (Linear Interior Point Solver). Большинство задач можно решать любым из алгоритмов. Просто соответствующие алгоритмы являются более эффективными по вычислительным затратам для решения задач своего круга. Практически, в большинстве случаев можно обойтись установленным по умолчанию методом Large-Scale.

• Параметр **Display** может иметь значения **'off', 'iter'** или **'final'** (по умолчанию). Если значение **'off'** — предотвращается вывод информации о промежуточных результатах решения, **'iter'** — выводятся результаты о достигнутой точности решения после каждой итерации (используется только для алгоритма Large-Scale), **'final'** — выводится сообщение о достижении (или недостижении) решения.

• Параметр **Diagnostics** может иметь значения **'on'** или **'off'** (по умолчанию) и позволяет выводить диагностическую информацию о решаемой задаче.

• Параметр **TolFun** позволяет задавать точность решения (точность по умолчанию равна 1.0e-8).

• Параметр **MaxIter** задает максимальное количество итераций, после достижения которого прекращается поиск решения (по умолчанию — 85).

Выходные переменные **exitflag, output и lambda** (последние две имеют тип struct) предоставляют информацию о ходе и результатах вычислений.

• **exitflag** — если эта величина положительна, то вычисления завершились нахождением решения, если она равна нулю, то достигнуто предельное число итераций, если данная величина отрицательна, то решение не найдено.

• **output** — информация о результатах оптимизации:

- **output.iterations** — число выполненных итераций;
- **output.cgiterations** — число PCG-итераций (Preconditioned Conjugate Gradients), применимо только для Large-Scale Algorithm;
- **output.algoritm** — используемый алгоритм.

• **lambda** — множители Лагранжа соответственно для различных типов ограничений:

- **lambda.ineqlin** — для неравенств;
- **lambda.eqlin** — для равенств;
- **iambda.upper** — для верхней границы **ub**;
- **lambda.lower** — для нижней границы **lb**.

Попробуем решить ту же задачу при помощи алгоритма средней размерности с выводом диагностических сообщений. Для этого заменим в М-файле обращения к **linprog** на более полные:

```
[x, Sval, exitflag, output, lambda] =
linprog(C,arow,rowsum,acol,colsum,lb,[],[],options)
[x, Sval, exitflag, output, lambda] =
linprog(C,[],[],[arow;acol],[rowsum;colsum],lb,[],[],options)
[x, Sval, exitflag, output, lambda] =
linprog(C,acol,colsum,arow,rowsum,lb,[],[],options)
```

Так как матрица **T** уже введена, выполним только следующие команды:

```
>>options = optimset('LargeScale', 'off', 'Diagnostics', 'on');
>>trzad
```

Получим следующий результат:

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%Diagnostic Information
Number of variables: 6

 Number of linear inequality constraints:    0
 Number of linear equality constraints:      5
 Number of lower bound constraints:          6
 Number of upper bound constraints:          0

Algorithm selected
   medium-scale: active-set

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%% End diagnostic information

The equality constraints are dependent.
The system of equality constraints is consistent. Removing
the following dependent constraints before continuing:
     5

Optimization terminated successfully.
x =
         0
  145.0000
  210.0000
         0
   27.0000
  133.0000

Sval =
  3.9663e+003

exitflag =
     1

output =
      iterations: 2
       algorithm: "medium-scale: activeset"
    cgiterations: []

lambda =
      lower: [6x1 double]
      upper: [6x1 double]
```

```
      eqlin: [5x1 double]
    ineqlin: [0x1 double]

x =
         0  210.0000   27.0000
  145.0000         0  133.0000

Sval =
  3.9663e+003
```

Диагностическая информация содержит: количество переменных — 6; количество ограничений неравенствами — 0, равенствами — 5; нижних границ переменных — 6, верхних — 0; а также используемый алгоритм — medium scale. Далее следует более интересное сообщение:

"Ограничения равенствами являются зависимыми.
Система ограничений равенствами непротиворечива.
Удаляем следующие зависимые ограничения перед продолжением: 5"

Очевидно, что если сложить первые два уравнения нашей системы и вычесть третье и четвертое, то получим пятое уравнение. Поэтому MATLAB удалил зависимое пятое уравнение из дальнейшего рассмотрения.

Выходная информация содержит сообщение об успешной оптимизации, значения переменных и целевой функции, которые совпали со значениями, найденными ранее при помощи алгоритма large scale (в сообщениях он называется lipsol). Кроме того, положительная величина переменной **exitflag** свидетельствует об успешной оптимизации, в структуре **output** отражено, что задача решена алгоритмом medium scale за две итерации.

Для более подробной информации о множителях Лагранжа [2] выведем значения полей структуры **lambda** для имеющихся ограничений:

```
>> lambda.lower
ans =
    2.4000
         0
         0
    6.3000
         0
         0

>> lambda.eqlin
ans =
  -12.0000
   -9.1000
    2.9000
    4.7000
         0
```

Отличные от нуля множители Лагранжа указывают активные ограничения. В нашем случае это нижняя граница первой и четвертой переменной и четыре равенства (зависимое пятое было исключено из системы). Численные их значения позволяют сделать вывод о степени влияния ограничений на значение целевой функции. Например, для нашей задачи наиболее сильное влияние оказывает первое ограничение, поэтому увеличение объема производимой продукции на первом предприятии будет способствовать наибольшему снижению (так как знак отрицательный) общей стоимости перевозок.

Упражнения

1. Завод производит 2 вида изделий — А и В, рынок сбыта которых неограничен. Каждое изделие должно быть обработано на фрезерном, токарном, и шлифовальном станках. На изделие А нужно 0,5 часа фрезерной, 0,4 токарной и 0,2 шлифовальной обработки, на изделие В — соответственно 0,25, 0,3 и 0,4 часа. Время работы фрезерного, токарного и шлифовального станков не может превышать 40, 36 и 36 часов в неделю соответственно. Спрос на продукцию таков, что на каждое изделие А должно приходиться не менее одного изделия В. Требуется определить недельные нормы выпуска изделий А и В, дающие наибольшую прибыль, если прибыль от изделия А составляет 30 ден.ед., от изделия В — 50 ден.ед.

2. Решить транспортную задачу при исходных данных, приведенных в таблице (для любознательных дополнительно предлагается сделать задачу открытой и повторить решение).

	Стоимость перевозок					Производство
	31	32	33	34	35	
П1	5,7	6,8	4,9	8,1	7,6	53
П2	4,2	9,2	6,2	3,5	8,3	95
П3	8,3	5,7	3,8	8,5	4,8	34
П4	5,6	7,1	4,9	5,2	4,7	73
Заказы	37	81	43	29	65	

3. С помощью стандартной программы Windows — **Блокнот** — создать текстовый файл с исходными данными для задачи из 2 и сохранить его под именем **trz.txt**. Соответствующей командой MATLAB считать данные из созданного файла в переменную **T**, решить задачу из 2 и записать результат решения из переменной **x** в файл под именем **trx.txt**. Открыть полученный файл в Блокноте и посмотреть результат.

Вопросы для самопроверки

1. Какова суть задачи линейного программирования?
2. Описать параметры функции **linprog**.
3. Как сформировать параметры функции **linprog** для решения транспортной задачи?
4. Пояснить написанные программы.
5. Как можно создать отдельный файл исходных данных и ввести их в переменную, и как из переменной поместить данные в файл?

Литература

1. Кузнецов А. В. и др. Высшая математика: математическое программирование. — Мн.: Вышэйшая школа, 1994. — 286 с.

2. Сакович В.А. Оптимальное решение экономических задач. — Мн.: Вышэйшая школа, 1982. — 272 с.

3. Ануфриев И.Е. Самоучитель Matlab 5.3/6.x. — СПб.: БХВ-Петербург, 2002. — 736 с.

4. http://www.matlab.ru/news.asp

Рисунок О.Попова

Е.ЛЕБЕДЕВ,
г.Череповец.

BestCrypt

В современном мире информация с каждым годом приобретает все большую и большую ценность. Соотвественно, с увеличением общих обьемов и ценности информации все более и более актуальным становится вопрос ее защиты. В Интернете существует огромное количество сайтов, освещающих темы безопасности (SecurityLab.ru, BugTraq.RU, SecurityFocus.com, Void.ru и др.). А как сохранить свою сокровенную информацию, которая в случае попадания в чужие руки может нанести вам урон? Еще во времена Древней Греции зародились и начали постепенно развиваться методы защиты, подобные современной "криптографии". Спустя несколько тысяч лет криптографические алгоритмы "мутировали" до неузнаваемости, создателями подобных алгоритмов двигала общая цель — создание почти совершенного криптоалгоритма, которому не было бы равных в скорости шифрации данных и, самое главное, в надежности. В настоящее время существует огромное количество сложнейших криптоалгоритмов, однако любой из них, в принципе, можно расшифровать — все зависит только от времени (по ходу статьи мы рассмотрим, как сделать нереальным алгоритм расшифровки нашей информации).

С программой BestCrypt я познакомился совершенно недавно, но, поработав с данным продуктом пару дней, очень сильно к нему привязался. Авторы программы (Jetico, Inc.) предлагают нам "easy-to-use" (легкую в использовании) систему защиты информации. BestCrypt шифрует данные одним из самых "крепких" всемирно известных алгоритмов: Blowfish, Twofish, Rijndael и GOST. Эти алгоритмы имеют ряд особенностей.

Blowfish — это самый быстрый алгоритм. Он был создан Брюсом Шнеером (Bruce Schneier), президентом Counterpane Systems, фирмы, занимающейся безопасностью. Blowfish специально разработан для шифрования информации на 32-разрядных процессорах, он быстрее алгоритмов DES и GOST. Программа BestCrypt использует Blowfish с ключом длиной в 256 битов и 16 циклами.

Дополнительную информацию можно почерпнуть по адресу http://www.counterpane.com/blowfish.html

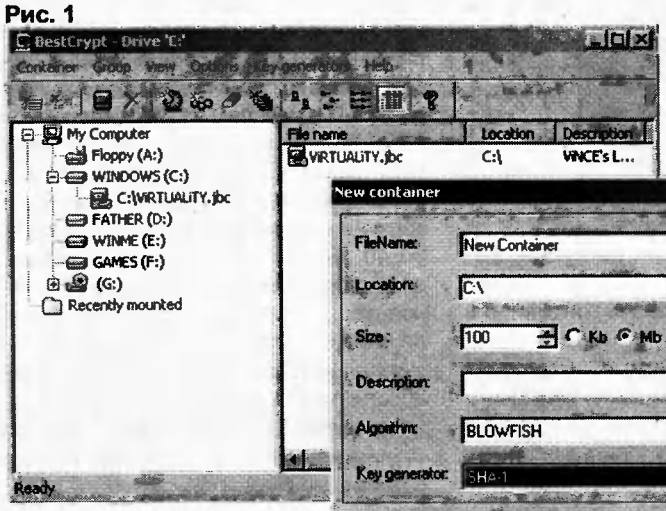
GOST — это правительственный стандарт шифрования информации. Он был разработан во времена СССР. Так же как и Blowfish, использует ключ длиной в 256 битов.

Rijndael — этот алгоритм разработали Джоан Дэемен (Joan Daemen) и Винсент Риджмен (Vincent Rijmen) в Национальном институте стандартов и технологий (http://www.nist.gov). По данному алгоритму можно шифровать данные с ключами длиной 128, 192 или 256 битов. В BestCrypt используется 256-битный ключ. Дополнительную информацию можно получить с сайта одного из разработчиков: http://www.esat.kuleuven.ac.be/~rijmen/rijndael/.

Собственно к делу...

Итак, мы установили программу. Теперь перезагружаем систему, и в области системного трэя (system tray) должна появиться иконка BestCrypt. Кликаем по ней правой клавишей мышки и тем самым вызываем контекстное меню. В меню представлены два пункта — "BestCrypt Control Panel" и "Dismount All BestCrypt Drives". Первый пункт меню вызывает панель управления системой BestCrypt, а второй отключает все монтированные в систему диски (об этом чуть позже). Кликаем по первому пункту, и загружается панель управления (**рис.1**).

Рис. 1



Работа программы основывается на понятии "контейнер" (Container) — пользователь создает файл-контейнер, в который складывает нужные данные (файлы), а BestCrypt шифрует контейнер. Создадим простой контейнер с данными. Для этого кликаем в меню Container—New Container.

Появится форма, показанная на **рис.2**.

Сначала укажем имя файла контейнера (поле "FileName") и место его расположения (поле "Location"). Теперь можно указать размер отводимого под контейнер места (в Кб\Мб\Гб). Заметим, что при установке галочки на "Kb", значение устанавливается в "32 767", при "Mb" — в "4 096", а при "Gb" — в "4". Если попробовать установить "Mb" и прокрутить элемент "SpinEdit" вверх, то значение "4 096" не будет меняться. Сначала я подумал, что количество мегабайтов, отводимых под контейнер, фиксировано особенностями программы, но это не так — мы можем вручную указать в SpinEdit'е нужный нам объем. В поле "Description" можно указать описание к нашему контейнеру. Пункты "Algorithm" и "Key Generator" отвечают за метод шифрования и за алгоритм генерации ключа шифрования. Ставим галочку в пункте "Mount and format now". Здесь мы включаем пункт "Добавить новый диск и отформатировать". Все, настройки сделаны — смело жмем "Create" (Создать). Появится окошко, в котором требуется указать пароль (ключ) для шифрования контейнера (**рис.3**).

Доступ к зашифрованному контейнеру с данными ограничивается паролем, который может состоять из 8 и более символов. После ввода пароля появится форма (**рис.4**), на которой нас попросят ввести случайные символы с клавиатуры для генерации ключа шифрования.

Как только необходимое число символов будет принято, кнопка "OK" станет активной. Жмем

Рис. 2

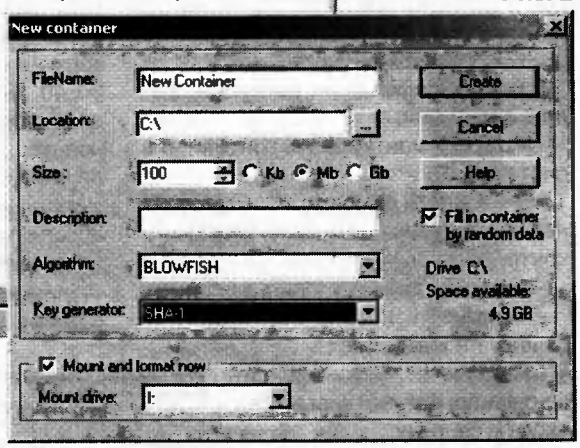
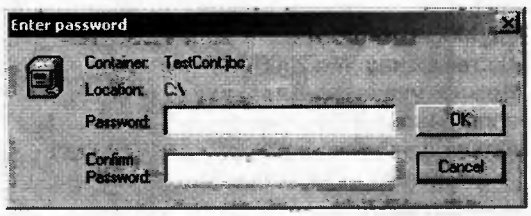
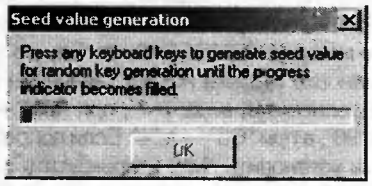


Рис. 3



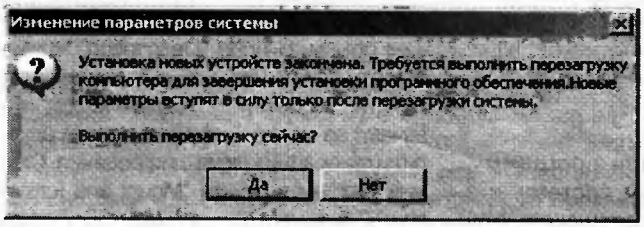
ее, и BestCrypt зашифрует новый контейнер. Операционная система произведет монтирование нового диска в систему

Рис. 4



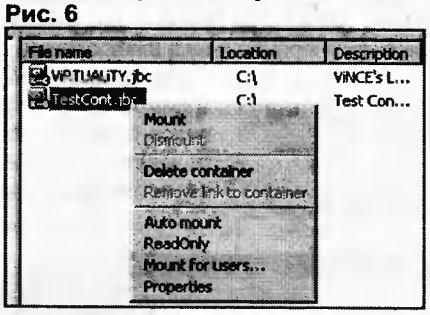
(с той буквой, которую мы выбрали на рис.2), и появится диалоговое окошко, как показано на **рис.5**.

Рис. 5



После перезагрузки в системе новый диск так и не появится! Почему? Да потому что это одна из мер безопасности BestCrypt. Чтобы созданный нами контейнер появился, делаем следующие манипуляции: открываем BestCrypt Control Panel. В правой части панели управления, как показано на **рис.6**, выбираем нужный нам контейнер и жмем на нем правую клавишу мыши.

Рис. 6



В появившемся меню доступны следующие опции:

- Mount — монтировать контейнер в систему;
- Dismount — демонтировать контейнер из системы (без удаления самого контейнера);
- Delete Container — удалить контейнер;
- Remove link to container — удалить ссылку на контейнер;
- Auto mount — монтировать контейнер автоматически при входе в систему;
- ReadOnly — запретить запись информации в контейнер;
- Mount for users... — если система типа Win'2000\XP, то можно указать, для каких пользователей будет монтироваться данный контейнер;
- Properties — свойства данного контейнера.

Рассмотрим пункт "Properties". После клика мышкой появиться окошко, как показано на **рис.7**.

Появившееся окошко можно условно разделить на две части (до "похудения" и после "похудения") — "Текущие параметры контейнера" и "Новые параметры...". В текущих параметрах мы видим имя файла контейнера (FileName), место расположения на диске (Location), алгоритм шифрования (Algorithm), описание (Description), генератор ключа (Key Generator). Для изменения параметров контейнера служат опции:

- Change Name and Location — изменить имя и место расположения;
- Change Algorithm and Password — изменить алгоритм шифрования и пароль;
- Create Hidden Part — создать скрытый раздел;
- Add New Password — добавить пароль;
- Remove Additional Password — убрать дополнительный пароль.

Предельная безопасность

Выше я уже упоминал, что любой из криптоалгоритмов можно расшифровать, и это всего лишь вопрос времени. Но какого времени? Секунды? Минуты? Часа? Суток? Недели? Как известно, чем длиннее ключ шифрования, тем дольше длится процесс дешифровки. Будет ли разница при расшифровке данных, зашифрованных паролем из 8 символов, и данных с паролем из 20 символов?

Конечно, будет! Данные с 20-символьным паролем практически нереально расшифровать, даже если использовать военные суперкомпьютеры. А если взять 30, 40, 50, а то и 60 символов? Процесс дешифровки затянется на пару биллионов лет ;). Тем самым достигается предельная безопасность. Хочу заметить — "предельная", а не "абсолютная", так как абсолютной безопасности не достичь!

Несколько советов для обеспечения максимальной безопасности системы с помощью BestCrypt:

1. В "Options" разрешите отключение монтированных дисков при неактивности системы (мало ли, вы куда-то отойдете!).
2. Там же на закладке "SysTray Shortcut Key" установите комбинацию горячих клавишь для включения BestCrypt Panel.
3. Там же поставьте галочку перед "Disable SysTray Icon" — иконка BestCrypt исчезнет из области часов (SysTray). Вызвать же саму программу вы сможете установленной комбинацией клавиш (см. пункт 2).
4. Там же поставьте галочку перед "Dismount at Log Off" — теперь все монтированные диски будут автоматически отключаться при выходе из системы.
5. Там же уберите галочку перед "Disable Container Guard Utility" — эта опция отключает защиту от ваших контейнеров. Если опция отключена, то посторонние пользователи не смогут удалить ваши контейнеры с информацией (только в текущей операционной системе!!!).
6. При создании нового контейнера вводите пароль длиной не менее 20 символов.

И наконец, рассмотрим несколько примеров, как можно эффективно использовать BestCrypt и защитить свои данные от посторонних глаз.

Вы работаете в MS Word (Excel, Paint, Access, 1C...) с личными (секретными) документами — сохраняйте их на любой из монтированных контейнеров.

По E-mail к вам приходит очень приватная (личная) почта, которую ни в коем случае нельзя читать посторонним — устанавливаете программу чтения почты прямо на монтированный диск.

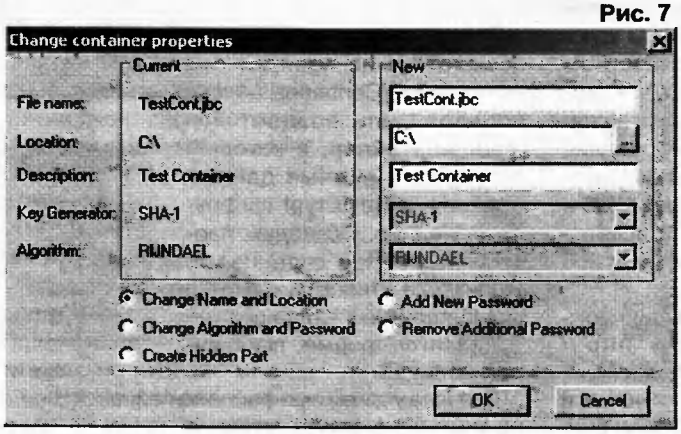
В вашей семье к компьютеру имеют доступ несколько человек — для сохранности данных вы можете хранить их (или их копии) в контейнере.

Заключение

BestCrypt — это простая, но, в то же время, очень мощная и надежная программа, которая защитит вашу информацию от посторонних глаз и тем самым сохранит уйму личного времени и нервов.

В статье была рассмотрена Jetico BestCrypt v7.0, которую вы можете скачать с сайта разработчика — http://www.jetico.com. Программа работает только 30 пробных дней, по истечении которых функция создания новых контейнеров будет заблокирована... это наводит на мысли ;)!

Рис. 7



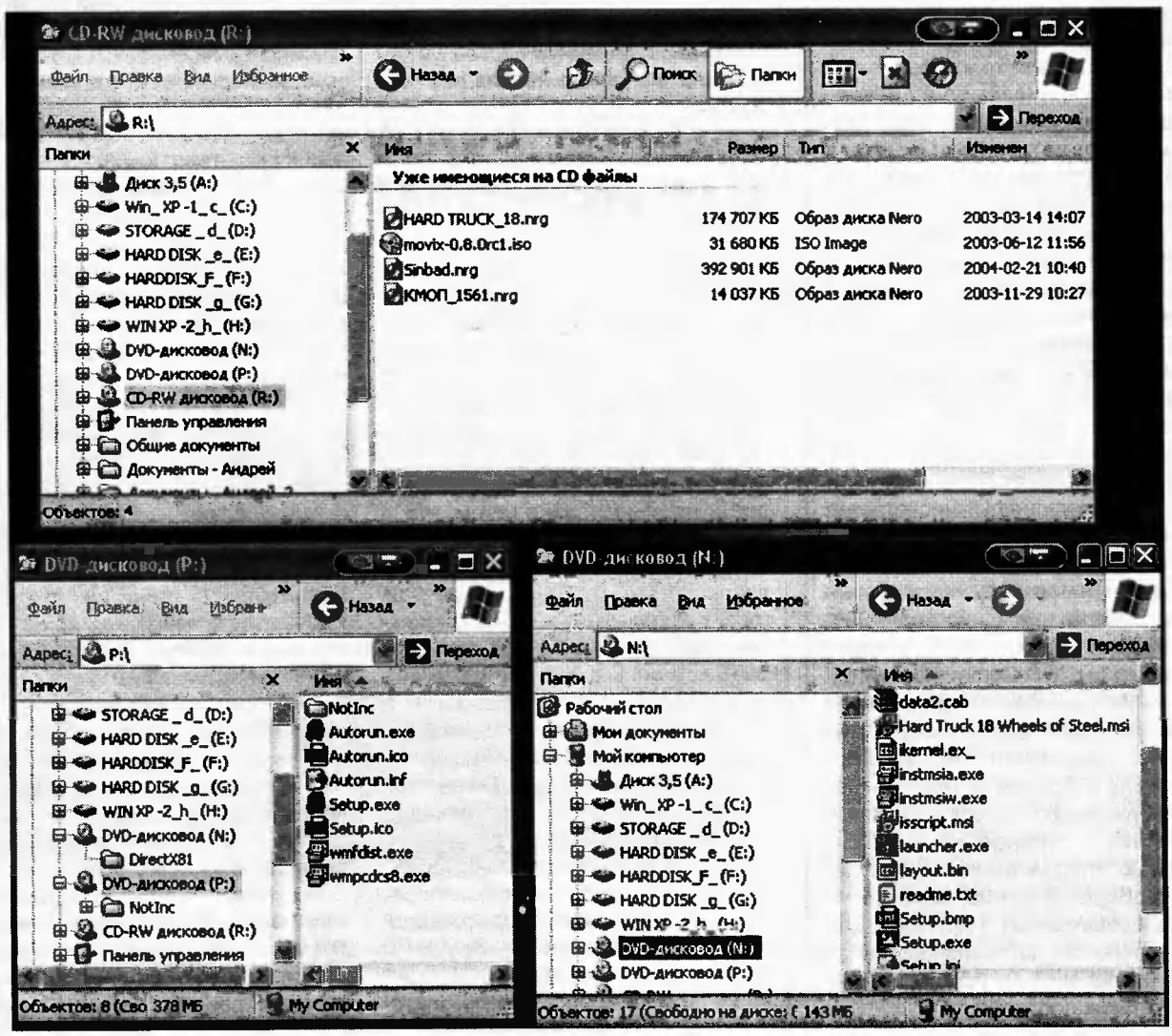
Экономим компакт-диски

А.БУТОВ,
с.Курба,
Ярославской области,
E-mail: andrey-rad@yandex.ru

В настоящее время все чаще можно встретить программы, которые для своей работы требуют наличия в приводе CD-ROM оригинального компакт-диска. Кроме того, обычно требуется создавать резервные копии дистрибутивов различных программ, которые могут устанавливаться только с компакт-диска или с его образа, который лежит на жестком диске и при необходимости "устанавливается" в виртуальный дисковод. Если у вас нет особого желания держать на жестком диске имиджи всех CD, с которыми вам приходится работать лишь изредка, а также нет желания тратить драгоценное время на взлом строптивых дистрибутивов или уже установленных программ, вы можете взять на заметку описанный ниже способ организации резервного копирования данных.

Итак, предположим, у вас имеются два компакт-диска с дистрибутивами игр, которым для установки, запуска или работы требуется наличие в приводе CD-ROM оригинального компакт-диска. Размер имиджа "HARD TRUCK_18.nrg" одной из них составляет 170 Мб, а другой, "Sinbad.nrg" — 383 Мб. Если вы решите сделать на CD резервные копии этих игр, то вовсе не обязательно для каждой из них использовать по отдельному диску CD-R или CD-RW. Оба образа можно скопировать на один компакт-диск, даже не делая его мультисессионным. Для этого файлы имиджей просто копируются на компакт-диск с помощью соответствующей программы для прожига, например, широко известной "Nero Burning ROM", или средствами операционной системы Windows XP. **На рисунке** показано, что на один реальный компакт-диск (дисковод "R"), записано четыре файла имиджей. Один — в общепринятом формате **.iso**, остальные — в формате "Nero" (расширение **.nrg**). При инсталляции программ из пакета "Nero" (http://www.nero.com) в системе было создано два виртуальных дисковода (программа "Nero ImageDrive"), которые предназначены для работы с имиджами CD- и DVD-дисков. На рисунке видно, что имиджи игр, находящиеся на компакт-диске, вставленном в дисковод "R", установлены в виртуальные дисководы "DVD-дисковод (N)" и "DVD-дисковод (P)". Два других файла размером 30,9 Мб и 13,7 Мб — резервные копии данных — записаны на компакт-диск для более полного использования его емкости.

Если заранее известно, что уста-



навливаемая программа будет работать только при наличии оригинального компакт-диска в приводе CD-ROM, то ее желательно устанавливать именно с того дисковода, в который и будет впоследствии устанавливаться имидж оригинального диска. Это правило необходимо соблюдать, потому что часть программ ищут "свой" диск по тому адресу, с которого устанавливались. Другие же программы ищут диск во всех доступных реальных и виртуальных оптических дисководах. Некоторые же, наиболее защищенные игры (видимо, от самих себя), могут устанавливаться и работать только при наличии дистрибутива на оригинальном компакт-диске, установленном в реальный оптический привод. Невозможность обычными средствами сделать работоспособную резервную копию такой программы приводит к ее потере при малейшем повреждении CD-диска, поэтому многие предпочитают такие программы не приобретать.

Таким образом, если несколько дистрибутивов имеют относительно небольшой размер, то их резервные копии можно записать на один компакт-диск и затем, при необходимости, монтировать файлы образа диска в виртуальном оптическом дисководе и обращаться к ним как к настоящему приводу. Различные справочники, энциклопедии, нередко содержащие тысячи и десятки тысяч файлов, также желательно хранить в одном файле именно как имидж CD (DVD), при обращении к ним это не будет ощутимо замедлять работу системы и антивирусных сканера и монитора.

При невозможности приобрести коммерческие полнофункциональные версии пакетов "NERO" и "Clone CD", для работы с CD можно воспользоваться пусть и менее функциональными, но бесплатными программами, например, Small CD-Writer (http://www.avtlab.ru), ISO Commander (http://www.turtleblast.com), Burrrn (http://www.apehaus.com/burrrn), CDBurnerXP Pro 2.1.7 (http://hem.bredband.net/cdburnerxp), Astonsoft DeepBurner (http://www.deepburner.com), CDmage 1.02.1 (http://cdmage.cjb.net), CD Manipulator (http://www.storeroom.info/cdm), IsoBuster 1.5 (http://www.smart-projects.net/isobuster) и другими. Для выполнения всех поставленных задач может потребоваться установить несколько таких программ.

А.ВЕНДИЛОВСКИЙ,
г.Минск.

Надежность и безопасность

Внимание! Перед любыми действиями с операционной системой сделайте аварийные копии файлов на случай непредвиденных ситуаций. Все, что вы будете делать, вы будете делать на свой страх и риск.

Если вы работаете в Интернете, то можете быть уверены, что в настоящее время работаете в самой агрессивной и опасной для пользователя среде. Обычный пользователь начинает это понимать, лишь когда становится слишком поздно, и он теряет результаты своей работы из-за деятельности какого-нибудь вируса или хакера. Статей на эту тему как в журналах, так и в Сети очень много, но часто люди просто не понимают, что их безопасность — это не просто установленный антивирус или фаерволл, а целый комплекс мер и правил. По роду своей работы я провожу в Сети в среднем часов 10 в день, поэтому и хочу поделиться своим опытом. Надеюсь, это будет для вас полезно. Если мы хотим создать более или менее защищенную систему, нужно начинать с самого начала.

Создание Крепости — установка операционной системы

В настоящее время наиболее актуальной ОС считается XP, о ней речь и пойдет. При установке обязательно используйте версию с уже интегрированным первым сервис-паком (SP1), это ***очень*** важно, так как защищенность системы без него — хуже некуда. После инсталляции ОС вы должны установить фаерволл. Я рекомендую Agnitum Outpost 2.1 — очень надежный и дружелюбный, имеющий русский интерфейс и великолепно настраивающийся. Демо-режим работает 30 дней, после чего с вас потребуют оплату. Не скупитесь, стоимость — просто копейки по сравнению с аналогами, всего 499 рос. рублей с лицензией на год. Пожалуйста, не ищите кряки и кейгены — люди, создавшие этот великолепный продукт, просят совсем

Объяви войну шпионам!

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

Наверное, многие пользователи слышали, а может быть, и сталкивались на практике с такими явлениями как компьютерные вирусы и "шпионы". Если первые каким-то образом внешне напоминают о своем присутствии, то вторые ведут себя незаметно и ничем себя не выдают. Большинство пользователей и не подозревают о том, что в их компьютер уже пробрались "незваные гости". Каким же образом появляются эти "шпионы", и чем нам грозит их присутствие в нашем компьютере? Шпионские модули могут быть встроены в программы, имеющие статус adware, и проникают к нам после установки последних. Очень часто шпионские модули содержатся в "звонилках" и "качалках" — программах, осуществляющих дозвон до провайдера и скачивание программ из Интернета.

Есть и другое, более точное определение программ, содержащих "шпионские штучки" — Spyware. Внедрившись, программа-"шпион" без ведома и согласия пользователя может с его компьютера собирать и отправлять по сети Интернет любую конфиденциальную информацию, в том числе имя и географическое местонахождение пользователя, адрес электронной почты, пароли и т.д.

Для борьбы с коварными "шпионами" разработано соответствующее программное обеспечение — так называемые специальные "антишпионские" утилиты. Как правило, все эти нужные и полезные программы для пользователей отнюдь не бесплатны. Но из этого правила есть приятные исключения. Одним из таких исключений является программа SpyBot (http://security.kolla.de). Объем инсталлятора программы составляет 3,5 Мб. Конечно, SpyBot — далеко не единственный "боец" на этом фронте. Кроме него известны и другие программы — Spyware Blaster (http://www.wilderssecurity.com/spywareblaster.html), AdAware и т.д.

Программа SpyBot предназначена для поиска и уничтожения модулей Spyware. По своему принципу действия SpyBot — это "антишпионский" сканер. В поисках "шпионов" и реклам-

немного, уважайте их работу! Для начала, установок по умолчанию Agnitum Outpost будет достаточно, чтобы зайти в Интернет и скачать ***все*** обновления и заплатки с сайта Microsoft. Сделать это несложно — в меню "Пуск" есть режим "Windows Update". Набираемся терпения, т.к. качать придется много и долго. Следующее по важности, что необходимо скачать с того же ресурса — это исправления к IE6 (сервис-пак для браузера).

Не старайтесь зайти в Сеть без включенного фаерволла! Эксперименты показали, что в среднем требуется всего 3 минуты, чтобы вашу машину поразил MSBlast!

После установки необходимых заплаток перейдем к препарированию системы — нам потребуется удалить некоторые программы и остановить сервисы, которые нарушают безопасность системы.

ВНИМАНИЕ! Чтобы уменьшить риск поражения вирусом или атакой, ***вы должны в обязательном порядке отказаться*** от следующих программ:

- Outlook — о печальных особенностях этой программы, способной запускать вредоносные скрипты из электронных писем без желания на то пользователя, не знает разве что ленивый. Поэтому этот почтовый клиент стоит **удалить** и **заменить** на **The Bat** версии **2.x**, который лучше приспособлен к работе с подобными письмами, имеет неплохую систему борьбы со спамом и возможность дополнительно подключать к нему антивирусные модули;

- IE — он и раньше-то не баловал своей надежностью, а после того как часть исходников системы попала в сеть, всплывшие уязвимости ставят большой и жирный крест на всякой мысли о возможности посещения сайтов без шанса поймать вредоносный скрипт… Есть прекрасные альтернативы — **Moziiia**, разработки на этом ядре от независимых производителей, либо **Opera**;

- ICQ — как это ни прискорбно, но вирусы сподобились "употреблять" и эту программу, замените ее на что-нибудь подобное и совместимое, вроде Miranda — благо, альтернатив тут больше чем достаточно;

- Windows Media Piayer — жадная до ресурсов, тяжеловесная прога, постоянно сующаяся в Интернет без ведома пользователя. Она не менее опасна, чем вышеупомянутые. Да и зачем она вам, если есть много других прог, которые намного лучше выполняют свои функции (**Winamp**, **BSPiayer** и др.).

Я не стану утверждать, что рекомендуемые программы абсолютно надежны (абсолютно надежных программ нет!), но в этих было найдено намного меньше "дыр", и меньше всего создано хакерского инструмента. Большинство хакеров (особенно тех "хакеров", что купили на развале диск типа "Все для хакера") будут в первую очередь использовать особенности уязвимости программ, стоявших в системе по умолчанию. Если вы их замените, то обезопасите себя от большинства хакеров-недоучек и многих вирусов, гуляющих в Сети.

Теперь приступим непосредственно к препарированию. В свойствах Проводника идем в *Сервис/Свойства папки/закладка "Вид"/Дополнительные параметры*, снимаем галочку с опции "*Скрывать защищенные системные файлы*" и отмечаем "*Показывать скрытые папки и файлы, скрывать расширения зарегистрированных типов файлов*". Открываем каталог с XP и в подкаталоге \inf открываем в Блокноте файл sysoc.inf. Без всякой жалости уничтожаем слово HIDE, не забыв оставить запятые. После сохранения файла, в апплете "Установка и удаление программ"

ных модулей он сканирует жесткий диск и системный реестр. Умеет SpyBot находить и эксплоиты — программы, использующие "дыры" в операционной системе и получающие доступ к другим программам и данным. Функция монитора в ней не предусмотрена. Рабочее окно SpyBot показано на **рисунке**.

Сканирование жесткого диска и системного реестра программой происходит с поразительной быстротой. Всего лишь 40 с понадобилось SpyBot'у, для того чтобы "прочесать" 60-Гб жесткий диск, и, к моему удивлению, обнаружить на нем целую дюжину "шпионов". После обнаружения "шпионских" модулей SpyBot обезвреживает их. Процесс обезвреживания "шпионов" происходит корректно: сначала обнаруженные файлы и реестровые записи помещаются в zip-архивы. Делается это с той целью, чтобы при необходимости можно было восстановить удаленное, так как не исключено, что программа, чей модуль был удален, может не запуститься. И только потом, спустя некоторое время, убедившись в том, что все программы функционируют нормально, удаление модулей можно произвести физически.

Работать с программой SpyBot легко и просто. Дополнительное удобство в работе создает включение поддержки русскоязычного интерфейса. SpyBot может работать в двух режимах — упрощенном (Easy Mode) и расширенном (Advanced Mode). Первый режим удобен для тех пользователей, кто не любит возиться с многочисленными настройками, а второй — наоборот, для тех, кому интересно получить доступ к дополнительным опциям программы. В программе имеется встроенная функция обновления. С ее помощью через Интернет можно пополнить и обновить базу распознаваемых в лицо "шпионов".

Имеет смысл периодически использовать SpyBot после установки операционной системы и различных программ. За все время "обкатки" никаких претензий и замечаний по работе с этим "антишпионским" сканером у меня не возникло. Тестирование проводилось в операционной системе Windows XP Home Edition (русская версия).

появится возможность удалить Messenger и Outlook (если вам не нравятся игры, их тоже можно ликвидировать).

Далее идем в *Пуск/Программы/Администрирование/Службы*. Здесь вы увидите список служб и сервисов, уже запущенных на вашем компьютере. Некоторые из них имеют недочеты, которыми легко воспользуется вредитель, да и толку от них на обычной машине пользователя нет никакого, поэтому находим и отключаем следующие сервисы:

- веб-клиент (WebClient);
- вторичный вход в систему (Secondary Logon);
- диспетчер автоподключений удаленного доступа (Remote Access Auto Connection Manager;
- диспетчер отгрузки (Upload Manager);
- диспетчер подключений удаленного доступа (Remote Access Connection Manager);
- диспетчер сеанса справки для удаленного рабочего стола (Remote Desktop Help Session Manager);
- доступ к HID-устройствам (Human Interface Device Access);
- маршрутизацию и удаленный доступ (Routing and Remote Access);
- модуль поддержки NetBIOS через TCP/IP (TCP/IP NetBIOS Helper Service);
- обозреватель компьютеров (Computer Browser);
- оповещатель (Alerter);
- сетевой вход в систему (Net Logon);
- сетевые подключения (Network Connections);
- сервер папки обмена (ClipBook);
- службу индексирования (Indexing Service);
- службу сообщений (Messenger);
- службы терминалов (Terminal Services);
- справку и поддержку (Help and Support);
- NetMeeting Remote Desktop Sharing (NetMeeting Remote Desktop Sharing);
- удаленный реестр (Remote Registry).

Если ваш компьютер дома или на работе стоит в гордом одиночестве, то без всего этого "богатства" вы вполне сможете обойтись, заодно оценив, насколько быстрее стала работать ваша машина.

Защитный ров и "Стражи" — защитные программы

Теперь необходимо установить еще две жизненно важные программы — Антивирус и Убийцу Троянов (червей).

Вы можете задать совершенно адекватный вопрос: "Зачем нам УЧ, если есть Антивирус?". Но, как показывает практика (да и тесты), именно черви являются слабым местом современных антивирусных программ, как по возможностям эвристического анализа, так и по качеству уничтожения и выходу обновлений базы. Зато специализированный Убийца прекрасно справляется со своей задачей, работая на пару с антивирусом и фаерволлом. На роль Антивируса я без колебаний рекомендую разработки Касперского (благо, лицензионную официальную месячную версию бесплатно распространяют с любым компьютерным изданием, где прилагается компакт-диск). А на роль Убийцы Троянов — Tau, опять от компании Agnitum (кстати, если вы решите сразу купить комплект Outpost 2 + Bat 2 + Tau, компании, торгующие софтом, обычно дают очень хорошую скидку!).

Осталось поставить последнюю линию обороны — борца с "официальными" шпионами, которые попадают на наши компьютеры вместе с легальным софтом от самих разработчиков — это Ad-Aware версии 6 и выше. Эту программу нужно будет обязательно запустить и проверить систему ***после установки всего необходимого софта, а также после инсталляции любой новой программы!***

Несколько дополнительных рекомендаций. Практически каждый использует на своей машине MS Office для работы с текстом, электронными таблицами и пр. Так вот, после инсталляции этого пакета его ***обязательно*** необходимо пропатчить! Это легко можно сделать или с помощью автоапдейта, или вручную на сайте Microsoft. Особенно это важно для владельцев версий 2000, XP, 2003.

Правила безопасности

Однако, даже если вы провели все описанные выше мероприятия, почивать на лаврах успокоенности категорически воспрещается, т.к. правильный режим безопасности — это не только установленные защитные системы, но и правильное поведение пользователя. В 80% случаев именно человеческий фактор становился основной "брешью" для создания неблагоприятной ситуации. Поэтому есть правила, которые нужно соблюдать жестко и точно:

- два раза в неделю делать автоматический апгрейд всех защитных программ (антивируса, фаерволла, УТ, Adware) для обновления баз вирусов и троянов;
- один раз в неделю проверять наличие заплаток для операционной системы и офиса, и при их наличии немедленно устанавливать (особенно из категории "критические обновления");
- в случае возникновения очередной вирусной эпидемии, делать это ***каждый*** день;
- откройте почтовый ящик на сервере, где имеются хорошие фильтры против спама и возможность антивирусной проверки, к примеру, как на Rambler.ru;
- прежде чем скачать письма к себе на компьютер, воспользуйтесь функцией The Bat по закачке заголовков. Выбирайте только те письма, в которых уверены абсолютно, остальные уничтожайте прямо на сервере. Никогда не открывайте письма от незнакомых корреспондентов и, тем более, не запускайте никаких вложенных в письмо файлов! Если это так необходимо, проверьте сначала письмо антивирусом на сервере, а потом ***всем*** комплексом программ на своей машине. При малейшем подозрении — уничтожить!
- будьте всегда настороже по отношению к программам, устанавливаемым на свой компьютер, так как большинство этих программ имеет несколько контрафактное происхождение. Всегда нужно быть готовым получить вместе с софтом бесплатное приложение в виде вируса или троянца;
- не качайте из Интернета что попало и откуда попало — только с сайтов, которым можно доверять. Каждый закаченный файл должен быть проверен всем комплексом защитных программ;
- не отключайте резидентные сканеры антивирусных программ, особенно, когда находитесь в Интернете;
- в установках фаерволла сделайте активными опции обязательного запроса на создание cookie, запуск ActiveX-элементов и java-аплетов. Прежде чем разрешить их выполнение, трижды подумайте, нужно ли вам это;
- в фаерволле, в разделе допуска программ в Сеть, сократите до минимума список разрешенных. Оставьте лишь те, без которых работать в Сети вы не сможете, все остальные — запретить.

Я вовсе не стану вас заверять, что, выполнив все данные в этой статье рекомендации, вы будете в абсолютной безопасности. И не верьте никому, кто вам ее будет обещать — ее просто не существует. Однако сложившаяся в настоящий момент ситуация такова, что для человека, не соблюдающего эти правила, вопрос: "будет ли компьютер заражен вредоносной программой", уже не стоит. На повестке дня вопрос: "Как скоро это произойдет?". Несмотря на свою простоту, эти нехитрые правила могут уберечь вас в 9 случаях из 10, что тоже совсем неплохо.

Что делать, если вражеский вирус прорвался на вашу машину, если Windows "рухнула" из-за некорректной работы драйверов или программ? Как восстановить работу системы за пять минут, не занимаясь полной переустановкой операционки и всех программ? Читайте в следующей статье!

Интернет-калейдоскоп, freeware #11

С.РЮМИК,
г.Чернигов,
DEON VAN DER WESTHUYSEN,
г.Кейптаун, ЮАР,
E-mail: ppjoy@inbox.lv

— Какой ваш родной язык?
— Конечно же, C++!
Deon van der Westhuysen

В Интернете существует немало интернациональных проектов, поддерживаемых энтузиастами из разных стран мира. В их основе обычно лежит одна программа, вокруг которой виртуально объединяются пользователи. Если программа интересная и полезная, то она быстро находит свою аудиторию. Отклики, замечания и пожелания создают эффективную обратную читательскую связь. Со временем появляются новые версии программы, в которых расширяются сервисные функции, добавляются новые свойства и возможности.

ПРЕДСТАВЛЯЕМ НАШЕГО НОВОГО АВТОРА.

Деон ван дер Вестхьюзен — профессиональный программист из г.Кейптауна, "морских ворот" Южно-Африканской Республики. Сопровождает и разрабатывает сложные программные проекты, которые внедряются в Австралии, Европе, США. Хобби — гоночные автомобили и Интернет.

Удачным примером такого проекта является программа PPJoy (автор — Deon van der Westhuysen, http://www.geocities.com/deonvdw/Docs/PPJoyMain.htm). Это универсальный системный драйвер для Windows 2000/XP, позволяющий подключать джойстики разных моделей к LPT-порту IBM PC.

Поддерживается более 30 типов внешних устройств, в том числе джойстики от Atari, Commodore, ZX-Spectrum, PlayStation, Dendy (NES), SuperNES, SEGA Mega Drive-II (Genesis), Gameboy. Кроме того, можно использовать Linux-джойстики, LPT-тастатуры, интерфейс радиоуправления авиамоделями, создавать виртуальные клавиатурные джойстики и даже имитировать джойстики при помощи "мыши". Для Windows все они представляются в виде обычных игровых устройств и без ограничения могут применяться в играх и эмуляторах.

Проект PPJoy (Parallel Port Joypad) был начат в 2002 г. К тому времени повсеместно использовалась программа DPP5 (Direct PadPro 5.0, автор — Earle F. Philhower III, ftp://ftp.radio.ru/pub/2003/03/dpp/dpadpr50.zip, 86 Кб). Она выполняла аналогичные функции, но для узкого круга джойстиков, с аппаратными ограничениями и только под Windows 95/98.

Рис. 1

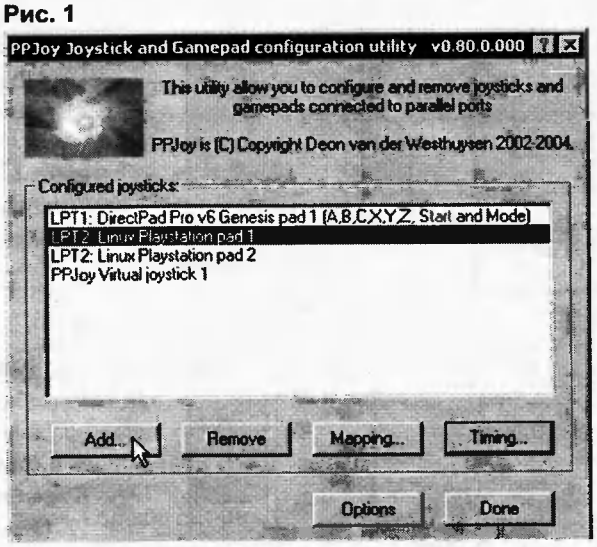
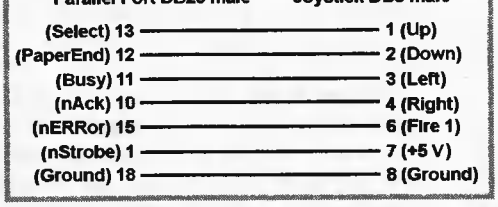


Рис. 2



PPJoy пришел на смену DPP5, имея качественно новую основу. Философия проекта — поддержка разноплановых периферийных устройств, которые могут подключаться к LPT-порту компьютера напрямую или через несложный адаптер. Джойстикам, без дела пылящимся на полке, дается "вторая жизнь", а вновь разрабатываемым интерфейсам, например, от авиасимуляторов — удобное управление.

Загрузка PPJoy осуществляется с сайта SimTel по ссылке Donwload на странице http://www.geocities.com/deonvdw/Docs/PPJoyMain.htm (версия 0.80, PPJoySetup.zip, 1,72 Мб). Инсталляция программы не вызывает затруднений, главное, чтобы компьютер находился в режиме "Администратор Windows 2000/XP" (**рис.1**). Электрические схемы адаптеров приведены в файле помощи, входящем в инсталляционный пакет. Там же указана последовательность установки драйвера, даются подробные подсказки, скриншоты экранов, так что "заблудиться" трудно.

Для пользователей стран СНГ наибольший интерес представляют адаптеры джойстиков от ZX-Spectrum (**рис.2**), Dendy (**рис.3**), SEGA Mega Drive-II (**рис.4, 5**), PlayStation (**рис.6**). Для справки, адаптер — это устройство, подключаемое между LPT-портом компьютера и разъемом реального джойстика. Длина соединительного кабеля — 1...1,5 м, детали адаптера обычно размещают внутри корпуса LPT-разъема.

После установки джойстика в систему, проводится его калибровка, в ходе которой по подсказкам необходимо перемещать его рукоятку "влево-вправо-вверх-вниз" и нажимать соответствующие кнопки. Важное отличие PPJoy от аналогичных программ — возможность подбора времени задержки управляющих сигналов, что пригодится при использовании нефирменных джойстиков или при их неустойчивой работе.

Рис. 3

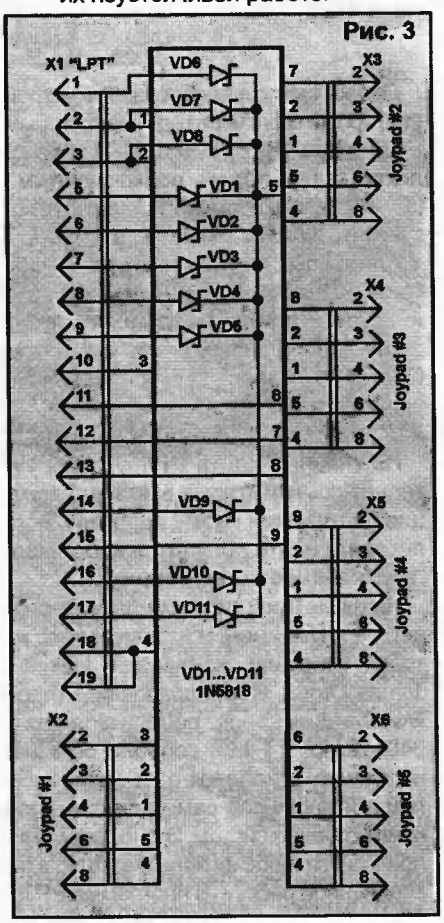


Рис. 4

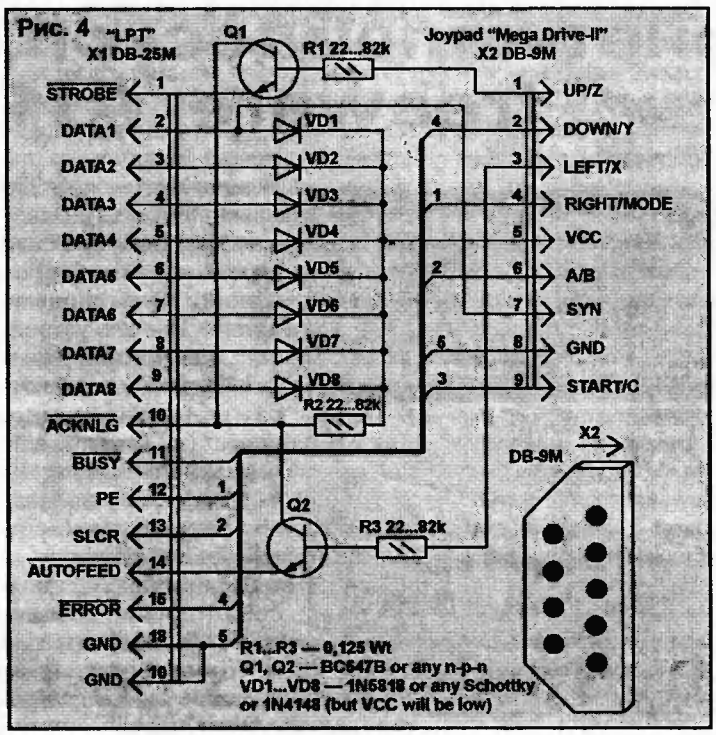


Рис. 5

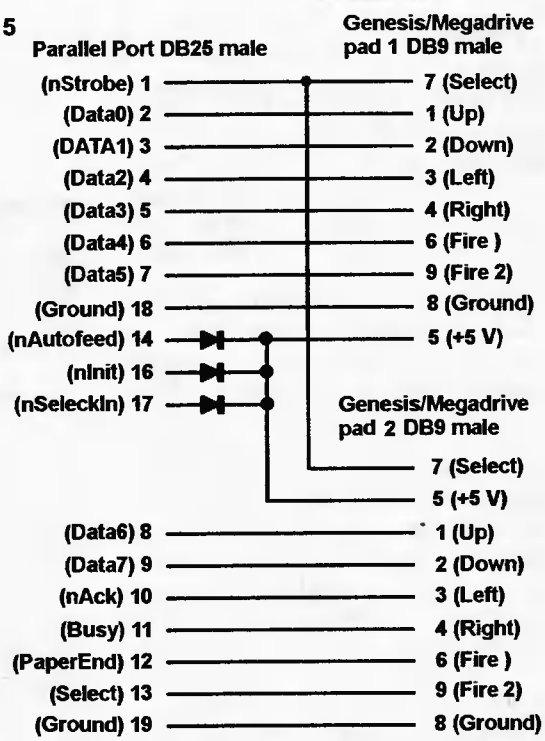


Рис. 6

DirectPad Pro PCX Interfase

pin numbers, looking at the front of the pad (male) connector

Controller	Parallel Port Pad 1	Pad 2
1 (Dat)	10	13
2 (Cmd)	2	
3 (+7V)	+7,6V	
4 (Gnd)	18	
5 (+3V)	+3,5V	
6 (Sel)	3	
7 (Clk)	4	
8	-	
9 (Ack)	12	15

Опробовать откалиброванное устройство проще всего через одну из программ-эмуляторов [1]. Для устранения конфликтов с принтером, подключенным к LPT-порту, в программе PPJoy следует установить флажок в меню Port Options.

Адаптеры, применяемые в PPJoy, были разработаны разными авторами и поставлялись каждый со своим драйвером. Однако теперь они не требуются, поскольку всеми процессами управляет PPJoy. Получается заметная экономия времени и ресурсов.

На главном сайте PPJoy даны Интернет-ссылки, откуда взята информация по тому или иному адаптеру. Если посетить их, то получится небольшое "кругосветное путешествие".

Hardware Book (http://www.hardwarebook.net) — таким именем назван проект, который основали скандинавы Joakim & Tomas Orgen. В их справочнике (http://www.hardwarebook. net/download/hwb-040412.zip, 1,1 Мб) собрана большая коллекция цоколевок разъемов, кабелей, интерфейсов самых экзотических компьютерных систем. Эта информация пригодится любому пользователю в повседневной работе. В программе PPJoy из Hardware Book использованы данные о "цифровых джойстиках", подобных Atari, ZX-Spectrum.

NTPad (http://www.ntpad.com.ar) — сайт аргентинского программиста Ariel Scarpinelli, который разработал драйвер NTPad для Sega-джойстика (рис.5) под Windows XP (http://ntpad.yi.org/dist/ntpadxp2.01.exe, 173 Кб). По сравнению с модернизированным адаптером DPP6 (Direct PadPro-6.0, рис.4, ftp.radio.ru/pub/2003/12/dpp6/dpadpr60.zip, 301 Кб), добавлена возможность управлять от одного параллельного порта сразу двумя джойстиками. "Плата за удовольствие" — LPT-порт должен поддерживать двунаправленный режим ECP/EPP. Для DPP6 достаточно и простого режима SPP, а второй джойстик подключается к порту LPT2. Кроме того, адаптер DPP6 имеет защитные диоды от "тиристорного пробоя" КМОП-структур внутри джойстика.

Авиасимулятор FMS (Flying-Model-Simulator) — программа под таким названием разработана в Германии. Авторы — Roman & Michael Möller (русскоязычный сайт — http://fms.rcdesign.ru). В отличие от фирменных самолетных и вертолетных симуляторов, программа FMS (версия 2.0 alpha 8.1, http://fms.rcdesign.ru/download/?path=fms2alpha81.exe, 4,7 Мб) абсолютно бесплатна, и при этом не уступает им в качестве. Кроме того, поддерживается режим радиоуправления от **переносных** передатчиков авиамоделистов.

Зачем это нужно? Представим ситуацию, когда в авиакружке построена очередная радиоуправляемая модель самолета. Чтобы первый "боевой" вылет не закончился фатальной неудачей, необходимо попрактиковаться в управлении моделью. Для этих целей и предназначен авиасимулятор FMS, позволяющий не только задавать аэродинамические свойства летательного аппарата и форму ландшафта, но и управлять моделью с помощью реального пульта.

Всем начинающим авиамоделистам желательно пройти курс обучения на компьютерном симуляторе, иначе модель, в которую вложено столько труда и средств, может сразу превратиться в груду железа. Остальным пользователям авиасимулятор предоставит уникальную возможность приобщиться к авиамодельному спорту, освоить фигуры высшего пилотажа, оценить красоту полета планера. Для удобства можно создать виртуальный клавишный джойстик PPJoy, настроив его по своему вкусу.

Фирменные и самодельные передатчики, используемые в радиоуправлении, как правило, имеют внешний разъем "Тренер—Ученик". К нему через адаптер, содержащий несколько радиоэлементов, можно подключить LPT- или COM-порт компьютера. Все остальное сделает программа PPJoy, которая трансформирует получаемые от передатчика сигналы в форму, понятную авиасимулятору FMS.

Ключевые слова для поиска: PPJoy, Hardware Book, NTPad, авиасимулятор FMS.

Литература

1. Рюмик С. Интернет-калейдоскоп, freeware #4. — Радиомир. Ваш компьютер, 2004, №3, С.18.

В.КУЗНЕЦОВ,
г.Минск, БГУИР, ф-т ИТиУ, 4-й курс,
E-mail: rea1@tut.by

Бот для IRC-сетей

Введение

В настоящее время индустрия компьютерных сетей переживает один из пиков своего развития. У нее очень долгая история, со своими трудностями и проблемами. Одной из проблем всегда являлась организация общения между пользователями. Эта проблема решалась разными способами, например, каждый компьютер "кидал" сообщение в сеть broadcast'ом (т.е. всем компьютерам), все компьютеры просматривали, кому предназначено это сообщение, и те сообщения, которые им не предназначены, отсеивались. Следует заметить, что такие системы существуют и в настоящее время, в некоторых случаях по-другому нельзя. Следующим шагом было создание виртуального пространства для общения с одним главным сервером, которому посылались все сообщения, а он уже решал, кому именно какое направить. Со временем для большей надежности количество серверов увеличили.

Сеть *IRC* как раз и является такой сетью. Эта сеть — виртуальное место встречи, где люди со всего мира могут встретиться и поговорить. Стандарт *IRC* (*Internet Relay Chat*) первоначально был написан *Jarkko Oikarinen* в 1988 г. Начало работе сети было положено в Финляндии, а дальше она получила широкое развитие и в остальных странах. *IRC* — многопользовательская система для беседы, в которой люди встречаются в "каналах", чтобы поговорить в группах или частным образом. Ограничения на количество людей или каналов зависит только от настройки конкретного сервера. Сервер хранит информацию о каналах и людях в *IRC*, а также другую информацию, например ту, которая отвечает за маршрутизацию ваших сообщений другим пользователям. Сама сеть *IRC* состоит из многочисленных серверов, которые подключены друг к другу.

Для работы с этой сетью существует несколько приложений. Наибольшее распространение получила программа ***mIRC*** — коллективный клиент беседы для *IRC*-сетей под ОС Windows, которая обеспечивает дружественный интерфейс с сетью *IRC*. Днем рождения ***mIRC*** можно считать 28 февраля 1995 г., т.е. уже больше 9 лет она популярна среди пользователей. Начиная с версии 6.1, автор (*Khaled Mardam-Bey*), использовал Visual C++ .NET 7.0, чтобы скомпилировать ***mIRC***.

У этого *IRC*-клиента очень широкие возможности:

- мощная динамическая справка;
- полностью перестраиваемые цвета, фон, шрифты;
- возможность повесить на клавишу какую-нибудь команду;
- и много-много всего другого...

Как известно, в общежитиях БГУИР очень хорошо развита компьютерная сеть, на данный момент численность компьютеров только в общежитии №1 составляет более 300. Постепенно к общению в нашей сети стали присоединяться пользователи из других ВУЗов, пользователи домашних сетей и те, кто просто хочет побеседовать.

Сейчас в нашем чате около 1000 пользователей, 400 каналов и 40 серверов. Но сеть быстро развивается, и количество пользователей будет возрастать. Все, что нужно для общения в сети — это подключиться к серверу, зайти на канал и говорить-говорить-говорить...

Что такое "бот"

Каналом является место в *IRC*, где происходит групповая беседа. Каждый пользователь может создать свой собственный канал по своей теме. Например, существуют каналы, где общаются люди из одного города, с одного фалультета, специальности, потока и т.д.

Кроме обычных пользователей, в сети существуют "боты". **Бот** (bot) — это робот, отдельный сетевой клиент, в котором все прогоняется согласно сценарию. Некоторые боты приносят дополнительную пользу, работая как файл- или инфо-серверы. Некоторые могут даже развлечь "странными" или "мозговыми" играми.

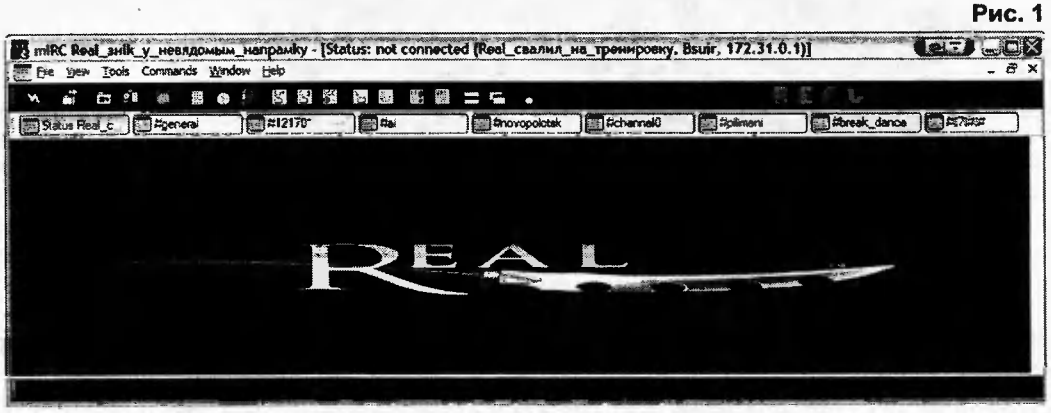
В созданном мною боте имеется несколько функций, основные из которых:

- слежение за выражениями (чтобы не было нецензурных высказываний);
- слежение за тем, кто на канале имеет какие возможности;
- выдача необходимой информации в ответ на запросы пользователей (расписание занятий, списки групп, полное ФИО преподавателя, время, дата, погода и т.д.);
- поддержание беседы и развлечение пользователей за счет ответов на некоторые фразы;
- удаленное управление ботом.

Интерфейс

На **рис.1** показано, как у меня выглядит *IRC*-клиент ***mIRC*** (у других пользователей он может выглядеть иначе, т.к. эта программа предоставляет широкие возможности по изменению интерфейса).

Рис. 1



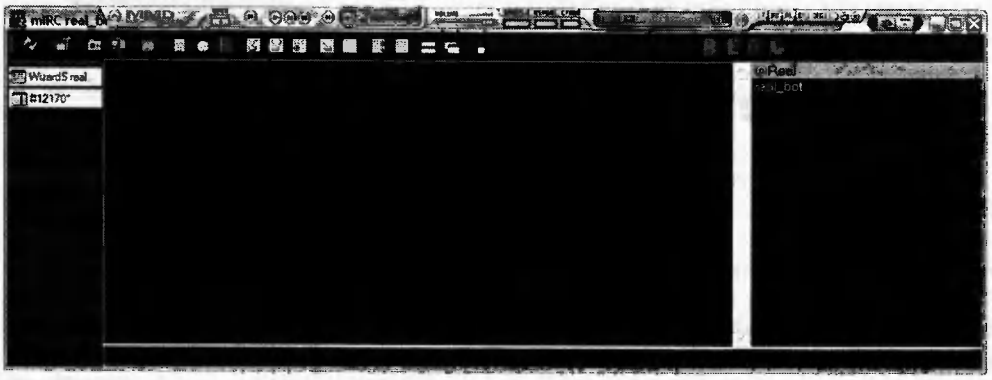
Клиент, запущенный для созданного бота, выглядит, как на **рис.2**.

Самое большое окно — это область, где происходит беседа, а область справа представляет собой список всех тех, кто сейчас присутствует на канале. Слева (не обязательно) находится список всех каналов, в которых находится в данный момент пользователь.

Как пользоваться ботом

Для того чтобы получить от бота какую-нибудь информацию, необходимо набрать специальную команду (все команды начинаются с восклицатель-

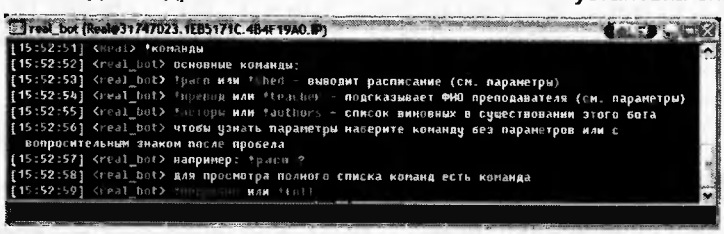
Рис. 2



ного знака). Для удобства все команды дублируются на русском и английском языках. Их можно набирать прямо в канале, а можно — лично боту, чтобы никого не отвлекать. Чтобы не засорять канал сообщениями, бот отвечает на все команды лично тому, кто произвел запрос (в приват).

При входе кого-нибудь на канал, ему автоматически посылается сообщение о том, что на канале существует бот, и приводятся несколько основных команд. Одной из них является команда "!инфо. Сведения об основных командах можно получить, набрав "!команды" (**рис.3**). Подробнее про реализованные на данный момент команды можно узнать с помощью команды "!подробно".

Рис. 3



Кроме этих основных команд, существует еще несколько, которые недоступны обычным пользователям. Они служат для удаленного управления ботом. Количество доступных команд постепенно возрастает, у бота появляются новые возможности.

Настройки клиента позволяют расположить окна приложения поверх всех окон, т.е. можно, например, набирать текст статьи и одновременно наблюдать на своем экране расписание занятий на среду второй недели.

Кроме реагирования на команды, бот реагирует на несколько слов и на кличку "голый", за счет этого с ним можно попробовать завести простенький диалог.

Кроме того, бот следит за тем, у кого какой уровень на канале. Например, поскольку он является главным, у него больше всего возможностей, у обычных пользователей доступ к настройкам канала закрыт. Если в канал зайдет старота одной из групп, то ему предоставляется возможность изменять заголовок канала, например, если надо сделать какое-то срочное сообщение.

Как это все устроено

В рассматриваемом случае бот представляет собой отдельно запущенный клиент ***mIRC***, реализованный с помощью собственного мощного скриптового языка.

Существуют обработчики на различные действия, например, на присоединение кого-нибудь к каналу, на сказанную фразу и т.д. Всего их около 70, в данном случае использовались только 8 из них:

- ***on START*** — установка значений по умолчанию при включении приложения;
- ***on CONNECT*** — действия при соединении с сервером (идентификация, захождение на канал и т.д.);
- ***on JOIN*** — действия на определенном канале при входе на него какого-нибудь пользователя;
- ***on OP*** — реакция на присвоение статуса оператора кому-либо (таковых не должно быть, кроме самого бота);
- ***on HELP*** — то же самое, только более низкого статуса — полуоператора (такой статус позволен только старостам групп);
- ***on DEOP*** — реакция на отбирание у кого-либо статуса оператора (необходимо для того, чтобы бот не лишили статуса оператора);
- ***on TEXT*** — отслеживание в каналах и приватах различных сообщений пользователей;
- ***on NOTICE*** — то же самое, только от третьего лица.

Возможно, в будущем количество обрабатываемых секций будет увеличено.

Команды, на которые реагирует бот, реализованы через обработчик ***on TEXT***. В зависимости от того, какие слова встречаются в сообщениях, производятся определенные действия, например, бот посылает в приват несколько строчек из некоторого файла.

Например, в результате выполнения этих строк

```
on 1:TEXT:*:*: {
if ((!фраза == $$1) || (!phrase == $$1)) {
  play -r $nick "Real\фразы.txt"
 }
}
```

как только бот встретит где-нибудь лексему *!фраза* или *!phrase*, пользователю с ником *$nick* проиграется случайная строка (за случайность отвечает флаг *-r* функции *play*) из файла *"Real\фразы.txt"*. Похожим способом реализованы и остальные команды.

Некоторые боты берут данные о погоде из Internet'а. Но на мирковских скриптах это реализовать сложно, и есть проблемы с Internet'ом, поэтому в этом боте все было реализовано гораздо проще — раз в сутки бот звонит в приват тем ботам, которые берут погоду в Internet'е, и записывает ее себе в файл. Когда пользователь вводит запрос на погоду, то бот просто проигрывает ему этот файл в приват.

Заключение

Этот бот работает уже в течение нескольких месяцев, за это время было услышано только несколько упреков о том, что бот иногда слишком много отвечал не в тему, либо просто засоряет своими сообщениями канал. Это было исправлено в последующих версиях. Также было очень много благодарностей. Ботом активно пользуются, в основном, командой *!расп*, которая выдает расписание занятий. Иногда в канал потока заходят пользователи, которые не имеют к нему отношения, для того чтобы посмотреть, что это за бот, некоторые просили посадить его и к ним на канал.

В перспективе планируется переписать бот полностью на код C++, хотя в ***mIRC*** очень силен и собственный скриптовый язык. Планируется добавить возможность автоматического заполнения расписания из формата *Excel* (в таком виде расписание предоставляет деканат), также планируется следить за тем, какая по счету — текущая неделя, и, возможно, будет добавлено регулярное обновление погоды через Internet (может быть, и не только погоды).

А.СОЛОНОЙ,
г.Минск, БГУИР, ФИТиУ,
гр.121703,
E-mail: Solig@tut.by

Программа для общения в локальных сетях Elite NetWorkChat

В наше время широкое распространение получили локальные сети. С каждым днем количество людей, подключенных к таким сетям, увеличивается. Локальные сети — это не только красивое название, но и возможность обмениваться информацией, проводить досуг, общаться с другими людьми. Для удовлетворения всех этих потребностей существует множество программ. Например, для поиска файлов и папок существует большое количество поисковых программ, для развлечений — сетевые игры, для общения — программы, которые получили название "чат" (от английского слова "chat" — разговор). Именно последнему типу программ посвящена эта статья.

Условно программы, которые позволяют осуществлять общение по сети, можно разделить на два типа:

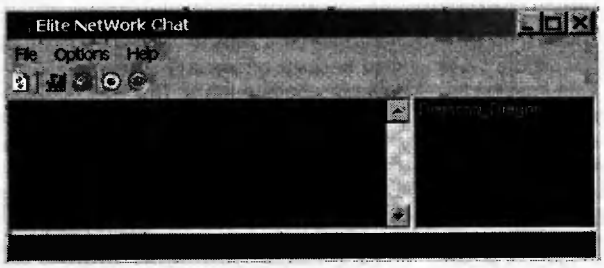
- программы, для работы которых требуется наличие выделенного сервера;
- программы, которые могут работать без выделенного сервера.

Первый тип программ чаще всего используется в больших локальных или в глобальных сетях. Примеры таких программ — Mirc, Xchat, ICQ. В малых сетях, состоящих из нескольких машин, не всегда есть возможность установить выделенный сервер, или не удается обеспечить его постоянную работу. Для решения этой проблемы используется второй тип программ. Примеры таких программ — IChat, Elite NeworkChat. Пользователь, который использует программу второго типа, не должен волноваться о наличии выделенного сервера. Единственное, что от него требуется — это запустить программу.

Elite NetWorkChat

Данная программа является чатом, который работает без выделенного сервера. Программа занимает мало места на жестком диске и не требует для своей работы наличия мощного компьютера. Интерфейс программы прост и интуитивно понятен (**рис.1**).

Рис. 1



Пользователь имеет возможность общаться сразу со всеми, кто пользуется в данный момент программой, или поговорить с кем-то наедине. Все, что нужно для такого общения — просто набрать фразу в поле ввода и нажать клавишу Enter. После выполнения этих действий набранное сообщение будет отправлено в сеть, и его увидят другие пользователи (**рис.2**).

Рис. 2

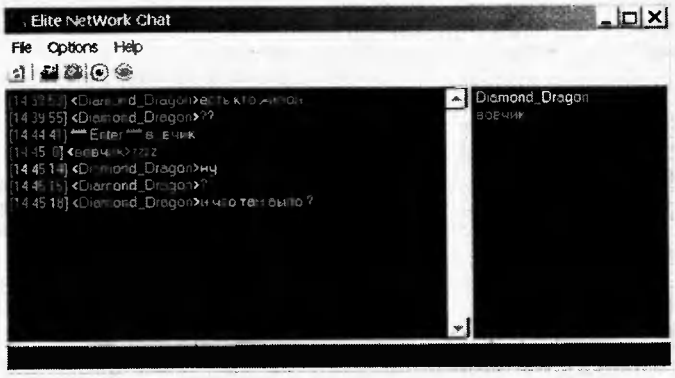
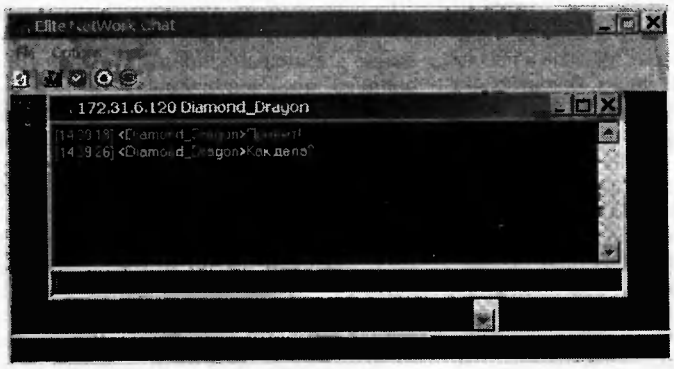


Рис. 3



Если необходимо пообщаться с кем-то наедине, для этого просто нужно выбрать из списка имя необходимого пользователя и дважды щелкнуть по нему мышкой. После этого откроется отдельное окно, в котором можно будет общаться тет-а-тет и не бояться, что кто-то посторонний увидит, о чем идет разговор (**рис.3**).

Помимо общения, программа предоставляет ряд дополнительных возможностей. Например, можно установить автоответчик, который будет на каждое сообщение, адресованное вам, выдавать сообщение, которое было ранее вами оставлено. Это довольно полезно, если вам нужно отлучиться, и вы хотите оставить информацию для тех, кто желает с вами пообщаться. Имеется возможность задать свой ник (от английского слова "nickname" — псевдоним). С личными сообщениями можно ассоциировать мелодию, которая будет проигрываться при получении нового личного сообщения. Кроме того, каждый сможет по своему вкусу настроить цвет текста и фона.

Все события, которые происходят во время работы программы (появление нового пользователя, выход из чата пользователя, смена ника), отображаются.

Заключение

Программа Elite NetWorkChat обладает всем необходимым для полноценного общения в сети. Она позволяет осуществить не только коллективное, но и индивидуальное общение. Эта программа уже давно используется на практике, и нареканий на ее работу не возникало.

ЛИСТАЯ СТРАНИЦЫ

Несколько программ для работы с электронной почтой (почтовых клиентов) были опробованы в редакции журнала "Подводная лодка", о чем рассказал *А.Орлов* в статье "Современные почтовые клиенты" **(Подводная лодка, №6/2004, С.88...95)**.

Outlook Express (OE) входит в стандартную поставку всех версий Windows, начиная с 98-й. Тот факт, что с OE можно работать сразу же после установки Windows, предопределил его популярность. При работе с ним набор параметров доступа к почтовым серверам (адрес, логин, пароль), называемый учетной записью, вводится вручную и сохраняется в системном реестре. При создании письма пользователь должен указать, через какую учетную запись он будет отправлять почту. Проверка почтовых ящиков и загрузкв находящихся в них сообщений, а также пересылка писем через соответствующие SMTP-серверы возможна как для каждой учетной записи отдельно, так и для нескольких сразу — в последнем случае они должны быть особым образом помечены в настройках. Для хранения корреспонденции в OE применяется система папок.

С одним экземпляром Outlook Express могут работать несколько пользователей — при этом у каждого из них будут персональные настройки, в т.ч. набор учетных записей и папок с сообщениями, хранящимися в разных местах. Совокупность индивидуальных настроек, учетных записей и папок с письмами в OE называется удостоверением, которое при желании закрывается пвролем.

Компания Microsoft снабдила свой продукт довольно мощными фильтрами (правилами), т.е. специальными алгоритмами обработки писем, которые на основании каких-либо сочетаний символов в теме, обратном адресе, тексте или полях заголовка сообщения осуществляют с корреспонденцией определенные действия. Например, автоматически распределяют ее по нужным папкам, переправляют на другие e-mail-адреса, самостоятельно отвечают на входящие сообщения (при должной настройке) или вовсе удаляют их из ящика без фактической загрузки на компьютер.

OE оснащен адресной книгой — подпрограммой организации базы данных для хранения имен, электронных адресов и прочих сведений о различных лицах. Среди прочих полезных функций OE автор отмечает автоматическую проверку почтовых ящиков и загрузку сообщений, работу с news-группами, систему поиска сообщений (по тексту, адресу отправителя или получателя, теме), проверку правописания, средство оформления письма шаблонами (бланками) в формате HTML с уже настроенными фоном и шрифтами, автоматическое добавление подписи (в виде небольшого текста или HTML-кода), возможность выбора кодировки для просмотра почты, систему оповещения о приходе новых сообщений.

The Bat! — разработка российской компании RitLabs. В отличие от OE, в The Bat! основной упор делается не нв одного и того же пользователя, а на один и тот же почтовый ящик: именно почтовый ящик является "независимой единицей", каковой в OE служит удостоверение. Если в OE входящие письма со всех учетных записей складываются в одну папку "Входящие", то в The Bat! для каждого из почтовых ящиков создается свой набор каталогов, и почта поступает именно в ту папку "Входящие", которая принадлежит этому ящику.

Интерфейс у The Bat! во многом более удобный, чем у Outlook Express. Так, в окне просмотра сообщения все вложения и варианты письма отображаются рядом, и пользователь может выбрать, какую версию письма он желает изучить. Окно создания сообщения в The Bbt! позволяет легко указать в письме обратный адрес, отличный от того, который записан в настройках почтового ящика: достаточно лишь ввести его в соответствующее поле. При написании ответа текст оригинала отображается в специальной области в верхней части окна создания сообщения.

Письма можно раскладывать в папки, сгруппированные по соответствующим почтовым ящикам. При необходимости доступ к ним закрывается паролем. За исключением стандартных для каждого ящика, папкам могут быть заданы свои параметры работы. Точно так же как в OE, в The Bat! сообщение можно составлять и на основании шаблонов. К ним, правда, нельзя применять HTML-оформление, однако допускается использование макросов — специвльных команд, автоматически выполняющих какие-либо действия. В The Bat! можно подготовить отдельные шаблоны для корреспонденции различного характера: нового письма, ответв, перенаправления.

Крайне полезной функцией The Bat! является "Диспетчер писем". Он соединяется с почтовым сервером и скачивает из соответствующего ящика только заголовки находящихся там писем, отображая их в специальном окне. Затем с письмами совершаются определенные действия, в частности, удаление без загрузки на компьютер. Функциональность почтовых фильтров The Bat! остааляет далеко позади все прочие продукты этой категории.

В The Bat! есть и прочие дополнительные сервисы: MailTicker (бегущая строка с информацией о поступивших письмах), "Предупреждение о днях рождения", "Мастер импортирования", "Резервное копирование", "Синхронизация", "Планировщик заданий" и многое другое.

Marlin — это бесплатный почтовый клиент, предназначенный для максимально безопасной работы. Умея превосходно отображать сообщения в формате HTML, Marlin не пропустит ни одного вируса, распространяющегося через такие письма. Он просто не будет выполнять ни ActiveX-программ, ни открывать вложенные файлы.

Схема функционирования почты в Marlin — такая же, как и в OE. Пришедшие письма со всех ящиков поступают в общую папку "Входящие", однако система удостоверений не предусмотрена: Marlin — программа для одного пользователя. Marlin предоставляет прекрасные возможности для чтения и создания писем в формате HTML, причем для отображения используются собственные алгоритмы, так что бреши творений Microsoft владельцам Marlin не страшны. При желании допускается редактирование исходного HTML-кода сообщения.

Функции создания шаблонов и настройки макросов почти не отличаются от своих аналогов в OE и The Bat!. Среди команд макросов предусмотрен ряд условий. Так, нетрудно подготовить особый шаблон только для переписки с каким-то одним адресатом, и Marlin будет применять его только в том случае, если вы отвечаете на письмо от определенного человека.

Наряду с обычным сортировщиком сообщений, реагирующим заданным образом на входящую почту, в Marlin встроена система фильтрации спама. Она способна анализировать пришедшие письма и, при наличии определенного фрагмента текста, автоматически их удалять.

Eudora — мало известный у нас, но популярный на Западе почтовый клиент. По схеме работы Eudora напоминает OE. Просмотр и составление писем в формате HTML Eudora выполняет хорошо. Имеется система шаблонов, именуемая Stationery. Из интересных функций этого клиента автор отмечает Eudora Sharing Protocol Groups (средство автоматического обмена файлами по e-mail для синхронизации папок на разных компьютерах) и PureVoice, позволяющую пересылать звуковые сообщения в сильно компрессированном виде.

В известный браузер Opera встроен почтовый клиент. Он органично вписан в браузер и подчиняется принципам его внутренней организации — в частности, многооконной структуре. Приходящая почта помещается в папку Received. В основном окне Opera открываются подокна со списком и областью просмотра сообщений для каждой папки. Данные доступа к почтовым ящикам хранятся в учетных записях — аккаунтах. Настройки их параметров довольно многообразны, и Opera почти не уступает по этому показателю OE.

Такие сервисы как адресная книга, почтовые фильтры или система поиска по загруженным сообщениям, в Opera тоже присутствуют. Однако единственное действие, которое выполняется системой фильтрации — отметка прочитанных сообщений. Несмотря на некоторое число интересных особенностей, данный почтовый клиент значительно менее функционален, чем остальные участники обзора.

Foxmail — плод работы китайских программистов. И надо сказать, что со своей задачей они справились неплохо. Если сравнивать Foxmail с упомянутыми программами, то многие увидят в ней упрощенную версию The Bat!. Foxmail предоставляет полноценную возможность для просмотра и подготовки писем в формате HTML. Средства редактора HTML-сообщений превосходят аналогичные в OE. К сожалению, Foxmail страдает тем же недостатком, что и Eudora — эта программа поддерживает только кодировку Windows-1251.

В Foxmail имеются и дополнительные сервисы, хотя их значительно меньше, чем в The Bat!. Реализована здесь и система почтовых фильтров, которые, пввда, предоставляют небольшой набор действий. По части прочих удобств Foxmail отстает от конкурентов.

Mozilla Thunderbird — некоммерческая программа с открытым кодом. Общий принцип действия Mozilla Thunderbird — по образу и подобию The Bat!: персональный набор папок для каждого из почтовых ящиков. Соответственно, приходящие на разные ящики письма помещаются в отдельные папки "Входящие", причем допускается создание общих каталогов, не принадлежащих ни одной из учетных записей.

Поддержка HTML в Mozilla Thunderbird реализована на более высоком уровне, чем в OE. Данный сервис позволяет вставлять в письмо таблицы и даже автоматически генерировать его содержание на основе наличествующих заголовков. Продуманы многие особенности работы с почтовым клиентом. Например, в окне создания сообщения имеется весьма удобный пункт меню — "Отправить копию", позволяющий скопировать послание в другую папку. Из прочих привлекательных сервисов автор отмечает шаблоны сообщений, красивую адресную книгу и мощную систему поиска. Очень хороша система борьбы со спамом.

В конце автор отмечает самые интересные особенности программ: Microsoft Outlook Express — удобный интерфейс, имеется почти на любом ПК, оснащенном Windows; The Bat! — огромное множество функций, почтовый робот; Marlin — отложенная загрузка, модуль шифрования PGP; Eudora — портфель с синхронизацией по e-mail, звуковые сообщения в письмах; Opera — встроен в браузер Opera; Foxmail — хорошая поддержка HTML; Mozilla Thunderbird — прекрасная поддержка HTML, интеллектуальный антиспам-фильтр.

Чем закончились гонки автомобилей, управляемых компьютерами, рассказал *Р.Соболенко* в заметке "За рулем — непьющий компьютер" **(Hard'n'Soft, №5/2004, С.12).**

В пустыне Мохаве (США) были проведены соревнования автомобилей-роботов. В них участвовали пятнадцать команд. Участникам предстояло преодолеть 200 миль (свыше 320 км) трассы, проложенной по пересеченной местности, и успеть сделать это за 10 часов. Вмешательство персонала было жестко ограничено возможностью подать две команды: на запуск системы и на ее выключение в случае схода с трассы. Всем остальным должна заниматься компьютерная система управления.

Организаторы состязаний предоставили координаты тысячи точек, описывающих трассу. Сведения об их прохождении машины получали с помощью системы глобального позиционирования GPS. Точность определения координат составила 1 м. Роботы были оснащены разнообразными датчиками — видеокамерами, лазерными сканерами, радиолокационными сенсорами и т.п. Поступающие от них данные немедленно обрабатывались системами управления, сравнивались с картографической информацией и саедениями от GPS-приемников, и в результате поступали к исполнительным механизмам, непосредственно управляющим автомобилями.

Невероятный объем вычислений и высокая сложность алгоритмов потребовали применения самых производительных компьютерных решений. Например, корпорация Intel оказала поддержку команде Red Team из Университета им. Карнеги Меллона, разработавшей машину Sandstorm на базе джипа Hammer. В ее системе управления был использован основной компьютер на базе четырех процессоров Itanium 2 и четыре вспомогательных, каждый из которых был оснащен двумя ЦП Xeon. Главный конкурент Intel — компания AMD — сотрудничала с командой Blue Team из Калифорнийского университета (г.Беркли), создавшей двухколесное транспортное средство, управляемое компьютером на базе процессоров Opteron и Athlon 64.

Победителя в этих соревнованиях не оказалось. Двум из пятнадцати команд не удалось выйти на старт (в том числе и Blue Team). Еще три машины не смогли его покинуть — у одной не отпустили тормоза, другая врезалась прямо на старте в стену, третья начала выписывать круги. Следующие три "жертвы" не проехали и мили, причем две из них перевернулись. Две участницы с первой милей справились, но отъехали от этой отметки недалеко, сбились с маршрута и были выключены. Дальше всех продвинулась машина Sandstorm, добравшаяся до гористой зоны, съехавшая с трассы, после чего у нее загорелись передние колеса. Пожар был быстро потушен, однако и ее пришлось снять с состязаний.

Результаты тестирования цифровых видеокамер нижнего ценового диапазона рассмотрены в статье "Бюджетное видео" **(CHIP, №6/2004, С.58...65).**

Сегодня, благодаря значительному снижению цены, видеокамеры стали доступны многим. Однако фирмы-производители не желают просто так дарить свою продукцию, а потому снижение цены происходит за счет уменьшения количества функций в недорогих видеокамерах. В базовых моделях отсутствует целый ряд функций, которые, по мнению автора, нельзя назвать незаменимыми. Сюда относятся, например, разъемы DV-In и режим фотосъемки. Это значит, что вы не сможете делать с помощью недорогих видеокамер фотоснимки или же перегонять отредактированный фильм обратно на DV-пленку. Фотографировать лучше все-таки фотоаппаратом, а обработанный видеофильм лучше сразу записать на DVD-болванку, которую можно будет воспроизвести на любом плеере.

Как известно, раньше видео- и аудиовыходы недорогих камер знающий специалист мог модифицировать, и они начинали функционировать еще и как входы. Умельцы даже мастерили специальные переходники, но теперь все это в прошлом. У современных видеокамер цифровые и аналоговые входы заблокированы необратимо, поэтому использовать видеокамеру в качестве цифрового видеомагнитофона теперь уже не удастся.

Несмотря на определенные ограничения, младшие модели в плане комплектации нельзя назвать обделенными. Они имеют целый ряд опций, которых нет и в более дорогих моделях. Однако самым примечательным является то, что, в отличие от дешевых цифровых фотоаппаратов, все протестированные недорогие видеокамеры обладают на удивление широкими возможностями ручной настройки параметров съемки. Возможность вручную установить диафрагму, баланс белого, выдержку и фокус — все это понравится опытному оператору, так как эти функции помогают "вытянуть" даже некачественные видеозаписи.

Тестировались видеокамеры Panasonic

Табл. 1

Модель	NV-DS60EN	MV600	GR-D20E	DCR-TRV147E	VP-D20i
Формат	Mini-DV	Mini-DV	Mini-DV	Digital8	Mini-DV
Вес, г	510	595	605	810	512
Габариты, мм	70x85x133	65x100x152	75x92x155	89x101x199	58x90x156
Размер матрицы, дюймов/пикселов	0,17/800000	0,17/800000	0,17/800000	0,17/540000	0,17/800000
Разрешение, пикселов	720x575	720x575	720x575	720x575	720x575
Коэффициент увеличения, оптич./цифр.	10x/500x	18x/360x	16x/700x	20x/560x	10x/800x
Светосила	1,8	1,6	1,6	1,6	1,4
Фокусное расстояние (35-мм эквив.), мм	43,5...435	54...972	51,8...828,8	42...840	52...520
Фокусное расстояние (видео), мм	2,3...23	2,8...50,4	2,7...43	2,5...50	2,7...27
Диаметр фильтра, мм	27	30,5	37	37	37
Размер дисплея, дюймов/пикселов	2,5/113000	2,5/112000	2,5/113000	2,5/123200	2,5/112000
Видоискатель	Черно-белый	Цветной	Черно-белый	Черно-белый	Черно-белый
Количество программ съемки	5	6	4	6	5
Ресурс аккумулятора, мин	85	75	70	120	90
Цена, USD	480	500	460	400	450

NV-DS60EN, Canon MV600, JVC GR-D20E, Sony DCR-TRV147E и Samsung VP-D20i. Их технические характеристики приведены в **табл.1**, оценки редакции журнала "CHIP" — в **табл.2**. Значок "Лучший продукт" был присвоен модели NV-DS60EN от Panasonic. Это пример удачно сконструированного и недорогого устройства, которое демонстрирует хорошие качества изображения и звука. Благодаря наличию большого числа ручных настроек, на эту камеру могут обратить внимание самые амбициозные любители видеосъемки. Значок "Оптимальный выбор" получила Sony DCR-TRV147E. Эта недорогая модель будет в первую очередь интересна тем, кто раньше работал с камерами формата Hi8. За 400 USD вы получите камеру нового поколения с качественной автоматикой и картинкой, характеризующейся нейтральными цветами. Ее недостатки — спартанская комплектация и отсутствие стабилизатора изображения. Да и габариты камеры трудно назвать компактными.

Табл. 2

Модель	NV-DS60EN	MV600	GR-D20E	DCR-TRV147E	VP-D20i
Качество изображения	78	74	70	64	60
Время работы аккумулятора	73	67	64	90	80
Удобство использования	82	87	77	60	77
Комплектация	77	69	65	59	69
Качество звука	79	61	68	70	62
Сервис/документация	70	73	62	61	52
Общая оценка	77	72	69	68	68

Подводя итоги тестирования, автор отмечает, что базовые модели видеокамер от ведущих производителей действительно очень недороги. Можно рекомендовать приобретение таких камер, поскольку применяемые в них технологии — те же, что и в более дорогих моделях. Вот почему качество отснятого изображения и записанного звука вполне сопоставимо с качеством старших моделей.

Однако чуда ждать не следует. Различия в цене между топ-моделями и протестированными камерами имеют вполне понятные причины: разница в комплектации и, как следствие, в возможностях. Но для первой в вашей жизни видеосъемки или для работы с камерой от случая к случаю подобные устройства подходят очень хорошо. Например, нет никаких препятствий для того, чтобы взять одну из протестированных моделей с собой в путешествие или просто на природу и почувствовать себя настоящим кинооператором.

Об оригинальном оформлении звуковых колонок сообщается в заметке "Прозрачный звук" **(Hard'n'Soft, №5/2004, С.40).**

Комплект колонок Harman/Kardon SoundSticks II состоит из сабвуфера округлой формы и двух сателлитов в виде больших колб, выполненных из прозрачного пластика, сквозь который видны внутренняя начинка и особенности устройства акустической системы. В качестве излучателя басов используется шестидюймовая низкочастотная головка, а каждая фронтальная колонка оснащена четырьмя динамиками диаметром 1 дюйм. Управлять громкостью звучания весьма удобно благодаря вынесенным на правый сателлит регуляторам. На сабвуфере имеется собственный регулятор, предназначенный для точной настройки уровня басов.

Благодаря внешнему блоку питания, исключаются низкочастотные помехи и наводки, вызываемые сетью переменного тока. Разработчики использовали высококачественные компоненты — при нулевом уровне громкости колонки замолкают полностью. Подключение к любому источнику сигнала — будь то звуковая карта компьютера, плейер или музыкальный центр — осуществляется очень просто: кабелем сабвуфера, а к последнему уже подсоединяют сателлиты.

Качество звучания акустики находится на высоком уровне, и в первую очередь это касается сателлитов. Они весьма чисто передают средние и высокие частоты. Разработчики точно рассчитали акустические свойства колонок с учетом их формы. Сабвуфер хорошо справляется с "прокачкой" басов, однако, в силу довольно компактных размеров, он начинает "захлебываться", достигнув определенного порога мощности. Стоимость системы — 260 USD. Ее основные технические характеристики следующие:

- мощность сабвуфера — 20 Вт;
- мощность сателлитов — 2x10 Вт;
- частотный диапазон — 44...20000 Гц;
- габариты сабвуфера — 232x258 мм;
- габариты сателлита — 50,8x254 мм;
- материал корпуса — оргстекло.

Небольшой обзор программ финансового учета, функционирующих на КПК, провел *В.Акулов* и рассказал об этом в статье с очень вктуальным для всех названием "Куда уходят деньги" **(CHIP, №6/2004, С.132...135)**.

Cash Organizer Deluxe 2003 (разработчик — Inesoft, сайт разработчика — www.inesoft.com, стоимость — 30 USD) доступна для скачивания на сайте разработчика. Там же можно найти русскоязычную документацию. Отличительной особенностью Cash Organizer является ее позиционирование не только для частной, но и для корпоративной финансовой деятельности. Она является платной, однако у пользователя есть 25 дней до регистрации, чтобы оценить и испытать ее.

Программа обладает всеми необходимыми функциями, в частности, контроля сразу над несколькими счетами, категориями и проектами; планирования будущих транзакций (финансовых операций) и бюджета; прогнозирования финансового результата на заданную дату. Кроме того, с ее помощью можно создавать отчеты в различных формах представления; экспортировать данные в форматы CSV, QIF и из них импортировать в программу. Поддерживается резервное копирование базы данных и защита ее паролем.

Программа тщательно следит за вашими действиями и предупредит вас, если вы, например, ошибочно введете расход по статье прихода. Можно отметить планирование бюджета, реализованное в этой программе. Помимо разделения его статей по категориям, возможно и разделение по проектам. То есть транзакции одной категории могут влиять на бюджет поразному, в зависимости от принадлежности к тому или иному проекту.

К недостаткам программы автор относит некоторую перегруженность интерфейса и требовательность к ресурсам. Обширное количество элементов управления и продолжительная загрузка на недорогих 200 МГц-машинах может отпугнуть начинающего пользователя. В целом же он считает, что Cash Organizer на сегодняшний день является самой мощной и функциональной программой для полноценного финансового учета.

Другим представителем данного класса программ является SPB Finance отечественной компании SPB Software House (сайт разработчика — www.softspb.com, стоимость — 20 USD). Она имеет полностью русскоязычный интерфейс. Программа является платной и предлагает пользователю воспользоваться ознакомительной версией, которая будет работоспособна в течение 15 дней.

Как и большинство программ данной компании, SPB Finance нетребовательна к ресурсам, имеет приятный, продуманный интерфейс и обладает всеми необходимыми средствами учета. Небольшие нарекания вызывает лишь слегка усложненный процесс выбора или создания новой категории при вводе транзакции. Автор отмечает и некоторые другие ее недостатки, но в целом SPB Finance является прекрасным примером качественной и полноценной программы финансового учета на КПК и, несомненно, заслуживает внимания.

Еще одно творение отечественных разработчиков — программа Disco Money, продукт компании ДИСКо (сайт разработчика — www.disco.ru, стоимость — 10 USD). Данная утилита является платной, но ее демоверсия также доступна на сайте разработчика. По мнению автора, программа еще довольно сырая и не обладает всем необходимым набором функций. В частности, в ней нет средств для планирования, а отчетность реализована лишь в виде набора фильтров списка транзакций, без какого-либо графического представления. Нет возможности импорта и экспорта информации, встроенных средств резервного копирования.

Основным недостатком явилась попытка разработчиков "изобрести велосипед" в области проектирования интерфейса. В итоге были изобретены две кнопки, имеющие смысл "ОК" и "Отмена", однако они расположены внизу, на панели инструментов — там, где у всех остальных программ находится меню. В результате при работе с программой человек, привыкший к стандартному расположению "ОК" в правом верхнем углу, вместо подтверждения ввода закрывает программу.

Программа Mastersoft Money компании

MasterSoft Mobile Solutions (сайт разработчика — www.mastersoftmobilesolutions.com, стоимость — 20 USD) представляет собой монстрообразное чудовище, загружавшееся более пятнадцати секунд и беспощадно сожравшее порядка 20 Мб оперативной памяти. Функциональная начинка ее при этом весьма посредственна — счета, транзакции, категории и подобие бюджета. Интерфейс также оставляет желать лучшего: мелкий шрифт, перегруженность цветом и непростительный расход драгоценного места огромными иконками.

В конце автор упоминает программу, работающую на настольном ПК. Это пакет "1С:Деньги" известного российского производителя 1С (сайт разработчика — www.1c.ru, стоимость — 18 USD). Помимо того что в этой программе есть все необходимое для учета, анализа и планирования, программа имеет отличный интерфейс, и для российского пользователя в ней припасена масса приятных моментов, например, можно распечатывать квитанции для коммунальных платежей в соответствии с государственными стандартами, закачивать курсы валют или обновленные конфигурации программы. Кроме того, на данный момент это единственная российская программа, способная помочь вам правильно составить налоговую декларацию.

Подводя итоги, автор отмечает, что если вас не пугает англоязычный интерфейс, и у вас есть КПК, хорошим выбором станет пакет Microsoft Money (разработчик — microsoft, сайт разработчика — www.microsoft.com, стоимость — 40 USD), о котором рассказывается в начале статьи. Особенно если вы собираетесь вести учет не только личных финансов, но и управлять своим бизнесом.

Если, находясь в постоянных разъездах, вы видите настольный компьютер только вечером, да и то в выключенном состоянии, наилучшим выбором для вас станет Cash Organizer или SPB Finance..

Ну а если ваш КПК еще лежит на прилавке магазина и не собирается перекочевать в ваш карман в ближайшие годы, то пакет "1С:Деньги" — это именно то, что вы так долго искали.

В компьютерной индустрии грядет очередная микрореволюция. Широко распространенная шина PCI уходит в прошлое, ее заменяет **новая шина PCI Express**. Это, конечно, улучшит характеристики компьютеров, но для нас, рядовых пользователей (чайников, самоваров, бойлеров, паровых котлов и прочей водонагревательной техники), это обернется одновременным старением всех имеющихся компьютеров и плат расширения. Короче — придется раскошелиться. Об этой "радостной" перспективе поведал *Р.Никитин* в статье "Экспресс маршрута PCI" **(CHIP, №6/2004, С.42-43)**.

Первая версия стандарта PCI (Peripheral Component Interconnect) была предложена в 1991 году, а через три года появился стандарт версии 2,0, который впоследствии и стал наиболее распространенным. Базовые принципы этой шины — увеличение пропускной способности (как за счет тактовой частоты, так и за счет разрядности), надстраивание над PCI специализированных интерфейсов, например, AGP или Cardbus.

Новая шина является последовательной. Это объясняется тем, что гораздо дешевле создать функциональный и производительный контроллер, работающий на высокой частоте в последовательном режиме, чем дальше иметь дело с большим количеством проблем параллельных интерфейсов — наводками и десинхронизацией данных по разным каналам. Новый стандарт позволит упростить разводку материнских плат, что должно положительно сказаться на стоимости последних. Несмотря на то что спецификация PCI Express предусматривает стандарт x32, то есть 32-канальный интерфейс, в настольных решениях максимально широким является стандарт x16, да и то физически 16-канальный интерфейс будет единственным — он заменит AGP. Все остальные интерфейсные разъемы на материнских платах могут быть одноканальными, к каждому из которых подводятся всего четыре сигнальные линии.

Еще одно преимущество — защита данных от помех, что обеспечивается избыточным кодированием: каждый байт данных пересылается десятью битами. Разработчики заявляют, что этого будет вполне достаточно, чтобы обеспечить надежную работу шины на тактовой частоте 2,5 ГГц. Свою лепту в дело надежности вносит и асинхронность передачи данных по каждому из каналов. По сути, тот же интерфейс x16 представляет собой 16 независимо функционирующих каналов передачи данных.

Пропускная способность новой шины даже в самом медленном исполнении составляет 200 Мб/с, что почти в два раза больше, чем у PCI. В случае с высокоскоростным режимом x16 этот показатель составляет 3,2 Гб/с, что также больше, чем у AGP. То есть геймеры могут смело смотреть в будущее и копить средства на новый графический адаптер.

Кроме того, применение PCI Express позволит снизить общее энергопотребление компьютерной системы, так как общее число сигнальных линий, нуждающихся в питании, серьезно сократится. Соответственно, снизится и нагрев системной платы и элементов стабилизации питания на ней. Дополнительно стоит заметить, что PCI Express несет некоторые алгоритмы управления питанием, что тоже должно положительно сказаться на энергетическом аппетите компьютера.

Наконец, новый стандарт изначально поддерживает горячую замену карт расширения. Возможно, рядовой пользователь этого и не оценит, но вот системщики отнесутся к этому нововведению очень благосклонно.

О новом корпусе, позволяющем создать очень тихий компьютер, рассказывает *Д.Ноеиков* в заметке "Zalman TNN 500A: тишайший" **(Подводная лодка, №6/2004, С.38...39)**.

Компания Zalman, специализирующаяся на малошумящих охлаждающих системах, предложила комплексное решение для создания бесшумного ПК — корпус TNN 500A, в котором нет вентиляторов. Аббревиатура TNN в названии этого корпуса расшифровывается как Totally No Noise — то есть абсолютно бесшумный. Основная идея, реализованная в этом решении — сделать корпус ПК активным элементом охлаждающей системы. Он изготовлен из алюминия, толщина его стенок колеблется от 5 до 7 мм, а боковые стенки имеют, к тому же, длинные и высокие ребра. Фактически TNN 500A — это огромный алюминиевый радиатор, внутри которого можно смонтировать компоненты ПК. Корпус окрашен в черный цвет, оснащен колесиками и массивными рукоятями на верхней панели — это необходимо, поскольку масса пустого корпуса составляет 25 кг.

В комплект поставки TNN 500A входят алюминиевые и медные тепловые интерфейсы с гнездами для крепления теплоотводящих трубок с легко испаряющейся жидкостью. Эти интерфейсы устанавливаются вместо штатных радиаторов на ЦП и графическую микросхему видеоплаты и соединяются теплопроводящими трубками с интерфейсами, закрепленными на боковых стенках TNN 500A. Таким образом, с помощью совмещения пассивного и жидкостного охлаждения тепло отводится от горячих микросхем к стенкам корпуса-радиатора. Еще один элемент с жидкостным охлаждением — рамка для жесткого диска, состоящая из двух массивных алюминиевых пластин, к которым подведены теплопроводящие трубки. К сожалению, предусмотрена только одна такая рамка, дополнительные жесткие диски (всего их может быть шесть) монтируются к закрепляемым на другой стенке алюминиевым пластинам. Этот вариант рассчитан прежде всего на серверы, содержащие большое количество дисков, не собранных в RAID-массив. Энтузиасты же компьютерных игр, нередко использующие RAID-массив из двух дисков для увеличения быстродействия, могут приобрести вторую рамку. За отдельную плату можно приобрести дополнительные тепловые интерфейсы и теплопроводящие трубки для охлаждения других микросхем системной платы.

Блок питания TNN 500A также не имеет вентилятора. Более того, он разделен на две части: в одной расположены разъемы шнуров питания и выключатель питания, а во второй — электронная часть блока. Последняя выполнена в плоском алюминиевом корпусе, расположенном непосредственно на боковой стенке-радиаторе.

Передняя и задняя стенки корпуса выполнены в виде дверок с магнитными замками. Монтаж комплектующих внутри корпуса TNN 500A — кропотливое и трудоемкое занятие: только для того чтобы отсоединить боковую стенку и открыть корпус, необходимо вывернуть 10 болтов. Кроме того, после монтажа комплектующих необходимо установить тепловые интерфейсы (каждый из них крепится двумя или четырьмя болтами) и подсоединить к ним теплопроводящие трубки. Всего же TNN 500A состоит из 25 основных элементов, части которых соединены более чем 150 болтами. Так что Zalman TNN 500A — это не просто тяжелый алюминиевый корпус, это достаточно сложное техническое устройство, конструкция которого тщательно продумана. Его размеры — 400x286x607 мм, мощность блока питания — 300 Вт, форм-фактор — ATX, стоимость — 900 USD.

Borland C++ Builder 6 для начинающих

О.ВАЛЬПА,
E-mail: sandh@narod.ru

(Продолжение. Начало в №№7-9/2004)

При разработке программ, особенное в первое время, возможно внесение как синтаксических, так и логических ошибок. Чем сложнее будет программа, тем труднее их будет обнаружить. Для облегчения обнаружения и устранения ошибок в среде разработки Borland C++ Builder 6 существует встроенный отладчик программ. Он позволяет выполнять программу по шагам, устанавливать точки останова, просматривать переменные и т.п.

Отладка программ

Несмотря на то что созданные нами программы очень просты, я намеренно хочу рассказать об отладке программ. Дело в том, что рано или поздно в создаваемой программе могут появиться ошибки. Это могут быть синтаксические или более сложные для обнаружения логические ошибки. И тогда потребуется инструмент для их устранения. Чем быстрее мы познакомимся с таким инструментом, тем легче нам будет создавать более сложные программы.

После разработки любого приложения необходимо выполнить его компиляцию, компоновку и тестирование. Эти операции выполняются автоматически каждый раз при выполнении команды **Run** из главного меню среды разработки Borland C++ Builder 6. Однако данные процедуры можно выполнять по отдельности, с целью сокращения времени на отладку всего проекта в целом. В нашем очередном примере мы будем пользоваться различными командами трансляции (компиляции и компоновки) проекта. Рассмотрим, в чем состоит отличие между этими командами. Например, компиляцию отдельного модуля программы можно выполнить с помощью команды главного меню **Project Compile**. При этом не будет затрачено время на компиляцию и компоновку всего проекта в целом. Команда главного меню **Project Make** позволяет выполнить компиляцию только тех модулей, которые были отредактированы, после чего выполняет компоновку проекта с созданием исполняемого модуля **exe**. Таким образом, эта команда также экономит время на трансляцию проекта в целом. И только команда **Project Build** компилирует все модули, независимо от того, редактировались они или нет. А затем компонует проект и готовит исполняемый файл с расширением **exe**. Данная команда необходима для переработки всего проекта после изменения настроек компилятора или среды разработки в целом. Команда **Project Run**, в отличие от **Project Build**, автоматически запускает исполняемый файл.

На этапе компиляции происходит автоматический поиск средой разработки синтаксических ошибок, неправильного объявления или использования переменных и т.п. При обнаружении подобных ошибок в инспекторе кодов будет выведено соответствующее сообщение об ошибке или предупреждение. Предупреждение, в отличие от ошибки, не блокирует выполнение программы, но не гарантирует корректность ее работы. Если ошибок в программе нет, компилятор создает объектный (машинный) код программы в модулях (файлах) с расширением **obj**. При компоновке программы также выявляются некоторые виды ошибок, связанные с отсутствием объявления переменных, функций, библиотек и т.п., и выполняется объединение всех объектных модулей, и библиотек при необходимости, в один исполняемый файл с расширением **exe**. Вместо исполняемого файла может создаваться библиотечный файл с расширением **lib** или **dll** — при соответствующих установках среды разработки. Рассмотрим выполнение описанных процедур на конкретном примере. Создадим небольшое приложение на форме **Form1** с двумя кнопками **Button1, Button2** и одним элементом надписи **Label1**. О том как размещаются эти элементы на форме, рассказывалось в предыдущих статьях. Это приложение должно сообщать нам о количестве нажатий на первую кнопку. При нажатии на вторую кнопку приложение должно закрываться. Разместите эти элементы на форме **Form1** и измените размер формы в соответствии с **рис.1**.

Рис. 1

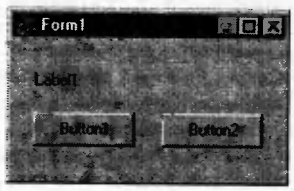
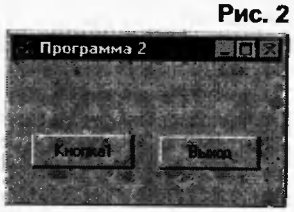


Рис. 2



Теперь замените свойство **Caption** всех объектов приложения в инспекторе объектов на заголовки **Программа 2, Кнопка 1, Выход** и **пустую строку** соответственно. В результате у вас должно получиться окно, приведенное на **рис.2**. Обратите внимание, что элемента **Label1**, располагающегося выше кнопок, не видно, поскольку мы заменили его свойство **Caption** на **пустую строку**, т.е. очистили это свойство.

Дважды щелкните левой кнопкой мыши по созданной кнопке **Выход**, и в открывшемся окне инспектора кодов впишите между фигурными скобками заготовки обработчика события команду закрытия приложения **Close();**. Вернитесь к окну формы и дважды щелкните левой кнопкой мыши по кнопке с названием **Кнопка 1**. В открывшемся окне инспектора кодов впишите между фигурными скобками строку команд:

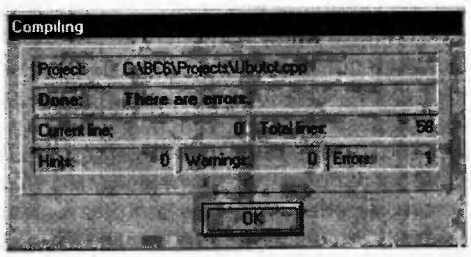
```
Label1->Caption="Кнопка 1 нажата " + IntToStr(++i) +" раз";
```

В этой строке команд выполняется присвоение (знак равно) свойству **Caption** элемента надписи **Label1** — текстовой строчки, состоящей из трех составных частей. Поскольку свойство **Caption** элемента надписи **Label1** предназначено для отображения текста, мы должны присваивать этому свойству только текстовые (строковые) значения. В языке C++ такие строковые значения заключаются в кавычки. Первая и последняя части присваиваемого значения таковыми и являются. Счетчиком количества нажатий на кнопку в программе будет служить переменная **i**, которая должна автоматически увеличиваться на единичку перед выводом. Для этой цели перед ней записаны два знака плюс. Данная операция в языке C++ называется автоинкрементном. Для превращения числовой переменной **i** в строковую переменную используется встроенная функция C++ преобразования целых чисел в строки **IntToStr()**. Итак, в одной строке команд мы осуществили целый ряд операций. Разве это не изящный язык программирования?!

Сохраните проект под именем **butct.bpr**, а программный модуль — под именем **Ubutct.cpp**. Впрочем, имена вы можете дать другие. Не изменяйте только расширения файлов.

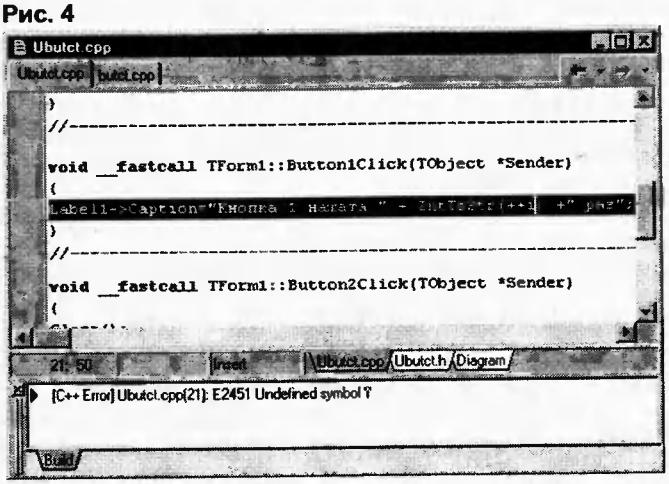
Теперь попробуем скомпилировать, скомпоновать и выполнить данное приложение, а заодно проверить, нет ли в нем ошибок. Выполните команду **Compile** из группы **Project** главного меню или нажмите "горячую" комбинацию клавиш для компиляции программы **Alt+F9**. Перед вами откроется окно, приведенное на **рис.3**.

Рис. 3



В верхней строке этого окна отображается путь размещения и имя проекта программы. В следующей строке вначале отображается процесс компиляции, сменяющийся записью о завершении данной операции **Done** и сообщением об обнаруженных ошибках **There are errors** (здесь есть ошибки). Ниже отображается номер текущей строки программы **Current line** и общее количество строк программы **Total line**. В нижней строке отображается обнаруженное на данный момент число замечаний **Hints**, предупреждений **Warnings** и ошибок **Errors**. Как видим, в программе есть одна ошибка. Нажмите кнопку **OK**, и перед вами окажется окно инспектора кода с выделенной строкой, имеющей ошибку, и сообщением о типе ошибки внизу окна (**рис.4**).

Рис. 4



Сообщение "**[C++ Error] Ubutct.cpp(21): E2451 Undefined symbol 'i'**" говорит о том, что компилятор C++ обнаружил ошибку в 21-й строке модуля **Ubutct.cpp**. Код ошибки — **E2451** — означает, что обнаружен необъявленный символ. В нашем случае это переменная с именем **i**. Действительно, мы не объявили ее в программе, хотя и сделали это намеренно, с целью получения сообщения об ошибке в качестве примера. В языке C++, как, впрочем, и в других языках программирования, все переменные, используемые в программе, необходимо объявлять. Это делается для того, чтобы компилятор знал, сколько ячеек памяти необходимо резервировать в памяти компьютера для ее хранения. Поскольку различные типы переменных требуют для своего хранения различного количества ячеек памяти (байтов), рационально резервировать для них лишь необходимый объем памяти. В нашем случае переменная **i** является целой, т.е. принимающей только целочисленные значения. Для ее объявления необходимо использовать директиву **int** (от integer — целый). Можно сделать такое объявление непосредственно в тексте программы обработчика событий, но тогда эта переменная будет недоступна в других функциях- обработчиках. Поэтому мы объявим ее в блоке **public** файла описания заголовков **Ubutct.h**. Для этого необходимо щелкнуть левой кнопкой мыши по закладке **Ubutct.h** инспектора кодов и вписать строку объявления переменной с комментариями сразу же после строчки **public**. Ниже эта запись приведена полностью.

```
public: // User declarations
int i;  // Переменная - счетчик
```

Теперь снова выполним компиляцию и обнаружим, что ошибок нет. Но не спешите радоваться. Дело в том, что мы не присвоили переменной **i** начального значения. Компилятор не обращает на это внимания и не выдает никаких сообщений по этому поводу. Но в программе это может привести к недоразумениям. Поэтому выполним присвоение начального значения переменой **i**, сделав запись **i=0;** (ведь при запуске программы кнопка 1 ни разу не была нажата) в тексте программы обработчика создания формы. Для этого дважды щелкните левой кнопкой мыши по форме приложения и в открывшемся окне инспектора кода впишите между фигурными скобками текста программы строку **i=0;**.

Выполним компоновку проекта с помощью команды **Make** из группы **Project** главного меню, или просто нажав "горячую" комбинацию клавиш для компоновки программы **Ctrl+F9**. В результате мы должны получить безошибочную компоновку приложения. Теперь приложение можно сохранить и запустить на выполнение с помощью команды **Run**, или воспользоваться "горячей" кнопкой запуска программы **F9**.

На экране появится окно нашей программы. Щелкая левой кнопкой мыши по кнопке с названием **Кнопка 1**, вы увидите, что каждый раз в окне будет появляться сообщение о количестве нажатий этой кнопки.

В будущем, каждый раз перед запуском приложения после редактирования, не забывайте сохранять проект. Это позволит избежать потери информации при зависании программы, вызванном наличием ошибок в ней.

Потренируйтесь самостоятельно, специально внося ошибки в программу и изучая возникающие при этом сообщения об ошибках при компиляции и компоновке проекта.

Структура файлов

Теперь рассмотрим более подробно структуру основных файлов программы, для того чтобы не теряться при возникновении сложных ошибок и иметь представление о том, как работает программа для устранения ошибок. Вернемся к нашему примеру, рассмотренному выше, и рассмотрим состав файлов проекта. Для лучшего понимания текста программ, тем, кто не знаком с языками программирования, я рекомендую познакомиться с основами языка программирования C++, например, по книге [1]. В языке программирования C++ главной функцией проекта является функция **main**. В программах, написанных для операционной системы Windows, главной функцией является **WinMain**. Данная функция содержится в программном

модуле **butct.cpp** нашего примера. Ниже приведено его содержимое.

```
//---------------------------------------------
#include <vcl.h>
#pragma hdrstopz
//---------------------------------------------
USEFORM("Ubutct.cpp", Form1);
//---------------------------------------------
WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)
{
   try
   {
            Application->Initialize();
           Application->CreateForm(__classid(TForm1), &Form1);
            Application->Run();
    }
    catch (Exception &exception)
    {
            Application->ShowException(&exception);
    }
    catch (...)
    {
             try
             {
                     throw Exception("");
             }
             catch (Exception &exception)
             {
                     Application->ShowException(&exception);
             }
    }
    return 0;
}
//---------------------------------------------------------
```

В этих нескольких строчках содержатся директивы и команды, с помощью которых формируется программа. Строчки, начинающиеся с символов "//", являются строками комментариев для разделения или снабжения текстовым описанием частей программы. В языке Си также допускается заключать комментарии между символами "/*" и "*/". В этом случае комментарии могут состоять из нескольких строк, заключенных между этими символами. Строка **#include <vcl.h>** является строкой препроцессора, т.е. программы, которая будет подключать к нашим модулям дополнительные файлы библиотек и описаний. В данном случае, строка **#include <vcl.h>** говорит препроцессору о том, что ему необходимо подключить к нашей программе заголовочный файл **vcl.h**, в котором содержатся описания визуальных компонентов Borland C++ Builder. Директива **#pragma hdrstop** завершает список файлов заголовка для препроцессора. Строка макрокоманды **USEFORM("Ubutct.cpp", Form1);** подключает к проекту модуль **Ubutct.cpp** и форму **Form1**. Далее следует заголовок программы главной функции проекта **WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)**. Перед названием главной функции находится дескриптор **WINAPI**, который позволяет использовать множество готовых полезных функций Windows в нашем приложении. В скобках заключены параметры главной функции, необходимые для опознания нашего приложения среди множества других работающих приложений, для передачи параметров в программу в командной строке и для определения конфигурации окна нашего приложения. После заголовка главной функции следует блок кода, в котором может произойти исключение — аварийная ситуация (останов, ошибка программы и пр.). Такой блок должен начинаться с ключевого слова **try**. В этом блоке заключены операторы инициализации компонентов приложения **Application→Initialize();**, создания формы приложения **Application→CreateForm(__classid(TForm1), &Form1);** и выполнения приложения **Application→Run();**.

Блок **catch (Exception &exception)** предназначен для включения операторов, вызываемых при возникновении аварийной ситуации. В нашем примере таким оператором является стандартный обработчик исключений **Application→ShowException(&exception);**. В следующих примерах мы рассмотрим, как его использовать. Аналогичная конструкция блоков повторяется для копии приложения, запущенного вторично. Последним оператором тела функции **WinMain** является **return 0;** — оператор, завершающий приложение с кодом 0.

Данный файл можно редактировать, предварительно изучив команды и функции Windows и получив некоторый опыт разработки программ. Поскольку этот файл создается средой разработки автоматически, мы намеренно не будем его изменять, чтобы не отягощать себя лишними проблемами. Тем не менее, при возникновении ошибок компилятора или компоновщика, нелишним будет посмотреть содержимое данного файла и отредактировать его в случае необходимости. Например, при создании новой формы и последующем ее удалении в этом файле могут сохраниться записи о несуществующей форме, приводящие к ошибкам во время трансляции. В таком случае необходимо удалить лишние записи из этого файла — для устранения подобных ошибок.

Далее рассмотрим структуру файла **butct.cpp**. Полное содержимое этого файла приведено ниже.

```
//---------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Ubutct.h"
//---------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
Label1->Caption="Кнопка 1 нажата " + IntToStr(++i) +" раз";
}
//---------------------------------------
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Close();
}
//---------------------------------------
void __fastcall TForm1::FormCreate(TObject *Sender)
{
i=0;
}
//---------------------------------------
```

Большинство строк этого файла нам уже известно из предыдущего описания и самого примера программы. Рассмотрим лишь неизвестные строки. Строка **#pragma package(smart_init)** определяет последовательность инициализации составных частей программы. Строка **#pragma resource "*.dfm"** сообщает препроцессору о том, что для формы необходимо использовать файл ***.dfm** с именем данного файла. В строке **TForm1 *Form1;** объявляется класс **TForm1** для формы **Form1**. Далее следуют подпрог-

раммы создания формы приложения и обработчики событий, рассмотренные выше.

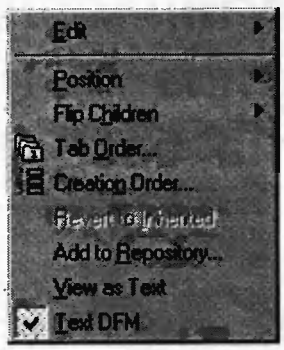
Обратите внимание на строку **#include "Ubutct.h"**, которая подключает к модулю заголовочный файл **Ubutct.h**, представленный ниже.

```
//---------------------------------------
#ifndef UbutctH
#define UbutctH
//---------------------------------------
#include <Classes.hpp>
#include <Controls.hpp>
#include <StdCtrls.hpp>
#include <Forms.hpp>
//---------------------------------------
class TForm1 : public TForm
{
__published:    // IDE-managed Components
        TButton *Button1;
        TButton *Button2;
        TLabel *Label1;
        void __fastcall Button1Click(TObject *Sender);
        void __fastcall Button2Click(TObject *Sender);
        void __fastcall FormCreate(TObject *Sender);
private:        // User declarations
public:         // User declarations
        int i;  // Переменная - счетчик
        __fastcall TForm1(TComponent* Owner);
};
//---------------------------------------
extern PACKAGE TForm1 *Form1;
//---------------------------------------
#endif
```

Ранее мы уже редактировали его в примере, вставляя строку инициализации переменной **i**. Рассмотрим этот файл более детально. Строчка **#ifndef UbutctH** является директивой условной компиляции. Эта директива проверяет, не был ли ранее определен идентификатор **UbutctH** директивой **#define**. Если не был, то предваряемый директивой фрагмент кода компилируется. В противном случае компиляция фрагмента не производится. Затем в строке **#define UbutctH** определяется идентификатор **UbutctH**. Далее следуют четыре строки включения файлов **#include**, в которых описаны компоненты, переменные, константы и функции, которые будут использованы в данном модуле. Сюда можно добавлять дополнительные директивы, не создаваемые средой разработки автоматически. После этих строк следует описание класса формы с тремя разделами. Раздел **__published:** содержит объявления размещенных на форме компонентов и обработчиков событий, которые среда разработки записывает в файл автоматически в процессе проектирования формы. Раздел **private:** предназначен для объявления закрытых переменных, т.е. доступных только для данного модуля. Раздел **public:** предназначен для объявления открытых переменных, т.е. доступных не только для данного модуля, но и для других классов и модулей. Именно в этот раздел мы включили (записали) переменную — счетчик **i**. Строка **extern PACKAGE TForm1 *Form1;** объявляет класс **TForm1** как доступный из других модулей. И завершает файл строка **#endif** окончания условной директивы **#ifndef**.

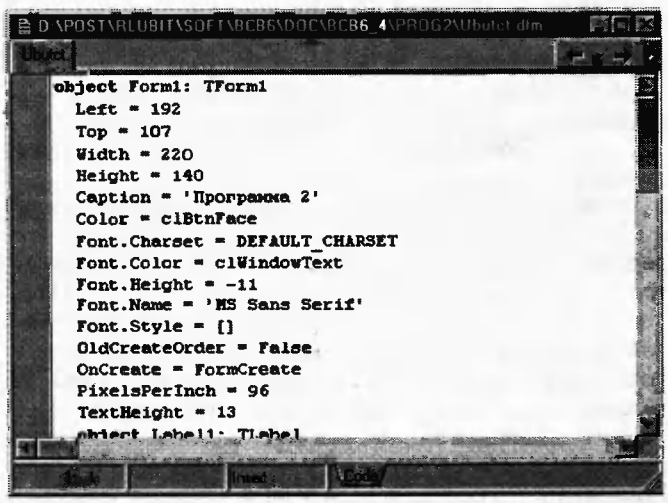
Следующие два файла — **butct.bpr** и **Ubutct.dfm** — представляют собой текстовые файлы установок проекта и формы данного проекта соответственно. Они создаются и редактируются автоматически в процессе разработки и изменения проекта в среде разработки. Файл **butct.bpr** содержит директивы и макрокоманды настроек среды разработки, и, во избежание внесения ошибок, его не следует корректировать вручную, хотя содержимое можно просмотреть с помощью любого текстового редактора. Файл **Ubutct.dfm** представляет собой текстовые записи содержания и свойств формы, его можно просмотреть и даже отредактировать в самой среде разработки. Для этого необходимо щелкнуть левой кнопкой на форме для ее выделения, а затем щелчком правой кнопки мыши открыть контекстное меню (**рис.5**) и выбрать в нем команду **View as Text**.

Рис. 5



Перед вами должно открыться окно формы (**рис.6**), в котором представлено содержимое файла формы в текстовом виде.

Рис. 6



Здесь можно просмотреть и, при необходимости, отредактировать любую запись. Естественно, это можно выполнять, точно представляя, что вы делаете, чтобы не внести новые ошибки. Например, в строках **Left = 192**, **Top = 107**, **Width = 220** и **Height = 140** можно изменить левый и верхний отступ, а также ширину и высоту формы, соответственно. В строчках **Caption = "Программа 2"** и **Color = clBtnFace** можно изменить соответственно заголовок формы и цвет ее фона, и т.д. Вернуться к исходному виду формы можно с помощью комбинации "горячих" клавиш **Alt+F12** или же через контекстное меню по команде **View as Form**.

Файлы **butct.res** и **Ubutct.ddp** являются двоичными файлами, хранящими ресурсы и диаграммы проекта соответственно, и не подлежат ручному редактированию. Исполняемый файл проекта имеет имя **butct.exe**. Наконец, файл с расширениями **tds** является отладочным файлом проекта, который всегда можно создать в среде разработки повторно, имея исходные файлы проекта. Помимо описанных выше файлов, среда разработки позволяет создавать дополнительные файлы, которые являются вспомогательными и не подлежат ручной корректировке.

Этот материал, на первый взгляд, может показаться очень трудным. Но со временем описанные директивы станут привычными и не будут казаться непонятными.

(Продолжение следует)

Минимальное приложение типа Win32Application на базе MFC

А.КОРБИТ,
г.Минск,
БГУИР, ФИТУ, каф.ВМиП.

В журнале был опубликован цикл статей [1], в основу которого были положены лабораторные работы, демонстрирующие основные приемы создания проектов под Windows на базе MFC при помощи AppWizard [2-4]. Каждое такое приложение должно иметь один объект класса, наследованного от CWinApp. Этот класс инкапсулирует основные черты поведения Windows-приложения, т.е. **CWinApp** — это базовый класс, из которого мы должны обязательно создать объект "Приложение" для ОС **Windows**.

CWinApp отвечает за:

- создание процесса в системе;
- инициализацию и запуск приложения **Windows** (например, главного окна);
- организацию и запуск цикла обработки сообщений;
- корректное завершение работы вашего приложения.

Т.е класс **CWinApp** включает в себя выполнение функции WinMain и цикл обработки сообщений.

Таким образом, все приложения на базе **MFC** используют данный класс, несмотря на то что он находится на четвертом уровне иерархии каркаса. Он может быть включен только один раз, и ваш класс, производный от **CWinApp**, должен быть объявлен на глобальном уровне. Это означает, что объект **CWinApp** создается до создания окон, и после этот класс создает поток выполнения **Thread**, за который теперь будет отвечать класс **CWinThread**.

Любое приложение на базе **MFC** может иметь только один объект класса CWinApp или производного от него. Помимо этого базового класса, в **MFC** есть еще несколько глобальных функций (**таблица**).

AfxGetApp	Получает указатель на объект CWinApp
AfxGetInstanceHandle	Получает указатель на текущее приложение
AfxGetResourceHandle	Получает указатель на текущие ресурсы
AfxGetAppName	Получает указатель на строку с именем приложения

Вначале дадим краткую характеристику элементов-данных класса **CWinApp**:

- m_pszAppName — хранит имя приложения;
- m_hInstance — дескриптор текущего приложения;
- m_lpCmdLine — хранит путь на командную строку;
- m_pszExeName — имя исполняемого модуля приложения;
- m_pszHelpFilePath — хранит имя файла помощи;
- m_pszProfileName — хранит имя INI-файла приложения.

В конце статьи будут приведены примеры, иллюстрирующие работу с некоторыми из этих элементов.

Теперь дадим характеристику основных методов класса **CWinApp**.

1. Группа методов, определяющих внешний вид приложения после его инициализации и отображения на экране монитора. Например:

- SetDialogBkColor() — устанавливает цвет фона диалоговых окон;
- Enable3dControls() — разрешает элементам управления иметь 3-мерный вид.

2. Методы для работы с курсором и пиктограммой. Например:

- DoWaitCurcor() — устанавливает курсор в виде песочных часов;
- LoadCursor() — читает курсор из файла ресурсов;
- LoadIcon() — читает пиктограмму из файла ресурсов.

3. Группа виртуальных функций.

InitInstance() — отвечает за инициализацию каждого нового экземпляра приложения. В своем первозданном виде функция не делает ничего.

Ее определение имеет вид

```
virtual BOOL CWinApp::InitInstance()
{
return TRUE
}
```

Заполнение этой функции содержанием целиком является задачей программиста. Это то место в программе, где:

- конструируется объект главного окна приложения;
- загружаются настройки из INI-файла или из реестра Windows;
- обрабатывается командная строка текущего экземпляра приложения;
- регистрируется шаблон документов, создаются новые или открываются существующие документы, формируется их отображение.

Т.е. в код данного метода включаются все действия по созданию ваших оконных объектов.

Фактически метод InitInstance() производит инициализацию экземпляра потока и должен быть обязательно переопределен, для того чтобы ваше приложение что-либо делало!!!

При создании приложения при помощи AppWizard генерируется класс, производный от CWinApp, в котором в области его реализации в InitInstance() вставляется стандартная инициализация экземпляров приложения.

Следующий виртуальный метод имеет декларацию

```
virtual  int Run();
```

и запускает цикл обработки сообщений, который получает и распространяет сообщения, до тех пор пока не получит сообщение WM_QUIT. Это сообщение инициирует завершение работы приложения вызовом виртуальной функции ExitInstance(). Значением этой функции является целое число, возвращаемое функцией WinMain.

Все сообщения направляются методу класса с декларацией

```
virtual BOOL PreTranslateMessage( MSG* pMsg );
```

который производит их специальную обработку (фильтрует сообщения), перед тем как они будут переданы Win32 API функциям TranslateMessage() и DispachMessage() для стандартной обработки. Данный метод вернет значение TRUE, если сообщение было обработано полностью и не нуждается в дальнейшей обработке. В противном случае функция вернет ноль. В качестве параметра методу передается указатель на структуру, которая содержит обрабатываемое сообщение. Метод можно переопределить для организации своего алгоритма фильтрации сообщений.

И еще, если сообщений нет, данный метод периодически вызывает обработчик простоя с декларацией

```
virtual BOOL OnIdle( LONG lCount );,
```

который выполняет обновление элементов интерфейса пользователя (меню, кнопки и т.д.) и очищает внутренние структуры данных. Методу передается целочисленный параметр *lCount,* изначально равный нулю, который увеличивается на единицу при повторном вызове **OnIdle**, если очередь сообщений пуста. Т.е. этот параметр можно использовать для определения относительного временного интервала простоя. В случае поступления очередного сообщения счетчик автоматически сбрасывается в ноль. По умолчанию **OnIdle** возвращает TRUE, что обеспечивает его последующие вызовы, если очередь сообщений оказывается пустой. Итак, данный метод производит обработку во время ожидания процесса, и его обычно не переопределяют.

Следующая виртуальная функция декларируется так:

```
virtual  int ExitInstance();,
```

и служит для выхода из экземпляра процесса. Эту функцию, как правило, переопределяют, если нужно выполнить какую-либо очистку памяти перед завершением работы приложения. Результатом работы функции является целочисленный код, и если он равен нулю, то ошибок нет, любое другое значение означает ошибку. Вызывается данный метод только из функции Run и осуществляет уничтожение объекта после завершения потока.

Таким образом:

1. В приложениях на базе MFC функция WinMain скрыта от программиста в определении класса CWinApp.

2. В каждом приложении определяется главный его класс, наследуемый от CWinApp, который отвечает за работу приложения в целом.

3. Приложение должно иметь только один объект главного класса, и он должен быть глобальным. Только в этом случае этот объект будет создан сразу после запуска приложения и сможет управлять всей работой приложения.

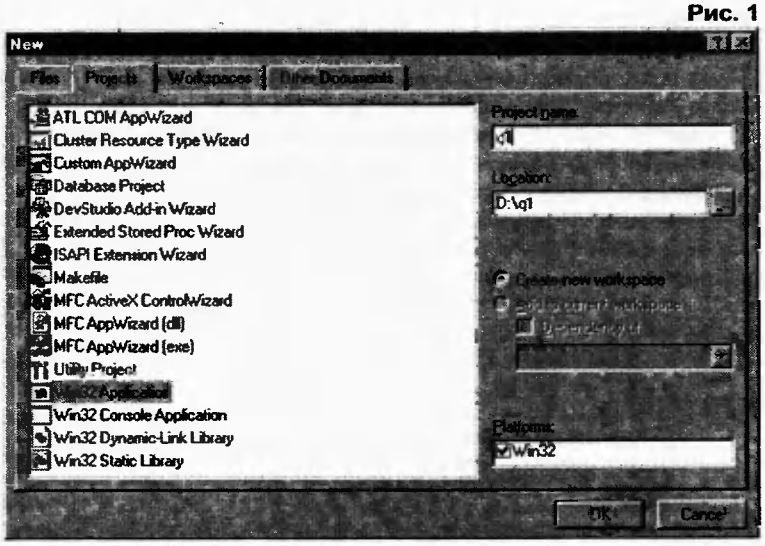
4. Класс имеет виртуальный метод InitInstance(), который необходимо обязательно переопределить, указав действия, составляющие суть вашего приложения. В конце этого метода вызывается оператор return, который, если возвращает TRUE, то после щелчка, например, на стандартной кнопе OK, приложение продолжит работу и приступит, например, к обработке сообщений. Если же вернуть FALSE, то щелчок на OK приведет к завершению работы. По умолчанию возвращается "истина".

На практике приложения создаются не при помощи AppWizard, а как Win32Application на базе MFC. Это позволяет в несколько раз сократить код и увеличить скорость работы программ. Проиллюстрируем это примерами.

Пример 1. Создадим самую короткую программу. Запустим Visual C++, выберем пункты меню **File/New/** и введем имя, например, q1 (**рис.1**). Затем выберем путь для размещения проекта и тип приложения — "**Win32Application**". Подтвердим выбор нажатием **OK**.

Далее выбираем "**An empty project**", щелкаем на "**Finish**", подтверждаем выбор нажатием **OK** и получаем проект с именем q1.

Рис. 1



Теперь идем в пункты меню **Project/Add To Project/New**, выбираем — **C++ Source File** File name:**q1** и **OK**.

Далее заходим в "пустое" окно текстового редактора. Подключаем MFC, для этого в пункте меню **Project/Settings...** вместо **Not Using MFC** выбираем **Use MFC In Shared DLL** и подтверждаем выбор нажатием **OK**.

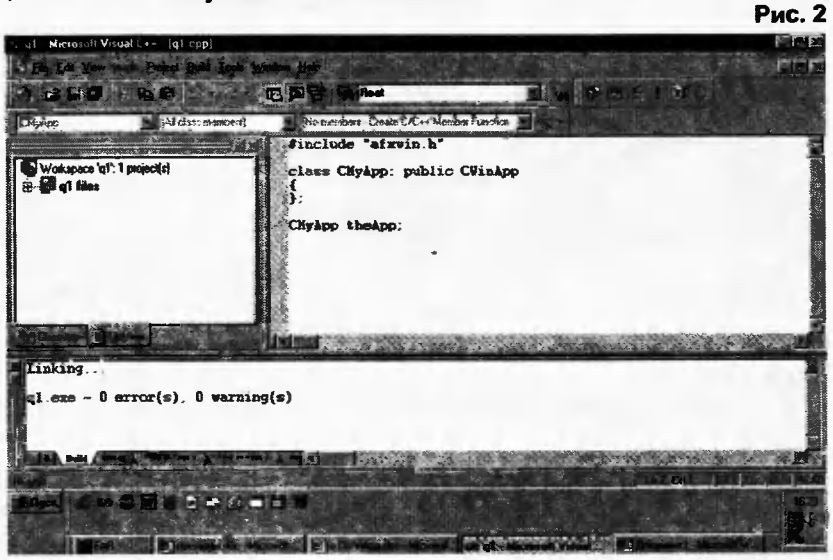
В редакторе набираем следующий текст:

```
#include "afxwin.h"

class CMyApp: public CWinApp
{
};

CMyApp theApp;
```

Запускаем проект — командой Build/Execute q1! или комбинацией клавиш Ctrl-F5. Мы получили приложение, которое только компилируется, выполняется и больше ничего не делает (**рис.2**).

Здесь первая строка подключает заголовочный файл со всеми определениями для библиотеки MFC. В нем автоматически подключаются Windows.h (Object Windows Library). Далее декларируем свой главный класс, наследуя его от CWinApp, и создаем глобальный от него объект. Как видите, функция WinMain благополучно исчезла, она запускается автоматически.

Рис. 2



Чтобы создать работающее приложение, необходимо переопределить виртуальную функцию InitInstance(). Сделаем это во втором примере. Переопределение ее будет заключаться в том, что InitInstance() просто выведет на экран сообщение "Hello MFC". Далее пойдут диалоговые окна со значениями некоторых элементов-данных класса CWiiApp.

Пример 2. Также покажем возможность переопределения ExitInstance. Пусть этот метод запросит подтверждение на завершение работы приложения. Создадим по уже известной методике приложение q2 и набирем текст:

```
#include <afxwin.h>
// подключили все, что нужно для MFC, и декларировали главный класс
class CMyApp:public CWinApp
{
public:
// прототипы функций
      virtual BOOL InitInstance();
      virtual BOOL ExitInstance();
};
// переопределили виртуальные функции
BOOL CMyApp::InitInstance()
      {
// поздоровались с каркасом
            AfxMessageBox("Hello MFC!");
// распечатали имя файла
            AfxMessageBox(m_pszProfileName);
// распечатали имя exe-файла
      AfxMessageBox(m_pszExeName);
// распечатали путь к файлу помощи
            AfxMessageBox(m_pszHelpFilePath);
            return TRUE;
      }
int CMyApp::ExitInstance()
      {
// затребовали подтверждение на завершение работы
      AfxMessageBox("YES?");
      return 0;
      }
// создали объект (глобальный) главного класса приложения
      CMyApp theApp;
```

Если теперь запустить созданное приложение, то получим последовательность диалоговых окон, приведенную на **рис.3...7**.

Рис.3

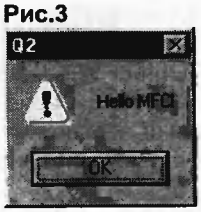


Рис.4

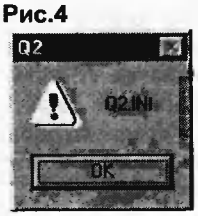


Рис.5

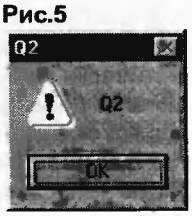


Рис.6

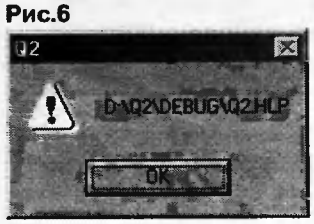
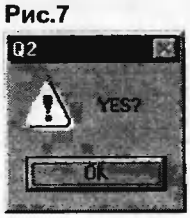


Рис.7



Литература

1. А.Корбит и др. Программирование в среде Visual C++ 6.0. — Радиомир. Ваш компьютер, 2003, №№4-7, 9-12, 2004, №№3-5.

2. Фролов А. Фолов Г. Microsoft Visual C++ и MFC. — М.: Диалог "МИФИ", 1997.

3. Баженова И. Visual C++ 6.0. Уроки программирования.— М.: Диалог "МИФИ", 1999.

4. Глушков С. и др. Программирование на Visual C++ 6.0. — Харьков: "Фолио", 2002.

К.МУЛЯРЧИК,
E-mail: kastysik@tut.by

Изготовитель бэджиков и этикеток

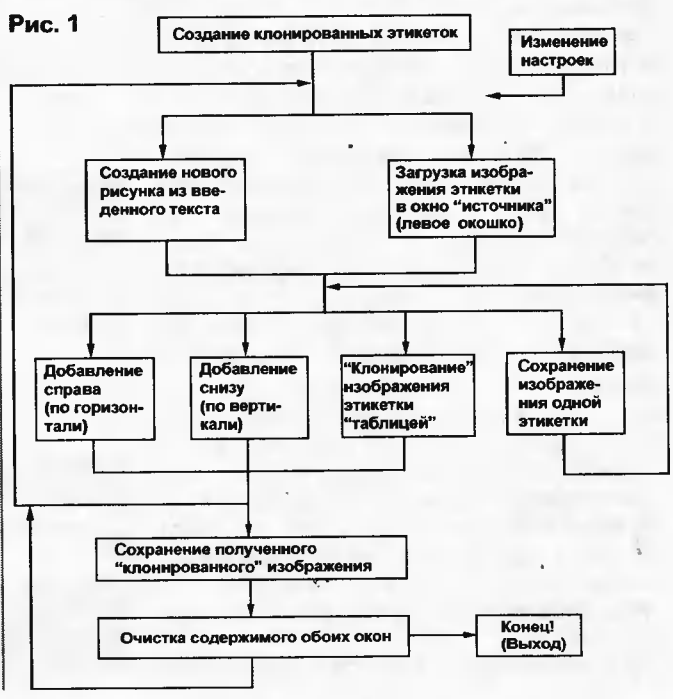
Вполне вероятно, что у многих из вас есть свой частный дом, в котором имеется погреб. И так же вполне вероятно, что не все, у кого есть этот самый погреб, собираются использовать его только как бомбоубежище на случай ядерной войны. Наверняка многие из вас делают домашние заготовки (опять же, не обязательно на случай войны) и хранят их в этом самом потенциальном бомбоубежище (в миру — погребе) у себя дома или на даче. А по виду банки трудно будет различить, что в ней находится, тем более, если объявят тревогу. Для того чтобы распознать свое любимое малиновое варенье в этом множестве банок и не "хватануть" вместо него, например, солидол, подорвав тем самым обороноспособность страны, и написана программа **Labels**. Она создает специальный графический файл с рисунком этикетки, которую можно размножить (клонировать) в любом количестве. Достаточно указать количество строк и столбцов на листе, и в каждой ячейке будет создана эта этикетка. Готовые этикетки вырезаются из бумаги и наклеиваются на банки (не слюной!) в соответствии с их содержимым. Быстро, качественно, безопасно и удобно! Впрочем, эту программу можно применять для любых других целей, например, для создания бэджиков. Более того, прогрвмма позволяет размножать любые изображения.

Описание программы

Программа **Labels** написана на языке программирования **Object Pascal** в среде **Delphi 5**.

На **рис.1** показан алгоритм ее программы. Первоначально создается изображение одной этикетки, которое потом можно размножить в любом количестве. Достаточно указать количество столбцов и строк таблицы, в каждой ячейке

Рис. 1



которой будет содержаться изображение одной этикетки. После "клонирования" полученное изображение можно сохранить и распечатать. Также можно указать максимальную высоту и ширину изображения, получившегося в результате размножения. Программа также "умеет":

- устанавливать фиксированную длину и ширину этикетки;
- изменять отступы от текста до рамки (если размеры не фиксированы);
- изменять промежутки между этикетками в полученном после размножения изображении;
- рисовать рамку вокруг этикетки.

Отличительной особенностью является возможность размножать любые загруженные в программу изображения, а также размножать этикетку не только "таблицей", но и добавлять новые этикетки к получаемому изображению справа (по горизонтали) или снизу (по вертикали). Программа имеет русскоязычный, и, будем надеяться, интуитивно понятный интерфейс.

Что внутри программы? или Секреты исходного кода

Основной внутренний код программы состоит из двух частей — модуля создания изображения одной этикетки (**рис.2**) и модуля, содержащего все функции, связанные с "клонированием". Рассмотрим каждый из этих модулей подробно.

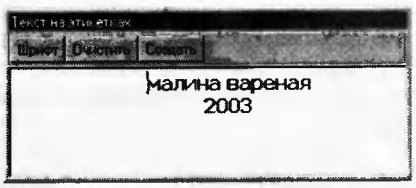


Рис. 2

Модуль по созданию изображения одной этикетки

Для начала взглянем на окошко "Текст на этикетках" и обозначим названия некоторых элементов в нем.

Наша цель — получить изображение надписи "малина вареная 2003" в виде какого-нибудь графического класса, например, **TBitmap**. Этот тип описывает матрицу определенного размера, в каждой ячейке которой записан код цвета, описывающего данную ячейку-пиксел. Мы поступим так — сначала скопируем вид **RichEdit1** в такую матрицу (**bmp**), а потом вырежем "кусочек" матрицы с нашей надписью и поместим в левое окошко (окно источника) основного окна программы. Копирование **RichEdit1** в матрицу(**bmp**) выглядит так:

```
bmp:=TBitmap.Create;
bmp.Width := RichEdit1.Width-i;
bmp.Height := RichEdit1.Height-i;
BitBlt(bmp.Canvas.Handle, 0, 0, RichEdit1.Width-i,
RichEdit1.Height-i,
GetDC(RichEdit1.Handle), 0, 0, SRCCopy);
```

Копированием изображения в изображение занимается особая процедура **BitBlt**. Она выполняет поблочную пересылку цветовых данных соответствующих прямоугольнику пикселов от указанного исходного объекта в объект адресата. Описание параметров BitBlt:

```
function BitBlt(DestDC: HDC; X, Y, Width, Height: Integer;
SrcDC: HDC; XSrc, YSrc: Integer; Rop: DWORD): BOOL;
```

Здесь:

- `DestDC` — указывает на "источник";
- `x, y` — координаты левого верхнего угла прямоугольника адресата;
- `Width, Height` — ширина и высота прямоугольника пикселов;
- `SrcDC` — указывает на объект адресата;
- `XSrc, Ysrc` — координаты левого верхнего угла исходного прямоугольника;
- `Rop` — код растровой операции. Эти коды определяют, как цветовые данные исходного прямоугольника должны быть объединены с цветными данными прямоугольника адресата для достижения конечного цвета (**см. таблицу**).

Значение	Описание
BLACKNESS	Заполняет прямоугольник адресата, используя цвет, связанный с индексом 0 в физической палитре (это черный цвет для заданной по умолчанию физической палитры)
DSTINVERT	Инвертирует прямоугольник адресата
MERGECOPY	Объединяет цвета исходного прямоугольника с определенной палитрой, используя логический оператор AND
MERGEPAINT	Объединяет цвета инвертированного исходного прямоугольника с цветами прямоугольника адресата, используя логический оператор OR
NOTSRCCOPY	Копирует инвертированный исходный прямоугольник в "адресат"
NOTSRCERASE	Комбинирует (объединяет) цвета прямоугольников источника и адресата, используя логический оператор OR, и затем инвертирует результирующий цвет
PATCOPY	Копирует определенную палитру в прямоугольник адресата
PATINVERT	Комбинирует (объединяет) цвета определенной палитрой с цветами прямоугольника адресата, используя логический оператор XOR
PATPAINT	Комбинирует (объединяет) цвета определенной палитры с цветами инвертированного исходного прямоугольника, используя логический оператор OR. Результат этой операции объединяется с цветами прямоугольника адресата, используя логический оператор OR
SRCAND	Комбинирует (объединяет) цвета прямоугольников источника и адресата, используя логический оператор AND
SRCCOPY	Копирует исходный прямоугольник непосредственно в прямоугольник адресата
SRCERASE	Комбинирует (объединяет) инвертированные цвета прямоугольника адресата с цветами исходного прямоугольника, используя логический оператор AND
SRCINVERT	Комбинирует (объединяет) цвета прямоугольников источника и адресата, используя логический оператор XOR
SRCPAINT	Комбинирует (объединяет) цвета прямоугольников источника и адресата, используя логический оператор OR
WHITENESS	Заполняет прямоугольник адресата, используя цвет, связанный с индексом 1 в физической палитре (это белый цвет для заданной по умолчанию физической палитры)

Если функция выполнилась успешно — возвращается значение `true`, в противном случае — `false`, тогда для получения расширенной информации об ошибке вызовите `GetLastError`.

Чтобы вырезать нужный нам прямоугольник и не промахнуться, сначала определим координаты левого верхнего и правого нижнего углов искомого прямоугольника. Для этого воспользуемся следующим методом — будем сканировать столбцы и строки матрицы bmp до тех пор, пока в них не останется черных элементов. Сканировать

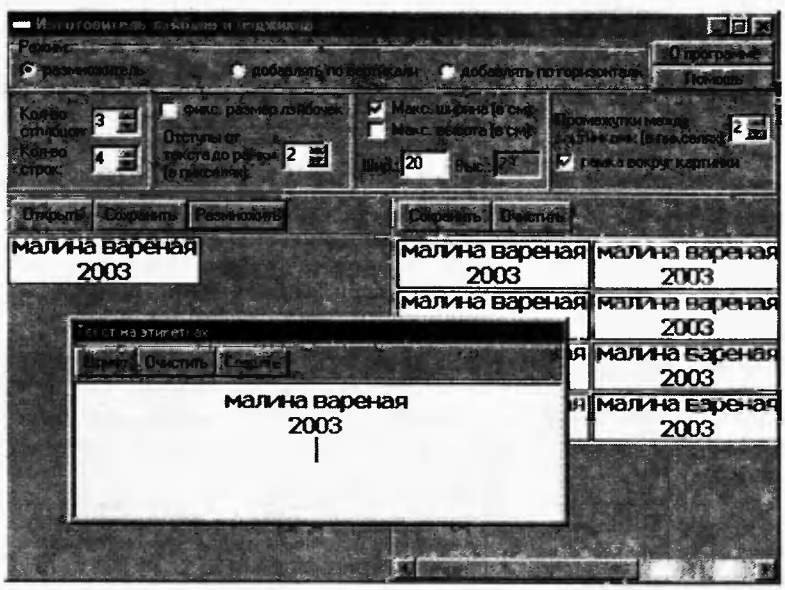
будем слева направо, сверху вниз и справа налево и снизу вверх. Но для начала нам необходимо определить размеры сканируемой матрицы по вертикали (**y0**) и по горизонтали (**x0**):

```
x0:=bmp.Canvas.ClipRect.Right-bmp.Canvas.ClipRect.Left;
y0:=bmp.Canvas.ClipRect.Bottom-bmp.Canvas.ClipRect.Top;
```

А вот и непосредственно само сканирование:

```
for y:=0 to y0 do
 for x:=0 to x0 do if bmp.Canvas.Pixels[x,y]=clBlack then begin
  R.Bottom:=y;
  break;
 end;
for y:=y0 downto 0 do
 for x:=0 to x0 do if bmp.Canvas.Pixels[x,y]=clBlack then begin
  R.Top:=y;
  break;
 end;
for x:=0 to x0 do
 for y:=0 to y0 do if bmp.Canvas.Pixels[x,y]=clBlack then begin
  R.Right:=x;
  break;
 end;
for x:=x0 downto 0 do
 for y:=0 to y0 do if bmp.Canvas.Pixels[x,y]=clBlack then begin
  R.Left:=x;
  break;
 end;
```

Рис. 3



Теперь мы в переменной **R** типа **TRect** получили координаты левого верхнего и правого нижнего углов прямоугольника с текстом. Тип **TRect** описывает координаты левого верхнего и правого нижнего углов произвольного прямоугольника, и достаточно удачно применяется во многих случаях. Теперь осталось вырезать из матрицы **bmp** прямоугольник с текстом и скопировать его в новую матрицу **Lab_bmp**:

```
…
Lab_Bmp:=Tbitmap.Create;
…
BitBlt(Lab_Bmp.Canvas.Handle, 0, 0, R.Right-R.Left+1,
R.Bottom-R.Top+1,bmp.Canvas.Handle, R.Left, R.Top, SRCCopy);
```

А скопировать полученное изображение текста из Lab_bmp в окошко источника основного окна программы уже не составляет труда (**рис.3**).

Модуль клонирования

Итак, после нехитрых манипуляций мы имеем в окошке источника основного окна программы изображение с текстом. Ну что же, будем его клонировать. Снова создадим переменную (матрицу) **Bmp** типа **TBitmap**, и в нее будем копировать изображение этикетки. Здесь переменная Bmp — это не та, что была раньше, а новая. Дело в том, что они находятся в разных модулях, и никак друг на друга не влияют. Определив высоту (**h**) и ширину (**w**) изображения текста и зная количество выходных столбцов (**nx**) и строк (**ny**) виртуальной таблицы размножения, определим высоту (**bh**) и ширину (**bw**) выходного изображения, учитывая промежутки (**d**) между изображениями в пикселах:

```
bw:=nx*w+(nx+1)*d;
bh:=ny*h+(ny+1)*d;
```

А теперь процедура клонирования сводится к двум циклам и точному определению координат очередного места, куда будет вставлено изображение текста, в выходной картинке:

```
for b:=1 to ny do
 for a:=1 to nx do
  Bmp.Canvas.Draw((w+d)*(a-1)+d,(h+d)*
 (b-1)+d,Form1.Frame2.Image1.Picture.Graphic);
```

Не пугайтесь Form1.Frame2.Image1.Picture.Graphic. Это ссылка на матрицу выходного изображения, куда производится копирование изображения этикетки.

Перейдем к добавлению по вертикали. Для того чтобы добавить изображение к имеющемуся выходному файлу, необходимо проделать следующее:

- расширить изображение-приемник (**Bmp**) так, чтобы снизу поместилось вставляемое изображение:

```
Bmp.Height:=Bmp.Height+h+d;
if Bmp.Width<w then Bmp.Width:=w;
```

- скопировать необходимое изображение в изображение-приемник:

```
T:=Bmp.Height;
Bmp.Canvas.Draw(0,t+в,Form1.Frame2.Image1.Picture.Graphic);
```

С добавлением по горизонтали решительно ничего не меняется, только необходимо высоту заменить на ширину и наоборот.

На **рис.4** показан результат работы программы.

Рис. 4

малина вареная 2003	малина вареная 2003	малина вареная 2003
малина вареная 2003	малина вареная 2003	малина вареная 2003
малина вареная 2003	малина вареная 2003	малина вареная 2003

Исходные тексты проекта, а также исполняемый файл программы с необходимыми пояснениями выложены в соответствующих архивах labels_source.zip и labels_exe.zip на веб-странице журнала в приложении к статье. Последнюю версию этой программы можно скачать с моей веб-страницы http://www.kastys.by.ru/labels/.

ИССЛЕДОВАНИЕ ПРОГРАММ

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

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ КРЭКИНГА

(Продолжение. Начало в №№8-12/2003, 1-9/2004)

В свое время самым популярным отладчиком для "Спектрума" был MONS (впрочем, некоторые люди, включая меня, предпочитали MON) — восьмикилобайтное порождение программистской мысли, способное загружаться в ОЗУ с любого адреса и управляемое из командной строки (прямо как SoftIce — внешнее сходство этих двух отладчиков вообще трудно не заметить). И, разумеется, MONS позволял ставить брейкпойнты — не имея в своем арсенале такой возможности, этот отладчик вряд ли стал бы столь популярен. Но поскольку процессор Z80, на основе которого был сделан "Спектрум", никаких отладочных средста не предоставлял, авторам MONS пришлось реализовывать точки останова чисто программными средствами. Реализация эта красотой отнюдь не блистала — "установка брейкпойнта" по-Спектрумовски заключалась в подстановке в нужное место кода трехбайтной команды CALL xxxx, которая передавала исполнение в недра самого отладчика и таким образом приостанавливала исполнение пользовательского кода. Старые команды, код которых затирался брейкпойнтом, копировались в специальный буфер и дополнялись командой JP (аналог jmp из набора команд x86) для возврата к следующей команде, не испорченной CALL'ом. Исполнение в пошаговом режиме выглядело не менее оригинально — исполняемая команда перебрасывалась в буфер отладчика, дополнялась все тем же JP, после чего отладчик передавал управление в этот буфер. Если учесть еще то, что в Z80 существовали недокументированные команды, которые были известны далеко не всем отладчикам (и потому могли обрабатываться некорректно), что отлаживаемая программа даже при абсолютно корректной работе могла испортить код отладчика, а под сам отладчик могло элементарно не хватить свободной памяти, и потому его загружали на место "ненужных" данных — вы поймете, что представляла собой отладка в старые добрые времена.

Разработчики линейки x86 проявили больше заботы о программистах. В этой линейке процессоров вместо самодельной "затычки" в виде команды вызова подпрограммы, для отладочных целей ввели отдельное прерывание за номером 3, которое вызывалось однобайтной командой (опкод команды int 3 — CC), в отличие от всех прочих прерываний, которые менее чем двумя байтами вызвать не получится. Другим полезным нововведением стала возможность исполнять код в пошаговом режиме через управление флагом трассировки (эта возможность, впрочем, мало актуальна для отладчиков пользовательского уровня под современные ОС). Однако, несмотря на такой, казалось бы, очевидный прогресс в развитии средств отладки, обыкновенные точки останова все так же, как и десятилетия назад, модифицируют исполняемый код, а потому легко обнаруживаются даже простейшими способами, например, проверкой контрольной суммы всех байтов (не говоря уже о CRC32 и использовании иных, еще более сложных и надежных хэш-функций). Самостоятельно убедиться в том, что точки останова модифицируют код, вы можете за считанные секунды — откомпилируйте при помощи любого ассемблера следующие две строчки, возьмите OllyDebug и загрузите в него откомпилированный код.

```
addr1: mov eax,addr1
mov al, byte ptr [eax]
```

Если вы просто выполните этот код в пошаговом режиме, то в регистре AL окажется число 0B8h. В этом нет ничего удивительного, B8 — это опкод команды mov eax, <число>. А теперь попробуйте поставить брейкпойнт на команду mov eax,addr1, и снова отрассируйте этот код. После выполнения второй команды вы увидите, что в регистре AL находится число 0CCh, хотя код в окне отладчика внешне совершенно не изменился (если, конечно, не считать изменением подсветку адреса, на который поставлен брейкпойнт). Самое интересное, что отладчики могут "приукрашать реальность" не только в окне кода, но и при просмотре данных.

Давайте проделаем еще один весьма поучительный в этом смысле эксперимент — загрузим наш пример из двух команд, поставим точку останова на первую и запишем адрес этой точки останова. Затем берем InqSoft Window Scanner и читаем байт по записанному адресу. Получаем, разумеется, 0CCh. А теперь взглянем на ту же область глазами отладчика (в OllyDebug это пункт меню Follow in dump|Selection) — и очень сильно удивимся. Отладчик показывает нам вовсе не то, что реально читается из памяти в регистр AL, а то, что должно было бы находиться по указанному адресу, если бы мы не поставили точку останова. Но и это еще не все! Посмотрите на динамические подсказки под окном кода — там-то как раз содержимое памяти отображается как надо.

Вот так "умные" отладчики помогают самообманываться начинающим крэкерам — отсутствие видимых изменений в коде наводит человека, не знакомого с тайнами устройства брейкпойнтов, на мысль о том, что прерывание исполнения программы в точке останова происходит по воле неких таинственных сил, с которыми отладчик находится в мистической связи. Хотя на самом деле "классические" точки останова — это не что иное как обычные memory patch'и, а потому и обнаруживаются они теми же самыми способами, что и любые другие исправления в коде.

Кстати, из того, что обычный (не аппаратный) брейкпойнт является ничем иным как исправлением программы, существует одно интересное следствие. Дело в том, что SoftIce'y, в общем-то, без разницы, каким образом в программе появилась команда int 3 — главное, что он может на этой команде остановиться не хуже, чем на настоящем брейкпойнте. А после того как отладчик остановится, можно внести любые поправки в содержимое регистра EIP и код программы, после чего продолжить исполнение как ни в чем не бывало (в OllyDebug, кстати, тоже можно проделать такую операцию при помощи пункта меню New origin here). Польза от такого эрзац-брейкпойнта (после срабатывания которого, к тому же, нужно вручную восстанавливать код, который находился на месте int 3, и корректировать EIP), на первый взгляд кажется весьма сомнительной, но она есть. Я уже упоминал глюк в SoftIce, когда отлаживаемая программа после загрузки Symbol loader'ом начинает немедленно выполняться, хотя крэкеру хотелось бы ее в этот момент притормозить. Так вот, если в Entry point исполняемого файла воткнуть опкод 0CCh, у подопытной программы не будет ни единого шанса избегнуть процесса отладки — поскольку первой командой окажется наш int 3, принудительно активирующий отладчик.

Теперь, когда мы знаем, что точки останова обнаружить можно (и даже знаем, как их можно обнаружить), вернемся к основному вопросу этой главы: "Почему точки останова не срабатывают?". В нашем случае этот вопрос можно даже конкретизировать: "Какими способами подопытная программа может удалить из себя точку останова?". В различных источниках мне неоднократно встречалось предложение использовать для этой цели коды коррекции ошибок, предваряя все "критичные ко взлому" участки программы вызовом функции проверки и восстановления кода. В случае изменения кода из-за появления точек останова, процедура восстановления должна откорректировать "неправильные" байты. Теоретически такая схема вполне возможна, но на практике алгоритмы коррекции ошибок довольно сложны в реализации и не слишком производительны, так что "народные массы" эту идею не приняли. А вот более простой вариант восстановления кода из "резервной копии", расположенной в другом конце программы (или прямого вызова этой резврвной процедуры вместо основной), таки имел место во времена ДОСа; впрочем, я не удивлюсь, если выяснится, что такой прием до сих пор в ходу — реализация очень проста, а какой-никакой эффект все-таки имеется.

На практике дело обстоит еще хуже — для удаления некоторых точек останова не нужны ни коды коррекции ошибок, ни резервные копии. И именно к таким точкам останова относятся всеми нами любимые BPX'ы на вызовы функций WinAPI (и, если смотреть шире, на вызовы практически любых функций). Поскольку "брейкпойнт на функцию" — это на самом деле всего лишь брейкпойнт на первый

байт этой функции, самый простой из приемов, удаляющих точки останова, выглядит следующим образом — заранее узнать адреса нужных функций при помощи GetProcAddress, прочитать их первые байты (если речь идет о внутренних функциях программы — то просто прочитать содержимое соответствующей ячейки) и сохранить значения этих эталонных байтов. Затем перед особо критичными вызовами нужно лишь сравнивать первые байты процедур с эталонным, и, если обнаружится несоответствие, восстанавливать их. Сам факт того, что процедура начинается с опкода 0CCh, говорит о том, что на эту процедуру поставлена точка останова, что может побудить программу предпринять некоторые действия по самозащите. Если учитывать, что многие процедуры начинаются стандартной последовательностью команд push ebp; mov ebp, esp (в шестнадцатеричном редакторе эти команды выглядят как последовательность 55 8B EC), то "действия по самозащите" могут быть простой записью в первые три байта процедуры той самой стандартной последовательности 55 8B EC. После этой операции точка останова, разумеется, исчезнет. Конечно, выявить защиту от брейкпойнтов, основанную на проверке содержимого неких адресов в памяти, не слишком сложно — нужно лишь поставить аппаратную точку останова на чтение/запись первого байта функции и посмотреть, где этот "капкан для защит" сработает.

Другой способ постановки бряков на импортированные из DLL функции основан на том, что вызов импортированной функции почти всегда выполняется не напрямую, а через "переходник". На практике вызовы через "переходник" обычно выполняются одним из двух способов.

Первый способ:

```
call <переходник_к_MyFunc> ; Вызов функции API
…
переходник_к_MyFunc: jmp MyFunc
(этот способ вызова функций наиболее распространен;
переходники вида "jmp истинный_адрес_функции" обычно собраны
в конце программы)
```

Второй способ:

```
mov edi, dword ptr ds:[элемент_в_таблице]
call edi
…
элемент_в_таблице: dd <истинный_адрес_функции_MyFunc>
 (данная техника вызова функций обычно используется в
продуктах Microsoft)
```

Идея заключается в том, что в первом случае точку останова можно поставить не на первый байт функции, а на переходник, через который вызывается эта функция. В первом случае это будет обычный BPX на адрес команды jmp MyFunc, во втором случае придется прибегнуть к аппаратной точке останова на чтение двойного слова по адресу "элемент_в_таблице". Поскольку это не будут брейкпойнты на функцию в прямом смысле слова, этот метод имеет одно существенное ограничение — если нужная функция вызывается не через "переходник", а непосредственно по значению ее адреса (получаемому, например, при помощи GetProcAddress), то такой брейкпойнт, понятное дело, не сработает. Разумеется, также существует возможность, что программа попытается проверить целостность "переходников", но как такие попытки обнаруживать, вы уже знаете.

Если некая процедура вызывается внутри программы стандартным образом, то большинство современных компиляторов генерирует последовательность команд push для помещения параметров функции на стек, собственно переход к процедуре выполняется при помощи команды CALL, а после того как функция отработает, управление возвращается на команду, следующую за CALL. Однако программист, имеющий достаточно высокую квалификацию, может внести заметное разнообразие в эту картину при помощи "ручного" вызова функций средствами ассемблера. Хотя великий Intel завещал нам вызывать процедуры и функции при помощи специально для этого придуманной команды CALL, отдельным гражданам закон не писан (надо отметить, в число этих граждан входят не только авторы защит, но и любители предельной оптимизации, а также фанаты нетрадиционного программирования). Эти странные граждане сочинили несколько комбинаций, имитирующих работу несчастной команды CALL, и эти имитации давно и прочно вошли в арсенал разработчиков защит. Из универсальных нетрадиционных средств вызова подпрограмм прежде всего нужно назвать следующие:

```
push <адрес возврата>
jmp <адрес процедуры>
или
push <адрес возврата>
push <адрес процедуры>
ret
```

Более сложные техники неявной передачи управления основаны на умышленном создании и обработке исключительных ситуаций, вызове прерываний и эксплуатации особенностей конкретных ОС. Эти техники сами по себе представляют весьма значительный интерес — с точки зрения как крэкера, так и программиста, однако их количество практически бесконечно, а сложность нередко выходит далеко за пределы "основ". Чтобы вы имели представление о том, насколько обширна эта тема, сообщу, что любые функции WinAPI (да и вообще любого другого API), в параметрах которых фигурирует callback-функция, могут служить инструментом неочевидного вызова пользовательского кода.

Вместо рассмотрения всего этого бесконечного разнообразия возможных приемов (большинство из которых вы, возможно, вообще никогда не встретите) мы углубимся в исследование возможностей приведенной выше пары базовых "заменителей CALL", понимание которых в итоге дает ключ к "раскалыванию" многих других способов неочевидного вызова процедур. Прежде всего следует отметить, что помещение на стек адреса возврата в этих методах отделено от собственно вызова процедуры, что позволяет вклинить между двумя этими действиями практически любой код, например, кусок вызываемой процедуры — и, соответственно, вызывать эту процедуру не с первого байта, а "с середины". Например, вот таким образом:

```
push <адрес возврата>
push ebp
mov ebp, esp
jmp MyProc+3 ; (1)
…

MyProc:
push ebp
mov ebp,esp
… ; При вызове процедуры в точке (1) будет выполнен переход в эту точку
```

Как видите, хотя приведенный кусок кода по функциональности полностью аналогичен тривиальному call MyProc, явным образом процедура MyProc нигде не вызывается. Причем при помощи макросов можно добиться того, что все вызовы процедуры MyProc в программе будут выглядеть именно таким образом! Так что, сколько бы вы ни ставили брейкпойнтов на адрес MyProc, ни один из них никогда не сработает — по той простой причине, что управление на этот адрес просто никогда не передается, хотя все внешние признаки могут говорить о том, что процедура MyProc успешно отработала. Впрочем, обмануть такой защитный прием обычно не составляет никакой сложности — нужно лишь ставить точку останова не на первую команду в процедуре, а где-нибудь подальше, например, после третьей (можно даже на команду выхода из процедуры, но при этом следует помнить, что процедура может содержать несколько точек выхода) или вообще в какой-нибудь подпрограмме, вызываемой этой процедурой (вторым способом я нередко пользуюсь, когда ставлю точки останова на функции WinAPI).

Другая проблема, которую порождают нетрадиционные способы вызова процедур, заключается в том, что адрес возврата может быть совершенно любым, а не только адресом команды, следующей за командой вызова. Этот прием встречается довольно часто, когда автор защиты хочет скрыть адрес какой-либо функции, к примеру, проверки корректности введенного серийного номера. Рассуждая логически, он понимает, что необходимо максимально осложнить крэкеру нахождение связи между появлением сообщения о неверном серийнике и процедурой, этот серийник проверяющей. А поскольку традиционным приемом поиска такой процедуры является BPX MessageBox с последующим наблюдением, куда вернется программа из MessageBox'а — программист делает вывод, что хорошо бы сделать так, чтобы программа вернулась "не туда", т.е. как можно дальше от процедуры проверки серийника. В этом случае, даже поставив брейкпойнт "куда надо", мы не узнаем, по какому поводу был произведен вызов функции — на стеке будет лежать совершенно другой адрес возврата. И особенно неприятно для начинающего крэкера, когда в качестве адреса возврата ока-

зывается что-нибудь вроде адреса функции ExitProcess.

В общем случае "победить" такой прием можно либо через долгую медитацию с массированным применением шестнадцатеричного редактора/дизассемблера для поиска всех точек, в которых программа явно или неявно оперирует адресами нужных функций WinAPI, либо через поиск "модифицированным методом Ньютона", описанным в предыдущей главе, либо, применив средства трассировки кода, имеющиеся в SoftIce или OllyDebug. Собственно, трассировка в таких случаях является орудием, по своим свойствам приближающимся к термоядерной бомбе — редкий код способен выдержать такой удар, но чтобы получить желаемый эффект, требуется весьма серьезное техническое обеспечение (процесс трассировки требует немалых объемов памяти и достаточно быстрого процессора) и грамотный выбор области применения. Так что, прежде чем пускать в ход "тяжелое вооружение", есть смысл подумать о решении проблемы более простыми средствами.

Одним из таких более простых средств является исследование содержимого стека на предмет "застрявших" в нем полезных данных и адресов. Суть метода заключается в следующем — любые кусочки кода в программе существуют не сами по себе, а находятся во взаимодействии с чем-то, и каждая из процедур может быть как вызываемой (из процедуры более высокого уровня), так и вызывающей "подчиненные" ей процедуры. И даже когда программист скрыл точку вызова конкретной процедуры (что, как я продемонстрировал, не так уж сложно), спрятать от пытливого взора следы, оставленные выше- и нижележащими процедурами, ему могло и не удаться. А если "могло не удаться" — есть смысл попробовать отыскать эти следы.

По традиции, для начала разберемся, что эти следы собой представляют. Представьте себе широко распространенную ситуацию — A→B→C, где "A→B" расшифровывается как "процедура A вызывает процедуру B". При вызове процедуры B стандартными средствами, т.е. командой CALL, в момент начала исполнения процедуры B на вершине стека будет лежать адрес возврата из процедуры B в процедуру A. Аналогичный процесс происходит при вызове процедуры C процедурой B. Получается, что если мы поставим брейкпойнт на точку входа в процедуру C, мы в этот момент сможем наблюдать в стеке следующую картину (вершина стека — вверху):

Адрес возврата из процедуры C в процедуру B
Адрес возврата из процедуры B в процедуру A

Немного усложним картину, и допустим, что в процедуры B и C передаются некие параметры (порядок передачи параметров для нас в данном примере несущественен). Стек в этом случае будет выглядеть следующим образом:

Адрес возврата из процедуры C в процедуру B
Параметры, переданные в процедуру C
Адрес возврата из процедуры B в процедуру A
Параметры, переданные в процедуру B

На практике процедуры обычно занимаются чем-то более сложным, чем простой вызов других процедур с параметрами, а потому довольно часто резервируют на стеке место под локальные переменные. Допустим, что процедуры A и B используют покальные переменные, место под которые выделяется на том же стеке, и посмотрим, что после этого будет твориться в стеке:

Адрес возврата из процедуры C в процедуру B
Параметры, переданные в процедуру C
Область локальных переменных процедуры B
Адрес возврата из процедуры B в процедуру A
Параметры, переданные в процедуру B
Область локальных переменных процедуры A

А теперь представим, что автор защиты на этапе B→C подменил адрес возврата из процедуры C в процедуру B своим собственным значением, и возврат теперь происходит не в B, а некую процедуру D. Что от этого изменится? Да только одна, самая верхняя строчка! А вот адрес возврата в процедуру A, локальные переменные и параметры вызова какими были, такими и останутся, и это можно использовать в качестве зацепки, позволяющей выявить все этапы пути от процедуры A к процедуре C. Другое дело, что информация, лежащая в стеке, никак не структурирована, поэтому вам придется самому угадывать, что там — локальные переменные, что — параметры вызовов, а что — адреса возврата. И хотя процесс проверки этих догадок может быть весьма трудоемким, лучше иметь хотя бы такую беспорядочную информацию, чем не иметь никакой. Иногда бывает полезно посмотреть, что находится выше вершины стека — там нередко тоже удается обнаружить следы деятельности процедур, отработавших перед тем как мы остановили программу.

(Продолжение следует)

Заметки о технологии Hyper-Threading

П.СОКОЛЕНКО /HI-TECH,
г.Ростов-на-Дону,
E-mail: pts@aaanet.ru
http://wasm.ru
http://www.macro.aaanet.ru

(Окончание. Начало в №9/2004)

Программирование

Микропроцессор Pentium 4HT относится к категории многопроцессорных систем, состоящих из одинаковых процессоров, использующих общее пространство памяти (SMP). Методы программирования таких систем хорошо известны, они описаны в литературе и преподаются в вузах студентам соответствующих специальностей. Однако, в отличие от традиционных SMP-систем, в данном случае существуют два логических процессора, разделяющие не только общую память, но и ресурсы одного физического процессора. Поэтому программирование для Pentium 4, работающего в режиме Hyper-Threading (HT), имеет свои специфические особенности, о чем и пойдет речь в этой части статьи.

Системное программное обеспечение

Как обычно, все начинается с BIOS (базовой системы ввода-вывода), находящейся на материнской плате ПК. Если в Setup вы обнаружили активный параметр (option) enable/disable Hyper-Threading, то микропроцессор и BIOS соответствуют друг другу. Необходимая версия BIOS находится на всех материнских платах, рассчитанных на установку Pentium 4 HT. В процессе загрузки компьютера BIOS определяет количество логических процессоров у МП и формирует соответствующую запись в специальной таблице Advanced Configuration and Power Interface (ACPI), с которой в дальнейшем работает операционная система (ОС). Если в Setup запрещены режим HT и создание таблицы ACPI, то ОС будет обслуживать только один физический процессор.

Традиционные ОС, предназначенные для обслуживания SMP-систем, рассчитаны на работу с несколькими физическими, а не логическими процессорами. Для работы с логическими процессорами необходимы изменения на абстрактном уровне оборудования (Hardware Abstraction Layer — HAL), позволяющие использовать сформированную BIOS таблицу ACPI. В технической документации рекомендуются две ОС, поддерживающие работу как с физическими, так и с логическими процессорами — это Windows XP Professional и Linux с ядром 2.4.XX, где XX больше или равно 19.

Для управления рабочими станциями, содержащими два МП Xeon с поддержкой технологии HT, используется ОС Windows

2000 Professional SP2. Судя по опубликованным результатам тестирования, она успешно справляется со своими задачами. Тем не менее, сотрудники Intel не рекомендуют использовать ее для управления МП Pentium 4 HT.

Многопроцессорные ОС способны планировать выполнение заданий (задач) и распределять общедоступные ресурсы между физическими и логическими процессорами так, чтобы достигалась максимальная производительность. А в поддерживаемый ими интерфейс прикладных программ (API) входят запросы, которые позволяют прикладным программам управлять процессом своего выполнения, запускать дочерние процессы и выполнять некоторые другие действия, специфические для работы в многопроцессорных системах.

Поведение некоторых задач для МП Pentium 4 зависит от того, выполняются они автономно или параллельно с другими задачами. Это значит, что задача содержит группы инструкций (участки кода), которые должны выполняться только при поддержке технологии HT. В процессе работы задача может получить требуемую информацию, издав инструкцию CPUID. Если в регистр EAX предварительно записана "1", то после выполнения CPUID, в случае поддержки технологии HT, 28-й разряд регистра EDX установлен (содержит "1"), в противном случае он очищен (содержит "0"). Кроме того, при поддержке HT в разрядах 16....23 регистра EBX указано количество существующих логических процессоров.

Параллельное выполнение процессов

В любой момент времени Pentium 4 HT может "одновременно" выполнять один или два вычислительных процесса. Во втором случае обычно говорят о параллельном выполнении процессов, нитей или цепочек инструкций. При выполнении прикладных задач возможны две категории таких процессов:

- одновременно выполняются две независимые задачи, каждая из которых представляет собой один неделимый процесс. Обычно инициатором выполнения таких задач является оператор, он может работать в интерактивном режиме, или заранее готовить пакет заданий, подлежащих исполнению.

- одновременно выполняются два процесса, имеющие общего родителя. Это означает, что в некоторой точке один последовательный процесс делится на два, выполняемых параллельно. Косвенным инициатором таких процессов является программист, создающий задачу, предназначенную для выполнения в многопроцессорной системе.

Кроме того, существуют процессы, порождаемые ОС. Они могут быть связаны с началом и прекращением выполнения заданий или с исполнением запросов прикладных задач. Это неизбежные накладные расходы (overhead), зависящие от особенностей ОС и выполняемых заданий.

Суммарное время выполнения двух процессов (**T**) в общем случае состоит из трех частей: $T = T_s + T_p + T_o$, где:

- T_s — время, в течение которого, по тем или иным причинам, выполнялся только один процесс;

- T_p — время, в течение которого выполнялись оба процесса;

- T_o — время, затраченное на выполнение системных процессов.

Наибольшая производительность МП достижима только при одновременном выполнении двух процессов, поэтому чем больше отношение T_p/T, тем лучше.

В общем случае производительность МП зависит от многих факторов. Но если две задачи выполнить в разных режимах работы Pentium 4, то можно определить влияние технологии HT на изменение производительности МП.

Условимся измерять продолжительность выполнения задач в количестве тактов работы МП, тем более, что это количество можно определить программно. Допустим, что при выключенной поддержке HT одна задача выполняется за n_1, а вторая — за n_2 тактов. В таком случае суммарная продолжительность их выполнения — $N = n_1 + n_2$ тактов. При параллельном исполнении обеих задач в режиме HT она составит $N - k$ тактов. Очевидно, что если $k > 0$, то при поддержке режима HT суммарное количество тактов, необходимых для выполнения обеих задач, сокращается. А это значит, что в течение **k** тактов Pentium 4 одновременно исполнял микрооперации, порожденные двумя процессами. Вычислив отношение **k/N** и умножив его на 100, вы получите увеличение производительности МП в процентах.

Повторив измерения для другой пары задач, вы почти наверняка получите другой результат. Как уже говорилось, технология HT только создает условия для повышения производительности ПК. А будет она повышена или нет — полностью зависит от того, насколько рационально используются ресурсы МП при параллельном исполнении задач. История повторяется — в свое время при программировании для Pentium I надо было заботиться о загруженности двух труб конвейера, теперь надо заботиться о загруженности всех исполнительных устройств Pentium 4. Все это делается ради повышения производительности МП.

Впрочем, есть и другой, на мой взгляд, более важный фактор. Программисты получили возможность, работая на ПК, осваивать программирование параллельных процессов. Инструмент не ахти как хорош, но скудность средств рождает изощренность мысли. И есть надежда, что Intel подправит инструмент.

Распараллеливание программ

В литературе по параллельному программированию обычно различают два подхода к декомпозиции программ для их выполнения в многопроцессорной среде — распараллеливание данных и распараллеливание последовательных действий. Альтернативой распараллеливанию данных являются векторные вычисления, выполняемые в однопроцессорной среде. Для этого предназначены специальные векторные компьютеры, которые манипулируют группами регистров, содержащих однородные данные. Группа может состоять, например, из 128 регистров. Все три способа декомпозиции могут применяться при программировании для МП Pentium 4 HT, разумеется, с учетом его реальных возможностей.

Распараллеливание данных применимо в тех случаях, когда вычислительный цикл повторяется многократно. При этом данные делятся на группы, количество которых совпадает с количеством процессоров, существующих в системе или выделенных для выполнения задачи. Затем на каждом процессоре запускается фрагмент цикла, обрабатывающий свою порцию данных. Такой способ распараллеливания наиболее прост при программировании и весьма эффективен при исполнении в истинно многопроцессорной среде. В нашем случае две выполняемых копии цикла будут использовать одни и те же исполнительные устройства процессора, поэтому существенное ускорение вычислений маловероятно.

Распараллеливание действий применяется в тех случаях, когда задача выполняет разные действия над независимыми наборами данных. Например, в задачах компьютерной графики производятся манипуляции с отдельными точками изображений, а каждая точка имеет два независимых набора характеристик — координаты и компоненты цвета, которые обрабатываются по разным алгоритмам. В нашем случае данный метод предпочтительнее предыдущего, поскольку есть шанс подобрать такие алгоритмы вычислений, при которых параллельные потоки будут использовать разные исполнительные устройства МП.

Векторные вычисления программируются с применением инструкций SIMD (одна инструкция много данных), входящих в состав групп MMX и SSE. Они работают с группами целых или вещественных чисел, упакованных в 128-разрядные операнды. Количество чисел в группе зависит от их разрядности, например, вещественных чисел может быть два или четыре, в

зависимости от требуемой точности вычислений. При замене инструкций FPU на инструкции SIMD двух- или четырехкратное ускорение вычислений не получится, но близкие значения вполне реальны. Поэтому, прежде чем заниматься распараллеливанием, имеет смысл изменить программу так, чтобы при вычислениях максимально использовались инструкции SIMD.

Инструментарий программиста

Для распараллеливания задачи ее исходный текст надо преобразовать так, чтобы в нем были описаны задания для каждого процессора и согласовано исполнение параллельных процессов. Преобразование текста программы вручную — процесс достаточно трудоемкий, требующий определенных знаний и навыков программиста. Для его упрощения существуют специальные средства автоматизации программирования многопроцессорных систем. Они делятся на две категории:

- специализированные "параллельные" компиляторы (например, для языков C/C++, Fortran и пр.), самостоятельно выполняющие частичное распараллеливание программируемой задачи. В большинстве случаев они самостоятельно распознают и преобразуют циклы, оставляя последовательные фрагменты текста без изменений;

- специализированные API высокого уровня, используемые при работе с обычными компиляторами с языков C/C++, Fortran и пр. В этом случае программист самостоятельно выбирает распараллеливаемые фрагменты и расставляет соответствующие директивы в нужных местах текста.

К числу последних относится стандарт OpenMP, разработанный для программирования SMP-систем. В нем описаны набор директив и их параметров, переменные окружения и функции библиотеки Run-Time. Стандарт учитывает особенности конкретного языка программирования, поэтому его версии для C/C++ и Fortran совпадают не полностью. Корпорация Intel реализовала OpenMP для своих компиляторов Intel C++ и Intel Fortran, которые предназначены для программирования Pentium 4 HT.

При работе с OpenMP вы сначала составляете и отлаживаете программу для выполнения на одном процессоре, т.е. в последовательном режиме. После этого в нужные места исходного текста вставляются директивы, выполняющие распараллеливание конкретных участков программы. Например, при программировании на C++ для распараллеливания конкретного вычислительного цикла достаточно непосредственно перед оператором цикла указать следующую директиву:

#pragma omp for [*необязательные параметры*]
<*распараллеливаемый оператор цикла типа for*>

Директива действует только на один оператор цикла или блок. Если ее параметры не указаны, то используются их значения, принятые по умолчанию. Откомпилировав и выполнив задачу, вы можете оценить эффективность распараллеливания, а затем, задавая параметры, попытаться ускорить выполнение задачи.

Директивы позволяют распараллеливать независимые последовательные фрагменты кода, определять "видимость" и ограничивать использование переменных параллельного региона, планировать выполнение параллельных процессов и выполнять другие полезные действия.

Распараллелить можно не любую задачу и даже не любой вычислительный цикл — во многих случаях приходится изменять реализацию алгоритма или сам алгоритм. Но это связано с особенностями параллельных вычислений и не имеет отношения к OpenMP.

Работу с OpenMP поддерживают компиляторы C++ начиная с версии 5.0 (имеются в виду компиляторы, распространяемые Intel). Компилятор C++ версии 6.0 способен самостоятельно, т.е. без явного указания директив OpenMP, распараллеливать циклы, но не более того. Во время подготовки данной статьи появились сообщения о начале продаж новой, 7-й версии компиляторов C++ и Fortran. В рекламе, в частности, говорится, что у седьмой версии усовершенствована поддержка OpenMP.

Синхронизация процессов

При разработке программ, предназначенных для выполнения на МП Pentium 4 HT, надо учитывать, что параллельные процессы исполняет один физический процессор. Поэтому вполне возможны случаи, когда оба процесса или обе задачи одновременно претендуют на один и тот же ресурс. Во избежание конфликтных ситуаций, в таких случаях рекомендуется синхронизация выполнения процессов. Это относится к любым видам задач, выполняемых на Pentium 4 HT, по этой причине в документации Intel рекомендуется проводить перекомпиляцию ранее созданных программ.

Практикующему программисту известны случаи, когда выполнение задачи надо приостановить до момента наступления некоторого (не важно какого) события. Для этого с событием ассоциируется целочисленная переменная, значения которой указывают, наступило событие или нет, и организуется цикл проверки и ожидания изменения ее значений.

Пример 1. Цикл проверки и ожидания изменения значения переменной lockvar

```
Spin_Lock:    CMP lockvar, 0 ; проверка состояния переменной
                             ; lockvar = 0 ?
              JNE Spin_Lock  ; нет - возврат на начало цикла
              ; продолжение программы
```

Важно! Если lockvar = 1, то Pentium 4 входит в бесконечный цикл, и вывести его из этого состояния можно только выключив питание! Причина заключается в следующем. Участвующие в выполнении цикла инструкции порождают три микрооперации. Одна из них считывает из памяти значение переменной lockvar, а две другие выполняют сравнение и ветвление. Это быстрые микрооперации, исполняемые МП за половину такта, поэтому вместе сравнение и ветвление исполняются за один такт, а чтение из памяти занимает два такта. В результате блокируется работа с памятью, и значение lockvar никогда не будет изменено.

Специально для решения описанной проблемы разработчики изменили логику выполнения инструкции PAUSE. У предшествующих моделей МП она интерпретировалась как NOP — нет операции, у Pentium 4 она просто исключает возможность исполнения сравнения и ветвления за один такт.

Как выглядит цикл проверки и ожидания для МП Pentium 4, показано в примере 2. А для того чтобы этот пример можно было использовать для синхронизации параллельно выполняемых процессов, после цикла добавлено еще четыре команды. Обращаю ваше внимание на то, что для определения состояния требуемого ресурса в обоих процессах должны выполняться одинаковые действия, описанные в примере 2, в противном случае цель не будет достигнута.

Пример 2. Работа с флагом по схеме "проверка", "проверка и установка" для МП Pentium 4.

```
Spin_Lock:CMP lockvar, 0     ; проверка состояния переменной
                             ; lockvar = 0?
          JE Get_Lock        ; да - выход из цикла проверки и
                             ; ожидание (ресурс свободен)
          PAUSE              ; нет - ресурс занят,
                             ; выполняется инструкция PAUSE
          JMP Spin_Lock      ; возврат на начало цикла
Get_Lock: MOV EAX, 1         ; EAX=1
          XCHG EAX, lockvar  ; перестановка EAX = lockvar,
                             ; lockvar = 1
          CMP EAX, 0         ; переменная lockvar была очищена?
          JNE Spin_Lock      ; нет, возврат на начало цикла
           <работа с требуемым ресурсом>
          MOV lockvar, 0     ; очистка переменной lockvar
          ; продолжение цепочки инструкций
```

Первые четыре команды образуют цикл проверки и ожидания для МП Pentium 4. Обсудим действия, выполняемые во второй части примера, начиная с команды, имеющей метку Get_Lock. Просто изменить значение переменной lockvar

с помощью команды "mov lockvar, 1" нельзя, поскольку в режиме HT за время перехода на метку Get_Lock второй процесс мог изменить состояние переменной lockvar. Для исключения конфликта, перед изменением lockvar ее старое значение сохраняется в регистре EAX. Если окажется, что оно было равно нулю, то цель достигнута, и данный процесс может использовать требуемый ресурс, а по окончании работы с ним надо очистить переменную lockvar, чтобы сделать ресурс доступным обоим процессам.

Замечание: *При программировании на C++ аналогом примера 1 является пустой оператор цикла* `do {} while`. *Чтобы избежать зацикливания МП Pentium 4, между фигурными скобками надо записать* `"asm pause"`, *т. е. тело цикла должно иметь следующий вид:* `do {asm pause} while`.

При выполнении цикла ожидания ресурсы МП используются не лучшим образом, поэтому, если ожидание продлится достаточно долго, то целесообразно временно приостановить работу логического процессора, исполняющего цикл. Для этого у Pentium 4 при работе в режиме HT изменяется логика выполнения инструкции **HLT**. Она остается привилегированной, но останавливает работу конкретного **логического процессора**, не влияя на работоспособность физического процессора.

По соответствующему программному запросу ОС может прекратить выполнение процесса и продолжить его по истечении указанного в запросе времени. Во время паузы может выполняться только один процесс, или вместо остановленного запущен второй процесс. Для использования этих и других возможностей управления процессами надо иметь под руками описание API конкретной ОС.

Провести четкую границу между коротким и длительным временем ожидания невозможно. Поэтому в каждом конкретном случае программист должен выбирать способ ожидания, с учетом того, что на исполнение функций API требуется определенное время, и может оказаться, что оно будет больше, чем время ожидания освобождения ресурса.

Заключение

Технология HT, реализованная в МП Pentium 4 с ядром Northwood, была первым шагом корпорации Intel на пути внедрения параллельных вычислений на персональных компьютерах. Pentium 4 с ядром Prescott поддерживает уже улучшенный вариант технологии HT — ее возможности расширены за счет распараллеливания работы с памятью. В перспективе Intel планирует размещение двух физических процессоров в одном кристалле, а это означает появление настольных двухпроцессорных систем. Учитывая сказанное, имеет смысл осваивать искусство программирования параллельных процессов, оно очень скоро может вам пригодиться.

Плагин-эмулирующий отладчик для дизассемблера IDA

Р.ФАСИХОВ,
г.Казань.
E-mail: fasihov@mail.ru

(Окончание. Начало в №9/2004)

Разберемся с сегментами IDA.

Сегменты в IDA — один из ее краеугольных камней. И надо ознакомиться со структурами, которые их описывают. Нам они понадобятся для создания сегмента стека отладчика. Без него работать, мягко говоря, некорректно.

Каждый сегмент имеет базовый сегментный адрес, который определяет положение сегмента в виртуальной памяти IDA. В поле Base может находиться базовый адрес, а может находиться индекс селектора. Селектор содержит 32-битный базовый адрес сегмента в виртуальном пространстве IDA. Подробнее смотрите [1].

Данные и функции, работающие с сегментами, содержатся в заголовочном файле **SEGMENT.HPP**. К сегментам можно обращаться разными способами. Тип структуры, описывающей сегмент — **segment_t**. В ней содержится вся информация о сегменте: база, начальный адрес, конечный адрес, имя сегмента, выравнивание сегмента, разрядность 16 или 32 бита, и.т.д.:

```
class segment_t : public area_t
{
public:

long name;          // имя сегмента
long sclass;        // класс сегмента
long orgbase;       // это поле зависит от IDP
uchar align;
uchar comb;         // Код комбинирования сегмента с другими
uchar perm;         // "Разрешение" сегмента (EXE, READ, WRITE)
uchar use32;        // разрядность сегмента
ushort flags;       // флаги сегмента
sel_t sel;          // селектор сегмента
uchar type;         // Тип сегмента

);
```

Вот как примерно выглядит создание сегмента стека для отладчика (подробное описание функций смотрите в SEGMENT.HPP):

```
// Находим пустой селектор и проецируем на виртуальную память
// N-селектор для сегмента
      sel_t N=allocate_selector(0);
// Создаем сегмент
      if(add_segm(N,0xFFFF0000,0xFFFFFFFE,"STCK","STACK"))
      {
            segment_t DEBUGGER_STACK_SEGMENT, *pDSS;
            pDSS = &DEBUGGER_STACK_SEGMENT;

            pDSS = get_segm_by_sel(N);
            set_segm_addressing(pDSS,1);

            msg("Debugger stack created\n");
      }
      else warning("Cant create segment\n");
```

Как видите, ничего сложного нет.

Об обработчике команд

В эмуляторе всю работу по исполнению инструкций выполняют обработчики команд. Было решено в обработчик передавать тип соотношения операндов (например, один операнд — регистр, другой — в памяти). В исходниках это выполняет функция **get_operand_info(OPER &)**, которая заполняет структуру OPER. Также она заполняет поля структуры вычисленными значениями операндов (например, при косвенной адресации). Сам обработчик команды знает о том, какие операнды и какие сочетания операндов возможны, и в зависимости от этого выполняется. Покажу это на примере инструкции push:

```
//  push instruction
BYTE emPUSH(OPER &O){
// Выводить при работе команды, как вы понимаете,
// не обязательно. Просто удобно при отладке обработчика
      msg("This is a PUSH cmd emulation\n");
```

```
// В стеке есть место??
      if(!checkESP(-4)) return false;

// Проверка, если один из операндов в памяти, на
// принадлежность к несуществующему адресу.
      if(!checkEA(0))   return false;

      EIP2NextCmd(0);
//  Операнд один - непосредственное значение 32 бита
      if(0.rel==i32) {
           *pemESP-=4;              //Уменьшили ESP
                                    //Пишем в вирт. память
            return (WriteVMDWORD(*pemESP,0.op0_i32));
      }

//  Операнд один - непосредственное значение 16 бита
      if(0.rel==i16) {
            *pemESP-=2;
            return (WriteVMWORD(*pemESP,0.op0_i16));
      }
// Операнд один - непосредственное значение 8 бит
      if(0.rel==i8) {
            *pemESP-=2;
            return (WriteVMWORD(*pemESP,(WORD)0.op0_i8));
      }
// Здесь должна быть проверка на операнды - регистры и операнды,
// находящиеся в памяти
//    if(0.rel==r32). . .
//    if(0.rel==r16). . .

//    if(0.rel==m32). . .
//    if(0.rel==m16). . .

// Если мы здесь, то операнды не опознаны
      warning("UNKNOWN OPERAND IN PUSH CMD");
      return INS_BAD_OP;
}
```

Это одна из "частных" инструкций, и для других писать обработчик намного проще. Например, эмуляция инструкции cmp:

```
//     cmp
void __declspec(naked) emCMP8() {_asm {cmp al,bl
                                       retn}
                                }
void __declspec(naked) emCMP16(){_asm {cmp ax,bx
                                       retn}
                                }
void __declspec(naked) emCMP32(){_asm {cmp eax,ebx
                                       retn}
                                }

//////////////////////////////////////////////////////
// cmp
//////////////////////////////////////////////////////
bool emCMP(OPER &O){
      if(!checkEA(0))   return false;

      EIP2NextCmd(0);
      return two_op_handle(0, emCMP8, emCMP16, emCMP32);
}
```

Вот и весь обработчик инструкции. Вся работа будет происходить в **two_op_handle**, которая и учтет типы операндов, их воздействие на флаги.

Перед написанием обработчика инструкции неплохо бы узнать, что возвращает структура **cmd**, так как это значение может отличаться для разных инструкций. Например, для инструкции **lea eax,[ebx] cmd.Op1.dtyp = dt_dword**, а **cmd.Op1.dtyp = dt_byte**. Поэтому, возможно, придется добавить обработку ситуации в функцию **get_operand_info**.

При написании обработчиков *ОЧЕНЬ* помогает справочник из [2], там расписаны все типы операндов команд и алгоритм их работы.

Послесловие

На данный момент поддерживаются:

- точки останова BPX;
- BPM_R, BPM_W, BPM_RW;
- эмулировано 122 инструкции x86 (кроме 3-операндных imul, shld, shrd, enter, leave и др., их можете реализовать сами).

Команды управления отладчиком можно уточнить как help(); из консоли IDA.

Надо добавить:

- создать нормальный показ регистров в отладчике в виде отдельного диалогового окна;
- при желании можно добавить эмуляцию FPU, MMX;

Самое серьезное — непонятно, как эмулировать вызовы WinAPI-функций. Похоже, что не получится, но, в принципе, в контекстном отладчике это и не нужно, хотя было бы и неплохо иметь.

На веб-странице журнала в приложении к статье содержится исходный код отладчика, а также скомпилированная версия плагина для IDA 4.5

Похоже, что файл **plugins.cfg** необязателен (IDA 4.5), т.к. и без него плагин вызывался нормально. Подозреваю, что IDA сканирует директорию с плагинами и узнает параметры через структуру PLUGIN. Если не получится, то добавьте следующую строку:

```
CDEx86    CDEx86    F11          0
```

Также можно добавить горячую клавишу к плагину.

Элементарные инструкции для работы с плагином:

- скопируйте CDEx86.plw в папку \plugins дизассемблера;
- запустите дизассемблер;
- подведите курсор к куску кода, который хотите изучить, и начинайте трассировку (по умолчанию — клавиша F11);
- значения регистров можно посмотреть, вызвав `r();` из консоли (F2). В следующей версии, возможно, будет отдельное окно с регистрами;
- если хотите сделать автоматическую трассировку, то подведите курсор к месту, где должна остановиться трассировка, и введите из консоли точку останова:

```
bpx();
at();
```

Точно так же устанавливаются виртуальные бряки на чтение/запись в память `bpm();`.

За время использования своего творения у меня появились некоторые замечания и рекомендации по работе с ним:

- отладчик предполагает хорошие навыки работы с IDA;
- не надо лезть в вызов WinAPI-функции, т.к. ее кода в дизассемблере нет, и, соответственно, нельзя ее трассировать. Лучше всего ее перепрыгнуть, сбалансировав стек;
- отладчик получает инструкцию по адресу курсора IDA. Поэтому, если вы прыгнете на середину команды, то получите сообщение Not code. Stay at code first. В этом случае надо сначала сделать превращение в Unknown, а затем превратить в Code.

Не надо требовать от контекстного отладчика сверхъестественного. Я писал его лишь из-за интереса к внутреннему устройства IDA. Если чего-то не хватает, добавьте сами — исходники есть.

Хотелось бы сказать большое спасибо Крису Касперски за его отличные книги и статьи.

Жду поправок и более "крутых" решений.

Литература

1. К.Касперски. Образ мышления IDA. — http://pilorama.com.ru/library/pdf/om_ida.pdf

2. В.Юров, С.Хорошенко. Ассемблер: учебный курс. — С.-Пб.: Питер, 1999.

3. С.Зубков. Assembler: для DOS, Windows и Unix.— М.: ДМК, 1999.

"PlayStation 2": аудио, видео, DVD/CD

С.РЮМИК,
г.Чернигов.

Не бойся, что не знаешь — бойся, что не учишься.
Китайская поговорка

На структурной схеме процессорной платы игровой приставки "PlayStation 2" (PS2) канал звука и видеокодер имеют одну общую точку — выходной разъем "AV MULTI OUT". На **рис.1** приведена электрическая схема прилегающих к нему цепей. Нумерация контактов, назначение сигналов, внешний вид разъема CN501 полностью совпадают с аналогичным, применяемым в "PlayStation" (PSX).

Канал звука

Звуковые сигналы левого и правого каналов выводятся на телевизор через контакты 9 и 11 разъема CN501. Их источником служит двухканальный аудио-ЦАП на микросхеме IC403 CS4335-KS (ф.Cirrus Logic). По структуре это последовательный однобитовый дельта-сигма-ЦАП, но более совершенный по возможностям, чем применявшийся в PSX AK4309AVM (ф.Asahi Kasei Microsystems Co. Ltd).

На вход IC403 поступают четыре сигнала от 48-канального звукового процессора IC401 CXD2950R (ф.SCEI). Их назначение: SDATA — последовательные 16-разрядные данные; LRCK — сигнал переключения каналов; SCLK — побитовая синхронизация; MCLR — тактовые импульсы, определяющие частоту дискретизации 44,1 или 48 кГц. Временные диаграммы аналогичны PSX (http://psxdev.narod.ru/docs/psx_circ3.htm).

Сборки цифровых транзисторов Q401, Q402 предназначены для полного отключения звука при начальных переходных процессах. Момент отключения определяют сигналы единичного уровня с выводов 40 и 105 микросхемы IC401. Как следствие — выводы 1 и 4 сборки Q401 соединяются с общим проводом.

Рис. 1

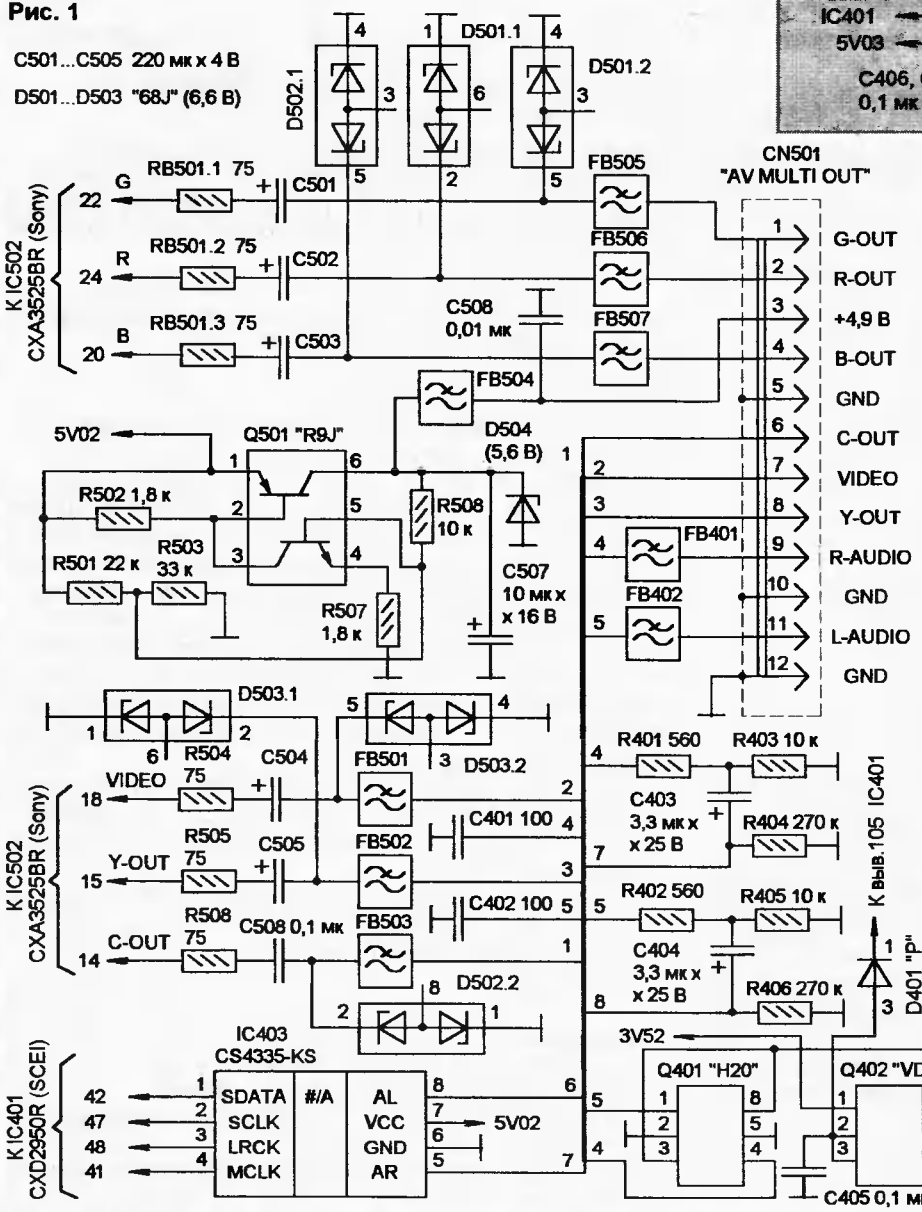
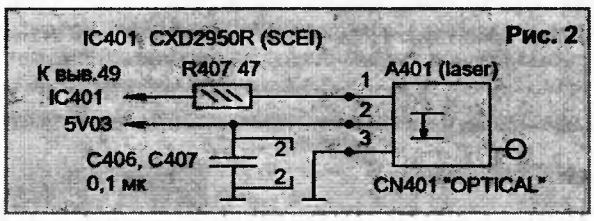


Рис. 2



К каналу звука относится разъем CN401 "OPTICAL" (**рис.2**), предназначенный для вывода звука в оптическом формате на систему "Домашний кинотеатр". Внутри блока A401 находится маломощный лазерный диод. Питание на него подается через вывод 2, управление — через вывод 1. Оптический выход можно включать и отключать вручную через меню "Audio Setup" на экране телевизора в пункте "Digital Out On/Off".

С внешней стороны глазок лазера закрыт пластмассовой крышкой, под которой имеются пазы для стандартного оптического разъема. К нему подключается оптоволоконный кабель, который соединяется с входом "Tos-link" или "OPTICAL" аппаратуры пространственного звучания. Поддерживаются стандарты: DTS, Dolby Digital (AC-3), MPEG Audio.

Неисправности канала звука

Причиной отсутствия звука или слабой громкости может быть замыкание в сборке транзисторов Q401. Чтобы убедиться в этом, достаточно измерить сопротивление между контактами 9 и 12, 11 и 12 разъема CN501. Если оно меньше 10,5 кОм, значит, имеется дефект. Для уверенности можно приподнять над платой выводы 1 и 4 сборки Q401. Производится это тонкой иголкой с прогревом выводов паяльником.

Основные параметры CS4335-KS:

- частота дискретизации F_s, кГц ..... 32...96;
- неравномерность АЧХ, дБ ..... 0,01...+0,08 в диапазоне частот 10 Гц...20 кГц;
- динамический диапазон, дБ ..... 86...94;
- переходное затухание между каналами, дБ 94;
- выходное напряжение на нагрузке 3 кОм, В ..... 3,25...3,75;
- мощность потребления при напряжении питания 4,75...5,5 В, мВт ..... 75...104.

Видеоканал

PS2 формирует на разъеме "AV MULTI OUT" набор видеосигналов для подключения к телевизору по низкой частоте. Чтобы задействовать антенный вход, необходимо дополнительно приобрести ВЧ-модулятор. Средств подачи видеосигнала на компьютерный VGA-монитор нет.

Европейские приставки PS2 формируют полный цветовой телевизионный сигнал VIDEO стандарта PAL. Он имеет частоту поднесущей 4,433 МГц. Сигнал VIDEO можно подавать как на цветной, так и на черно-белый телевизор.

Сигналы Y-OUT, C-OUT — это композитные видеосигналы, несущие информацию соответственно о яркости и цветности. Изображение, получаемое с их помощью, лучше, чем получаемое через VIDEO. Однако для просмотра требуется телевизор со специальным четырехконтактным входом "S-VHS", "S-Video" или "Composite Video", а также соединительный кабель.

Самая качественная картинка получается при использовании сигналов R, G, B. Это красный, зеленый и синий аналоговые видеосигналы, которые вместе с сигналом яркости Y-OUT можно подавать на CGA-монитор или на три видеоусилителя телевизора через согласующий модуль, например, такой, как в домашних компьютерах "ZX-Spectrum". Нагрузочная способность всех видеовыходов (VIDEO, Y-OUT, C-OUT, R, G, B) составляет 75 Ом, размах сигналов — 0,7...1 В. Все они защищены от коротких замыканий в нагрузке резисторами RB501, R504...R506, а от превышения напряжения — сборками стабилитронов D501...D503.

Для питания внешнего антенного ВЧ-модулятора или других периферийных устройств используется выведенное на контакт 3 разъема CN501 напряжение 4,9 В с током нагрузки до 80 мА. Оно обеспечивается каскадом на сборке транзисторов Q501, аналогично схемотехнике PSX.

Неисправности видеоканала

Причиной отсутствия изображения или срыва синхронизации могут быть пробои стабилитронов D501...D503. Это проверяется омметром между контактами 1-12, 2-12, 4-12, 6-12, 7-12, 8-12 разъема CN501, без вскрытия корпуса PS2. Другая причина — обрыв низкоомных (75 Ом) согласующих резисторов RB501, R504...R506, фильтров FB501...FB503, FB505...FB507 или переходных конденсаторов C501...C506.

Интерфейс DVD/CD

PS2 первой среди игровых приставок получила доступ к дискам DVD. Новые игры с объемными, почти "живыми" персонажами и продолжительными по времени видеороликами уже не могут поместиться на обычные CD емкостью 650 Мб. Для PS2 необходимы односторонние однослойные диски DVD емкостью 4,7 Гб. Чтобы потребителю было легче ориентироваться, ф.Sony ввела цветную маркировку своих дисков (см. **таблицу**).

Цвет нижней стороны фирменного диска	Надпись на верхней стороне	Назначение фирменного диска
Серебристый	DVD-ROM	Игровой DVD для PS2
Синий	Compact disk	Игровой CD для PS2
Черный	Compact disk	Игровой CD для PSX

Пользоваться таблицей нужно с осторожностью, поскольку сейчас имеют хождение нелицензионные CD и DVD самых разных цветов. Например, "пиратские" CD серебристого и черного цвета могут содержать урезанные копии DVD-игр. Отличить невооруженным глазом фирменный диск DVD от нефирменного CD довольно трудно, поэтому следует опасаться подделок и не очень доверять надписям "DVD ROM" на этикетке упаковочной коробки. Признак "фирменности" — это наличие трех надписей "PlayStation 2" вперемежку с логотипом на внутреннем кольце диска (смотреть в отраженном свете), а также четкие, не расплывающиеся буквы надписей на его верхней, иллюстрированной стороне.

Поскольку приставка PS2 должна читать диски разных форматов, в лазерном блоке содержатся цепи как для DVD, так и для CD. На **рис.3** приведена схема лазерной головки KHS-400C и входные цепи интерфейса DVD/CD для модели SCPH-30004R (версия 6).

Как известно, на диске DVD информация записана в виде микроуглублений меньшего размера, чем на CD. Следовательно, для уверенного считывания данных требуется лазер с меньшей длиной волны излучения. В блоке A1 находятся два лазерных диода, излучающих на волне 780 нм (CD) и 650 нм (DVD), а также два фотоприемника: один — для CD, другой — для DVD. Получается конструкция "два в одном", где общей частью служат линза объектива и сервопривод с катушками YA1, YA2, управляемый микросхемой IC705. Для справки, в зарубежной литературе сервопривод называют "actuator" (активатор, актюатор), а микросхему, управляющую его работой, — "driver" (драйвер).

Питание на лазерные диоды поступает от транзисторных каскадов Q701 (CD) и Q702 (DVD). Токи через них регулируются подстроечными резисторами R1 (DVD) и R2 (CD) в пределах 40...60 мА, что можно измерить по падению напряжения на резисторах R704, R705 (CD) и R707, R708 (DVD). Если приглядеться, то схемотехника этих каскадов ничем не отличается от PSX (http://psxdev.narod.ru/docs/psx_circ4.htm).

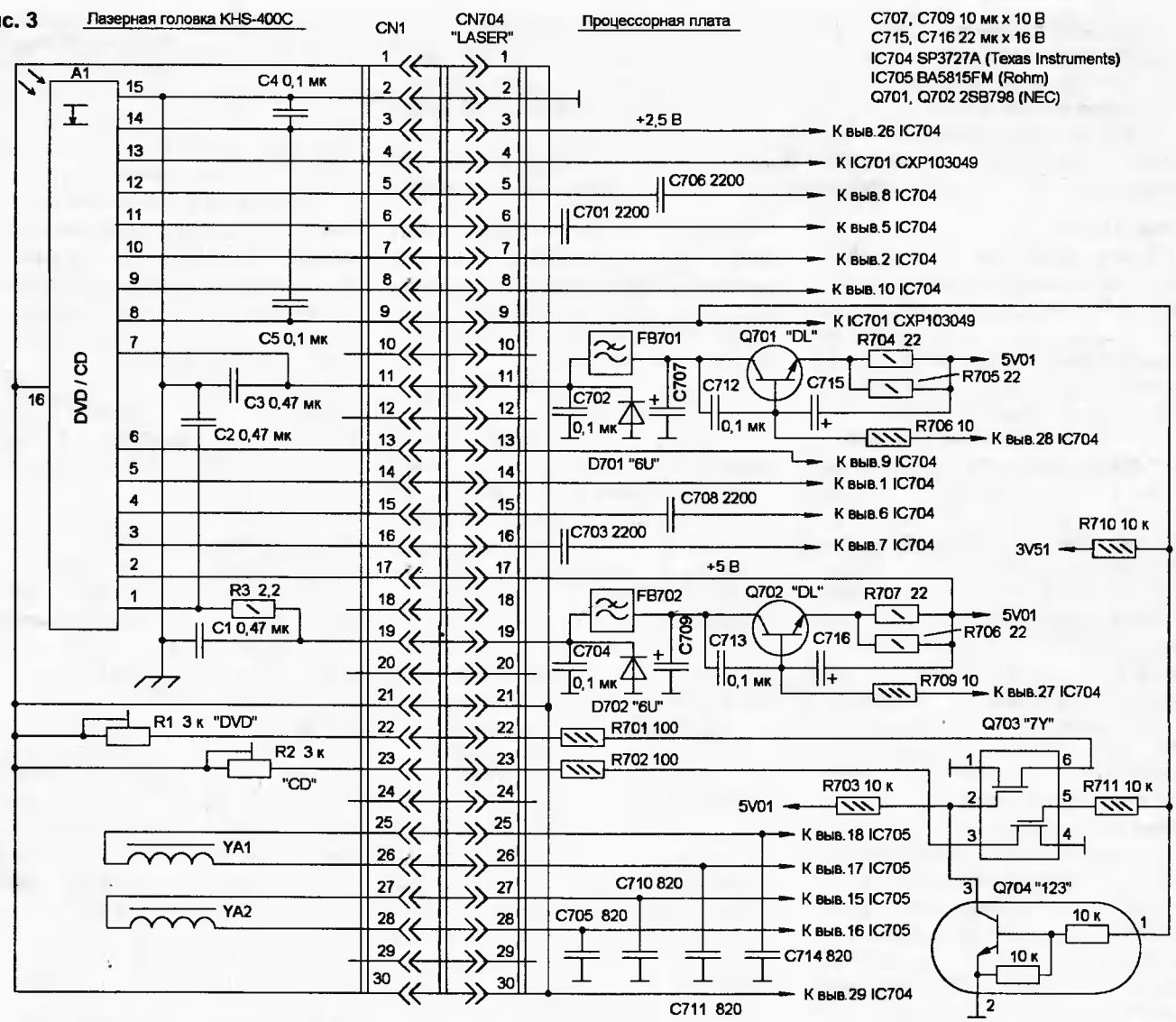
Сигнал выбора CD или DVD поступает с контроллера IC701 на контакт 9 разъема CN704, и далее — на базу "цифрового" транзистора Q704. Если последний открыт, то замыкается цепь между выводами 3, 4, а если закрыт — между выводами 1, 6 сборки транзисторов Q703. Тем самым запрещается одновременное излучение лазеров CD и DVD, поскольку в любой момент времени правый по схеме вывод одного из подстроечных резисторов R1 или R2 обязательно будет "висеть в воздухе".

Внутри блока A1 имеется 8 фотодиодов, по 4 для CD и DVD. Однако прозвонить омметром их не удастся. Так же как и в лазерной головке PSX, фотодиоды PS2 подключаются к внутренним операционным усилителям, повышающим уровень выходного сигнала. Питание 5 В на усилители подается через контакт 17, напряжение для средней точки 2,5 В — через контакт 3 разъема CN704.

Сигналы с фотоприемников блока A1 обрабатывает специализированная микросхема IC704 SP3727A (ф.Texas Instruments). К сожалению, на сайте ее изготовителя отсутствуют не только параметры, но и упоминание о ней самой. Это случается с микросхемами частного применения, которые изготавливаются только под определенный заказ.

На **рис.4** приведены схемы включения двигателей в приводе DVD/CD. В PS2, в отличие от PSX, лазерный диск

Рис. 3



вращается с постоянной, а не переменной угловой скоростью (Constant Angular Velocity, CAV). Это означает, что данные, находящиеся у внешнего края диска, считываются быстрее, чем вблизи его центра. Скорость передачи данных привода DVD принято указывать в безразмерном множителе относительно базовой скорости 1250 Кбит/с. Множитель 1x DVD примерно равен множителю 8x CD. В PS2 на внешнем крае диска множитель скорости составляет 4x DVD, а на внутреннем — 1...2x DVD. Для CD в фирменной спецификации параметров записано значение 24x.

Микросхема IC706 BA6664FP (ф.Rohm) обеспечивает постоянную скорость вращения трехфазного двигателя M1 ($R_{обм}$=4,5 Ом). Ее параметры: $U_{пит}$=7...15 В, $P_{рас}$=2,2 Вт, $I_{вых}$=1,3 А, корпус — HSOP-28. Для стабилизации

Рис. 4

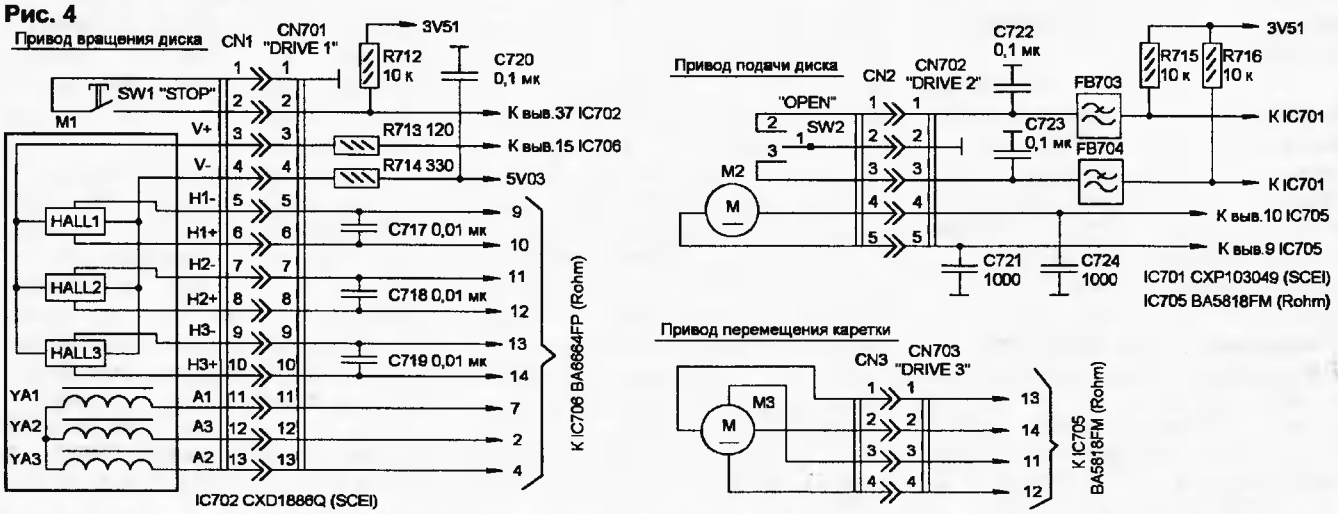
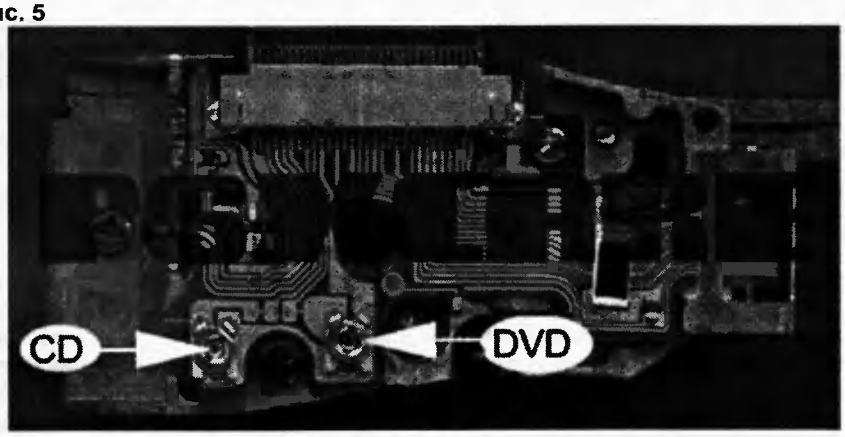


Рис. 5



скорости с точностью до десятых долей процента применяется следящая обратная связь с использованием трех датчиков Холла HALL1...HALL3. На вывод 22 IC706 подается сервосигнал, усиленный и частотно-скорректированный двухкаскадным операционным усилителем на микросхеме IC707 NJM12904 (ф.JRC).

Кнопка SW1 замыкает свои контакты при механическом достижении лазерной головкой внутреннего обода диска. Затем контроллер IC701 выдает сигнал микросхеме IC705 на запрет дальнейшего перемещения каретки.

Вращением двигателей подачи диска M2 и перемещения каретки M3 управляет микросхема-драйвер IC705 BA5815FM (ф.Rohm). Ее параметры: $U_{пит}$=4,2...13,2 В, $P_{рас}$=2,2 Вт, $R_н$=8 Ом, 5 каналов управления, два из которых — на основе широтно-импульсного регулирования, корпус — HSOP-M28. Двигатель M2 — линейный ($R_{обм}$=3,7 Ом), а M3 — шаговый двухобмоточный ($R_{обм}$=20 Ом).

В переключателе SW2 контакты 1-2 замыкаются при полном "въезде", а контакты 1-3 — при полном "выезде" привода DVD/CD из или в корпус PS2.

Неисправности канала CD/DVD

Судя по форумам в Интернете, этот канал чаще всего подвержен поломкам. Стандартная неисправность — после нескольких месяцев нормальной работы перестают читаться фирменные и нефирменные диски DVD, хотя все CD, в том числе музыкальные, вводятся без проблем.

К сожалению, в подавляющем большинстве случаев причина кроется в лазерной головке KHS-400, замена которой вместе с приводом DVD/CD обойдется почти в четверть стоимости PS2. Обычно головку не меняют, смирившись с отсутствием канала DVD. В таком состоянии приставка прослужит еще долгие годы. Проверку исправности проводят заменой привода заведомо годным от той же самой модели PS2, поскольку они механически не взаимозаменяемы.

Известны способы восстановления работоспособности головки увеличением тока накачки лазерного диода. Для этого на гибкой плате лазерной головки меняют сопротивление подстроечных резисторов R1 (CD, ближе к краю платы), R2 (DVD, в центре платы) в пределах 10...20% от исходного значения (**рис.5**, http://www.ps2.nu/laserkalibrering7834/image8-s.jpg).

Увеличенный ток накачки лазера 60...80 мА контролируют по падению напряжения на резисторах R704,.R705 (CD), R707, R708 (DVD). Подготовленным радиолюбителям можно рекомендовать настройку лазерной головки по осциллограммам, приведенным на сайте http://www.ps2.nu.

При испытаниях надо учитывать, что лазер DVD работает только при наличии диска DVD, а лазер CD — только при наличии диска CD. Кроме того, оба лазера кратковременно включаются после подачи питания 220 В и нажатия кнопки "OPEN" на лицевой панели PS2. При этом, глядя сбоку на лазерную головку, по отраженному свету хорошо видна вспышка луча лазера длительностью 0,5 с. Меры предосторожности аналогичны PSX — категорически запрещается напрямую смотреть в объектив лазера во избежание повреждения роговицы глаза!

Ремонтируем проводную клавиатуру ПК

А.КАШКАРОВ,
г.С.-Петербург.

Всем известные предостережения типа: "Не ставьте кофе вместо CD" и "Не превращайте рабочее место в обеденный стол", уже, что называется, набили оскомину. Однако, в процессе ежедневной многочасовой работы, когда ПК превращается в незаменимый рабочий инструмент, внимание и опытных, и начинающих пользователей притупляется. Случаются такие неприятности как раз при нарастающей Луне, видимо, человек подвержен еще не вполне изученному влиянию этого ночного светила. И вот тогда магазины по продаже комплектующих перевыполняют план.

Мне неоднократно приходилось возвращать к "жизни" клавиатуры, отданные в ремонт со словами: "Я пролила случайно чай, и теперь не все клавиши выскакивают на экране". Простим некоторое невежество домохозяйкам и разберемся в сути. Обычные недорогие клавиатуры, которыми сегодня наводнен рынок — это приборы, как правило, китайского либо корейского производства и made in Thailand. Они могут различаться по способу подключения (PS/2, USB), иметь различные, казалось бы, наименования (Chicony KB 9810, KB 9850, KWD 820, Microsoft RT 9480, GodeGen 1307, 1616, Cherry и др.), но всех их объединяет внутреннее строение. Внутри корпуса клавиши давят на плоские, прижатые друг к другу пленки с нанесенными на них токопроводящими дорожками и контактными участками. Такие клавиатуры отличает относительно низкая цена — от 200 до 1000 руб. Другие клавиатуры, (в том числе беспроводные), имеющие более "весомую" стоимость, сделаны "по-человечески" — на основе нормальных замыкающихся контактов. Однако, поскольку сегодня наиболее популярны именно простые (пленочные) клавиатуры, рассмотрим вопрос их восстановления более подробно.

Современная проводная клавиатура соединяется с системной платой ПК посредством специального шестиконтактного разъема (PS/2). Когда пользователь нажимает какую-либо клавишу, контроллер клавиатуры пересылает на материнскую плату последовательность импульсов отрицательной полярности, которые можно зафиксировать на контакте 2 этого разъема.

Как правило, жидкость, попавшая внутрь клавиатуры, является пищевой — это сладкие чай и кофе, различные лимонады и пиво. Все перечисленные, популярные у пользователей составы — органические, и после своего высыхания оставляют тончайшую изолирующую пленку из слипающихся молекул. Эту липкую пленку можно почувствовать при прикосновении пальцами к пораженным участкам — пальцы будут прилипать. Жидкость, попавшая внутрь корпуса пленочной клавиатуры (в зависимости от объема пролитого), нарушает посеребренное покрытие контактных площадок клавиш, из-за этого нарушается сам контакт; приходится сильнее нажимать на клавишу, для того чтобы системная плата среагировала на замыкание. В результате пользователь с прискорбием констатирует, что его ПК по-разному реагирует при нажатии на разные клавиши, в зависимости от степени физического давления пальца на клавишу. Это вызывает дискомфорт, и появляется желание купить новую клавиатуру.

Однако это еще не все негативные аспекты сложившейся ситуации. Попавшая внутрь клавиатуры жидкость и после свого высыхания опасна — она разводит плотно прижатые заводским способом друг к другу пленки с токопроводящими дорожками и контактными площадками клавиатуры. Этот дополнительный зазор (десятые доли миллиметра) между токопроводящими пленками клавиша, чтобы осуществить замыкание, должна преодолевать уже с усилием.

Не спешите выбрасывать свою "подмоченную" клавиатуру и покупать новую, ее можно реанимировать. Это под силу сделать, пожалуй, любому пользователю ПК. Корпус клавиатуры аккуратно разбирают, шурупы складывают в отдельную баночку, чтобы потом не осталось лишних или не появилась "недостача". Мягкой тряпочкой, слегка смоченной с растворителе (№646, №650, №651), аккуратно протирают липкие места на контактных пленках, стараясь не задеть пластмассовый корпус (он от воздействия растворителя плавится). Обычно пораженные места достаточно протереть один раз. Обработку контролируют визуально и заканчивают по мере удаления засохшей пленки от попавшей внуть корпуса клавиатуры жидкости.

Затем в свободные от контактов и проводящих дорожек участки нижней контактной пленки клавиатуры, в местах ее максимального отслоения от верхней пленки, наносят микрокапли клея "Супер-Момент" (как правило, пленки отслаиваются частично, а не полностью, это облегчает реанимацию). Выждав 1...3 с, верхнюю контактную пленку равномерно с усилием прижимают к нижней. Теперь клавиатура приобрела почти первозданный вид, и ею (после сборки корпуса) можно пользоваться практически неограниченное время, до следующего заливания жидкости.

Описанным способом мне удалось реанимировать порядка двух десятков принесенных в ремонт клавиатур.

"Сносим" Linux

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

Этот материал рассказывает о пытливом уме пользователя, который, не придумав ничего лучшего, решил удалить при помощи старых DOS-приложений Linux со своего компьютера. Задача показалась архиинтересной, и вот что из этого вышло.

"О том, как установить Linux, написано много, но как ее удалять, что-то не пишут," — сказал я сам себе, глядя на свой IBM286. Нужно придумать что-то новое для моего любимого "downgrade", только на этот раз провернуть это на компьютере Pentium. Для тех, кто не понимает, о чем идет речь, придется рассказать. Есть такое веяние в компьютерном мире — "downgrade", когда чудные люди, вроде меня, пытаются при помощи старых программ или DOS-приложений сделать на компьютере что-то новое. В этот раз "жертвой" эксперимента стала не Windows, а Linux Mandrake 8.0. В компьютерных "зверствах" также принимали участие 3 раздела FAT 32 и кучка старых добрых DOS-программ, которые могут прикончить на диске все, что не так лежит. Итак, ситуация, банальная до простоты.

После того как я установил Linux Mandrake 8.0 на место бывшего диска D:, мне странным образом сразу же захотелось ее удалить. Для этого мне сначала нужно было избавиться от менеджера загрузки под названием Lilo, который, как всегда, загружался перед моей любимой надписью "Starting Windows". Для этого мне пришлось загрузиться с дискеты и выполнить команду **fdisk /mbr**, которая восстанавливает загрузчик, если он поврежден. Перезапустив компьютер, я убедился в том, что Lilo бесследно исчез.

Далее мне предстояла еще более сложная задача — вернуть раздел на его прежнее место. Для этой цели я решил воспользоваться Norton Disk Doctor (NDD) из пакета NU2000. Обычно он находится в распакованном виде на диске с инсталляцией. К сожалению, у меня, как и у большинства пользователей, в компьютере установлен только один привод CD-ROM, поэтому пришлось позаботиться, чтобы его было видно с дискеты под DOS. Для этого в файле autoexec.bat нужно вписать строку

```
A:\DOS\MSCDEX.EXE /S /d:mscd001.
```

Предварительно файл mscdex должен быть записан в папку DOS на диске A:, и в файле config.sys нужно добавить строку

```
DEVICE=A:\gscdrom.sys /d:mscd001
```

Файл gscdrom.sys — это DOS-драйвер CD-ROM, обычно он есть на дискете, которая прилагается к приводу. После всех этих манипуляций можно работать с компакт-дисками под DOS.

Итак, загрузившись с дискеты, я решил проверить возможность восстановления раздела FAT 32. Для этого я запустил NDD для DOS (Norton Disk Doctor for DOS). Он, "покряхтев" жестким диском, выдал в красивом зеленом окне сообщение, что обнаружен раздел DOS, и попросил разрешения попытаться его восстановить. И я дал "добро". После нескольких минут скрипения жестким диском, NDD выдал сообщение, на этот раз в красном окне, что все восстановлено, и нужно перезагрузить компьютер. Что и было мною сделано.

После перезагрузки компьютера, в списке доступных дисков в Norton Commander я обнаружил диск D:, который еще совсем недавно был разделом Linux. После этого оставалось только заново отформатировать диск D:.

Вот так при помощи теории "downgrade" можно справится с любой проблемой.

Thief: Deadly shadow

А.ВЕНДИЛОВСКИЙ,
г.Минск.

... Вы думаете, легко быть профессиональным вором? Я уже битый час торчу в тени этого здания, ожидая, когда этот глупый стражник или повернется ко мне спиной, или пойдет в обход. Уже третий час ночи, и думаю, ему очень хочется вздремнуть. Если я правильно рассчитал, то раньше чем взойдет Луна, я должен оказаться на крыше этого Хранилища. Ох, прошу прощения, я забыл представиться, меня зовут Гаррет. Некоторые недалекие люди зовут меня воришкой, тем самым пытаясь серьезно меня унизить. На самом деле я один из самых профессиональных взломщиков в этой стране, достаточно часто работающий на Корону, выполняя весьма скользкие поручения власти, а также на себя, чтобы обеспечить надлежащий уровень собственного существования, не говоря уже о редкой страсти к прекрасному.

Если вы решили, что перед вами этакий оборванец, то смею вас уверить, вы вновь сильно ошиблись. Я получил серьезное образование в одном очень странном монашеском ордене, целью которого было... В общем, говорить о них мне сейчас не очень хочется, воспоминания о том, как один из членов этого ордена едва не привел мир к катастрофе, до сих пор заставляют меня просыпаться по ночам (естественно, в те редкие ночи, когда я не на работе). Поддержать беседу с любым дворянином на практически любые темы, от моды до политических интриг, мне не составит труда, да и книг я прочитал, думаю, больше, чем вся наша аристократия вместе взятая. Но эта жажда приключений, пьянящее душу ощущение безграничной свободы просто сводят меня с ума. О, когда в твоих руках оказывается бесценная вещь, и ты можешь наслаждаться игрой света на гранях бриллиантов, ощущать их в своих руках, вспоминая, как был на волосок от гибели — наверное, это не позволяет стать моей жизни пресной, позволяя быть самим собой. Я не люблю насилие, и убиваю лишь спасая свою жизнь, когда другой альтернативы просто не остается. Я известен этим всем, как и своим талантом подбирать отмычки к практически любым замкам. Меня очень смешит, когда я вижу на витрине замок с "атигарретовской" защитой и гарантией, что я его не вскрою. Помню, как-то, вдоволь нагулявшись, с хмельной головы, я забрался в лавку ремесленника, вскрыл все "антигарретовские" замки, а потом во дворе выложил из них свое имя. Никогда до этого не видел, чтобы подобная шалость вызвала столько переполоха.

Но это все дела минувшие. А вот настоящее меня пугает, и будущее скрывается во мраке. Все началось весьма безобидно — сначала мне в руки попал небольшой, но весьма симпатичный амулетик, а вместе с ним и странная переписка о древнем пророчестве, найденном в катакомбах под городом. Я очень осторожно отношусь к магии, так как сам видел, на что эта штука способна. В пророчестве говорилось о человеке, который с помощью Талисмана погубит город. Жители города пытались спасти Талисман и уничтожить этого человека. Я прочитал это пророчество раз, другой, и у меня под ложечкой нехорошо засосало — человек, которого весьма точно описывало пророчество, был похож на меня как брат-близнец. Очень скоро мне пришлось убедиться, что подобные мысли пришли не только в мою голову, а значит, за мной уже началась охота. Для меня единственный шанс выжить — понять, что происходит, и один из ключиков к разгадке находится там, внутри этого здания.

... Так долго продолжаться не может! Пускаем в переулок шумовую стрелу. Меня всегда поражало, сколько грохота от этой маленькой безделушки. Пока стражник удивленно таращится в темноту, проскальзываем за его спиной в коридорчик, не забыв погасить факел на стене. Во тьме я невидим. Бесшумной тенью пробираюсь вверх по лестнице и практически упираюсь носом в спину еще одного стражника. Легкий взмах дубинкой, и обмякшее тело устраивается в самом темном углу комнаты. Завтра его ждет головная боль, но и только. Замок на сейфе весьма сложен, и мгновенно не откроется, я чувствую каждое движение шестереночек там, под сталью. В коридоре послышались шаги, руки стали еще быстрее вращать тонкую проволочку. Сзади меня со скрипом начинается открываться дверь, в этот момент раздается громкий щелчок. Одной рукой хватая содержимое ящика, второй я бросаю в сторону двери световую гранату, и в миг, когда яркий свет на время ослепил опешившего стража, прыгаю в окно. Следующий ход будет мой!

Именно первая часть этой игры в свое время стала для многих любителей экшенов откровением, введя даже новый раздел игр — "стелс-экшены", где ваша задача — не покрошить врагов как капусту, а сделать все тихо и бесшумно. Между второй и третью частью был огромный перерыв, фирма-прародитель игры почила в Бозе, сведения о начале разработки прерывались сведениями о заморозке проекта, студия, взявшая на себя смелость продолжить серию, в один прекрасный день лишилась высшего руководства в лице Джона Ромеро — все это заставляло серьезно задуматься о судьбе "Вора". Конечный результат оказался не так плох, хотя впечатление осталось несколько двойственное.

Сначала, по традиции, о движке игры. Отказавшись от собственных разработок (да уж, эти ходячие параллелепипеды из первой и второй части под кодовым названием "стражники" я не могу забыть до сих пор!) товарищи прикупили лицензию на Unreal W. В настоящее время это один из самых качественных движков, позволяющий создавать огромные пространства, и с очень красивой графикой, особенно что касается персонажей игры. Эффекты света и тени, так важные здесь, тоже на уровне. Добавив для большего глянца некоторый косметический лоск, создатели не ввели в движок каких-то особых нововведений, а на что он способен, думаю, каждый знает, благо, игр на нем уже создано предостаточно.

Далее разработчики стали исходить из принципа, что не надо чинить то, что не сломано, и что новое — это хорошо забытое старое. Поэтому различия между первыми двумя частями и третьей становятся практически эфемерными. Арсенал игрока не претерпел изменений — лук со стрелами: шумовыми, обычными, с водой (в т.ч. и святой), нож, дубинка, отмычка... Такая же реакция на свет и шум и перемещение игрока. Интерактивность мира тоже не претерпела изменений — патрулирующие или стоящие стражники, время от времени разговаривающие о своих делах, ночные прохожие... Геймплей не изменился практически ни на йоту, что, с одной стороны, хорошо, так как сохраняется вся атмосфера игры, а с другой стороны — гложет ощущение, что ждал чего-то большего, чем просто технически более грамотный адд-он. Одно радует — что многие молодые игроки смогут оценить все прелести законодателя жанра.

На этом можно было бы и завершить статью, но ложка дегтя в бочке меда все же присутствует. Как ни странно, это относится именно к нововведениям! Видимо, разработчики все-таки хотели вставить свою фишку и изменили принцип взламывания дверей. Если раньше это сводилось к перебору отмычек, то теперь превратилось в долгий и нудный процесс вращения мышкой, когда, с трудом сдерживая зевок, наблюдаешь за неспешным вращением шестеренок замка. С непривычки убиваешь на это довольно много времени, и именно поэтому игру просто забрасывают. Остается надеяться, что достаточно скоро выйдет код или мод, убирающий подобное безобразие.

Системные требования

Minimum:

COMPUTER — IBM PC or 100% compatible
OPERATING SYSTEM — Windows 2000 or Windows XP (95/98/ME/NT not supported)
CPU — Intel Pentium IV 1.5 GHz (or AMD Athlon XP equivalent)
MEMORY — 256 MB system memory
GRAPHICS — 64 MB video memory, Direct3D 9.0, and Pixel Shader 1.1
SOUND — 100% DirectSound 9 compatible sound card
HARD DRIVE — 3 GB free hard disk space
CD-ROM — Quad-speed CD drive or DVD drive (DVD required for European versions)
INPUT DEVICES — Keyboard and mouse

Recommended (are as above with the following changes):
CPU — Intel Pentium IV 2.0 GHz (or AMD Athlon XP equivalent)
MEMORY — 512 MB system memory
GRAPHICS — 128 MB video memory, Direct3D 9.0, and Pixel Shader 1.1.

Диск с игрой предоставлен интернет-магазином "MediaCraft" www.MediaCraft.shop.by

КУПЛЮ, ПРОДАМ, ОБМЕНЯЮ

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

119454, г.Москва, а/я 37 или
220095, г.Минск-95, а/я 199,

передавать по тел. в Минске ***(017) 249-41-47****, либо через E-mail:*

rl@radiopage.by
rm@radio-mir.com
WWW: ***http://radio-mir.com***

Продам недорого импортный контроллер для общей шины ("Электроника 60") с даумя винчестерами; приборы Ц-315, Ц-435, Ц4317М, Ц20-05, ТЛ4М, ПНТ-59, Ц4312, ГНЧР-2, ГЦТ-03 (PAL), ГИР; цифровую шкалу; однокассетную магнитолу.
140033, Московская обл., пос.Малаховка, Быковское шоссе, 45 — 32,
Кутузов Е.В.

Куплю плату ISA WinRadio за символическую цену.
E-mail: rogojinandrey@mail.ru

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

Продам-куплю программы "ZX-Spectrum" в среде TR-DOS и ОС CP/M. Предлагаю каталог с дискетой.
632383, г.Куйбышав, НСО, 1 — 20 — 45,
Третьяков А.А.

Меняю: мультиметр цифровой МИЦ-01, компьютер БК-01, видиконы ЛИ-441 с ОС-5-01, телекамеру КТП-63, клавиатуру герконовую ЕС 9003 на радиостанции "Ангара", Р-143 или аналогичные.
399540, Липецкая обл., пос.Тербуны, ул.Первомайская, 13, Харитонов В.П.
Тел. 8-274-2-29-29, с 19.00.

Продаю тюнер winradio WR-1000 (частота до 1,3 ГГц, CW, SSW, AM, FMN) в виде платы в компьютер.
Тел. 8-029-644-88-90.
E-mail: winradio@mail.ru

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

Куплю запрограммированное ПЗУ для "ZX-Spectrum 128 к" с TR-DOS версии 5.03, схему "ZX-Spectrum 128 к" с музыкальным синтезатором на МС YM2149F и с контроллером НГМД.
617833, Пермская обл., г.Чернушка, ул.Дружбы, 15, Хатмуллину И.Х.

Ученик 8-го класса **примет с благодарностью в дар** любой компьютер б/у.
Ищу схему "Романтика" первых моделей.
Куплю привод CD-дисков, можно б/у, но в хорошем состоянии.
640001, г.Курган, ул.Станционная, 8 — 26,
Шорин А.

Куплю SIMM для компьютера 486DX, HDD емкостью до 1 Мб.
E-mail: servis-centrl@mail.ru

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

Подписные индексы:

Радиомир

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

Радиомир. КВ и УКВ

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

Радиомир. Ваш компьютер

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

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

Получить подписку или заказать имеющиеся в наличии отдельные номера журналов можно из редакции. Текущие цены приведены в таблице. Наложенным платежом журналы не высылаются.

Год	Можно заказать следующие номера журналов			Расценки на 1 экз. (с учетом пересылки)		
	Радиолюбитель	Радиолюбитель. Ваш компьютер	Радиолюбитель. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
2000	3 — 6, 8 — 12	3 — 8, 10 — 12	6, 7, 9, 11, 12	20	800	6
2001	4 — 5	2, 5, 6	2, 3, 5, 6	25	900	6
	Радиомир	Радиомир. Ваш компьютер	Радиомир. КВ и УКВ	По России и СНГ (рос. руб)	По Беларуси (бел. руб.)	По Украине (гривны)
2001	7, 9, 10, 12	7, 9 — 12	7, 8, 10, 11	30	1000	7
2002	1 — 12	1 — 12	1 — 3, 5 — 12	35	1500	7
2003	1 — 12	1 — 12	1, 5 — 8, 10 — 12	40	1800	8
2004	1 — 6, 8 — 12	1 — 6, 8 — 12	1 — 12	45	2100	9
2005	1 — 12	1 — 12	1 — 12	45	3000	11

Наши платежные реквизиты:
- для жителей России и стран СНГ (кроме Беларуси) —
получатель: ООО "Радиомир Пресс", ИНН 7729404741, КПП 772901001,
р/с 40702810900010000084 в ООО КБ "Агропромкредит", ф-л Центральный, г.Москва,
корр. счет 30101810500000000109, БИК 044525109
Адрес банка: 125315, г.Москва, Ленинградский пр-т, дом 78, корп. 4;
- для жителей Беларуси —
получатель: УП "РЛД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" а г.Минске код 121,
Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.

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

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

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

Приобретение отдельных номеров журналов

В РОССИИ:
В **ЗАО "Интерпресс — Экспресс"**: тел. в Москве (095) 208-85-50, 208-67-55, e-mail: inter@aif.ru, http://www.interpress-express.com.

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

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

В **ООО "Альянс-Книга"**: 115533, г.Москва, ул.Нагатинская, д.6, эт.1, оф.VII (ст. метро "Нагатинская"), тел. (095) 258-91-94, 258-91-95, 258-91-96, 111-05-98, 111-63-28. E-mail: abook@mail.ru, abook@inbox.ru
На радиорынках: "Царицыно" (место 13/А), "Митино" (ряд 1, контейнер 17 и место Т-8).

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

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

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

Компьютеры для дома и офиса

от компании СитиПринт

Компьютер – это инвестиция в развитие Вашего бизнеса!

Приобретая в компании "СитиПринт" компьютер на базе процессора Intel® Pentium® 4 с технологией HT, вы инвестируете в свое будущее.

Новый компьютер предоставит необходимую поддержку для Вашей автоматизированной системы бухгалтерского учета, позволит оперативно создавать и рассылать электронные письма клиентам и поставщикам, а также изготавливать профессиональные презентации.

Подарите Вашему бухгалтеру точность современных технологий.

СитиПринт

КОМПЬЮТЕРЫ И ОРГТЕХНИКА

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

RADIO РАДИО PAGE ПЭЙДЖ

БелРадиоПэйдж 233-00-00

МЕЖРЕГИОНАЛЬНАЯ ПЕЙДЖИНГОВАЯ СЕТЬ

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

www.radiopage.by

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

7 способов отправки сообщений!

Стационарные охранные системы для дома и офиса

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

Атлант Моторс

Спутниковые автомобильные охранные системы

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