http://radio-mir.com
индексы: 48925, 45995

радиомир

Ваш компьютер

9•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

intel inside pentium 4

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

Главный редактор
Ольга СТРЫЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

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

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

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

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

220095, РБ, г.Минск-95, а/я 199

E-mail: rl@radiopвge.by
rm@radio-mir.com
WWW: http://radio-mir.com

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

Материалы для публикации принимаются в рукописном, печвтном и электронном вариантах

Требования к графическим материалам рекламного характера в электронном виде: CorelDRAW до 10.0, асе 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 совпедает с мнениями авторое*

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

Подписано к печати 27.07.2004 г.
Формат 60 x 84 1/8.
Печать офсетная. 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ № 2350.

радиомир
Ваш компьютер

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

№9/сентябрь/2004

ЧИТАЙТЕ В НОМЕРЕ

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

В.КУЦ,
E-mail: victor_kootz@mail.ru

TV-тюнеры — быть или не быть?

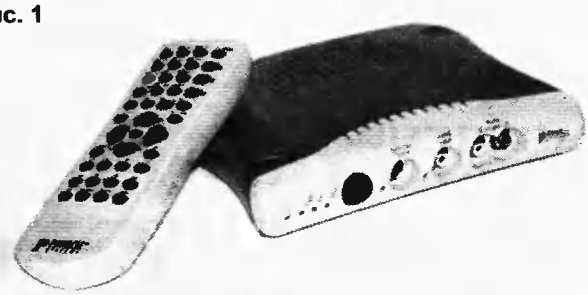
В последние годы, когда общее количество одних только эфирных телеканалов в нашем городе приблизилось к двум десяткам, оказалось, что один телевизор в каждой семье — это слишком мало. Да что там один — многим и двух уже не хватает. Но в условиях наших малогабаритных квартир найти место для второй, а тем более, третьей "говорящей головы" становится весьма проблематично. Но для владельцев персональных компьютеров достаточно эффективным решением проблемы "теленезависимости" от остальных членов семьи может стать TV-тюнер, обеспечивающий прием видеосигнала от различных источников и просмотр телепередач на экране компьютерного монитора. Помимо просмотра телепрограмм, что является основной задачей любого TV-тюнера, подавляющее большинство современных моделей могут еще и осуществлять захват отдельных кадров или даже записывать целые видеоролики непосредственно на жесткий диск, принимать и просматривать текстовую информацию (телетекст) и многое другое. Многие TV-тюнеры выпускаются с интегрированным FM-тюнером, позволяющим прослушивать радиопередачи в диапазоне УКВ/FM, что может быть весьма полезным для любителей работать за компьютером под ненавязчивый фон радио. Так же как и телевизионные программы, радио можно не только слушать, но и записывать. Кстати, видеоданные телетюнер способен получать не только из эфира, но и от таких устройств как, например, бытовые видеокамеры или видеомагнитофоны. Также следует отметить, что качество изображения у компьютерных мониторов (как CRT, так и LCD) заведомо лучше, чем у традиционного кинескопа бытового телевизора (например, по разрешению, яркостному диапазону, глубине цвета).

В первом приближении все компьютерные "телевизоры" можно разделить на внутренние и внешние. Понятно, что внутренние тюнеры устанавливаются в компьютер, являясь обычными платами расширения, ну а внешние представляются отдельными устройствами с собственным питанием.

Внешние TV-тюнеры

На **рис.1** изображен внешний USB 2.0 TV-тюнер Pinnacle PCTV Deluxe.

Рис. 1



Внешние TV-тюнеры, в свою очередь, бывают двух типов: использующие интерфейс USB и подключаемые в разрыв кабеля между монитором и графическим адаптером компьютера.

Особая ценность тюнеров второго типа заключается в том, что они позволяют смотреть ТВ без включения ПК, используя лишь компьютерный монитор — телевизионная картинка появляется на экране сразу же после включения устройства. Преимущества подобного подключения очевидны: не надо вскрывать системный блок и что-то туда устанавливать. Это раз. Во-вторых, компьютер как таковой внешнему тюнеру не нужен — нужен лишь монитор, что исключает процедуру установки и настройки программного обеспечения. Налицо действительная легкость подключения. Третьим плюсом такой конструкции является то, что внешний вариант тюнера, вследствие отсутствия помех и наводок от близко расположенного компьютерного оборудования, обладает гораздо лучшей чувствительностью, чем внутренний, что является наиважнейшим условием получения качественного изображения.

Недостатком, и притом довольно серьезным, можно считать некоторую функциональную ограниченность подобного типа устройств — в большинстве случаев подобные TV-тюнеры не в состоянии принимать радиопередачи. Но куда более существенным их недостатком является невозможность просмотра телепередач во время работы, т.е. в оконном режиме (доступен только полноэкранный режим), а также записи понравившихся передач на жесткий диск компьютера. Таким образом, внешние TV-тюнеры, подключаемые в разрыв кабеля монитора, нельзя признать полноценными компьютерными устройствами — это, скорее, своеобразный подвид бытовой техники, поэтому в последнее время наметилась отчетливая тенденция их вытеснения TV-тюнерами, выполненными в виде внешних USB-устройств.

Тюнеры этого типа принимают с антенны высокочастотный сигнал, декодируют его, оцифровывают и передают в компьютер в виде цифрового потока, используя универсальную последовательную шину USB. К вышеперечисленным достоинствам внешних TV-тюнеров у USB-устройств добавляется еще и отсутствие необходимости внешнего питания, так как они получают его непосредственно по шине USB, что делает их хорошим выбором для использования с мобильными компьютерами.

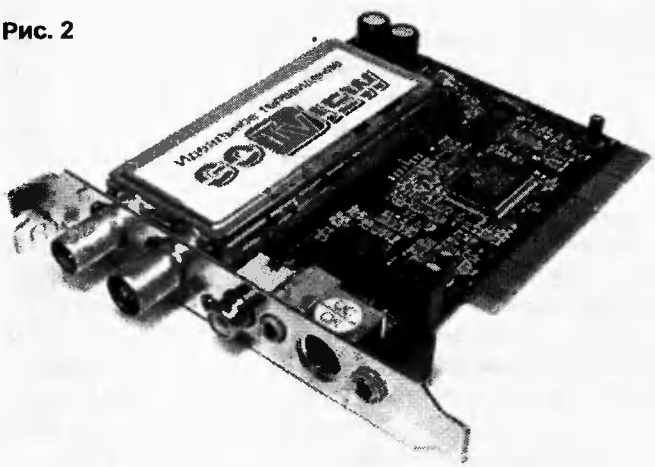
Разумеется, и у такого решения есть свои минусы. Прежде всего, качество получаемого изображения ограничивается пропускной способностью использованного интерфейса. В случае USB 1.1 максимальный поток данных составляет 12 Мбит/с, но это в теории, в реальности все обстоит гораздо хуже — обычно скорость обмена не превышает 1,5 Мбит/с, чего, конечно, для передачи видеоизображения слишком мало. Поэтому в TV-тюнере изображение сжимается (чаще всего по алгоритму MJPEG). В компьютере же осуществляется операция декодирования MJPEG (а для современных высокопроизводительных процессоров это элементарная задача), и после этого видеопоток выводится на дисплей. Реальное разрешение, при котором не происходит "выпадение" кадров при полноцветной картинке, не превышает 352 x 288, чего по современным

меркам совершенно недостаточно. Но уже появились первые модели тюнеров, использующие современные высокоскоростные интерфейсы IEEE-1394 и USB 2.0 — в этом случае максимальная пропускная способность интерфейса составляет уже 400...480 Мбит/с, и распространен вид сжатия DV (базирующийся на MPEG-2). Но, к сожалению, пока ни одной модели с интерфейсом IEEE-1394 в продажу еще не поступало, а многие из имеющихся моделей с USB 2.0, особенно недорогие, не используют всех возможностей, предоставляемых скоростным интерфейсом.

Внутренние TV-тюнеры

На **рис.2** изображен внутренний TV/FM-тюнер GoTView PCI.

Рис. 2



Практически все современные внутренние TV-тюнеры выпускаются в виде PCI-карт, устанавливаемых в соответствующий слот материнской платы. Все они отличаются богатством сервисных функций, серьезно превосходя по этому параметру большинство внешних моделей, и ограничены лишь возможностями используемого с ними программного обеспечения. Так, окно для просмотра телепередачи формируется программно, поэтому легко меняется его положение и размер на рабочем столе, оно может быть свернуто, развернуто на весь экран, или может превратиться в живые обои рабочего стола Windows, а в отдельных моделях можно даже регулировать его прозрачность. Как правило, все внутренние устройства имеют развитое программное обеспечение, позволяющее сохранять видеоизображение и отдельные кадры в файлах, средствами центрального процессора сжимать его по алгоритмам MPEG или MJPEG, осуществлять запись (в том числе и по расписанию). Список достоинств можно продолжать довольно долго, но владелец такого тюнера обязательно припомнит и недостатки, главным из которых стоит признать относительно высокий уровень помех и наводок на радиочастотный модуль тюнера от работающего компьютерного оборудования, что отрицательно сказывается на качестве выводимого изображения. Особенно это актуально для тех, кто имеет антенну не самого высокого качества. Кроме того, на материнской плате занимается PCI-слот и, как минимум, используется одно аппаратное прерывание, которых и без этого вечно не хватает. При просмотре ТВ сильно загружается PCI-шина, а сжатие и оцифровка изображения в файл сильно загружает еще и процессор (вместе с дисковой подсистемой) — не всякий современный Pentium 4 справится со сжатием MPEG в реальном времени и с высоким качеством.

Для обеспечения звукового сопровождения внутренние TV-тюнеры, как правило, подключаются к внешнему линейному входу аудиокарты компьютера. В случае отсутствия аудиокарты, к линейному выходу самого тюнера можно подключить активные акустические системы.

В отдельный подвид внутренних TV-тюнеров можно выделить комбинированные устройства. В них тюнеры конструктивно объединены на одной плате с видеоадаптером. Такие комбайны издавна являются "коньком" компании ATI (достаточно привести в пример известную серию видеоадаптеров ATI All-In-Wonder Radeon 9600/9800). Замечу, что в свое время специалисты из ATI потратили немало сил, чтобы достичь хорошей работы своих видеочипов с форматами MPEG. Недостаток комбинированных устройств очевиден: если TV-тюнеры покупаются с расчетом на долгую эксплуатацию, то самые современные видеоадаптеры морально устаревают очень быстро, буквально за год-два. Ну а "апгрейд" комбинированного видео обойдется значительно дороже обычного.

Внутреннее устройство TV-тюнеров

Аппаратная начинка всех типов современных TV-тюнеров предельно унифицирована. С антенного входа видеосигнал поступает на высокочастотный (ВЧ) модуль, в котором происходит его усиление, фильтрация и преобразование. ВЧ-модуль защищен металлическим экраном, роль которого заключается в защите слабого полезного сигнала от помех и наводок от работающих компонентов системного блока компьютера. В его составе часто присутствует мультиплексор — электронный переключатель, который осуществляет выбор одного из нескольких источников видеосигнала (современные TV-тюнеры, помимо стандартного эфирного или кабельного ТВ, как правило, также способны принимать сигналы с видеовходов — композитного или S-Video). Дальше аналоговый видеосигнал передается в другой модуль — декодирующий, основу которого составляют аналогово-цифровой преобразователь (АЦП) и собственно декодер видеосигнала, чаще всего выполненные в виде одного чипа. В нем сигнал дискретизируется и преобразуется в цифровую форму. С этим сигналом впоследствии и работает компьютер. Этот модуль, пожалуй, является важнейшим компонентом тюнера, предоставляющим последнему самые широкие функциональные возможности. Практически все современные TV-тюнеры, представленные на нашем рынке, построены на нескольких чипах.

Conexant Fusion 878A (он же Bt 878) — уже устаревший чип, хотя все еще довольно распространенный. Он имеет 8-разрядный АЦП и декодирует видео в стандартах NTSC/PAL/SECAM и поддерживает захват изображений до 768 x 576 точек (PAL). Чип Fusion 878A обеспечивает вполне приличное (но не более того) качество изображения, благодаря же своему "почтенному" возрасту и широкой популярности, он хорошо совместим с большим количеством альтернативного "софта".

Philips SAA713xHL — серия более новых чипов с 9-битовым АЦП демонстрирует лучшее на сегодняшний день качество изображения для всех вариантов сигналов PAL, NTSC и SECAM и одинаково хорошо подхо-

дит для захвата как аналогового, так и цифрового форматов телевещания. Разница между SECAM и PAL на нем практически незаметна, цветопередача очень хорошая, вполне сопоставимая с обычным бытовым телевизором. Поэтому TV-тюнеры на чипах этой серии сегодня являются самыми популярными в России. Декодер от Philips существует в четырех модификациях: 7130HL (базовая и самая простая версия), 7133HL, 7134HL и 7135HL. Хотелось бы обратить ваше внимание на то, что поддержка функции цифрового кодирования стереозвука A2 и NICAM реализована только в модификациях 7134HL и 7135HL (последний поддерживает еще и Dolby Pro Logic). А чип Philips SAA7133HL отличается от SAA7130 возможностью декодирования стереосигналов формата NTSC, принятого в США и Японии.

Conexant CX23881 — совсем новый 10-битный чип, прямой конкурент линейки Philips SAA713xHL. Он обеспечивает неплохое качество изображения (но SECAM все-таки у Philips будет немного получше), однако, по причине новизны чипа, пока еще наблюдается плохая его совместимость с программным обеспечением сторонних производителей.

При этом следует отметить, что разница в качестве изображения (в первую очередь это касается цветопередачи) между старыми чипами (878-й серии) и новыми продуктами Philips и Conexant, конечно, имеется, однако она сравнительно невелика. Гораздо большее влияние на качественные показатели оказывает программное обеспечение и качество драйверов.

Что касается звукового сопровождения телетрансляций, то потенциальная возможность вывода звука по PCI-шине заложена во многих чипах, на базе которых производятся ТВ-тюнеры. Однако реально такая возможность реализована и доведена до конца только для чипа Philips SAA7134. Причем, скорее всего, не для любого ТВ-тюнера на базе этого чипа, а только для тех, в которых реализована полноценная поддержка программы Fly2000TV. Сегодня только в этой программе имеется реально работающая поддержка вывода звука по PCI-шине. Такая функция интересна даже не столько возможностью избавиться от лишнего кабеля для подключения аудиовыхода тюнера к звуковой карте, сколько возможностью при захвате обходиться вообще без звуковой карты, поскольку звук оцифровывается, так же как и видео, чипом тюнера.

Видеозахват

Современные TV-тюнеры позволяют получить достаточно высокое качество записи как с TV-, так и с RCA- или с S-Video-входов, поэтому все более популярным становится их использование для создания или редактирования любительского видео. И здесь сегодня сложилась двойственная ситуация. С одной стороны, платы на Bt 878 демонстрируют полную совместимость с большей частью имеющегося программного обеспечения, в том числе и стороннего, а с другой стороны — качество изображения тюнеров на этом чипе уступает более новым чипам CX23881 и SAA713xHL, а для захвата это очень важно — гораздо важнее, чем для просмотра. Платы же на новых чипах в большинстве своем все еще плохо совместимы со сторонним ПО, а "родное" — сплошь и рядом откровенно "сырое". Остается надеяться, что в ближайшее время "софт" догонит "хард", и будет нам счастье... Нет, к сожалению, не будет. Ведь далеко не все из нас являются владельцами последних моделей высокопроизводительных процессоров, позволяющих компрессировать видеосигнал в реальном времени, или десятков, а то и сотен гигабайт свободного дискового пространства для сохранения несжатого изображения. Ведь без соблюдения одного из этих условий (а лучше — обоих сразу) даже и мечтать не стоит о домашней видеостудии, пусть и самого что ни на есть начального уровня. Хорошим выходом в таком случае может стать использование TV-тюнеров, поддерживающих сжатие видео в MPEG2 на аппаратном уровне, кстати, стремительно набирающих популярность в последнее время. Являясь симбиозом старого доброго телевизионного тюнера и не менее "доброй" платы нелинейного видеомонтажа, такие устройства позволяют серьезно снизить требования как к CPU, так и к HDD компьютера. Конечно, серьезные задачи таким тюнерам все еще "не по плечу", однако для массовой оцифровки VHS "силенок" им вполне хватит.

Быть или не быть?

Так как же можно ответить на вопрос, вынесенный в заголовок обзора — быть или не быть TV-тюнерам? Еще совсем недавно TV-тюнеры (особенно недорогие внутренние модели) действительно не могли похвастать ни качеством изображения, ни особым богатством функциональных возможностей. Но, начиная с конца прошлого года, ситуация постепенно начала меняться. И, в первую очередь, это произошло благодаря новым декодерам Philips, с которыми качество изображения даже самых дешевых внутренних тюнеров существенно улучшилось, поэтому они из игрушки для энтузиастов сегодня уже могут превратиться в добротное (и при этом недорогое) решение для массового использования. Более того, в отдельных случаях им вполне по силам составить конкуренцию даже бытовым телевизорам. А для продвинутых пользователей может оказаться не только интересной, но и по-настоящему полезной другая новинка последнего времени — телевизионный тюнер с аппаратным сжатием видео, способный превратить обычный домашний компьютер в новомодный цифровой видеомагнитофон. Так что только этих двух моментов может оказаться вполне достаточно, чтобы сделать вывод — TV-тюнерам все-таки быть!

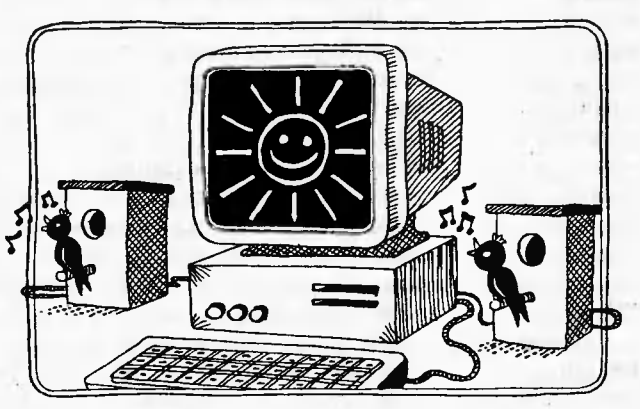


Рисунок О.Попова

Смотрим MPEG4 на телевизоре

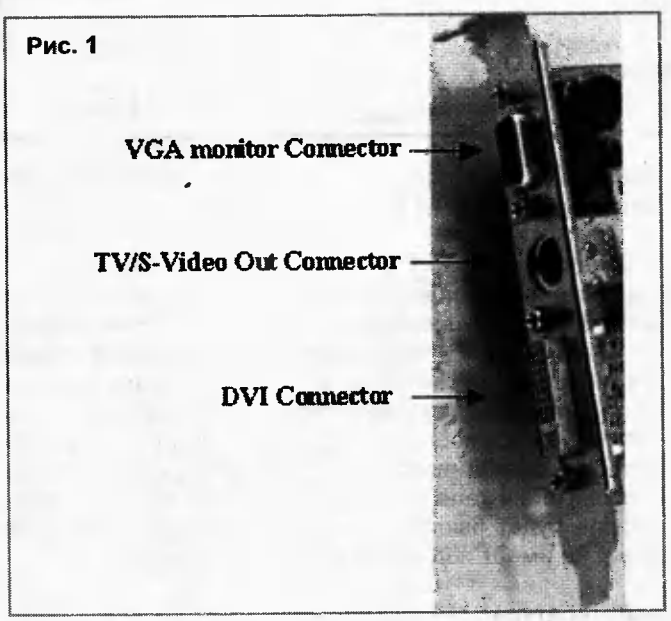
А.ГОРЯЧКИН,
г.Кыштым, Челябинской области.
E-mail: anatoliy_goryach@mail.ru

Если в обозримом будущем запланирован апгрейд видеосистемы компьютера, то лучше всего приобрести видеокарту с расширенными функциональными возможностями — с ТВ-выходом (TV-out). Это позволит выводить изображение не только на монитор, но и на телевизор. Стоимость такой видеокарты не намного дороже той, у которой TV-выход отсутствует, а пользу трудно переоценить. Просмотр видеофильмов и видеоклипов в формате MPEG4 и в других форматах цифрового видео на цветном телевизоре с размером экрана по диагонали 21...25 дюймов доставит больше удовольствия, чем на 15...17-дюймовом мониторе. Изображение на телевизоре отличается более яркими и насыщенными цветами, не говоря уж о размере картинки. Да и просидеть за монитором полтора-два часа подряд — это лишняя нагрузка для зрения, а перед телевизором на диване можно провести хоть целый вечер.

Для подключения телевизора к компьютеру, кроме обычного разъема D-SUB, к которому подключается монитор, на видеокарте должен еще присутствовать 4-пиновый разъем S-Video. Так как в телевизоре видеовход осуществляется через разъем RCA (типа "тюльпан"), для соединения понадобится специальный переходник S-Video-to-RCA [1]. Некоторые производители видеокарт (Palit, EliteGroup и др.) устанавливают на свои изделия разъем RCA, что упрощает подключение телевизора, так как переходник в этом случае не нужен. В продаже имеются видеокарты, оснащенные, помимо разъемов D-SUB и S-Video, разъемом DVI, предназначенным для подключения ЖК-монитора с цифровым интерфейсом (**рис. 1**).

Разъемы видеокарты
D-Sub — разъем для подключения ЭЛТ-монитора.
S-Video — разъем для подключения телевизора (разделенный видеосигнал).
RCA — разъем (типа "тюльпан") для подключения телевизора (композитный видеосигнал).
DVI — разъем для подключения ЖК-монитора с цифровым интерфейсом.
VIVO — Video In Video Out.

Рис. 1



На разъемы RCA и S-Video выведен аналоговый низкочастотный телевизионный видеосигнал. Отличие между сигналами на разъемах RCA и S-Video заключается в следующем. На разъем RCA выходит композитный сигнал, который получается в результате сложения яркостного сигнала Y, цветовой поднесущей, модулированной сигналами цветности U и V, а также синхроимпульсов. На разъеме S-Video присутствуют два выходных сигнала. Один из них несет яркостной сигнал Y и синхроимпульсы, а другой — цветовой модулированный сигнал C. Качество воспроизводимого изображения сигнала S-Video выше по сравнению с композитным видеосигналом, так как обеспечивается разрешение в 400 телевизионных линий, тогда как в композитном сигнале разрешающая способность ограничена всего 240 линиями. На видеокартах, оснащенных видеовходом, позволяющим оцифровывать аналоговое видео (с видеокамеры, с видеомагнитофона или телевизора) разъем S-Video служит как для входа, так и для выхода сигнала.

Если комплектация видеокарты — OEM, то в этом случае могут отсутствовать соединительные шнуры и переходники. Шнуры для подключения телевизора к компьютеру несложно изготовить своими силами. Понадобится сделать два шнура, один из которых предназначен для передачи видеосигнала, другой — для звука. Для этого надо приобрести соответствующие штекеры и кабель — потребуются три штекера RCA и один штекер Jack 3,5 mm Stereo для подключения звука (телефонный). Возможно, могут понадобиться иные штекеры — в зависимости от имеющихся в вашем телевизоре разъемов, например, SCART. Для передачи видеосигнала применяется тонкий коаксиальный (антенный) кабель с волновым сопротивлением 75 Ом, а для передачи звука используется экранированный двухжильный провод. При длине соединительного кабеля 7 метров и изображение, и звук у меня на телевизоре были устойчивыми, без помех.

Звуковое сопровождение обеспечивается встроенным аудиотрактом телевизора, правда, как правило,только в монофоническом режиме. При воспроизведении звука динамиком телевизора появляется дополнительное удобство — возможность регулировки громкости звука с помощью пульта дистанционного управления. Поскольку с компьютера выходит стереофонический звуковой сигнал, для подключения к монофоническому входу телевизора можно сделать простейший микшер, который сформирует суммарный моносигнал из сигналов левого и правого каналов. Любители стереофонического звука смогут озвучивать видеофильмы компьютерными активными звуковыми колонками, которые желательно разместить по бокам телевизора для достижения стереоэффекта.

После изготовления шнуров можно переходить к ком-

мутации. Соединять шнурами компьютер и телевизор рекомендуется в выключенном состоянии. В этом случае отсутствует риск повредить дорогостоящее оборудование. Видеосигнал, снимаемый с видеокарты, подключают к гнезду телевизора Video Input, звуковой сигнал — соответственно, к гнезду Audio Input. Разъемы Video Input и Audio Input обычно располагаются на задней панели телевизора.

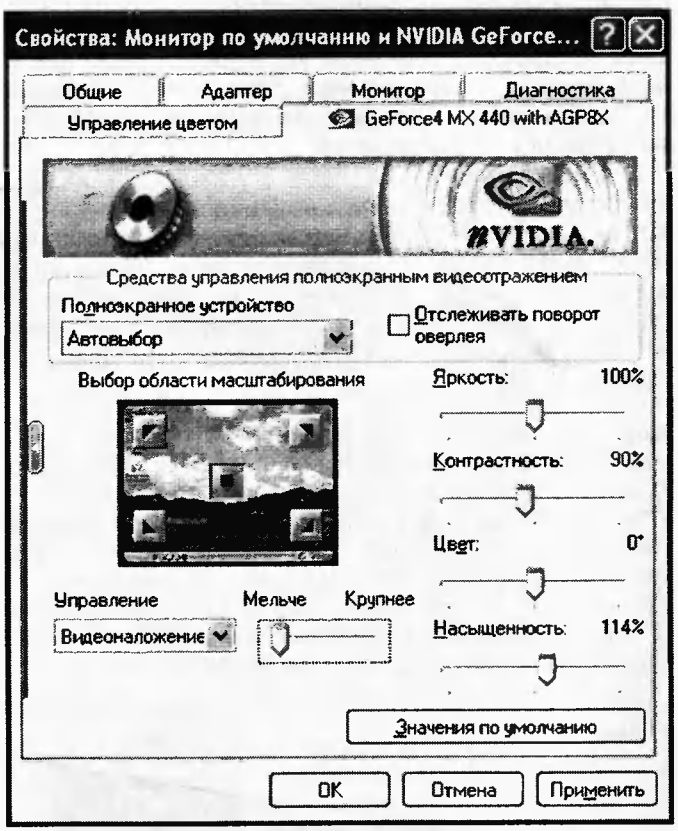
После соединения шнуров включают телевизор и компьютер. Для того чтобы телевизор смог отображать сигнал, подаваемый с компьютера, его переводят в специальный режим с помощью пульта дистанционного управления нажатием кнопки выбора внешнего входа (AV). В свойствах экрана выбирают дополнительный монитор (т.е. телевизор) и производят соответствующие настройки, в частности, включают опцию "Расширить рабочий стол на этот монитор" **(рис.2)**. После этого на экране телевизора появится изображение рабочего стола. Если видеокарта построена на чипсете от nVidia, то для воспроизведения видео в полноэкранном режиме необходимо установить настройки полноэкранного устройства в положение "Автовыбор" **(рис.3)**. После просмотра видеофильма отсоединять шнуры от телевизора и компьютера не обязательно, надо лишь отключить от компьютера (звуковой карты) звуковой кабель.

Для того чтобы видеофильм или видеоклип воспроизводился на телевизоре в полноэкранном режиме, совсем не обязательно включать полноэкранный режим в видеоплеере, с помощью которого ведется просмотр на мониторе. Более того, окно видеоплеера можно свернуть на панель задач или в системный лоток (System Tray), и продолжать параллельно работать на компьютере в другой программе, например, в Word. Проигрывание видеофайла при этом будет выполняться в фоновом режиме. Это удобно в том случае, когда ваши домочадцы хотят посмотреть новый "видик", а вам нужно в это время поработать на компьютере в другой программе. Одно другому — не помеха.

Желательно, чтобы телевизор и компьютер находились в непосредственной близости друг от друга (в пределах одной комнаты). Идеальным считается такое расположение, когда компьютер и телевизор будут находиться в пределах прямой видимости от точки просмотра, это позволит управлять с помощью пульта дистанционного управления не только телевизором, но и программным видеоплейером, например, Light Alloy v2.4 и выше. Для управления необходимо собрать небольшую схему, которая подключается к свободному COM-порту компьютера. Схема и подробное описание методики на русском языке находятся в файле помощи программы.

Рис. 2

Рис. 3



Проигрыватель Light Alloy доступен на официальном сервере программы http://la.video-soft.com.

Заключение

Применение видеокарты с ТВ-выходом позволяет расширить функциональные возможности современного компьютера и превращает просмотр видеофильмов в коллективное, семейное, а не персональное занятие. Для пользователей персональных компьютеров отпадает необходимость приобретения аналоговых и цифровых видеомагнитофонов. Бытующее мнение, что при подключении телевизора к компьютеру кинескоп телевизора будет быстро терять свои эмиссионные свойства, не имеет под собой никаких оснований.

Литература

1. А.Федоров. Переходник "S-Video".— Радиомир, 2004, №6, С.27.

А.ГРИНЧУК,
E-mail: gav@nihe.niks.by

Анимация с Macromedia Flash MX

Создание flash-сайта

После изучения основных возможностей Flash MX у нас наконец-то появляется возможность выбора: можно создать некоторое количество flash-клипов и разнообразить ими web-страницы, а можно попытаться соединить мини-фильмы в один большой, который и внешне, и поведением будет напоминать web-сайт. О создании клипов мы говорили практически во всех предыдущих статьях, здесь требуются, скорее, идея и фантазия. В создании же flash-сайта гораздо большую роль начинают играть чисто технические приемы. Список таких приемов не ограничивается двумя-тремя основными способами действий. Flash MX — достаточно мощный инструмент, в работе с ним постоянно имеются разные варианты ответа на вопрос "Как это сделать?".

В продолжение прежней линии, ограничимся достаточно простыми, но вполне работоспособными средствами. Для начала сформулируем основные требования к создаваемому сайту. Сайт состоит из ряда страниц, поэтому желательно сконструировать нечто, напоминающее навигационную панель. Конечно, flash-сайт невозможно представить без анимации, а также без ссылок на другие сайты (Интернет есть Интернет). Со ссылками более или менее все понятно — в качестве гиперссылок вполне подойдут кнопки с действием **getURL**. Внутри сайта можно организовать переходы на основе кнопок **goto**. Остается выбрать "кандидата" на роль web-страницы. Так как не хочется смотреть на постоянное мельтешение кадров, в качестве страницы лучше использовать остановленный кадр с ActionScript-кодом **stop**. Возникает вопрос: а где же будет анимация? Ответ заключается в использовании символов-клипов, которые будут проигрываться даже при остановленном кадре основной сцены. Итак, вырисовывается соответствие: сайт — фильм, страницы — кадры, flash-вставки — символы-клипы и, наконец, гиперссылки — кнопки.

Для рассмотрения основных технических вопросов нам будет достаточно небольшого трехстраничного сайта ("Главная", "О себе", "Ссылки"). В первую очередь вспомним, что при вставке нового ключевого кадра анимации происходит дублирование предыдущего. Поэтому, если создать кадр только с навигационной панелью и кнопками, снабженными необходимыми действиями, то не возникнет необходимости повторять эту работу трижды. В качестве фона выберем сине-зеленый цвет, размеры сцены фильма — 600×450 pt (**Modify/Document** — **рис.1**). Основа навигационной панели — белый четырехугольник. Затем создаем кнопку перехода на главную страницу (**Insert/New Symbol/Button**, имя символа — **Glavnaya**). Кнопка представляет собой прямоугольник с надписью (**рис.2**), причем цвет надписи будем изменять: в кадре **Up** — синий, **Over** — зеленый, **Down** — красный. Чтобы создать одинаковые по размеру кнопки, лучше поступить следующим образом. В открытой библиотеке фильма (**Window/Library**) необходимо щелкнуть правой клавишей мыши по кнопке и выбрать **Duplicate** (создание копии),

Рис. 1

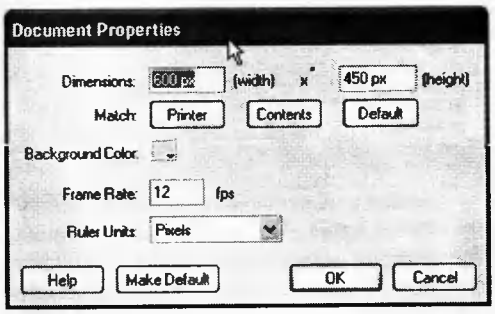


Рис. 2

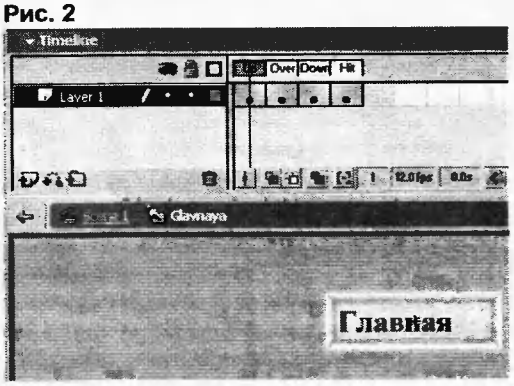


Рис. 3

Рис. 4

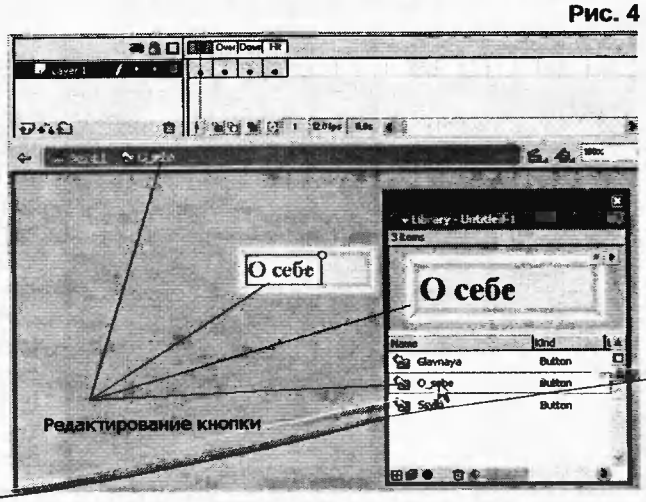


Рис. 5

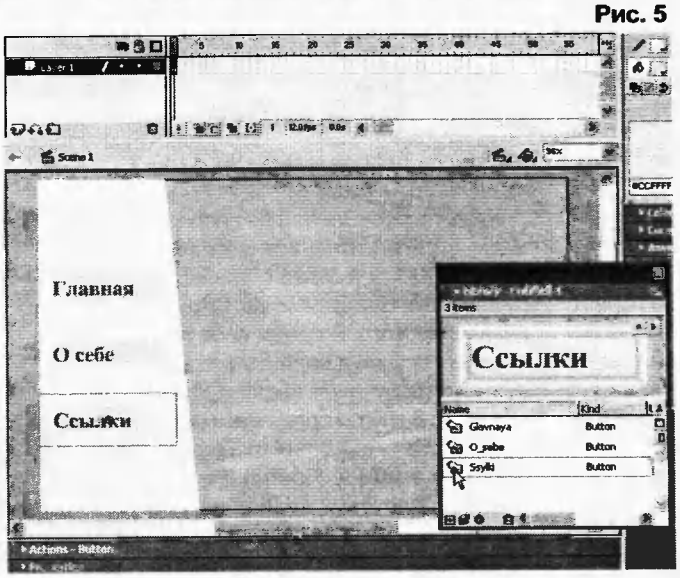


Рис. 6

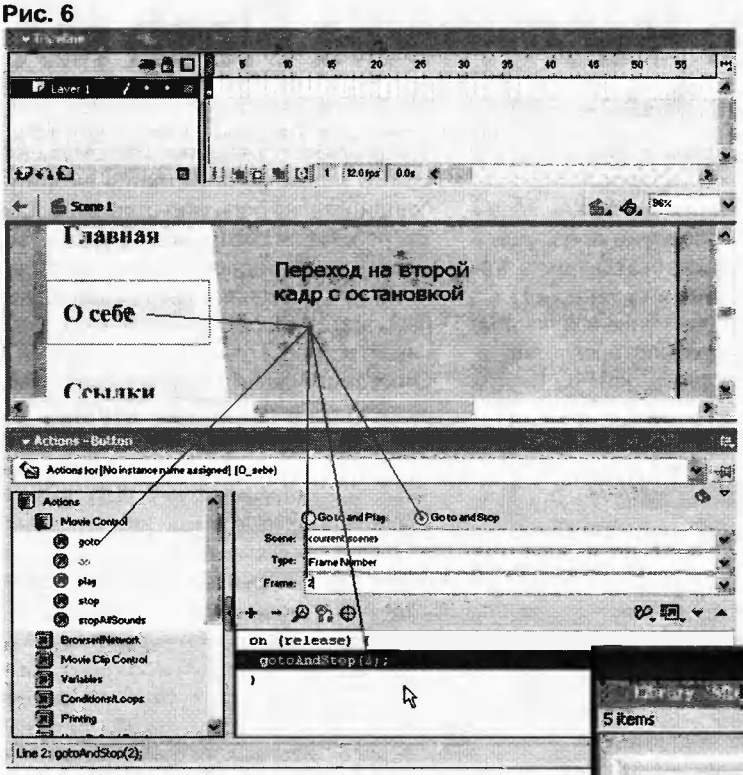


Рис. 7

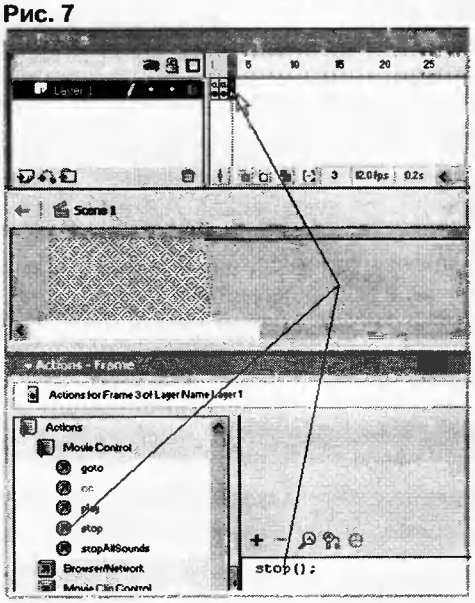


Рис. 8

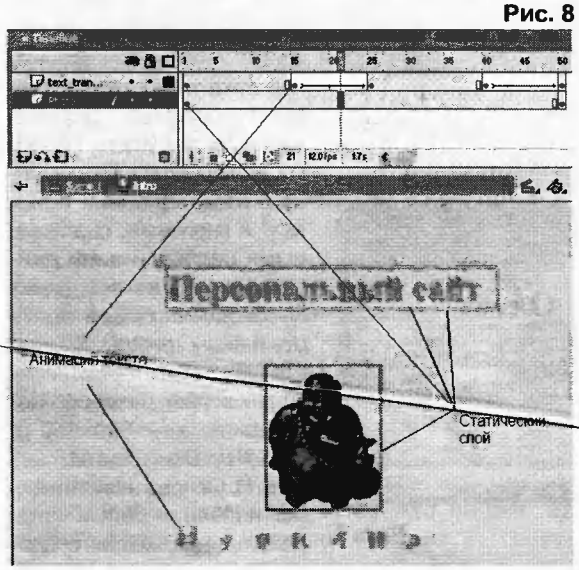


Рис. 9

затем копии переименовываются в **Ssylki** и **O_sebe** (**Rename — рис.3**). Остается двойным щелчком мыши открыть каждую кнопку и отредактировать в каждом кадре текст надписи (**рис.4**).

Следующий шаг — создание панели. Перетаскиваем кнопки из библиотеки (**рис.5**) и добавляем действия. С кнопкой **Glavnaya** связываем переход на первый кадр **gotoAndStop(1)**. Некоторые вопросы могут возникнуть с символами **Ssylki** и **O_sebe**. Здесь мы создадим переход с остановкой на (пока) не существующие кадры с номерами 2 и 3 (**рис.6**).

Все готово для создания "пустого" сайта. Второй и третий кадр делаем ключевыми (**F6**), и все три кадра по очереди снабжаем скриптом остановки (открываем панель **Actions** и выбираем команду **stop** — **рис.7**).

Настала пора заняться наполнением страниц. Вначале создаем символ-клип для первой страницы (**Insert/New Symbol/Movie Clip**) с названием **Intro**. В одном слое поместим рисунок и надпись (можно и две для получения эффекта тени — **рис.8**). В другом организуем морфинг текста — фамилия трансформируется в имя и обратно. Не забудьте в ключевых кадрах перед указанием типа анимации (**shape**) произвести разгруппировку текста двойным выполнением команды **Modify/Break Apart**. Возможна и такая ситуация, когда при создании анимации захочется использовать символы-заготовки из других файлов. Желание реализуемо — после выбора **File/Open as Library** (открыть как библиотеку) и указания нужного flash-файла окно библиотеки временно пополнится символами из другой анимации (**рис.9**). Используя их, можно создать новый клип (**beer**). У нас это рыбка, выпрыгивающая из пивной кружки по направляющей линии (**рис.10**). Осталось только нарисовать две кнопки для перехода на другие сайты, естественно, с надписями "О рыбках" и "О пиве", и приступать к окончательной сборке.

На первый кадр перетянем символ **Intro**. На второй — **beer** и добавим текстовую надпись (**рис.11**). Третья страница у нас посвящена ссылкам. Переместим кнопки на третий кадр и добавим скрипты открытия URL в новом окне (**рис.12**). А для большего эффекта снабдим сайт еще и звуковым сопровожде-

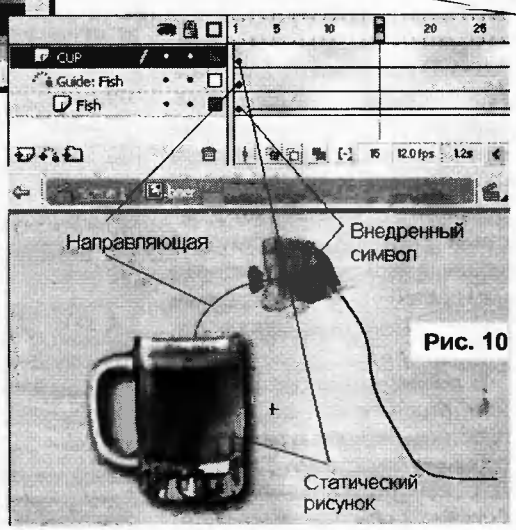


Рис. 10

Рис. 11

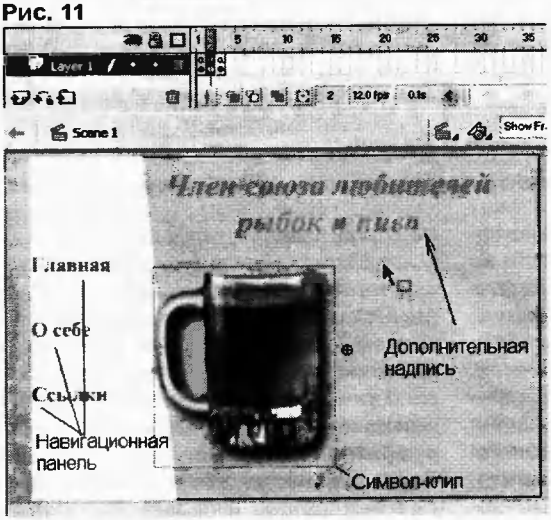


Рис. 12

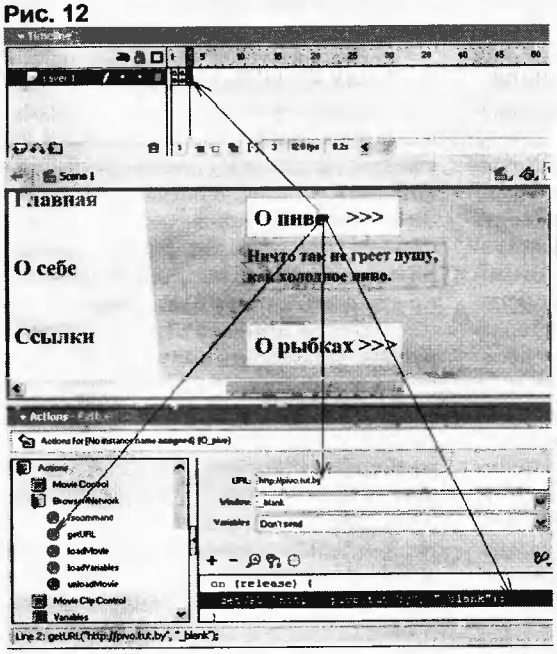


Рис. 13

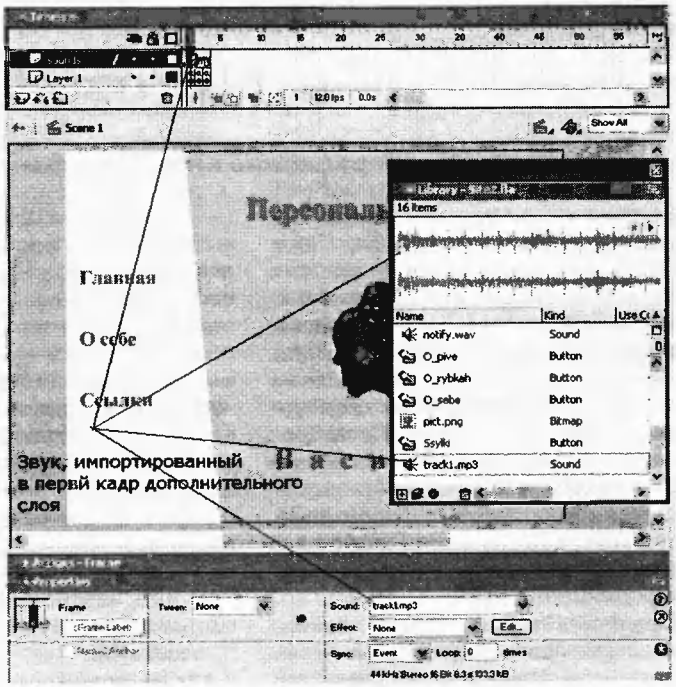
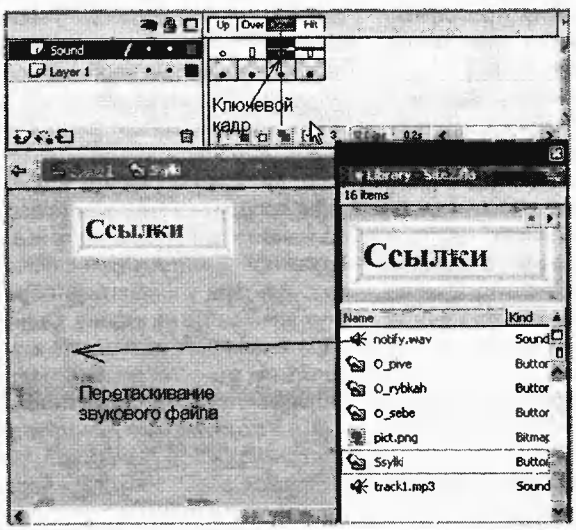
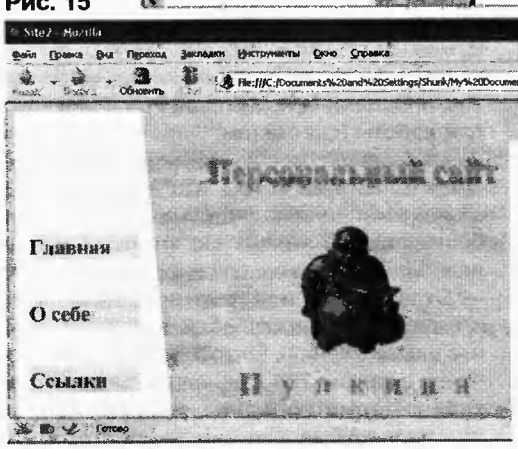


Рис. 14



нием. Для этих целей лучше всего подойдут файлы формата WAV и MP3. После импорта звуковых файлов (**File/Import**), соответствующие значки появляются в библиотеке (**рис.13**). Для звука лучше создать отдельный слой (**Insert/Layer**). Затем выделяется первый кадр дополнительного слоя, но звуковой файл переносится прямо на рабочую область. На кадровой линейке появляется изображение графика звукового сигнала (рис.13). Импортированный файл будет звучать каждый раз при открытии первого кадра анимации. Примерно так же добавляется звук и к кнопке. Если, например, кнопка ссылок должна реагировать на нажатие — двойным щелчком по символу **Ssylki** в библиотеке переходим в режим редактирования, добавляем дополнительный слой, третий кадр (**Down**) делаем ключевым, перетаскиваем значок файла на рабочую сцену (**рис.14**).

Рис. 15



Можно приступать к публикации — **File/Publish**. Файл со звуком получился несколько "тяжеловатым" (у автора — 244 Кб). Но это же сайт целиком! Очевидный плюс — размещение такого файла в Интернет не вызовет особых затруднений, не придется следить за десятками или сотнями файлов, да и вопросы совместимости с различными типами браузеров отпадают. В Mozilla (**рис.15**) он выглядит (да и звучит) точно так же, как и в Internet Explorer. Пути сокращения размеров файла достаточно очевидны — минимум звука, желательно MP3 высокой степени сжатия, да и рисунки лучше бы создавать прямо во Flash. Этого удается достичь не всегда. Чтобы не раздражать доблестных обозревателей Всемирной паутины длительной загрузкой flash-сайта, применяют следующие приемы. Во-первых, внедренные символы можно подгрузить и после загрузки начального кадра. В этом случае используют команды ActionScript вида **loadMovie** и **unloadMovie**. Во-вторых, на время загрузки запускают клипы-предзагрузчики, которые показывают процент загрузки, оставшееся время и т.д. Все это требует уже более основательного владения программированием, и эти вопросы мы оставим за рамками нашего рассмотрения. А тому, где это можно увидеть, как это работает и, если возник интерес, где узнать технические детали, будет посвящена следующая статья.

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

Разработка web-сайтов в Microsoft Office FrontPage 2003

(Продолжение. Начало в №8/2004)

Управление интерфейсом. Основные приемы работы с web-страницами

Сегодня мы продолжаем знакомство с программой Microsoft Office FrontPage 2003 и в этот раз рассмотрим вопросы управления интерфейсом программы, а также самостоятельного создания и оформления сайта. Чтобы приступить к работе, выполните уже знакомый нам

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

Если в конце нашего прошлого занятия перед завершением работы с Microsoft FrontPage вы вначале завершили работу с текущим сайтом по команде **Файл/Закрыть узел**, то сейчас при запуске программы автоматически будет создана новая web-страница с именем **нов_стр_1.htm**. В том случае, если вы просто закрыли программу вместе с текущим сайтом, при запуске Microsoft FrontPage автоматически откроется последний использовавшийся web-сайт, т.е. созданный нами в прошлый раз личный сайт myweb1. Поскольку он нам уже не понадобится, закройте его с помощью команды **Файл/Закрыть узел**.

А сейчас мы приступаем к самостоятельной разработке web-сайта. В качестве примера создадим небольшой персональный сайт о себе. Для начала нам необходимо создать пустой новый сайт, а затем наполнить его нужным количеством страниц и связать их с помощью гиперссылок.

2. Создание нового web-сайта.

Как вы знаете, для создания нового сайта можно воспользоваться командой **Создать страницу или узел...** в области задач **Приступая к работе** или выполнить команду **Файл/Создать...**, после чего в разделе **Создать веб-узел** области задач **Создание** выбрать команду **Другие шаблоны веб-узлов...**. Те, кому удобнее работать с панелями инструментов, могут вместо вышеуказанных действий просто открыть список инструмента панели **Стандартная** и выбрать команду **Веб-узел...**

В прошлый раз мы уже говорили о том, что для создания нового сайта Microsoft FrontPage предлагает использовать шаблоны или мастера. Напомним лишь, что для самостоятельной разработки и оформления web-сайта, как правило, используются шаблоны **Одностраничный веб-узел** или **Пустой веб-узел**. Поэтому в появившемся диалоговом окне **Шаблоны веб-узлов** выберите шаблон **Одностраничный веб-узел**, который позволяет создать сайт, содержащий пустую начальную страницу. А для определения местоположения нового сайта щелкните по кнопке **Обзор...**, откройте свою рабочую папку и, воспользовавшись кнопкой **Создать папку** диалогового окна **Место для нового веб-узла**, создайте корневую папку сайта с именем **myweb2**. С помощью кнопки **Открыть** вернитесь в окно создания нового сайта, убедитесь в правильности указания пути к корневой папке нового сайта и подтвердите создание сайта щелчком по кнопке **ОК**.

Как вы помните, при этом Microsoft FrontPage помещает в корневую папку сайта документ **index.htm** (начальную страницу сайта), папку **images** (для хранения графических изображений) и папку **_private** (невидимую для браузеров, для хранения любых файлов, которые необходимо держать скрытыми от посетителей сайта после его публикации).

На прошлом занятии мы рассмотрели основные элементы интерфейса Microsoft FrontPage 2003. А сейчас познакомимся с тем, как управлять отображением этих элементов на экране. Однако предварительно двойным щелчком мыши откройте в режиме редактирования начальную страницу сайта **index.htm**.

3. Управление интерфейсом Microsoft Office FrontPage 2003

Прежде всего обсудим вопрос вывода на экран панелей инструментов. Выполнив команду **Вид/Панели инструментов**, вы можете ознакомиться с полным перечнем панелей инструментов, предлагаемых Microsoft FrontPage для работы над сайтом. Флажки, установленные рядом с именами панелей, свидетельствуют о том, что в данный момент эти панели инструментов отображаются на экране. По умолчанию используются две панели инструментов — **Стандартная** и **Форматирование**.

Наверное, вы обращали внимание, что в приложениях Microsoft Office, начиная с версии 2000, панели **Стандартная** и **Форматирование** могут располагаться друг за другом в одной строке. При этом может оказаться, что для отображения всех кнопок этих панелей инструментов просто не хватает места. Поэтому зачастую удобнее, когда эти панели расположены в две строки. Кроме того, при выборе любого пункта меню, как правило, отображается лишь сокращенная версия меню, содержащая только часто используемые команды. Согласитесь, что это тоже далеко не всегда удобно. Чтобы настроить отображение в меню всех команд, а также расположение панелей инструментов, выполните команду **Вид/Панели инструментов/Настройка...** и на вкладке **Параметры** установите флажки **Стандартная панель** и **панель форматирования в две строки**, **Всегда показывать полные меню**.

Как вы помните, в самом низу рабочего окна расположена строка состояния с информацией об ожидаемом времени загрузки текущего документа из Интернета при заданной скорости соединения и текущий размер страницы. За отображение на экране строки состояния отвечает команда **Сервис/Параметры.../вкладка Общие/флажок показывать строку состояния**. В том же диалоговом окне в разделе **При запуске** можно задать вывод на экран области задач **Приступая к работе** при запуске Microsoft FrontPage, а также автоматическое открывание последнего web-сайта, использовавшегося в Microsoft FrontPage.

При работе над web-страницей очень удобно использовать линейки. Чтобы включить отображение линеек, необходимо выполнить команду **Вид/Линейка и сетка/Показать линейку** (выключение отображения линеек производится с помощью той же команды). Заметьте, что в качестве единиц измерения линеек по умолчанию используются пикселы, а положение курсора мыши показывается на линейках с помощью линий. Таким образом, использование линеек помогает расположить объекты на странице. Еще одним средством точного позиционирования отдельных элементов web-страницы является использование координатной сетки. Включение и выключение отображения сетки производится по команде **Вид/Линейка и сетка/Показать сетку**. По умолчанию линии сетки прорисовываются через интервал в 50 пикселов. Определить единицы измерения для линеек, а также изменить шаг координатной сетки можно с помощью команды **Вид/Линейка и сетка/Настроить...** В качестве единиц измерения вы можете установить точки (пикселы), дюймы, сантиметры и пункты. Понятно, что в области сайтостроения удобнее манипулировать именно

пикселами. В том же диалоговом окне настройки линеек и сетки можно установить шаг координатной сетки (поле **Интервал** в разделе **Отображение сетки**), а также стиль и цвет линий сетки (поля **Тип линии** и **Цвет линии**). Итак, мы уже говорили о том, что линейки позволяют более точно позиционировать объекты на странице или же определить точные размеры объекта. В последнем случае, когда стоит задача определения точных размеров элементов web-страницы (например, ячеек таблицы), чтобы не производить расчеты с использованием линейки, удобно просто переместить начало координат (к примеру, в левый верхний угол объекта). Для этого достаточно установить курсор мыши на место пересечения линеек (в начало отсчета) и, нажав левую кнопку мыши, переместить начало координат в нужную точку (например, в точку с координатами 50, 50). При этом шкалы на линейках изменятся, и координаты начнут отсчитываться от новой точки отсчета. Чтобы вернуть начало координат в исходное положение, достаточно выполнить двойной щелчок мышью на месте пересечения линеек или же выполнить команду **Вид/Линейка и сетка/Отменить исходные параметры**.

На прошлом занятии мы уже упоминали о том, что в Microsoft FrontPage 2003 упростился доступ к генерируемому программой HTML-коду благодаря появлению нового представления **С разделением**, сочетающего визуальное проектирование web-страницы с одновременным доступом к HTML-коду. Помимо этого, для упрощения работы с тегами языка HTML в любом представлении документа под ярлыком с именем редактируемого документа появилась новая панель тегов. Эта панель похожа на обычную панель инструментов, только кнопки этой панели соответствуют используемым в документе тегам и расположены слева направо в порядке вложенности. Благодаря этой панели можно быстро выделить нужный объект на странице (для этого достаточно выполнить щелчок мышью по тегу, описывающему этот объект), а воспользовавшись кнопкой раскрытия списка, можно изменить тег либо настроить его свойства. Включение/выключение отображения этой удобной панели тегов производится с помощью команды **Вид/Быстрый выбор тега**.

Вот и все, что касается настройки основных элементов интерфейса. А сейчас займемся оформлением начальной страницы нашего сайта. Перед размещением на странице требуемых объектов желательно вначале произвести настройку общих параметров документа. Подобная настройка параметров web-страницы включает определение заголовка web-страницы, возможности использования фонового звука, определение цветового оформления страницы (установка цветов фона, текста, гиперссылок, а также определение возможности использования фонового рисунка), а также указание языка документа и кодировки, используемой при сохранении web-страницы.

4. Настройка параметров web-страницы

Подобная настройка осуществляется в режиме редактирования web-страницы с помощью диалогового окна **Свойства страницы**. Для вызова этого окна воспользуйтесь командой **Файл/Свойства...** или же выполните щелчок правой кнопкой мыши в любом месте на странице и в появившемся контекстном меню выберите **Свойства страницы...**

Прежде всего определим заголовок нашей web-страницы, для чего на вкладке **Общие** в поле **Название** введите: **Фамилия Имя Отчество (рис.4)**. Этот текст будет отображаться при просмотре документа в браузере в строке заголовка программы. Здесь же в разделе **Фоновый звук** с помощью кнопки **Обзор...** вы можете выбрать звуковой файл, который будет воспроизводиться при открытии документа в браузере. При этом в поле **Число повторов** необходимо указать, сколько раз должен проигрываться звуковой файл, или же определить для него непрерывное звучание (флажок **Непрерывно**).

Перейдите на вкладку **Форматирование (рис.5)**. Здесь находятся параметры определения цветового оформления страницы. Как уже упоминалось выше, в качестве фона web-страницы может использоваться обычная цветная заливка или же графическое изображение. В последнем случае для определения фонового рисунка в разделе **Фон** необходимо установить флажок **Фоновый рисунок** и с помощью кнопки **Обзор...** выбрать файл рисунка, который требуется использовать в качестве фонового. При этом, если рисунок по ширине или высоте меньше размеров web-страницы, он повторяется по всему фону страницы. Кроме того, при просмотре подобного документа в браузере, в случае использования полос прокрутки, фоновый рисунок прокручивается вместе с текстом страницы. Чтобы рисунок оставался неподвижным при прокручивании текста страницы, установите флажок **Сделать подложкой**. Здесь же в разделе **Цвета** можно задать цветовое оформление для основных элементов web-страницы — основного текста и гиперссылок. В поле **Текст** установите подходящий цвет для оформления текста документа. Поля **Гиперссылка**, **Просмотренная ссылка** и **Активная ссылка** (та, по которой пользователь выполняет щелчок) оставьте без изменений, с тем чтобы использовались стандартные цвета для ссылок (как правило, синий, фиолетовый и красный). Даже в случае использования фонового рисунка, желательно в разделе **Цвета** в

Рис. 4

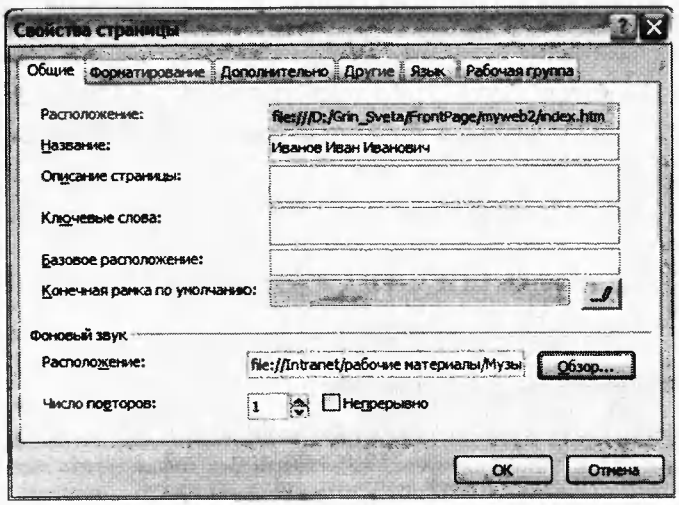
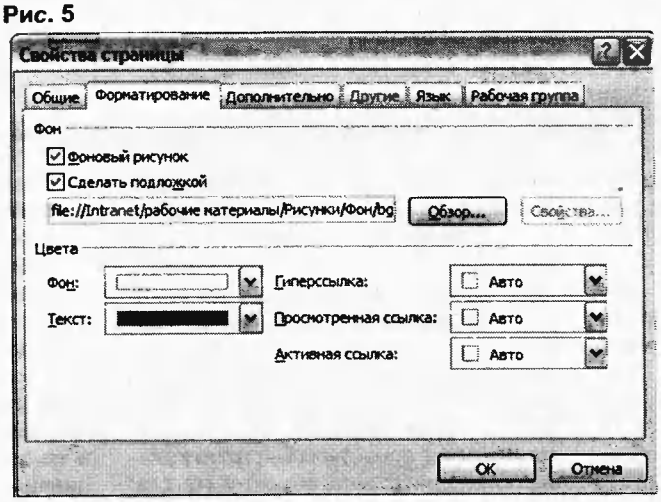
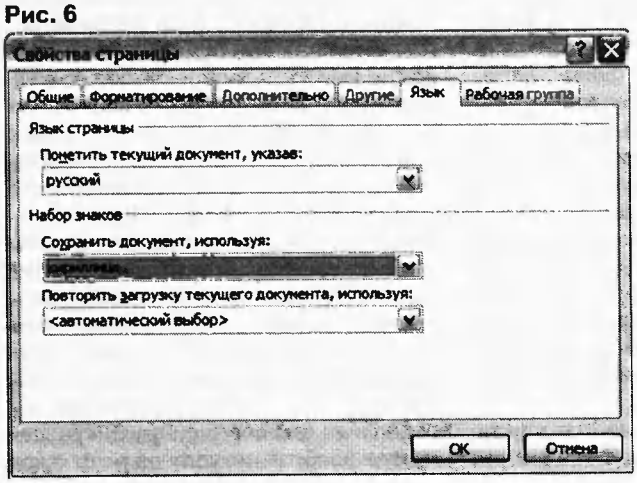


Рис. 5



поле **Фон** выбрать цвет фона, наиболее близкий к цвету фонового изображения. В том случае, если пользователь при просмотре вашей страницы в браузере откажется от загрузки рисунков, вместо фонового изображения будет использован указанный цвет фона. Подобная предосторожность позволит вам добиться сходного отображения вашей страницы в режиме включения и отключения загрузки рисунков.

Рис. 6



И в заключение перейдите на вкладку **Язык (рис.6)**. Здесь необходимо определить язык текста документа и кодировку, используемую при сохранении web-страницы. Для этого в разделе **Язык страницы** в поле **Пометить текущий документ, указав** установите **русский** (выбранный язык будет использоваться и при проверке орфографии на стрвнице) и в разделе **Набор знаков** в поле **Сохранить документ, используя** установите **кириллица** (информация о кодировке web-стрвницы позволяет браузерам правильно интерпретировать текст на странице). Подтвердите изменение общих свойств документа, выполнив щелчок по кнопке **ОК**. Вот сейчас можно приступать к наполнению начальной страницы сайта.

5. Ввод и форматирование текста. Создание структуры документа с помощью заголоеков. Вставка горизонтальных линий

Введите на начальную страницу сайта текст, приведенный на **рис.7**. Если окажется, что цвет текста был подобран не совсем удачно, еще раз выполните щелчок правой кнопкой мыши в любом месте на странице, выберите **Свойства страницы...** и на вкладке **Форматирование** изменитв цвет текста.

Рассмотрим, каким образом на web-страницу можно вставить специальные символы. Продемонстрируем это на примере знака авторского права. Установите текстовый курсор после слова **Copyright** и выполните команду **Вставка/Символ...** В появившвмся диалоговом окне выберите символ авторского права ©, щелкните по кнопке **Вставить** и закройте окно вставки символов. Заметьте, что специальные символы вставляются на web-страницу так же, как и в случае документа Microsoft Word.

Форматирование текста web-страницы выполняется обычным способом. Достаточно выделить нужный фрагмент текста, а затем, воспользовавшись инструментами панели **Форматирование**, оформить выделенный фрагмент. Правда, для оформления текста web-страницы желательно использовать не прямое форматирование, а стили. Под **стилем** понимают набор параметров форматирования. Стили позволяют с помощью одного действия применить сразу целую группу атрибутов форматирования. Таким образом, стили являются средством быстрого изменения внешнего вида объектов (текста, таблиц, списков). Кроме того, на web-страницах стили выполняют еще одну немаловажную функцию: стили, использованные для оформления заголовков различных уровней, определяют логическую структуру текста. Воспользуемся стилями для оформления заголовка страницы. Выделите заголовок **Личный сайт Фамилия Имя Отчество**, откройте список инструмента **Стиль** Обычный на панели **Форматирование** и выберите **Заголовок 1**. Дополнительно с помощью инструментов и панели **Форматирование** задайте выравнивание заголовка по центру и измените цвет заголовка web-страницы. Абзац с информацией об авторских правах также выровняйте по центру.

Чтобы отделить заголовок web-страницы от основного текста, вставим после заголовка горизонтальную линию. Для этого установите текстовый курсор в конце заголовка и выполните команду **Вставка/Горизонтальная линия**. Для изменения параметров горизонтальной линии щелчком мыши выделите линию и выполните команду **Формат/Свойства** (более простой вариант вызова диалогового окна свойств любого объекта — использование контекстного меню, вызываемого щелчком правой кнопкой мыши по объекту). Для горизонтальной линии можно настроить размер (причем ширина линии может задаваться либо точно в пикселах (точках), либо в процентах от ширины окна, в котором отображается страница), выравнивание линии на странице и ее цвет (**рис.8**). Определим размер линии — в разделе **Размер** проверьте

Рис. 7

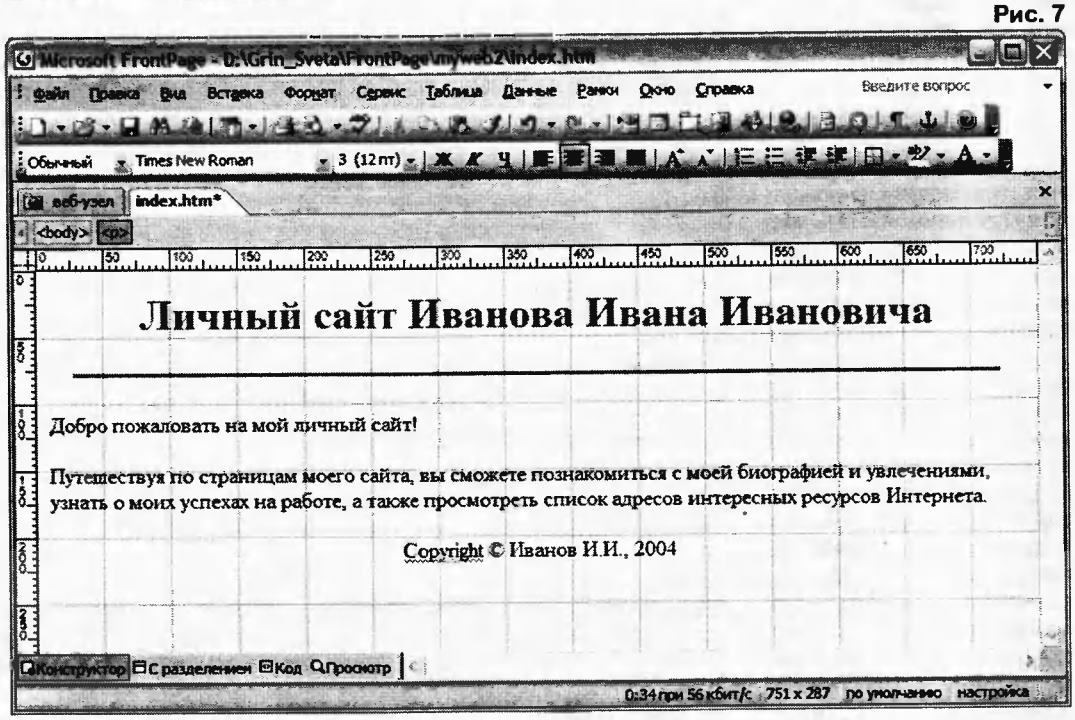
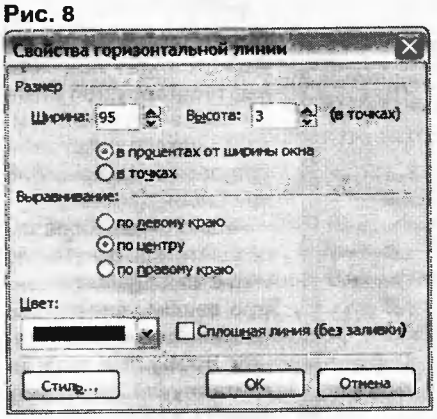


Рис. 8



наличие переключателя **в процентах от ширины окна**, после чего в поле **Ширина** установите **95**, в качестве высоты линии установите **3** пиксела. Задайте выравнивание линии по центру страницы и определите цвет линии. Не забудьте подтвердить применение выбранных параметров с помощью кнопки **ОК**.

Обратите внимание, что после наших манипуляций на ярлыке рядом с именем редактируемого документа появился символ *. Наличие этого символа означает, что произведенные изменения еще не были сохранены. Самое время позаботиться о сохранении результатов нашей работы.

6. Сохранение web-страницы. Сохранение внедренных файлов

Сохраним начальную страницу нашего сайта, выполнив команду **Файл/Сохранить** или воспользовавшись инструментом панели **Стандартная**. Выполняя эту операцию, Microsoft FrontPage проверяет, не содержит ли сохраняемый документ внедренных объектов (в прошлый раз мы уже говорили о том, что все мультимедийные объекты, представленные на web-странице, хранятся в виде отдельных файлов). В случае положительного ответа на этот вопрос Microsoft FrontPage выводит на экран диалоговое окно **Сохранение внедренных файлов**, в котором предлагает сохранить найденные внедренные объекты в папке сайта (что достаточно удобно, т.к. избавляет разработчика от необходимости предварительного копирования всех необходимых мультимедийных файлов в папку сайта). Поэтому, если при настройке общих параметров документа вы определили фоновый звук и/или рисунок для web-страницы, в появившемся диалоговом окне **Сохранение внедренных файлов** будут перечислены используемые файлы (**рис.9**). В этом диалоговом окне вы должны проследить за тем, чтобы в именах всех файлов использовались только строчные латинские буквы и цифры (при необходимости вы можете произвести переименование мультимедийных файлов, выделив нужный файл и воспользовавшись кнопкой **Переименовать**), а все графические изображения будут помещены в папку **images** вашего сайта. Чтобы фоновый рисунок был сохранен в нужной папке, выделите его имя в списке и, воспользовавшись кнопкой **Сменить папку...**, щелчком мыши выберите папку **images**. Подтвердите свой выбор щелчком по кнопке **ОК** и убедитесь в том, что в диалоговом окне **Сохранение внедренных файлов** напротив имени фонового рисунка появилось название папки images. Фоновый звук сохраните прямо в корневой папке сайта. Не забудьте подтвердить сохранение внедренных файлов щелчком по кнопке **ОК**.

В прошлый раз мы уже упоминали о том, что при разработке web-страницы следует как можно чаще тестировать ее внешний вид и работоспособность всех ее элементов с помощью программы-браузера. Давайте же посмотрим, в каком виде наша страница предстанет перед посетителями сайта.

7. Просмотр web-страницы в браузере. Завершение работы с web-страницей

Как вы помните, для ускорения просмотра отдельной страницы можно воспользоваться представлением **Просмотр**. Однако для более корректного тестирования документа все же желательно web-страницу просматривать непосредственно программой-браузером. Для этого выполните команду **Файл/Просмотреть в обозревателе** или воспользуйтесь инструментом панели **Стандартная**. При этом открывается список доступных браузеров с разными размерами окон. Просмотрите домашнюю страницу сайта в браузере Microsoft Internet Explorer, используя текущий размер окна.

Если для оформления начальной страницы сайта вы использовали фоновый рисунок, проконтролируйте внешний вид страницы в режиме отключения графики. Для этого отключите загрузку рисунков (**Сервис/Свойства обозревателя.../вкладка Дополнительно/в блоке Мультимедиа** снимите флажок **Отображать рисунки/ОК**) и обновите текущую страницу (кнопка **Обновить** на панели инструментов). Вот сейчас и пригодится определенный нами в свойствах страницы цвет фона. Конечно, желательно, чтобы страница выглядела практически одинаково в режиме включения и отключения загрузки рисунков.

Кроме того, если для фонового рисунка вы дополнительно установили параметр **Сделать подложкой**, проверьте работоспособность этого параметра. Для этого включите загрузку рисунков и еще раз обновите текущую страницу. Затем уменьшите размеры рабочего окна Microsoft Internet Explorer так, чтобы появилась вертикальная полоса прокрутки. Воспользуйтесь полосой прокрутки и обратите внимание, что фоновый рисунок не прокручивается вместе с текстом страницы. Не забудьте после этого завершить работу с браузером и вернуться в Microsoft FrontPage.

На этом работу с начальной страницей сайта мы завершаем. Закройте ее с помощью команды **Файл/Закрыть** или воспользовавшись кнопкой в заголовке окна документа.

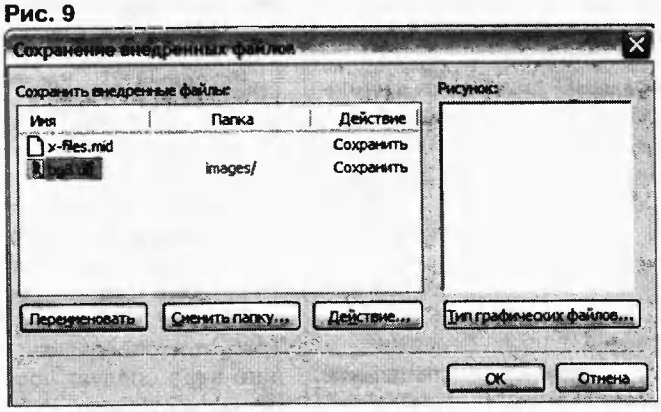
8. Создание новых web-страниц

Нам осталось добавить в наш сайт еще несколько страниц, оформив их точно по такому же образцу, как и начальную страницу сайта.

Для создания новой web-страницы можно воспользоваться командой **Файл/Создать...**, после чего в разделе **Создать страницу** области задач **Создание** выбрать команду **Другие шаблоны страниц...** Или же отройте список инструмента панели **Стандартная** и выберите команду **Страница...**

И в том, и в другом случае на экране появится диалоговое окно **Шаблоны страниц (рис.10)**. Как и при создании сайта, для создания отдельных страниц Microsoft FrontPage также предлагает использовать шаблоны. Заметьте, что в диалоговом окне создания нового web-документа представлены три вкладки, позволяющие реализовать создание документов различного типа: на вкладке **Общие** расположены шаблоны для создания обычных web-страниц, на вкладке **Страница рамок** — для создания страниц, содержащих фреймы, а вкладка **Таблицы стилей** содержит шаблоны для создания внешних таблиц стилей (CSS-файлов). О фреймах и таблицах

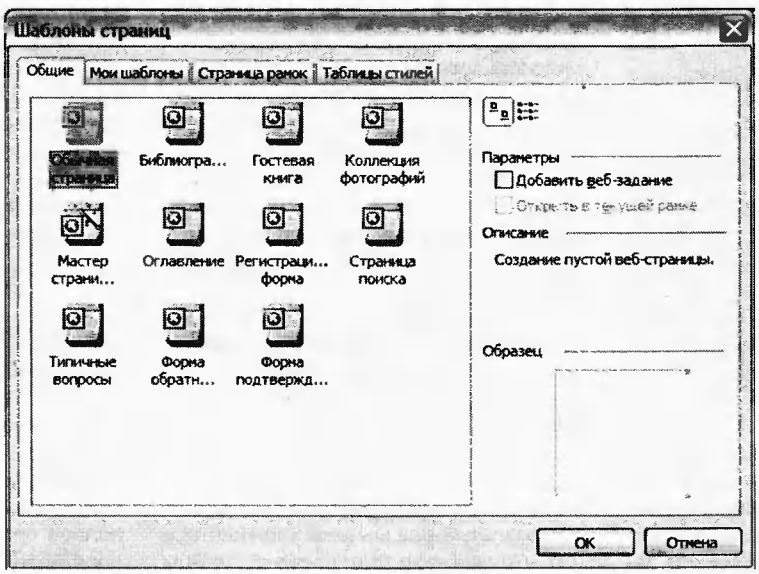
Рис. 9



стилей мы поговорим позже, а сейчас обсудим вопросы создания обычных web-страниц.

Шаблон web-страницы определяет макет и основную структуру документа. Поэтому использование шаблонов значительно упрощает задачу подготовки нового web-документа, т.к. при этом создается готовая web-страница, содержащая примерный текст и элементы оформления. Для создания пустой web-страницы используется шаблон **Обычная страница**. Этот шаблон не содержит никаких элементов оформления и позволяет создавать web-страницу "с нуля". Воспользуемся этим шаблоном для создания второй страницы нашего сайта.

Рис. 10



Посвятим эту страницу рассказу об основных вехах своей жизни. Прежде всего настройте общие параметры новой страницы, определив для нее в качестве названия текст **Фамилия Имя Отчество | Биография**. Задайте цветовое оформление новой страницы в соответствии с оформлением начальной страницы сайта — определите такой же фоновый рисунок и те же цвета фона и текста, как и для начальной страницы, при этом не забудьте, что теперь фоновый рисунок хранится в папке **images** нашего сайта **myweb2**. Определите в качестве языка документа русский и кириллицу в качестве кодировки, используемой при сохранении web-страницы (см. п.4).

Далее на web-страницу поместите заголовок **Моя биография**. Примените к заголовку web-страницы стиль **Заголовок 1**, выровняйте его по центру, измените цвет заголовка. Под заголовком вставьте горизонтальную линию и настройте ее параметры (см. п.5). Под горизонтальной линией введите краткие сведения о своей жизни.

В заключение не забудьте сохранить новую web-страницу под именем **biography.htm** в папке сайта **myweb2** (см. п.6.), протестировать ее в браузере и завершить с ней работу (см. п.7).

Аналогичным образом создайте еще одну новую web-страницу, посвященную вашим увлечениям. Сохраните ее под именем **hobby.htm** в папке сайта **myweb2** и протестируйте в браузере.

В результате описанных действий наш персональный сайт должен содержать три web-страницы:

- **index.htm** — начальная страница сайта;
- **biography.htm** — страница с краткими биографическими сведениями;
- **hobby.htm** — страница, содержащая информацию об увлечениях.

А вот о том, как эти страницы связать с помощью гиперссылок и дополнительно оформить, мы поговорим в следующий раз. На этом наше сегодняшнее занятие мы заканчиваем. Не забудьте выполнить ***9. 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.

В предлагаемой и следующей статьях будут рассмотрены простейшие приемы решения задач линейного программирования и связь системы MATLAB с таблицами Excel на примере таких задач.

Краткие сведения и примеры

Среди задач оптимизации наиболее широко применяются и лучше всего изучены задачи линейного программирования [1, 2]. Они используются при планировании производства, распределении ресурсов, выборе технологий, организации работы транспорта и т.д.

Задача линейного программирования заключается в нахождении таких значений параметров $x_1, x_2, \ldots, x_n$, которые обеспечивают минимум (или максимум) *линейной* целевой функции

$$S = c_1x_1 + c_2x_2 + \ldots + c_nx_n \qquad (1)$$

при условиях

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &<=> b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &<=> b_2 \\ \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots & \\ a_{k1}x_1 + a_{k2}x_2 + \ldots + a_{kn}x_n &<=> b_k, \end{aligned} \qquad (2)$$

где: c_i, a_{ij}, b_i $(i = 1,\ldots k;\ j = 1, 2,\ldots n)$ — заданные действительные числа; символ "<=>" — любое из соотношений "<", "≤", "=", "≥", ">".

Как видно, целевая функция (1) и ограничения (2) линейны относительно переменных $x_1, x_2,..x_n$.

Пример 1

Молочный завод выпускает два вида молочных продуктов — 1 и 2, для чего используются два вида исходного сырья — А и В. Суточные запасы этих видов сырья составляют, соответственно, 6 и 8 т. Расходы А и В на 1 т соответствующих продуктов приведены в **табл.1**.

Табл. 1

Исходный вид сырья	Расход исходных видов сырья на тонну продуктов, т		Запас сырья, т
	Продукт (1)	Продукт (2)	
A	1	2	6
B	2	1	8

Изучение рынка сбыта показало, что суточный спрос на продукт 2 не превышает спроса на продукт 1 более чем на 1 т и, кроме того, не превышает 2 т. Прибыль от продукта 1 составляет 3000 ден.ед./т, от продукта 2 — 2000 ден.ед./т. Требуется определить, какое количество продуктов каждого вида следует производить, чтобы получить максимальную прибыль.

Построение математической модели задачи

Обозначим через x_1 и x_2 (т) суточный объем производства продуктов 1 и 2 соответственно.

Целевой функцией S является прибыль:

$$S = (3x_1 + 2x_2)1000 \rightarrow max \quad (3)$$

(Запись $f \rightarrow max$ означает, что ищется максимум функции f).

Запишем условие задачи в виде системы ограничений:

1) ограничение по сырью А:

$$x_1 + 2x_2 \le 6; \quad (4)$$

2) ограничение по сырью В:

$$2x_1 + x_2 \le 8; \quad (5)$$

3) ограничения по спросу:

$$x_2 - x_1 \le 1, \quad (6)$$

$$x_2 \le 2. \quad (7)$$

Очевидно, что должно также выполняться условие неотрицательности параметров:

$$x_1 \ge 0;\ x_2 \ge 0. \quad (8)$$

Как видно, рассматриваемая математическая модель записывается в форме общей задачи линейного программирования: найти значения оптимизируемых параметров x_1 и x_2, обеспечивающих максимум целевой функции (3) при ограничениях (4)...(7), а также при ограничениях на неотрицательность параметров (8).

Решение задачи в системе MATLAB

В прикладном пакете **Optimization Toolbox** системы MATLAB имеется процедура **linprog** для решения задач линейного программирования [3, 4]. Ее описание на русском языке приведено в архиве новостей за 20.12.02 [4]. Процедура имеет много вариантов применения и может использовать различные алгоритмы.

Процедура решает задачу:

$$\min_x S(x)$$

при ограничениях:

$$A \cdot x \le b,$$

$$Aeq \cdot x = beq,$$

$$lb \le x \le ub,$$

где S, x, b, beq, lb и ub — векторы, а A и Aeq — матрицы.

Наиболее простая форма обращения к процедуре:

x = linprog(S,A,b,Aeq,beq).

Для наших примеров достаточно формы:

[x,Sval] = linprog(S,A,b,Aeq,beq,lb).

Здесь:

- **Sval** — значение целевой функции;
- **x** — вектор решений;
- **S** — вектор коэффициентов целевой функции;
- **A** — матрица коэффициентов неравенств;
- **b** — вектор правых частей неравенств;
- **Aeq** — матрица коэффициентов равенств;
- **beq** — вектор правых частей равенств;
- **lb** — вектор левых границ значений переменных. В нашем случае это нули, соответствующие условию (8).

Если необходимо найти максимум целевой функции, то ее коэффициенты следует взять с противоположным знаком.

Если в ограничениях неравенствах стоят знаки "?", то знаки левых и правых частей неравенств следует взять с противоположным знаком.

Рассмотренный пример может быть решен с помощью следующих команд:

```
% Линейное программирование
% Целевая функция
%  S = (3*x(1) + 2*x(2))*1000 -> max
S = [-3; -2];
% Ограничения
%  x(1) + 2*x(2) <= 6
%  2*x(1) + x(2) <= 8
%  x(2) - x(1) <= 1
%  x(2) <= 2
%  x(1) >= 0; x(2) >= 0
A =  [1 2
      2 1
     -1 1
      0 1];
b = [6; 8; 1; 2];
lb = [0; 0];
%
% Ограничений в виде равенств нет, поэтому
% вместо значений Aeq и beq вводим пустые массивы
[x,Sv] = linprog(S,A,b,[],[],lb)
```

Результат решения:

```
Optimization terminated successfully.
x =
    3.3333
    1.3333
Sval =
  -12.6667
```

Таким образом, первый продукт должен производиться в объеме 3,3333 т, а второй — 1,3333 т. При этом прибыль составит 12,6667 тыс.ден.ед. и будет максимальной.

Пример 2

К задачам линейного программирования относится и так называемая транспортная задача. Рассмотрим один из простейших классических вариантов этой задачи.

На предприятиях П1, П2 производят одинаковую продукцию. Необходимо доставить эту продукцию заказчикам 31, 32, 33. Известна стоимость перевозки продукции от каждого предприятия каждому заказчику. Необходимо составить план перевозок таким образом, чтобы стоимость перевозок была минимальной. Пока рассмотрим случай, когда общий объем производства равен общему объему заказов (закрытая модель). Исходные данные представлены в **табл.2**.

Табл. 2

Стоимость перевозок				Объем производства
Предприятие\Заказчик	**31**	**32**	**33**	
П1	11,5	7,3	12	237
П2	6,2	10,7	9,1	278
Количество заказов	145	210	160	

Табл. 3

Предприятие\Заказчик	**31**	**32**	**33**
П1	x_1	x_3	x_5
П2	x_2	x_4	x_6

Обозначим количество перевозок через x_1, x_2, ..., x_6 в соответствии с **табл.3**.

Общая стоимость перевозок (целевая функция) равна:

$$S = 11.5x_1 + 6.2x_2 + 7.3x_3 + 10.7x_4 + 12x_5 + 9.1x_6.$$

Так как по условию общий объем производства равен общему объему заказов, то должны быть выполнены следующие равенства (ограничения):

- по производству продукции (строки таблицы)

$$x_1 + x_3 + x_5 = 237,$$
$$x_2 + x_4 + x_6 = 278,$$

$$\text{матрица коэффициентов: } \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 1 \end{pmatrix}; \quad (9)$$

- по заказам (столбцы таблицы)

$$x_1 + x_2 = 145,$$
$$x_3 + x_4 = 210,$$
$$x_5 + x_6 = 100,$$

$$\text{матрица коэффициентов: } \begin{pmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \end{pmatrix}. \quad (10)$$

Количество перевезенной продукции не может быть отрицательным, поэтому введем ограничение на нижнюю границу переменных $x_1, x_2, \ldots, x_6$, равную нулю.

Решение задачи в MATLAB:

```
% Ввод матриц S, Aeq, beq, lb
S = [11.5;6.2;7.3;10.7;12;9.1];
Aeq = [1 0 1 0 1 0
       0 1 0 1 0 1
       1 1 0 0 0 0
       0 0 1 1 0 0
       0 0 0 0 1 1];
beq = [237;278;145;210;160];
lb = zeros(6,1);
% Ограничений в виде неравенств нет, поэтому
% вместо значений A и b вводим пустые массивы
[x,Sval] = linprog(S,[],[],Aeq,beq,lb)
```

Результаты решения:

```
Optimization terminated successfully.
x =
    0.0000
  145.0000
  210.0000
    0.0000
   27.0000
  133.0000
Sval =
  3.9663e+003
```

В результате получили значения количества продукции, перевозимой от каждого из предприятий каждому заказчику, при этом общая стоимость перевозок составит 3966,3 ден.ед.

Нетрудно заметить закономерности (см. матрицы (9), (10)) в построении матрицы **Aeq**, а следовательно, и составить процедуру для решения задачи с любой размерностью табл.1. Более того, легко учесть случаи, когда общий объем производства не равен общему объему заказов (открытая задача). Здесь возможны два варианта: объем производства превышает объем заказов; объем заказов превышает объем производства. В этих вариантах просто не нужно объединять матрицы (9), (10) в одну.

Исходные данные будем задаааать в виде матрицы T, которая соответствует таблице исходных данных:

```
T = [11.5  7.3   12   237
      6.2  10.7  9.1  278
      145  210  160    0]
```

Последний 0 в матрице добавлен, чтобы сохранить размерность.

Для решения данной задачи создадим М-файл, который может иметь вид:

```
% Сохранение матрицы T
C = T;
% Определяем размер матрицы C
[n m] = size(C);
% Размер матрицы стоимостей перевозок
n = n-1; m = m-1;
% Выделяеи векторы производства и заказов
rowsum = C(1:n,m+1); colsum = C(n+1,1:m)';
% Находим количество переменных
k  = n*m; lb = zeros(k,1);
% Выделяеи матрицу стоимостей перевозок
% и преобразуем ее в вектор-столбец всех цея
C = C(1:n,1:m); C = C(:)';
% Формируем матрицу коэффициентов ограничений по строкан
arow = [];
for i = 1:m
  % Объединяются единичные матрицы, создаваемые функцией eye
  arow = [arow eye(n)];
end
% Формируем матрицу коэффициентов ограничений по столбцам
acol = [];
j = 0;
for i = 1:m
  j = j+1;
  % Объединяются матрицы, полученные как нулевая матрица,
  % в которой j-я строка заменяется единицами
  v = zeros(m,n); v(j,:) = 1; acol = [acol v];
end
% Находим суммы производства и заказов
Sr = sum(rowsum); Sc = sum(colsum);
% В зависимости от соотношения этих сумм
% (больше, меньше или равны) решаем задачу
if Sr > Sc
  [x, Sval] =  linprog(C,arow,rowsum,acol,colsum,lb);
elseif Sr == Sc
  [x,Sval] = linprog(C,[],[],[arow;acol],[rowsum;colsum],lb);
else
  [x, Sval] = linprog(C,acol,colsum,arow,rowsum,lb);
end
% Для удобства преобразуем вектор решений в матрицу,
% соответствующую исходной матрице
x = reshape(x,n,m)
% Выводим общую стоимость перевозок
Sval
```

Здесь встроенная функция **reshape(x,n,m)** возвращает матрицу размерностью n×m, сформированную из **x** путем последовательной выборки по столбцам.

Сохраним полученный М-файл, например, под именем **trzad.m**. Теперь для решения подобной транспортной задачи любой размерности достаточно ввести матрицу **T** исходных данных и вызывать созданный М-файл. Для нашего примера это будет:

```
>> T = [11.5 7.3 12 237
        6.2 10.7 9.1 278
        145 210 160 0];
>>trzad
```

В результате получим:

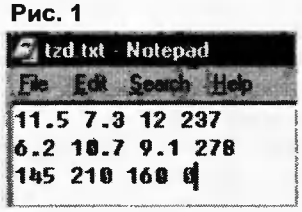
```
Optimization terminated successfully.
x =
    0.0000  210.0000   27.0000
  145.0000    0.0000  133.0000
Sval =
  3.9663e+003
```

При большой размерности матрицы ее ввод в командной строке представляет определенные неудобства. MATLAB предоставляет возможность ввода и вывода матриц в текстовые файлы при помощи двух следующих функций:

- **T = dlmread('имя файла','разделитель')** — чтение данных из текстового файла в переменную **T;**
- **dlmwrite('имя файла', имя переменной, 'разделитель ')** — запись данных из переменной в текстовый файл.

В качестве разделителя можно использовать пробел, запятую (принимается по умолчанию) и т.д. Значение разделителя \t указывает, что численные значения в строке разделяются при помощи символа табуляции. Текстовые файлы можно создавать, редактировать и распечатывать при помощи как собственного редактора MATLAB, так и других текстовых редакторов (Блокнот, WordPad и т.п.). Например, сформируем матрицу исходных данных последнего примера с помощью стандартной программы Windows — **Блокнот (рис.1)** и сохраним под именем **tzd.txt** в папке, доступной для MATLAB.

Рис. 1



В командном окне наберем:

```
>> D = dlmread('tzd.txt',' ')
```

Получим:

```
D =
11.5000    7.3000   12.0000  237.0000
  6.2000   10.7000    9.1000  278.0000
  145.0000  210.0000  160.0000         0
```

(Окончание следует)

С.РЮМИК,
г.Чернигов.

Интернет-калейдоскоп, freeware #10

Таинственные вещи еще нельзя называть чудесами.
Иоганн Вольфганг Гете

Испокон веков тайны окружают человечество. И если в древние времена загадкой номер один были гром и молния, то сейчас пытливые умы стараются узнать, есть ли во Вселенной братья по разуму, и можно ли построить машину времени.

Вокруг сверхъестественного, необычного и необъяснимого, как правило, много шарлатанства. Отделить "зерна от плевел", читая отрывочные журнальные статьи или заангажированные уфологические газеты, нелегко. А ведь подсказки рядом — в Интернете. В частности, на сайте http://www.skeptik.net/ собраны разгадки и научные объяснения многих необычных фактов — от "изгибания взглядом металлических ложек" и "угадывания мыслей на расстоянии" до "филиппинских хилеров" и "парящих в воздухе карандашей".

Разумеется, не все явления современная наука пока может объяснить, но это дело времени. Как шутят медики, "абсолютно здоровых людей не бывает, есть не полностью обследованные".

Об огромных резервах человеческого организма, феноменальных рекордах памяти можно прочитать на сайте центра "Идея" http://www.mtu-net.ru/gomy/. Руководит центром знаменитый Юрий Горный, чьи показательные выступления (психологические этюды) вот уже более 30 лет шокируют публику. Однако разгадку своих способностей он видит не в "эзотерической энергии, получаемой из Космоса", а в физиологии нервной системы человека.

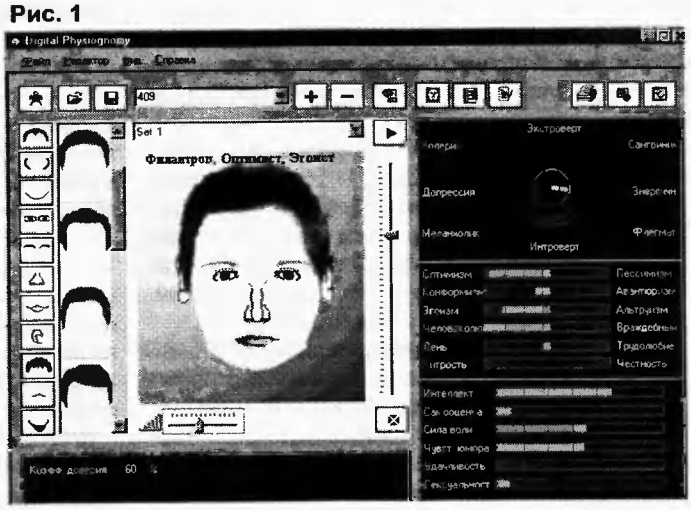
Кстати, всех "экстрасенсов" и "телепатов", желающих показать обществу какой-либо сверхъестественный факт, ожидает приз в 1 млн. американских долларов (http://www.skeptik.net/possib/million.htm). Именно такую сумму учредил Образовательной фонд Джеймса Рэнди (James Randi Educational Foundation, Флорида, США) за успешное лабораторное обследование феномена группой ученых. Информация для особо ретивых читателей — приз настоящий, но с 1998 г. он еще не был вручен ни одному человеку в мире...

Физиогномика в цифрах

Физиогномика (греч. "physiognomike" — природа и знание) — это учение о толковании характера, ума человека по его чертам лица и форме тела. Считается, что опытный физиогномист может по фотографии дать психологический портрет человека. Чудо? Скорее, тонкая наблюдательность, интуиция, практика и немного везения. Последнее обстоятельство обязательно, ведь зачастую желаемое не совпадает с действительным. И все же, на сайте http://www.uniphiz.com/rus/ можно познакомиться с методами диагностики, параметрами, историей, полезными советами по физиогномике.

Существует компьютерная программа Digital Physiognomy (автор — Сергей Васильев, http://www. uniphiz.com/rus/dig_phys.exe, 794 Кб), которая для жителей СНГ распространяется бесплатно (пароль на странице http://www.uniphiz.com/rus/physiognomy_register.htm). Программа позволяет создать фоторобот человека (**рис.1**) и определить наиболее вероятные черты его характера. Тестируя своих знакомых, не забывайте относиться к результатам с юмором, ведь физиогномика — это всего лишь эмпирическое учение, а не точная наука. В любом случае программа пригодится для составления фотороботов, поскольку ее аналоги, как правило, платные.

Рис. 1



Ключевые слова для поиска: физиогномика, фоторобот, physiognomy.

"Heshby" — улучшенный "чет-нечет"

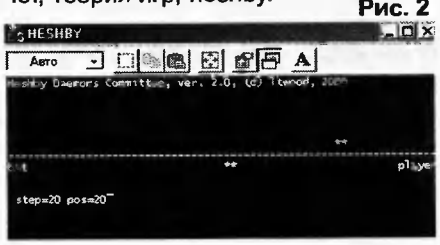
Может ли компьютер совершить "чудо" и обойти законы теории вероятности? Профессиональный математик Michael Machin (он же LtWood) смоделировал такую ситуацию в разработанной им программе "Heshby" (http://ltwood.hotbox.ru/heshby/files/heshby.zip, 45 Кб), Это разновидность известной с детства игры "чет-нечет". Суть задачи — один из играющих загадывает "чет" или "нечет", другой — отгадывает. Если угадал, то выиграл, не угадал — проиграл.

В теории игр строго доказывается, что при большом количестве попыток и оптимальной стратегии игроков, общий результат будет ничейным. Однако, если сделать поправку на психологию человека и поручить обработку данных компьютеру, то чашу весов можно склонить в свою пользу ([1], http://radio-mir.com/files/wk_2004/wk09/onetwo.bas). В программе "Heshby" применен более изощренный алгоритм анализа на основе стохастического предиктора, рефлексивного "зеркала", инфляции с монотонно растущим поощрением. В результате программа выявляет закономерности поведения человека на подсознательном уровне, приспосабливается к ним и, как правило, выигрывает!

Объяснение в том, что человек "генерирует" не случайные, как ему кажется, числа "+" (чет) или "-" (нечет), а модулированные по вполне определенному закону. Специалисты в области теории вероятности могут спать спокойно, поскольку существует алгоритм выигрыша над программой "Heshby v.2.0", который начинается с кодов "+ - - - - - - - - + + - + - - + - + - + - + +" (**рис.2**). Продолжить последовательность до полного выигрыша (верхние звездочки должны достичь правого края экрана) предлагается самим читателям.

Ключевые слова для поиска: чет-нечет, теория игр, heshby.

Рис. 2



"Данетки" или "yesno"?

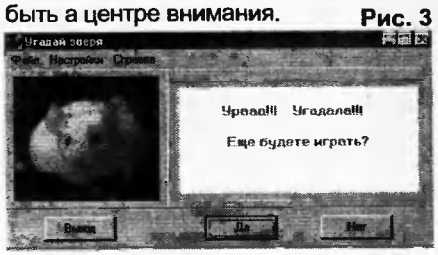
Речь пойдет о разновидности логических головоломок ситуационного типа. По определению лингвистов, это короткая загадка, которую нужно отгадать, задавая вопросы ведущему и получая на них ответы: "да", "нет" или "не важно, "не корректно". Исторически игра

начала свой путь в США, с издания книги Пола Слоана, название которой можно перевести как "Загадки для нестандартно думающих". Распространенные слова-синонимы "данетки", "игра в ситуации", "situation puzzles", "yesno".

На сайтах http://isabase.philol.msu.ru/~timik/, http://www.danetka.ru/, http://danet.exler.ru/ можно не только ознакомиться с русскоязычными вариантами "данеток", но и поучаствовать в их разгадывании в режиме "online". Пример вопроса: "На олимпиаде средний боксер выиграл три боя. Сначала удары были обычные, но постепенно крепчали, достигая такой силы, будто били камнем. Потом он был дисквалифицирован. Почему?" Ответ: "Руки обмотаны бинтом, который пропитан гипсом".

Добраться вопросами до истины порой бывает очень сложно. Ситуации, как правило, парадоксальные и, как будто бы, не имеющие логического решения. Но после объяснения "чуда" всегда хочется сказать: "И как я сам до этого не додумался!" Кстати, любители шумных компаний обязательно должны аыучить пару-тройку "данеток", чтобы всегда быть а центре внимания.

Рис. 3



Компьютерные варианты "данеток" реализованы в программах "Угадай зверя" (автор — Александр Веселов, http://www.vives.chat.ru/programs.files/Animals1.zip, 542 Кб, **рис.3**, http://www.vives.chat.ru/programs.files/Animals2.zip, 186 Кб) и "Угадай животное" (http://sofist.by.ru/program/danet.rar, 408 Кб). В программах в качестве рассказчика выступает компьютер. Игрок должен задавать ему (выбирать из списка) вопросы и по ответам пытаться отгадать задуманное животное. Для повышения зрелищности рекомендуется создать коллекцию с файлами фотографий зверей в разных ракурсах, сохранить их в отдельной папке и включить в программе опцию слайд-шоу с периодичностью 4...6 с.

Программисты могли бы на основе этой идеи разработать свои собственные программы "данеток" с речевым сопровождением и видеороликами.

Ключевые слова для поиска: данетки, головоломки, situation puzzles.

Литература

1. Рюмик С. "Чет-нечет" и теория игр. — Радиолюбитель. Ваш компьютер, 1997, №3, С.32.

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

Немного об FTP

FTP (File Transfer Protocol — протокол передачи данных) предназначен для передачи файлов с удаленного компьютера на локальный.

Многие начинающие пользователи, которые хотят открыть свой сайт или уже это сделали, сталкивались с "вечной" проблемой web-интерфейса на бесплатных хостингах — через него, "родимого", очень медленно закачиваются файлы, скорость загрузки страниц оставляет желать лучшего, а в совокупности с нашим "тормозным" dial-up обновление сайта превращается в настоящую нервотрепку. Другое дело — протокол обмена данными FTP. Он не имеет бесполезных картинок и баннеров, "туповатых" силиконовых меню и все тормозящих java-скриптов. Главное назначение FTP — пересылать, копировать, передавать. Этот протокол может использоваться как самостоятельно, так и через другие системы. Например, WWW содержит в себе FTP как часть своего протокола. Каждый сервер в сети Internet имеет свой ftp-сервис, разумеется, если он не заблокирован системным администратором. Всякий уважающий себя бесплатный хостинг также предоставляет своим пользователям доступ к FTP наряду с более понятным и простым web-интерфейсом. О правилах доступа обычно можно узнать после регистрации. Например, всем известный www.narod.ru имеет адрес ftp.narod.ru.

Предположим, что вам известны адрес ftp-сервера, логин и пароль доступа. Допустим даже, что это ваш сайт. Хорошо было бы соединиться с ним и закачать на него десяток-другой файлов вашей домашней странички. Для этого нужна программа-клиент (это сервисная программа, с помощью которой можно произвести соединение с сервером). В Windows 98...XP есть встроенная программа этого класса, а также возможность подключения к FTP через стандартный Explorer. Если у вас нет возможности скачать или взять откуда-нибудь графический пакет, то все можно сделать стандартными средствами. Для этого в командной строке Windows пишем: ftp 127.0.0.1 (где ftp — команда, а 127.0.0.1 — имя сервера, к которому вы хотите подключиться). Далее, если не возникло никаких сбоев при подключении, у вас появится приглашение к работе вида: ftp>. Это очень напоминает стандартный command.com, даже принципы те же. Для перемещения по удаленному компьютеру вам необходимо уметь пользоваться командами.

Основные команды FTP

open имя_сервера — открывает соединение с сервером. Это имя можно указать сразу при вводе команды, загружающей клиента.

cd имя_директории — осуществляет переход в другой рабочий каталог на ftp-сервере.

dir [имя_файла] — выдает список файлов а текущей директории. Не забывайте, что можно использовать шаблоны групповых операций — это "*" и "?".

get имя_файла [имя_локального_файла] — переписывает файл с удаленного компьютера на локальный. Если указано имя локального файла, то записывает его под этим именем, иначе — в каталог по умолчанию.

mget [имя_файла] — то же самое, что и get, но разрешается использовать шаблоны. Перед копированием каждого файла будет запрашиваться подтверждение. Для отмены подтверждений введите prornpt.

prompt — отменяет подтверждение в командах mget и mput.

put имя_файла [имя_удаленного_файла] — переписывает файл с локального компьютера на удаленный под именем (имя_удаленного_файла). Если оно не указано, то файл записывается в текущий каталог с именем локального. Команда запрещена для анонимных пользователей.

mput [имя_файла] — записать группу файлов, то же самое, что и put, но разрешается использовать шаблоны. Перед записью каждого файла будет запрашиваться подтверждение. Для отмены подтверждений введите prompt.

ascii — устанавливает ascii-способ передачи файлов. Используется для пересылки текстовых файлов на английском языке. Однако для надежности лучше использовать режим binary.

binary — устанавливает двоичный способ пересылки файлов. При этом файл при передаче не перекодируется и записывается в неизмененном виде — это наиболее надежный способ передачи файлов.

close — закрывает соединение с данным сервером и производит возврат в командный режим. Эта коман-

да автоматически выполняется при выходе из ftp-клиента.

quit — выход из ftp-клиента.

user — регистрирует на текущем сервере с новым именем. Используйте эту команду, если вы первый раз по ошибке неправильно ввели имя анонимного пользователя и не хотите снова перенабирать команду open.

lcd [имя_директории] — осуществляет переход на локальном компьютере в указанный каталог.

pwd — выводит на экран текущий каталог на удаленном компьютере.

system — выводит на экран тип операционной системы на удаленном компьютере.

help [ftp-команда] — помощь, выдает краткую информацию о командах ftp-клиента или о конкретной указанной команде.

Возможны два случая соединения по FTP. Первый — если вы просто хотите посетить определенный сервер, чтобы что-то скачать. Для этого вы вводите логин guest (гость) или anonymous (аноним), это при условии поддержки анонимного доступа, а также, если на сервере отсутствует регистрация. Второй — если вы имеете доступ на сервер. Тогда вводится ваш логин и пароль. Однако, независимо от ситуации, в итоге вы оказываетесь в некой директории на удаленном сервере. Чтобы у вас не возникло проблем, поясню, что на сайтах, подобных www.narod.ru, в текущем каталоге вы не обнаружите никаких папок, вам придется их создавать самим. Наличие директорий \cgi-bin, \php и т.д. обусловлено, прежде всего, сервисами, поддерживаемыми на данном конкретном хостинге.

Зайдя на сервер, вы можете без проблем копировать файлы! Советую начать исследование сервера с каталога \pub, так как обычно все полезные файлы помещаются именно в него. Для того чтобы перейти в нужный каталог, используется команда cd. Из текущего каталога в каталог pub можно перейти так:

```
ftp> cd pub.
```

Получить список файлов в текущем каталоге можно командой dir:

```
ftp> dir.
```

А чтобы увидеть все файлы с определенным расширением, вы можете использовать шаблоны "*" и "?":

```
ftp> dir *.zip.
```

Предположим, что вы нашли файл, который хотите переписать к себе на компьютер. Прежде чем сделать это, надо установить двоичный режим передачи файлов binary:

```
ftp> binary.
```

Возьмите себе за правило — как только соединитесь с сервером, сразу вводить эту команду. Если этого не сделать, то файл будет перекодирован и далее непригоден для использования (если только это не текст на английском языке). При появлении во время копирования файла сообщения

```
'Opening ASCII mode to transfer file'
```

немедленно прервите процесс копирования файла и запустите команду binary. Многие современные ftp-клиенты автоматически посылают эту команду.

Пересылает файл на локальный компьютер команда get:

```
ftp> get file.zip.
```

Если вы сразу захотите поместить файл в определенное место на локальном компьютере, то укажите путь как второй аргумент команды:

```
ftp> get file.zip D:\file.zip.
```

По умолчанию ftp-клиент помещает файл в текущую директорию на локальном диске. Для ftp-клиента под Windows этой директорией будет каталог Windows. Можно переписать содержимое сразу всего каталога, для этого надо указать его имя в команде get. Зачастую требуемая информация расположена не в одном, а в нескольких файлах.

Я специально начал свой рассказ о подключении к FTP при помощи текстового клиента, так как основа основ — это команды, на которых базируется любая система. В настоящее время можно без труда использовать простые графические программы. Уже почти никто не утруждает себя текстовым вводом, этот режим рассматривают лишь как пример для ознакомления с азами. Для пользователей ОС Windows написано множество различных ftp-клиентов. Например, очень "радует взор" ftp-клиент, встроенный в Total Commander. Для того чтобы создать подключение к удаленному серверу, вам следует нажать клавиши "Ctrl+F", после чего у вас появится диалоговое окно (**рис.1**).

Для того чтобы выбрать интересующий нас ftp-сервер, нужно нажать кнопку *Добавить*. В строке *Заголовок* вам нужно будет ввести название соединения с сервером. Допустим, это будет MYHOMEAGE. В строке *Имя сервера* вводим адрес, допустим, ftp.narod.ru. В строке *Учетная запись* вводим логин, а в строке *Пароль* — ваш пароль доступа. Остальные функции вас не должны волновать. Если все выполнено верно, то окно клиента у вас будет заполнено примерно так, как на **рис.2**.

После указания и внесения всех данных, вам следует лишь нажать "*Соединиться*" и подключиться к вашему ftp-серверу. На одной из панелей Total Commander высветится директория на удаленном компьютере. Теперь вы можете закачивать любые файлы.

Рис. 1

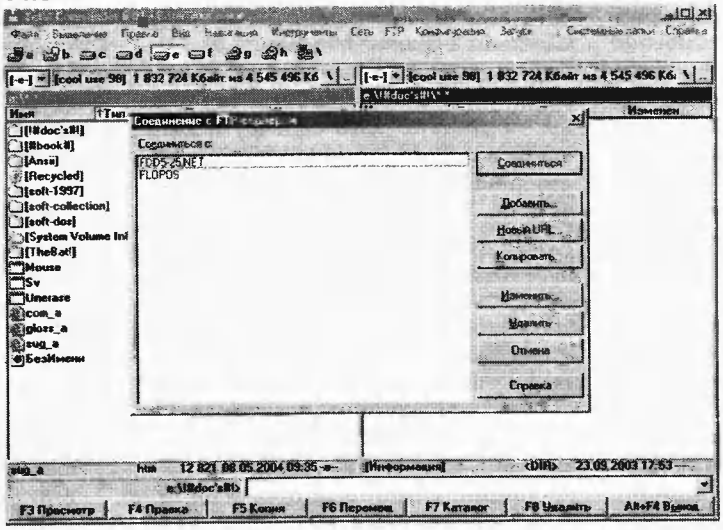
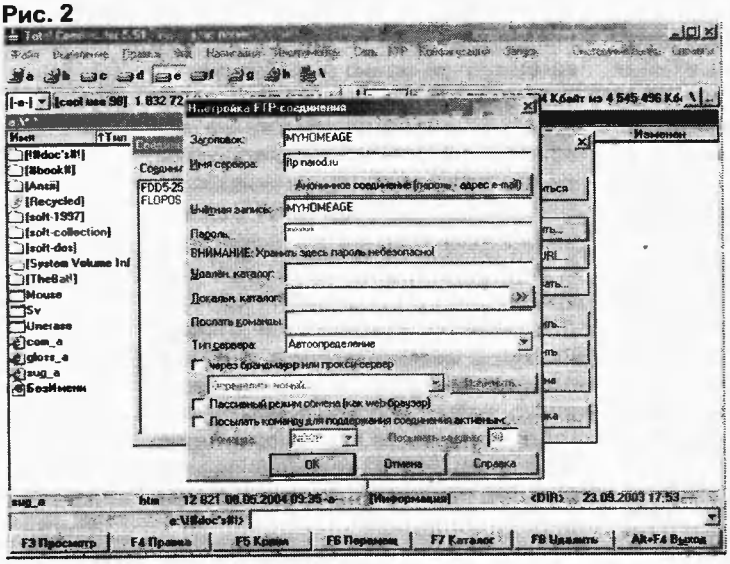


Рис. 2



Однако не следует забывать, что если ваш хостинг работает под управлением unix-подобной операционной системы, то строчные и заглавные символы будут интерпретироваться как различающиеся, и у начинающих пользователей это часто вызывает неразбериху. Например, файлы index.HTML и index.html будут считаться разными. Поэтому в своих ссылках всегда правильно указывайте название документа.

Как показывает практика, в настоящее время основным сервисом Internet является WWW. Каждый, кто работает с WWW, знает, что такое URL (Uniform Resource Locator) — сегодня это основной способ указания ресурсов Internet. Например, URL для html-файла — http://www.fdd5-25.net, для почтового адреса — mailto:Commander-Norton@tut.by. FTP также является ресурсом Internet, URL для него выглядит следующим образом: ftp://<user>:<password>@<host>:<port>/<url-path>. Здесь user — имя пользователя, password — его пароль, host — доменное имя или IP-адрес сервера, url-path — путь к файлу. На практике наиболее часто используемым вариантом доступа по FTP является анонимный. Как уже упоминалось, анонимный FTP ничем не отличается от "остального", просто в качестве имени пользователя достаточно указать anonymous, а в качестве пароля — свой почтовый адрес. Синтаксис URL для анонимного FTP упрощенный — ftp://<host>/<url-path>, т.е. при отсутствии имени автоматически будет вставлено anonymous. Порт также обычно не указывается, а используется стандартный 21. В качестве host можно указывать и IP-адрес. Основное применение URL нашли в WWW-браузерах (Internet Explorer, Netscape Navigator, Mosaic, Lynx, Opera и др.), на сегодня это, пожалуй, самые популярные программы в Internet. Поэтому имеет смысл использовать их и как ftp-клиент. Правила формирования адреса уже были описаны, все остальное предельно просто. Если в качестве пути указан только путь к некоторому каталогу, а не сам файл, то браузер покажет вам список файлов в этом каталоге. Если же путь указан вплоть до имени файла, то вскоре файл с некоторой вероятностью окажется у вас на диске. Почему с некоторой вероятностью? Потому что, к сожалению, протокол HTTP использует для FTP собственную подсистему пересылки файлов, что отнюдь не привело к повышению надежности. Искушенный пользователь WWW заметит, что иногда адреса файлов в URL в качестве параметра scheme содержат значение http вместо ftp. Это разные, и в то же время, одинаковые вещи. Дело в том, что указывая http, вы "поручаете" www-серверу искать файл в области каталогов, которые отведены html-файлам. Если указывать ftp, то каталоги будут совпадать с теми, которые доступны через классический FTP. Сравните:

`ftp://ftp.karelia.ru/pub/unix /home/ftp/pub/unix` и

`http://ftp.karelia.ru/pub/unix /files/www/pub/unix`.

Хотя реально принципы передачи информации в обоих случаях одинаковы, здесь кроется одна очень большая проблема. Некоторые организации, имеющие быстрый доступ к Internet, забывают, что не у всех он такой же быстрый. Таким пользователям удобно использовать классический FTP вместо WWW-браузера. Дело в том, что реализация FTP а HTTP оставляет желать лучшего. Основной предмет критики — отсутствие такой полезной функции как reget (докачка). Это особенно актуально для низкоскоростных отечественных коммуникаций, где скорость трафика порой падает до нескольких десятков байтов в секунду. При такой скорости разрыв соединения — очень частое явление, и переслать файл размером в несколько мегабайтов уже является большой проблемой. При потере соединения вы можете воспользоваться функцией reget в ftp-клиенте, однако в среде WWW вам, увы, придется начать все сначала. Более того, если вы работаете с WWW через proxy, основанном на программном обеспечении от CERN, то есть вероятность, что при обрыве соединения он решит, что весь файл уже переслан, и на все попытки получить его с удаленного WWW-сервера proxy будет выдавать урезанный файл со своего диска. В таком случае можно дать два совета: либо смените proxy-сервер, либо вообще отключите его использование. Однако следует иметь в виду, что сейчас очень популярны так называемые firewall, когда реальный выход в Internet имеют только proxy, и последний совет в таком случае бесполезен. Есть еще третий совет — подождите несколько дней, пока proxy не позабудет о вашем файле, и снова обратитесь в Internet для его пересылки. Однако это, безусловно, не рабочее решение.

Довольно часто http- и ftp-каталоги синхронизированы, хорошим представителем таких каталогов является ресурс ftp.cdrom.com.

И напоследок одна "пилюля". В соответствии с протоколом http, через который осуществляются коммуникации WWW, после каждого сеанса связи соединение между компьютерами разрывается. Это означает, что если вы решите в свое удовольствие погулять в браузере по каталогам, то, возможно, это не всегда будет так здорово, как кажется. На установление соединения, регистрацию пользователя уходит несоизмеримо больше времени, чем на интерактивную работу в ftp-клиенте. Это плата за современный модный интерфейс.

Подведем итоги

Преимущества FTP:

- высокая интеграция в гипертекстовое пространство Internet;
- современное программное обеспечение.

Недостатки:

- отсутствие reget;
- невысокая надежность соединения на плохих линиях;
- проблемы при обрыве соединения со включенным proxy;
- невысокая скорость работы из-за закрытия соединения после пересылки;
- недоступность через FTP файлов, адресуемых через протокол HTTP (хотя это и не недостаток самого HTTP, это его особенность).

Однако, как нетрудно заметить, все эти недостатки могут быть компенсированы надежными и быстрыми линиями связи. Уже достаточно канала в 256 k у провайдера, и основные проблемы исчезают.

Но, столкнувшись с работой по протоколу FTP на выделенных линиях, я заметил одну очень интересную особенность белорусских провайдеров. Почти у всех компаний скорость работы с FTP оставляет, мягко говоря, желать лучшего. Таких "тормозов" не бывает даже при соединении по dial up. А ведь крупные сайты обновлять приходится именно через выделенный канал. В итоге ни в одном из компьютерных клубов не было достойной скорости соединения. Господа провайдеры, имейте совесть! Понятно, что вы платите за трафик, а люди — за время. Но как быть тем, кому нужно выкладывать на сайт большие объемы информации? Не ходить же каждый раз изводить своего хостера?!

При написании статьи были использованы материалы замечательного автора А.Стояновского.

ЛИСТАЯ СТРАНИЦЫ

Истории развития и современному состоянию компьютерных и докомпьютерных шрифтов посвящена статья *Ю.Анищенко* "От папируса до монитора: долгая история шрифтов" **(Подводная лодка, №5/2004, С.92...96)**.

Письменность зародилась более 6 тысяч лет назад, в Древнем Египте и в Месопотамии, однако как иероглифы, так и знаки клинописи не позволяли писцам в достаточной мвре выразить свою индивидуальность. Только с появлениям в первом тысячелетии до нашей эры финикийского алфавита, прямыми наследниками которого стали греческий и еврейский, люди получили в свое распоряжение достаточно гибкий и удобный в повседневной жизни инструмент. За счет этого доля грамотного населения начала неуклонно расти, и спрос на обученных писцов привел к возникновению специальных школ. Каллиграфические школы основывались, как правило, на непререкаемом авторитете их основателя. Ученики старательно копировали почерк своего наставника и передавали его своим собственным воспитанникам. Именно так и зарождались первые семейства шрифтов. Благодаря бережно хранимым традициям, сегодня мы используем ряд начертаний, возраст которых насчитывает около двух тысяч лет.

Гибель античного мира на какое-то время затормозила развитие каллиграфического искусства Запада, но пережившая культурную катастрофу церковь не только сохранила для мира латинскую письменность, но и помогла создать на ее основе алфавиты для зарождавшихся национальных языков новой Европы. В это же время греческие миссионеры создали ряд алфавитов, базирующихся на греческой письмвнности, в том числе и фундаментальный вариант нашей кириллицы.

После превращения империи Каролингов в ведущее государство Западной Европы, на ее территории стали возникать собственные каллиграфические школы. Их участники не ограничивались сохранением римского наследия, активно развивая новые письменные формы. Так, именно на севере современной Франции зародился хорошо знакомый всем "готический" шрифт, который сегодня воспринимается как один из непременных атрибутов средневековья.

В середине XV века немецкий изобретатель Иоганн Гутенберг, посвятивший свою жизнь работе с металлами и ювелирному делу, создал печатный пресс, литерный пуансон и наборный алфавит. На первых порах изготоаление шрифтов и набор с их помощью текстов было далеко не простым занятием. Печатные литеры отливались из специального сплава, но этому предшествовал этап формовки и создания шаблонов. Сначала опытный каллиграф рисовал все знаки, которые должны были войти в состав шрифта, а затем гравер переводил эти изображения на металлические бруски и вырезал прообраз литеры. С него делалась пробная отливка, после чего качество полученной буквы проверяли на печатном станке. Подчас этот процесс приходилось повторять заново, пока мастер-печатник не оставался доволен всеми литерами гарнитуры. Не удивитвльно, что полный шрифт с набором шаблонов ценился на вес золота, а популярные шрифтографы быстро богатели.

На рубежв XV-XVI веков Европа вступает в золотой век типографики. Первым в нем заслуженно идет имя Альда Мануция — основателя венецианской типографии Aldine Press. Мануций много и охотно экспериментировал со шрифтами, черпая вдохновение в работах античных мастеров. Не удивительно, что одну из лучших своих гарнитур он так и назвал — Antiqua. Самым же значительным достижением Мануция считается изобретение "италических" шрифтов, литеры которых, подобно рукописным буквам, наклоняются вправо.

В тридцатые годы XVI века дизайнерские идеи Мануция были восприняты и творчески развиты французским типографом Клодом Гарамоном. Им были созданы шрифты, учитывающие особенности различных европейских языков, благодаря чему печатные кассы его производства можно было встретить в типографиях многих стран.

Постепенно искусство типографики распространялось на север Европы. В XVII веке славились шрифты, изготоаленные голландскими мастерами. Они хорошо продавались во все сопредельные страны, в том числе и в Англию, где зародились свои собственные школы шрифтового дизайна. Наиболее видными представителями британского направления типографики считаются Вильям Каслон и Джон Баскервиль. Открыв собственную типографию в 1729 году, Каслон какое-то время использовал проверенные временем голландские гарнитуры, однако постоянно совершенствовал их и вывел нв качественно новый уровень. Спустя короткий промежуток времени, практически все печатники Англии стали пользоваться если не самими литерами Каслона, то по крайней мере заложенными в них дизайнерскими находками. На долгие годы гарнитуры Каслона стали de facto стандартом в Великобритании, а также ее многочисленных колониях. Даже американская "Декларация о Независимости" была вперые набрана именно его шрифтом. Кроме того, Каслон прославился и тем, что создал шрифты для ряда восточных языков, заметно расширив тем самым рынок полиграфической продукции.

Баскервиль прославился как изобретатель и рационализатор типографских технологий. Созданные им шрифты отличались особой утончвнностью и элегантностью, которые зачастую пропадали атунв из-за несовершенства оборудования, поэтому мастер много и успешно экспериментировал с химическим составом краски, предпечатной обработкой бумаги, деталями оборудования. К 60-м годам XVIII века его типография была одной из наиболее совершенных и во многом определила развитие отрасли на последующие годы.

Конец классической эпохи шрифтового дизайна в Европе связывают с именем Джамбаттисты Бодони. Генивльный итальянский шрифтограф жил и работал на рубеже XVIII—XIX веков и еще при жизни был признан не только как автор великолепных гарнитур, но и как исследователь теории шрифтов. Его двухтомный труд "Типографическое руководство" систематизировал накопленный за предыдущие столетия опыт, а также содержал рекомендации по разработке и использованию шрифтов. Гарнитурами из его коллекции продолжали пользоваться на протяжении всего XIX века.

Имена великих мастеров прошлого увековечены в названиях разработанных ими гарнитур. Сегодня на многих персональных компьютерах встречаются шрифты Bodoni, Caslon, Baskerville, Garamond. Но, каких бы высот ни достигал полет фантазии дизайнера, слишком многое в те временв зависело от возможностей полиграфического оборудования — печатных и наборных аппаратов. Для того чтобы кардинально повысить эффективность типографского процесса, необходимо было придумать, как максимально упростить процесс формирования печатной формы, то есть матрицы, содержащей текст и иллюстрации будущей страницы. Понятно, что сборку макета из отдельных литер и даже гранок невозможно ни серьезно ускорить, ни полностью автоматизировать, поэтому чисто механический метод набора быстро исчерпал свой потенциал, поставив печатную отрасль пвред необходимостью поиска новых технологий.

Решение было найдено в использовании фотографического метода. Будущая печатная форма покрывалась светочувствительным составом и обрабатывалась точно так же, как и обычная фотобумага при печати снимков: сперва на нее проецировался трафарет страницы со своеобразного "негатива", затем она подвергалась химической проявке. В результате на форме оставались символы, а пробелы между ними растворялись и вымывались. Здесь, однако, появлялся новый вопрос — каким образом быстро и качественно создавать негативные пленки?

Эту задачу стали выполнять так называемые фотонаборные устройства. Принцип их действия был довольно прост. Вместо того чтобы хранить полную коллвкцию металлических символов шрифта различного размера, в машину закладывали их контурные трафареты на прозрачном носителе. Оператор-наборщик аводил текст с клавиатуры. Как только он нажимал на ту или иную клавишу, механизм изалекал негатив соответствующей буквы и проецировал через него пучок яркого света на фотопленку. Проявив последнюю, мастер получал готовую матрицу для изготовления печатной формы. Кроме того, фотонаборная машина позволяла масштабировать размер знаков оптическим методом. Однако как минимум две проблемы — качество печати и разработку новых шрифтов — фотомеханическая технология решить не смогла.

Ситуация оставалась неопределенной вплоть до начала широкого внедрения вычислительной техники в различные от-

расли, в том числе и полиграфическую. Впервые для оцифровки шрифтов компьютер был задействован во второй половине шестидесятых годов прошлого века. На первых порах замятного преимущества перед фотомеханическим методом добиться не удавалось: новая технология не предусматривала революционного изменения способа проецирования набранного текста на светочувствительную пленку. Единственное новшество заключалось в том, что вместо наборной машины использовалась обычная электронно-лучевая трубка, на которую выводились хранящиеся в памяти машины строки. При этом наконец-то удалось абстрагировать смысловое содержание будущего печатного документа от визуального представления его шаблона: один и тот же текст можно было вывести на печать, изменяя как размеры, так и типы задействованных шрифтов.

Однако для того чтобы попасть в память компьютера, шрифт должен был быть предварительно оцифрован на сканирующем устройстве с какого-либо оригинала. Для хранения информации о символах использовалось матричное представление. С тех пор и пошло семейство так называемых экранных шрифтов. Именно символами экранных шрифтов выводятся во время загрузки компьютера все сообщения программы BIOS, а также информация о диагностике ранних этапов запуска операционной системы. Отличительная особенность экранных шрифтов — ограниченные возможности масштабирования. Фактически каждый шрифтовой файл подобного типа содержит несколько типоразмеров матриц, гарантирующих качественное воспроизведение шрифта на экране при строго заданной величине последнего.

Развитие компьютеров и компьютерной графики в корне изменило роль шрифтов в формировании прообраза печатной страницы. Если раньше их матричная проекция накладывалась на фон, содержащий прочие элементы оформления, то теперь сами символы можно прорисовывать точно так же, как остальную графику, и компоновать тем самым целостный образ всей страницы. Новая идеология использования шрифтов подразумевала новые методы их описания и хранения. Теперь символы рассматривались как рисунки, состоящие из набора кривых — то есть элементов векторной графики. Вместо того чтобы записывать в память компьютера готовые "образы" знаков, файл описания шрифта стал включать в себя набор векторных функций, задающих свойства отрезков на координатной сетке.

Первой промышленной технологией использования векторных шрифтов стал знаменитый язык PostScript. Его разработка была начата в корпорации Xerox, затем его авторы основали компанию Adobe. Развитие этого языка поддержала Apple. Инвестиции на сумму более двух миллионов долларов серьезно помогли в доведении нового стандарта "до ума", и вскоре Apple смогла создать первую в истории полностью компьютеризированную систему верстки. Использование PostScript гарантировало полный контроль над шрифтами, начиная с приложений и заканчивая разнообразными устройствами вывода.

Однако при всех своих достоинствах PostScript-шрифты не были универсальной панацеей. Во второй половине 80-х годов XX века ведущие производители операционных систем — Microsoft, Apple и IBM — четко обозначили курс на разработку графического интерфейса пользователя, активно использующего масштабируемые векторные шрифты. Поскольку Adobe настаивала на лицензионных отчислениях за встраиваемый PostScript, были предприняты попытки создать альтернативную технологию управления шрифтами на экране монитора. Apple и Microsoft приступили к совместной разработке платформы TrueImage, договорившись о взаимном кросс-лицензировании ее результатов. Хотя эта попытка окончилась неудачей, накопленный опыт не пропал даром. В 1987 году Apple опробовала в своих продуктах принципиально новый стандарт шрифтов — TrueType, а несколькими годами позже Microsoft адаптировала его для Windows 3.1. Последующие годы получили название "войны шрифтов": Adobe, Apple и Microsoft активно продвигали свои собственные решения, не желая идти на сотрудничество друг с другом.

К счастью, накал страстей спал, и необходимость консолидации отрасли взяла верх над корпоративными амбициями. Adobe и Microsoft вернулись к сотрудничеству и разработали новый стандарт OpenType, который поддерживался операционными системами Windows и имел некоторые преимущества по сравнению практически со всеми предыдущими технологиями.

В это же время Apple разработала собственную технологию GX, которая выигрывает у OpenType в тех случаях, когда дело касается сложных символов, присутствующих в ряде языков народов мира в качестве самостоятельных единиц или образующихся из комбинаций соседних символов. Компания ведет работу по адаптации шрифтов OpenType через промежуточный интерфейс, который позволит использовать их во всех приложениях, ориентированных на GX.

По мере вытеснения бумажных носителей печатной информации электронными все больше и больше внимания уделяется повышению качества экранных шрифтов. В этом контексте примечательна инициатива Microsoft, направленная на разработку специальной технологии ClearType. Этот механизм изменяет стандартную процедуру растеризации символов, сглаживая их контуры и делая текст намного более удобным для чтения. ClearType предназначена в первую очередь для владельцев портативных компьютеров и жидкокристаллических мониторов, но даже на обычном ЭЛТ-мониторе разница бросается в глаза.

Перспективам перехода к использованию оперативной памяти стандарта DDR2 посвящена статья "DDR2: двойное ускорение" **(CHIP, №5/2004, С.46...49).**

Переход к новому стандарту будет непростым. Все помнят, что процесс стандартизации слегка забуксовал при переходе от DDR333 (PC2700) к следующей спецификации. Тогда лидер по производству чипсетов — корпорация Intel — решительно противилась внедрению модулей памяти DDR400 (PC3200) и хотела намного раньше вывести на рынок DDR2. Причиной этого были технические трудности в создании чипсетов под DDR400. И лишь в конце марта 2003 года, когда были созданы чипсеты i875P и i865, поддерживающие этот тип памяти, Intel официально признала спецификацию для DDR400.

Сегодня промышленность опять медлит с переходом на новую технологию памяти. Но поколение DDR закончится с уходом модулей оперативной памяти DDR400 с рынка. Надежная работоспособность более быстрых модулей (типа DDR533/PC3700) не может больше обеспечиваться, поскольку в будущем для них не будет существовать официального стандарта JEDEC. JEDEC (Joint Electron Device Engineering Council, объединенный совет по электронным устройствам) — это альянс, объединяющий в своих рядах около 300 производителей чипсетов и чипов ОЗУ, занимающийся разработкой стандартов оперативной памяти.

Передача данных в модулях DDR2 также осуществляется методом Double Data Rate, однако различие в том, что у модуля DDR2 буфер ввода-вывода работает на частоте, превышающей частоту ядра в два раза. Так, за один такт передается два блока данных, и скорость передачи по сравнению с DDR увеличивается в два раза. Объем буферной памяти в идвале удваивается, а интерфейс связи ядра чипа и буфера ввода-вывода расширяется до четырех конвейеров. Следовательно, для достижения той же полосы пропускания, что у DDR-памяти, ядро чипа DDR2 может работать в два раза медленнее.

Преимущество налицо — благодаря меньшим значениям тактовой частоты ядра уменьшается нагрузка на чип, за счет чего значение питающего напряжения может быть снижено до 1,8 В. Это приводит к уменьшению выделения тепла модулями оперативной памяти. Использование DDR2 позволит в будущем повысить и рабочую частоту ядра чипа — до той величины, при которой можно будет контролировать увеличивающуюся частоту буфера ввода-вывода.

Приведенная оценка применима лишь в идеальном случае. Наряду с тактовой частотой ядра памяти, реально достижимая пропускная способность зависит еще и от задержки при ожидании информации, затребованной от накопителя, которая указывает на количество тактов, необходимых для подготовки данных. Вероятность того, что буфер ввода-вывода модулей DDR2 при одиночном доступе будет заполняться по всем имеющимся четырем конвейерам, ниже, чем у DDR SDRAM, имеющей два конвейера. В связи с этим эффективность DDR2 несколько снижается. Поэтому первые модули DDR2 (вероятно, это

будут DDR2-400/PC2-3200) продемонстрируют в тестах скорость, на 5...10% меньшую, чем их соперники из лагеря DDR400. Этот недостаток через некоторое время будет ликвидирован, поскольку тактовая частота ядра DDR2 быстро возрастет.

Еще одно преимущество DDR2 заключается в терминации шины — методе согласования нагрузок с полным волновым сопротивлением шины передачи данных, при котором устраняются отражения сигнала от концов шины. На материнской плате с чипсетом для DDR-памяти имеется заземленное сопротивление, которое очищает приходящие на интерфейсы ввода-вывода сигналы от помех. Эти помехи возникают в результате того, что каждый сигнал на шине данных отражается соседними компонентами. Чем больше модулей установлено, тем больше вероятность нежелательных отражений. Сопротивления на материнской плате могут подавить интерференцию в шине, но полностью устранить ее они не в состоянии. Вот почему полоса пропускания по-прежнему остается значительно ниже теоретически возможного уровня.

Память DDR2 обходит это препятствие намного элегантнее — сопротивления для терминации размещены прямо на самом модуле. При этом контроллер памяти посылает на шину сигнал, заставляющий все неактивные модули DDR2 переключаться в режим терминации. Поэтому в проводнике остается лишь активный сигнал, что практически полностью исключает интерференцию. Кроме того, материнские платы для DDR2 не так сложны в проектировании и дешевле в разработке.

Первым продуктом, выигравшим от введения новой технологии, станет Pentium 4 Prescott. Этот процессор в сочетании с чипсетом, поддерживающим DDR2 (кодовое наименование Grantsdale), позволит получить значительный прирост производительности. Он в состоянии работать с модулями PC2-3200 (DDR2-400) и с PC2-4300 (DDR2-533). Последний вариант был бы идеальным добавлением к Prescott, который Intel намеревается эксплуатировать на частоте системной шины 266 МГц вместо нынешних 200 МГц.

В лабораториях производителей оперативной памяти уже давно работают прототипы стандарта DDR2. Признанные лидеры данной отрасли (Samsung, Infineon, Micron и Elpida) вовсю работают с утвержденной консорциумом JEDEC в сентябре 2003 года спецификацией DDR2.

Однако для конечного потребителя переход от DDR на DDR2 означает необходимость приобретения новой материнской платы. Несмотря на то что по длине модули обоих стандартов одинаковы (около 13,35 см), модуль DDR2 имеет 240 контактов и чисто механически не совместим с обычным DDR-слотом, который предусмотрен для модулей, имеющих 184 контакта. Кроме того, DDR2-модули требуют напряжения 1,8 В, тогда как DDR работают при напряжении 2,5 В.

Ноутбуки выиграют от введения даже первого поколения модулей DDR2. Не повышение скорости работы, а снижение энергопотребления делают эту технологию интересной для мобильных компьютеров. Уменьшение питающего оперативную память напряжения позволит продлить время работы аккумулятора в среднем на 6%.

Первые модули оперативной памяти DDR2 (DDR2-400/PC2-3200) обеспечат в лучшем случае такую же производительность, что и нынешние DDR400, но их энергопотребление будет значительно ниже. Реальный прирост производительности ожидается примерно в середине текущего года, когда возрастет внутренняя частота чипов спецификаций DDR2-533 (PC2-4300) и DDR2-667 (PC2-5300).

О современном состоянии DVD-технологий и возможных путях их развития рассказывает *Р.Соболенко* в статье "Три взгляда на одну историю" **(Hard'n'Soft, №5/2004, С.74...82).**

В конце 70-х годов прошлого века компании Philips и Sony приступили к разработке нового типа носителей, базирующихся на оптической технологии записи и считывания данных, которые были призваны прийти на смену виниловым грампластинкам. В результате в 1982 г. появились привычные сегодня музыкальные компакт-диски CD Digital Audio (CD-DA), спустя два года вышли и компьютерные CD-ROM. Их характеристики были определены исходя из поставленной задачи — записи музыки. С этим связан выбор спиральной дорожки, более пригодной для последовательного, а не для произвольного доступа к данным, характерного для компьютерных систем. Емкость дисков, составлявшая 650 Мб, была выбрана с точки зрения возможности записи 74 мин. музыки.

Эта новинка пришлась "ко двору" в ПК. Именно благодаря приводам CD-ROM возникло понятие "мультимедиа", в компьютерах платформы PC появились аудиофункции, пришли большие деньги в игровую индустрию. Технология развивалась, вслед за "штампованными" заводскими CD-ROM были созданы диски с однократной (CD-R) и многократной (CD-RW) записью. Однако, несмотря на многообразие принятых стандартов, острых проблем с совместимостью дисков разных типов и форматов удалось избежать. Уже тогда стандарты стали разделять на "физические", которые касались непосредственно структуры носителей и принципов записи, и "прикладные", описывающие представление данных.

Предпосылкой для разработки нового типа оптических носителей, сначала называвшихся Digital Video Disc, переименованных затем в Digital Versatile Disc, что подчеркивает их универсальность, послужило желание вслед за музыкой "переложить на цифру" и фильмы. За это взялись многие фирмы, по ходу дела разделившиеся на два лагеря. Matsushita Electric, Toshiba и киноконцерн Time/Warner предложили технологию Super Disc, в то время как Philips и Sony выступили с Multimedia CD. Оба формата были абсолютно несовместимы, что заставило вмешаться лидеров компьютерной индустрии. Под их давлением одиннадцать конкурирующих фирм с целью разработки единого стандарта сформировали DVD Forum. К концу 1995 г. были опубликованы спецификации DVD, большей частью основанные на технологии Super Disc. Окончательные версии стандарта были приняты в течение 1996 г.

Как и в случае компакт-дисков, стандарты разделились на физические и прикладные. Первые предусматривали пять типов дисков: DVD-ROM, предназначенные для дистрибуции компьютерных данных; DVD-Video, ориентированные на запись фильмов; DVD-Audio, содержащие только аудиоданные при исключительно высоком качестве воспроизведения; DVD-R, обеспечивающие однократную запись (конкурентом им стал формат DVD+R); DVD-RAM, позволяющие многократную перезапись и в большей степени ориентированные на компьютерные применения (этим дискам приходится соперничать с DVD-RW и DVD+RW). Унификация коснулась файловой системы. Было решено для дисков всех типов использовать систему Universal Disc Format (UDF), предложенную организацией OSTA. Это дало возможность избежать переписывания стандартов каждый раз, когда появляются новые приложения, как это было в случае CD. Однако поддержку UDF корпорация Microsoft реализовала начиная с Windows 98, а до этого в DVD-ROM пришлось применять UDF Bridge — гибрид UDF и стандарта CD ISO 9660.

После появления предварительных спецификаций DVD спохватилась киноиндустрия, потребовавшая реализовать в новом стандарте системы защиты от пиратского копирования. В свое время музыкальные издатели не побеспокоились об этом, в результате их бизнес понес громадные потери. По требованию Голливуда в стандарт DVD-Video к концу 1996 г. была включена система защиты Content Scrambling System (CSS), после чего и поступили в продажу диски с фильмами. В ноябре 1996 г. они стали доступны в Японии, спустя полгода — в США и лишь осенью 1998 г. — в Европе.

Компьютерные диски DVD-ROM появились в начале 1997 г., а уже летом от DVD Forum откололись Sony и Philips, которые сочли себя обиженными тем, что при разработке единого стандарта оказались "за бортом" многие аспекты предложенной ими технологии Multimedia CD. Фирмы, являющиеся родоначальницами эры компакт-дисков, на базе имеющегося у них опыта приступили к созданию стандарта DVD+RW, компанию им составила HP. Примкнувшие впоследствии к их инициативе Dell, Mitsubishi Chemical, Ricoh, Thomson, Yamaha сформировали организацию DVD+RW Alliance, к которой затем присоединилась и Microsoft. Впрочем, Sony все же решила не отказываться полностью от участия в DVD Forum. На фоне начавшегося раскола осенью 1997 г. появились однократно записываемые носители DVD-R.

В январе 1998 г. DVD Forum удалось утрясти проблему, вызванную использованием в Америке и Европе разных технологий для записи звуковых дорожек на дисках DVD-Video — соответственно Dolby Digital AC-3 и MPEG-2. Решение было принято "радикальное" — в стандарт включили оба варианта. Одновременно вышла черновая спецификация DVD-Audio. Летом того же года были выпущены первые серийные диски DVD-RAM емкостью 2,6 Гб, поддерживающие многократную перезапись, а осенью стали доступны и дисководы DVD-ROM, способные читать такие носители.

Окончательная версия 1.0 стандарта DVD-Audio была опубликована весной 1999 г., но еще год потребовался на реализацию в нем технологий защиты от копирования, без которых никто не согласился бы начать выпуск таких дисков. В октябре 1999 г. появился первый коммерческий двухслойный двусторонний диск DVD-Video, обладающий максимальной для стандарта DVD емкостью 17 Гб. Однако до сих пор индустрия предпочитает обходиться более дешевыми и простыми в производстве односторонними или однослойными носителями. С принятием спецификации 2.0 емкость DVD-RAM возросла до 4,7 Гб. Однако намечавшемуся триумфу этого формата помешало появление в конце 1999 г. поддерживающих перезапись накопителей DVD-RW (в Японии) и "раскольнического" привода HP DVD Writer 3100i, использующего диски DVD+RW емкостью 3,0 Гб.

Борьба "минусовых" и "плюсовых" стандартов вступила в открытую фазу. Впрочем, выход на рынок стандарта DVD+RW не прошел гладко, и еще полтора года потребовалось на решение проблем совместимости со многими приводами DVD-ROM и увеличения емкости носителей до 4,7 Гб. Только в марте 2001 г. организация DVD+RW Alliance смогла с чистой совестью отрапортовать о поступлении в продажу как компьютерных приводов данного формата, так и использующих его бытовых рекордеров.

В лагере DVD Forum тем временем не сидели, сложа руки. В мае 2000 г. была опубликована спецификация DVD-R версии 2.0, согласно которой однократно записываемые диски разделялись на профессиональные и пользовательские. Профессиональные носители предназначались для записи в заводских условиях и стали непригодны для записи на обычных приводах DVD-R. В следующем, 2001 г., DVD Forum принял спецификацию DVD Multi. Удовлетворяющие ее требованиям дисководы поддерживают запись и чтение всех продвигаемых DVD Forum записываемых носителей — DVD-RAM, DVD-R и DVD-RW. Они полностью совместимы и с дисками CD-R и CD-RW.

Первые носители DVD+R от Verbatim были представлены в начале 2002 г. Примерно в это же время девять фирм, входящих в состав DVD Forum, образовали Blu-ray Disc Consortium и приступили к разработке оптических дисков нового поколения, которые должны обладать значительно большей емкостью и обеспечивать гораздо более высокую скорость передачи данных. Актуальность этой проблемы связана с начинающимся вводом в эксплуатацию систем телевидения высокой четкости HDTV.

Осенью 2002 г. наметился путь к примирению враждующих лагерей DVD-разработчиков. О возврате к идее единого стандарта и речи не идет. Просто компании Sony и NEC выпустили первые мультиформатные DVD-приводы, поддерживающие одновременно и "минусовые", и "плюсовые" носители. Способ, конечно, не самый дешевый и не очень-то простой, но пользователи и ему рады. Все же выбор становится куда проще, если не надо на свой страх и риск принимать ответственное решение о том, какой из стандартов перспективнее.

Судить же об этом непросто, ведь работы над конкурирующими технологиями не прекращаются. В октябре 2003 г. на выставке Ceatek Japan 2003 компании Philips и Mitsubishi Chemical (известная по марке Verbatim) анонсировали свою новейшую разработку — двухслойные записываемые носители DVD+R. Благодаря второму слою их емкость увеличена с 4,7 до 8,5 Гб при сохранении совместимости с выпускающимися сегодня бытовыми плейерами и компьютерными дисководами. Остается дождаться появления рекордеров, поддерживающих такие носители. Конкуренты решили не отставать. Той же осенью компания Pioneer, один из наиболее активных участников DVD Forum, объявила, что разрабатывает двухслойные диски стандарта DVD-R.

В феврале 2003 г. началось лицензирование технологии Blu-ray Disc (BD). Судя по всему, уже в ближайшие год или два они станут выпускаться серийно. Однако надеждам пользователей на "мирное" будущее, похоже, сбыться не суждено: компании Toshiba и NEC при поддержке DVD Forum продвигают альтернативную разработку, которая получила название Advanced Optical Disc (AOD) и уже вот-вот будет закреплена окончательной спецификацией версии 1.0, одобренной DVD Forum.

Таким образом, в 2004 г., накануне десятилетнего юбилея DVD, мы снова имеем дело с двумя перспективными, но совершенно несовместимыми между собой стандартами High Definition (HD) DVD. Ситуация осложняется тем, что DVD Forum намерена отказаться от рассмотрения технологии Blu-ray, хотя над ней давно трудятся входящие в эту организацию компании. Последние же отнюдь не расположены принести в жертву единому стандарту свои немалые инвестиции в оптическую технологию завтрашнего дня.

Далее автор рассказывает о технических характеристиках разных технологий DVD, останавливается на коммерческой стороне вопроса. В заключение он отмечает, что с учетом исторического опыта, на основании анализа технологических решений и коммерческих интересов становится очевидно, что переход на технологии HD DVD не принесет избавления от противостояния двух несовместимых стандартов. И все же существует ли какая-нибудь сила, способная настоять на устранении противоречий? Представители компьютерной и программной индустрии уже не могут выступать в качестве третьей стороны, требующей принятия единого и технически совершенного стандарта, поскольку сами вовлечены в войну разных лагерей DVD. Более того, по мере продвижения собственных концепций "цифрового дома", также конкурирующих между собой и во многом зависящих от технологий оптических дисков, IT-компании все глубже будут увязать в борьбе за лицензионные отчисления.

Существует мнение, что независимым и авторитетным арбитром могла бы выступить признанная всемирная организация, например, ISO. Да, ее влияние велико, но она ведь не занимается сама разработкой технологий, а лишь наделяет некоторые из предложений статусом международных стандартов, соблюдение требований которых, кстати, не является обязательным условием для осуществления производства и продаж коммерческой продукции. Это дело компаний, как и каким правилам они считают нужным следовать.

Используя материнскую плату, описанную в статье *И.Рогожкина* "Jetway N2View: два компьютера в одном" **(Подводная лодка, №5/2004, С.25), можно сэкономить один системный блок**.

Изюминка системной платы — фирменная технология MagicTwin. Она принимает сигналы от двух клавиатур и мышей и обеспечивает разделение ресурсов системы между двумя станциями. По ходу установки программа MagicTwin напоминает, что пользователю нужно иметь две лицензии на Windows, и просит задать имена и пароли для входа в систему. Каждому пользователю предоставляются свои настройки "Рабочего стола", отдельные меню "Пуск", адресные книги и т.д. Им обоим автоматически выделяется своя мышь, клавиатура, выход графического адаптера и звуковая подсистема. Жесткие диски всегда доступны на обеих станциях, а накопители со сменными носителями (CD-ROM, DVD-ROM, флэш-диски и т.д.) можно распределять между виртуальными компьютерами.

При тестировании двухпользовательская система работала очень стабильно. На обоих виртуальных станциях выполнялись разные программы, даже ткие ресурсоемкие, как тест трехмерной графики 3Dmark03 и тест бизнес-приложений PC Magazine Business Winstone 2004, которые работали параллельно. На большинстве типовых офисных задач пользователи практически не ощущают друг друга.

Итак, система MagikTwin позволяет с минимальными затратами времени и ресурсов увеличить число рабочих мест за компьютером. Это, конечно, снижает надежность системы. Если, например, один пользователь "подвесит" компьютер, перезагружать систему придется обоим; если один испортит файловую систему на основном диске, выйдут из строя обе станции. Это следует учитывать при использовании системных плат с функцией MagikTwin.

Borland C++ Builder 6 для начинающих

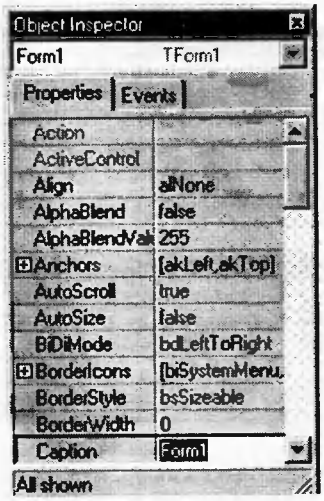
О.ВАЛЬПА,
E-mail: sandh@narod.ru

(Продолжение. Начало в №№7-8/2004)

Инспектор объектов

Инспектор объектов предназначен для редактирования свойств объектов (форм и визуальных компонентов) во время разработки программы. С его помощью можно установить начальные свойства объектов и назначить обрабатываемые события. Рассмотрим окно инспектора объектов (**рис.1**).

Рис. 1



В верхней части окна инспектора находится поле выбора объектов, в котором отображается название выбранного объекта. Для выбора нового объекта необходимо щелчком левой кнопки мыши по кнопке со стрелкой открыть выпадающий (раскрывающийся) список, затем выбрать необходимый объект из списка. После этого можно редактировать свойства выбранного объекта. Второй способ выбора объекта заключается в том, что производится щелчок левой кнопкой мыши по самому объекту. Третий способ состоит в том, что производится щелчок левой кнопкой мыши по названию объекта в окне просмотра дерева объектов **Object TreeView** (**рис.2**).

Рис. 2

Ниже поля выбора объектов находятся кнопки закладок свойств **Properties** и событий **Events**. На закладке **Properties** располагаются все не скрытые свойства выбранного объекта. Закладка состоит из двух полей. В левом поле отображается название свойств, а в правом — их значение. Свойства могут быть вложенными с обозначением в виде плюса в квадратике, расположенном левее названия свойства. Для их раскрытия и редактирования необходимо щелкнуть левой кнопкой мыши по этому значку, а затем — по самому свойству. Если свойств много, их можно просматривать с помощью кнопки прокрутки. Например, свойство **Caption** отражает заголовок объекта, а свойство **Name** — имя объекта.

На закладке **Events** располагаются события, на которые может реагировать объект. Эта закладка позволяет связать объект с событиями, происходящими в момент выполнения программы. Закладка **Events** также разделена на два поля. В левом поле отображается название событий, на которые может реагировать объект, а в правом поле — название методов их обработки, или проще — функций-подпрограмм обработки. Для создания обработчика необходимо выполнить двойной щелчок левой кнопкой мыши в правом поле, напротив названия выбранного события. При этом появится окно редактора кода, в котором автоматически создастся заготовка текста программы обработчика события. Между фигурными скобками заготовки разработчиком дописывается программа обработки события. В результате, при возникновении данного события для выбранного объекта будет выполняться данная программа. Например, событие **OnClick** означает щелчок левой кнопкой мыши по объекту.

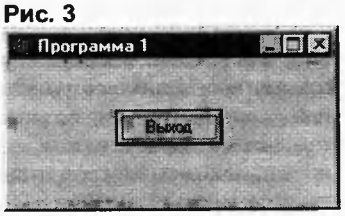
Рассмотрим пример простой программы. Пусть эта программа будет иметь всего одно окно, содержащее одну-единственную кнопку, щелчок по которой будет закрывать приложение. Запустите Borland C++ Builder 6 и поместите на форму **Form1** одну кнопку **Button1**. Уменьшите размеры формы и переместите кнопку в центр формы. Выберите в инспекторе объектов форму **Form1** и выделите ее свойство **Caption**. Замените слово **Form1** на слово **Программа 1**. Вы увидите, что заголовок самой формы тоже изменился. Теперь выберите в инспекторе объектов **Button1** и также измените его свойство **Caption** с **Button1** на **Выход**. Обратите внимание, что кнопка на форме стала называться **Выход**. Далее переключитесь в инспекторе объектов на закладку событий **Events** и дважды щелкните в поле напротив события **OnClick**. При этом откроется окно редактора кодов с заготовкой текста программы обработчика события. Впишите между фигурными скобками текста программы строку **Close();**. Это команда закрытия приложения. В результате, должен получиться код (текст) программы, приведенный ниже.

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
Close();
}
//---------------------------------------------------
```

Здесь вначале записан заголовок функции-подпрограммы обработки события щелчка **Click** по кнопке **Button1** на форме **Form1**. Тип **void** означает, что функция не возвращает параметры в основную программу. Между фигурными скобками могут записываться операторы, команды и функции. Последняя строка текста является комментарием-разделителем. Если вы допустите ошибки при написании команды, в инспекторе кода откроется окно сообщений, в котором будет выведен текст сообщения об ошибке. Для вызова справки о команде **Close()**, поместите курсор на текст этой команды и нажмите клавиши **Ctrl+F1**. Аналогично вы можете вызвать справку и по любым другим командам и операторам.

Теперь запустите программу, выполнив команду **Run** из главного меню, или из панели быстрых кнопок, или нажав клавишу **F9** на клавиатуре. После компиляции программа запустится, и появится ее окно, подобное тому, что приведено на **рис.3**.

Рис. 3



Щелкните левой кнопкой мышки по кнопке **Выход**, и приложение закроется. Программа работает и уже выполняет команды, пусть даже такие простые. Если в программе имеются ошибки, и она не будет закрываться, выполните команду **Program reset** из главного меню или нажмите клавиши **Ctrl+F9** на клавиатуре. После этого проверьте программу и, устранив ошибки, запустите снова.

Для доступа к дополнительным командам и настройкам инспектора объектов необходимо нажать правую кнопку мыши на окне инспектора. При этом на экране появится окно контекстного меню команд (**рис.4**).

В этом окне содержатся команды просмотра — **View** — свойств и событий по группам, размещения (упорядочивания) — **Arrange** — записей полей по категории или имени, размещения окна поверх всех других окон — **Stay on top**, скры-

тия окна — **Hide**, вызова справки — **Help**, изменения настроек — **Properties** и прочие. После щелчка левой кнопкой мыши по команде настроек **Properties** откроется окно настроек инспектора объектов (**рис.5**). Это же окно можно открыть, вызвав команду главного меню **Tools Environment Options** и затем открыв закладку **Object Inspector**.

В данном окне можно настроить цвета записей свойств и событий на полях инспектора объектов, а также осуществить выбор (**Options**) отображения некоторых

Рис. 4

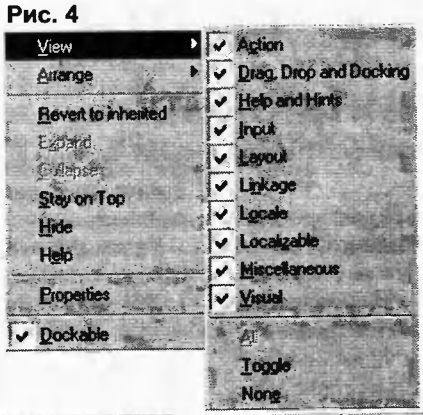
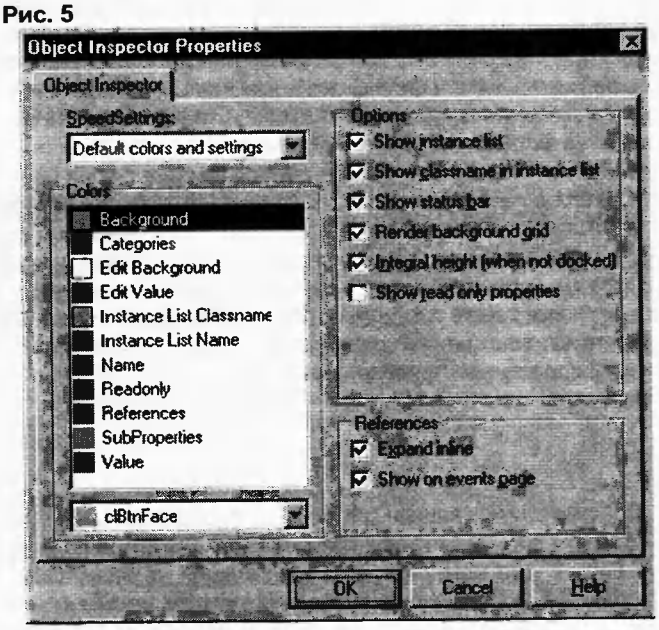


Рис. 5



свойств и событий, в том числе и тех, на которые имеются ссылки (**References**).

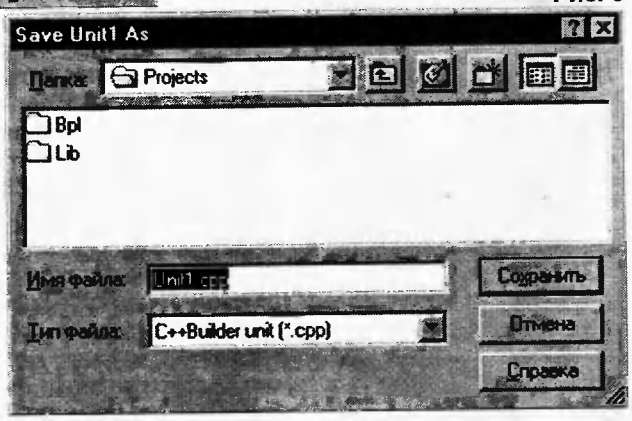
При изучении визуальных компонентов и разборе новых примеров программ мы еще не раз вернемся к инспектору объектов. А пока, для закрепления материала на практике, я предлагаю самостоятельно попробовать изменить некоторые свойства формы и кнопки и посмотреть, что при этом получается при выполнении программы.

Файлы проекта

Теперь пора подробнее познакомиться с файлами проекта и операциями над ними. При работе в среде разработки Borland C++ Builder 6 довольно часто приходится открывать и сохранять файлы. Все изменения в проекте перед запуском программы необходимо сохранять в файлах проекта, во избежание их потери в случае возникновения ошибок или зависания компьютера. При этом можно производить сохранение всего проекта целиком или сохранять отдельно необходимые файлы. Для сохранения всего проекта целиком вызовите из главного меню команду **Save Project As...** (Сохранить проект как...), или **Save All** (Сохранить все), или нажмите клавиши **Shift+Ctrl+S** на клавиатуре. На экране отобразится стандартное диалоговое окно сохранения файлов (**рис.6**).

В данном окне необходимо выбрать место расположения проекта в имеющихся каталогах или создать для этого отдельный каталог (папку). Последний вариант предпочтительнее, поскольку позволяет систематизировать все проекты и устраняет возможность наложения файлов с одинаковыми именами друг на друга. После этого следует последовательно произвести сохранение всех файлов проекта. Среда разработки предлагает по умолчанию свои имена для сохраняемых файлов. Например, для сохранения файлов с текстами программ предлагаются имена **Unit1.cpp**, **Unit2.cpp** и т.д. Для сохранения файлов проектов будут предлагаться имена **Project1.bpr**, **Project2.bpr** и т.д. Вы вправе изменять эти имена файлов на собственные названия. Это даже полезно сделать, поскольку по окончании разработки программы ее исполняемый файл с расширением **exe** будет иметь имя **Project1.exe**. Согласитесь, что это имя

Рис. 6



не очень подходит для названия любой программы. При переименовании файлов рекомендуется использовать только английские названия или названия, состоящие из латинских букв алфавита. Рассмотрим сохранение проекта на примере простой программы, описанной выше. Нажмите клавиши **Shift+Ctrl+S** на клавиатуре или соответствующий данной команде значок на панели быстрых кнопок. В открывшемся окне (рис.6) замените имя сохраняемого файла **Unit1.cpp** на **uprog1.cpp** и нажмите кнопку **Сохранить**. Далее замените предлагаемое для сохранения имя файла проекта **Project1.bpr** на **Prog1.bpr** и вновь нажмите кнопку **Сохранить**. Теперь запустите программу на выполнение с помощью команды **Run** из главного меню. После того как на экране появится окно программы, сверните его. Вы увидите, что в свернутом виде программа имеет название **Prog1**, а не **Project1**, как раньше. Таким образом, можно давать программе осмысленные имена.

Завершите работу приложения (программы). Теперь, если вы просмотрите содержимое каталога, в котором сохраняли файлы проекта, например, с помощью проводника (**Explorer**) из среды Windows, то обнаружите в этом каталоге множество файлов, в том числе и с именами, которые задали мы с вами. Ценными файлами в этой группе являются файлы текстов программ с расширением **cpp**, заголовочные файлы с расширениями **h** и **hpp**, файл установок формы проекта с расширением **dfm**, файл ресурсов с расширением **res** и файл проекта с расширением **bpr**, **bpg** или **bpk**. Файлы с расширениями **~bp**, **~df**, **~cp**, **~h**, **obj**, **tds** и **exe** создаются в момент компиляции проекта, и после удаления могут быть восстановлены в любое время. Наибольшим объемом обладает файл с расширением **tds**, предназначенный для отладки программы. Со временем количество проектов у вас будет расти, и для их хранения может понадобиться значительный объем на жестком диске. Для рационального использования места на диске рекомендуется не хранить вспомогательные файлы проекта, а обходиться хранением только кодов программ, файлов проектов, исполняемого файла и созданных вами файлов рисунков, иконок, фонов, курсоров, звуковых файлов и пр. Для об-

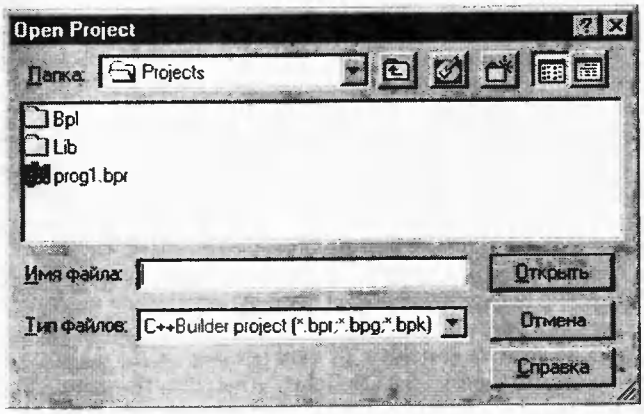
легчения данной задачи, я предлагаю вам использовать командный файл очистки проекта от всех вспомогательных файлов. Текст данного командного файла приведен ниже.

```
del .~
del *.obj
del *.tds
```

Как видите, он состоит всего из трех строчек с командами удаления вспомогательных файлов с некоторыми расширениями. Создайте с помощью любого текстового редактора файл с именем **bcbclr.bat** и занесите в него данные строки команд. После этого поместите данный файл в каталог с файлами проекта, закройте среду разработки Borland C++ Builder 6 и запустите файл **bcbclr.bat** на выполнение. В результате каталог будет очищен от лишних файлов, и тем самым освободится немало места на жестком диске.

Теперь рассмотрим, как можно открыть сохраненный проект для продолжения работы с ним или для его корректировки. Запустите среду разработки Borland C++ Builder 6. Вызовите из главного меню команду **Open Project** (Открыть проект), или нажмите клавиши **Ctrl+F11** на клавиатуре. На экране отобразится стандартное диалоговое окно открытия файлов (**рис.7**).

Рис. 7



В данном окне необходимо выбрать место расположения сохраненного проекта в имеющихся каталогах и выделить имя проекта. После этого следует нажать кнопку **Открыть**. Перед вами появится проект в том виде, в котором вы закончили с ним работу перед сохранением. Аналогично можно отдельно открыть любой файл проекта, используя команду **Open** главного меню.

Для закрытия всего проекта или файла без сохранения необходимо использовать команду **Close All** (Закрыть все) или **Close** главного меню соответственно.

Кроме того, интерфейс среды разработки Borland C++ Builder 6 запоминает пути хранения нескольких проектов и позволяет быстро загрузить их без поиска по каталогам с помощью команды **Reopen** главного меню.

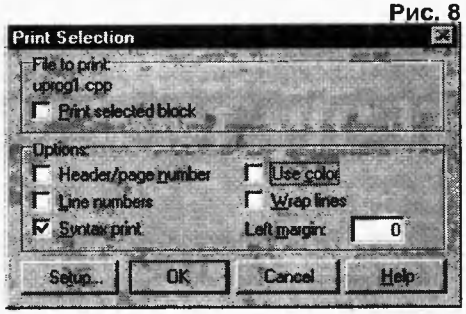
Для практики сохраните, а затем откройте проект описанными выше способами. Создайте файл очистки **bcbclr.bat** и запустите его для удаления вспомогательных файлов.

Вывод на принтер

При разработке программ вам может потребоваться распечатать какой-либо файл. Для этого необходимо предварительно выделить этот файл и выполнить настройки для печати. Выделите небольшой блок в файле инспектора кода и выполните команду **File Print** из главного меню. При этом откроется окно (**рис.8**), в котором производится настройка параметров печати.

В верхней части окна находится группа команд под названием **File to print,** в которой отображается имя выделенного файла для печати. Если в файле предварительно был выделен блок, то становится доступной возможность печати только данного блока путем установки флажка перед строкой **Print selected block**.

Рис. 8



Под группой выбора модулей для печати **File to print** располагается группа установок параметров печати специальных полей **Options**. С помощью установки флажка перед строкой **Header/Page numbers** можно обеспечить автоматическую нумерацию страниц текста при печати. Флажок, установленный перед строкой **Line numbers**, обеспечивает нумерацию строк печатаемого текста. Установка флажка перед строкой **Syntax print** позволяет печатать жирный шрифт, курсив, символы подчеркивания и другие синтаксические выделения шрифтом. Для использования цвета необходимо установить флажок перед строкой **Use color**. Для заворота строк, выходящих за пределы правой границы листа, необходимо установить флажок перед строкой **Wrap lines**. Поле **Left margin** позволяет при печати задать отступ слева.

Рис. 9

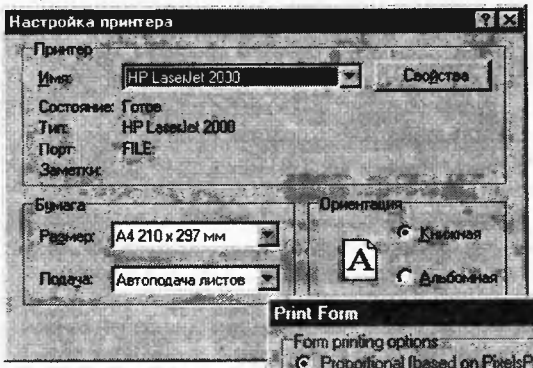
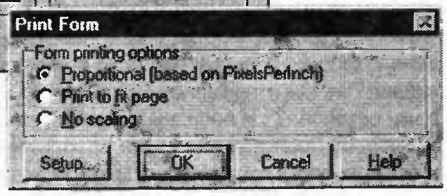


Рис. 10



Для установки дополнительных параметров печати необходимо нажать кнопку **Setup**. При этом откроется окно настройки принтера (**рис.9**). Здесь можно осуществить стандартные настройки принтера и формата бумаги для печати.

Кроме текстовых файлов, в среде разработки Borland C++ Builder 6 можно распечатать и графические файлы, например, форму приложения. Для этого необходимо открыть форму в инспекторе форм, тем самым выделив ее, и выполнить команду **File Print** из главного меню. При этом откроется окно, подобное тому, что приведено на **рис.10**.

Это диалоговое окно позволяет определить масштабирование формы при печати. Масштабирование зависит от размера бумаги принтера. Окно имеет три опции масштабирования. Опция **Proportional** позволяет осуществить пропорциональное масштабирование, т.е. распечатать форму в таком масштабе, в котором она видна на экране. Опция **Print To Fit Page** позволяет растянуть печатаемую форму на всю ширину страницы бумаги. Опция **No Scaling** никак не масштабирует форму и при печати использует настройки режима экрана. Если вы выберите эту опцию, форма может напечататься на нескольких страницах. При распечатке формы ее заголовок не печатается.

Попробуйте распечатать одну и ту же форму для трех различных установок масштабирования, и вы сразу же поймете, что это за режимы.

На веб-странице журнала в приложении к статье выложены файлы проекта рассмотренной в качестве примера программы.

(Продолжение следует)

ИССЛЕДОВАНИЕ ПРОГРАММ

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

ТЕОРЕТИЧЕСКИЕ ОСНОВЫ КРЭКИНГА

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

Если бряк оказался вдруг...

Наверное, вы уже попытались что-нибудь взломать. Может быть даже, вам это удалось — за счет знаний и способностей к анализу, благодаря интуиции, или же в силу вашего трудолюбия и настойчивости. Возможно также, что вам просто очень повезло с первой в жизни программой, и защита оказалась слабее, чем в большинстве других прогрвмм. Однако тех, кто не смог с первой попытки одержать победу над мегабайтами кода, гораздо больше. Кто-то споткнулся об антиотладочные приемы, кому-то "повезло" встретиться с запакованной программой, кто-то принял близко к сердцу огромнейшие возможности, предоставляемые OllyDebug и SoftIce, и погрузился в изучение этих инструментов, отложив до времени собственно копание в коде. Некоторые отступили, не добравшись до подходящей зацепки, с которой можно было бы начать "раскручивать" защиту. Свежие ощущения, новые знания, предвкушение будущих побед — все, что знаменовало рождение крэкера, осталось в светлом прошлом, куда вы сможете вернуться лишь в мечтах. В общем, одни радуются своей первой победе, другие переводят дыхание и с тоской глядят на заоблачные выси, которых не удалось достичь. Если вы попали в число "других", значит, у нас есть кое-что общее — свою первую программу я взломал далеко не с первой попытки. Надеюсь, после этих слов у вас появился повод для оптимизма — возможно, именно вам в будущем суждено написать свои собственные "Теоретические основы...". Однако сейчас вас, наверняка, больше интересует другой вопрос: "Почему мне не удалось сломать программу?". Причем нередко этот вопрос обретает еще более конкретную форму: "Почему я ставлю брейкпойнты, а они не срабатывают?" и "Как отлаживаемая программа может обнаружить мои точки останова?". И вопросы о неработающих (или "странно" работающих) брейкпойнтах — это отнюдь не повод упрекнуть в невнимательности начинающего крэкера, но основание для подробного разговора об особенностях Win32 API, тонкостях работы точек останова и антиотладочных приемах.

Брейкпойнты — лучшие друзья крэкера, готовые в любой момент прийти вам на помощь. Однако эти друзья отнюдь не всемогущи; как и живым людям, им присущи определенные слабости и врожденные особенности. И чтобы "поладить" с точками останова, нужно обладать знаниями об этих особенностях и слабостях — это, в конечном итоге, позволит вам при помощи нехитрых приемов отлавливать весьма замысловатые ситуации и успешно обходить защитные механизмы, направленные на "вырубание" брейкпойнтов. Но прежде чем приступать к познанию столь высоких сфер как внутреннее устройство и принципы функционирования точек останова, разберемся с куда более приземленными причинами возможной неработоспособности брейкпойнтов.

Самой простой (и, к сожалению, отнюдь не самой редкой) причиной такого поведения наших верных друзей являются ошибки в коде отладчиков. Да-да, вы не ослышались, крэкерам нередко приходится тратить часы на поиски несуществующих защит именно из-за недоработок в используемом инструментарии. "SoftIce не ставит бряк на функции", "Symbol loader не останавливает программу после загрузки" и другие подобные проблемы, с которыми сталкивался едва ли не каждый пользователь этого отладчика, уже который год отравляют жизнь крэкерам. При некотором упорстве и настойчивости эти проблемы иногда удается обойти разными "шаманскими" приемами, например, использованием аппаратного брейкпойнта вместо обычного, или указанием адресов в явном виде, но даже такие "танцы с бубном" не всегда оказываются эффективными против сущностей, скрывающихся по ту сторону отладчика. Никакие конкретные рекомендации тут, понятное дело, дать невозможно — программные глюки бесконечно разнообразны, и без точного знания причины с ними можно бороться разве что методом терпеливого перебора всех "обходных путей", какие только придут вам в голову. Если вас не прельщает сей метод — есть смысл поискать другую версию продукта (поскольку глюки, присущие одной версии программы, могут полностью отсутствовать в другой, пусть даже более старой), либо вообще подумать об обновлении инструментария.

Ненамного отстают по популярности среди авторов защит всевозможные приемы определения присутствия отладчиков, от откровенно примитивной проверки наличия определенных файлов/ключей реестра (отдельные разработчики защит даже удаляют эти ключи, нимало не утруждая свой беспросветно могучий интеллект мыслями о том, что SoftIce можно приобрести легально и использовать не для взлома их поделок) до довольно изощренных антиотладочных приемов, использующих особенности конкретных отладчиков. Примерами таких особенностей могут служить "черные ходы" в SoftIce для взаимодействия с Bounds Checker'ом или "нездоровая" реакция на вызов IsDebuggerPresent в OllyDebug и всех остальных отладчиках, использующих Debug API. Кстати, признаки наличия отладчика могут быть не только информационными, но и физическими — программа может "догадаться" о том, что ее ломают, по ненормально большому времени выполнения тех или иных процедур. Задумайтесь над тем, сколько времени уходит на выполнение десятка команд в "ручном режиме", когда вы исступленно давите кнопку F8 в OllyDebug, и вы сразу поймете, что я имею в виду. К этой же группе можно отнести использование в защитных механизмах отладочных регистров процессора — поскольку эти регистры используются отладчиком для установки брейкпойнтов, одновременная их эксплуатация программой и отладчиком невозможна. Если попытаться проделать такое, то либо отладчик "забудет" аппаратные точки останова, либо защитные процедуры выдадут некорректный результат со всеми вытекающими из этого последствиями. Большинство антиотладочных приемов, разумеется, давно и хорошо известны, и их описание несложно найти в руководствах по крэкингу и написанию защит. Впрочем, авторы защит на такие приемы обычно всерьез не рассчитывают, поскольку идентифицировать (а часто — и обойти) антиотладочный код в дизассемблерном листинге обычно несложно (например, если прикладная программа пытается оперировать с отладочными регистрами, это очевидный признак того, что в коде "что-то нечисто"), а некоторые крэкерские инструменты среди своих функций имеют отслеживание популярных антиотладочных приемов (примером может служить старая утилита FrogsIce, которая умела выявлять и побеждать множество защитных трюков, направленных против SoftIce).

Наиболее популярным среди начинающих крэкеров, по-видимому, еще долго будет оставаться вопрос: "Я поставил брейкпойнты на GetWindowText'ы и GetDlgItemText'ы, и все равно не могу поймать момент чтения серийника из окна". Действительно, формально все вроде бы сделано правильно, и все подходящие функции из "поминальника" обвешаны точками останова, как новогодняя елка — игрушками, но отладчик все равно не подает ни малейших признаков активности. При этом все точки останова находятся на своих местах и вполне успешно срабатывают — но, увы, не по тому поводу, который вам интересен.

В общем, у неопытного кодокопателя может сложиться впечатление, что серийный номер считывается при помощи телепатии или, как минимум, весьма недокументированным способом, чтобы обнаружить который, нужно иметь не меньше семи пядей во лбу. Однако в действительности никаких телепатических датчиков в вашем компьютере нет (а если даже и есть, то вряд ли они используются для чтения текста из диалоговых окон), да и подозревать недокументированные приемы я бы тоже не торопился, поскольку существует куда более простое объяснение этого явления. В Windows с давних пор сосуществуют два различных механизма, позволяющих управлять окнами и некоторыми другими объектами. Об одном из этих механизмов — системных вызовах Windows API — я уже говорил, и даже дал в предыдущей главе небольшой список наиболее употребительных функций с комментариями по поводу области их применения. Другая же сторона Windows до настоящего момента как-то оставалась в тени, за исключением эпизодических упоминаний "по поводу". Вы, наверное, уже догадались, что это за "другая сторона Windows" — я говорю о широко используемых в нашей любимой ОС сообщениях (хотя, если быть до конца точным, сообщения в том или ином виде присутствуют в большинстве современных операционных систем).

Если функции WinAPI безраздельно властвуют в темном и мрачном царстве невизуальных объектов, таких как файлы, процессы, средства синхронизации и прочее, то в "оконной" области ситуация отличается разительным образом. Сравнительно небольшой набор системных вызовов общего назначения ("создать-включить-удалить окно") с лихвой компенсируется огромным разнообразием системных сообщений (в англоязычной документации — "messages"; общее число документированных сообщений уж перевалило за тысячу), подчас дублирующих функции WinAPI (например, сообщение WM_GETTEXT, которое способно читать текст окна не хуже, чем уже известная вам функция GetWindowText). Некоторые типы управляющих элементов, такие как обычные или выпадающие списки, вообще не имеют полноценной "обвязки" функциями Win32 API, и управляются с ними именно при помощи сообщений. Вы не сможете добавить в такой список строчку или перейти к нужной позиции, вызвав WinAPI'шную функцию с названием, вроде ComboBoxAddString или ComboBoxSetPos — таких функций в системных библиотеках Windows просто нет. Зато есть сообщения CB_ADDSTRING и CB_SETCURSEL соответственно, воспользовавшись которыми, вы легко выполните задуманное. То есть, сообщения играют роль параллельного механизма управления объектами ОС, на работу которого совершенно не влияют традиционные брейкпойнты, которые мы щедрой рукой сеяли в предыдущей главе.

Поскольку сообщение — не функция, брейкпойнт на него поставить нельзя. Но очень хочется. А если очень хочется — значит, все-таки можно, хотя и не так просто, как хотелось бы. Прежде всего, вам нужно определиться, **что именно** вы хотите отловить — момент и точку отправки сообщения либо подпрограмму обработки этого сообщения. Если вас интересует второй вариант, и вы являетесь поклонником SoftIce — считайте, что о вас уже позаботилась фирма NuMega (ныне — Compuware). Встроенная в SoftIce команда BMSG как раз для этого и предназначена, но чтобы успешно ее использовать, вам понадобится узнать хэндл окна, которому предназначено сообщение. Если нужные данные у вас имеются — просто набирайте **BMSG <хэндл_окна> <код_сообщения>** и ждите, когда "всплывет" отладчик. Разумеется, команда BMSG, как и любая другая команда установки брейкпойнтов, позволяет создавать условные точки останова, срабатывающие, например, при поступлении сообщений только с определенными значениями wParam и lParam.

А что делать тем, кто пользуется другими отладчиками, в которых аналог BMSG отсутствует? Ответ на этот вопрос находится, как ни странно, именно в руководстве по SoftIce. В частности, там написано, что действие команды BMSG может быть воспроизведено установкой условного брейкпойнта на оконную процедуру, причем в качестве условия нужно указать следующее: **IF (esp->8)==<имя_сообщения>**. Адаптация этого условия под синтаксис, принятый в конкретном отладчике, обычно сложности не представляет, хотя вместо символьного имени сообщения, скорее всего, придется подставить его код (коды сообщений можно найти в файлах windows.inc, winnt.h или Messages.pas — в зависимости от того, компилятор какого языка у вас есть под рукой; те, кто не обзавелся подходящим компилятором и не планирует им обзаводиться в ближайшем будущем, могут заглянуть в файл messages.lst из состава InqSoft Window Scanner).

Такой подход несколько сложнее, чем вызвать команду BMSG с нужными параметрами, но зато он дает одно немаловажное преимущество. В предыдущем абзаце я сделал замечание насчет необходимости знать хэндл окна для успешного применения этой команды. Однако хэндл окна может быть вам известен далеко не во всех случаях. Рассмотрим, к примеру, ситуацию, когда вам нужно оттрассировать обработчик сообщения WM_INITDIALOG. Вы не сможете просто взять и посмотреть нужный вам хэндл, поскольку окно только-только создано и могло еще даже не появиться на экране. Конечно, можно "заморозить" все исследуемое приложение и затем предпринять поиск среди всех окон в системе, но не кажется ли вам, что это несколько сложнее, чем хотелось бы?

Кроме того, при каждом запуске программы хэндлы меняются, что тоже отнюдь не упрощает отладку. А вот оконная процедура всегда находится на одном и том же месте (справедливости ради надо отметить, что создание "плавающей" по адресному пространству от запуска к запуску процедуры теоретически возможно, хотя я такое и не встречал). И потому адрес этой процедуры можно просто записать на бумажке, чтобы затем восстанавливать соответствующий бряк без каких-либо сложностей. Нам осталось только раздобыть адрес этой самой процедуры. Тут тоже, в принципе, ничего сложного нет — разумеется, если под рукой имеются соответствующие инструменты (к примеру, Microsoft Spy++ или все тот же InqSoft Window Scanner). Наведите "прицел" программы на интересующее вас окно и прочитайте желанный адрес оконной процедуры собственно окна (обычно этот адрес обозначается как WndProc) или оконной процедуры, сопоставленной классу окна.

Иной путь получения адреса оконной процедуры заключается в том, чтобы при помощи API-шпионов обнаружить системный вызов, посредством которого производится регистрация класса окна (функции WinAPI RegisterClass и RegisterClassEx) либо непосредственно создание окна (список соответствующих функций я приводил в предыдущей главе). Операция эта выполняется в четыре этапа:

- запускаем под API-шпионом, настроенным на отслеживание процедур создания окон, и ждем появления нужного окна;
- как только окно появится — останавливаем работу шпиона и при помощи любого сканера окон получаем хэндл этого окна;
- если адрес оконной процедуры находится среди параметров функции создания окна — ищем в логе, сгенерированном API-шпионом, функцию, которая возвращает значение нашего хэндла, и считываем ее параметры, среди которых находим искомый адрес оконной процедуры;
- если адрес оконной процедуры находится в структуре, указатель на которую передается в функцию регистрации классов — считываем адрес, откуда был произведен вызов этой функции. Затем загружаем программу в отладчик, ставим точку останова на этот адрес (или чуть выше — на том месте, где происходит запись в стек указателя на структуру, это уж как вам больше понравится) и запускаем программу. Как только исполнение программы прервется на нашем брейкпойнте, находим в памяти структуру, указатель на которую передается в RegisterClass[Ex], и аккуратно переписываем содержимое поля этой структуры, содержащее адрес оконной процедуры для регистрируемого класса.

Вас может смутить сложность четвертого пункта, который выполняет весьма несложные функции, но при этом его описание едва ли не длиннее предыдущих трех. Казалось бы, что нам мешает просто вытащить из лога API-шпиона значение указателя на структуру, по-быстрому снять дамп нужной области и прочитать желанный адрес? В принципе, ничего не мешает — но вот истинное содержимое структуры WNDCLASSEX вы таким способом вряд ли прочитаете — потому что, скорее всего, в момент снятия дампа эта структура уже давно будет затерта другими данными. Дело в том, что регистрация класса — событие разовое, и потому память под структуру, описывающую класс, редко выделяют статически; обычно же программист обходится для этих целей куском стека. Так что когда вы заберетесь своим дампером в адресное пространство процесса, в том месте, где находилась желанная структура, давно уже будут лежать другие данные. И единственным решением в данном случае мог бы быть интеллектуальный API-шпион, которому можно было бы объяснить правила извлечения полей структур из памяти. К сожалению, на данный момент API-шпионы с такими свойствами мне не известны.

Другой распространенной причиной, по которой могут "не работать" брейкпойнты, является маскировка действий защиты под что-нибудь совершенно безобидное, или нетривиальная реализация защитных механизмов. В повседневной жизни мы очень часто руководствуемся правилом: "если что-то выглядит как утка и крякает как утка — значит, это и есть утка". Более того, данное правило — один из столпов того, что мы называем здравым смыслом. Однако вы наверняка замечали, что правило это — не без изъяна, и не так уж редко видимая картина мира отнюдь не соответствует истинной. В крэкинге это противоречие между видимым эффектом и скрытым от невооруженного глаза назначением защитного кода может быть доведено до предела, поскольку, взламывая программу, крэкер не просто копается в машинном коде, но ведет интеллектуальный поединок с автором защиты. И со стороны противника можно ожидать всего — блефа в виде процедур-"пустышек", имитирующих защиту; сверхсложных схем, решающих простейшие задачи; ловких имитаций, призванных повести крэкера по ложному пути, и, наконец, многоуровневой системы проверок, которые не слишком сложно реализуются, но достаточно долго и нудно обезвреживаются. При написании защит редко задаются вопросами оптимальности, скорости и расхода ресурсов — все эти добродетели программирования приносятся в жертву защищенности.

Я уже приводил пример того, как программа считывала дату своей установки под видом поиска плагинов в своей директории, и разумеется, этим список возможных приемов маскировки одних действий под другие не исчерпывается. Программа eXeScope, например, в качестве сообщения об ограничении в незарегистрированной версии выдает окно, внешним видом точь-в-точь повторяющее стандартный MessageBox, но в действительности нарисованное визуальными средствами в Delphi. Отображение файла в память вместо обычного чтения в буфер — прием известный, и тем не менее, чтение файла лицензии таким способом вполне может поставить в тупик начинающего крэкера. Я уж не говорю о таких изощренных техниках как парсинг ini-файлов "вручную" (после чего можно очень долго возиться с точками останова на GetPrivateProfile* — разумеется, с нулевым результатом) или экспорт кусков реестра при помощи утилиты regedit во временный файл с последующим анализом этого файла (что позволяет обойтись без вызова функций работы с реестром внутри программы).

Однако наиболее интересным для читателя, я думаю, будет рассмотрение причин, по которым точки останова просто исчезают из отлаживаемой программы. Я мог бы просто назвать причину таких мистических исчезновений и изложить типовой способ решения этой проблемы, но, думаю, вам будет гораздо интереснее понять причины, по которым "теряются" брейкпойнты. А уж теоретические знания помогут вам самостоятельно найти подходы к решению этой проблемы еще до того, как вы доберетесь до готовых рецептов. Очевидно, что прежде чем разбираться в защитных приемах, подавляющих точки останова, нужно сначала понять физический смысл этих самых точек, то есть узнать, что они собой представляют, как устанавливаются, и по каким признакам программа может догадаться об их наличии. А поскольку точки останова — изобретение отнюдь не новое, рассказ о них следует начать с исторического экскурса в седую древность.

(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

Технические основы

Технология Hyper-Threading (HT) впервые появилась у микропроцессора (МП) Xeon в феврале 2002 г., а затем в ноябре того же года она перекочевала и в семейство Pentium 4. Большинство последующих моделей Pentium 4 выпускается только с поддержкой HT, поэтому МП, не поддерживающие технологию HT, постепенно исчезнут с компьютерного рынка.

В технической литературе говорится о том, что при поддержке HT процессор исполняет две цепочки инструкций ***одновременно*** (simultaneously), вместе с тем, количество исполнительных устройств МП (Execution Units) и выполняемые ими функции не изменились. В данной статье мы попробуем разобраться, что на практике означает и за счет чего достигается одновременность исполнения двух цепочек (threads) инструкций.

Работа процессора в обычном режиме

Технической основой для внедрения HT-технологии явилась внутренняя архитектура МП седьмого поколения, которая называется NetBurst (пакетно-сетевая). Напомним ее основные особенности.

Все узлы или модули микропроцессора образуют один большой (20-стадийный) конвейер. За один такт он способен обрабатывать несколько инструкций, находящихся на разной стадии исполнения. В технической документации при описании конвейера выделяют три части, отражающие основные этапы обработки инструкций.

Фронтальный отдел (Front End). Здесь производится выборка инструкций, их декодирование, т.е. преобразование в последовательность микроопераций (uops или (ops) и размещение в кеш трассировки (Trace Cache). Если инструкция порождает более 4-х микроопераций, то в ее декодировании участвует Microcode ROM, содержащая необходимые последовательности микроопераций. В итоге формируется очередь микроопераций (Uop Queue), готовых для обработки центральной частью конвейера.

Во фронтальном отделе находится блок предсказания ветвлений (branch prediction). Он формирует адрес инструкции, которая будет выполняться после условного ветвления.

В дальнейшем, после исполнения этой инструкции, сравниваются реальный и предсказанный адреса. В случае их расхождения (branch mispredictions) приходится очищать конвейер от частично или полностью исполненных ненужных инструкций, загружать и обрабатывать нужные. Ошибки прогноза существенно снижают производительность МП.

Исполнительное ядро (execution core) является центральной частью конвейера и может выполнять микрооперации, игнорируя их естественный порядок (out-of-order). Из очереди Uop Queue микрооперации попадают в распределитель (allocator), выделяющий ресурсы, необходимые для их выполнения — регистры и буферы различного назначения. Одновременно производится переименование встречающихся в инструкциях регистров общего назначения и заполнение специальной таблицы псевдонимов регистров (register alias table), которая используется в последней части конвейера. Затем планировщики (Schedulers) определяют состояние микроопераций, и готовые к исполнению передают соответствующим исполнительным устройствам, если они свободны. Планировщики не учитывают исходную последовательность микроопераций. В некоторых случаях это позволяет исключить холостые циклы в работе исполнительных устройств и тем самым повысить производительность МП.

Замечание: Неупорядоченное исполнение инструкций впервые появилось у МП Pentium II. Это нововведение избавило программистов от необходимости заботиться о спаривании применяемых инструкций, которое требовалось для использования специфических особенностей конвейера предшествующих моделей Pentium. Спариваемые инструкции могли выполняться одновременно двумя разными трубами (U и V) конвейера, что повышало производительность МП.

Устройство упорядочения (in-order retirement unit) выполняет заключительные действия. Микрооперации, исполненные центральной частью конвейера, накапливаются в специальном буфере (reorder buffer) и группируются в последовательности, соответствующие исходным инструкциям, затем по таблице псевдонимов восстанавливаются имена регистров. Если при выполнении микроопераций возникали исключительные (аварийные) ситуации, то восстанавливается их естественная последовательность. При обнаружении инструкции ветвления в блок предсказания посылается истинный адрес перехода. Это необходимо для проверки правильности ранее сделанного прогноза.

Изменения в микроархитектуре процессора

При внедрении технологии HT, в микроархитектуру МП были внесены изменения, позволяющие одному физическому процессору выполнять функции двух логических процессоров. Основные изменения коснулись фронтальной части конвейера, в центральной части изменилась только работа распределителя ресурсов. Изменения в заключительной части конвейера вызваны тем, что теперь надо восстанавливать не только исходные последовательности микроопераций, но и их принадлежность к двум различным цепочкам инструкций.

Для того чтобы один физический процессор мог одновременно обрабатывать две независимые цепочки инструкций, разработчики продублировали некоторые его ресурсы. К ним относятся ***все виды регистров***, которые могут явно или неявно использоваться в системных и прикладных программах — общего назначения, сегментные, управляющие, отладочные, x86 (FPU), MMX, XMM, указатель инструкций, регистр признаков (EFLAGS), программируемый контроллер прерываний (APIC), указатели системных таблиц и пр. Кроме того, продублированы скрытые от программиста ресурсы МП, участвующие в формировании и обработке адресов инструкций (но не данных), например, специальный буфер ITLB, обеспечивающий выборку инструкций, и блок предсказания ветвлений.

Перечисленные выше ресурсы МП в документации называются повторяемыми (replicated). По утверждению разработчиков, они составляют небольшую часть от всех структур данных МП. Кроме того, появились еще поделенные пополам (partitioned — расчлененный) и общедоступные (shared — разделяемые) ресурсы.

Большинство ресурсов МП являются общедоступными, но только некоторые из них могут явиться причиной конфликтов при исполнении двух цепочек инструкций. Мы вернемся к этому вопросу во второй части статьи, посвященной программированию.

Разделенные пополам ресурсы некоторые авторы называют "динамически разделяемыми", потому что деление существует ***только*** при исполнении двух цепочек инструкций. При работе в однозадачном режиме такие ресурсы полностью отданы одной цепочке.

Деление пополам, прежде всего, направлено на то, чтобы ни одна из двух цепочек не могла использовать больше половины ресурсов, необходимых для исполнения микроопераций — регистров для размещения целочисленных и вещественных операндов и буферов для их загрузки и сохранения. Напомним, что указанные ресурсы выделяет распределитель.

Также пополам разделены буферы для хранения очередей, содержащих результаты, полученные на основных стадиях работы конвейера. Это позволяет продолжать исполнение одной цепочки, когда исполнение другой приостановлено по тем или иным причинам. А при ошибке предсказания ветвления будет очищена только та половина очередей, которая в этом нуждается.

Обработка двух цепочек инструкций

Во фронтальной части конвейера осталось два общедоступных устройства — декодер инструкций и Microcode ROM. Если активны две цепочки, то они обслуживаются поочередно, через такт. Декодируемые инструкции находятся в специальных 128-разрядных потоковых регистрах, которые продублированы, что и позволяет декодеру поочередно обрабатывать их содержимое. В декодировании сложных инструкций участвует Microcode ROM, содержащая соответствующие цепочки микроопераций.

Замечание. Некоторые авторы считают, что декодер содержит 15 ступеней (во всем конвейере их 20!), и декодирование инструкции продолжается 15 тактов. Как совместить это утверждение со следующей фразой из документации: "Во фронтальной части находится один декодер, способный декодировать инструкции с максимальной скоростью одна инструкция за такт"?

При нормальном протекании вычислительного процесса в разделенной пополам очереди Uop Queue находятся микрооперации, порожденные двумя независимыми цепочками инструкций. В центральной части конвейера распределитель также поочередно, через такт, обрабатывает микрооперации, относящиеся к разным цепочкам.

В результате на входы планировщиков поступает смесь микроопераций, относящихся к двум разным цепочкам.

Планировщики оценивают степень их готовности независимо от принадлежности к конкретным цепочкам и исходной последовательности и передают готовые микрооперации требуемому устройству, если оно свободно.

Таким образом, ***главная особенность*** технологии HT заключается в том, что на входы планировщиков поступает смесь микроопераций, порожденных двумя независимыми цепочками инструкций.

Вычислительные ресурсы микропроцессора

Для того чтобы понять возможные источники повышения производительности МП, рассмотрим состав и функции его исполнительных устройств. В разных документах приводится рисунок, на котором изображено 7 устройств, но фактически одно из них, а именно FPU1, является группой исполнительных устройств. Судя по всему, это модифицированный вариант того, что раньше называлось встроенным математическим сопроцессором.

Готовность микрооперации к исполнению означает, что операнды находятся в закрепленных за ними регистрах МП. Поэтому существуют два специальных устройства, выполняющие загрузку (считывание из памяти в регистр) и сохранение (запись из регистра в память) операндов. В большинстве случаев под словом "память" подразумевается кеш, однако существуют специальные инструкции (в группах SSE), которые позволяют работать непосредственно с ОЗУ, минуя кеш.

Итак, рассмотрим вычислительные ресурсы МП.

Memory Store — исполняет микрооперации записи содержимого регистра в память. Они порождаются инструкциями, у которых приемник (первый операнд) находится в памяти, или обращение к памяти требуется, но явно не указано, например, в инструкциях CALL, STOS, PUSH. Подключено к индивидуальному порту 3.

Memory Load — исполняет микрооперации чтения из памяти в регистр. Они порождаются инструкциями, у которых источник (второй операнд) или приемник находятся в памяти, или обращение к памяти требуется, но явно не указано, например, в инструкциях RET, SCAS, POP. Подключено к индивидуальному порту 2.

ALU0 — быстрое арифметическое устройство. Исполняет микрооперации, порождаемые инструкциями ADD, SUB, AND, OR, XOR, CMP, TEST, MOV, MOVSB, MOVSW, MOVZB, MOVZW, NEG, NOT, NOP, SAHF. Разделяет порт 0 вместе с устройством FPU0.

ALU1 — быстрое арифметическое устройство. В отличие от ALU0, исполняет только микрооперации сложения и вычитания. Разделяет порт 1 вместе с устройствами ALU2 и FPU1.

ALU2 — работает с обычной скоростью (одна операция за такт). Используется, например, для выполнения арифметических и логических сдвигов. Разделяет порт 1 вместе с устройствами ALU1 и FPU1.

FPU0 — выполняет простые (не требующие вычислений) операции с вещественными числами — пересылки, перестановки и т.п. Разделяет порт 0 вместе с устройством ALU0.

FPU1 — группа устройств, выполняющих вычислительные "микрооперации" (скорее, макрооперации), порожденные инструкциями FPU, MMX и SSE. Разделяют порт 1 вместе с устройствами ALU1 и ALU2:

- **FP_ADD** — сложение, вычитание, сравнение, нахождение минимума и максимума и т.п. для одиночных (FPU) и упакованных (SSE) вещественных операндов;
- **FP_MUL** — умножение одиночных (FPU) и упакованных (SSE) вещественных операндов;
- **FP_DIV** — деление и извлечение квадратного корня одиночных (FPU) и упакованных (SSE) вещественных операндов;
- **MMX_ALU** — все арифметические операции с упакованными целочисленными операндами MMX и SSE (напомню, что операции деления таких чисел не существует);
- **MMX_MISC** — вычисление обратных величин (группа SSE1) и некоторые операции с упакованными целыми числами;
- **MMX_SHFT** — все разновидности сдвигов и перемещений отдельных частей упакованных операндов (целочисленных и вещественных);
- **FP_MISC** — похоже еще на одну группу устройств, состав и назначения которых не описаны.

Замечание. Условные ветвления относятся к числу быстрых операций, исполняемых ALU0 за половину такта, поэтому сочетание инструкций, осуществляющих выработку признака и ветвление, может быть выполнено за один такт. Но если при последующей проверке окажется, что предсказанный ранее адрес перехода был ошибочным, то на исправление ошибки будет потрачено много тактов работы конвейера.

Из приведенного описания видно, что все исполнительные устройства подключены к 4 портам. За один такт работы МП они могут передать до шести микроопераций. Это происходит в том случае, когда в начале такта передаются 4 микрооперации и в середине такта — еще две быстрые микрооперации для устройств ALU0 и ALU1. Трудно сказать, как часто возникают такие случаи на практике, но условия для их реализации есть.

Возможности повышения производительности микропроцессора

При внедрении технологии HT количество исполнительных устройств и выполняемые ими функции не изменились. За счет чего в таком случае может быть повышена производительность МП при исполнении двух независимых цепочек инструкций? За счет того, что микрооперации одной цепочки выполняются во время пауз, неизбежно возникающих при выполнении другой цепочки. Микрооперация готова к исполнению, если операнды находятся в закрепленных за ней регистрах. Этому могут мешать два фактора: операнд находится в памяти, или он является результатом выполнения предшествующей операции.

При выполнении вычислительных операций может потребоваться от 0 до 2 обращений к памяти. Например, инструкция add result, ebx будет декодирована в одну быструю микрооперацию сложения и две микрооперации для чтения операнда и записи результата. При исполнении инструкций общего назначения (группа P6) обращение к памяти занимает 2 такта, а при исполнении инструкций групп FPU, MMX и SSE — 6 тактов. Эти цифры справедливы при условии, что операнд находится в кеше. В случае кеш-промаха вынужденная пауза может существенно затянуться. Таким образом, суммарные непроизводительные затраты времени, связанные с ожиданием завершения обращений к памяти, достаточно велики.

Если МП работает в обычном (однозадачном) режиме, то неупорядоченному исполнению микроопераций препятствует их взаимозависимость. А при одновременной обработке смеси микроопераций, порожденных двумя независимыми цепочками инструкций, вероятность того, что

Плагин-эмулирующий отладчик для дизассемблера IDA

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

Каждой твари по паре.
Каждой виртуальной машине — по виртуальному процессору.

Вступление

Как-то в Интернете наткнулся я на отрывок из книги Криса Касперски. Цитата:

"!IDA DEBUGGER! В чем новизна и удобство идеи? А в том, что можно сделать не полный, а _контекстный_ отладчик! Что это такое? Обычный дебагер отлаживает всю программу целиком. Это хорошо, но чаще всего нас интересуют только выбранные фрагменты. В IDA можно подогнать курсор к нужному месту, задать начальные значения регистров и пустить на выполнение эмулятор. Такой подход прежде всего упрощает задачу, т.к. тут скорость не требуется. А как удобно! Можно видеть работу кода в динамике! И не ломать голову, какие будут значения регистров\флагов на выходе из такого и такого фрагмента, и куда метнется условный переход. Можно просто прогнать чисто локальный кусок с любой его точки."

Я понимаю, что отладчик для IDA, начиная с версии 4.5, существует. Но, во-первых, не у всех она есть, во-вторых, создать отладчик своими руками — очень даже интересное дело, это поможет поближе познакомиться с устройством IDA и ее плагинов. Информация по написанию плагинов для дизассемблера есть в примерах SDK.

Сразу надо сказать, что это будет контекстный эмулирующий отладчик 32-разрядного режима. Он будет тесно взаимодействовать с пользователем. Например, если эмуляция какой-то команды отсутствует, то предполагается, что пользователь сам проэмулирует ее и скорректирует значения регистров и стека.

Предлагаю свой проект как один из вариантов такого плагина. Я его писал из желания поглубже узнать IDA. Если этот плагин "довести до ума", то может получится очень полезный program product, который поможет при ручной распаковке и анализе программ.

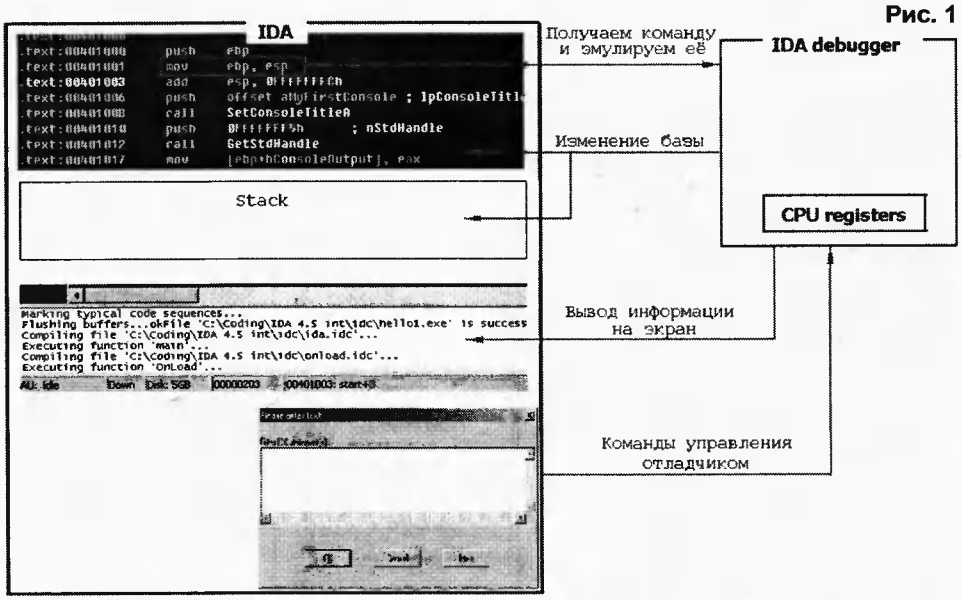
Общее построение

Лучше всего все показать на рисунке. В принципе, такое построение подойдет не только для отладчика, но и для любого другого плагина (**рис.1**).

Чтобы можно было корректно эмулировать, надо дополнительно создать сегмент стека в виртуальном пространстве дизассемблера — это необходимо для работы команд со стеком. Как его создать, будет показано ниже.

Управление ведется из консоли дизассемблера — как вызов функций. Также можно организовать "горячие" клавиши. Вывод информации на экран — посредством окна сообщений и информативных окошек.

Рис. 1



в нужный момент будут существовать готовые к исполнению микрооперации, существенно возрастает.

Кроме того, при специальном подборе одновременно исполняемых приложений можно сделать так, что цепочки микроопераций будут использовать разные исполнительные устройства. Например, в одном приложении — инструкции общего назначения (группа P6), а в другом — инструкции FPU или SSE, работающие с вещественными операндами, или инструкции MMX, работающие с упакованными целыми числами. В таких случаях возможна оптимальная загрузка всех исполнительных устройств МП.

Еще одна категория вынужденных пауз в исполнении цепочки связана с ошибками прогнозов ветвлений. В таких случаях продолжительное время могут обрабатываться микрооперации одной цепочки, а это не лучший способ загрузки ресурсов МП. В инструкции говорится, что вероятность правильного прогноза весьма велика (более 90%), но лучше не проверять статистику, а применять условные ветвления тогда, когда им нет альтернативы. Например, если с помощью ветвления выбирается одно из двух чисел или кодов, то лучше использовать специальные инструкции из группы CMOV. Они сочетают проверку признака и пересылку операнда. Если последний находится в регистре, то инструкция выполняется за один такт. Но главное — при этом исключается работа блока прогнозирования ветвлений.

Таким образом, технология HT только создает технические возможности для более рационального использования ресурсов МП, а как эти возможности будут использованы, зависит от одновременно исполняемых приложений. О подборе и программировании приложений мы поговорим во второй части статьи.

(Окончание следует)

Самое интересное — эмуляция команд. После подвода курсора к нужному адресу и ввода начальных значений в регистры, начинается работа отладчика.

В качестве "подопытной" была взята IDA 4.5 internal (лежала на www.crackbest.com). Также необходим SDK. Без SDK в создании плагинов делать нечего.

Команды управления отладчиком были введены при помощи IDC-скрипта — **"deb_comm.idc"**. Этот скрипт компилируется во время инициализации плагина (скрипт должен лежать в директории **idc** дизассемблера), в случае неудачи можно попытаться загрузить скрипт вручную из меню **File\IDC file**. Именно в этом скрипте находятся вызовы функций отладчика, здесь можно повесить на функции "горячие" клавиши и добавить новые. Формат вызова команды отладчика:

```
#define MY_FUNC_ID xx

static MY_NEW_COMMAND()
{
        RunPlugin("IDA_deb",MY_FUNC_ID);
}
```

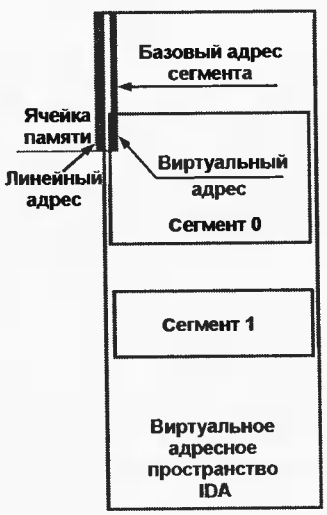
После добавления в "deb_comm.idc" команду можно будет вызывать прямо из консоли. Например, так — MY_NEW_COMMAND();. Только не забудьте предварительно вставить в функцию run() плагина обработчик этой команды, она будет обрабатываться, если аргумент будет равен MY_FUNC_ID.

Немного о виртуальной памяти IDA

Для написания отладчика надо иметь некоторые знания о виртуальной памяти, подробнее смотрите в [1].

Прошу извинить меня за такое представление виртуального адресного пространства IDA и ее базы данных. Все это упрощено, но отладчик работает именно с данными в виртуальной памяти, в которой лежит база данных. На **рис.2** показаны соотношения между адресами (названия взяты от самого создателя IDA).

Рис. 2



Линейный адрес (тип ea_t) — это адрес в виртуальном пространстве IDA.

Базовый адрес сегмента — это линейный адрес начала сегмента в виртуальном пространстве IDA.

Виртуальный адрес — это смещение между линейным адресом и базовым адресом сегмента:

Виртуальный адрес = Линейный адрес - Базовый адрес сегмента.

Как вы уже, наверно, знаете, IDA работает со своим виртуальным пространством, а для дизассемблированной программы создаются сегменты со своими атрибутами (базой, селектором, разрядностью и т.п.). С ним же будет работать и наш отладчик, через функции get_byte(..),get_word(), get_long, patch_byte, patch_word, patch_long — все они определены в BYTES.HPP, там есть еще много интересного.

Линейный адрес вычисляется как (***Base<<4+смещение***), где ***Base*** — база сегмента.

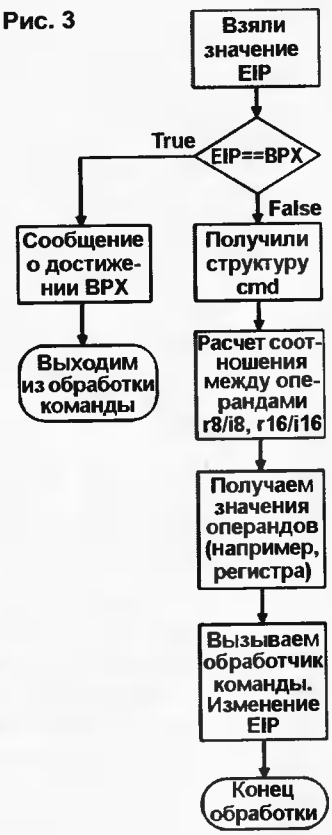
Как эмулировать?

Полагаю, для этого нужно узнать, как работает то, что мы будем эмулировать, т.е. центральный процессор. Если заглянуть в талмуды Intel, то можно найти примерно следующую последовательность действий по исполнению инструкций:

```
Like the Intel486 CPU, integer instructions traverse a 5 stage
pipe-line. The pipe-line stages
are as follows:

PF Prefetch
D1 Instruction Decode
D2 Address Generate
EX Execute - ALU and Cache Access
WB Writeback
```

Рис. 3



Т.е:

- предвыборка команды;
- декодирование команды;
- генерация адреса для определения адреса операндов в памяти;
- выполнение операций с помощью АЛУ;
- запись результата.

Придержимся этой последовательности и мы. Посмотрите на вариант эмулятора инструкций (**рис.3**). Предвыборку, дешифрацию команд уже сделала IDA, нам остается только разобраться в операндах и вызвать обработчик команды.

Чтобы приступить к непосредственной эмуляции, давайте разберемся, как IDA хранит инструкции. Для этого имеется несколько структур, они описаны в файле **INCLUDE\UA.hpp**. Этот файл *очень* важен при написании эмулятора, т.к. в нем описывается формат данных команды, ее операндов, и их типы.

```
// Структурка хранит информацию о команде

idaman insn_t ida_export_data cmd;      // Текущая инструкция

// Класс, описывающий инструкцию (сокращено)
class insn_t
{
public:

  ushort itype;  // Код инструкции (инициализируется в 0 ядром
                 // IDA), все инструкции содержатся в директории
                 // INCLUDE\allins.hpp

  ulong cs;      // База сегмента, в котором находится
                 // инструкция (инициализируется ядром IDA)

  ulong ip;      // Виртуальный адрес инструкции (адрес ВНУТРИ
                 // сегмента). От этих виртуальных адресов
                 // уже сам начинаешь виртуалиться
                 //(УСТАНАВЛИВАЕТСЯ ядром ИДА, а кем же еще?)

  ea_t ea;       // Линейный адрес инструкции

  ushort size;   // Размер инструкции в байтах
```

```
// Сведения об операндах инструкции

#define UA_MAXOP 6
  op_t Operands[UA_MAXOP]; // Массив содержит структуру,
                           // описывающую каждый операнд в
                           // команде. К нему удобно обратиться
#define Op1 Operands[0]    // например так: op_t x= cmd.Op1 -
#define Op2 Operands[1]    // получаем информацию о первом
#define Op3 Operands[2]    // операнде (странно, что не о нулевом)
#define Op4 Operands[3]
#define Op5 Operands[4]
#define Op6 Operands[5]

};

// Класс-тип операнда, (показано не полностью)
class op_t
{
public:
  char n;                  // Порядковый номер операнда

// Структура, описывающая тип операнда (очень интересное поле)
  optype_t        type;

// Тип значения операнда

  char          dtyp;

#define dt_byte         0   // 8 bit
#define dt_word         1   // 16 bit
#define dt_dword        2   // 32 bit
#define dt_float        3   // 4 byte
#define dt_double       4   // 8 byte
#define dt_tbyte        5   // variable size (ph.tbyte_size)
#define dt_packreal     6   // packed real format for mc68040
#define dt_qword        7   // 64 bit
#define dt_byte16       8   // 128 bit
#define dt_code         9   // ptr to code (not used?)
#define dt_void         10  // none
#define dt_fword        11  // 48 bit
#define dt_bitfild      12  // bit field (mc680x0)
#define dt_string       13  // pointer to asciiz string
#define dt_unicode      14  // pointer to unicode string

};

// Ну и, наконец, описание типа операнда
typedef uchar optype_t;
const optype_t                                        Поле в op_t
  o_void     =  0,  // нет операнда                    ------
  o_reg      =  1,  // Основной регистр (al,ax,es,ds...)   reg
  o_mem      =  2,  // Прямое обращение к памяти          addr
  o_phrase   =  3,  // Memory Ref [Base Reg + Index Reg]  phrase
  o_displ    =  4,  // Memory Reg [Base Reg + Index Reg +
                    // Displacement]    phrase+addr
  o_imm      =  5,  // Непосредственное значение          value
  o_far      =  6,  // Непосредственный дальний адрес     addr
  o_near     =  7,  // Непосредственный ближний адрес     addr
  o_idpspec0 =  8,  // IDP specific type
  o_idpspec1 =  9,  // IDP specific type
  o_idpspec2 = 10,  // IDP specific type
  o_idpspec3 = 11,  // IDP specific type
  o_idpspec4 = 12,  // IDP specific type
  o_idpspec5 = 13,  // IDP specific type
  o_last     = 14;  // first unused type

  char          specflag1;
  char          specflag2;
  char          specflag3;
  char          specflag4;
}
```

В структуре `optype_t` есть следующие поля:

```
  char          specflag1;
  char          specflag2;
  char          specflag3;
  char          specflag4;
```

Эти флаги заполняются ядром IDA (точнее, процессорным модулем). Как заполняются — не сказано, но, проведя эксперименты с различными инструкциями и способами адресации, удалось выяснить вот что.

Флаг specflag1 и все остальные флаги равны нулю, если имеется:

- прямая адресация (OpX.type=o_mem). Т.е., например, Mov eax,[401000];
- косвенная адресация (OpX.type=o_phrase) — mov eax,[eax];
- косвенная адресация со сдвигом (OpX.type=o_displ) — mov eax,[eax+402031].

Если specflag1==1, то в specflag2 содержится информация о составляющих операнда (например, при индексной адресации с масштабированием).

Табл. 1

7	6	5	4	3	2	1	0
Коэффициент масштабирования		Индексный регистр			Базовый регистр		

Табл. 2

EAX	0
ECX	1
EDX	2
EBX	3
ESP	4
EBP	5
ESI	6
EDI	7

Если OpX.type=o_phrase, то specflag2 будет расшифровываться в соответствии с **табл.1**.

Кодировка регистров — в точности как принята у Intel (**табл.2**).

Коэффициент масштабирования определяет степень двойки, на которую нужно умножить значение индексного регистра (**табл.3**).

Например:

```
lea     eax, [ebx+ecx*4]

Op2 type is: 3
Op2 dtyp is: 0
Op2 value is: 0
Op2 reg is: 4
Op2 phrase is: 4
Op2 address is: 0
Op2 offb is: 0
Op2 offo is: 0
Op2 specflag1 is: 1
Op2 specflag2 is: 9B
Op1 specflag3 is: 0

Op2 specflag2 is: 9B  10001011
10-4
011-ebx
001-ecx
```

Табл. 3

Коэффициент масштабирования	Коэффициент умножения
00	1
01	2
10	4
11	8

Если OpX.type=o_mem, то specflag2 будет расшифровываться в соответствии с **табл.4**.

Табл. 4

7	6	5	4	3	2	1	0
					1	0	1
Коэффициент масштабирования		Регистр			Точно не выяснил, но всегда равно константе		

Если OpX.type=o_displ, то specflag2 будет расшифровываться в соответствии с **табл.5**.

Табл. 5

7	6	5	4	3	2	1	0
Коэффициент масштабирования		Индексный регистр			Базовый регистр		

Также в этом случае в поле OpX.addr будет находиться значение смещения.

Однако здесь есть одно исключение для обращения к памяти, вида: **`[esp+xxx]`**.

В этом случае **`specflag1=1; specflag2=0x24`**

(Окончание следует)

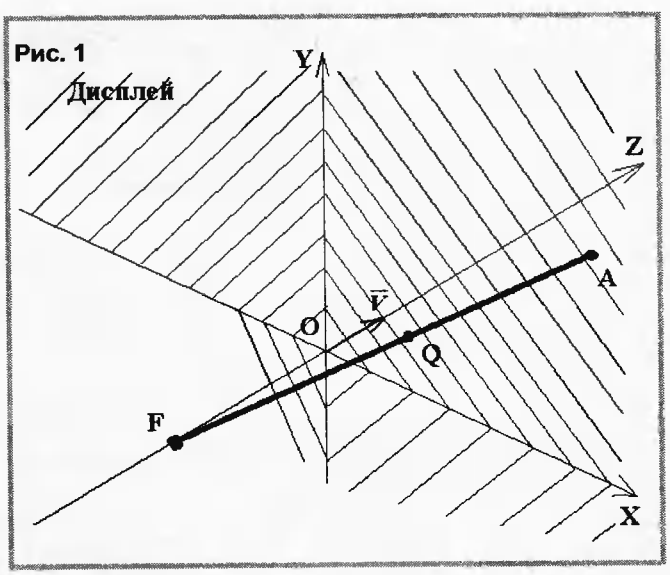
А.СЛОМИНСКИЙ,
г.Минск, ФИТУ, БГУИР,
E-mail: Slominsky_ag@tut.by

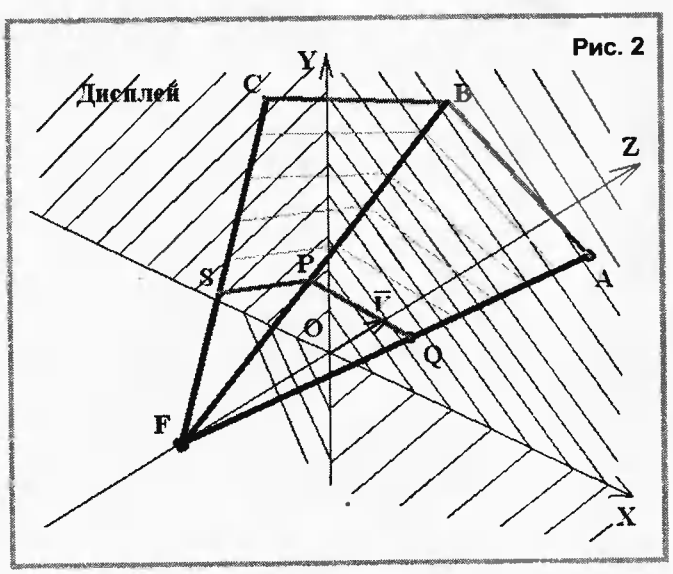
Построение поверхностей в трехмерном изображении методом центрального проецирования

В статье [1] были описаны алгоритм и работа функции анализа и вычисления алгебраических выражений на примере функции z=SIN((ox^2+oy^2)^0.5), которая описывает волну в трехмерном пространстве. Также прилагался графический файл с ее изображением. В этой статье будет описана та часть программы, которая отвечает за проецирование функции от двух переменных на плоскость экрана дисплея.

Описание метода центрального проецирования

По существу, нам необходимо перейти от трехмерного изображения в пространстве к двумерному изображению на дисплее. Сначала рассмотрим этот метод на примере самой простой геометрической фигуры — точки. Поверхность строится в виде сетки, узлы которой представляются самими точками. На **рис.1** показано построение для одной точки, на **рис.2** — для нескольких точек.





У нас есть точка **A** и фокус (точка **F**). Координаты **x** и **y** точки **A** задаются программно от **-100** до **100** с шагом **step=5** по умолчанию. Координата **z** точки **A** вычисляется функцией **z=POLIZ_D(string[]='=SIN((ox^2+oy^2)^0.5);', perem[]={x,y})**. Координаты фокуса по умолчанию — **F(0,0,-150)**.

Находим уравнение прямой **AF**. У нас есть нормаль к плоскости экрана — вектор **v(0,0,1)**. Плоскость экрана совпадает с плоскостью **XOY**. Значит, нам известно и уравнение плоскости экрана, проходящей через точку **O(0,0,0)** и перпендикулярной данному вектору **v(0,0,1)** — **Ax+By+Cz+D=0, A=B=D=0, C=1, z=0**.

Находим точку пересечения прямой **AF** с плоскостью экрана **z=0**.

Нам нужно разложить точку **Q** в базисе векторов **h(1,0,0), v(0,0,1) и w=[h,v]:**

$$\begin{cases} Xw=Zv*Yh-Yv*Zh; \\ Yw=-Zv*Xh+Xv*Zh; \\ Zw=Yv*Xh-Xv*Yh; \end{cases}$$

Теперь точка **O** — начало координат на дисплее. Таким образом, следующими тремя группами преобразований мы можем вычислить координаты **pXs, pYs, pZs** точки **A**, спроецированной на дисплей в соответствующем координатном пространстве (т.е. осуществить переход от мнимой трехмерной системы координат к реальной двумерной системе координат дисплея).

$$\begin{cases} Xo=Xf+f*Xv; \\ Yo=Yf+f*Yv; \\ Zo=Zf+f*Zv; \end{cases}$$

$$\begin{cases} Xm=*pXq-Xo; \\ Ym=*pYq-Yo; \\ Zm=*pZq-Zo; \end{cases}$$

$$\begin{cases} Xh*pXs+Xw*pYs+Xv*pZs=Xm; \\ Yh*pXs+Yw*pYs+Yv*pZs=Ym; \\ Zh*pXs+Zw*pYs+Zv*pZs=Zm; \end{cases}$$

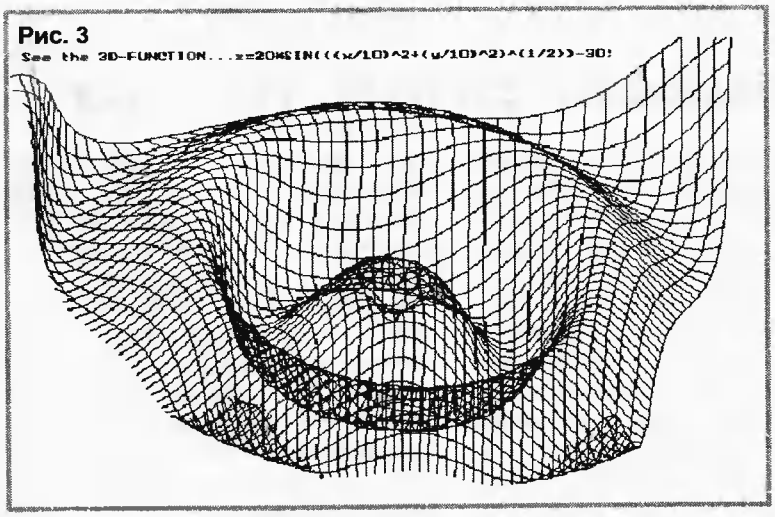
Получаем координаты **x=pXs** и **y=pYs** на плоскости экрана и **z=pZs** (стремится к 0).

Константные значения **f** и **scale** позволяют преобразовывать (увеличивать) мнимое изображение в хорошо просматриваемую поверхность, т.е. если **f=1** и **scale=1**, то поверхность построится в центре дисплея и будет ничтожно малой. Поэтому по умолчанию **f=10** и **scale=10**.

Помимо реализации самого алгоритма построения поверхности, над изображением можно также производить некоторые операции (**рис.3...5**):

- масштабировать (уменьшать или увеличивать) поверхность;

Рис. 3



базисе. Поворот осуществляется на угол, равный переменной **st_alpha**.

Вот пример поворота вокруг вектора **v**. Первоначальные координаты векторов **h** и **w** — **Xh**, **Yh**, **Zh** и **Xw**, **Yw**, **Zw** соответственно. Координаты векторов h и w после поворота — **Xh1**, **Yh1**, **Zh1** и **Xw1**, **Yw1**, **Zw1**:

```
Xh1=Xh*cos(st_alpfa)+Xw*sin(st_alpfa);
Yh1=Yh*cos(st_alpfa)+Yw*sin(st_alpfa);
Zh1=Zh*cos(st_alpfa)+Zw*sin(st_alpfa);

Xw1=Xw*cos(st_alpfa)-Xh*sin(st_alpfa);
Yw1=Yw*cos(st_alpfa)-Yh*sin(st_alpfa);
Zw1=Zw*cos(st_alpfa)-Zh*sin(st_alpfa);
```

Перспектива

Для изменения глубины восприятия трехмерного изображения нужно изменить переменную **f** на значение **st_focus**.

Рис. 4

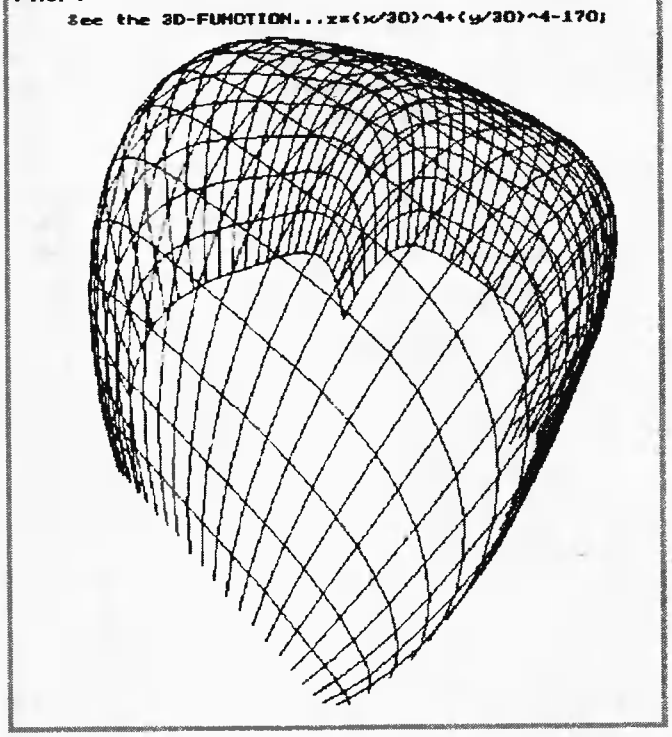
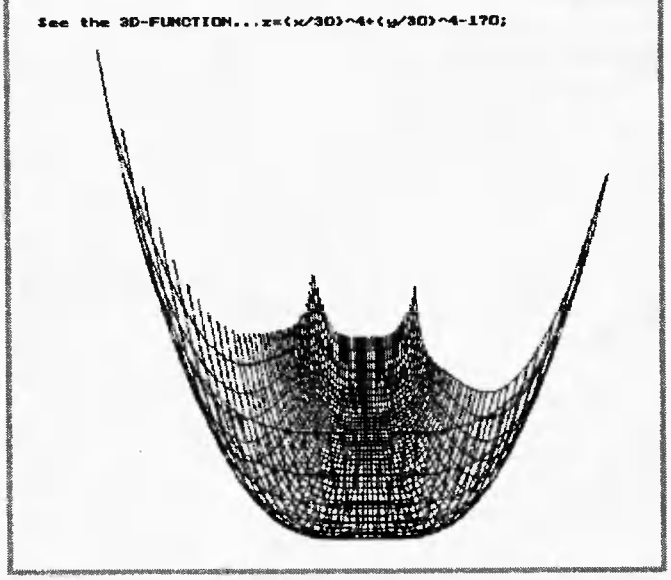


Рис. 5



- поворачивать относительно осей координат (точнее, поворачивается не поверхность, а наблюдатель);
- изменять глубину восприятия изображения (перспектива);
- перемещаться вдоль осей координат;
- изменять координаты фокуса;
- изменять густоту сетки.

Масштабирование

По умолчанию масштаб (переменная **scale**) равен **10**. Масштаб можно увеличивать или уменьшать на значение переменной **st_scale**, по умолчанию равной **1**.

Поворот

Это самая сложная из операций над поверхностью. Чтобы избежать искажения изображения при повороте и максимально точно производить поворот, нужно поворачивать базис поверхности, а не саму поверхность в

Перемещение

Для перемещения изображения вдоль осей координат, так же как и при повороте, целесообразнее изменять координаты базиса (перемещать базис, а не поверхность в базисе).

Фокус

Перемещая фокус, мы добьемся некоторого рода искажения, т.е. одновременно возможны и поворот, и перемещение.

Сетка

Для ускорения проведения вышеописанных операций можно уменьшить густоту сетки, а затем увеличить ее до желаемого качества изображения.

Литература

1. А.Слонимский. Анализ и вычисление алгебраических выражений, задаваемых аналитически. — Радиомир. Ваш компьютер, 2004, №8, С.40.

"PlayStation 2": процессорная плата

С.РЮМИК,
г.Чернигов.

Силач побеждает одного, а знающий побеждает тысячу.
Каракалпакская поговорка

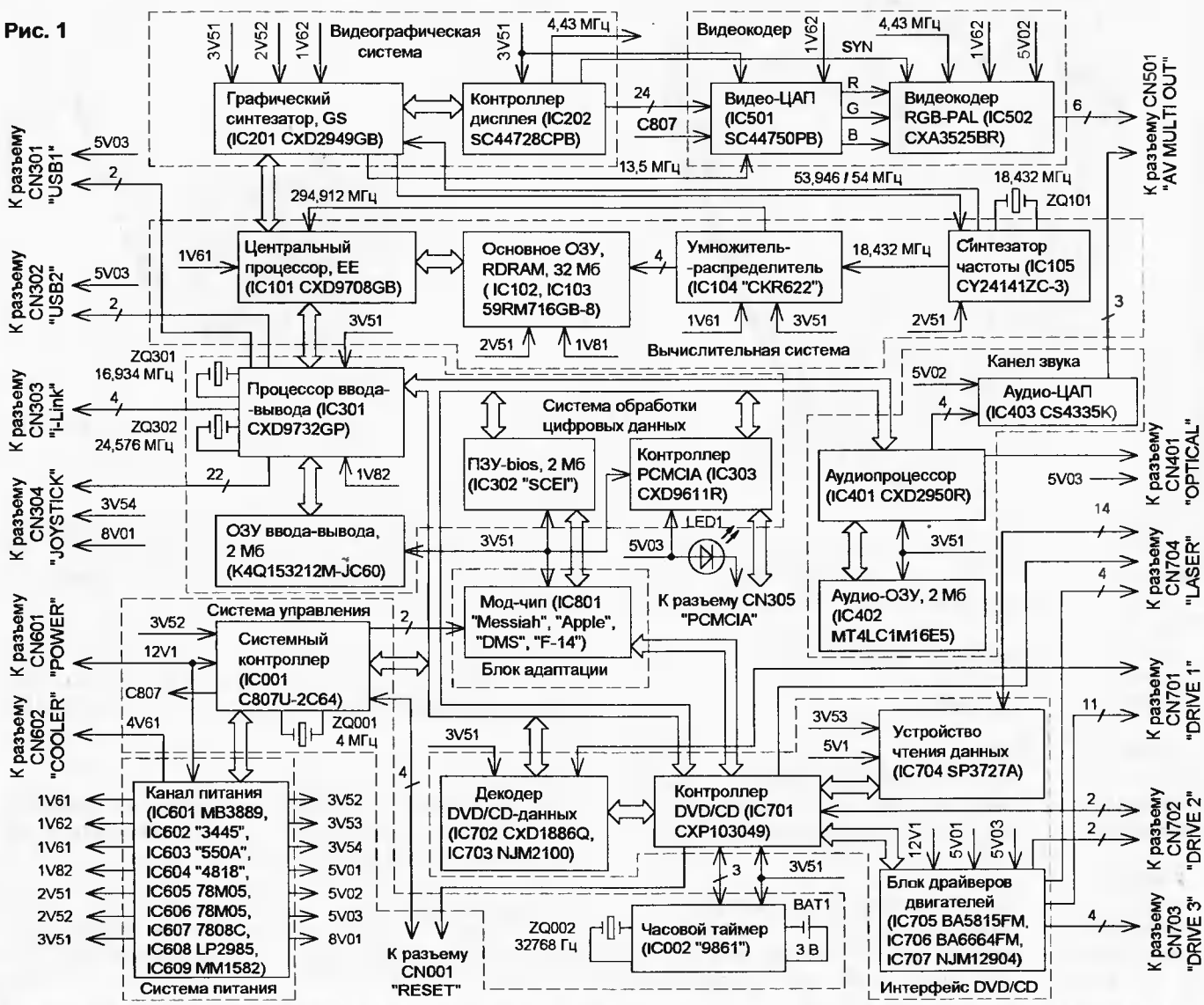
При анализе работы сложных систем, примером которых является игровая приставка "PlayStation 2" (PS2), на первое место выходит четкое понимание взаимосвязей между ее составными частями. Главная роль здесь отводится структурной схеме процессорной платы.

За четыре года выпуска PS2 было разработано более 20 типов процессорных плат с маркировками GH-001...GH-026. К сожалению, на платах отсутствуют номера микросхем, транзисторов, разъемов, по которым можно было бы отнести их к определенной системе. Связано это, скорее всего, с экономией изготовления, так как не требуется наносить дополнительный слой надписей методом шелкографии. Попутно затрудняется жизнь хакерам в части обмена информацией о месте установки мод-чипа (МЧ).

При составлении структурной схемы процессорной платы применяется старый испытанный способ — прозвонка электрических связей омметром. Визуально проследить проводники трудно, поскольку плата шестислойная. Результаты прозвонки для платы GH-015 модели SCPH-30004R (версия 6) представлены на **рис.1**. Судя по фотографиям процессорных плат из Интернета, эта схема подходит для всех PS2 версий 4...8.

По аналогии с приставкой "PlayStation" (PSX), на структурной схеме PS2 выделены 8 систем. Для удобства сохранены их прежние названия и порядковые номера: 0 — система управления, 1 — вычислительная, 2 — видеографическая, 3 — обработки цифровых данных, 4 — канал звука, 5 — видеокодер, 6 — система питания, 7 — интерфейс DVD/CD, 8 — блок адаптации. Радиоэлементы на электрических схемах условно обозначены так, как принято в PSX: микросхемы — IC, транзисторы — Q, диоды — D, разъемы — CN. Порядковые номера элементов — трехзначные, где первая цифра указывает на принадлежность к системе. Исключение из правила — предохранители.

Рис. 1



Они, единственные, имеют фирменную маркировку PS1...PS13, TH1...TH3, выполненную "прогалинами" в защитном слое зеленого цвета.

Для каждого блока на структурной схеме указаны названия входящих микросхем. Если какую-то из них не удалось идентифицировать, то в кавычках приведена надпись на ее корпусе.

Питающие напряжения обозначены четырьмя символами. Первые две цифры обозначают номинал напряжения в вольтах, при этом буква "V" служит десятичной точкой. Последняя цифра указывает порядковый номер цепи. Пример: 3V52 — напряжение 3,5 В, вторая по счету цепь; 8V01 — напряжение 8 В, первая по счету цепь.

Взаимодействие систем

После включения PS2 в сеть 220 В, на разъем CN601 от платы питания поступает напряжение +12 В (12V1). Это напряжение подается в канал питания, где из 9 стабилизаторов запускается только один, выполненный на микросхеме IC603. Он формирует напряжение 3V52, которое подается на систему управления. Контроллер IC001 начинает посылать сигналы опроса к блоку кнопок через разъем CN001. На лицевой панели PS2 загорается светодиод красного цвета, свидельствующий о наличии питания и о готовности к дальнейшей работе. В этом режиме PS2 будет находиться до тех пор, пока пользователь не нажмет кнопку "RESET" или "⏏".

После нажатия кнопки контроллер IC001 поочередно разрешает работу стабилизаторов канала питания, каждый раз контролируя аварийные сигналы. Первым включается стабилизатор IC608 воздушного вентилятора, что легко заметить на слух. При обнаружении неполадок в стабилизаторах, контроллер IC001 полностью или частично блокирует систему питания, а индикаторы на лицевой панели синим цветом свечения или миганием красного указывают на неисправность.

При отсутствии аварий цвет индикатора на лицевой панели меняется с красного на зеленый. После этого центральный процессор IC101 считывает из ПЗУ-bios IC302 программу начального запуска и с помощью видеографической системы, канала звука и видеокодера выдает на разъем CN501 телевизионный и звуковой сигналы фирменной заставки. Когда на экране телевизора появляется начальное меню, управление передается процессору ввода-вывода IC301, который опрашивает состояние кнопок джойстиков, подключенных через коммутационную плату к разъему CN304.

Для того чтобы вставить в PS2 игровой диск, требуется вначале выдвинуть наружу привод DVD/CD. Происходит это после нажатия кнопки "⏏" на лицевой панели PS2. Системный контроллер IC001 принимает сигнал нажатия кнопки и транслирует его в контроллер DVD/CD IC701. Последний выдает команду в блок драйверов двигателей. Микросхема IC705 через разъем CN702 приводит в движение механизм подачи диска. Контроллер IC701 анализирует сигнал механического датчика "STOP" от разъема CN702 и после полного "выезда" привода останавливает работу блока драйверов.

Пользователь вставляет лазерный диск в привод и вновь нажимает кнопку "⏏". Все действия по управлению системой повторяются снова, только двигатель подачи диска вращается в противоположную сторону, а контроллер IC701 останавливает работу блока драйверов при полном "въезде" привода внутрь корпуса PS2.

Далее начинают формироваться сигналы вращения диска (CN701) и перемещения каретки (CN703). Устройство чтения данных через разъем CN704 подает напряжение на лазерные диоды, формируя луч, и принимает сигналы, отраженные от поверхности диска. После предварительной обработки эти сигналы поступают в контроллер IC701 и в декодер DVD/CD-данных IC702, IC703.

После декодирования информация передается по общей шине в процессор IC301. Именно он определяет тип диска (DVD или CD), формат представления данных (Audio, Video или ROM), а также совместимость со стандартом PSX или PS2. Если обнаружен обычный музыкальный AudioCD или диск с играми для PSX, то в работу включается вычислительное ядро R3000A, находящееся в процессоре IC301. "Начинка" IC301 эквивалентна большей половине всей электроники PSX! При этом EE и GS выступают в качестве сопроцессоров, добавляющих различные видеоэффекты. Теперь становится понятным, почему "старые" игры от PSX смотрятся на PS2 визуально гораздо привлекательнее.

Если обнаружен диск DVD или с играми для PS2, то центральным процессором становится EE. Микросхеме IC301 отводится роль периферийного сопроцессора, на который возлагается задача общения с устройствами, подключенными к разъемам CN301...CN304. В частности, во время игры он постоянно анализирует сигналы от джойстиков, расшифровывает их и передает в EE. Последний производит анализ действий игровых персонажей, рассчитывает их дальнейшую судьбу по формулам "искусственного интеллекта", вычисляет физические свойства окружающего мира (трение шин автомобиля, высоту волн на море и т.д.), определяет геометрические координаты объектов. При просмотре DVD-фильмов EE автоматически производит распаковку данных формата MPEG-2.

Результаты вычислений хранятся в основном ОЗУ объемом 32 Мб, которое состоит из двух одинаковых микросхем IC102, IC103. По структуре это быстродействующая динамическая память Direct Rambus DRAM, применяемая в компьютерах Pentium-IV. Ее главное достоинство заключается в небольшом числе соединительных линий и высокой скорости доступа — 3,2 Гб/с. Платить приходится повышенной тактовой частотой 400 МГц, специальным контроллером памяти RAC (находится в EE), двумя линиями питания и точной выдержкой топологии полосковых соединительных проводников. Данные передаются по фронту и срезу тактовых импульсов, поэтому иногда говорят, что Rambus ОЗУ работает с эквивалентной частотой 800 МГц.

После проведения вычислений EE посылает в GS макрокоманды на рисование многоугольников, фильтрацию изображения, сглаживание углов. GS облекает объекты в текстуры, вводит видеоэффекты тумана, прозрачности и т.д. Внутри GS имеется динамическое ОЗУ объемом 4 Мб. Это не видео-ОЗУ, где хранится картинка экрана (как многие думают), а инструмент для многопотоковой обработки изображения с внутренней 2560-разрядной (ошибки нет!) шиной.

Цифровые сигналы от GS поступают на контроллер дисплея. Дисплеем в данном случае выступает телевизор, причем как обычный — с пропорциями сторон 4:3, так и широкоформатный — с пропорциями 9:16 (переключается из меню). Компьютерный монитор подключить напрямую к PS2 нельзя.

Видео-ЦАП IC501 переводит три восьмиразрядных цифровых потока (R, G, B) в аналоговую форму. Видеокодер IC502 осуществляет формирование полного цветового телевизионного сигнала ПЦТС в стандарте PAL

или NTSC, в зависимости от модели PS2. Соответственно, частота поднесущей цветности PAL, подаваемая на него, равна 4,43 или 3,58 МГц.

Работа всех систем процессорной платы тактируется генераторами с кварцевой стабилизацией частоты. Базовой для вычислительной и видеографической систем является частота 18,432 МГц (IC105), из которой путем умножения в микросхеме IC104 получается частота для EE — 294 МГц, а затем, после деления, внутренняя частота видеошины GS 147 МГц.

К процессору IC301 подключаются сразу два резонатора. Они обеспечивают работу ядра R3000A PSX на обычной (33,8688 МГц) и повышенной (36,864 МГц) частоте. Смена режимов производится в меню PS2.

Микросхема IC105 подает на GS сигнал частотой 53,946 или 54 МГц в зависимости от потребностей контроллера дисплея. Частота дискретизации видео-ЦАП IC501 составляет 13,5 МГц.

"Часовой" резонатор ZQ002 32768 Гц идеально подходит для отсчета секундных интервалов времени. Микросхема часового таймера IC002 имеет встроенный календарь, работа которого поддерживается батарейкой BAT1 при выключении питания PS2.

Мод-чип (МЧ) IC801 блока адаптации позволяет запускать на PS2 копии лицензионных игр и смотреть DVD-фильмы разных зон. Наличие МЧ превращает PS2 из фирменной в PS2-совместимую. И хотя на нее уже не распространяется гарантия ф.Sony, многих это не смущает. Существует несколько десятков разновидностей МЧ с различными схемами включения и разным количеством соединительных проводов. Считается, что хорошей совместимостью обладают чипы DMS3, Messiah, Apple, Magic и F-14. Ядром МЧ служит заказная программируемая логическая интегральная схема (ПЛИС), распаянная на отдельной печатной плате. Дополнительно на ней могут содержаться кварцевый резонатор, чип-резисторы, конденсаторы и даже микроконтроллер.

Схемотехника узлов процессорной платы

Нет смысла составлять полную электрическую схему PS2 при наличии подробной структурной. Гораздо важнее, с точки зрения ремонта, иметь под рукой электрические схемы цепей, к которым подключены разъемы, а также схему системы питания. Именно здесь чаще всего возникают отказы, которые поддаются устранению.

Блок кнопок

На **рис.2** приведена совмещенная электрическая схема блока кнопок и входных цепей разъема CN001 "RESET". На лицевую панель PS2 выведены два индикатора: трехцветный LED1 (красный, зеленый или желтый) и LED2 синего цвета свечения. Яркость первого из них определяют резисторы R1 и R2, второго — R3. Низкое сопротивление R3 обусловлено повышенным падением напряжения на LED2, которое составляет 3 В против 1,6...1,8 В у LED1.

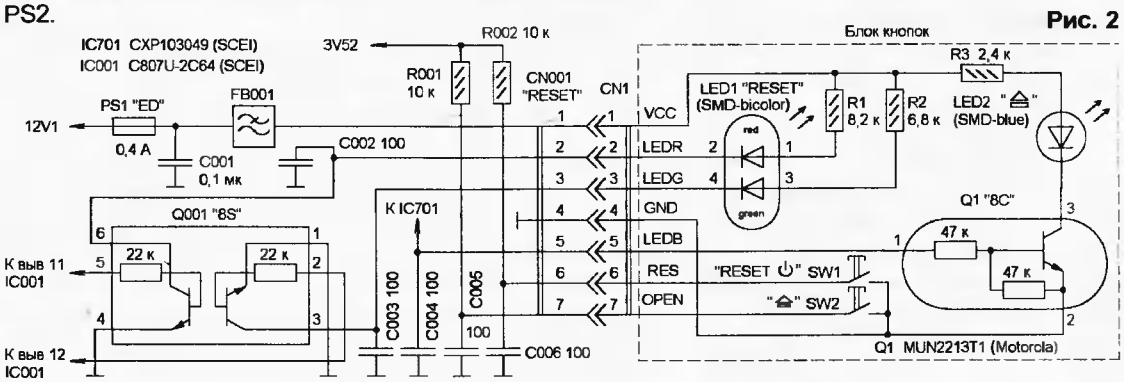


Рис. 2

Синим индикатором управляет "цифровой" транзистор Q1 в блоке кнопок, а красным и зеленым — два "цифровых" транзистора в сборке Q001. Желтый цвет получается, когда одновременно открыты оба транзистора в Q001.

Кнопки "RESET" и "⏏" подключены к нагрузочным резисторам R001, R002. Конденсаторы C002...C006 устраняют импульсные помехи, которые могут наводиться на длинном (20 см) гибком кабеле, соединяющем блок кнопок и процессорную плату.

Неисправности блока кнопок

Если не светится ни LED1, ни LED2, то следует прозвонкой проверить предохранитель PS1 и SMD-фильтр FB001. Их суммарное сопротивление не должно превышать 2 Ом. Постоянное свечение одного из индикаторов может быть вызвано пробоем цепи "коллектор-эмиттер" в транзисторах Q1, Q001. Заменить их можно практически любыми n-p-n-транзисторами, например, КТ315Б, и навесными резисторами.

Светодиоды LED1, LED2 — поверхностно-монтируемые. Их аналоги имеются в каталогах продукции ф.Kingbright. Если корпус PS2 кем-то уже ранее вскрывался, то имеется вероятность механического повреждения гибкого кабеля, соединяющего блок кнопок и разъем CN001. Исправность линий проверяется прозвонкой. Устранение дефекта — заменой кабеля или дублированием цепи обычным тонким проводом, распаянным прямо на контакты разъемов.

Канал "i-Link"

На **рис.3** приведена схема цепей разъема "i-Link". Через него передаются сигналы высокоскоростной последовательной шины международного стандарта IEEE1394 (1995 г.). В середине 80-х годов он был разработан ф.Apple Computer под названием "FireWare". Ф.Sony в лицензионных целях изменила название на "i-Link" (торговая марка), сохранив все электрические и конструктивные параметры.

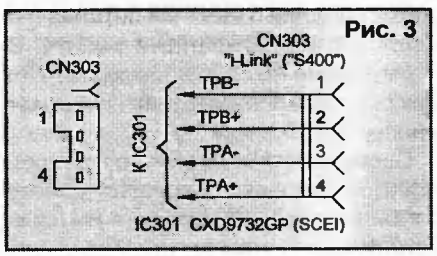


Рис. 3

Стандарт IEEE1394 предусматривает подключение до 63 устройств при скорости передачи данных 100...400 Мбит/с на расстояние 4,5...72 м. Маркировка "S400" возле разъема свидетельствует о том, что в PS2 максимальная скорость составляет 400 Мбит/с. На разъем CN303 "i-Link" выведены две пары симметричных сигналов: TPB± (вход) и TPA± (выход). Для двух PS2 через "i-Link" можно организовать сетевую игру, например, в "Gran Turismo 3". В соединительном кабеле выходы TPA одного разъема должны быть соединены с входами TPB другого и наоборот.

Планы разработчиков PS2 по подключению к "i-Link" разнообразной периферии остались на бумаге. Более того, начиная с модели SCPH-50000, ф.Sony вывела этот разъем из PS2, заменив его сетевым адаптером Ethernet.

Неисправности канала "i-Link", чаще всего, механического плана. Они могут возникать, если применялось чрезмерное усердие при состыковке разъема. Электрические неполадки проверяются измерением сопротивления 115 Ом между контактами пар TPB+, TPB- и TPA+, TPA-. Если сопротивление в норме, то неисправна микросхема IC301. Проследить, к каким выводам этой микросхемы подходят сигналы TPA и TPB, практически невозможно, поскольку они "спрятаны" под корпусом и не имеют доступа извне.

Канал USB

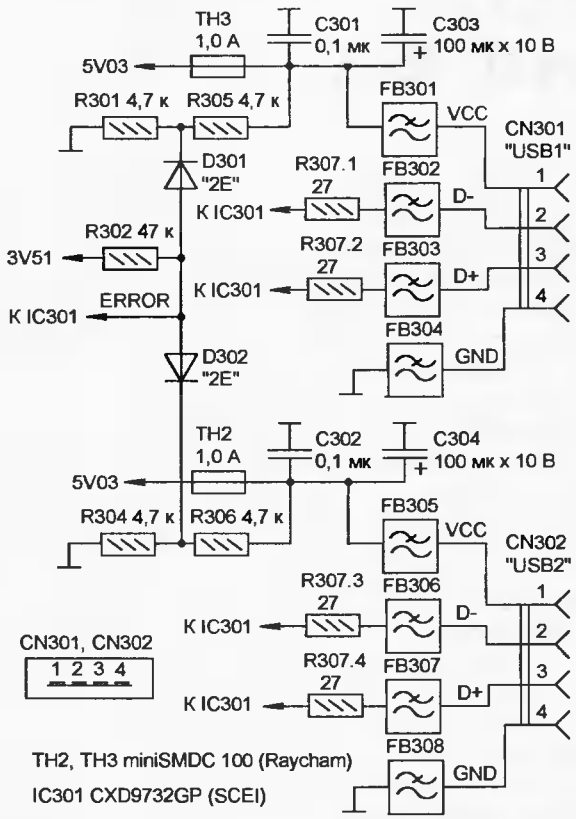
На **рис.4** приведена электрическая схема включения разъемов CN301, CN302 ("USB1", "USB2"). Шина USB (Universal Serial Bus) предназначена для передачи данных со скоростью 12 Мбит/с. Она может обслуживать до 127 устройств и допускает их "горячее" подключение или отключение без снятия питания. Сделать это позволяют два "полисвича" (самовосстанавливающиеся предохранители) TH2, TH3 и монитор питания на элементах R301...R306, D301, D302.

В рабочем состоянии на контакты 1 разъемов CN301, CN302 подается напряжение питания 5 В, при этом сигнал ERROR имеет уровень логической "1". Если в результате перегрузки произойдет срабатывание любого из предохранителей TH2, TH3, то сигнал ERROR установится в логический "0", и процессор ввода-вывода IC301 отключит информационные цепи D+, D- от разъемов USB, предотвращая повреждение PS2.

Неисправности канала USB проявляются в отсутствии управления от периферийных устройств, подключенных к разъемам CN301 и (или) CN302. Проверке подлежат предохранители TH2, TH3 (их сопротивление должно составлять не более 5 Ом), фильтры FB301...FB308 (сопротивление не более 1 Ом), резисторы R301, R302, R304...R307, диоды D301, D302. Заменой самовосстанавливающихся предохранителей TH2, TH3 могут быть обычные поверхностно-монтируемые, рассчитанные на ток срабатывания 1 А, или перемычки из тонкой проволоки диаметром 0,08...0,1 мм.

Если все исправно, то дефект, очевидно, кроется в микросхеме IC301. Она ремонту или замене не подлежит из-за безвыводной (BGA) конструкции корпуса.

Рис. 4



Цветовые коды HTML

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

При разработке HTML-документов в текстовом процессоре Word 2000/XP существенным недостатком является значительный объем HTML-кода получаемых файлов. Это заметно снижает скорость загрузки Web-страниц, особенно если последние будут размещены в Интернете. При определенном знании языка HTML, компактные Web-страницы можно создавать, используя в качестве HTML-редактора стандартный "Блокнот" Windows.

В этом случае при оформлении Web-страниц важно знать значения цветовых кодов HTML. Знание и умелое применение цветовых кодов может дать возможность более выразительно отобразить различные элементы HTML-документов (фон, текст, фоновую заливку ячеек в таблицах и т.д.). Код цвета задается в виде шестнадцатеричных значений интенсивности красной, зеленой и синей составляющих. При этом в тексте значению кода цвета должен предшествовать символ # (решетка). Кроме шестнадцатеричных значений, код цвета в некоторых случаях может быть задан в виде стандартных названий цветов на английском языке, например, "black" или "white". Большинство современных Web-браузеров обеспечивают правильное отображение цветов с использованием как шестнадцатеричных значений, так и стандартных названий цветов. Например, вставив в HTML-документ следующую строку

```
<body bgcolor=#003366 text=#E0E0E0>,
```

мы увидим текст светло-серого цвета на темно-сизом фоне.

В **таблице** в алфавитном порядке указаны коды некоторых цветов.

Цвет	Код
Белый	White
Бледно-голубой	#99ccff
Бледно-зеленый	#ccffcc
Вишневый	#993366
Голубой	#00ccff
Желтый	Yellow
Зеленый	Green
Коричневый	#993300
Красный	Red
Оливковый	#333300
Оранжевый	#ff6600
Розовый	#ff99cc
Светло-бирюзовый	#ccffff
Светло-коричневый	#ffcc99
Серый 12%	#e0e0e0
Серый 40%	#999999
Серый 50%	Gray
Серый 7%	#f0f0f0
Синий	Blue
Темно-сизый	#003366
Темно-синий	Navy
Черный	Black

Screens Viewer в iS-DOS на Sinclair

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

Гуще трава — легче косить.
Вождь готтов Аларих

Загружаемая операционная система iS-DOS для Sinclair-совместимых машин предоставляет несравненно больше возможностей и удобств для массированных операций над структурированными данными, чем традиционно используемая в силу своей "твердотельности" ОС TR-DOS [1]. Достаточно будет напомнить, что непрерывное операционное пространство, доступное под управлением ОС iS-DOS, может достигать 16 Мб.

Одним из типов данных, над которыми могут потребоваться массированные файловые операции, являются файлы с графической информацией, количество форматов которых на платформе Sinclair уже приближается к десятку. Для примера можно назвать чанковые экраны, цветные чанковые экраны, двухэкранные изображения (часто называемые "image"), а также "триколорные" RGB-изображения и их AGA-разновидность (BRG). Сюда же можно добавить и упакованные варианты перечисленных форматов. Но самым распространенным и востребованным форматом, конечно же, остается стандартный формат "screen", представляющий собой сохраненное в файле содержимое экранной области памяти Sinclair-совместимого компьютера.

Для работы с файлами стандартного формата "screen" в среде ОС iS-DOS существует утилита "exescr.com", имеющая статус, близкий к системному. Она расположена в главном, системном каталоге SHELL и необходима для вывода на экран файлов графических изображений (картинок) с расширением ".scr" (в том числе и предварительно сжатых iS-DOS'ной программой-упаковщиком "spacker.com"). Как правило, командный файл "exescr.com" для просмотра экранов вызывается по нажатию клавиши <3> (View) в системном меню. Этот вызов описан в конфигурационном текстовом файле "extview.txt". Обычно графический просмотрщик вызывается с ключом "/w" (wait), для того чтобы иметь возможность рассматривать выведенное изображение столько времени, сколько понадобится до нажатия любой клавиши:

```
scr:Q:SHELL\exescr /w
```

При всех своих, в том числе и не упомянутых здесь, достоинствах, область применения "exescr.com" имеет и свои ограничения. Так, например, просмотрщик работает с графическими файлами только "поодиночке". В фирменной сопроводительной документации на программу описан способ использования утилиты для циклического просмотра нескольких изображений, требующий предварительного описания всех вызовов в пакетном bat-файле, создаваемом специально для этой цели. Но удобным этот способ группового просмотра экранных файлов назвать трудно, так как никакое управление процессом просмотра со стороны пользователя не предусматривается, а выход из такого "просмотра" возможен только аварийный — по клавише <Breaк>, либо по ошибке — из-за переполнения стека возврата интерпретатора bat-файлов.

Таким образом, существует очевидная необходимость в групповом "вьюере" экранных файлов в среде iS-DOS, дополняющем возможности стандартного просмотрщика "exescr.com". Надо сказать, что такую необходимость можно считать объективной, поскольку в настоящее время известно, по крайней мере, о двух попытках реализации программы подобного назначения, предпринятых в разных местах, но примерно в один и тот же период времени, и при этом совершенно независимо друг от друга. Одна из этих реализаций и описана в предлагаемой публикации.

Ниже приводится исходный текст утилиты-просмотрщика в формате iS-DOS Assembler, содержащий достаточно подробные комментарии для понимания принципа работы программы и взаимодействия отдельных составляющих ее частей. По нашему глубокому убеждению, стремление к экономии на комментариях и примечаниях к исходным текстам программ нельзя считать оправданным. Согласно одной "старинной китайской мудрости", отсутствие комментариев в программе — веская причина для увольнения программиста [2]. В данном случае, особое внимание было уделено комментированию обращений к системным рестартам ОС iS-DOS, поскольку именно эти фрагменты сильнее всего выделяют и отличают стиль программирования под операционные системы от программирования под конкретное "железо", обычно практикуемое в среде TR-DOS.

sviewer.asm

```
;Screens VIEWER
;
;Групповой просмотрщик экранных файлов для ОС iS-DOS.
;Прахов А. А. / ist, г. Балашов.
;статус: freeware.
;режим работы: интерактивный.
;v1.00: 28.2.01  Тестовая beta-версия;
;v1.20: 8.9.02   Апгрейд;
;v1.21: 10.8.03  Изменено управление - курсорные клавиши
;                вместо <Delete> и <Space> - И.Е.В / ist;
;v1.22: 22.3.04  Мелкий "жучок" выловлен и засушен, а также
;                усилена "противоракетная оборона" - И.Е.В.
;
;Для генерации sviewer.com компоновать sviewer.obj с rst.obj
;или с rst.gtb - таблицей номеров системных рестартов,
;например: link sviewer /old/sym T:AS\rst.

          ORG 24000         ;#5DC0
                                .
          CALL TAKE_F       ;"Берем" файл из-под курсора
          CALL OPENFILE     ;и открываем его.
          CALL TEST_FN
          JR C,EXIT         ;Если не экран, то - на выход.
          CALL INI          ;Установка цв. атрибутов экрана.
          CALL PRT_SCR      ;Вывод файла на экран.

;-------------------------------------------------------------
;Опрос клавиатуры.

G1        LD C,$ktest       ;Проверка на нажатие какой-либо
          RST 16            ;клавиши. Выход: Z - не нажата.
          JR Z,G1
          LD C,$key         ;Ввод символа. Выход: A - код
          RST 16            ;нажатой клавиши.
          OR A
          CP #10            ;<Symbol_Shift>/<A>
          JR Z,EXIT
          CP #A             ;<Down_arrow>
          CALL Z,NEXT
          CP "A"            ;<A>
          CALL Z,NEXT
```

```
        CP "a"          ;<a>
        CALL Z,NEXT
        CP #B           ;<Up_arrow>
        CALL Z,BACK
        CP "Q"          ;<Q>
        CALL Z,BACK
        CP "q"          ;<q>
        CALL Z,BACK
        JR G1           ;Опрашиваем клавиатуру снова.

EXIT    XOR A
        LD A,#F4        ;Выход в систему с перепечаткой
        RET             ;обеих панелей.

;=======================================
;Переход к предыдущему файлу.

BACK

;Проверка на количество файлов.

        LD A,(NUMBER)   ;Не "пришли" ли мы к первому
        CP 0            ;экранному файлу в каталоге?
        JR Z,BA_G1

        DEC A           ;Если нет, то готовимся к
        LD (NUMBER),A   ;обработке нового файла.
        CALL OPENFILE
        JR C,BA_G3

        LD A,0          ;Устанавливаем черный бордюр,
        OUT (#FE),A     ;так как он мог быть изменен.
        CALL TEST_FN    ;Проверяем файл на "экранность".
        JR C,BACK       ;Если не экран, то - к другому.

        CALL PRT_SCR    ;Выводим изображение на экран.
        LD A,0
        RET

;Перебор только по существующим файлам.

BA_G3   LD A,(NUMBER)
        DEC A
        LD (NUMBER),A

;Перебор по количеству файлов.

BA_G1   LD A,2          ;Сигнализируем красным бордюром
        OUT (#FE),A     ;о достижении начала каталога.
        LD A,0
        RET

;-------------------------------------------------------------
;Переход к следующему файлу.

NEXT    LD A,(NUMBER)   ;Не "уперлись" ли мы в последний
        CP 127          ;экранный файл каталога?
        JR Z,NE_G1

        INC A           ;Если нет, то готовимся к
        LD (NUMBER),A   ;обработке следующего файла.
        CALL OPENFILE
        JR C,NE_G3

        LD A,0          ;Устанавливаем черный бордюр,
        OUT (#FE),A     ;так как он мог быть изменен.
        CALL TEST_FN    ;Проверяем файл на "экранность".
        JR C,NEXT       ;Если не экран, то - к следующему.

        CALL PRT_SCR    ;Выводим изображение на экран.
        LD A,0
        RET

;Перебор только по существующим файлам.

NE_G3   LD A,(NUMBER)
        DEC A
        LD (NUMBER),A

;Перебор по количеству файлов.

NE_G1   LD A,2          ;Сигнализируем красным бордюром
        OUT (#FE),A     ;о достижении конца каталога.
        LD A,0
        RET

;-------------------------------------------------------------
;Вывод на экран открытого файла.

PRT_SCR LD C,$rifle     ;Чтение файла в блоках:
        LD B,27         ;количество блоков,
        LD DE,0         ;номер блока начала,
        LD HL,#4000     ;куда считывать (dec: 16384).
        RST 16
        RET C           ;Выход, если ошибка.
        RET

;-------------------------------------------------------------
;Проверка на соответствие по расширению имени файла,
;а также на разрешение доступа согласно системным атрибутам.
;IN:  расширение в TFN_D,
;OUT: FLAG "C"=0 - OK.

TEST_FN LD HL,TF_D+8
        LD DE,TFN_D
        LD BC,3         ;Расширение имени нового файла
        LDIR

        LD HL,TFN_D
        LD A,(HL)
        CP "s"          ;Расширение имени файла
        JR NZ,TFN_G1    ;должно быть "scr"
        INC HL
        LD A,(HL)
        CP "c"
        JR NZ,TFN_G1
        INC HL
        LD A,(HL)
        CP "r"
        JR NZ,TFN_G1

;Проверка на длину файла.

        LD IX,TF_D
        LD A,(IX+14)    ;Нужно, чтобы длина
        LD DE,6912      ;исходного файла
        CP E            ;была 6912 байт,
        JR NZ,TFN_G1    ;то есть, -
        LD A,(IX+15)    ;стандартный экран.
        CP D
        JR NZ,TFN_G1

;Проверка системных атрибутов файла.

        LD IX,TF_D
        LD A,(IX+11)    ;В описателе файла
        BIT 0,A         ;проверяем атрибут "удаленный",
        JR Z,TFN_G1
        BIT 2,A         ;-//- атрибут "защищен от чтения",
        JR NZ,TFN_G1
        BIT 4,A         ;проверяем атрибут "скрытый",
        JR NZ,TFN_G1
        BIT 5,A         ;
        JR NZ,TFN_G1    ;А это вообще каталог!
        SCF
        CCF             ;Порядок,
        RET             ;можем работать дальше.

TFN_G1  SCF             ;Нет, это не то, что нужно.
        RET

TFN_D   DEFS 3          ;Для расширения имени файла.

;-------------------------------------------------------------
;"Берем" файл из-под курсора.

TAKE_F  LD C,Sg_curs    ;Возвращаются параметры
        RST 16          ;панельного курсора:
```

```
          LD (NUMBER),A     ;в A - номер файла.
          RET

;-------------------------------------------------------------
;Открытие файла;
;IN:  "A" - номер,
;OUT: TF_D - FNAME.

OPENFILE  LD C,$opnum       ;Открывается файл по номеру
          RST 16            ;в текущем каталоге.
          RET C
          EXX
          LD DE,TF_D
          LD BC,32          ;Заголовок файла.
          LDIR
          EXX
          RET

TF_D      DEFS 32           ;Для параметров описателя файла.
NUMBER    DEFB 0            ;Для номера открытого файла.

;-------------------------------------------------------------
;Инициализация цветовых атрибутов.

INI       LD A,0            ;Так как в аккумуляторе - "0",
          LD C,$cls         ;то очистка экрана с цветами
          RST 16            ;внутрисистемных переменных ОС.
          LD A,7
          LD C,$cls         ;Подкраска экрана цветом,
          RST 16            ;заданным в A (то есть, =7).
          LD A,0
         OUT (#FE),A      ;Установка цвета бордюра (=0).
          RET
```

Для пользователей, испытывающих затруднения при программировании на Ассемблере, приводится также "дамп" — шестнадцатеричная распечатка машинных кодов готовой программы, которую можно занести в оперативную память с помощью любого шестнадцатеричного монитора или редактора и сохранить затем в файле как блок машиных кодов длиной 320 байтов, не забыв указать в заголовке стартовый адрес 24000 (#5DC0).

```
sviewer.com, LSA: 24000, Size: 320

      block 0                          block 1

#5DC0: CD BA 5E CD C1 5E CD 69   #5EC0: C9 0E 26 D7 D8 D9 11 D0
#5DCF: 5E 38 33 CD F1 5E CD 5C   #5ECF: 5E 01 20 00 ED B0 D9 C9
#5DD0: 5E 0E 09 D7 28 FB 0E 07   #5ED0: 00 00 00 00 00 00 00 00
#5DDF: D7 B7 FE 10 28 20 FE 0A   #5EDF: 00 00 00 00 00 00 00 00
#5DE0: CC 2F 5E FE 41 CC 2F 5E   #5EE0: 00 00 00 00 00 00 00 00
#5DEF: FE 61 CC 2F 5E FE 0B CC   #5EEF: 00 00 00 00 00 00 00 00
#5DF0: 02 5E FE 51 CC 02 5E FE   #5EF0: 00 3E 00 0E 73 D7 3E 07
#5DFF: 71 CC 02 5E 18 D3 AF 3E   #5EFF: 0E 73 D7 3E 00 D3 FE C9

#5E00: F4 C9 3A F0 5E FE 00 28
#5E0F: 1F 3D 32 F0 5E CD C1 5E
#5E10: 38 0F 3E 00 D3 FE CD 69
#5E1F: 5E 38 E7 CD 5C 5E 3E 00
#5E20: C9 3A F0 5E 3D 32 F0 5E
#5E2F: 3E 02 D3 FE 3E 00 C9 3A
#5E30: F0 5E FE 7F 28 1F 3C 32
#5E3F: F0 5E CD C1 5E 38 0F 3E

#5E40: 00 D3 FE CD 69 5E 38 E7
#5E4F: CD 5C 5E 3E 00 C9 3A F0
#5E50: 5E 3D 32 F0 5E 3E 02 D3
#5E5F: FE 3E 00 C9 0E 2B 06 1B
#5E60: 11 00 00 21 00 40 D7 D8
#5E6F: C9 21 D8 5E 11 B7 5E 01
#5E70: 03 00 ED B0 21 B7 5E 7E
#5E7F: FE 73 20 39 23 7E FE 63

#5E80: 20 33 23 7E FE 72 20 2D
#5E8F: DD 21 D0 5E DD 7E 0E 11
#5E90: 00 1B BB 20 20 DD 7E 0F
#5E9F: BA 20 1A DD 21 D0 5E DD
#5EA0: 7E 0B CB 47 28 0F CB 57
#5EAF: 20 0B CB 67 20 07 CB 6F
#5EB0: 20 03 37 3F C9 37 C9 00
#5EBF: 00 00 0E 8A D7 32 F0 5E
```

Как в случае ассемблирования/линкования, так и в случае набора "дампа", для обеспечения работоспособности и дальнейшего контроля целостности следует сосчитать и установить контрольную сумму полученной программы (например, с помощью служебной утилиты "calc.com", "calc.res" или "ch+2.com"). В оригинале, если исходный текст не менялся, контрольная сумма должна быть равна 33464 (#82B8). Нелишним будет также установить "родную" дату в служебном заголовке (утилитой "rename.com") — 22.3.04.

Работа с приведенной программой предельно проста. Установив панельный курсор на любом из экранных файлов в текущем каталоге (только с расширением имени "scr" и длиной 6912 байта — упакованные с помощью "spacker.com" экраны пока не поддерживаются), нужно вызвать просмотрщик любым стандартным для системы iS-DOS способом: по "горячей" клавише, из монитора командной строки или из пакетного "bat"-файла. "Вьюер" выведет на экран графический файл "из-под курсора", а перейти затем к просмотру ниже- или вышележащих в каталоге графических файлов можно будет с помощью курсорных клавиш (стрелок) "Вниз" и "Вверх", а также "A" и "Q". Если не отпускать управляющую клавишу, то переход от одного файла к другому будет выполняться без задержки (мультипликационный эффект). Это позволяет быстро "перелистывать" экраны, скроллируя весь список файлов в текущем каталоге. Так, например, переход от первого файла к последнему в списке из 120 экранных файлов на винчестере выполняется за время около 4 с (на компьютере KAY-1024 turbo). О достижении крайнего экранного файла в каталоге бордюр сигнализирует красным цветом. Файлы, не относящиеся к стандартному (с точки зрения программы) экранному формату, игнорируются. Выход из режима просмотра стандартный для системы iS-DOS — "откат" по комбинации клавиш <Symbol_Shift>/<A>.

Пользоваться этой утилитой можно, описав ее вызов в уже упоминавшемся конфигурационном текстовом файле "extview.txt", который находится в системном каталоге SHELL. Туда же, в системный каталог, можно поместить и сам групповой просмотрщик. В этом случае можно порекомендовать добавить в "extview.txt" следующую строку:

```
#:Q:SHELL\sviewer,
```

что соответствует той же клавише <3> (View) системного меню, что и для "exescr.com", но только вместе с регистровой клавишей <Symbol_Shift>.

Готовую программу "sviewer.com" и еще несколько прикладных iS-DOS'ых утилит, не входящих в стандартные поставки фирмы (c)IskraSoft, можно найти в первом выпуске любительского электронного сборника "IS-files", распространяемом на дискете в формате iS-DOS. Условия распространения — в 17-м номере синклеровской газеты "Абзац", на с.2., либо в маркированном конверте, присланном по адресу, указанному в начале этой публикации.

Литература

1. Скутин В.Г. Информационное пространство Sp-платформы. — Радиолюбитель, 2002, №2, С.18.

2. Очков В.Ф., Пухначев Ю.В. 128 советов начинающему программисту. — М.: Энергоатомиздат, 1992.

А.ВЕНДИЛОВСКИЙ,
г.Минск.

Painkiller

Несмотря на ангельский свет и нимб над головою, сержант остается всегда сержантом, даже в раю. Тем более, если это сержант Воинства Неба.

— Что это за хрень мне досталась на этот раз? — он недовольно рявкнул на строй. — Вы, видимо, вообразили себя супергероями, спасающими мир от зла? За вечность я перевидал множество засранцев, но такое вижу в первый раз! Может там, на Земле, вы и были крутыми бойцами, но здесь вы никто и ничто! — Последние слова он буквально прорычал в лицо одному из новобранцев, и сержантский нимб, казалось, светился весьма угрожающе. — Да самый дохлый бесенок сделает вас одной левой! Запомните, мне не нужны здесь расфуфыренные херувимчики! Я протащу вас по тренировкам, после которых вам Ад покажется Раем! Денно и нощно вас будут учить, как должны сражаться ангелы — свинцом и словом Божьим! Вы будете петь "Аллилуйя!" и поражать десять целей в минуту! Теперь вы приписаны в полк Архангела Гавриила, это самый лучший полк за всю вечность! Он сбросил Люцифера с Небес во время Первой Войны. Помните это! Сам Старик будет принимать у вас экзамены в конце тренировок. И если... — тут голос сержанта стал смертоносно тих и опасен — и если мне придется краснеть за вас перед ним, я лично позабочусь, чтобы провинившийся чистил зубной щеткой сортиры от меня и до Конца Света! Ну что уставились на меня, охломоны, живо на поле тренировок!

— Да сэр! — рявкнул отряд новопризванных ангелов и как можно скорее полетел к световому порталу. Рядом с сержантом вспыхнул ослепительный свет, и две фигуры стали рядом с ним.

— Мы не сможем остановить их, — сержант устало помотал головой,— после того как талисман оказался в руках Преисподней, мы долго не протянем. Нас уже выбили из потустороннего мира. Эти новобранцы — не чета старой гвардии... — Сержант посмотрел на Архангела Гавриила. Вы пытались связаться с Люцифером?

— Люцифер больше не командует силами Ада. — донесся глубокий голос второй фигуры с множеством лиц. — Когда его демонам попала в руки первая часть Талисмана, она свела их с ума, и теперь вряд ли он сможет остановить начавшееся. Теперь их желания подстегиваются мощью Талисмана, но если их не остановить, то скоро Врата Небес рухнут под напором дьявольских сил, и все миры вовеки погрузятся во тьму.

— Мы должны призвать Хранителя Талисмана. — Было видно, что сержант давно обдумывал это предложение, но не решался произнести его вслух. — Он был крутым бойцом, и только ему по силам переломить ситуацию! Я...

— Но он совершил грех похуже Люцифера! — Гавриил был в ярости, — и в том, что сейчас происходит здесь, есть и его вина! Вызвав его, мы накличем еще большие бедствия, или ты забыл, сержант, ЧТО было тогда?

Они сверлили друг друга глазами, пока тот, кого называли Сыном Божьим, не принял решение...

* * *

Визг тормозов, слишком поздно — это я еще успел понять, когда огромный грузовик на всей скорости ударил в мой джип. В какое-то мгновение я даже успел увидеть, как сминается железо и плоть, кровь ударила во все стороны, заливая багровой струей чудом не вылетевшее лобовое стекло и крышу автомобиля. Боль была хотя и мгновенной, но очень сильной. И все окутал мрак... Так вот как умирают... С некоторым изумлением я посмотрел со стороны на свое изувеченное тело. Была определенная обида — это тело послужило мне верой и правдой, и как любой профессиональный солдат, который всю свою жизнь посветил армии, я гордился столь натренированным телом. Я не получал удовольствия от смертей, просто лучше всего у меня получалось сражаться и защищать, словно навыки бойца прочно угнездились в моей крови. Теперь же я летел все выше и выше по темному туннелю, к свету, где меня будет ждать вечный отпуск и покой. Но так же внезапно я со всего маху налетел на барьер. Там был свет! И меня туда не пускали! Словно тысячи голосов зашептали мне в уши. Сожаление, понимание, что не

заслужил, что грехи на руках моих, и что нужно решение Самого... потому-то... Я словно увидел тысячами глаз, как толпы демонов рвутся к Раю, как отчаянно сражаются небесные воины, но перевес явно не на их стороне, и невообразимое число ртов в отчаяньи воззвало ко мне. Всколыхнулось странное и смутное воспоминание о Талисмане, который попал в руки демонов, и невообразимая ярость, неизвестно откуда взявшаяся, исторглась из меня. Меня кружило в гигантском водовороте света и тени, пока я не пришел в себя здесь, в Лимбо.

Путь людей заканчивается на кладбище. По иронии судьбы, для меня он отсюда начался. Обломки разломанных гробов и покрошенные могильные камни, словно чья-то чудовищная пасть щерилась в ночь. Туман скрывал окрестности, и мрачный заунывный вой пронесся под небесами в свете полной луны. Все так напоминало мне мой мир, что я, грешным делом, подумал, что воскрес — я чувствовал как человек, видел как человек, все было так знакомо... и чуждо. Я ущипнул себя и почувствовал боль, что-то подсказало, что здесь я тоже смертен, и мир этот находится между Землей и Небом (хотя под описания Чистилища он мало подходил). Всю средневековую пастораль напрочь портил впечатанный в землю силуэт погибшего ангела. У меня волосы зашевелились на голове от непонятного испуга, я чувствовал, что что-то во мне знает, почему ангелы могут погибать, и это знание рвется наружу. Рядом с павшим лежало его ружье, весьма оригинальный образчик оружия ближнего боя. От всех этих размышлений меня оторвал грохот разваливающихся могил. Граждане, со всех ног бежавшие ко мне, могли спокойно играть в массовке "Ночи живых мертвецов" а уж в "Восставших из Ада" им была гарантирована роль первого плана. Я печенкой ощущал, что эти скелетики бегут ко мне не для того, чтобы по-братски обнять. Поэтому, недолго думая, я нажал на спусковой крючок. Световая плеть хлестко ударила в эти порождения Ада, превращая падшие души в труху. А через миг на том месте вспыхнуло странное сияние, которое перетекло в меня, наполняя силой и могуществом. Я засмеялся, решение пришло само: пока небесам некогда мной заниматься, я тут устрою маленькую войну и стяну Талисман, думаю, это освободит меня от всех грехов. Тысячи шагов... С двух шагов из винчестера выстрел в голову твари, что пыталась швырнуть в меня топором... Чудом увернувшись от косы, пригвоздил осиновым колом еще и этого летающего отморозка, вообразившего себя Жнецом... Вся это смертопляска происходила в бешеном темпе, но с каждым выстрелом я становился все ближе к цели. Эй, Ад! Я уже иду!

* * *

Такие игры, как хорошее блюдо, надо правильно подать. Начните играть поздно вечером, выключите в комнате свет, настройте колонки, а если соседи жалуются, то оденьте наушники. Рядом поставьте стакан с валерьянкой и приступайте. Давно ли нас баловали качественными horror-играми? А еще с убийственным геймплеем, который легко переплюнет любого Серьезного Сэма? Хватило пальцев на одной руке? То-то же! Игра затягивает как наркотик, безумный слэш под смертоубийственный готический андеграунд, от которого у многих голливудских режиссеров от зависти начнется изжога. Все это приправлено потрясающим геймплеем и завернуто в красивейшую упаковку.

Начнем, как обычно, с движка игры. Что-то в последнее время количество качественных и мощных движков стало расти как на дрожжах, что не может не радовать. Правда, владельцам первых и вторых, а также значительно урезанных четвертых GeForse придется рыдать в отчаяньи, питать черную зависть к владельцам более мощных моделей видеокарт и судорожно собирать деньги на апгрейд. Не спорю, запустить игру можно и на таком железе (с десятой попытки, поставив все настройки на минимум), но от полученной картинки хочется просто плакать навзрыд. Зато КАКАЯ картинка получается на хорошей видюхе! На каждую модель не жалели полигонов, сам антуражик сделан с большой любовью, все детали игрового мира, которые должны вызывать страх и омерзение, вызывают именно страх и омерзение и не кажутся пенопластовыми декорациями из третьесортного американского ужастика. При этом компьютер с параметрами далеко не топовых моделей прекрасно справляется с максимальными настройками в графике, даже при большом скоплении на сцене противника количество fps не падает ниже предела, когда игра становится невозможной.

Не последнее место в формировании атмосферы игры, простите за ка-

вас в бешеный ритм, где звуки гитары и ударника смешиваются с криками и звуками выстрелов.

Монстрятник тоже просто "радует" глаз — такого количества отвратительных морд я уже давно не видел :). Как и полагается в таких играх, ума в них, как в тапочке, зато количеством берут просто "на ура". Иногда возникает ситуация, когда противник начинает драку со своими соседями, лишь бы пробраться к тебе, любимому, и стукнуть тебя хорошенечко. Так что приходится носиться по уровню как угорелому, стреляя во все, что движется. Номинально оружия только восемь наименований, но каждое имеет несколько вариантов стрельбы — как вам нравится арбалет с осиновыми кольями и подствольным гранатометом? А тут разработчики еще одну фишку предложили — займись селекцией, пойди по стопам Мичурина, и скрести гранатомет с винчестером. Весьма смертоносный гибрид получается! В общем, фантазии и рукам дан зеленый сигнал — изготовь любимую пушку своими руками.

О системе сейвов хочется сказать отдельно. Никогда еще я не встречал такой заботы об игроке. Записать игру можете в любой момент, но для тех, кого поглотил азарт, и кто забыл записаться сразу после тяжелого боя, на каждой развилке, перед каждой очередной разборкой вы увидите на полу чек-поинт в виде пентаграммы, вступив в которую, вы дадите команду сделать автосейв игры. Кстати, игра сохраняет ВСЕ такие автоматические сохраненки, к которым вы можете вернуться в любой момент.

Управление в игре, можно сказать, классическое, так что, если человек хоть раз в жизни играл в экшен, то сразу найдет необходимые клавиши. А геймплей... Игру хочется проходить по нескольку раз, одному и по сети, что, скажем, в нынешние времена достаточная редкость.

ламбур, играет звук. Мрачные грегорианские напевы и музыка в стиле "Gothic", сдобренные всевозможнейшими потусторонними завываниями, пробуждают в подсознании все самые забытые атавистические страхи перед темнотой. Напряжение возрастает, и вы невольно шарахаетесь в сторону, когда вдруг слева или справа раздается пронзительный заунывный вопль падшей души, руки сами тянутся к валерьянке и валидолу, и холодный пот проступает на спине. Когда на сцену выскакивают монстры, тяжелый рок впрыскивает в кровь адскую смесь адреналина, и музыка заводит

Системные требоеания

Minimum

OS — Windows 98SE/ME/2000/XP.

CPU — 1.5 GHz Pentium III or AMD Athlon processor.

RAM — 384 MB.

CD-ROM/DVD-ROM — 8X speed.

Hard Drive Space— 1,2 GB available.

Video — 64 MB DirectX 8.1 card with hardware Transform & Lighting support (NVIDIA GeForce2 GTS or better).

Sound — DirectX 8.1b or better compatible.

Input — Keyboard and mouse.

Recommended:

OS — Windows XP.

CPU — 2,4 GHz Pentium 4 or AMD Athlon processor.

RAM — 512 MB.

CD-ROM/DVD-ROM — 8X speed.

Hard Drive Space — 1,2 GB available.

Video — 128 MB DirectX 9 card, NVIDIA GeForce FX 5700 or better.

Sound — DirectX 8.1b or better compatible.

Input — Keyboard and mouse.

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***

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.Куйбышев, 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.

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

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 — 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 76, корп. 4;

- для жителей Беларуси —
получатель: УП "РЛД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" в г.Минске код 121.
Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.

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

На УКРАИНЕ:

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

В КАЗАХСТАНЕ:

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

В БЕЛАРУСИ:

В Минске в магазинах **"Книга XXI век"**, пр.Ф.Скорины, д.92, тел. (017) 264-27-97 (ст.метро "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