9•2001
http://radio-mir.com
Индексы: 48925, 45995

радиомир
Ваш компьютер

В НОВЫЙ ВЕК — С НОВЫМ НАЗВАНИЕМ!

РУССКАЯ и Английская версии
redhat 7.0
LINUX
Лучшая операционная система среди UNIX-систем!

ТОЛКОВЫЙ СЛОВАРЬ РУССКОГО ЯЗЫКА
С.И. Ожегова и Н.Ю. Шведовой
Содержит около 40000 слов

TeachPro Internet
ОБУЧАЮЩАЯ СИСТЕМА ПО INTERNET
Подарок в каждым компактом

ВСЕ для СЕТЕЙ
СЕТЬ
Сетевые утилиты и программы
100%
Управление сетями
FTP клиенты
Серверы, прокси серверы
Защита сетей
Утилиты для тестирования сети
ALEX SOFT

виртуальный Доктор
Медицинская библиотека
Справочник лекарственных средств v2 03
Integrative Medicine ACCESS Patient Information
REAL PEOPLE MEDICAL LIBRARY VOL 1
Materialise Surgicase Dental v2 0
Клинические классификации внутренних заболеваний
ALEX SOFT

Вирусов нет!
мультисистемный загрузочный диск
REANIMATOR
NEW!
2001
Classic
Операционные системы, Оффис
Антивирусы, архиваторы
Файловые менеджеры
Мультимедиа
Утилиты
Bootable CD

русская и английская версии
Macromedia
FLASH 5
Профессиональный пакет для создания WEB-сайто
+ русский учебник

В.Ермоленко. Персональный развлекательный коммуникатор

Учредитель и издатель:
ООО "НТК ИНФОТЕХ"

Свидетельство о регистрации
ПИ №77-7399 от 19.02.2001

Главный редактор
Ольга СТРЫЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

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

Контактные телефоны:
(095) 105-99-89,
(+375-17) 249-41-47

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА,
Игорь ШЕВКУН

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

Отдел рекламы — **ООО "Альтекс-А"**
тел. (095)336-13-58

Отдел экспедирования и рассылки журналов
— **Татьяна ЖУКОВСКАЯ,**
тел/факс (+375-17) 221-43-02,
(+375-29) 677-39-43

Адреса для писем:
141406, РФ, г.Москва, Химки-6,
ул.Библиотечная, 18-84;

220095, РБ, г.Минск-95, а/я 199

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

Наши платежные реквизиты:
получатель: ООО "НТК ИНФОТЕХ",
ИНН 7703155561,
р/с 40702810100022120172
в АКБ "Межтопэнергобанк"
корр. счет 30101810900000000237
БИК 044585237
Адрес банка: 107078, г.Москва,
ул.Садовая-Черногрязская, 6

Материалы для публикации принимаются в рукописном, печатном и электронном вариантах

Требования к графическим материалам рекламного характера в электронном виде: CorelDRAW 6.0, 7.0 все шрифты в кривых, bitmaps 300 dpi; TIFF, 300 dpi; CMYK в сопровождении печатной копии

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

Адрес редакции: 141406, г.Москва, Химки-6,
ул.Библиотечная, 18-84, тел. (095) 105-99-89

Подписано к печати 6.08.2001 г.
Формат 60 x 84 1/8.
Печать офсетная. 5,5 печ. л.
Цена свободная

Отпечатано в типографии ЗАО "Красногорская типография".
Заказ №2422. Тираж 1700 экз.

ВНИМАНИЕ!
НАЧИНАЯ С №7/2001 ЖУРНАЛ ВЫХОДИТ ПОД НОВЫМ НАЗВАНИЕМ
(подписка на "Радиолюбитель. Ваш компьютер" действительна. Подписные индексы сохранились)

радиомир Ваш компьютер

Ежемесячный массовый журнал.
№9/2001. Издается с сентября 1995 г.

ЧИТАЙТЕ В НОМЕРЕ:

ПЕРСОНАЛЬНЫЙ РАЗВЛЕКАТЕЛЬНЫЙ КОММУНИКАТОР

В.ЕРМОЛЕНКО,
г.Минск.

За последние пару лет стали все шире распространяться малогабаритные устройства, в той или иной степени заменяющие собой "полноразмерный" настольный компьютер. Их называют персональными цифровыми помощниками (PDA — Personal Digital Assistant), и одним из наиболее популярных представителей устройств этого класса является Palm. PDA позволяют исполнять аналоги обычных офисных программ (таких как текстовый редактор, электронные таблицы, базы данных и электронная почта) находясь в дороге или на природе, а затем пересылать данные в обычный компьютер. Для присоединения к сети Internet, конечно, можно использовать телефонную линию и модем, однако в мобильных условиях для этого потребуется, например, сотовый телефон с протоколом WAP. В итоге, удобство малогабаритного помощника становится дорогим удовольствием.

Однако неожиданно среди подобных устройств появилось нечто совсем иное, обещающее перевернуть все представления о том, что должны уметь устройства такого класса. Речь идет о Cybiko.

Предназначая его в первую очередь школьникам от 12 до 16 лет, изготовители называют Cybiko коммуникационным развлекательным компьютером. На нем работают игры, офисные приложения, простые звуковые и графические редакторы, переводчики. Но самое главное — Cybiko находит по радио аналогичные устройства вокруг себя и соединяется с ними. Важно подчеркнуть, что локальная сеть организуется и поддерживается автоматически, и все услуги по пересылке информации оказываются бесплатными. Пользователь платит при покупке только за само устройство — и все. Получается гораздо дешевле, чем подписка на услуги сотовой связи.

С конца 1999 года в США проданы сотни тысяч устройств Cybiko. Летом 2001 г. его можно было купить за $69.95 (BestToys.com), либо непосредственно от производителя за $99.95 вместе с приставкой — 8 Мб MP3-плеером, который вначале продавался отдельно за $99.95 [1]. Как отмечают обозреватели [2], из беспроводных устройств дешевле только пульты управления к телевизорам...

Как это выглядит

Cybiko (рис.1) имеет прозрачный корпус из разноцветной пластмассы. Цвета, как вы догадываетесь, носят очень "современные" названия — "электрошоковый желтый", "плазменный пурпурный", "магический прозрачный" и т.п.

Рис. 1

Табл. 1

Технические характеристики:	
Процессор	Hitachi H8S/2246 (32 бит, 11 МГц)
Сопроцессор	Atmel AT90S2313 (4 МГц)
Память	1 Мб, расширяемая до 8 Мб
ЖКИ-дисплей	160x100 точек, 59x40 мм, 4 уровня серого
Разъем картриджа расширения	68-контактный
Разъем соединения с PC	Последовательный порт RS232
Клавиатура	69 кнопок
Размер	145 x 72 x 22 мм
Вес	120 г
Радиопередатчик	RF2915
Диапазон частот	902...928 МГц
Количество цифровых каналов	30
Скорость передачи данных	19200 бит/с
Дальность связи	50 м в помещении, 100 м вне здания (зависит от окружения)
Максимальное количество устройств в сети	3000 (100 устройств на каждом из 30 каналов)
Встроенные устройства	Вибрационный сигнализатор, громкоговоритель, выдвижная антенна
Комплектность	NiMH-аккумулятор (прямоугольный, типоразмер F6, емкость 700 мА·ч), сетевое зарядное устройство, соединительный кабель RS232, пластиковый стилус
Минимальные требования к IBM-совместимой системе	Pentium 90 или выше, Microsoft Windows 95/98/NT 4.0 (SP5), 32 Мб ОЗУ, мышь, Microsoft Internet Explorer 4.0, Netscape Navigator 4.0 или более поздние, один свободный порт RS232, 20 Мб свободного места на жестком диске, возможность подключения к Internet по Dial-Up или LAN

Технические данные Cybiko приведены в табл.1. Да, по сравнению с современными PC, цифры выглядят не очень впечатляюще. Однако, это устройство карманных размеров, а 32-разрядный процессор Hitachi во многих отношениях ближе к Motorola 68000, чем к Intel x86. Тем, кто привык мыслить в терминах 8-битной шины, покажется невероятным, что практически все арифметические команды выполняются за 1 машинный цикл, умножение — за 3 или 4, а деление — за 12 или 20. Операции со знаком требуют на 1 цикл больше.

Cybiko снабжен обычным последовательным портом, связывающим его с компьютером. Как видно из табл.1, минимальные требования к IBM-совместимой системе делают его довольно доступным.

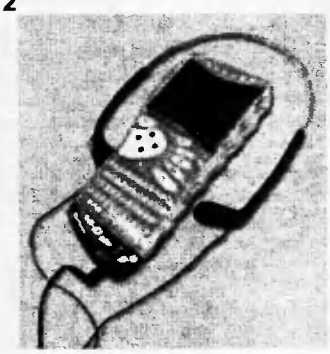
Радиопередатчик Cybiko работает в диапазоне 900 МГц, выделенном для сотовых телефонов, поэтому присутствие даже нескольких устройств в помещении не нарушает работу "настоящих" радиомодемных сетей по протоколу 802.11.

По словам автора обзора [2], устройство не лишено недостатков. Оно достаточно медленно загружается, и управлять им не всегда удобно. В условиях города объявленная дальность связи реализуется далеко не всегда. Среди других недостатков первой версии пользователи [3] отмечают отсутствие подсветки дисплея и слишком маленькие клавиши, что является платой за малые габариты и дешевизну.

Одним из важнейших дополнений к устройству является картридж с MP3-плеером (показан в подключенном виде на рис.2), который включает в себя также ЧМ-радиоприемник, микрофон для записи голоса, наушники, слот расширения памяти с помощью карточки SmartMedia (приобретается отдельно) и порт USB (типа B). Порт USB может использоваться для более быстрого программирования Cybiko и пересылки MP3-файлов.

Осенью 2001 г. планируется появление радиомодема с возможностями присоединения к стандартным сетям RadioEthernet, сотовым сетям и др.

Рис. 2



Без проводов

Как уже говорилось, вхождение в сеть осуществляется автоматически, а о каждой установленной связи Cybiko сообщает звуковым сигналом или вибрационным моторчиком. Подъезжая к школе, обладатель Cybiko по количеству вибраций может определить, сколько его друзей уже приехали.

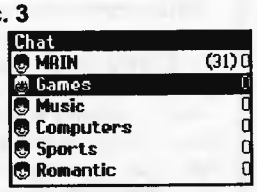
Устройство идентифицируется семисимвольным индивидуальным кодом, который указан на корпусе. В память Cybiko можно занести сведения о владельце (профиль), которые смогут прочитать все подключившиеся к нему пользователи. Оригинальной является технология Friend Finder (поиск друга) — если среди обнаруженных вокруг Cybiko находится кто-то, подходящий для вас, устройство отметит это звуковым сигналом, а на экране появится знак в виде одного, двух или трех сердечек (чем точнее подходит профиль, тем больше сердечек). Для эффективного поиска Cybiko сканирует всю радиосеть каждые 60 секунд.

Для занесения в профиль своей фотографии придется воспользоваться помощью компьютера — картинка из обычного графического формата сначала преобразуется в вид, удобный для отображения на маленьком черно-белом дисплее, а затем пересылается в Cybiko.

Если вы не хотите указывать в профиле свой точный возраст, можно выбрать для себя одну из групп: A — до 13 лет, B — 14...17, C — 18...29, D —30 и старше. По выбору, можно указать свой рост, хобби, желания и т.д. Но инструкция [4] не рекомендует раскрывать слишком личную информацию для всеобщего обозрения.

Для чата Cybiko предлагает выбор из нескольких "комнат" по темам — главная, игры, музыка, компьютеры, спорт, романтика (рис.3). Сообщения можно высылать как всем, так и конкретному адресату. Поскольку дальность связи ограничена и зависит от внешних условий, Cybiko предпринимает несколько попыток доставить частное сообщение, а в случае неудачи помечает его как недоставленное.

Рис. 3



Для удобства пользователя автоматически ведется адресная книга, где содержится список всех людей вокруг, а находящиеся в вашей "комнате" выделяются жирным шрифтом. Чтобы сократить набор на довольно неудобной маленькой клавиатуре, используются сокращения и кодовые фразы. Инструкция к Cybiko [4] предлагает собственный, основанный на традициях Internet-чатов и электронной почты, набор сокращений. Поскольку компьютер рассчитан на подростков, объем лексикона позволяет нам привести таблицу кодовых фраз полностью (табл.2). К тому же, в памяти можно сохранить целых 10 наиболее употребительных фраз, чтобы впоследствии не набирать их полностью.

Каждый владелец Cybiko получает бесплатный адрес электронной почты, соответствующий идентификационному коду его устройства. Так, если код устройства AAAAELA, то e-mail-адрес — AAAAELA@cybiko.com. Пользоваться электронной почтой можно как по сети Internet через сайт www.cybiko.com, так и прямо из самого устройства. Во-первых, можно писать письма для отправки впоследствии, когда Cybiko будет на время подключено к компьютеру, присоединенному к Internet. Во-вторых, подключенный к такому компьютеру Cybiko сам становится портом в Internet (CyWIG — Wireless Internet Gate) для тех Cybiko, которые соединены с ним радиосетью. Таким образом, при наличии в здании одного соединенного с Internet Cybiko, остальные могут пользоваться доступом к Internet, отсылать электронную почту, загружать файлы и т.п. практически неограниченно.

Работа с веб-сайтом www.cybiko.com упрощена до предела. Достаточно присоединить выключенный Cybiko кабелем к последовательному порту компьютера, соединенного с Internet, и включить его — все остальное Cybiko сделает сам. Весь сайт с доступными для загрузки файлами будет виден как папка на вашем устройстве.

В режиме CyWIG Cybiko становится сервером. Для нормальной работы его следует перезагрузить (вот почему неплохо иметь два устройства, одно из которых будет постоянно служить сервером).

"Офисные" приложения

На Cybiko установлены адресная книга, органайзер, текстовый редактор и дневник, музыкальный и графический редакторы, калькулятор, будильник. Новые приложения можно регулярно загружать с сайта компании.

Органайзер напоминает вам о делах, запланированных на определенную дату и время. Задачи сортируются по дате и приоритету (высокий — +!, средний — +, низкий — не помечается), при этом можно определить ежедневные, еженедельные или ежемесячные задачи.

Текстовый редактор со встроенным дневником позволяет, кроме редактирования обычных текстовых файлов, организовывать свои записи по дням в виде журнала.

Музыкальный редактор позволяет редактировать одноголосые мелодии и сохранять их не только на своем устройстве, но и пересылать их участникам сети. На экране музыкального редактора изображен нотный стан и кнопки выбора транспозиции (первая/вторая октава) и темпа (медленный, средний, быстрый), а также кнопка останова. Белым клавишам соответствуют кнопки ZXCVBNM (первая октава), QWERTYU (вторая), а черным — SDGHJ и 23567 соответственно. Пауза — пробел.

Графический редактор позволяет редактировать изображения размером до полного экрана (160 x 100 точек) в 4 градациях серого (белый, светло-серый, серый и черный). Рисовать можно пером либо заливкой, причем перо рисует точками, пунктиром или линиями, а толщина пера варьируется от 1x1 до 8x8 точек.

Адресно-телефонная книга может содержать записи о 100 друзьях — телефон, адрес, e-mail, ICQ и краткая информация. Кстати, именно столько Cybiko может находиться одновременно в сети. Визитные карточки могут заноситься в адресную книгу автоматически.

Часы в Cybiko продолжают идти, даже если устройство выключено. Единственный способ полностью прекратить разряд аккумуляторной батареи — это извлечь ее из Cybiko. Будильник, однако, в выключенном Cybiko не работает. Часы все время видны внизу рабочего стола, но их можно развернуть на весь экран, нажав клавишу пробела.

Встроенный калькулятор выполняет простейшие арифметические операции, но при необходимости можно загрузить себе и более сложный. С сайта можно также загрузить разговорники, содержащие перевод английских фраз на различные языки, орфографический словарь и другие полезные приложения.

Игротека

Наверное, главной маркетинговой находкой Cybiko стало постоянное предложение новых игр. Ежедневно на сайте компании появляется новая игра для бесплатной загрузки. Именно так — каждый день новая игровая программа бесплатно. Как отмечает автор обзора [5], пока основная задача разработчиков Cybiko — добиться, чтобы это устройство стало "модным". Впоследствии можно будет и взимать небольшую плату за каждую загрузку.

Основное преимущество игр на Cybiko состоит в том, что в них можно играть нескольким игрокам, причем радиосеть позволяет вступать в игру, даже не зная, кто твой соперник. Персонажи игр могут "переселяться" к другим игрокам, так что не удивляйтесь, если вы вдруг обнаружите новых существ и новые возможности в своем Cybiko. Вот несколько примеров.

CyLandia

Игра CyLandia, в которой вы должны заботиться о кибернетическом персонаже, по своей идее напоминает игрушку Тамагочи. Ваш новый кибер-друг — Су-В (сай-би) — ест, пьет, осваивает хорошие привычки, зарабатывает деньги, телепортируется и торгует с другими игроками, вступает в брак, отчего появляются новые Су-В... Цель игры — довести своего Су-В до глубокой старости, набирая очки за его "жизненные успехи". Один человеческий день равен одному году в CyLandia, а каждый такой

Табл. 2

Значение	Оригинал (English)	Cybish		Примеры
		Ед.ч.	Мн.ч.	
женский	female	f	ff	**i f** означает "Я женщина (девочка)"
мужской	male	m	mm	**I m** означает "Я мужчина (мальчик)"
человек	person	p, }	pp	**}}}** означает "люди, народ"
я	I, me	i,)	we,)}	
ты	you	u, {	uu, {{	**r u f?** означает "Ты девочка? (Are you girl?)"
письмо	letter	t	tt	
картинка	picture	[]	[] []	
электронное письмо	e-mail	@		
спасибо	thank you	thx		
работа	job	$		
глаза	eyes	ii		**I've blue ii** означает "У меня голубые глаза"
волосы	hair	iii		**I've long iii** означает "У меня длинные волосы"
да	yes	+, y		**+ u right** означает "Да, ты прав(а)"
нет	no	-, n		**i -w** означает "Не хочу"
конечно	sure	++		
		Слово	**Усиленное значение**	
идти, следовать	go, follow	>		**> I** означает "Следуй за мной"
возвращаться, обратно, из	return, back, from	<		
хотеть	want	w	ww	**i w }(** означает "Я хочу поговорить"
скучно	bored			**i zzz**
быть	are	r		
любить, нравиться	love, like	*	**	**I ** u** означает "Я люблю тебя"
разговаривать	chat	}(		
ссориться	umbrage, upset	(-)		
получать	get	g		
делать	do	d		
удовольствие	fun	;	;;	
клевый	cool	#	##	
хороший	good	)	))	
плохой	bad	(	((	
глупый	stupid	[	[[	
симпатичный	neat	~		
хорошенький	cute	~~		
сумасшедший	crazy	%		
привет	hello	h		
пока	bye	b		
о'кей	OK	ok, =		
и	and	&		
для	for	4		
к, до	to	2		
что	what	:		
где	where	::		
когда	when	:::		
твой	your	yr		
пожалуйста	please	pls		
извини	sorry	rr		
добро пожаловать	welcome	wl		
возраст/пол/местонахождение	age/sex/location	asl		
громкий смех	laugh out loud	lol		
не имеет значения	never mind	))		
молодец	good boy	(}		
лично	person to person	p2p, }2{		**I w }(p2p** означает "Я хочу поговорить наедине"
Привет всем!	Hello everybody!	h pp!		
Хочу поговорить	I wanna chat	i w }(		
Ты откуда?	Where are you from?	:: r u <?		
Пришли свой возраст/пол/местонахождение, пожалуйста	Send your age/sex/location, please	yr asl pls		
Я новичок в этом чате	I'm new in this chat	} new in }(		
Мне скучно	I'm bored	i zzz		
Чем ты развлекаешься?	What do you do for fun?	: d u d 4 ;?		
Я тебя очень люблю!	I love you very much!	I ***** u!		

год состоит из 4 дней и 4 ночей. Когда на экране появляется Луна, Су-В пора спать. Вначале вам придется помогать своему Су-В, но затем он или она научатся обслуживать себя самостоятельно. Су-В, происходящий от разумных родителей, также будет способен хорошо обучаться.

Игра начинается с небольшой суммы кибер-денег — Су-Баксов, запас которых можно пополнить, если ограничивать затраты, покупать акции, и даже продавать товары другим игрокам. Газета CyLandia News печатает объявления о приеме на работу, предложении акций, ожидаемых нехватках Су-Бургеров и т.п. Телепортируясь в другие устройства, Су-В имеет шанс встретить будущего супруга. Начав игру, остановить ее уже невозможно, пока ваш Су-В жив. Игра продолжается и тогда, когда вы используете другое приложение, и даже когда ваш Cybiko выключен.

Су-В живет в квартире (рис.4). Первые два года он маленький, хрупкий и не имеет ушей. Присматривать за Су-В рекомендуется часто, не реже 4...8 раз в день, в особенности пока они маленькие. Если вы успешно его тренируете, первый период может закончиться и раньше. В каждом периоде вам предлагается список действий, к которым важно приучить своего Су-В. Счет возрастает каждый раз, когда Су-В выполняет нужное действие самостоятельно, а дополнительные очки начисляются за быстрое освоение задач.

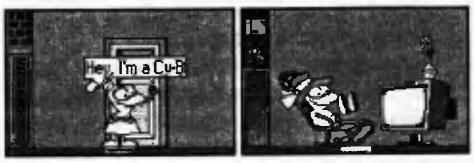
В подростковом возрасте Су-В становится крупнее, и должен привыкнуть убирать комнату, читать книги, заниматься спортом, спать ночью, лечиться, оплачивать счета за электроэнергию, воду и транспорт, а также общаться с другими Су-В. В 15 лет ему уже разрешается работать.

Когда у Су-В появляются уши, он становится взрослым. Теперь он сам способен телепортироваться в другие Cybiko. Если вы плохо заботились о нем, он может вообще вас покинуть. Следует поощрять ежедневный обмен визитами, так как это дает вашему Су-В шанс создать семью.

Вступив в брак, Су-В женского пола может обзавестись ребенком. Заботьтесь о нем так же, как и о своем первом Су-В — правда, теперь потребуется больше денег, чтобы всех прокормить.

Между прочим, заработанные в игре виртуальные Су-Баксы иногда можно обменять на веб-сайте компании на совершенно реальные подарки и скидки.

Рис. 4



Потерянные в Лабиринте

В этой игре вы — CyBerg, задача которого — найти утерянную в Лабиринте Спираль Знаний, чтобы достичь беспредельного могущества. Продвигаясь по коридорам с помощью собственной сообразительности и оружия, вы можете телепортироваться с уровня на уровень. Но будьте осторожны — вы не единственный CyBerg в Лабиринте. В вашей игре могут участвовать другие владельцы Cybiko.

Спираль Знаний находится на высшем уровне — нулевом. Игра начинается с уровня 99. В начале игры у вас 12 жизней, но если вы потеряете их все, то провалитесь обратно на уровень 99. За время каждой сессии вам необходимо найти Телепортер, для того чтобы подняться на следующий уровень. Если вы соединяетесь с другим Cybiko, Лабиринт расширяется, и в нем появляется Гипертелепортер, способный телепортировать на 11 уровней за один раз. Обычно телепортеры поднимают вверх, но иногда они портятся, и вы можете оказаться ниже. Чем выше уровень, тем меньше времени остается для поиска телепортера. По дороге полезно собирать разные припасы (оружие, патроны к нему, золото, ключи для открывания дверей), или же забирать их у побежденных вами CyBerg'ов. Предусмотрен и вариант на случай пропадания радиосвязи с партнерами.

Биллиард

Игра может вестись против компьютера или против другого игрока, с которым сначала надо установить соединение. Все шары одного цвета, и ваша задача — забить в лузы как можно больше шаров, ударив их другим шаром. Клавишей Tab выбирается шар для удара, а клавишами вправо-влево — направление кия. Сила удара регулируется временем нажатия клавиши Enter, в момент отпускания которой и происходит удар. Шар можно дополнительно еще и подкручивать клавишами Ins и Del.

Среди других сокровищ игротеки — CyBattle (вариант игры Морской бой), несколько вариантов шахмат, уголки, поддавки, гольф, стрельба по тарелочкам, Дартс, и т.д. Вы можете сразиться с космическими пришельцами, управляя спасательным шаттлом, поискать сокровища на архипелаге кровожадных пиратов, сыграть в покер с карточным автоматом. Но всего не перечислить.

Понятно, что возможности карманного устройства ограничены. Для загрузки новой большой игры чаще всего потребуется выгрузить предыдущую. Ограничена и скорость связи. В играх на точность рекомендуется отключать радиопередатчик, чтобы улучшить управляемость.

Талант предвидения

Руководство фирмы Cybiko — выходцы из России, выпускники МФТИ и МГУ, что для американской компании пока еще довольно нетипично. Исполнительный директор компании — Давид Ян, один из ведущих российских IT-бизнесменов. В 1989 г. он основал фирму BIT-Software (впоследствии — ABBYY Software, основные продукты которой — компьютерный словарь Lingvo и система распознавания текстов FineReader).

Разработчики Cybiko нащупали свою рыночную нишу, увидев главную потребность молодежи в общении со сверстниками. Не случайно самыми популярными приложениями Internet стали электронная почта, чаты и ICQ. Та же тенденция наблюдается и с сотовыми телефонами, где растет популярность технологии текстовых сообщений (SMS). По словам Давида Яна [6], технология Cybiko как бы увеличивает коммуникационный радиус человека. В отличие от Internet вообще, она служит для снятия первичного психологического барьера общения — ведь те, с кем установлен контакт посредством Cybiko, уже находятся где-то поблизости.

Собственно, проект Cybiko начался с того, что в 1998 г. два инженера (Давид Ян и Дон Висневски) впервые встретились в Internet-чате, обсуждая проблемы электроники. Затем состоялась их встреча в Чикаго, и через 30 дней после этого новая компания приступила к работе. Штаб-квартира ее находится в пригороде Чикаго, но основная группа программистов работает в России, а производство — на Тайване. Часть акций компании с сентября 2000 г. принадлежит крупнейшей компьютерной сети — America Online, и это дает серьезные средства для "раскрутки". Сейчас разработки для Cybiko ведет целый ряд независимых фирм. Так, компания EhomeRoom из Атланты разрабатывает программное обеспечение для школ — такое, чтобы учитель, входя на урок, сразу видел на своем Cybiko результаты выполнения домашнего задания целого класса. Ожидается сотрудничество Cybiko с ведущими коммуникационными компаниями, в частности, с ICQ и системами сотовой связи.

Любой посетитель веб-сайта компании может сгрузить набор для разработки программ (SDK — Software Developer's Kit), заслуживающий отдельного описания. SDK позволяет каждому, знакомому с языком Си, включиться в число разработчиков ПО для этого устройства. Основу SDK составляет компилятор Си, но имеется также побайтный интерпретатор, руководство по написанию игр и приложений, примеры нескольких игр с комментариями, библиотеки готовых модулей. Многозадачная операционная система, поддерживающая радиосеть, позволяет программисту выполнять большинство сетевых операций (например, пересылку блока данных или нахождение участника сети, наиболее подходящего по персональному профилю в данный момент) с помощью единственного системного вызова, вместо того чтобы заботиться об установлении нужного соединения или о надежности протокола обмена.

Практически, несмотря на бурное развитие технологии PDA и игровых приставок, прямых конкурентов у Cybiko до сих пор нет. Но компания смотрит в будущее.

Лучшее — враг хорошего

По словам Давида Яна, новая версия Cybiko будет рассчитана на более старшую группу пользователей, в первую очередь на студентов. Он будет иметь цветной дисплей и более широкий радиус действия — до 1...1,5 км. Скорость передачи данных возрастет до 1 Мбит/с. Кроме того, каждый компьютер сможет служить ретранслятором сигналов сети, так что сеть сможет покрывать район размером 20...30 км. Экран устройства будет цветным, а его разрешение — соответствовать новым стандартам для сотовых телефонов (smart phones). Процессор также станет гораздо более мощным.

Новая версия будет использовать частоту 2,4 ГГц, что избавит от лицензионных проблем в Европе и Азии. Тем не менее, сообщается, что устройства уже появились на "сером" рынке в Москве [7].

Время покажет, станет ли Cybiko новым молодежным культом. Однако то, что это устройство стало заметной вехой на компьютерном горизонте, уже очевидно.

Литература

1. Sybico, Inc. — www.cybiko.com

2. Edward C. Baig. Funky Cybiko gizmo is no Gen Y gem. — USA TODAY, 20 декабря 2000 г. — www.usatoday.com/life/cyber/ccarch/cced050.htm

3. Rob Flickenger. Cybiko: no strings attached. — O'Reilly Network. 28 марта 2001 г. — www.oreillynet.com/cs/weblog/view/wlg/192

4. Cybiko Model 6411. — On-line Guide.

5. James Heckman. Cybiko plans capture of teen market. — Local Business, 5 июля 2000 г. — www.localbusiness.com/Story/Print/0,1197,NOCITY_219485,00.html

6. Александр Костинский. Cybiko: беседа с Дэвидом Яном. — www.svoboda.org/programs/sc/2001/sc.041001.shtml

7. Евгений Козловский. Cybiko №6, пол — женский, или Здравствуй, мальчик-бананан. — Компьютерра, 2001, №6, С.16-17.

А.ГРИНЧУК,
г.Минск, РИВШ БГУ,
E-mail: gav@nihe.unibel.by

Работа в системе Mathematica 3.0/4.0.
Простейшие вычисления

В предыдущей статье [1] мы рассмотрели основные принципы работы в диалоговом режиме в системе Mathematica. Следующий шаг — применение этих знаний на практике. Мы рассмотрим арифметические вычисления, основные команды для преобразования символьных выражений, а также некоторые особенности определения функций пользователя.

Вычисления с целыми и рациональными числами в Mathematica производятся без потери точности. Например, вычисление 100! приводит к результату: 93326215443944152681699238856266700490715968264381621468592963895217599993229915608941463976156518286253697920827223758251185210916864000000000000000000. Более того, √2 или Sin[3/2] воспринимаются как абсолютно точные выражения, которые не будут преобразовываться в число с плавающей запятой. Так, $(\sqrt{2})^3$ с точки зрения системы равно $2\cdot\sqrt{2}$, но никак не 2.8284... Однако при появлении во входном выражении десятичной точки картина кардинально изменяется. Вычисление 100.0! приводит уже к результату 9.33262×10^{157}. По умолчанию в ответе выводится только шесть значащих цифр, но сами вычисления ведутся с шестнадцатью цифрами. Для вывода всех шестнадцати цифр проще всего поставить курсор после числа в выходной ячейке и нажать пробел.

Естественно, иногда возникает необходимость преобразовать тот же √2 к виду 1.4142... Mathematica позволяет проделать это с произвольной точностью при помощи команды **N**. Возможны два варианта этой команды:

- **N[√2]** производит преобразование с используемой по умолчанию точностью (шестнадцать значащих цифр с отображением на экране первых шести);

N[√2, 40] — с точностью, задаваемой пользователем (в этом примере — 40 значащих цифр).

Если при вычислениях используются числа с различным числом значащих цифр, то Mathematica действует предельно аккуратно — определяется число с наименьшей точностью, и с этой же точностью возвращается результат. Так, при умножении **N[√2]** на **N[√2, 40]** получим результат с шестнадцатью значащими цифрами. Число значащих цифр может также задаваться и пользователем при помощи символа "`". Пример: 2.0`30 — 30 значащих цифр. Более общая запись (часто встречается в выходных ячейках) имеет вид: 1.234`20*^7, и расшифровывается следующим образом: 1.234×10^7, число значащих цифр — 20.

Вычисления с комплексными числами производятся аналогично. Мнимая единица набирается либо с палитры (BasicInput), либо с клавиатуры (**Esc i i Esc**), или просто как **I**. Для комплексных чисел определены команды извлечения действительной (**Re[...]**) и мнимой (**Im[...]**) частей, вычисления модуля (**Abs[...]**), аргумента (**Arg[...]**) и сопряженного числа (**Conjugate[...]**).

Символьные выражения имеют более сложную структуру, поэтому и список команд здесь значительно больше. Чаще других используются команды упрощения выражений **Simplify** и **FullSimplify**. Последняя команда выполняется значительно медленнее, но при этом используются и специальные свойства математических функций (тригонометрических, ги-

Действие	Реализация
Арифметические вычисления	
1) Точные вычисления	Если в представлении числа не используется десятичная точка, то оно воспринимается как точное, вычисления ведутся с сохранением всех значащих цифр: 2+2 6^20 6^200. При этом математические константы (e, π), а также функции от констант и точных чисел воспринимаются как точные числа и в процессе вычислений не заменяются на десятичные числа: √2, (√2)3=2√2
2) Вычисления с заданной точностью	**N[выражение, число значащих цифр]** **N[**6^200, 17**]** = 4.26825223812027 40*10^155 **N[**√2,33**]** **N[**π,100**]**
3) Вычисление с числами с плавающей запятой	2.3*π**/Shift+Enter**, ответ — 7.22566. Если в выражении встречается число с плавающей запятой, все дальнейшие вычисления будут вестись также с плавающей запятой
4) Установка количества значащих цифр	**число`количество_знач._цифр*^степень** 3.14`30*^8. Точность вычислений равняется наименьшей точности используемых чисел
5) Ввод комплексного числа	**действительная_часть + EsclEsc мнимая_часть.** (Мнимая единица набирается также с палитры или как I)
6) Операции с комплексными числами	**Abs[z]** — модуль числа **Re[z]** — действительная часть **Im[z]** — мнимая часть **Conjugate[z]** — сопряженное число **Arg[z]** — аргумент числа
Команды символьного процессора	
1) Раскрыть выражение	**Expand[**(1+x)^3 + (2+x)^4 +(1+y)^2**]** — раскрыть выражение **Expand[**(1+x)^3 + (2+x)^4 +(1+y)^2, (1+x)**]** — раскрыть подвыражения, содержащие (1+x) **Expand[**Sin[2 x]**,Trig→True], TrigExpand[Sin[**2 x]**]** — раскрыть тригонометрические и гиперболические функции в выражении **PowerExpand[**(-x^2*y^4)^(1/2)**]** раскрывает вложенные степени без учета многозначности функций
2) Разложить на множители	**Factor[**x^17 + 1**]** — разложить на множители **Factor[**Sin[2 x] Cos[2 x]**], Trig→True], Trig Factor[**Sin[2 x] Cos[2 x]**]** — разложить на множители с учетом свойств тригонометрических функций
3) Приведение общих членов по степеням переменной	**Collect[выражение, переменная]** **Collect[**(1+y)^2+(2-y)^3, y**]**
4)Упрощение выражений	**Simplify[**x^2 + 2 x + 1**]** — упрощение выражений **FullSimplify[**x (1 + x) Gamma[x]**]** — упрощение выражений с учетом специальных свойств функций
5) Подстановка	**ReplaceAll[**3 x^2 + x^4, x→a**]**, 3*x^2 + x^4/.x→a

перболических, цилиндрических, гипергеометрических и др.). Разница видна при упрощении выражения с гамма-функцией Эйлера: **Simplify[x Gamma[x]]** возвращает исходное выражение, а **FullSimplify [x Gamma[x]]** — возвращает **Gamma[1+x]**. Эти команды находятся также на палитре AlgebraicManipulation. С использованием палитры можно упростить как все выражение, так и его часть. В первом случае при помощи мыши выделяется вся ячейка, затем необходимо щелкнуть левой клавишей мыши по соответствующей кнопке на палитре. Но если в выражении $x^2+2xy+y^2+2xy+3y^2$ вы хотите применить команду упрощения только к трем первым слагаемым, вам необходимо выделить только эти слагаемые и воспользоваться кнопкой на палитре.

Другие часто используемые команды:

- **Expand** — раскрыть произведения и положительные целые степени;
- **Factor** — разложить на множители;
- **Apart** — представляет рациональное выражение в виде суммы членов с минимальными знаменателями;
- **Cancel** — сокращает дроби;
- **TrigExpand, TrigFactor** — аналоги команд **Expand** и **Factor** для работы с тригонометрическими выражениями.

Команда **Collect** позволяет сгруппировать слагаемые по степеням указанной переменной. Например, **Collect[(x+y+1)^2, y]** возвращает результат $1+2x+x^2+(2+2x)y+y^2$.

В символьных вычислениях также широко используются подстановки. В Mathematica они реализуются двумя способами:

- **ReplaceAll[x+a, a→y+1]**;
- **x+a/.a→y+1**.

В обоих случаях получим ответ *x+y+1*, т.е. *a* заменяется на *y+1*.

Говоря о символьных и численных вычислениях, нельзя обойти стороной такой важный момент как определение функций пользователя. Mathematica проявляет при этом исключительную гибкость и предоставляет большой набор команд присваивания. Всего их больше десятка, рассмотрим только самые простые из них.

Для определения функций пользователя в системе Mathematica используются:

- немедленное присваивание — **f[x]=x^2**. При этом функцию **f[x]** можно интегрировать и дифференцировать по **x**, но вычисление **f[y]** или даже **f[2]** приводит к повторению выражения в выходной строке. Т.е. данная функция работает только с переменной **x**;
- немедленное присваивание с шаблоном — **f[x_]=x^2** (с подчеркиванием после каждого аргумента) — лишено этого недостатка. В этом случае **f[...]** может оперировать с любым аргументом (с числом или с символьной переменной), а также допускает интегрирование и дифференцирование по произвольному аргументу;
- отложенное присваивание — **f[x]:=x^2**. Используется только для переменной **x**;
- отложенное присваивание с шаблоном — **f[x_]:=x^2**.

Для демонстрации различия между немедленным присваиванием и отложенным определим две функции: **f[x_]=Expand[x^2]** и **g[x_]:= Expand[x^2]**. А затем применим их к выражению **(x+a)**.

В первом случае (вычисление **f[x+a]**) сначала выполняется операция **Expand[x^2]** во время определения функции, а затем вместо **x** подставляется новое значение аргумента — **(x+a)**. Результат — **(x+a)^2**.

Во втором (**g[x+a]**) применение функции идет после подстановки аргумента (отложенное присваивание) — подставляется новое значение аргумента **(x+a)** и выполняется операция **Expand[(x+a)^2]**. Результат — **a^2 + 2*a*x + x^2**.

Рассмотренные команды и приемы работы по традиции собраны в таблицу. И хотя в ней представлена лишь небольшая часть богатых возможностей системы, этого вполне достаточно, чтобы начать работу с функциями математического анализа в Mathematica. В следующей статье мы познакомимся с командами вычисления пределов, дифференцированием, интегрированием (аналитическим и численным), а также с командами для работы с рядами.

Литература

1. А.Гринчук. Работа в системе Mathematica 3.0/4.0. Интерфейс и входной язык. — Радиомир. Ваш компьютер, 2001, №8, С.6.

S. KUROWSKI.

Тайны криптологии (4): PGP на практике

В последней части нашей статьи речь пойдет о практическом применении одной из программ шифрования. Pretty Good Privacy (англ. — довольно хорошая секретность), или, сокращенно, PGP — одна из наиболее широко распространенных программ. Обусловлено это наличием ее некоммерческой версии, достаточно высокой "дружественностью" к пользователю и, не в последнюю очередь, довольно высокой надежностью.

В PGP используется т.н. гибридное шифрование. Это означает, что здесь используются как метод симметричных, так и метод асимметричных ключей. Так как симметричное шифрование осуществляется примерно в 1000 раз быстрее, чем асимметричное, шифрование в данном случае реализуется только с помощью симметричного ключа. Сам открытый текст обрабатывается с помощью быстрого симметричного алгоритма.

Ключ

В качестве асимметричного способа PGP использует алгоритм RSA или алгоритм DSS/Dittie Hellmam. Оба они в настоящее время причисляются к наилучшим асимметричным алгоритмам. Для симметричного шифрования PGP использует алгоритм IDEA (при применении RSA). Если же используется Dittie/Hellmam, то в качестве симметричных алгоритмов могут быть применены IDEA, CAST и Triple DES. Для получения хэш-значения используется MD5 (при RSA) и SHA1 (для Dittie/Hellman).

Применения

PGP допускает свободный выбор длины ключа. Например, можно выбрать длину ключа в 4096 бит; такую длину пока еще "взломать" никому не удалось.

В задачу PGP входит надежное шифрование E-mail или других файлов. Пересылку E-mail программа осуществляет с помощью программы-клиента, которая "внедряется" в программу E-mail, поскольку она поддерживает такую функцию. Другие тексты или данные PGP шифрует во временной папке Windows.

Для надежной передачи E-mail должны выполняться следующие требования:

- должна быть гарантия того, что отправитель аутентичен, т.е. никто не должен иметь возможности выдать себя за другого;
- содержание E-mail должно быть защищено от манипуляций (изменений текста);
- содержание E-mail должно быть защищено от несанкционированного прочтения;
- отправитель не должен иметь возможности отречься от содержания или происхождения своего E-mail.

PGP обладает необходимыми для выполнения этих требований инструментами. Для защиты от фальсификации отправителя и манипуляций с сообщением, PGP вычисляет хэш-значение, которое после шифрования личным ключом отправителя превращается в цифровую сигнатуру (подпись). Защита от несанкционированного доступа обеспечивается в PGP с помощью упоминавшегося выше алгоритма шифрования.

Надежность через информацию

Детальное объяснение работы PGP выходит далеко за рамки этой статьи,

поэтому мы здесь остановимся только на некоторых моментах.

Естественно, что предпосылкой высокой степени защиты является отсутствие ошибок при инсталляции и обслуживании программы. PGP поддерживает пользователя многочисленными обеспечивающими надежность советами и указаниями как при инсталляции, так и при работе.

Другим центральным пунктом является управление ключами. Новый ключ можно получить с сервера ключей (keyserver) или из E-mail отправителя. Однако нельзя недооценивать опасность фальсификации во втором случае.

Зачастую необходимые ключи можно хранить в так называемом кольце ключей (key ring). Т.к. ключи — открытые, его иногда называют файлом кольца открытых ключей — pubring-file. С каждым ключом связан ряд дополнительных параметров, обеспечивающих надежность. В частности, к ним относятся тип ключа, имя и E-mail-адрес владельца, а также дата изготовления ключа.

Свойство "Validity" указывает, насколько пользователь уверен в подлинности ключа. Свойство "Trust" говорит о том, насколько доверяют владельцу открытых ключей, и что он надежно хранит соответствующий личный ключ.

Само собой разумеется, что с помощью PGP можно генерировать новые ключи. И в этом случае программа информирует владельца о возможном риске.

Потребности и спрос

На рис.1 приведен пример информации, которую можно сохранить по каждому ключу. Тем самым PGP облегчает пользователю оценку надежности чужого ключа.

Остается ответить на вопрос, кому необходим PGP? После ознакомления с соответствующей литературой и страничками Интернета возникает неприятное ощущение, что за тобой уже давно "шпионят". Конечно, в большинстве случаев это обманчивое ощущение. Однако нельзя забывать и о том, что за информацией нередко стоят деньги и власть. Если по E-mail передается не только "болтовня за чашкой кофе", но и конфиденциальные данные, можно посоветовать использовать шифрование. Ведь так легко "выловить" данные из сети...

Дополнительную информацию о PGP можно найти на: www.nai.com, www.pdp.de

Так шифрует PGP

Процесс шифрования является многоступенчатым. На первом шаге PGP с помощью алгоритма ZIP сжимает открытый текст. Тем самым подлежащий шифровке текст становится короче первоначального, что позволяет сэкономить на расходах на пересылку. Однако гораздо важнее, что уже на этом этапе читаемый текст превращается в запутанный "винегрет символов", в котором только с огромным трудом можно обнаружить какую-либо логику. Кроме того, тем самым убирается имеющаяся в любом открытом тексте избыточность.

После этого сжатый текст шифруется с помощью симметричного алгоритма IDEA (или другого). PGP используют для этого т.н. сессионный ключ (session-key). Этот ключ используется только внутри и не имеет ничего общего с парой ключей пользователя. В свою очередь, сессионный ключ шифруется открытым ключом получателя с помощью алгоритма RSA или DSS/Dittie Hellmam.

И только после этого отправитель пересылает получателю зашифрованный текст вместе с зашифрованным сессионным ключом.

У получателя все происходит в обратном порядке. Он вначале дешифрует с помощью своего личного ключа сессионный ключ. После этого расшифровывается сам текст. И, наконец, текст распаковывается и приобретает исходный вид.

Так PGP изготовляет цифровую подпись

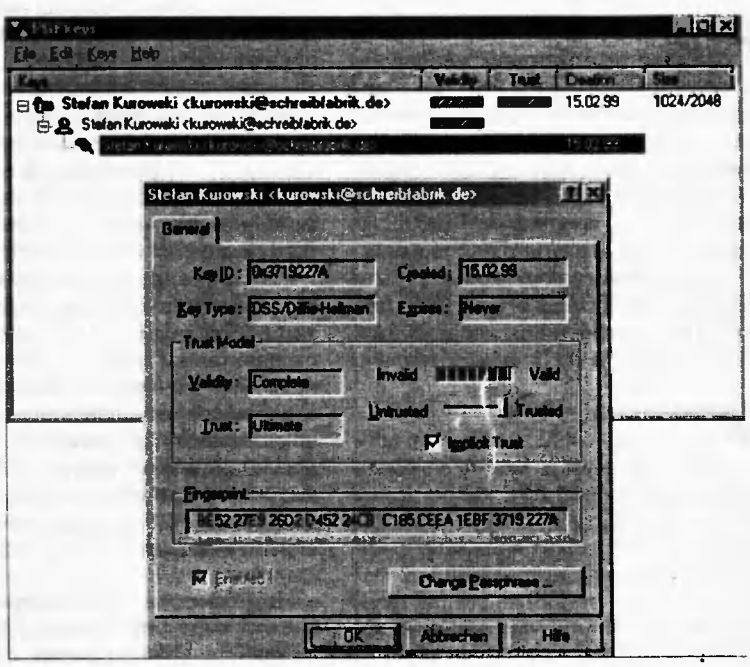
Хотя отправитель использует для шифрования послания открытый ключ получателя, получатель не может быть полностью уверен, что отправитель — именно тот, за кого он себя выдает. Во избежание подлогов и фальсификаций в этом отношении PGP изготавливает цифровую подпись по следующей схеме.

Программа PGP, используя метод MD5 (или SHA1), вычисляет по открытому тексту т.н. хэш-значение. Это значение однозначно определяется текстом, но не позволяет сделать какие-либо выводы о его содержании, и заметно изменяется при малейших манипуляциях с исходным материалом. Хэш-значение (которое называют также Digest) шифруется личным ключом отправителя. Кроме того, отправитель должен указать свой пароль, относящийся к данному ключу. Все это и составляет цифровую подпись (сигнатуру) под документом. Сигнатура вместе с зашифрованным текстом пересылается получателю.

Получив сообщение, PGP дешифрует его и, в свою очередь, вычисляет Digest по получившемуся открытому тексту. Кроме того, программа дешифрует открытым ключом отправителя его сигнатуру. Т.о. программа будет иметь два хэш-значения. Если они отличаются друг от друга, то или для дешифровки был использован другой ключ, что указывает на подложного отправителя, или кто-то несанкционированно манипулировал открытым текстом. В любом случае сообщение может считаться ненадежным.

Если же хэш-значения совпадают, то получатель, если только он убежден в подлинности открытого ключа, который он получил от отправителя, может считать подлинными как открытый текст, так и отправителя. Ведь никто не может иметь частный ключ отправителя, с помощью которого получена сигнатура, кроме владельца соответствующего открытого ключа.

Рис. 1



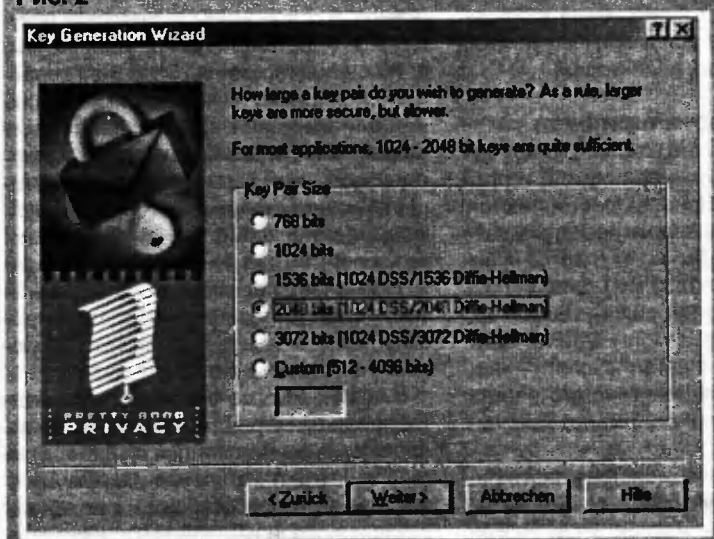
Термины

Сервер ключей (keyserver). На сервере ключей хранятся (и выдаются по запросам) открытые ключи.

При изготовлении ключа PGP предлагает наиболее часто используемые длины. Ключ RSA уже нельзя изготовить с помощью безлицензионной версии 5.0, поскольку теперь она уже лицензионная. Тот, кто все же вынужден ограничиваться RSA из-за ее широкого распространения, может изготовить ключ с помощью более старой версии 2.63i (рис.2).

Фраза-пароль. Это короткая фраза, которую должен задать пользователь при изготовлении пары ключей. Фраза-пароль представляет собой своего рода длинный пароль, которым "подписывается" зашифрованное сообщение; тем самым она является доказательством подлинности ключа.

Рис. 2



Экспортные ограничения

Почти достойно сожаления то, что большинство широко используемых методов криптографии были разработаны в США. Там алгоритмы шифрования рассматриваются как военное оружие; вероятно, так оно и есть, если учесть, какую огромную роль играет информация для успешного исхода войны.

Именно по этой причине в США имеются ограничения на экспорт — мощные шифраторы могут покинуть страну только с ключами такой ограниченной длины, что о надежности зашифрованных данных вряд ли можно вести речь.

Чтобы обойти эти ограничения, воспользовались "брешью" в экспортных ограничениях и напечатали исходный код PGP в книге — поскольку литература подпадает под менее жесткие ограничения. С помощью сканера книга была передана за рубеж — таким путем мощные средства защиты данных попали в Европу.

Тем временем были несколько смягчены экспортные ограничения, так что в настоящее время ограниченное количество копий PGP можно купить и за пределами США.

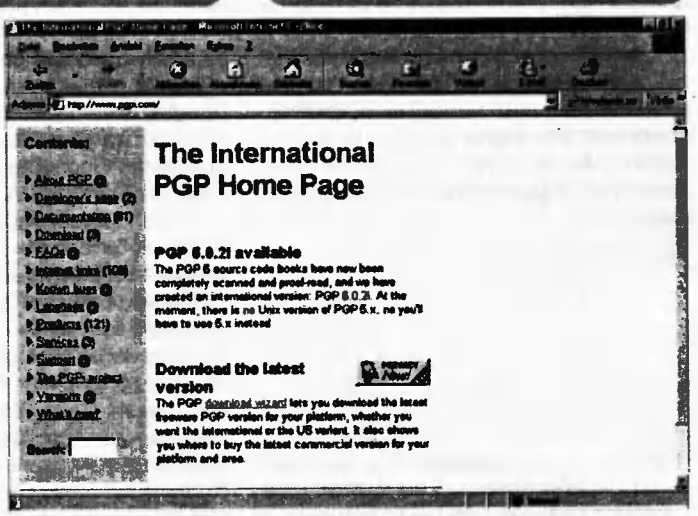
PGP-Download

PGP (сокращение от Pretty Good Privacy) – программа шифрования, которую впервые написал Phil Zimmerman в1991 г., сегодня стала de facto стандартом шифрования данных для Internet-e-mail. На "International PGP Home Page" имеется для бесплатной загрузки современная версия PGP для компьютеров с различными платформами (рис.3).

В наши дни имеется download-версия – PGP 6.0.2i. Адрес сайта в Интернете, с которого можно не только загрузить эту версию, но и узнать дополнительную информацию по теме – www.pgpi.com.

Funkamateur, №7/99.
Перевод А.Бельского.

Рис. 3



"ПЕРЕВЕРНУТЫЙ" ШРИФТ ДЛЯ ACCEL EDA

С.РЮМИК,
г.Чернигов.

До последнего времени одной из наиболее популярных программ разводки печатных плат для IBM PC считалась P-CAD, разработанная фирмой Personal CAD Systems. Нынешний ее владелец — фирма ACCEL Technologies (Сан-Диего, Калифорния, США) — решилась на кардинальные изменения и 29 февраля 1996 года выпустила версию P-CAD для Windows, получившую название ACCEL EDA или проект Sequoia. "ACCEL" — это фирма, "EDA" — Electronic Design Automation — автоматизированное проектирование электроники, все вместе в обиходе называют "аксел едэ" или просто — "аксел". Презентация программы в России состоялась в июне 1996 года. По мере совершенствования одна за другой вышли версии 12.0, 12.1, 13.0, 14.0, 15.0.

ACCEL EDA выполняет полный цикл проектирования печатных плат, включающий в себя графический ввод схем, упаковку на печатную плату, размещение компонентов, интерактивную или автоматическую трассировку проводников, контроль ошибок в схеме и на печатной плате, выпуск документации. Имеется ряд принципиальных преимуществ ACCEL EDA по сравнению с P-CAD. В частности, улучшен пользовательский интерфейс, усовершенствованы алгоритмы трассировки, скоордини-

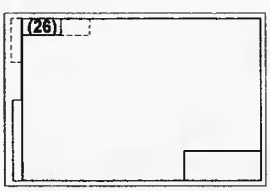
рованы библиотеки символов и корпусов компонентов [1].

Несомненным удобством является работа со шрифтами TrueType, что позволяет наносить надписи в схемах по-русски (символы кириллицы). Это особо ценно при подготовке конструкторской документации (КД) согласно требованиям ГОСТов. Несмотря на географические изменения последнего десятилетия, ЕСКД по-прежнему действует в странах СНГ, опираясь на межправительственные соглашения.

Знание норм ЕСКД традиционно требуется от студентов-дипломников и инженеров промышленных предприятий. В последнее время эти нормы коснулись и малых фирм, обязанных проводить сертификацию выпускаемой продукции. ACCEL EDA легко позволяет создавать документы (в частности, электрические схемы), максимально соответствующие ЕСКД. Одна незадача — в ACCEL EDA не существует прямых способов печати "перевернутых" текстов.

Как известно, на каждый конструкторский документ наносятся основные надписи и дополнительные графы к ним (ГОСТ 2.104). Особое место занимает графа с порядковым номером 26 (рис.1), где приводится децимальный номер документа, повернутый относительно основной надписи на 180°. Идея понятна — в больших чертежных архивах поиск того или иного документа облегчается, если будет обеспечен просмотр его обозначения с любой стороны листа.

Рис. 1



В ACCEL EDA существует режим поворота текста только на 90° и 270°. Для полного "переворота" нужны специальные приемы. Простейший выход — рисовать перевернутую надпись вручную отрезками линейно-ломаных линий по командам **Place/Line**. Однако для создания "стильной" надписи потребуются талант и терпение художника, ведь используемые шрифты содержат массу мелких деталей, без учета которых надпись становится "малосимпатичной".

Не лучше и другое решение — автоматизировать процесс с помощью специальной программы, генерирующей файл макрокоманд рисования текста по аналогии с 32-разрядными версиями P-CAD. В этом случае рутинный процесс создания надписи всего лишь переносится с экрана монитора на листок бумаги. Тонкости стиля шрифта останутся, как и прежде, "за кадром".

Есть более изящный способ, заключающийся в разработке своего собственного шрифта, в котором символы будут повернуты относительно исходных ровно на 180°. Один раз создав такой шрифт, можно будет в дальнейшем оперативно вносить изменения в любую электрическую схему ACCEL EDA. Кроме того, он может пригодиться в оформительских целях для различных текстовых и графических редакторов.

Разновидности шрифтов

Общеизвестно, что каждый шрифт определяется стилем (гарнитурой) и размером печати. Шрифты поставляются семействами, имеющими несколько размеров (кеглей), обычно от 6 до 72 пунктов (1 пункт равен 0,35 мм). Следует отличать гарнитуру (typeface), например, Courier, от шрифта (font) Courier New 9 пунктов. Слово "шрифт" (Schrift) — немецкого происхождения, но его иногда заменяют на английский манер синонимом "фонт".

Шрифты внутри семейства классифицируются по ряду признаков: толщина, цвет, форма букв, наклон и т.д. Различают до 36 независимых параметров. Количество шрифтов в семействе индивидуально, например, семейство Гелиос содержит 19 видов начертаний. Любой стандартный шрифт — это плод многолетнего труда профессиональных художников. Пропорции и наклон символов, насыщенность и контрастность тщательно выверены. Неловкие манипуляции любителя могут изменить эстетическое восприятие в худшую сторону.

Различают 4 категории гарнитур: Serif (шрифт с засечками), Sans Serif (шрифт без засечек), Script (рукописный шрифт), PI (шрифт из спецсимволов) [2]. В КД, выполненной с применением компьютеров, обычно используют две первые гарнитуры (рис.2), причем Serif (Times New Roman) — в текстовых, а Sans Serif (Arial) — в графических документах. Почему так?

Рис. 2

Шрифт с засечками

Шрифт без засечек

Исторически первыми появились шрифты гарнитуры Serif. Исследователи отмечают, что шрифты с засечками (а это небольшие "хвостики" в виде клиньев, пластин или черточек по краям основных штрихов литеры) облегчают распознавание форм при ведении глаза вдоль строки текста. Их основное применение — в книгоиздании. Плохо читаются шрифты Serif очень малых (менее 8 пунктов) и очень крупных размеров. В текстовых документах книжного формата А4, где высота букв должна превышать 2,5 мм, чаще применяются шрифты с засечками размерами 12...14 пунктов.

Шрифты гарнитуры Sans Serif впервые появились в Англии в 1816 г., но изначально считались неуклюжими и гротескными. Тем не менее, их четкие формы оказались незаменимыми для мелких надписей и крупных плакатных заголовков. В отношении графических документов, например электрических схем, применение гарнитуры Sans Serif оправдано, поскольку очень часто большие схемы подвергают пропорциональному уменьшению, и разглядеть на них шрифты с засечками было бы трудно. На типовых схемах формата А1 используют, как правило, шрифт Arial следующих размеров: 11...13 пунктов для надписей и обозначения элементов, 11...12 пунктов — для нумерации выводов микросхем, 11...16 пунктов — для текста в основной и дополнительных графах.

В ACCEL EDA можно работать с масштабируемыми шрифтами TrueType, стандарт на которые появился в 1990 г., благодаря совместным усилиям фирм Apple и Microsoft. Разработанные на их основе русифицированные шрифты широко применяются в локализованных вариантах Windows, начиная с версии 3.1. Файлы TrueType имеют расширение .ttf. Их структура весьма сложная, для описания подробностей не хватит объема целого номера журнала.

Пользователю, собирающемуся модифицировать существующий или создавать свой собственный шрифт, важно знать, что внутри ttf-файла имеется область, где разрешается сохранять данные о названии фонта, классификации, авторском праве разработчика. TrueType-шрифты создавались как универсальные, общемировые. В каждом из них может храниться до нескольких тысяч (!) начертаний символов самых разных алфавитов, включая китайские иероглифы и японское письмо катакана. Это следствие применения двухбайтной кодировки символов Unicode с порядковыми номерами от 0 до 65535. Неудивительно, что заглавная буква "А" кириллицы в ttf-файле может иметь кодировку, превышающую 255, например 572 в шрифте Zurich или 583 в Arial.

Методика создания "перевернутого" шрифта

В мире насчитывается несколько десятков тысяч шрифтов, многие из которых стандартизованы международной организацией Association Typeographique International. Подборку из 7000 шрифтов можно найти на CD-ROM серии "Золотая коллекция" [3]. Имеются "залежи" и в Internet, например на `http://www.ora.com/homepages/comp.fonts` или `http://www.web.idirect.com/~freejack/fonts.html`

За долгие годы работы со шрифтами был создан добротный банк сопутствующего программного обеспечения:

- программы-просмотрщики (например, встроенные средства просмотра в Windows);
- программы-стилизаторы (добавление спецэффектов, объемности к уже существующим шрифтам);
- программы-редакторы (создание собственных шрифтов).

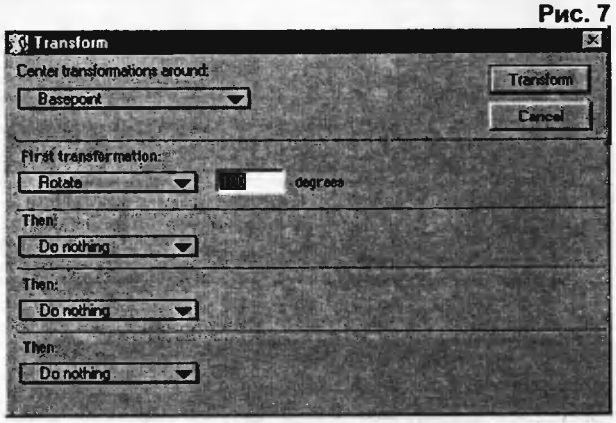
По количеству больше всех выпущено первых, а меньше — последних программ. В нашем случае нужен редактор шрифтов. На диске [3] имеются 3 подходящие по теме программы: "Scanfont v3.0" (FontLab Developers Group, 1996), "The Font Creator Program v1.2.1" (High-Logic, 1997) и "Fontographer v4.1" (Macromedia Inc., 1996). Первая из них не позволяет сохранять файлы редактируемых шрифтов (демо-версия), вторая — не имеет режима произвольного вращения для символов, созданных цельной "картинкой". Остается программа "Fontographer", которая, несмотря на демо-версию, позволяет создавать "перевернутый" шрифт. Единственная трудность заключается в том, что осуществить задуманное наскоком, "с шашкой наголо", не получится, необходимо учитывать ряд нюансов.

Первоначально во избежание накладок следует скопировать в отдельную директорию C:\USERFONT файл шрифта, которым выполняются текстовые надписи в ACCEL EDA на электрической схеме. Обычно это **arial.ttf**, находящийся в папке **Windows/Fonts**.

Затем запустить на выполнение программу "Fontographer" (fontog.exe) и открыть в ней файл шрифта — **File/OpenFont**(C:\USERFONT\arial.ttf). Далее щелчком левой кнопки мыши выбрать редактируемый символ, например, цифру "4" (рис.3). Двойным щелчком войти в меню редактирования (рис.4). Как видно, контур цифры "4" ограничен прямоугольником с габаритами 1466 x 1139 единиц (units). Базовая точка basepoint находится в начале сетки координат 0,0. Если установить базовую точку в геометрический центр прямоугольника и произвести вращение символа вокруг нее на 180° (такая функция в программе имеется), то как раз и получится желаемый поворот цифры "4".

Рис. 6

Рис. 7



Казалось бы, все просто, однако при детальном рассмотрении гарнитуры Arial выясняется, что символы разнятся друг от друга не только по ширине (шрифт с переменным шагом), но и по высоте. Например, цифры "4" и "9" при печати кажутся одинаковыми, хотя это всего лишь оптическая иллюзия (рис.5), и таких примеров множество.

Чтобы все символы после вращения выстроились в один ровный ряд, а не в извилистую "тропинку", необходимо задать условную линию "перевернутого" горизонта. Высота 65% всех цифр и прописных букв составляет 1466 единиц. Этот размер и примем за базовую горизонталь X. Размер по ширине Y_i для разных символов будет разный, он легко определяется наведением курсора на вертикальную линию с отсчетом по верхней шкале. Например, для цифр 0...9 значение Y_i равно 1399 единиц, для прописной буквы "M" — 1706 единиц и т.д. Измеренные данные желательно свести в таблицу. Искомые координаты базовой точки центра симметрии i-го символа — $X/2$, $Y_i/2$,

где $X/2$=733 единицы — постоянный размер;

$Y_i/2$ — переменное значение половины ширины прямоугольника.

Технология коррекции следующая. Выбрать редактируемый символ щелчком левой кнопки мыши. Не входя в меню редактирования, задать координаты базовой точки — **Points/SetBasepoint/Horizontal**(значение $Y_i/2$)**/Vertical**(733)**/OK** (рис.6). Произвести вращение символа — **Element/Transform/Center...**(Basepoint)**/First transformation**(Rotate, 180 degrees)**/Transform** (рис.7). Нажать клавишу <Курсор вправо> для перехода к очередному символу.

Теперь следует осуществить аналогичные операции для всех цифр, а также прописных и строчных букв кириллицы. Латинские буквы не понадобятся, поскольку децимальные номера отечественных изделий их не содержат. Весь процесс, включая составление таблицы значений Y_i, занимает не более часа, ускорить действия помогут "горячие" клавиши <Ctrl>+<`>, <Ctrl>+<\>. Напоследок производится вращение символа "точка", точнее, сдвиг его по вертикали на 1261 единицу — **Element/Transform/First transformation**(Move)**/Horizontal**(0)**/Vertical**(1261 units).

По окончании коррекции необходимо провести правку служебной информации внутри файла — **Element/FontInfo/General/Family Name**(ArialUser)**/Notice**(...RADIOLUBITEL-2000) (рис.8). Теперь можно записать полученный шрифт на жесткий диск, при этом файлу будет автоматически присвоено имя **ariau__.ttf** — **File/Generate Font Files/Advanced/Encoding**(Windows95)**/Format**(TrueType)**/SetDirectory**(C:\USERFONT)**/Overwrite existing file/Generate** (рис.9).

Подключение откорректированного шрифта к Windows95/98 особенностей не имеет:

Мой компьютер/Панель управления/Шрифты/Файл/Установить

Рис. 3

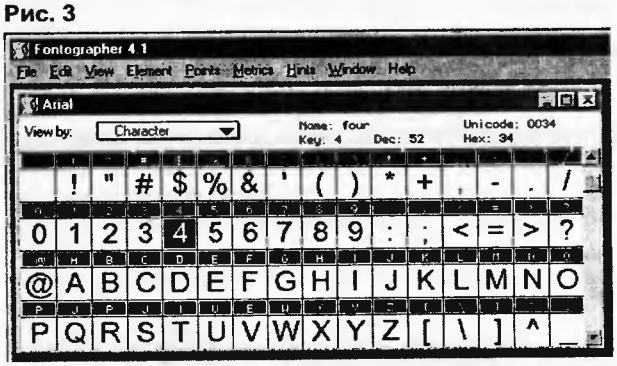


Рис. 4

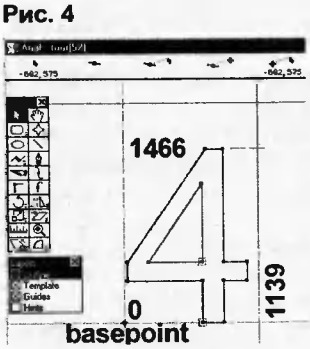


Рис. 5

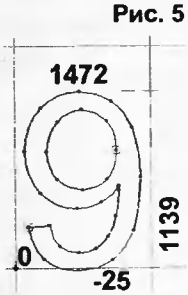


Рис. 8

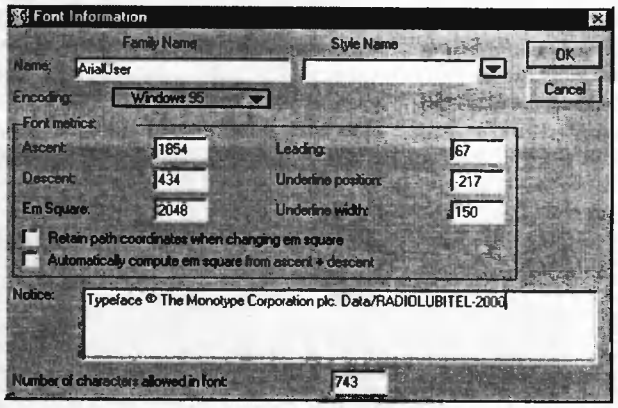


Рис. 9

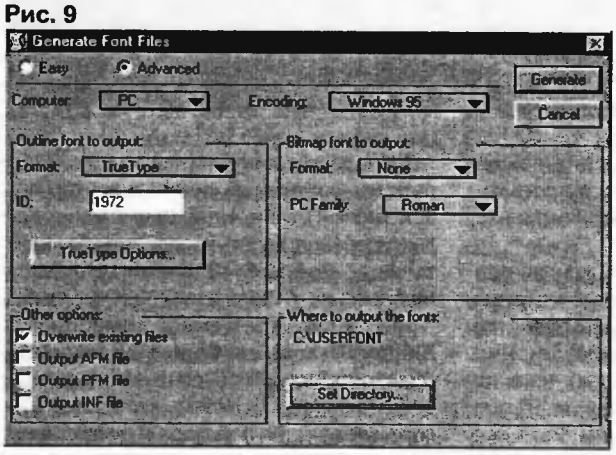


Рис. 10

Рис. 11

шрифт(C:\USERFONT\ArialUser)/**OK.** На рис.10 показан вид тестового экрана со знаменитой фразой: "Съешь еще этих мягких французских булочек, да выпей чаю", в состав которой в минимальном наборе входят все буквы русского алфавита. Кстати, аналогичная по значению фраза для английских букв выглядит так: "The quick brown fox jumps over the lazy dog" (Быстрая коричневая лиса прыгает через ленивую собаку).

Теперь впору проверить результат на программе ACCEL EDA, предварительно подключив вновь созданный шрифт **ArialUser** стандартным для нее способом [1] — **Options/Text Style/ Add...** Пример реальной надписи десятичного номера АДБК.469455.079 Э3, выведенного "перевернутым" шрифтом, показан на рис.11. Разумеется, сама надпись при вводе должна правильно читаться не слева направо, как мы привыкли, а по-восточному — справа налево.

Литература

1. Разевиг В.Д. Система проектирования печатных плат ACCEL EDA 12/1 (P-CAD для Windows). — М.: "СЛ Пресс", 1997, 368 с.

2. Молчанов Б. Настольные издательства от А до Я. — КомпьютерПресс, 1992, №3, С.5-13.

3. "Золотая коллекция шрифтов и программ для их создания" — CD-ROM, 1999.

WinOnCD Power Edition 3.7

А.ВЕНДИЛОВСКИЙ,
г.Минск.

В жизни каждого компьютерщика наступает момент, когда появляется недовольство состоянием данных на винте. Делать бэкапы на дискеты — дискет не оберешься, а если массивы данных под гигабайт, то приходится задумываться о месте их хранения. Да и дискеты сегодня очень ненадежны и очень быстро приходят в негодность. Всевозможнейшие ZIPы и стримеры по-прежнему остаются дорогой экзотикой. А если ваша работа требует постоянного переноса и копирования файлов больших размеров, то старенький трехдюймовый дисковод уже никак не может помочь вам, а носить с собой винчестер для таких нужд — занятие трудоемкое, больно хрупкое устройство. А как раздражает, когда у вас под два десятка софтовых дисков, и на каждом требуется лишь однадве программы. А вот если бы они все были на одном... И собственный винчестер уже переполнен данными, которые уже трижды заархивированы, и обязательно могут потребоваться когда-нибудь, но не сейчас.

В настоящий момент эти проблемы решаются просто. Открываете газету — и вскоре вы уже гордый обладатель CD-recorder/rewritable driver. Установив его в свой компьютер, подстрекаемый жаждой творчества, вы сталкиваетесь с проблемой — какой софт использовать для записи компактов? Те программы, что идут вместе с рекордером, чаще всего являются урезанными и облегченными версиями крупных пакетов и обладают лишь десятой долей всех возможностей. Поэтому для начинающих я рекомендую простой и очень надежный пакет, одобренный всеми производителями записывающих устройств — WinOnCD Power Edition 3.7. Он позволяет производить запись на CD-R/RW 650 и 700 Мб во всех принятых стандартах. Его интерфейс очень похож

на интерфейс Проводника Windows и не должен вызвать проблем у пользователя.

После инсталляции у вас будет установлено два программных продукта — PacketCD и WinOnCD.

Внимание! Перед работой отключите все резидентные антивирусы, мастера-планировщики, которые могут активироваться в момент записи, и прочие резидентные утилиты. Проверьте, что в вашей машине отключен режим энергосбережения, проведите дефрагментацию винчестера и освободите восемьсот мегабайт пространства на одном из дисков для хранения промежуточных форм программ.

PacketCD — очень простая и полезная программа, форматирующая ваш CD-R/RW диск. Процедура эта, в процессе которой вам останутся доступными на матрице 530 Мб — длительная, и занимает порядка двадцати-тридцати минут. Эти жертвы не напрасны, в результате матрица будет использоваться вами как обычная дискета или жесткий диск. Для записи на нее данных не потребуется пользоваться специальными программами, достаточно обычной команды копирования из любой программы-оболочки, Проводника Окон или привычного метода "drag and drop". Резидентно находящийся в памяти PacketCD сделает все остальное сам. Только нужно запомнить несколько мелочей.

Если вы отформатировали CD-R-диск, то естественно, вы сможете только записывать на него данные, считать которые можно только на рекордере, где установлен PacketCD. Чтобы сделать этот диск видимым для обычных устройств чтения, придется выделить еще двадцать мегабайт свободного пространства и закрыть сессию. Но использовать обычные матрицы сейчас уже не имеет смысла. На форматированных матрицах CD-RW вы можете не только записывать, но и стирать ненужные вам файлы, не ощущая разницы между матрицей и дискетой. Формат OSTA/UDF полностью совместим с программной оболочкой Окон и позволяет читать диск на любом современном дисководе без резидентного PacketCD или закрытия сессии.

Сам процесс тоже не вызывает проблем — в первом диалоговом окне (рис.1) мы выбираем "*Volume name*" будущего диска, во втором (рис.2) программа предлагает использовать дополнительную компрессию данных на матрице для увеличения свободного места (очень напоминает DriveSpace из Окон), но честно предупреждает о возникающих проблемах совместимости (для чтения такого диска потребуется установленный на компьютер PacketCD). Нажимаем клавишу "*Format*", и остается лишь смотреть на бегущие проценты выполненного форматирования. После окончания процесса диск готов к работе.

WinOnCD — после запуска перед вами откроется окно выбора проекта записи (рис.3). Вот описание основных возможностей.

В каталоге **"Data"**:

- **"CD-ROM ISO 9660/ Joliet"** — старый стандарт записи, позволяющий создавать диски, совместимые со старыми версиями операционных систем. Имеет ограничения на создание уровней каталогов, длинные имена файлов и прочие "пережитки" DOS. Наличие стандарта Joliet позволяет вручную отключить эти ограничения;
- **"Append Session (ISO 9660/ Joliet)"** — если на вашей матрице осталось свободное место и диск не был "закрыт", то, используя эту функцию, можно дозаписать на него новую информацию;
- **"UDF+ ISO 9660/ Joliet"** — позволяет создавать диски в соответствии с новым стандартом, разработанным для DVD и CD-RW-дисков, но не исключает совместимости с прошлыми стандартами;
- **"UDF+ ISO 9660/ Joliet Hybrid CD"** — для тех, кто пишет диски к Apple Macintosh;
- **"Bootable CD"** — создание загрузочного диска;
- **"*.CPJ"** — проекты, которые были созданы вами.

Рис. 1

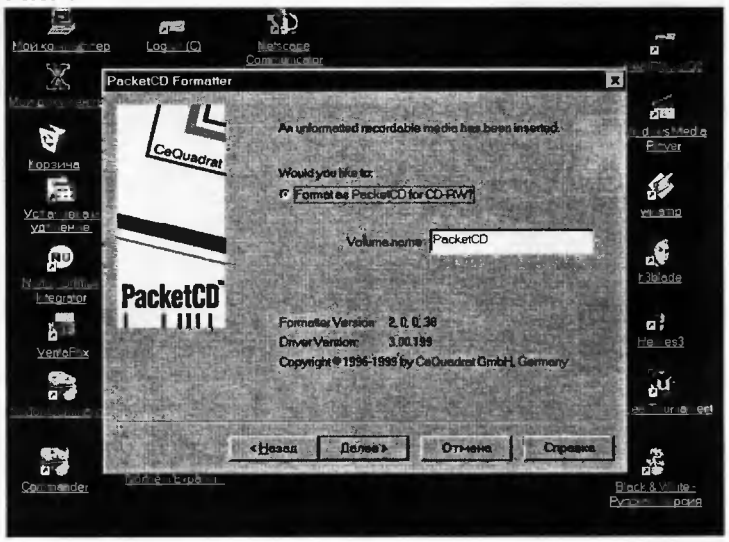


Рис. 2

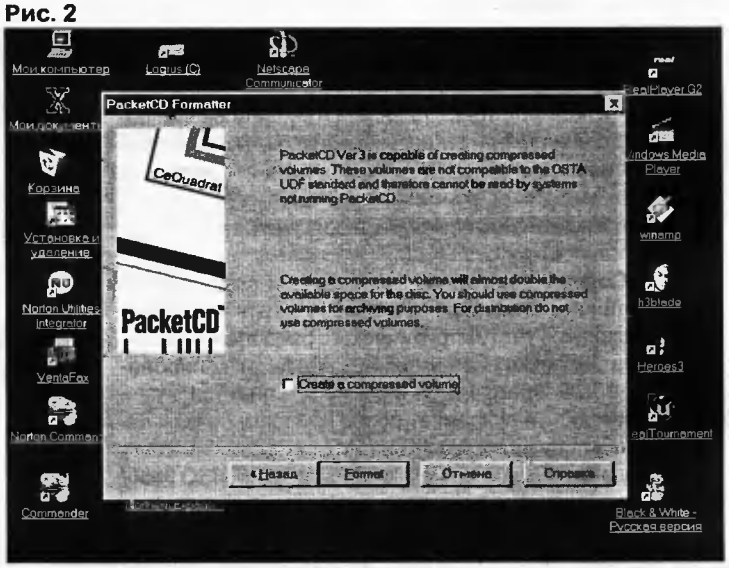


Рис. 3

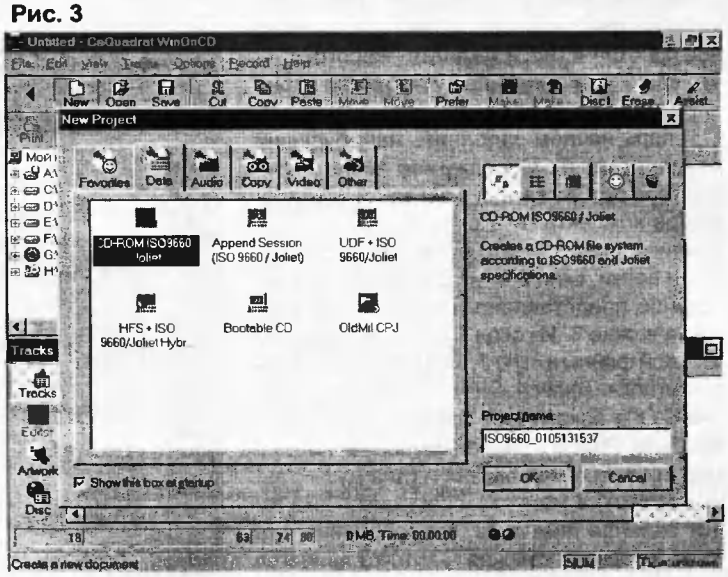
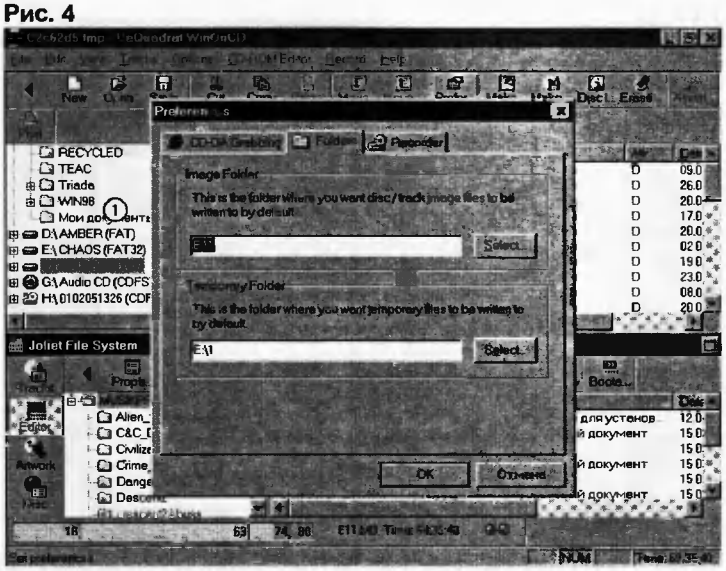


Рис. 4



В каталоге **"Audio"**:

- **"CD Digital Audio"** — привычный аудио-компакт:
- **"CD Extra"** — очень важный формат, который позволяет в течение одной сессии записывать треки различных форматов, к примеру, трек с данными и музыкальными дорожками. Если вспомните игровые компакты, в которых музыка была записана в виде обычных треков, то все они записаны только в таком формате. В противном случае, программы не смогли бы найти музыкальные дорожки.

В каталоге **"Copy"**:

- **"CD Copy"** — позволяет копировать сразу целый диск, аналог команды DOS "Disc Copy". Хотя копирование происходит с учетом физического расположения информации на диске, на системы защиты от копирования (вроде "CD Guard") это не распространяется;
- **"CD Image"** — сохраняет образ компакта на винчестере для последующей записи.

Для создания обычного диска выберите **"UDF+ ISO 9660/ Joliet"** — и попадете в основное меню программы.

Первым делом, нажмите на иконку **"Preferences"** и найдите закладку **"Folders"** (рис.4). Убедитесь, что папки **"Temporary"** и **"Image"** располагаются на том диске, где у вас есть свободное пространство; в закладке **"Recorder"** — что ваше записывающее устройство верно определено программой.

Теперь начнем заполнять наш будущий компакт (рис.5). Содержимое вашего компьютера отображается в окнах 1 и 2, содержимое компакта — в окне 3. Как и в Проводнике, окно 1 — это носители информации и папки, окно 2 — их содержимое. Аналогично представлено и содержание компакта в окне 3. Из окон 1 и 2 перетаскиваем мышкой файлы в окно 3, внимательно наблюдая за появившейся внизу экрана зеленой линией. Она показывает степень заполнения матрицы. Рядом с линией отображается размер проекта в мегабайтах и (для музыкальных дисков) в минутах и секундах. Пока горит зеленая лампочка, переполнения нет. Если загорится красная — что-то придется удалить. На самой линии есть несколько отметок. Отметка **"74"** означает границу матрицы 650 Мб, отметка **"80"** — 700 Мб.

Напоследок можно создать autorun для вашего диска. Это можно сделать как простым способом (набрать текст вручную в любом редакторе текста), так и используя **WinOnCD**, через иконку **"Properties"**, выбрав закладку **"autorun"**. Там же вы найдете дополнительные настройки совместимости файловой системы диска — поддержку файловых систем ISO 9660, Joliet, OSTA UDF — если настройки программы "по умолчанию" не будут вас устраивать.

Теперь, когда нужные файлы выбраны, нажмите иконку "Disk" на крайней левой панели, и попадете в окно записи (рис.6).

В меню **"Fixation"** доступны следующие функции:

- **"Create Multisession CD"** — сессия записи, которую вы сейчас производите, будет закрыта, но остается возможность последующей записи на оставшееся свободное место на матрице в будущем. Только не забудьте, что каждая следующая сессия будет забирать дополнительно 20 Мб под файловую систему. Все современные приводы CD-ROM читают такие диски. Старые устройства чтения не поддерживают этот режим (это касается и музыкальных проигрывателей);

Рис. 5

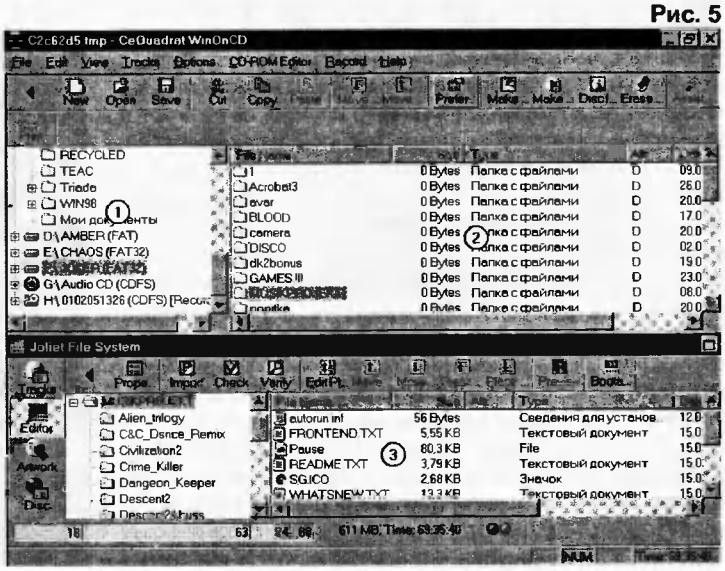
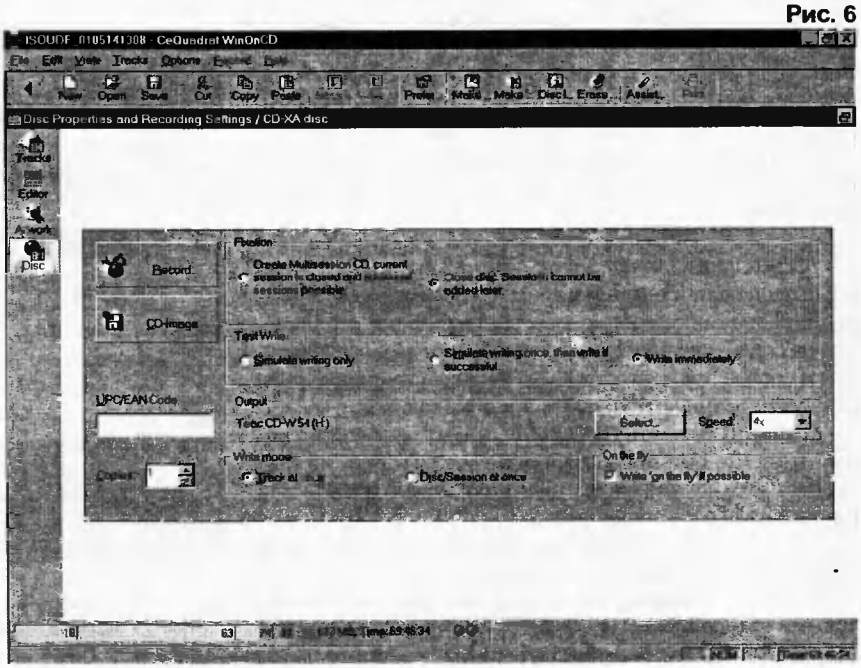


Рис. 6



- **"Close Disc"** — запись на диск больше невозможна. Не стоит пренебрегать этой функцией. Если на матрице остается меньше 30 Мб свободного пространства, то стоит закрыть диск — дальнейшие попытки записи на него бесперспективны. Как показывает практика, до сих пор еще существует много программ, которые некорректно работают с "незакрытых" дисков, что тоже нужно учитывать.

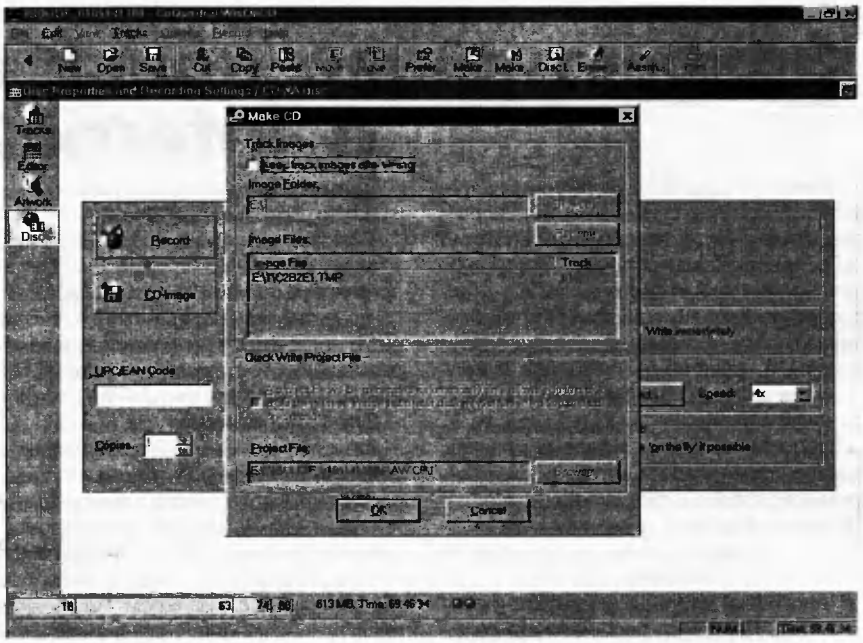
В меню **"Test Write"**:

- **"Simulate Writing only"** — будет полностью имитироваться процесс записи диска, с передачей данных на рекордер. В случае сбоя вы всегда сможете определить его причину без ущерба для себя и матрицы;

- **"Simulate writing once, then write if successful"** — рекомендуется для всех начинающих, а также в случае, если вы не уверены, что запись пройдет без ошибок. В первый раз программа имитирует процесс записи, и если в процессе имитации ошибок не возникло, автоматически начинается "чистовая" запись матрицы;

- **"Write immediately"** — немедленная запись.

Рис. 7



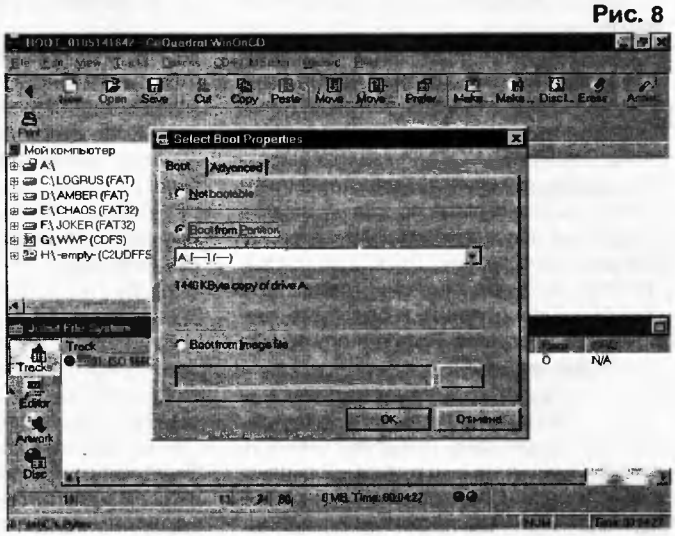
В меню **"Output"** еще раз убедитесь, что ваш рекордер правильно определен, если нет — используйте клавишу **"Select"**. В окошке **"Speed"** выберите скорость записи. Скорость записи зависит от возможностей вашего рекордера и качества матрицы. На всех фирменных матрицах указана максимальная скорость записи, если же вам досталось нечто китайское безымянное, или вы не уверены в ее качестве, то лучше выбрать маленькую скорость, к примеру, **"2x"**. Лучше потратить больше времени на запись, чем позже выяснить, что диск некорректно читается :-(.

Теперь заглянем в меню **"Write mode"**. Этот раздел представляет интерес при записи музыкальных компактов. Если выбрать режим **"Track at once"**, то между музыкальными дорожками будет пауза в две секунды. В режиме **"Disc/Session at once"** проигрывание треков будет идти единым потоком.

Меню **"On the fly"**. На это меню стоит обратить особое внимание. Главное при записи матрицы — чтобы процесс был непрерывным, или вы получите из вашей матрицы очень симпатичную подставку для кофе. Для решения этой проблемы производители устанавливают большой кэш в рекордерах. Но это не панацея, ведь увеличивается и скорость записи. Скорость передачи данных на рекордер должна быть минимум в два раза выше скорости записи. Вызвать такой сбой (прерывание потока) может множество причин, но основная — пустой буфер.

При копировании диска на диск, если исходный CD-ROM был низкого качества, и за то время, пока на сбойном участке подбиралась оптимальная скорость чтения и производилась коррекция ошибок, буфер опустел. Эта проблема возникает и в случае, если данные, которые вы записываете, находятся на другой машине, а вы получили к ним доступ через сеть. Даже если они находятся на вашем винчестере, этот вариант возможен при небольшом объеме оперативной памяти и, как следствие, маленьком кэше винчестера, большой дефрагментации данных и (абсолютно серьезно) если вы записываете большое количество маленьких файлов (к примеру, дистрибутив Окон 3.11). И конечно, могут внести свою лепту всевозможные резидентные антивирусы и другие программы, которые в процессе своей работы "захватывают" ресурсы компьютера. Если вы абсолютно уверены, что эта проблема вас не коснется, то оставьте галочку в этом окне "On the fly". Если не уверены, WinOnCD предлагает удобное средство — уберите галочку, и перед записью он создаст на винчестере в одном файле полный образ вашего будущего диска, и запись будет производиться с использованием этого файла. Настоятельно рекомендуется так и поступать. Программа спросит вас (рис.7), сохранять ли имидж диска после записи, и уточнит место его расположения. Выберите то, что вас больше устраивает, и жмите "OK". Вам остается роль наблюдателя. Если все прошло удачно, появится надпись об успешном завершении работы. Поздравляю, вы записали свой первый диск!

Рис. 8



Создание загрузочного компакта

При создании загрузочного компакта, сначала вы должны создать или загрузочную дискету, или подготовить один из загрузочных разделов вашего винчестера. Проследите, чтобы все пути к файлам в системе были указаны правильно, и диск нормально загружался. Убедившись в этом, запустите WinOnCD и выберете проект "Bootable CD". Далее программа спросит, где находится системный диск или системная дискета (рис.8). Укажите точное расположение и нажимайте "OK". Системная дискета должна остаться в дисководе, а дальнейшие процедуры ничем не отличаются от выполненных ранее при записи обычного компакта.

Г.ТРОЯН,
г.Минск, РИВШ БГУ,
E-mail: G-Trojan@yandex.ru

Эффективный поиск информации в Internet

Вначале, чтобы было удобно ориентироваться, обозначим основные группы вопросов, которые будут рассмотрены в статье.

♦Постановка задачи поиска. Информационно-поисковые системы (ИПС) — определение и главные задачи. Понятие релевантности. Обобщенная структура и основные компоненты ИПС для WWW. Понятие индекса. Особенности процедуры индексирования.

♦Средства поиска — поисковые машины, тематические каталоги, метапоисковые системы, программы ускоренного поиска.

♦Приемы работы с тематическими каталогами: перемещение с уточнением темы, поиск по ключевым словам.

♦Поисковые машины (автоматические индексы). Простой и расширенный режимы поиска. Обобщенные возможности формирования запроса — перечисление ключевых слов, поиск по маске, поиск точной фразы, поиск в части документа с использованием специальных операторов. Поиск с использованием логических операторов (AND, OR, NEAR, NOT). Режим расширенного (advanced) поиска. Представление и обработка результатов поиска — поиск в найденном, поиск похожих документов, управление сортировкой результатов поиска.

♦Метапоисковые системы (поисковые службы). Определение и основные возможности программ ускоренного поиска (поисковых агентов).

♦Параметры эффективности поиска — полнота, точность, актуальность, скорость. Факторы, влияющие на эффективность поиска. Сравнительные возможности поисковых систем. Планирование поисковой процедуры. Приемы эффективного поиска.

♦Поиск статей в группах новостей. Поиск файлов. Поиск адресной информации организаций и людей.

Постановка задачи поиска

Рассмотрим постановку задачи поиска в общем виде. Для этого нам необходимо ответить на три вопроса — что искать (какие источники информации); где искать (место размещения этих источников) и как искать (какие инструменты для этого использовать).

Источники информации в Internet

Выделим основные источники информации, представленные в Internet. Это:

- документы WWW;
- статьи в группах новостей и списках рассылки;
- файлы в библиотеках файлов;
- справочники адресной информации организаций и людей (электронная почта, адрес, телефон);
- статьи в тематических базах данных, энциклопедиях.

Заметим, что перечисленный список не претендует на полноту.

Размещение источников информации в Internet

Теперь ответим на вопрос, где размещаются эти источники информации. Это такие популярные ресурсы Internet как WWW, группы новостей, списки рассылки и FTP-серверы. В настоящее время основным местом размещения информации в Internet является всемирная паутина (WWW).

Способы поиска

Безусловно, можно искать источники информации "вручную", начиная с какого-либо стартового адреса и переходя по нужным ссылкам. Вы можете узнать адреса из специализированных журналов по информатике и из Internet, либо использовать справочники под названием "Желтые страницы" с классифицированными по категориям адресами фирм и учреждений. Подобные справочники выпускаются в бумажном варианте или на CD-ROM. Однако для эффективного поиска информации в таком изменчивом пространстве как Internet необходимо научиться пользоваться специальными инструментами, цель которых — собирать данные об информационных ресурсах глобальной компьютерной сети и предоставлять пользователям возможность быстрого поиска.

Информационно-поисковые системы (ИПС). Определение

Таким образом, мы подходим к понятию автономного инструмента поиска — информационно-поисковой системы.

ИПС — это система, обеспечивающая поиск и отбор необходимых данных в специальной базе с описаниями источников информации (***индексе***) на основе информационно-поискового языка и соответствующих правил поиска [1].

Главная задача ИПС

Главной задачей любой ИПС является поиск информации в соответствии с информационными потребностями пользователя, формируемыми в виде запроса. Очень важно в результате проведенного поиска ничего не потерять, то есть найти в индексе все документы, относящиеся к запросу (полнота поиска), и не найти ничего лишнего (точность поиска). Поэтому вводится качественная характеристика процедуры поиска — релевантность.

Релевантность — это соответствие результатов поиска сформулированному запросу.

Основные показатели ИПС для WWW

Далее мы будем, в основном, рассматривать ИПС для всемирной паутины (WWW). Основными показателями ИПС для WWW являются ***пространственный масштаб и специализация*** [1]. По пространственному масштабу, ИПС можно разделить на локальные, глобальные, региональные и специализированные. Локальные поисковые системы могут быть разработаны для быстрого поиска страниц в масштабе отдельного сервера. Региональные ИПС описывают информационные ресурсы определенного региона, например, русскоязычные страницы в Internet. Глобальные поисковые системы, в отличие от локальных, стремятся по возможности наиболее полно описать ресурсы всего информационного пространства сети Internet.

Кроме того, ИПС могут специализироваться по поиску различных источников информации, например, документов WWW, файлов, адресов и т.д.

Основные задачи проектирования ИПС для WWW

Рассмотрим подробнее основные задачи, которые должны решить разработчики ИПС. Как следует из определения, ИПС для WWW проводят поиск в собственной базе (индексе), в которой содержится результат описания распределенных источников информации. Значит, сначала нужно описать информационные ресурсы и создать индекс. Построение индекса начинается с определения начального набора URL источников информации [2]. Затем проводится процедура индексирования.

Индексирование — описание источников информации и построение индекса.

Индекс — специальная база данных для эффективного поиска описанных информационных ресурсов.

В некоторых информационно-поисковых системах описание источников информации проводится персоналом ИПС, то есть людьми, которые составляют краткую аннотацию на каждый ресурс. Затем, как правило, проводится сортировка описанных ресурсов по темам (составление тематического каталога). Конечно, описание, составленное человеком, будет адекватно источнику. Правда, в этом случае процедура индексирования занимает значительный период времени, поэтому формируемый индекс имеет, как правило, ограниченный объем. Зато поиск в подобной системе можно будет проводить так же легко, как в тематических каталогах библиотек.

В ИПС другого типа процедура описания информационных ресурсов автоматизирована. Для этого разрабатывается специальная программа-робот, которая по определенной технологии обходит ресурсы, описывает их (проводит индексирование) и анализирует ссылки с текущей страницы для расширения области поиска. Как может описать документ программа? Чаще всего просто составляется список слов, которые встречаются в тексте и других частях документа, при этом учитывается частота повторения и

местоположение слова, то есть слову приписывается своеобразный весовой коэффициент в зависимости от его значимости. Например, если слово находится в названии Web-страницы, робот присвоит ему более высокий коэффициент. Поскольку описание автоматизировано, затраты времени невелики, и индекс может оказаться очень большим по размеру. Таким образом, следующей задачей для ИПС второго типа является разработка робота-индексировщика.

Робот-индексировщик — программа, которая служит для сканирования Internet и поддержки базы данных индекса в актуальном состоянии.

Для поиска в системах данного типа пользователю необходимо научиться составлять запросы, в простейшем случае состоящие из нескольких слов. Тогда ИПС будет искать в своем индексе документы, в описаниях которых встречаются слова из запроса. Для проведения более качественного поиска необходимо разрабатывать специальный язык запросов для пользователя. В зависимости от особенностей построения модели индекса и поддерживаемого языка запросов, разрабатываются механизм поиска и алгоритм сортировки результатов.

Поскольку индекс имеет значительный объем, количество найденных документов может оказаться достаточно большим. Следовательно, чрезвычайно важно, как поисковая машина проведет поиск и отсортирует его результаты.

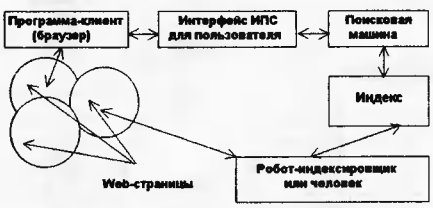
Существенное значение имеет внешний вид поисковой системы, предстающий перед пользователем, поэтому одной из задач является разработка удобного и красивого интерфейса.

Наконец, исключительно важна форма представления результатов поиска, поскольку пользователю необходимо узнать как можно больше о найденном источнике информации, чтобы принять правильное решение о необходимости его посещения.

Структура ИПС для WWW

Рассмотрим обобщенную структуру ИПС для всемирной паутины WWW (рис.1) [1].

Рис. 1



Для навигации в WWW и обращения к поисковому серверу пользователь использует стандартную программу-клиент для всемирной паутины — браузер. По адресу домашней страницы ИПС пользователь работает с интерфейсом поисковой системы, который служит для ввода запросов и просмотра результатов поиска.

Основным компонентом ИПС является поисковая машина, которая проводит в индексе поиск ссылок на информационные ресурсы и выдает результаты поиска пользователю.

Как уже говорилось ранее, поиск осуществляется в специальной базе, именуемой индексом. Архитектура индекса устроена таким образом, чтобы поиск проходил максимально быстро, и можно было использовать эффективные алгоритмы сортировки результатов поиска. В идеале результаты поиска должны быть отсортированы таким образом, чтобы наиболее релевантные ссылки находились вверху списка.

Источники индексирования для документов WWW

Как известно, Web-страница — это сложный документ, состоящий из множества элементов. При описании подобного документа программой-роботом необходимо учитывать, в какой именно части Web-страницы встретилось данное слово. Источниками индексирования для документов WWW обычно являются [2]:

- заголовок Web-страницы (Title);
- заголовки различных уровней (H1...H6);
- аннотация (Description);
- списки ключевых слов (KeyWords);
- гипертекстовые ссылки;
- полные тексты документов.

Поисковые системы, которые описывают весь текст документа WWW, называются полнотекстовыми.

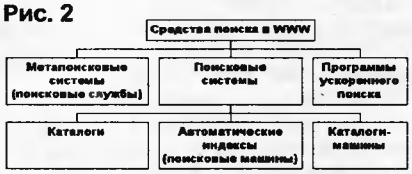
Особенности процедуры индексирования

Во время процедуры индексирования часто производится нормализация лексики (приведение слова к базовой форме). Некоторые неинформативные слова, например, союзы или предлоги, не индексируются. В каждой ИПС существует свой список так называемых стоп-слов, которые игнорируются в процессе индексирования. В системах с сильно изменяемыми языками, например, русским, проводится учет морфологии. Учет морфологии означает умение работать с различными формами слов конкретного языка. Здесь следует отметить относительную сложность русского языка, слова которого изменяются по числам, падежам, родам и временам, причем зачастую неожиданным образом (например: идет, шел, пойдет, идут и т.д.). Все существующие ИПС с учетом морфологии русского языка используют "Грамматический словарь русского языка", составленным А.А.Зализняком. Словарь включает 90000 словарных статей, по каждому слову даются сведения о том, изменяемо ли оно, и как именно оно склоняется или спрягается [3].

Средства поиска в WWW

Из вышеизложенного следует, что основными инструментами поиска информации в WWW являются ИПС. Однако в Internet существуют средства поиска, имеющие принципиальные отличия от рассмотренных выше ИПС. В общем случае, можно выделить следующие поисковые инструменты для WWW — поисковые системы, метапоисковые системы (поисковые службы) и программы ускоренного поиска (поисковые агенты). Структура средств поиска в WWW представлена на рис.2.

Рис. 2



Центральное место по праву принадлежит поисковым системам, которые, в свою очередь, подразделяются на каталоги, автоматические индексы (поисковые машины) и каталоги-машины. Только поисковые системы почти в полном объеме обладают возможностями и свойствами ИПС.

Каталог (Directory) — поисковая система, в которой описание ресурсов проводится персоналом (людьми). Затем проводится сортировка описанных ресурсов по темам (составление тематического каталога).

Поисковая машина (Search Engine) — поисковая система, которая для автоматизации процедуры описания информационных ресурсов использует программу-робот.

Последнее время во всемирной паутине стали появляться системы, автоматически осуществляющие поиск сразу в двух индексах (индексе каталога и индексе поисковой машины). Подобные системы позволяют использовать преимущества поисковых серверов обоих типов и называются ***каталогами-машинами***.

Принципиальным отличием метапоисковых систем и программ ускоренного поиска от ИПС является отсутствие своего собственного индекса. Данные инструменты проводят поиск в индексах других поисковых систем.

Метапоисковая система (Metacrawler) — поисковая система, не имеющая своего индекса, но способная послать запросы пользователя одновременно нескольким поисковым серверам, затем отобрать самые релевантные результаты, объединить их и представить пользователю в виде документа со ссылками.

Программа ускоренного поиска (Searchbots) — программа, устанавливаемая на компьютере пользователя, способная отправить запрос нескольким поисковым серверам и отсортировать полученные результаты, удаляя дубликаты.

Заметим, что большинство поисковых систем являются одним из компонентов многофункциональных сайтов Internet — так называемых порталов.

Портал — многофункциональный сайт Internet, предлагающий разнообразные услуги: поиск информации, бесплатную электронную почту и т.д.

Литература

1.Информационные системы и поисковые технологии. — РГГУ, 1999, http://s2.vntic.org.ru/iis_intro.htm.

2. Информационно-поисковые системы — http://www.comptek.ru/yandex/yand_about.html.

3. А.Аликберов. Поисковые машины. — http://citforum.ru/win/internet/search/index.shtml.

(Продолжение следует)

ЛИСТАЯ СТРАНИЦЫ

Службы спутникового доступа в Интернет на территории России описывают *Егор Поваляев* и *Андрей Фильчаков* **(КомпьютерПресс, №5/2001, С.32-34).**

Среди технологий широкополосного доступа сегодня можно найти такие, которые по стоимости сопостаимы со стандартным коммутируемым доступом. В статье описываются характеристики и тарифные планы асинхронных систем спутниковой связи с асимметричным доступом (спутниковый канал для приема информации, плюс обычный наземный для организации запросов). Их преимущества и недостатки перечислены в таблице.

Преимущества	Недостатки
Доступно по цене (порядка 250 USD)	Для спутникового доступа необходимо наличие любого другого соединения с Интернетом, посредством которого посылаются запросы пользователя на ресурс (на спутниковую антенну "сбрасывается" запрошенный ресурс)
Скорость получения информации значительно увеличиеается (>100 Кбит/с)	Спутниковая антенна служит только для приема информации (отсылка происходит через Dial-up-соединение), поэтому оперативный обмен информацией происходит не так быстро
Возможность дополнительно принимать спутниковые телевизионные каналы	Зависимость от метеорологических условий
Использование push-технологий. Низкая абонентская плата	В случае если спутник имеет большую загрузку, передача информации происходит дольше
В некоторых тарифах отсутствуют ограничения по объему принимаемой информации и времени ее принятия (то есть нет платы за трафик, а вносится только постоянная абонентская плата)	

Системы EOL и "НТВ-Интернет" предполагают фиксированную ежемесячную абонентскую плату — примерно 20 USD, независимо от времени работы и объема трафика. В системах DirectPC и HeliosNet абонент платит за фактически полученный по спутниковому каналу объем информации — независимо от времени соединения. Система SpeedCast отличается высокой скоростью и широким выбором тарифов.

"НТВ-Интернет" (www.ntv-internet.ru) предоставляет доступ в Сеть через спутники "Бонум-1" и Eutelsat W4 (36° в.д.). Прием может осуществляться с помощью стандартного комплекта оборудования "НТВ+". Скорость канала — 365 Кбит/с.

Europe Online (EOL). Реальная скорость работы этой системы зависит от множества факторов и составляет в среднем 50...220 Кбит/с для WWW, 120...420 Кбит/с для FTP, 360 Кбит/с для DigitalDownload и при нескольких конкурентных сессиях. Система предоставляет дополнительные услуги, включая доступ к телевизионным каналам и программам формата MPEG-4.

DirectPC. Скорость потока — до 400 Кбит/с. Европейскую территорию охватывает вещание через спутник HotBird3, а Дальний Восток России — спутник PAS2.

HeliosNet. Скорость поступления информации — до 1,5 Мбит/с. Доступны другие услуги передачи данных: централизованное распространение файлов и изображений, репликация баз данных, IP-телевидение, IP-радиовещание и т.д. HeliosNet базируется на двух спутниках. "Ямал 100" (90° в.д.) в частотном С-диапазоне охватывает всю территорию России (кроме Чукотки) и все страны СНГ. Спутник Intelsat 604 (60° в.д.) ведет трансляцию в Ku-диапазоне в DVB-потоке на территорию от Западной Европы до Восточного Урала, а также на большую часть стран СНГ.

SpeedCast (www.speedcast.ru). Максимальная скорость — до 1,5 Мбит/с. В системе задействован транспондер С-диапазона спутника AsiaSat-38 (105,5° в.д.), в зону покрытия входит практически вся территория России.

Об изобретении эргономичной мыши сообщает *Мартин Уильямс* **(ComputerWorld Россия, №17/2001, С.29).**

Мышь, которая по внешнему виду скорее напоминает джойстик, удерживается рукой за рычаг — так же как обычная пишущая ручка. Изобретатель, шведский доктор Ульман (Johan Ullman), считает, что такая конструкция не будет вызывать у пользователей довольно распространенное заболевание, называемое "синдромом длительного напряжения" (repetitive strain injury — RSI).

Поскольку мышь держат как ручку, мышцы предплечья и плеча не испытывают напряжения, рука пользователя находится в более естественном положении и не делает вращательных движений в горизонтальном направлении. По словам доктора Ульмана, работа крупных мышц плеча оправдана, когда человек рубит дрова, но не тогда, когда необходимо выполнять точные действия.

По информации, размещенной на веб-сайте www.ullmanmouse.com, производство новой мыши планируется начать в III кв. 2001 г.

Производительность процессоров Pentium 4 и Athlon сравнивает *Sean Carruthers* **(The Computer Paper, июнь 2001 г.).**

Pentium 4 1,7 ГГц использует системную шину 400 МГц (100 МГц x 4) и содержит встроенный набор расширенных команд SSE2 (Streaming SIMD Extensions), предназначенный для мультимедиа- и Интернет-программ нового поколения. Правда, лишь немногие программы пока используют это преимущество (список обновленных программ можно найти на сайте Intel, http://www.intel.com/home/pentium4/software.htm). Как видно из таблицы, по производительности Pentium 4 в ряде важных программ уступает Athlon с меньшей тактовой частотой. Результаты получены на системе с 256 Мб ОЗУ, GeForce Ultra!, SoundBlaster Live! и с жестким диском 40 Гб ATA (7200 rpm). Pentium 4 испытывался на материнской плате Intel D850GB с памятью PC800 RDRAM (самая быстрая на момент тестирования), Athlon — на плате Gigabyte GA-7DX с памятью PC2100 DDR SDRAM (также самая быстрая на тот момент). В целом, Pentium 4 выгодно применять для работы с мультимедиа, а Athlon — для офисных приложений. Однако интересно, что даже программы аналогичного назначения от разных фирм ведут себя по-разному. Так, Photoshop быстрее работает на Pentium 4, а CorelDraw — на Athlon. Так что, ориентируясь на приобретение нового процессора с материнской платой, имеет смысл сначала задать себе вопрос: а зачем?

Показатели производительности	Intel Pentium 4 1,7 ГГц	Intel Pentium 4 1,5 ГГц	AMD Athlon 1,33 ГГц	AMD Athlon 1,2 ГГц
Общая оценка по Sysmark 2000:				
Создание страниц для Internet	223	200	236	221
Офисная производительность	195	179	246	231
Общая производительность	207	188	242	227
Оценки индивидуальных программ по Sysmark 2000:				
Bryce 4	257	230	306	289
CorelDraw 9	268	237	333	307
Elastic Reality 5.1	264	229	311	285
Excel 2000	207	191	280	251
Naturally Speaking Pro 4.0	227	206	210	200
Netscape Communicator	170	155	239	217
Paradox 9.0	182	168	223	210
Photoshop 5.5	161	151	145	138
PowerPoint 2000	223	203	282	272
Premiere 5.1	156	137	244	227
Word 2000	130	125	205	199
Windows Media Encoder	342	304	234	221
Производительность по 3DMark 2001	3786	3646	3891	3786
Время кодирования MP3 (320kbps, 37:42 CD):				
Creative	3:04	—	3:18	—
eJay	2:45	—	3:25	—

Об адаптере SCSI/USB фирмы Adaptec (www.adaptec.com) сообщает журнал **Hard'n'Soft (№5/2001, С.34).**

Работа с внешними SCSI-устройствами теперь может быть упрощена. Адаптер USBXchange работает с платформами PC и Macintosh при поддержке спецификации USB 1.1. Производитель не рекомендует подключать адаптер к USB-хабам или астроенным в разные устройства разветвитвлям. В комплекте есть дополнительный переходник для подключения устройств, оборудованных не 50-контактным (High-Density Connector 50-pin), а 25-контектным (DB 25-pin) разъемом.

Сначала к выключенному SCSI-устройству подсоединяется вдаптвр USBXchange, после этого устройство должно быть включено, а затем кабель адаптера подключается к разъему USB метеринской платы. Во время первого подключения ОС обнаруживает новое устройство и установливает с компакт-диска драйвер USBXchange. После установки драйвера обнаруживается уже то устройство, которое подключено посредством USBXchange.

Максимально достижимая скорость обмена — 1,5 Мбайт/с.

На вопросах защиты сетей от вредоносных скриптов, ActiveX, HTML в электронной почте, вирусов, "червей", "троянских коней" останавливается *Роберт Виберт* **(Журнал сетевых решний/ LAN, №5/2001, С.104...108).**

По оценкам компании Network Associates, к январю 2001 г. число известных вирусов и "червей" достигло 54 тыс. Наиболее распространенные из них описаны в каталоге Wild List (http://www.wildlist.org).

Рекомендуемые аатором меры по безопасной настройке Netscape и Internet Explorer 5.x приведены в таблице.

Инфицированные объекты могут содержаться в более чем 200 типах файлов, поэтому сканированию при получении должны подвергаться не только файлы, расширение которых содержится в списке, заданном по умолчению в антивирусной программе. Перечислим список расширений потенциально опасных файлов в системе Windows, которые рекомендуется блокировать на межсетевом экрене или на шлюзе электронной почты: .exe, .com, .asf, .ade, .adp, .bas, .bat, .chm, .cmd, .cnt, .cpl, .crt, .css, .hip, .hte, .hto, .inf, .ins, .isp, .js, .jse, .lnk, .mdb, .mde, .msc, .msi, .msp, .mst, .pcd, .pif, .reg, .scr, .set, .shb, .shs, .uri, .vb, .vbe, .vbs, .wsc, .wsf, .wsh.

Настройки защиты при доступе через Internet

Программа	Параметры защиты, обновления	Рекомендуемые настройки
Netscape	Options/Security Preferences/Java или Edit/Preferences/Advanced/Enable Java	Отключить
	JavaScript	Отключить
	JavaScript for Mail and News	Отключить
Internet Explorer 5.x	Tools/Internet Options/Security Restricted Sites: задать **High Security** (высокий уровень защиты), а затем использовать Custom Leval, чтобы изменить следующие параметры:	
	Элементы управления ActivaX, отмеченные как безопасные для использования в сценариях	Отключить
	Software Channel Permissions	High Safety (высокий уровень безопасности)
	User Authentication Logon	Запрос имени пользователя и пароля
	Java	Отключить
	Сайты	Блокировать любые сайты, о которых известно, что они содержат вредоносные программы
	Internet: задать **Medium Security** (средний уровень защиты), а затем использовать Custom Level, чтобы изменить следующие параметры:	
	Заслуживающие доверия сайты	Присвоить значение Medium/ Low Security (средний/низкий уровень защиты)
	Сайты	Добавить сайты, для которых вы хотите разрешить использование ActivaX и Java.
	Запуск программ и файлов в IFRAME	Отключить
	Software Channel Permissions	High Safety (высокий уровень безопасности)
	User Authentication Logon	Запрос имени пользователя и пароля
	Security Updates from Microsoft: загрузить и применить все последние заплатки	

Виртуальный звуковой кабель описывает *Евгений Музыченко* **(КомпьютерПресс, №5/2001, С.188...189).**

Идея драйвера заключается в программном соединении различных звуковых программ, подобно тому как вход и выход блоков звуковой аппаратуры соединяются обычным кабелем.

Программа VAC соединяет несколько звуковых программ в цепочку так, что кеждая очередная программа получает звук в цифровом виде непосредственно от предыдущей, без потери качества звучания. Она позволяет смешивать, размножать и сохранять звуковые сигналы, создвнные программами, которые могут только проигрывать сигнал в реальном времени не звуковой адаптер. К каждому из портов In и Out может быть присоединено любое количество приложений (клиентов). От приложений требуется лишь умение работать со стандартными Wave-устройствами Windows — и ничего больше. Установка и управление драйвером осуществляются стандартными средствами Windows.

Смешивание и передача звуковых данных выполняются внутри VAC строго равномерно, по прерываниям от системного таймера, с тем чтобы каждое виртуальное устройство работало как реальное, обеспечивая заданную скорость звукового потока. Когда один из концов виртуального кабеля свободен (не имеет присоединенной программы), он ведет себя как обычный провод: звук, выводимый в порт Out, теряется, а из порте In вводится абсолютная тишина.

VAC смешивает звуковые сигналы с насыщением (называемым также клиппированием), что позволяет избежать заметных искажений в результате превышения максимальной амплитуды полученного сигнала.

Демонстрационные версии доступны на страницах http://www.ntonyx.com/vac.html (VAC 2.05) и http://www.ntonyx.com/vac111.html (более старая версия, VAC 1.11).

В последующих версиях автор предполагает использовать более недежные алгоритмы передачи звуковых данных, исключающие потери блоков из-зе недостаточного размера буфера в подключенных приложениях, а также ввести визуальную настройку с отображением текущего формата звукового потоке для каждого кабеля, упревление громкостью и преобразование форматов.

Сообщения о новых версиях автор резмещеет на своей веб-странице http://spider.nrcde.ru/music/software.html.

Журналы листал В.Ермоленко.

М.ДОЛИНСКИЙ,
г.Гомель.

Некоторые факты из теории чисел

Введение

Существует значительный класс олимпиадных задач по программированию, простое решение которых предполагает знание учеником следующих разделов из теории чисел:

- свойства X MOD Y (остаток от деления X на Y);
- позиционные системы счисления и быстрое вычисление многочлена;
- показатель, с которым данное простое P входит в произведение N!;
- свойства НОД (наибольшего общего делителя).

Изложению этих вопросов и иллюстрации их применения для решения конкретных задач и посвящена данная статья. В целом она продолжает серию материалов [1...7], направленных на повышение уровня подготовки школьника к участию в олимпиадах по информатике.

Свойства X MOD Y

Пусть X и Y — целые числа. Поделив нацело X на Y, мы получим два числа — частное (обозначим его Q) и остаток (обозначим его R).

В Паскале для нахождения Q и R имеются соответствующие операции:

```
Q := X div Y;
R := X mod Y;
```

Свойство 1. (A+B)modY=(A mod Y + B mod Y) mod Y

Прочитать это свойство можно, например, так:

Для того чтобы найти остаток от деления суммы двух чисел A и B на число Y, достаточно найти остатки от деления чисел A и B на Y по отдельности, полученные числа сложить, и уже для вычисленной суммы взять остаток от ее деления на Y.

Пример 1.

Пусть требуется вычислить остаток от деления числа 197 на 5.

Традиционный способ предполагает деление числа 197 на 5 уголком:

```
 197 |5
-15  39
 47
-45
  2
```

Таким образом, частнов — 39, а остаток — 2.

Но ведь частное у нас никто не спрашивал — значит, скорей всего, мы выполнили лишнюю работу. А давайте подсчитаем, сколько всего действий было выполнено:

1. 3*5=15
2. 19-15=4
3. 9*5=45
4. 47-45=2

На самом деле мы выполнили даже еще больше действий. Ведь чтобы найти цифры 3 и 9 частного, мы должны были пользоваться подбором такой цифры, чтобы ее произведение на 5 было максимальным, но меньше 19 в первом случае (когда мы подбирали цифру 3) и меньше 47 во втором случае, когда мы подбирали цифру 9.

А теперь попробуем решить эту же задачу с помощью свойства 1:

```
197 mod 5 = (190 + 7) mod 5 = (применяем свойство 1)
((190 mod 5) + (7 mod 5)) mod 5 = (0 +2) mod 5 = 2
```

Очевидно, что так вычислить ответ проще (особенно в более общем случае — при произвольном A).

Свойство 2. (A*B) mod Y = ((A mod Y)*(B mod Y)) mod Y

Прочитать это свойство можно, например, так:

Для того чтобы найти остаток от деления произведения двух чисел A и B на число Y, достаточно найти остатки от деления чисел A и B на Y по отдельности, полученные числа перемножить, и уже для вычисленного произведения взять остаток от его деления на Y.

Пример 2.

Пусть требуется вычислить остаток от деления числа 100 на 7. Традиционный способ предполагает деление числа 100 на 7 уголком:

```
 100 |7
-7   14
 30
-28
  2
```

Таким образом, частное — 14, а остаток — 2.

Теперь попробуем найти тот же результат с помощью свойства 2.

```
100 mod 7 = (10 * 10) mod 7 = (применяем свойство 2) =
((10 mod 7) * (10 mod 7)) mod 7 = (3 * 3) mod 7 =
9 mod 7 = 2
```

Надеюсь, не требуется доказательств того, что вторым способом результат находить проще.

Позиционные системы счисления и быстрое вычисление многочлена

Рассмотрим многочлен (полином)

$a_0 + a_1 * X + ... + a_{n-1} * X^{n-1} + a_n * X^n$.

И пусть нам задан массив коэффициентов многочлена a[0], a[1], ... a[n] и некоторое число x, для которого нужно вычислить значение этого многочлена.

Тогда в Паскаль-программе можно написать, например, так:

```
s := 0; y := 1;
for i:= 0 to n do
  begin
    s := s + a[i]*y;
    y := y * x;
  end;
```

В Паскале нет оператора возведения в степень, и потому приходится делать это в программе умножением в строке — `y := y * x;`. Математики же для сокращения записи многочленов используют знак суммы:

$$\sum_{i=0}^{n} выражение$$

Читается следующим образом: *сумма при i от 0 до n выражений, которые стоят за знаком суммы.*

С использованием этого знака многочлен может быть записан так:

$$\sum_{i=0}^{n} A_i \cdot x^i$$

А правила вычисления остатка от деления многочлена, вытекающие из свойств 1 и 2, могут быть записаны следующим образом:

1.
$$\left(\sum_{i=0}^{n} A_i \cdot x^i\right) mod\, M = \left(\sum_{i=0}^{n} A_i \cdot (x^i \bmod M)\right) mod\, M$$
(для $A_i < M$)

Прочитать это правило можно, например, так: *для вычисления остатка от деления многочлена на M можно вычислять остатки деления на M степеней числа x, затем находить нужную сумму произведений и уже для нее находить остаток от деления на M.*

2. $x^i \bmod M = (x * (x^{i-1} \bmod M)) \bmod M$ *(для $x < M$)*

В случае, если $x < M$, для нахождения остатка от деления x в степени i на M достаточно вначале найти остаток от деления x в степени $i-1$ на M, затем умножить его на x и найти остаток от деления на M вычисленного произведения.

3. $(P + Q) \bmod M = (P \bmod M + Q) \bmod M$ *(для $Q < M$)*

В случае, если одно из слагаемых меньше M, то остаток от деления суммы на M равен остатку от деления на M суммы этого слагаемого с остатком от деления на M второго слагаемого.

Теперь вернемся к программе вычисления полинома:

```
s := 0; y := 1;
for i := 0 to n do
```

```
begin
  s := s + a[i]*y;
  y := y * x;
end;
```

Некоторые читатели уже заметили, наверно, что можно было написать и так:

```
s := a[0]; y := x;
for i := 1 to n do
  begin
    s := s + a[i]*y;
    y := y * x;
  end;
```

Тогда количество действий и сложения, и умножения стало бы на одно меньше! Теперь их выполняется n и 2n соответственно.

А можно ли написать эту программу более оптимально в смысле сокращения количества действий — сложений и/или умножений?

Итак, пусть имеем полином:

$A = a_0 + a_1x_1 + a_2x^2 + ... + a_Nx^N \ \ (0 \le a_i < N)$,

который представляет число *A* в позиционной системе счисления, где a_i — цифры, *x* — основание системы счисления.

Обозначим операцию возведения в степень значком ^. Тогда формула вычисления полинома может быть переписана следующим образом:

$A = a_0 + a_1x + a_2x^2 + ... + a_Nx^n \ \ (0 \le a_i < N)$

Обозначив a_i как *a[i]* и поменяв порядок членов в полиноме на обратный, получим:

*A = a[N]*x^N +a[N-1]*x^[N-1] + ... + a[0]*

Введем обозначения:

S[N] = a[N]

*S[N-1] = S[N]*x + a[N-1] = a[N]*x + a[N-1]*

*S[N-2] = S[N-1]*x + a[N-2] = a[N]*x^2 + a[N-1]*x + a[N-2]*

...

*S[0] = S[1]*x + a[0] = a[N]*x^N + a[k-1]*x^[N-1] + ... + a[0]*

Теперь алгоритм вычисления полинома выглядит следующим образом:

```
S[N]=a[N]                                  ! БЫСТРОЕ ВЫЧИСЛЕНИЕ
Для i от N до 1  S[i-1]=S[i]*x+a[i-1] ! ПОЛИНОМА (многочлена)
```

А соответствующий фрагмент программы:

```
s[n] := a[n];
for i := n downto 1 do s[i-1]:= s[i]*x + a[i-1];
```

Более того, поскольку каждый раз для вычисления следующего значения S нам нужно только одно предыдущее, этот фрагмент программы может быть еще упрощен:

```
s := a[n];
for i:= n downto 1 do s := s*x + a[i-1];
```

И окончательный вариант программы, которая по заданным аргументу x и коэффициентам полинома *a[i]* вычисляет сам полином, может выглядеть следующим образом:

```
program poly3;
var
  a              : array [0..100] of longint;
  s, y, n, x,i   : longint;
begin
  readln(x)
  readln(n)
  for i:=0 to n readln (a[i]);
  s := a[n];
  for i:= n downto 1 do s := s*x + a[i-1];
  writeln (s);
end.
```

Ну и чем она лучше ранее написанной программы, в которой было n сложений и 2n умножений? Тем, что в ней ровно n сложений и n умножений. То есть умножений стало в 2 раза меньше. И при достаточно больших значениях n программе потребуется в 2 раза меньше времени на выполнение. Существуют задачи, для которых это очень важно.

Теперь рассмотрим задачу об остатке полинома:

Задача 1. Дано число в K-ичной системе счисления

$a_n a_{n-1} ... a_0 \ \ (K \le 36)$.

Найти остаток от деления его на M. Числа K, n, M, как и остаток от деления на M, представляются в десятичной системе счисления. Идея решения — воспользоваться свойствами MOD и алгоритмом быстрого вычисления полинома.

Если у нас K — основание системы счисления, то число

$a_n a_{n-1} ... a_0$

нужно вычислять следующим образом:

*S = a[n]*K^n + a[n-1]*K^[n-1] + ... + a[0].*

А соответствующий алгоритм выглядит следующим образом:

```
Ввод n, a[0..n], K, M

S = a[n] mod M                  Быстро вычисляем остаток
Для i от n до 1                 от деления полинома на M,
  S = ( S*K + a[i-1] ) mod M    попутно пользуясь свойствами
                                 MOD
Вывод S
```

А программа, решающая поставленную задачу, может выглядеть следующим образом:

```
program poly4;
var
  a              : array [0..100] of longint;
  s, n, i, K, M  : longint;
begin
  read(n)
  for i:= 0 to n read (a[i]);
  read(K,M)
  s := a[n] mod M;
  for i:= n downto 1 do s := (s*K + a[i-1]) mod M;
  writeln (s);
end.
```

Еще одна задача.

Задача 2. Число вводится своим двоичным представлением (длина числа не превышает 10000 двоичных разрядов). Необходимо определить, делится ли число на 15.

Рассуждения будем проводить в шестнадцатиричной системе счисления. К ней легко перейти, заменяя во введенном числе тетрады на соответствующие шестнадцатиричные цифры справа налево! АНАЛОГИЧНО ПРИЗНАКУ ДЕЛИМОСТИ НА 9 В ДЕСЯТИЧНОЙ СИСТЕМЕ СЧИСЛЕНИЯ — если сумма шестнадцатиричных цифр делится на 15, то и число делится на 15. Ниже приводится доказательство:

$$\sum_{i=0}^{n} a(i)\cdot 16^i = a(0) + \sum_{i=1}^{n} 16\cdot a(i)\cdot 16^{i-1} =$$

$$= \sum_{i=0}^{n} a(i) + \sum_{i=1}^{n} 15\cdot a(i)\cdot 16^{i-1}$$

В результате преобразований образовались два слагаемых. Второе делится на 15, так как каждый элемент суммы имеет сомножитель 15. Таким образом, исходное число делится на 15, если сумма цифр числа *a(i)* делится на 15.

Алгоритм решения задачи может быть записан следующим образом:

```
Ввод n, A[0..n]                 a[i] - коэффициент при 2^i

K = (n MOD 4)+1                 Количество 16-х цифр в исходном числе
Преобразование A в H[0..K]      Строим массив 16-х цифр (числа 0 - 15)

S = H[0]                        В S заносим младшую цифру
Для i от 1 до K                 Для всех остальных цифр
  S = ( S + H[i] ) mod 15       Складываем цифру с S и берем остаток

Если S=0                        Если сумма цифр делится на 15,
  то       делится             то и число делится,
  иначе  не делится            иначе - нет

ПРЕОБРАЗОВАНИЕ A -> H

Для i от n+1 до 4*K-1  A[i]=0   Дополняем нулями до полной
                                старшую тетраду
Для i от 0 до K-1               Формируем по тетраде 16-ю цифру H[i]
   m    = 4*i
   H[i] = A[m] + A[m+1]*2 + A[m+2]*4 + A[m+3]*8
```

А полный текст программы таков:

```
program poly5;
var
  a,h              : array [0..10000] of byte;
  s, y, n, i, K, m : longint;
begin
  readln(n)
  for i := 0 to n read (a[i]);
  K := (n mod 4) + 1;
  for i:= n+1 to 4*K-1 do a[i]:=0;
```

```
  for i: = 0   to K-1   do
    begin
      m    := 4*i;
      H[i] := A[m] + A[m+1]*2 + A[m+2]*4 + A[m+3]*8
    end;
  s := H[n] mod 15;
  for i:= n downto 1 do s := (s + H[i-1]) mod 15;
  if s=0 then writeln ('Yes') else writeln('No');
end.
```

Формула вхождения простого множителя в N!

Рассмотрим число N! Очевидно, что оно представимо в виде *N! = 2^k2*3^k3* 5^k5*7^k7* ... *P^kP.*

Где:

- *k2* — количество 2 в разложении N!;
- *k3* — количество 3 в разложении N!;
- *k5* — количество 5 в разложении N! и т.д.

Показатель kP, с которым данное простое P входит в произведение N!, равен:

$$kP = \left[\frac{N}{P}\right] + \left[\frac{N}{P^2}\right] + \left[\frac{N}{P^3}\right] + \ldots \quad (\text{пока } N > P^i),$$

где [Y] обозначает целую часть числа Y, а P^i обозначает P в степени i.

Пояснение смысла формулы:

- каждое P-ое число делится на P;
- каждое P^2 -ое число делится на P^2;
- каждое P^3 -ое число делится на P^3;

...

для всех *i*, пока *N > P^i*.

Задача 3. Вводится число N. Необходимо найти, на сколько нулей оканчивается выражение N!=1*2*3*...*N.

Идея решения:

*N! = 2^k2 * 3^k3 * 5^k5 * 7^k7* ...,*

где:

- k2 — количество 2 в разложении N!
- k3 — количество 3 в разложении N!
- k5 — количество 5 в разложении N!

...

K равно количеству нулей и равно min(k2, k5) = k5, поскольку пятерок в разложении значительно меньше чем двоек. Тогда

$$K = \left[\frac{N}{5}\right] + \left[\frac{N}{25}\right] + \left[\frac{N}{125}\right] + \ldots$$

Алгоритм решения:

```
Ввод N
   K = N div 5,  D = K
    Пока ( D >= 5 )                Пока есть что прибавлять,
    D = D div 5,   K = K + D       вычисляем и прибавляем.
```

И полный текст программы, решающей поставленную задачу, таков:

```
program pfact;
var
  N, K, D   : longint;
begin
  readln(N);
  K := N div 5; D := K;
  While ( D>=5 ) do
    begin D := D div 5;  K := K + D; end;
  writeln(K);
end.
```

Свойства НОД (наибольший общий делитель)

По определению, наибольшим общим делителем (НОД) двух чисел *a* и *b* называется наибольшее из чисел, являющихся делителями одновременно и числа *a*, и числа *b*.

Например, для чисел 12 и 18 наибольшим общим делителем является число 6. Действительно, делители числа 12— 1, 2, 3, 4, 6, 12; делители числа 18 — 1, 2, 3, 4, 6, 9, 18.

В школьном курсе математики для нахождения наибольшего общего делителя предлагался следующий алгоритм, естественно вытекающий из определения НОД — *найти все делители каждого из чисел и выбрать наибольший из общих делителей.*

В информатике применяются различные ускоренные варианты нахождения НОД двух чисел *a* и *b*, основанные на следующих свойствах НОД:

$$1.\ НОД(a,b) = \begin{cases} НОД(a-b,b) & \text{если } a > b \\ a & a = b \\ НОД(a,b-a) & a > b \end{cases}$$

$$2.\ НОД(a,b) = \begin{cases} 2 \cdot НОД(a/2, b/2) & \text{если } a \text{ четное} & b \text{ четное} \\ НОД(a/2, b) & a \text{ четное} & b \text{ нечетное} \\ НОД(a, b/2) & a \text{ нечетное} & b \text{ четное} \\ НОД(a, |b-a|/2) & a \text{ нечетное} & b \text{ нечетное} \end{cases}$$

3. НОД (a, b)= НОД (a-b, b)= НОД (a+b, b) (a > b)

Примером может служить программа Euclid, находящая НОД с помощью рекурсивной функции NOD(a,b);:

```
program euclid;
var
  a, b : longint;
  function NOD (a,b:longint):longint;
    begin
      if a>b
        then NOD := NOD (a-b,b)
        else
          if a<b
            then NOD := NOD (a,b-a)
            else NOD := a;
    end;

begin
  readln (a,b);
  writeln (NOD (a,b));
  readln
end.
```

У читателей, не знакомых с понятием рекурсивных процедур и функций, справедливо возникли вопросы — а что это такое, в чем отличие рекурсивных процедур и функций и т.д.? Но это, как говорится, совсем другая история. Теме рекурсии, рекурсивных процедур и функций предполагается посвятить отдельную статью.

Заключение

Данная статья продолжает серию публикаций, ориентированных на самообучение программированию [1...7]. Очевидно, что для закрепления данного материала необходимо проверить полученные знания на компьютере, как самостоятельным решением поставленных в данном материале задач, так и применением полученных знаний к другим задачам. Полезно также ознакомиться с соответствующими материалами учебных пособий, например [7].

Для проверки полученных знаний автор рекомендует использовать сайт Гомельского государственного университета им.Ф.Скорины— http://dl.gsu.unibel.by. Важно подчеркнуть, что пользоваться ресурсами этого сайта для проверки собственных решений предлагаемых там задач можно как в режиме Internet-online, так и с помощью E-mail посредством почтового робота, отослав для начала пустое письмо по адресу dl-service@gsu.unibel.by.

Литература

1. М.Долинский. Памятка участнику олимпиады по информатике среди школьников. — Радиолюбитель. Ваш компьютер, 2000, №1, С.20.

2. М.Долинский. Начинаем программировать самостоятельно. — Радиолюбитель. Ваш компьютер, 2000, №№1-6.

3. М.Долинский. Решение задач на тему "Геометрия на плоскости". — Радиолюбитель. Ваш компьютер, 2000, №8, С.21; №9, С.19.

4. М.Долинский. Разработка программ с использованием определяемых программистом типов данных. — Радиолюбитель. Ваш компьютер, 2001, №1, С.29; №3, С.26.

5. М.Долинский. Решение задач с помощью стека и очереди. — Радиолюбитель. Ваш компьютер, 2001, №4, С.26.

6. М.Долинский. Обзор возможностей языка программирования ПАСКАЛЬ. — Радиомир. Ваш компьютер, 2001, №№7, 8.

7. Котов В.М., Волков И.А., Лапо А.И. Методы алгоритмизации. Учебное пособие для 9-го класса общеобразовательной школы с углубленным изучением информатики. — Мн.: Народная асвета, 1997.

Программирование разветвляющихся алгоритмов

А.ШАКИРИН,
г.Минск, БАТУ, каф. ВТ.

Цель этой статьи — освоить использование простейших компонентов-переключателей и создать приложение, которое использует разветвляющийся алгоритм.

1. Пример создания приложения

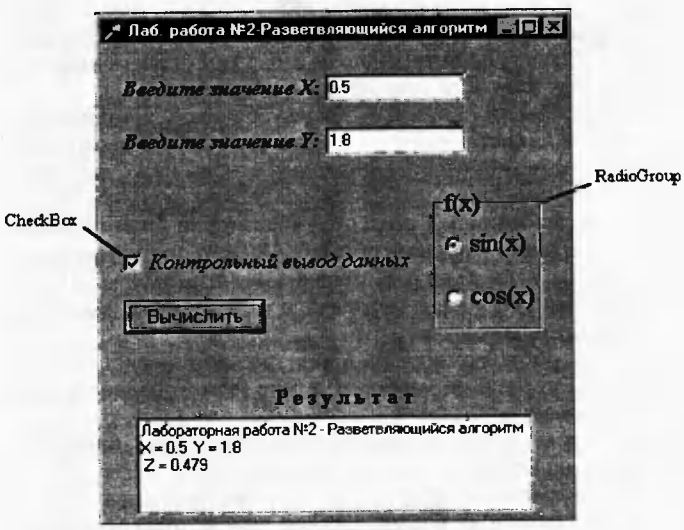
Необходимо создать Windows-приложение для вычисления выражения

$$Z = \begin{cases} f(x), x < y \\ y, \text{иначе} \end{cases},$$

где $f(x)=sin(x)$, либо $f(x)=cos(x)$. В панели интерфейса следует предусмотреть возможность управления контрольным выводом исходных данных.

1.1. Размещение компонентов на Форме

Будем размещать компоненты на Форме так, чтобы они соответствовали панели, показанной на рисунке.



При создании приложений в Delphi часто используются компоненты в виде кнопок-переключателей. Состояние такой кнопки (включено-выключено) визуально отражается на Форме. На панели представлены кнопки-переключатели двух типов — CheckBox и RadioGroup.

Компонент **CheckBox** организует кнопку независимого переключателя, с помощью которой пользователь может указать свое решение типа "да/нет".

Компонент **RadioGroup** организует *группу кнопок* — зависимых переключателей. При нажатии одной из кнопок группы все остальные кнопки выключаются.

Поместите на Форму компоненты Label, Edit и Memo в соответствии с рисунком. Выберите в Палитре Компонентов на странице Standard пиктограмму компонента CheckBox и разместите ее в нужном месте Формы. В свойстве Caption Инспектора Объектов замените надпись **CheckBox1** на ***Контрольный вывод данных***. Чтобы при запуске приложения кнопка CheckBox оказалась включена, свойство Checked установите равным True.

Выберите в Палитре Компонентов Standard пиктограмму компонента RadioGroup и поместите ее в нужное место Формы. В свойстве Caption измените заголовок **RadioGroup1** на **f(x)**. Для размещения кнопок в один столбец, свойство Columns установите равным 1. Дважды щелкните мышью по правой части свойства Items — появится строчный редактор списка наименований кнопок. Наберите две строки с именами: в первой строке — sin(x), во второй — cos(x), и нажмите OK. После этого на Форме появится группа из двух кнопок — переключателей с соответствующими надписями. Чтобы при запуске приложения первая кнопка RadioGroup оказалась включена, свойство ItemIndex установите равным 0.

1.2. Создание процедур обработки событий FormCreate и Button1Click

Технология создания процедур обработки событий FormCreate и Button1Click ничем не отличается от описанной в [1]. Внимательно наберите операторы этих процедур, используя текст модуля UnRazvAlg.

1.3. Текст модуля UnRazvAlg

```
Unit UnRazvAlg;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics, Controls,
Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Label1: TLabel;
    Edit1: TEdit;
    Label2: TLabel;
    Edit2: TEdit;
    Label4: TLabel;
    Memo1: TMemo;
    Button1: TButton;
    RadioGroup1: TRadioGroup;
    CheckBox1: TCheckBox;
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
  private
    { Private declarations }
  public
    { Public declarations }
  end;

var
  Form1: TForm1;

implementation
{$R *.DFM}
// Процедура обработки события создания Формы:
procedure TForm1.FormCreate(Sender: TObject);
begin
 Edit1.Text:='0.5';      // начальное значение X
 Edit2.Text:='1.8';      // начальное значение Y
 Memo1.Clear;            // очистка  Memo1
// Вывод строки в  Memo1:
 Memo1.Lines.Add('Лабораторная работа №2 - Разветвляющийся '+
'алгоритм');
end;
// Процедура обработки события нажатия кнопки Button1:
procedure TForm1.Button1Click(Sender: TObject);
 var
  x,y,z,fx : extended;      // объявление локальных переменных
begin
 x:=StrToFloat(Edit1.Text); // X присваивается содержимое Edit1
 y:=StrToFloat(Edit2.Text); // Y присваивается содержимое Edit2
 fx:=sin(x);                // fx присваивается начальное значение
// Выбор функции, соответствующей нажатой кнопке:
 case RadioGroup1.ItemIndex of
  0:fx:=sin(x);
  1:fx:=cos(x);
 end;
 // Вычисление выражения:
 if x<y then
          z:=fx
         else
          z:=y;
// Проверка состояния кнопки CheckBox1:
  if CheckBox1.Checked then
                        Memo1.Lines.Add('X = '+Edit1.Text+
```

```
  '  Y = '+Edit2.Text);    // контрольный вывод X, Y, Z в Memo1
                           // Вывод результата в Memo1:
 Memo1.Lines.Add(' Z = '+FloatToStrF(z,ffFixed,8,3));
end;
end.
```

Если нажата первая кнопка RadioGroup1, в переменную целого типа RadioGroup1.ItemIndex заносится ноль, если вторая — единица. Если кнопка CheckBox1 нажата, логическая переменная CheckBox1.Checked имеет значение True, если нет — False.

1.4. Работа с приложением

Запустите созданное приложение. Используя все управляющие компоненты панели интерфейса, убедитесь в правильном функционировании приложения во всех предусмотренных режимах работы.

2. Индивидуальные задания

Необходимо создать Windows-приложение для вычисления соответствующего выражения и протестировать его работу. На панели интерфейса следует предусмотреть возможность выбора одной из трех функций f(x): sh(x), x^2, e^x.

1. $a=\begin{cases}(f(x)+y)^2-\sqrt{f(x)y}, & x\le 0\\ (f(x)+y)^3+\sqrt{|f(x)y|}, & 0<x\le 1\\ (f(x)+y)^4+1, & \text{иначе.}\end{cases}$

2. $b=\begin{cases}ln(f(x))+\left(f(x)^2+y\right)^3, & x/y>0\\ ln|f(x)/y|+(f(x)+y)^2, & x/y<0\\ \left(f(x)^3+y\right)^2, & -2\le x<2\\ 0, & \text{иначе.}\end{cases}$

3. $c=\begin{cases}f(x)^2+y^2+sin(y), & x+y>0\\ \left(f(x)-y^3\right)^2+cos(y), & x>0,\ y<0\\ (y-f(x))^2+tg(y), & x-y<0.\end{cases}$

4. $d=\begin{cases}f^2(x)+arctg(f(x)), & 1\le x<5\\ (y-f(x))^2+arctg(f(x)), & y>x\\ (y+f(x))^3+0.5, & \text{иначе.}\end{cases}$

5. $e=\begin{cases}\sqrt[i]{f(x)}, & i-\text{нечетное},\ x>0\\ i/2\sqrt{|f(x)|}, & i-\text{четное},\ x<0\\ \sqrt{|if(x)|}, & \text{иначе.}\end{cases}$

6. $g=\begin{cases}e^{f(x)-|b|}, & 0.5<xb<10\\ \sqrt{|f(x)+b|}, & 0.1<xb<0.5\\ 2f(x)^2, & \text{иначе.}\end{cases}$

7. $s=\begin{cases}e^{f(x)}, & 1<xb<10\\ \sqrt{|f(x)+4\cdot b|}, & 12<xb<40\\ bf(x)^2, & \text{иначе.}\end{cases}$

8. $j=\begin{cases}sin\left(5f(x)+3m|f(x)|\right), & -1<m<x\\ cos\left(3f(x)+5m|f(x)|\right), & x\le m\\ (f(x)+m)^2, & \text{иначе.}\end{cases}$

Литература

1. А.Шакирин. Программирование линейных алгоритмов. — Радиомир. Ваш компьютер, 2001, №8, С.21.

2. В.В.Феофанов. Delphi 3. Учебный курс. — М.: Нолидж, 1981.

3. Э.Возневич. Delphi. Освой самостоятельно. — М.: Восточная книжная компания, 1996.

4. Дж.Матчо, Д.Р.Фолкнер. Delphi. — М.: БИНОМ, 1995.

5. М.Канту. Delphi 2 для Windows 95/NT. — М.: ООО "Малип", 1997.

HI-TECH,
E-mail: serrgio@gorki.unibel.by

НИЗКОУРОВНЕВОЕ ПРОГРАММИРОВАНИЕ

Обращение к читателям

— Что может сказать собака о сущности дзена?
— Гав, — скажет собака.
Одна из многочисленных дзенских баек

Киньте грязью в того, кто вам скажет, что Ассемблер — очень сложный для изучения язык. И никогда не читайте глупых книг, в которых написана подобная чушь. О том, что это очень сложно, говорят и пишут люди, у которых в свое время не хватило смелости (и/или способностей) попытаться "въехать" в "машинные коды", "прерывания", "порты ввода-вывода" и прочую низкоуровневую "чепуху", с которой рано или поздно сталкивается любой профессиональный программист. Можно сколько угодно ругать глюки в "винде", кривой SQL в Delphi, Билла Гейтса или "эту проклятую зидовскую мамку" — это не избавляет от элементарного невежества в области "компьютерных технологий".

Мы, авторы этого курса — "Низкоуровневое программирование для дZенствующих" — считаем, что принципы функционирования компьютера и некоторые основы низкоуровневого программирования (наверное, последнее точнее будет назвать "кодированием") относятся к категории элементарных знаний, владеть которыми обязан КАЖДЫЙ программист.

В общем, если у вас есть желание изучить ЭТО — читайте дальше. Был бы ученик толковый, а уж учителями мы постараемся стать хорошими :).

Короче, добро пожаловать в мир, где программист — хозяин компьютера, а не наоборот; в мир низкоуровневого программирования.

Что такое "низкоуровневое программирование", наверное, понятно, но почему оно "для дZенствующих"?

Такое название выбрано по нескольким причинам.

1. "По приколу" (с нашим плоским компьютерным юмором вы будете сталкиваться постоянно).

2. Потому что программирование для нас — намного больше, чем просто программирование. Во-первых, низкоуровневое программирование — это состояние души. Во-вторых, это состояние сознания; использование некоторых медитативных приемов для настройки на рабочий лад; иногда — состояние транса (например, при отладке программы); и в конце концов — полная нирвана (когда, наконец, написанная на чистом asm'е программа — работает!).

3. А теперь самая главная причина. Авторы проекта — люди дZенствующие, "по жизни". Понимайте это как хотите...

Материал, который мы вам предлагаем — это адаптированная и слегка переработанная версия одноименной рассылки, которую мы периодически распространяем через сервисы subscribe.ru и maillist.ru (7500 подписчиков).

Весь этот "бред" (самокритика) был протестирован на "полных чайниках" и доведен до такой степени "удобоваримости", что, чтобы не понять его, нужно уж очень постараться.

Скажем сразу, определенный круг лиц был недоволен нашим способом и стилем "преподношения" материала. Они говорили, что это некрасиво, что это несерьезно, что это непедагогично... Это в основном были люди старшего поколения, "творящие" свои произведения по принципу — "нужна публикация". Это были люди, часто имеющие смутное представление о программировании, но весьма успешно подтверждавшие свою "компетентность" обилием ссылок на различные "авторитетные" источники, преемствующие, наряду с опытом предшественников, также и их технические ошибки... :(

Молодое же поколение (т.е. те, для кого мы это все "рожали") приняло наш курс "на ура", выражая полнейшее одобрение "специфическому" подходу к методике обучения и стилю изложения материала. Они отстаивали нас перед лицом многочисленных маститых критиков, они говорили нам: "вы ДЛЯ КОГО это пишете?? Для нас?? Вот и ориентируйтесь НА НАС!!". Они давали нам советы, они фокусировали наше внимание на удачных и неудачных местах курса, они присылали нам свои статьи и свое видение отдельных программерских вопросов... В минуту нашего отчаяния (было и такое), они ругали нас благим матом, требуя ПРОДОЛЖЕНИЯ...

В общем, тяжел и тернист был авторский труд :). То, что вы сейчас читаете — далеко от совершенства и имеет ряд серьезных недостатков, но мы с гордостью, от имени 7500 человек, которые систематически прочитывали нашу рассылку, можем сказать, ЧТО ЭТОТ УЧЕБНЫЙ КУРС — ИМЕННО ТАКОЙ, КАКИМ ВЫ ХОТЕЛИ ЕГО ВИДЕТЬ!

Наш сайт вы можете найти по адресу http://hi-tech.nsys.by. Называется он "САЙТ ИЗОБРЕТАТЕЛЕЙ ФОНАРИКОВ НА СОЛНЕЧНЫХ БАТАРЕЙКАХ" (только не спрашивайте, почему). Там вы можете найти разнообразные справочники, документацию и программное обеспечение по "ассемблерной" (да и не только) тематике. Для ответов на глупые вопросы (а также для обсуждения вопросов особо умных) создана mail-конференция, и работает экспертная группа (http://hi-tech.nsys.by/forum/).

С.Воробьев, А.Белоусов, А.Беженарь, А.Вербицкий, К. Кошелев, В.Буторин

ВВЕДЕНИЕ

Все же попытаемся отговорить вас от "колупания" низкого уровня. Ну подумайте хорошенько — зачем вам это нужно?! В Delphi приложения за пять минут ваяются, а на "асме" знаете сколько?!

У вас что, забот больше в жизни нет? Хотите писать программы — пишите. В конце концов, если есть задача — для ее решения любые способы хороши. И не имеет большого значения ни объем откомпилированной программы, ни ее требовательность к ресурсам — "машины" нонче знаете какие пошли?! Сплошная крутизна и мегагерцы!

А многочисленные среды для создания приложений!? Ни строчки кода! Сплошное кликанье мышкой! Ведь это так круто!

Прогу написать? В два счета! (Я крутой, крутой, крутой). Открываем форму. Небрежная пробежка по многочисленным панелям с компонентами... Кликаем на нужный, перетаскиваем на форму... готово! Объект Button1 типа TButton! Хотите — в виде звездочки будет выглядеть? Ща — только 1st Class проинсталлю...

Готово! Окно программы, с КНОПКОЙ — за 5 с! Полноценное окно и кнопка полноценная! А в окне даже менюшка выскакивает — если в левом вернем углу нажать! И все это — одним легким небрежным движением! Какой там Ассемблер? Delphi — вот это вещь! Бац-бац — и слабал какое-нить бухгалтерское приложение. Бац — и продал его подороже с умным видом какому-нибудь мелкому промышленному предприятию. А бабки "состриг" — и на пиво есть, и на Интернет тоже. Хорррошая жизнь! А коли жизнь хороша — можно и в арканоида с чистой совестью...

А Ассемблер? Ну что такое Ассемблер? Средневековому мракобесию сродни. Дьявольщина и сатанизм сплошной. Черная магия! Демонов каких-то из преисподней вызывать... Заклинания... call-jmp-mov-nop-ret, чур меня! А талмуды ассемблерные??? Это же вообще еретические труды, чертями писаные!

Не, ребята! Не нужен вам asm, ой как не нужен! И страшно, и не пишет на нем уже никто давным-давно!

На улицу достаточно выйти, чтоб все понять. Где вы видели многоэтажный дом из кирпича? Из блоков все сделано — из блоков! Ну и что, что разваливается? Строителям тоже жить на что-то нужно! Ну прикиньте, что было бы, если бы каждый выстроенный дом не нужно было обслуживать, ремонтировать, что-нибудь переподгонять, замазывать, маскировать, переделывать... А то и было бы, что строители бы тогда без работы остались — а кому это нужно?!

Понятна аналогия? Изучение Ассемблера — это есть большая идеологическая ошибка. Asm как язык программирования должно предать забвению. И его же, но как состояние души — на "костер" визуального программирования! И MMX-у туда побольше с 3Dfx-ом... знаете, как рванет?

И не нужно надеяться на то, что скоро будет достигнут "предел минимизации", когда уже нельзя будет на одной и той же площади расположить большее количество транзисторов. Тогда, мол, железо будет уже невозможно совершенствовать — начнут совершенствовать софт. Оптимизировать процедуры, избавляться от балласта и все такое. И тут ваши знания и востребуются. Не верьте! Говорили, что "предел" — это 0,18 микрон, а в газетах пишут, что уже по 0,15-микронной технологии до черта чего делается! А Intel, так тот вообще в выходящем Tualatin на 0,13 микрон перешел...

ЧАСТЬ 1. РАЗМИНАЕМ ПАЛЬЦЫ

1.1. Система счисления

Вы читали "Хроники Амбера" Роджера Желязны? Там есть такой эпизод:

Главный герой находится в заточении. В абсолютной тьме. У него были выколоты глаза, но за год они регенерировали, и зрение постепенно к нему возвращается...

И однажды каким-то чудом в одной камере с ним оказывается загадочный Дворкин — создатель Лабиринта. Именно "чудом" — он просто появился неизвестно откуда. Он тоже находится "в заключении", но, в отличие от Корвина (главного героя), может спокойно ходить через каменные стены.

Удивленный Корвин спрашивает его:

— Как ты оказался в моей камере? Ведь здесь нет дверей.

Дворкин отвечает:

— Двери есть везде. Просто нужно знать, как в них войти.

#1. Наверняка среди ваших знакомых есть "крутые" программисты, или люди, таковыми себя считающие ;). Попробуйте как-нибудь проверить их "на вшивость". Предложите им в уме перевести число 12 из шестнадцатеричной в двоичную систему счисления. Если над подобным вопросом "крутой программист" будет думать дольше 10 с — значит, он вовсе не так крут, как говорит.

#2. Система счисления (сие не подвластное человеческой логике опредвление взято из математической энциклопедии) — это совокупность приемов представления обозначения натуральных чисел. Этих "совокупностей приемов представления" существует очень много, но самая распространенная ныне — та, которая подчиняется позиционному принципу. Согласно этому принципу, один и тот же цифровой знак имеет различные значения в зависимости от того места, где он расположен. Такая система счисления (number system) основывается на том, что *n* единиц (radix — основание) объединяются в единицу второго разряда, *n* единиц второго разряда объединяются в единицу третьего разряда и т.д.

В книжках пишут, что позиционная система счисления — самая "совершенная", но вы им не верьте. Например, "непозиционный код Грея" имеет целый ряд преимуществ перед позиционными кодами. Более того, строго говоря, в математике форма ПРЕДСТАВЛЕНИЯ ЧИСЕЛ не имеет значения, поскольку само число от этого не меняется — позиционность удобна разве что при ручном применении законов арифметики. К тому же, существуют системы с плавающим основанием — например, система счисления времени, у которой в основаниях и 1000, и 100, и 365/366, и 24, и 60...

Для начала давайте разберемся с азами позиционной системы, так как именно ее мы пока что и будем юзать ;).

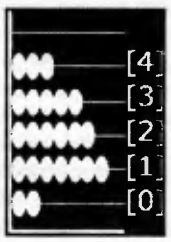
#3. "Разрядам" нас учили еще в начальных классах школы. Например, у числа 35672 цифра "2" имеет первый разряд, "7" — второй, "6" — третий, "5" — четвертый и "3" — пятый. А "различные значения" цифрового знака "в зависимости от того места, где он расположен" и "объединение в единицу старшего разряда" на тех же уроках арифметики "объяснялось" следующим образом:

35672 = 30000 + 5000 + 600 + 70 + 2

35672 = 3*10000 + 5*1000 + 6*100 + 7*10 + 2*1

$35672 = 3*10^4 + 5*10^3 + 6*10^2 + 7*10^1 + 2*10^0$ (1)

#4. Очень наглядно это отображают обыкновенные бухгалтерские счеты. Набранное на них число 35672 будет выглядеть следующим образом (см. рисунок):



Чтобы набрать число 35672, мы должны передвинуть влево две "костяшки" на первом "прутике", семь — на втором, шесть — на третьем, пять — на четвертом и три — на пятом. (У нас ведь одна "костяшка" на втором прутике — это то же самое, что и десять "костяшек" — на первом, а одна на третьем, равна десяти на втором, и так далее) Пронумеруем наши "прутики" снизу вверх — да так, чтобы номером первого был "0". И снова посмотрим на наши выражения (на то, что жирным выделено):

$35672 = \mathbf{3}*10^4 + \mathbf{5}*10^3 + \mathbf{6}*10^2 + \mathbf{7}*10^1 + \mathbf{2}*10^0$

Это (если сверху вниз считать) сколько на каждом "прутике" "костяшек" влево отодвинуто.

$35672 = 3*10^4 + 5*10^3 + 6*10^2 + 7*10^1 + 2*10^0$

(тут, по идее, степени выделены, только это не очень заметно).

Это номер прутика (самый нижний — 0), на котором отодвинуто определенное число костяшек.

$35672 = 3*\mathbf{10}^4 + 5*\mathbf{10}^3 + 6*\mathbf{10}^2 + 7*\mathbf{10}^1 + 2*\mathbf{10}^0$

А это на каждом прутике — по 10 костяшек нанизано, не все влево отодвинуты, но всего-то их — 10!

Кстати, цифра 10 в последнем выражении соответствует основанию системы счисления.

Медитируйте...

#5. Пальцев на руках у человека — 10, поэтому и считать мы привыкли в системе счисления с основанием 10, то есть в десятичной. Если вы хорошо представляете себе счеты и немного поупражнялись в разложении чисел аналогично выражению 1, то перейти на систему счисления с основанием, отличным от привычной, особого труда для вас не составит. Нужно всего лишь представить себе счеты, на каждый прут которых нанизано не 10 костяшек, а, скажем, 9, или 8, или 16, или 32, или 2, и... попробовать мысленно считать на них.

#6. Для обозначения десятичных чисел мы используем цифры от 0 до 9, для обозначения чисел в системах счисления с основанием менее 10 мы используем те же цифры:

```
radix 9 – 0, 1, 2, 3, 4, 5, 6, 7, 8;
radix 8 – 0, 1, 2, 3, 4, 5, 6, 7;
radix 2 – 0, 1.
```

Если же основание системы счисления больше десяти, то есть больше чем десять привычных нам чисел, то начинают использоваться буквы английского алфавита. Например, для обозначения чисел в системе счисления с основанием 11 "как цифра" будет использоваться буква A:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A.

В системе счисления с основанием 16 — буквы от A до F:

0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

И так далее...

Правда, при определенном основании (при каком?) буквы англицкого алфавита закончатся. Но для нас это пока не имеет значения, так как работать мы будем только с тремя radix'ами — 10 (естественно), 16 и 2. Правда, если кто на старых системах DEC или на "содранных" с них ДВК поизучать это дело собирается, тому еще и radix 8 понадобится.

#7. Числа в любой системе счисления строятся аналогично десятичной. Только на "счетах" не с 10, а с другим количеством костяшек.

Например, когда мы пишем десятичное число 123, то имеем в виду следующее:

1 раз 100 (10 раз по 10)
+ 2 раза 10
+ 3 раза 1

Если же мы используем символы 123 для представления, например, шестнадцатеричного числа, то подразумеваем следующее:

1 раз 256 (16 раз по 16)
+ 2 раза 16
+ 3 раза 1

Короче — полный беспредел. Говорим одно, а подразумеваем другое. И последнее — не для красного словца сказано. А потому что так оно и есть...

Истина где-то рядом...

#8. Трудность у вас может возникнуть при использовании символов A, B, C и т.д. Чтобы решить эту проблему раз и навсегда, необходимо назубок вызубрить ма-а-аленькую табличку "соответствия" между употребляемыми в "компьютерном деле" системами счисления.

RADIX 10	RADIX 16	RADIX 2
0	0	0
1	1	1
2	2	10
3	3	11
4	4	100
5	5	101
6	6	110
7	7	111
8	8	1000
9	9	1001
10	A	1010
11	B	1011
12	C	1100
13	D	1101
14	E	1110
15	F	1111

Следуя этой таблице, число 5BC в шестнадцатеричном формате "строится" так:

5 раз по 256 (16 раз по 16)
+ 11 раз по 16 (11 — потому что по таблице B как бы равно 11)
+ 12 раз по 1 (12 —потому что по таблице C как бы равно 12)

А теперь, если пораскинуть мозгами, с легкостью переведем 5BC из шестнадцатеричной в десятичную систему счисления:

5*256 + 11*16 + 12 = 1468.

Вот и объединили цифры с буквами. Пространство со временем поучимся объединять немного позже — если не испугаетесь сложностей низкоуровневого программирования.

В общем-то, решать вам. В Delphi тоже много чего объединять можно...

#9. Двоичная (бинарная) система по-компьютерному обзывается "bin" (от binary), "родная" десятичная — "dec" (от decimal), а шестнадцатеричная — "hex" (от hexadecimal). Это так компьютерщики обозвали те системы счисления, с которыми имеют дело. А обозвали потому, что у них ведь полный бардак в голове, оказывается!

Например, 10 — что это за число? Да это вообще не число! Палка и барабан — и только. А вот 10d или же 10_10 — уже понятно, это — число, соответствующее количеству пальцев на обеих руках. И именно на обеих, а не на двух. Почему не на двух? А потому что на двух в какой системе? Если в двоичной, так это на десяти! То бишь — 100, если в десятичной. Вот и придумали программисты после числа буковку писать — b, d или h. А самые ленивые еще и директиву специальную придумали — напишут в самом начале программы какой-нибудь ".radix 16", и все числа, которые без этих букв, будут автоматически приниматься за шестнадцатеричные.

#10. Еще немного про перевод между "радиксами" (вообще-то, это плевое дело, конечно, если представлять себе, что такое "совокупность приемов представления обозначения натуральных чисел").

Например, преобразование числа 42936 из десятичного в шестнадцатеричный формат проводится следующим образом (в скобках — остаток):

```
42936/16 = 2683(8)   ; 8 - младшая цифра
2683/16 = 167(11)    ; B (11d=Bh по таблице )
167/16 = 10(7)       ; 7
10/16 = 0(10)        ; A - старшая цифра
42936d=A7B8h
```

А вот и обратный процесс — перевод из HEX в DEC числа A7B8h:

```
10*16=160 (10 - потому что Ah=10d); 160+7=167
167*16=2672; 2672+11=2683
2683*16=42928; 42928+8=42936
A7B8h=42936d
```

Преобразования чисел в системы счисления с другим основанием проводятся аналогично. Счеты, обыкновенные бухгалтерские счеты, только с "плавающим" числом "костяшек" на каждом "прутике"!

Медитируйте...

#11. Если честно, то конкретный "рисунок" цифр, когда единица "палкой" обозначается, двойка — "лебедем", — это всего лишь историческая случайность (в игре "Что? Где? Когда?" на вопрос о цифрах был дан ответ, что значение цифры изначально соответствовало количеству углов в исходном арабском начертании). Мы запросто можем считать в "троичной" системе счисления с цифрами %, *, _ (где запятая — это знак препинания, а вовсе не число):

%, *, _, *%, **, *_, _%, _*, __, *%%, *%*, *%_, **%...

Или использовать родные цифры в десятичной системе счисления, но по другому "вектору упорядоченных цифр" — 1324890576:

1, 3, 2, 4, 8, 9, 0, 5, 7, 6, 31, 33, 34, 34,38, 39, 30, 35, 37...

Правда, при этом немножко затрудняется понимание происходящего? А ведь тоже десятичная система! И рисунок цифр как бы знакомый :-))).

Или вообще считать в 256-ричной системе счисления, используя в качестве "рисунка цифр" таблицу ASCII-символов!

#12. Теперь самая интересная часть Марлезонского балета :))).

Компьютер, как известно, считает только в двоичной системе счисления. Человеку привычна десятичная. Так зачем еще и шестнадцатеричную какую-то знать нужно?

Все очень просто. В умных книжках пишут, что "шестнадцатеричная нотация является удобной формой представления двоичных чисел". Что это значит?

Переведите число A23F из шестнадцатеричной "нотации" в двоичную (один из возможных алгоритмов приведен в п.10). В результате длительных манипуляций у вас должно получиться —1010001000111111.

А теперь еще раз посмотрите на таблицу в п.8 (которую вы как бы уже и выучили) и попробуйте то же самое сделать в уме:

```
Ah=1010b
2h=0010b
3h=0011b
Fh=1111b
A23Fh = 1010'0010'0011'1111b
```

Каждой шестнадцатеричной цифре соответствует тетрада (4 штуки) ноликов и единичек. Все, что потом нужно сделать — "состыковать" эти тетрады. Круто? Вас еще не то ждет!

#13. Кстати (наверняка вы это уже знаете):

00000123 = 123, но!

123 <> 12300000

Но это так, кстати...

#14. И, напоследок, еще несколько слов про HEX и BIN :). Зайдите в Norton Commander, наведите указатель на какой-нибудь файл и нажмите F3. А когда он откроется — на F4. Там какие-то "нездоровые цифирьки" попарно сгруппированы. Это и есть "нолики и единички" (которыми в компьютере все-все-все хранится), но в шестнадцатеричном формате...

Следует основательно разобраться с системой счисления. Минимум, что должен вынести из этой главы "юзвер", вступивший на скользкий путь низкоуровневого программирования — это научиться переводить числа между DEC, HEX и BIN... хе-хе... В УМЕ!

1.2. Поднятие боевого духа

Прежде чем начать программировать на "низком уровне", необходима небольшая теоретическая подготовка. Однако мы допускаем, что небольшое теоретизирование в самом начале курса отпугнет определенное количество программеров. А посему, прямо сейчас мы предлагаем вам написать вашу первую низкоуровневую программу :)

Внимательно следуйте приведенной ниже инструкции, и очень быстро достигнете потрясающего результата!!!

1. По "адресу" (предположительно) C:/WINDOWS/COMMAND/ находится одна замечательная программулька, ее "имя собственное" — DEBUG.EXE. Перекреститесь и запустите ее.

Все, что у вас должно появиться на мониторе — это белый "-" на черном фоне. Только это не просто "минусик", а ПРИГЛАШЕНИЕ. Приглашение что-нибудь ввести.

2. Вот и вводим буковку "a" (английскую, регистр значения не имеет).

После нажатия на Enter, на монитор должно вывалиться что-то похожее на 119B:0100. Это "адрес памяти" (сегмент плюс смещение), в который мы сейчас запихнем нашу программу (первая часть адреса может быть любая).

3. Белым по черному пишем загадочную команду — int 18. Жмем на Enter.

Вывалится что-нибудь вроде 119B:0102. Игнорируем и снова жмем на Enter. Должен снова появиться "-". Снова приглашение.

4. Вводим после "минусика" — r cx. В ответ на это напечатается — CX 0000. И ниже — ":". После этого самого ":" вводим цифру "2" и снова жмем на Enter. Снова появится приглашение — "-".

5. Дальше на "приглашение" отвечаем вот что — n reboot.com. Жмем на Enter. Снова "минусик" выскочил.

6. Последний шаг. Вводим букву (команду) "w" после "минусика" и жмем на Enter. Должно всплыть сообщение — "Запись: 00002 байт". Если оно появилось, значит, все нормально.

7. Вводим команду "q" (от английского Quit), жмем на Enter и вываливаемся из этой программульки.

8. Теперь "по адресу" C:\WINDOWS\COMMAND\ у вас находится еще одна программулька — reboot.com размером всего лишь в 2 байта (мы ее только что написали). Запустите ее из "чистого" DOS и посмотрите, чего будет...

"Фишка" заключается вот в чем: эта двухбайтовая (!) прога (написанная, кстати, без использования компилятора!), перезагружает компьютер (если из DOS), да еще и матерится целой строкой — "PRESS ANY KEY FOR REBOOT".

В свое время нас — чайников — это поразило 8-). "В два байта целая строка влезла!! Ух ты!!".

Эх... было время золотое...

Кстати, если у вас ничего не получится по этой инструкции — не расстраивайтесь. Мы вам еще расскажем, почему это может не получиться.

Обещаем! ;)

Д.ЯМАЙКИН,
E-mail:
thydmirty@hotmail.com

Бегущие рядом — два эффекта синхронности

Посмотрите на... например, WinAmp, и на его дополнительные окошки. Сцепил их вместе — и таскаешь всю конструкцию за главное окно. Рассмотрим два способа, как можно повторить (или почти повторить) этот эффект. Первый исполнен на чистом Delphi (использовался Delphi 5), второй же требует небольшой экскурсии в API.

Способ 1. Очень удобный, но не совсем правильный. К тому же, имеющий пару незначительных недостатков, которые будут подробно рассмотрены. Основан этот способ на событии *OnCanResize*. Ниже в примере будут использоваться формы, но, тем не менее, он подходит для любого объекта, обладающего этим событием. Событие *OnCanResize* вызывается для формы не только тогда, когда она меняет размеры, но и при изменении положения. В примере будет участвовать цепочка форм, которые по очереди будут передавать от "головы" к "хвосту" сообщение о движении.

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    Timer1: TTimer;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure FormCanResize(Sender: TObject; var NewWidth,
      NewHeight: Integer; var Resize: Boolean);
    procedure Timer1Timer(Sender: TObject);
    procedure Timer1Destroy(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action: TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
    tLeft, tTop: integer; // переменные, хранящие позицию
                          // текущего окна
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

uses Unit2;

var
 Tail:TForm2=nil;
// первая форма - управляющая, и поэтому она отличается от
// "хвостовых" окошек
 NextItem:TForm2=nil;

// по нажатию кнопки создается новое окомко и помещается в
// "хвост"
procedure TForm1.Button1Click(Sender: TObject);
begin
 Tail.NextItem:=TForm2.Create(Self);
 Tail.NextItem.Left:=Tail.Left+40;
 Tail.NextItem.Top:=Tail.Top+40;
 Tail:=Tail.NextItem;
 Tail.Visible:=True;
end;

// при создании главной формы создается также и "хвостик"
procedure TForm1.FormCreate(Sender: TObject);
begin
 Tail:=TForm2.Create(Self);
 NextItem:=Tail;
 NextItem.OnCanResize:=nil;
 NextItem.Left:=Form1.Left-100;
 NextItem.Top:=Form1.Top-50;
 NextItem.OnCanResize:=NextItem.FormCanResize;
 NextItem.Visible:=True;
end;

// "костяк", при движении формы по цепи передается сообщение
procedure TForm1.FormCanResize(Sender: TObject; var NewWidth,
  NewHeight: Integer; var Resize: Boolean);
begin
 Timer1.Enabled:=False;
 Timer1.Enabled:=True;
 if NextItem<>nil then NextItem.MoveIt(Left-tLeft,Top-tTop);
 tLeft:=Left;
 tTop:=Top;
end;

procedure TForm1.Timer1Timer(Sender: TObject);
var
 CR:boolean;
 NW,NH:integer;
begin
 CR:=False;
 NW:=100;
 NH:=100;
 if NextItem<>nil then NextItem.OnCanResize(NextItem,NW,NH,CR);
end;

procedure TForm1.Timer1Destroy(Sender:TObject);
begin
 NextItem.Visible:=False;
 if NextItem.NextItem<>nil then NextItem:=NextItem.NextItem
else PostQuitMessage(0);
end;

procedure TForm1.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
 Timer1.Enabled:=False;
 Timer1.Enabled:=True;
 Timer1.OnTimer:=Timer1Destroy;
 Action:=caNone;
 Visible:=False;
end;

end.
```

Последние процедуры требуют некоторого разъяснения.

Процедура *FormClose* перебрасывает обработчик таймера на *Timer1Destroy*, и, перед тем как выгрузиться из памяти, программа по очереди прячет окошки от "головы" к "хвосту" — просто так, для красоты.

Процедура *PostQuitMessage* — API-функция, говорящая приложению, что ему пора закрыться. Закрывать программу вызовом *PostQuitMessage* корректнее, чем используя Halt. Еще можно сделать *Application.Terminate*, но лично мне этот способ не нравится.

Процедура *Timer1Timer*, как понятно из названия, а также из некоторых общих соображений, "висит" на таймере и кое-как скрашивает недостаток *OnCanResize*, о котором будет рассказано позже. Если не вдаваться в подробности — благодаря таймеру "хвост" выравнивается, если какое-то время (точно равное задержке таймера) ничего не делать. Благодаря каким-то внутренним особенностям системы, таймеру удается "глохнуть", пока, например, окошко перетаскивается мышкой с места на место. Но это уже мелочи.

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics,
Controls, Forms, Dialogs,
  ExtCtrls, jpeg;

type
  TForm2 = class(TForm)
    Timer1: TTimer;
    Image1: TImage;
    procedure FormCanResize(Sender: TObject; var NewWidth,
      NewHeight: Integer; var Resize: Boolean);
    procedure FormCreate(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action:
TCloseAction);
    procedure Timer1Timer(Sender: TObject);
    procedure Image1MouseDown(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Image1MouseMove(Sender: TObject; Shift:
TShiftState; X,
      Y: Integer);
    procedure Image1MouseUp(Sender: TObject; Button:
TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
    NextItem:TForm2;
    tLeft,tTop:integer;
    procedure MoveIt(dLeft,dTop:integer);
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

procedure TForm2.MoveIt;
begin
 Left:=Left+dLeft;
 Top:=Top+dTop;
end;

procedure TForm2.FormCanResize(Sender: TObject; var
NewWidth,
  NewHeight: Integer; var Resize: Boolean);
begin
 if NextItem<>nil then NextItem.MoveIt(Left-tLeft,Top-tTop);
 tLeft:=Left;
 tTop:=Top;
end;

procedure TForm2.FormCreate(Sender: TObject);
begin
 tLeft:=Left;
 tTop:=Top;
end;

procedure TForm2.FormClose(Sender: TObject; var Action:
TCloseAction);
begin
 Action:=caNone;
end;

procedure TForm2.Timer1Timer(Sender: TObject);
var
 CR:boolean;
 NW,NH:integer;
begin
 CR:=False;
 NW:=100;
 NH:=100;
 if NextItem<>nil then NextItem.OnCanResize(NextItem,NW,NH,CR);
end;

{*************************************************************************}
var
 DraggingM:boolean;
 deltaX, deltaY:integer;

procedure TForm2.Image1MouseDown(Sender: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
 deltaX:=Left-Mouse.CursorPos.X;
 deltaY:=Top-Mouse.CursorPos.Y;
 DraggingM:=(Button=mbLeft);
end;

procedure TForm2.Image1MouseMove(Sender: TObject; Shift:
TShiftState; X, Y: Integer);
begin
 if DraggingM then
  begin
   Left:=Mouse.CursorPos.X+deltaX;
   Top:=Mouse.CursorPos.Y+deltaY;
  end;
end;

procedure TForm2.Image1MouseUp(Sender: TObject; Button:
TMouseButton;
  Shift: TShiftState; X, Y: Integer);
begin
 DraggingM:=not(Button=mbLeft);
end;

end.
```

Начнем с "хвоста". Три последние процедуры отвечают за расположение "хвостовых" формочек на экране — можно взяться за любую точку такой формы и перетащить ее туда, куда следует (или хочется). Не точку, а, конечно, саму форму. Все остальные процедуры — те же самые, что и для головной формы.

Рассмотрим теперь недостатки этой схемы.

Первое, что бросается в глаза — вся конструкция (несколько окошек) жутко "трясется", пока ее перетаскиваешь с места на место. На WinAmp не очень похоже, ну разве что, на очень "паралитический" :-). Почему? Как это ни странно, из-за собственно события *OnCanResize*, которое почему-то происходит со сдвигом — т.е. не сразу, когда окошко двигается, а потом, когда оно двигается второй раз. Виною этому, наверное, та же мышка, которая тащит окно. Тем не менее, если смотреть на это как на интересный визуальный эффект, а не на недостаток, то получается довольно симпатично.

Второй недостаток не столь заметен, но его можно получить, "слепив" достаточно длинный "хвост". Некоторые сообщения от головной формы не успевают отрабатываться, и "голова" "отрывается". Т.е. оказывается не там, где ей положено относительно "хвоста". Иногда отрываются части собственно "хвоста"...

Способ 2. Как говорится, есть способ лучше. Я не знаю, насколько точно он отражает то, как сделано именно в WinAmp, но, как минимум, он обеспечивает некоторую синхронность.

При передвижении окно получает сообщение *WM_MOVE*. В параметре *lParam* этого сообщения содержатся новые координаты окна: в младшем слове — смещение по горизонтали, в старшем — по вертикали. Сообщение приходит без задержки, и поэтому легче держать синхронность окошек. Хотя замечу, что это, в свою очередь, нагружает саму систему.

Программа изменяется незначительно, поэтому я не буду повторять одинаковые процедуры:

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics,
  Controls, Forms, Dialogs, StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Button1: TButton;
    procedure Button1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure Timer1Destroy(Sender: TObject);
    procedure FormClose(Sender: TObject; var Action:
                                          TCloseAction);
  private
    { Private declarations }
  public
    { Public declarations }
    tLeft, tTop: integer;
    procedure CatchMove(var M:TMessage); message WM_MOVE;
  end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

uses Unit2;

procedure TForm1.CatchMove;
begin
 if NextItem<>nil then NextItem.MoveIt(Left-tLeft,Top-tTop);
 tLeft:=Left;
 tTop:=Top
end;

end.
```

Во втором юните изменений немного больше:

```
unit Unit2;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics,
  Controls, Forms, Dialogs, ExtCtrls, jpeg;

type
  TForm2 = class(TForm)
    Image1: TImage;
    procedure FormClose(Sender: TObject; var Action:
                                        TCloseAction);
    procedure Image1MouseDown(Sender: TObject; Button:
                                          TMouseButton;
      Shift: TShiftState; X, Y: Integer);
    procedure Image1MouseMove(Sender: TObject; Shift:
                        TShiftState; X, Y: Integer);
    procedure Image1MouseUp(Sender: TObject; Button:
                                          TMouseButton;
      Shift: TShiftState; X, Y: Integer);
  private
    { Private declarations }
  public
    { Public declarations }
    ForcedMove:boolean;
    tLeft,tTop:integer;
    NextItem:TForm2;
    procedure MoveIt(dLeft,dTop:integer);
    procedure CatchMove(var M:TMessage); message WM_MOVE;
  end;

var
  Form2: TForm2;

implementation

{$R *.DFM}

procedure TForm2.MoveIt(dLeft,dTop:integer);
begin
 ForcedMove:=True;
 SetBounds(Left+dLeft,Top+dTop,Width,Height);
 if NextItem<>nil then NextItem.MoveIt(dLeft,dTop);
end;

procedure TForm2.CatchMove;
begin
 if not(ForcedMove) then if NextItem<>nil then
NextItem.MoveIt(Left-tLeft,Top-tTop);
 tLeft:=Left;
 tTop:=Top;
 ForcedMove:=False;
end;

end.
```

Добавленный механизм с переменной *ForcedMove* предотвращает некорректный перерасчет при передаче сообщения по цепи. Если этот механизм убрать, то получится интересный эффект — "хвост" начинает "носить" по экрану, особенно же "летают" последние окошки. Почему так получается? Не надо забывать, что когда окно двигается (само по себе или нет), то получает сообщение *WM_MOVE* и два раза передвигает следующее окно.

С этим хорошим синхронным способом есть одна проблема, достаточно неприятная. Если дополнительных окошек мало — все в полном порядке, и движение выглядит просто отлично. Но стоит количеству окошек подобраться к десяти, как система перестает справляться. Впрочем, я использовал непрямоугольные формочки (этого в самом листинге программы нет), если же использовать стандартные очертания окошек (что-то вроде квадратов и прямоугольников), то какие-либо протормаживания начинают быть заметны где-то после пары дюжин окошек.

Эти два способа — отнюдь не единственные, даже на их базе можно построить несколько вариантов. Так что — выбор есть.

Литература

1. Голова. — Справочное пособие на все случаи жизни.
2. Visual Studio 6.0 Help.

Рисунок О.Попова

А.ЗАЙЦЕВ,
602265, Владимирская обл.,
г.Муром,
а/я 427.

Как сделать карманный справочник

Информацию, хранящуюся в компьютере, иногда бывает необходимо перенести на бумагу. Результат такого переноса обычно выглядит как стопка листов формата А4, скрепленная с одной стороны с помощью степлера. Но удобнее было бы иметь книжку меньшего формата, в виде обычной тетради из вложенных друг в друга согнутых листов или (если страниц много) состоящую из нескольких тетрадей, сшитых вместе. Как же быть?

В принципе, можно получить тетрадь формата А5, если на каждой стороне листа А4 разместить по две страницы. Если жв на одной стороне разместить сразу четыре страницы, то после разрезания таких листов из них можно изготовить удобную "карманную" книжку формата А6. Разбивка текста на страницы необходимого формата не составит особого труда — с этим справится даже простейший текстовый редактор. Трудности возникают при расстановке страниц в определенной последовательности, чтобы после сборки книжки они шли по порядку. Для этого и предназначена предлагаемая мной программа. Она написана на упрощенной версии Бейсика и может быть легко приспособлена к любому компьютеру, а при необходимости — усовершенствована. Но сначала давайте договоримся о терминологии.

Страница — отдельная страница текста, "вписанная" в необходимый формат с учетом полей. После разбивки текста на страницы становится известным число страниц в книге — N.

Тетрадь — конструкция из вложенных друг в друга сомкнутых листов, скрепленная по сгибу. Число страниц в тетради — n.

Книга — законченное изделие, содержащее весь необходимый текст. Может состоять из одной или нескольких тетрадей, в зависимости от числа страниц. Число тетрадей в книге — P.

Печатный лист — лист, который распечатывается на принтере. Число печатных листов в книге — S, в тетради — K.

Если лист предполагается только сгибать, на нем размещаются четыре страницы (по две с каждой стороны). В этом случае параметр d равен единице (на рис.1 приведен пример расчета для d=1, P=3). Если исходный лист будет разрезаться на две части, число страниц увеличивается до восьми и параметр d равен двум (на рис.2 показан пример расчета для d=2, P=1). Число страниц в тетради не рекомендуется выбирать больше чем 40...50. Если книга получается толстой, лучше разбить ее на несколько тонких тетрадей.

Программа запрашивает число страниц (N), число тетрадей (P) и значение параметра d. После этого она рассчитывает число печатных листов, и если оно оказывается не целым числом, предлагает увеличить число страниц. Это можно сделать, вставив чистые "титульные" листы или добавив в конце книги "листы для заметок".

Рис. 1

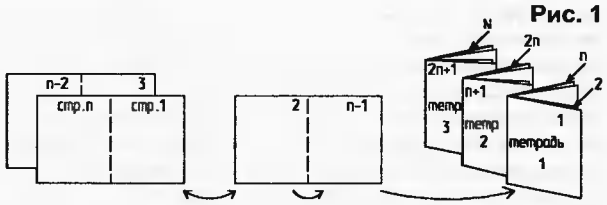
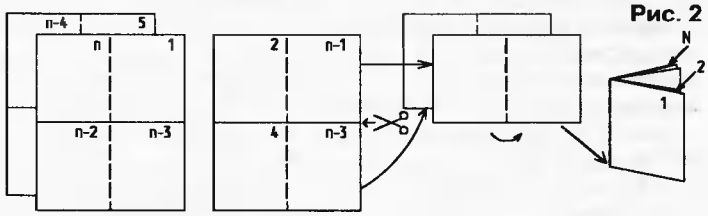


Рис. 2



Затем программа выводит расчетное количество печатных листов и список страниц в порядке их размещения на листах, а также соответствующие им номера тетради — J и листа в пределах тетради — I. С помощью тексового редактора страницы размещаются в соответствии с этим списком.

Если для работы используется матричный принтер, удобнее распечатывать листы полностью — сначала сторона 1, затем сторона 2, после этого распечатывается титульный лист. При использовании более современных принтеров удобнее распечатать все листы со стороны 1, затем перевернуть стопку листов (при этом лист 1 окажется сверху) и распечатать сторону 2. Только не зарядите листы в принтер "вверх ногами"!

После этого листы разрезаются (если требуется), сгибаются, разбираются по тетрадям (если P>1), скрепляются — и книжка готова.

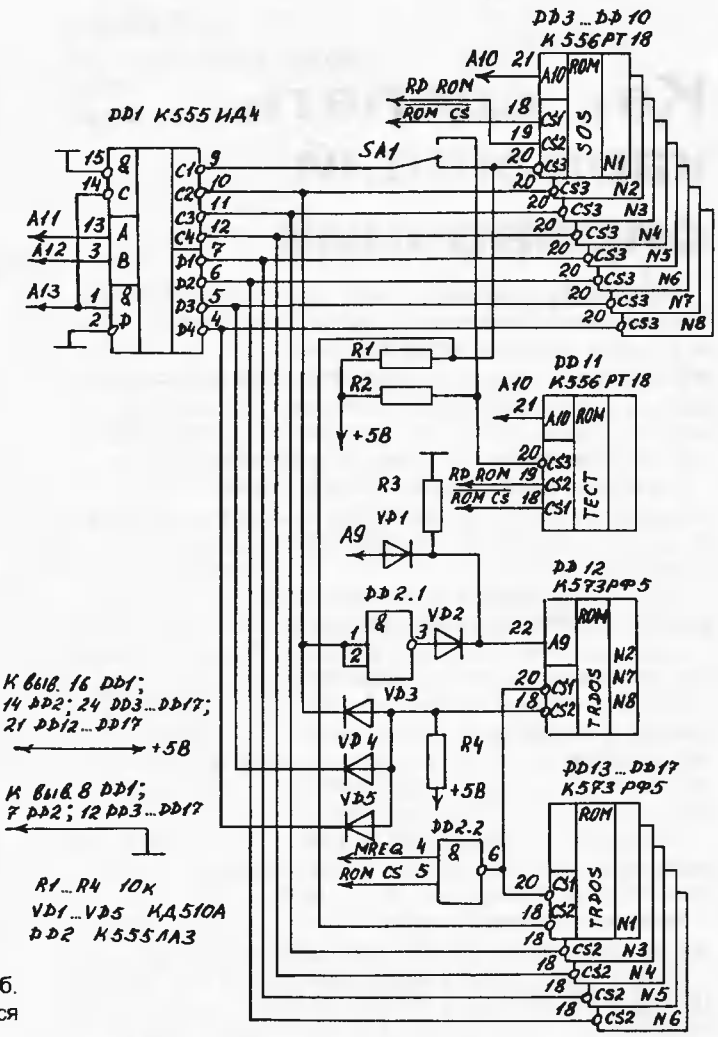
```
  10 REM Программа расстановки страниц
  20 INPUT "N=", N
  30 INPUT "P=", P
  40 INPUT "d=", d
  50 K= N/(4*P*d)
  60 IF K-INT(K)=0 THEN GOTO 90
  70 K=INT(K)+1: N=4*P*d*K
  80 PRINT "увеличить N="; N
  90 S=K*P: PRINT "число листов S="; S
 100 PRINT "I"; TAB 5; "I"; TAB 12; "Сторона 1"; TAB 22;
"Сторона 2"
 110 FOR J=1 TO P
 120 X=K*(J-1)/(4*d): Y=K*J/(4*d)
 130 FOR I=1 TO K
 140 A=2*d*(I-1)
 150 PRINT I; TAB 5; I; TAB 13; Y-A; TAB 18; X+A+1; TAB
23; X+A+2; TAB 28; Y-A-1
 160 IF d=1 THEN GOTO 180
 170 PRINT TAB 13; Y-A-2; TAB 18; X+A+3; TAB 23; X+A+4;
TAB 28; Y-A-3
 180 NEXT I
 190 NEXT J
 200 END
```

Замена микросхем ПЗУ в микро-ЭВМ на примере "ZX-SPECTRUM"

В.СОЛОНИН,
г.Конотоп.

В результате научно технического-прогресса повышается степень интеграции микросхем, в том числе и микросхем ПЗУ. У разработчиков микро-ЭВМ появляется возможность использовать микросхемы постоянной памяти большей емкости, повысив этим надежность и уменьшив размеры и потребляемую мощность компьютера. Применение микросхем ПЗУ большой емкости не устраивает радиолюбителя, у которого в запасв таких микросхем нет, или имеющийся программатор не программирует такие микросхемы. Микросхема постоянной памяти может сгореть или потерять записанную в ней информацию, и нужно будет восстанавливать ПЗУ. Набрать без ошибок программу емкостью 8 Кб довольно трудно. Дело бы упростилось, если бы был программатор, подключенный к компьютеру с дисководом, тогда можно было бы переносить программу с диска. Но такая техника еще не изготовлена. Даже новая микросхема с ультрафиолетовым стиранием через 5 лет может самопроизвольно стереть-

ся (а если окошко не закрыто — то и через 1 год) , поскольку информация в ней хранится за счет зарядов затворов транзисторов, которые не могут не разряжаться со временем. Поэтому раз в год, чтобы не рисковать, приходится с помощью программатора восстанавливать заряды или заменять микросхему, потому что если информация исказится, то может оказаться, что неискаженную информацию найти негде, и ремонт компьютера станет невозможным. Чтобы убедиться в правильности информации, записанной в ПЗУ, нужно иметь возможность проверки ее по контрольным суммам. Такую функцию выполняют тесты "ZX-SPECTRUM", программы которых занесены в отдельные микросхемы. Тестовая микросхема устанавливается на место микросхемы ПЗУ компьютера. Однако если микросхема ПЗУ будет отключена, то она не будет проверяться тестом. В результате тест укажет на ошибку контрольных сумм ПЗУ. Если ПЗУ составлено из микросхем емкостью 2 Кб, то тест будет проверять контрольные суммы каждой микросхемы, кроме первой, вместо которой включена тестовая микросхема. При этом тестовую микросхему можно установить постоянно и включать при необходимости с помощью переключателя. Самое главное преимущество такого разбиения ПЗУ на части емкостью 2 Кб заключается в том, что появляется возможность использовать доступные микросхемы с прожигаемыми перемычками, информация в которых не искажается со временем, и ее не нужно периодически восстанавливать. Если приобретена плата малых размеров, использующая микросхемы ПЗУ большой емкости, то задача размещения 17 микросхем емкостью по 2 Кб каждая решается просто. Их можно спаять параллельно в стопку, одну над другой. Кто раньше изготавливал устройства, использующие ПЗУ, например музыкальный звонок или другие схемы на микропроцессоре, тот имеет в запасе микросхемы емкостью 2 Кб, которые можно будет использовать и не тратиться на покупку новых ИМС. Если какая-то одна микросхема сгорит (на что укажет постоянно задействованный тест — все же сразу сгореть не могут), то набрать на программаторе информацию емкостью 2 Кб, имея только бумажную распечатку, не так уж сложно — это не 8 Кб. Если в ПЗУ компьютера имеется свободное место, то появится возможность экономии микросхем.



На схеме, приведенной на рисунке, показано включение 15 (а не 17, так как 2 сэкономлено) микросхем емкостью 2 Кб для "ZX-SPECTRUM", в которых записаны программы SOS (DD3...DD10) компьютера, TRDOS (DD12...DD17), контроллера дисковода и теста (DD11). При этом использован всего лишь один дешифратор DD1 на микросхеме К555ИД4. Адресные пространства программ SOS и TRDOS разбиты на 8 равных по длине областей — N1...N8 по 2 Кб в каждой. Каждая область занесена в отдельную микросхему, кроме областей N2, N7, N8 TRDOS, которые удалось поместить в одну ИМС DD12 благодаря наличию в ней свободного места. Для этого использован дешифратор на элементах VD1...VD5, DD2.1, R3, R4, позволяющий создать простую его конструкцию, припаяв диоды и резисторы непосредственно к выводам микросхем ПЗУ. В качестве примера для SOS и ТЕСТ использованы микросхемы К556РТ18, а для TRDOS — К573РФ5, имеющие преимущество в надежности по сравнению с К573РФ6, так как они допускают в два раза больше циклов записи информации и в обеспечивают в два раза большее время хранения информации. При ремонте компьютера, в одну К573РФ5 проще записать информацию, чем в одну К573РФ6. В компьютере и контроллере дисковода, использующих по две микросхемы К573РФ6, первая включается при нулевом уровне шины адреса A13 процессора, а вторая — при единичном. Так и в приведенной схеме — нулевой уровень A13 включает дешифратор C микросхемы DD1, соединенный с микросхемами N1...N4, а единичный уровень A13 включает дешифратор D микросхемы DD1, соединенный с микросхемами N5...N8. В компьютере микросхемы ПЗУ выбираются при наличии сигнала RD ROM (чтение ПЗУ) и отсутствии сигнала -ROM CS (выбор ПЗУ контроллера дисковода), так и в приведенной схеме эти сигналы заведены на входы 18 и 19 выбора кристалла микросхем DD3...DD11. При отсутствии контроллера дисковода, вместо сигнала -ROM CS нужно подать сигнал +5 В. Микросхемы ПЗУ контроллера дисковода выбираются сигналами MREQ (чтение памяти) и ROM CS. И в приведенной схеме эти сигналы заведены на входы элемента "И" DD2.2, выход которого подключен к выводам 20 выбора кристалла микросхем DD12...DD17. Также и для микросхем SOS DD3...DD11 можно использовать элемент "И" с заведенными на его входы сигналами RD ROM и инверсный ROM CS, если в используемых ИМС всего два входа выбора кристалла, один из которых задействован дешифратором, то есть когда используются микросхемы К573РФ5.

С помощью переключателя SA1 можно отключить от дешифратора ИМС N1 и вместо них подключить тестовую микросхему. При этом будут проверены контрольные суммы информации областей SOS N2...N8. Для TRDOS Ver 5.04s в области N8 оказались чистыми адреса от 000H до 400H, то есть те, которые заняты в области N7 (от 000H до 1FDH). Поэтому, если области TRDOS N7, N8 записать в одну микросхему, то никакой накладки информации произойти не может, так как области займут разные адреса этой микросхемы. При этом сигналы с дешифратора DD1, выбирающие области N7, N8, нужно подать на вход выбора кристалла объединенной микросхемы DD12 через элемент ИЛИ, выполненный на диодах VD4, VD5. То есть объединенная микросхема DD12 выбирается, когда процессор пользуется областями N7 и N8. Если выбирается область N8, то процессор "не зайдет" на адреса объединенной микросхемы, занятые областью N7, так как в области N8 эти адреса чистые. Также при выборе области N7 процессор "не заходит" на адреса объединенной микросхемы, занятые областью N8. Область N2 — почти вся чистая, кроме 5 байтов вначале, но эти адреса заняты и в объединенной микросхеме DD12. Если

при выборе процессором области N2 проинвертировать адресный сигнал A9 микросхемы DD12, то эту область можно занести в DD12 по адресам начиная с 200H, которые в ней свободны. Подачу инверсного A9 осуществляет элемент D2.1 через развязывающий диод VD2 только во время выбора процессором области N2. При выборе областей TRDOS N7, N8, входящих в объединенную микросхему DD12, на вход A9 подается нормальный адресный сигнал через диод VD1. Можно занести область N2 в микросхему DD12 и по другим адресам, проинвертировав для этого другие адресные шины — одну или несколько. Можно не экономить микросхемы, и каждую область TRDOS N1...N8 занести в отдельную ИМС, тогда надобности в дополнительном дешифраторе не будет, и все 8 микросхем можно включить аналогично DD3...DD10. Подобную экономию микросхем можно осуществить в любых компьютерах, где есть свободные места. Сначала нужно записать на бумаге адреса свободных мест, и сразу будет видно, как скомпоновать информацию в одной микросхеме и как для этого составить дешифратор по описанному примеру. Увеличение количества микросхем ПЗУ практически не усложняет конструкцию компьютера, потому что нет необходимости использовать дополнительную печатную плату для установки микросхем ПЗУ и дешифратора. ИМС DD1, DD2 можно закрепить над какой-либо микросхемой компьютера геометрически параллельно ей, использовав для этого два коротких отрезка толстого медного провода, припаянных к выводам питания нижней и верхней микросхем. На них и держатся дополнительные микросхемы. Вместо каждой микросхемы К573РФ6 можно использовать стопку из 4 ИМС емкостью 2 Кб. Включают такую стопку в панельку микросхемы К573РФ6. Корпуса микросхем в стопке, как полки на этажерке, расположены один над другим с миллиметровым зазором для охлаждения. Их одноименные выводы ложатся один на другой, их и спаивают вместе в один провод, проходящий через все корпуса стопки микросхем и выходящий за нижний корпус для включения его в панельку. Так должны быть запаяны все выводы (одноименные выводы — в отдельный провод), кроме выводов, подключаемых монтажным проводом с помощью пайки к дешифратору DD1. Выводы микросхем, выходящие из стопки, которые согласно схеме и таблице не должны войти в гнездо панельки, откусывают, а гнездо панельки заклеивают изолентой, чтобы не произошло случайного контакта. Эти объединенные и укороченные провода, составленные из выводов микросхем, соединяются с печатной платой компьютера с помощью монтажных проводов и пайки. Можно спаять все микросхемы ПЗУ в одну стопку, если используется всего одна панелька. Такой столбик можно положить, чтобы занимал меньше места, закрепив его толстыми медными проводами (шинвми питания) над микросхемами компьютера и соединив монтажными проводами со схемой компьютера все объединенные выводы микросхем (всего каких-то два десятка дополнительных проводов). Сгоревшую микросхему будет легко заменить, так как имеется возможность отпаивать и припаивать по одному выводу. В таблице (в средней ее части) приведены номера и наименования (фунции) выводов заменяемой микросхемы К573РФ6 как на самой микросхеме, то есть по возрастанию номера в левой колонке — сверху вниз, а в правой — снизу вверх (как будто бы микросхема РФ6 лежит посередине таблицы). Соответствующие выводы заменяющих микросхем приведены в колонках таблицы: слева — для левой стороны микросхемы и справа — для правой. Из таблицы видно, что стопка из четырех К573РФ5 полностью устанавливается на месте одной К573РФ6, только ее нужно сместить на два вывода назад так, чтобы первый вывод микросхемы вошел в третье отверстие панельки. При этом только выводы 21 и 24 микросхемы К573РФ5, на которые должны быть подано напряжение питания +5 В, не должны попасть в расположенные возле них отверстия панельки, а проводом их нужно соединить с гнездом 28 панельки — точно так, как это делают при включении тестовой микросхемы во время наладки компьютера. Не должны попасть в гнезда панельки и выводы 18, которые припаянными монтажными проводами соединяются с дешифратором. Также должны быть смещены на два вывода назад и другие микросхемы, а стопка микросхем К1623РТ1, кроме того, должна быть установлена в панельку "задом наперед", то есть ее первый вывод должен совпасть с гнездом 15 панельки. Если в микросхеме не совпадает с панелькой расположение адресного вывода A10, то его нужно откусить, чтобы он не вошел в расположенное напротив него гнездо панельки, и проводом с помощью пайки соединить с точкой монтажа компьютера, где присутствует сигнал A10, так как с соответствующим гнездом панельки этот провод соединять неудобно из-за расположенных над ним других выводов стопки микросхем. В гнездо 21 панельки, на которое подан сигнал A10, не должны попасть выводы микросхем серий К556, К1623, на которые подается сигнал выбора микросхемы — CS. Инверсные входы CS должны быть подключены к дешифратору DD1 или к элементу DD2.2, а на прямые — должны подаваться сигналы (RD ROM и -ROM CS) или (MREQ и ROM CS). Вывод 6 микросхемы К1623 должен быть соединен с общим проводом, так как это вывод, на который подается напряжение во время программирования. Выводы 24 микросхем серии К556 и вывод 12 микросхемы серии К1623 нужно соединить проводом с шиной питания +5 В.

Припаянные провода никаких неудобств для установки и извлечения микросхем не создают, их можно легко припаивать и отпаивать когда это нужно, или вынимать микросхемы с панельки не отпаивая провода. "Закреплять" информацию (то есть записывать в микросхемы то, что в них уже записано) можно не распаивая стопку микросхем РФ5. Ее всю подключают к программатору, только на выводы 21 непрограммируемых микросхем подают напряжение +5 В. Вывод 21 программируемой микросхемы должен быть подключен к программатору.

Для чтения информации программатором нужно закрыть выходы всех микросхем стопки, кроме читаемой, подачей на их выводы 18 или 20 логической "1", а выводы 18 и 20 читаемой микросхемы должны быть подключены к программатору. Выводы, на которые подается напряжение +5 В или логическая "1", должны быть отключены от программатора.

В таблице выводы выбора микросхемы и разрешения выходов обозначены одинаково — как выбор кристалла CS, так как для данного использования они идентичны и одинаково воздействуют на выходы микросхем. Микросхемы К556РТ7 И К556РТ7А читаются одинаково, однако алгоритм их программирования разный.

Обозначения в таблице:

- A — адреса. Например, 23 A8 — вывод 23, 8-й разряд адреса;
- D — данные. Например, 9D0 — вывод 9, нулевой разряд данных;
- CS — выбор микросхемы;
- EPR — вывод подачи напряжения программирования;
- 5 В — напряжение питания;
- / — знак инверсии.

Вначале указан номер вывода, а затем — его функция.

Номер выводы и его функция									
Левая сторона микросхем					Правая сторона микросхем				
К1623 РТ1	К556 РТ18, РТ7А	К556 РТ7	К573 РФ5	К573 РФ6	К573 РФ6	К573 РФ5	К556 РТ7	К556 РТ18, РТ7А	К1623 РТ1
				1 EPR	28 5B				
				2 A13	27 CS				
13 A7	1 A7	1 A7	1 A7	3 A7	26	24 5B	24 5B	24 5B	12 5B
14 A6	2 A6	2 A6	2 A6	4 A6	25 A8	23 A8	23 A8	23 A8	11 A8
15 A5	3 A5	3 A5	3 A5	5 A5	24 A9	22 A9	22 A9	22 A9	10 A9
16 A4	4 A4	4 A4	4 A4	6 A4	23 A11	21 EPR	21 A10	21 A10	9 A10
17 A3	5 A3	5 A3	5 A3	7 A3	22 /CS	20 /CS	20 /CS	20 /CS	8 CS
18 A2	6 A2	6 A2	6 A2	8 A2	21 A10	19 A10	19 CS	19 CS	7 /CS
19 A1	7 A1	7 A1	7 A1	9 A1	20 /CS	18 /CS	18 CS	18 CS	6 EPR
20 A0	8 A0	8 A0	8 A0	10 A0	19 D7	17 D7	17 D7	17 D7	5 D7
21 D0	9 D0	9 D0	9 D0	11 D0	18 D6	16 D6	16 D6	16 D6	4 D6
22 D1	10 D1	10 D1	10 D1	12 D1	17 D5	15 D5	15 D5	15 D5	3 D5
23 D2	11 D2	11 D2	11 D2	13 D2	16 D4	14 D4	14 D4	14 D4	2 D4
24 0B	12 0B	12 0B	12 0B	14 0B	15 D3	13 D3	13 D3	13 D3	1 D3

Ю.ПАКИН,
211030, Витебская обл.,
г.Орша, ул. Мира, 52 — 2.

Расширение памяти ПК "Квант 48К"

Данный вариант расширения памяти — это не HIGH END, а скорее, попытка помочь рядовому пользователю реанимировать или заапгрейдить свой "КВАНТ". Плата расширения памяти до 128 Кб выпускалась в г.Минске ориентировочно в 93...94 г.г. Электрической схемы на данную плату нет. Поэтому схема была восстановлена по оригинальной плате расширения, и после многократного контроля связей решено было рассказать об этом пользователям, чтобы и они могли самостоятельно вдохнуть в свой Spectrum вторую жизнь. Данный тип расширения применим для компьютеров с общим полем памяти, и может быть использован для других однотипных машин.

Ввиду того что оригинальная схема грешит ошибками, вам необходимо будет сделать небольшие доработки. Нумерация микросхем и контактов для подключения платы расширения дана условно — из-за отсутствия авторской схемы, и необходима лишь для того, чтобы хоть как-то ориентироваться.

Расширение памяти следует начать с установки восьми микросхем 565РУ5Г. Наиболее простой способ — это напайка второго этажа над первой линейкой "нога к ноге", кроме выводов 15, которые следует объединить в отдельную линию. Лучше будет, если РУ5 будут напаиваться не внакладку, а только торцами выводов, чтобы обеспечить максимальный зазор между верхней и нижней микросхемами. При ограниченном объеме корпуса компьютера черезмерный нагрев микросхем — бич подобного типа расширения. Мне же лично больше "греет душу" другой вариант, конечно, если позволяет объем корпуса. Если у вас "завалялась" плата, неизвестно чего, с линейкой "РУшек", выпилите линейку и установитв на кронштейнах таким образом, чтобы обеспечить наиболее благоприятный терморежим. Учитывайте также то обстоятельство, что в будущем там будет находиться периферия. На выводы +5 В и GND каждой РУ5 следует "посадить" конденсатор емкостью порядка 0,1 мкФ для устойчивой работы ОЗУ. Если РУ5 установлены "вторым этажом", конденсаторы напаиваются на каждую пару микросхем ОЗУ. В принципе, у меня на напаяных "вторым этажом" РУ5 стоят конденсаторы емкостью 47 нФ — и все работает.

Теперь возьмемся за плату расширения. Принципиальная схема доработанной платы приведена на рис.1, внешний вид — на рис.2. Внесенные в схему изменения показаны "жирными" линиями. В таблице приведена разводка контактов платы.

Рис. 1

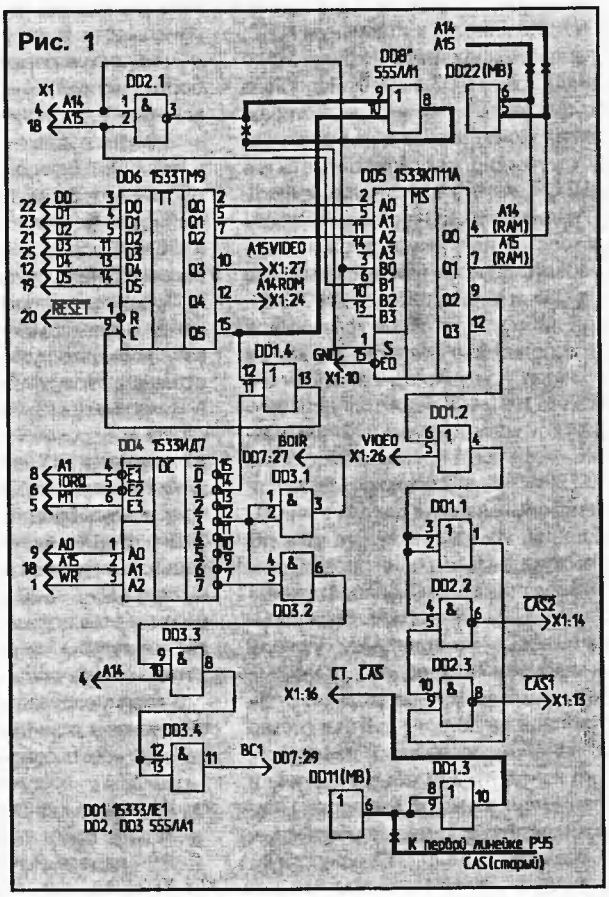
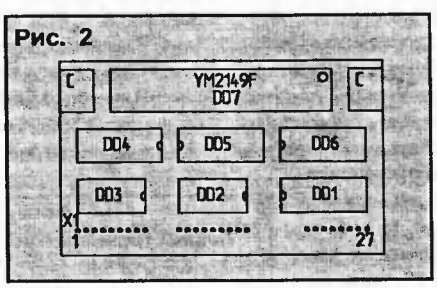


Рис. 2



Для начала на плате расширения в микросхеме (далее м/с) DD2 следует освободить вывод 3 или, в крайнем случае, перерезать дорожку, идущую от вывода 3 DD2 к выводу 1 DD5 платы расширения. На материнскую плату установите м/с ЛЛ1 (обозначена на схеме как DD8') и соедините один из ее элементов согласно схеме (рис.1): один вход — к выводу 3 DD2 платы расширения (далее ПР), другой вход — к выводу 15 DD6 ПР, выход соединяем с выводом 1 DD5 ПР. Можно пробуферировать и сигнал CAS, который раньше шел на выводы 15 линейки РУ5, расположенных на материнской плате. Для этого контакт 16 разъема X1 соединяем с выводом 10 свободного элемента DD1 ПР.

Теперь разберемся с сигналом CAS. Этот сигнал, ранее шедший на выводы 15 первой линейки ОЗУ, следует "оторвать". Можно, конечно, и перерезать дорожку, идущую от вывода 6 DD11 (ЛА3) к выводам 15 м/с РУ5, но лучше вывод 6 DD11 выпаять.

Теперь посмотрите. Если вы пробуферировали "старый" CAS (установили перемычку от контакта 16 разъема X1 на вывод 10 DD1), то провод от вывода 6 м/с DD11 платы "КВАНТ" должен идти на выводы 8 и 9 м/с DD1 платы расширения, если не буферировали — на контакт 16 разъема X1 вашей платы. Далее, контакт 13 X1 ПР соединяется выводами 15 первой (нижней) линейки РУ5, а контакт 14 X1 ПР — с выводами 15 верхней линейки. Сигнал VIDEO (X1, контакт 26 ПР) следует "взять" с вывода 2 DD19. Далее выпаиваем выводы 5 и 6 м/с DD22 "КВАНТ" (туда раньше шли сигналы A14 и A15 соответственно). "Набросим" перемычку с вывода 4 м/с DD5 ПР на вывод 5 DD22 "КВАНТ" — сигнал A15(RAM). Чтобы завести сигнал A15VIDEO, идущий с контакта 27 разъема X1 (вывод 10 DD6 ПР), нужно выпаять вывод 4 м/с DD12 "КВАНТа" и завести на него этот сигнал. Сигнал A14ROM с контакта 24 разъема X1 ПР должен идти на вывод 27 ROM м/с 27256 или 27512. Этот же сигнал должен идти на вход контроллера FDD для блокировки выхода в TR-DOS. Откуда взять остальные сигналы, показано на левой части схемы (рис.1), например, с процессора или буферов (входов м/с DD2.1, DD3.3, DD4 и DD6 платы расширения). Для грамотного буферирования внимательно ознакомьтесь с разделом HARD в электронном журнале Spectrum Expert (№№1, 2). Вообще-то, плата расширения памяти считается устройством внутренним, в отличие от, например, контроллера FDD, AY-"карты" — устройств внешних, периферийных, поэтому можно в данном случае обойтись и без буферов.

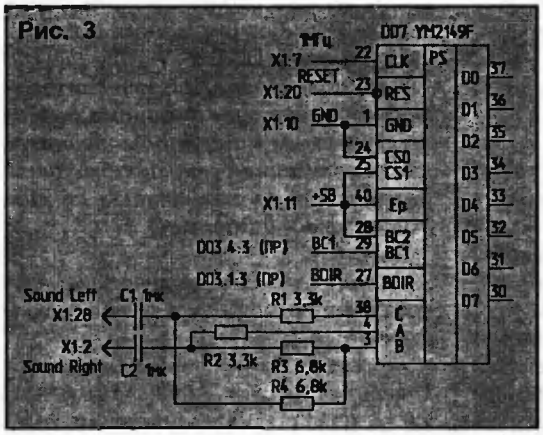
Теперь остались сигналы музыкального сопроцессора — BDIR и BC. Честно говоря, просматривать его связи на заводской плате мне было лень. К тому же, AY-чип был впаян намертво, вместо того чтобы быть вставленным в хорошую панельку.

Контакт	Идентификатор сигнала	Источник/приемник сигнала
1	WR	Вывод 22 CPU
2	AY-SOUND RIGHT	
3	SOFE Sound Out from port #FE	Точка соединения резистора R35 и транзистора VT1 (дополнительный выход на BEEPER)
4	A14	Вывод 4 CPU
5	M1	Вывод 27 CPU
6	IORQ	Вывод 20 CPU
7	AY-CLK	Вывод 24 ВГ93 (1 МГц) или вывод 2 DD19 (1,75 МГц)
8	A1	Вывод 31 CPU
9	A0	Вывод 30 CPU
10	GND	
11	+5B	
12	D4	Вывод 7 CPU
13	-CAS1	Выводы 15 первой линейки РУ5
14	-CAS2	Выводы 15 второй линейки РУ5
15	D7	Вывод 13 CPU
16	-CAS	Старый CAS или сигнал с выхода свободного элемента DD1 на плате расширения, используемого в качестве буфера (см. схему)
17	D6	Вывод 10 CPU
18	A15	Вывод 5 CPU
19	D5	Вывод 9 CPU
20	-RESET	Вывод 26 CPU
21	D2	Вывод 12 CPU
22	D0	Вывод 14 CPU
23	D1	Вывод 15 CPU
24	A14ROM (или ROMCS)	Вывод 27 м/с 27256 или 27512 (DD27 материнской платы)
25	D3	Вывод 8 CPU
26	VIDEO	Вывод 2 DD22 материнской платы (переключатель мультиплексоров ОЗУ на ввод информации с адресной шины либо на опрос экранной области)
27	A15VIDEO	Вывод 4 DD22 (переключатель экранов)
28	AY-SOUND LEFT	

Но так как схемы подключения AY — все типовые, я "стащил"схему включения (рис.3) у ZEG'a из его статьи [8].

Думаю, в левой части схемы все понятно. У меня на плате также стоят конденсаторы (C1, C2), поэтому я их включил в схему. Сигнал от точки соединения резисторов R3, R4 у меня подан на контакт 3 разъема X1 (см. таблицу) — это сигнал смикшированного звука, который можно завести в ПК на BEEPER (на базу транзистора VT1). Сигналы данных на выходе DD7 подключаются к одноименным сигналам процессора. Правильно подключенная плата, кроме всего прочего, может проигрывать музыку Digital Player.

Рис. 3



Вообще-то, я бы рекомендовал при доработках по возможности не перерезать дорожки на платах, а выпаивать нужные выводы микросхем — это поможет ремонтникам без излишних усилий разобраться в вашем компьютере и повысит его ремонтопригодность.

Впрочем, недостатки описанного варианта учтены в аналогичной плате расширения BC1, которая подключается так же. Следует учесть, что расположение и нумерация контактов платы BC1 отличаются от этого варианта. При покупке авторской платы BC1 к ней прилагается описание по подключению, в котором вы можете ознакомится со всеми подробностями.

Со всеми вопросами, связанными с приобретением платы BC1, обращайтесь по телефону в Минске (017) 250-23-06.

Стоит еще добавить, что сейчас вы занимаетесь не чем иным как апгрейдом. Поэтому постарайтесь приобрести блок питания от IBM PC — этим вы впоследствии избавите себя от множества хлопот. Помните, что блок питания чересчур мощным не бывает. Поинтересуйтесь, также, имеются ли в блоке напряжения -5 В и 12 В. Зачем? А вот об этом, я думаю, еще не раз нам поведают авторы статей.

Мне остается сказать, что представленный материал подготовлен на компьютере "КВАНТ 128К" в текстовом редакторе IS-EDIT v4.9 при поддержки операционной системы IS-DOS.

Немного критики

Данная компоновка обладает, на мой взгляд, одним главным недостатком. Я до сих пор не могу понять, что делают на одной плате два совершенно разных устройства? Еще лет 5...6 назад штатным "звуковым" устройством был Beeper, и саунд музыки в Chronos или Savage был верхом кодинга. Сейчас штатное устройство — AY. А ведь есть еще и Covox, и Soundrive, и DMA. Интересно, если таким образом апгрейдить Spectrum, количество звуковых карт в нем в чем будет измеряться? Думается, самое время подумать об универсальной звуковой карте. Или уж хотя бы о программировании с обращением по полному адресу портов и о создании коммутатора портов звуковых карт.

Принимается любая деловая критика, по принципу: отвергая — предлагай.

Литература

1. Схема ПК "КВАНТ" Оршанского завода ПАК.

2. Расширение памяти персонального компьютера "SPECTRUM". — ZX-РЕВЮ, 1991, Вып. 1...12, С.15...18.

3. Ю.Дудник. ZX-SPECTRUM 128К — что это такое? — Радиолюбитвль, 1991, №11, С.8; 1992, N5, С.8.

4. А.Девликамов. Расширение компьютера "ЛЕНИНГРАД-2" до 128К. — Радиолюбитель. Ваш компьютер, 1996, № 7, С.19.

5. С.Рюмик. "СПЕКТРУМ-128К". — Радиолюбитель, 1994, №2, С.8; №3, С.10.

6. В. Зелинский. Переделка львовского варианта "ZX-SPECTRUM-48" в "ZX-SPECTRUM-128". — Радиолюбитель. Ваш компьютер, 1997, №5, С.20.

7. С.Бандура. Расширение памяти компьютера "ЛЕНИНГРАД-2" на микросхемах К565РУ7. —Радиолюбитель. Ваш компьютер, 1997, №11, С.24.

8. Е.Зарецкий. Расширение ПК "Байт" до варианта "ZX-Spectrum-128". — Радиолюбитель. Ваш компьютер, 2000, №3, С.39.

9. Kvazar&Elssland. Музыкальный процессор AY-3-8910/8912. — Радиолюбитель. Ваш компьютер, 2000, №7, С.42; №8, С.35.

10. С.Рюмик. "Правильные" и "неправильные" CLK музыкального сопроцессора. — Радиолюбитель. Ваш компьютер, 2000, №11, С.35.

11. "BC1". Плата расширения ОЗУ до 128Kb с музыкальным процессором 8910 (8912). Руководство.

12. МУЗЫКАЛЬНЫЙ ПРОЦЕССОР AY-8912/8910. — Системные программы для ZX-SPECTRUM 128K. Справочник. М.: VA-PRINT, 1993, С.228...241.

13. ZX-FORUM-1. — Сборник статей, С. 146...147.

По страницам электронных изданий

Ремонт Scorpion'a своими руками

(c) Сергей Зеров,
(c) ZX-news.

Занимаясь ремонтом компьютеров в фирме Scorpion в качестве главного настройщика уже более двух лет, я пришел к выводу, что 70...80% неисправностей опытный пользователь может устранить сам. Особенно это касается отдаленных областей бывшего Советского Союза. Была на моей памяти плата, которая проехала от Сахалина до Санкт-Петербурга и обратно из-за одного транзистора КТ361. Пользователя жалко!

А теперь несколько комментариев.

Статья предназначена для пользователей, имеющих опыт работы с паяльником и осциллографом.

Автор не несет ответственности за неквалифицированный ремонт.

Вопросы по некачественной работе "левых" плат не обсуждаются.

Экстрасенсорными способностями я не обладаю, поэтому по телефону и по фотографии не ремонтирую. Хотя некоторые советы дать могу.

Компьютеры Scorpion (модели плат до версии SC13) собраны на микросхемах серии 555. Все остальные версии плат, включая TURBO+, собраны исключительно на микросхемах серии 1533.

Исключение составляют выходные буфера клавиатуры (155ЛР9), принтера (555ИР23) и дисковода (155ЛА13 и ЛН5). В моделях TURBO и TURBO+ применена м/с 531ТМ2 во входном делителе 14/7 МГц, а также специализированная м/с ALTERA, представляющая собой аналог ПЛМ.

Заменять м/с следует с учетом серии. В противном случае плата может работать со сбоями (особенно это касается плат TURBO и TURBO+).

Особо нужно поговорить о ПЗУ. В платах TURBO+ используются микросхемы ППЗУ 27с512 с версией теневого монитора 2.95 и ПРОФ ПЗУ версии 3.2 и выше.

В платах TURBO используются микросхемы ППЗУ 27с512, некоторые экземпляры 27512 и ПРОФ ПЗУ всех версий.

В обычных платах применяются любые версии обычных ПЗУ (с учетом всех доработок) и ПРОФ ПЗУ версии 3.40.

По статистике причины неисправностей делятся в следующей пропорции:
- 40% — блоки питания;
- 30% — статика (особенно зимой);
- 15% — ошибки подключения;
- 15% — разное (залили пивом и т.п.).

Для начала, думаю, хватит.

Далее рассмотрим характерные неисправности, выявленные в процессе настройки и ремонта плат "Scorpion ZS256 TURBO+" выпуска 1996 г. Все ссылки даны по принципиальной схеме.

Ошибки в номиналах

Конденсатор С17. По схеме — 150 пф. Возможна установка 120...200 пф.

В ряде плат стоит 270 пф. Вследствие этого могут не работать профессиональные ПЗУ (ПРОФ ПЗУ), особенно при установке дополнительных контроллеров (IBM-клавиатуры и HDD). При этом обычные ПЗУ работают. Аналогичным образом влияет резистор R58 (его номинал должен быть 560 Ом).

На плате применен преобразователь 5 В/12 В для питания контроллера ВГ93. Номинал дросселя указан 10...20 мкГ. Должно быть — 20...30 мкГ. При индуктивности менее 15 мкГ возможно уменьшение выходного напряжения преобразователя до 10 В и, как следствие, появление ошибок при записи на диск (порча диска). Иногда в этом виноват VT7 КТ315.

Конденсаторы С2, С3. По схеме — 1 нФ. Из-за ошибок в номиналах или дефектов в самих деталях, плата может самопроизвольно переключать режим Turbo/Normal.

Резистор R33. По схеме — 1,5 кОм. Некоторые типы процессоров могут потребовать уменьшения этого сопротивления до 560 Ом (увеличение амплитуды тактового сигнала).

Конденсатор С9. Номинал по схеме — 33 мкф. При уменьшении номинала (за счет высыхания) может не проходить начальный сброс по включению питания.

Типовая неисправность

Принимая во внимание статистику по настройке и ремонту, на первое место могу поставить входной делитель, выполненный на ИМС 531ТМ2. Наблюдаются три типа неисправностей:

1. Нет выходного сигнала 7 МГц. Плата "молчит, как бревно", 531ТМ2 "греется, как утюг".

2. Плата работает, но на начальное меню выходит медленно, постепенно прорисовывая линии и рамки.

3. Плата работает, на меню выходит нормально, но не реагирует на клавиатуру из-за отсутствия сигнала INT-.

Подобная неисправность происходит по причине сильного перегрева ИМС при неисправности блока питания. При этом 531ТМ2 используется как дополнительная защита (обычно сгорая вместе со стабилитроном КС156).

Дисковод

Теперь расскажу вам о ремонте дисководной части "Скорпиона". Нумерация элементов приведена по старой схеме (не ТУРБО). Неисправность проявляется в том, что дисковод не читает или не форматирует.

1. ***Светодиод на дисководе (дисководах) светится постоянно. Вставляемые дискеты портятся.***

Действия:
- проверить ВГ93;
- проверить схему защиты по +12 В (КТ361, КТ315, КС139). Неисправность схемы защиты может быть причиной регулярного выхода из строя ВГ93;
- проверить наличие на ВГ93 всех напряжений;
- проверить блок питания по +12 В (должны отсутствовать броски напряжений в моменты включения/выключения).

2. ***ВГ93 исправна, напряжения в норме.***

Действия:
- проверить дисковод;
- проверить кабель между платой, блоком питания и дисководом;
- проверить наличие питающего напряжения (4,4...4,8 В) на выводе 14 ИМС ФАПЧ D63 (155РТ11) при обращении к дисководу. Если напряжения нет или оно меньше 3,5 В — заменить КТ361;
- проверить наличие сигналов управления на ВГ93 (D54, D65) и следующие за ним схемы;
- проверить схему чтения (D68.3, D67) и ФАПЧ (D62, D63);
- иногда вылетает мультиплексор D63 (1533КП11) по сигналам DRQ или INTRQ. При этом тестирование в теневом мониторе происходит с быстрой выдачей ошибок (исправных секторов нет). Заменить D63.

3. ***ВГ93 исправна. В процессе форматирования или проверки дискеты теневым монитором возникают систематическив ошибки (один или несколько одинаковых секторов на каждой дорожке).***

Действия:
- неисправен сам дисковод (частота вращения привода диска значительно отличается от 300 об/мин);
- проверить схему формирования тактовых частот для ВГ93, ТМ9 (D63), ИР16 (D61). Она собрана на ИМС

1533ЛН1, 1533ИЕ10 и 1533ЛП5 (D1, D56 и D11 соответственно).

4. ***ВГ93 исправна. Дисковод не форматирует вообще.***

Действия:

- проверить схему формирования сигнала записи (D10, D61, D63), проверить наличие тактовых импульсов на выводе 9 D61;

- если плата перестает форматировать после прогрева или при пониженном питании (особенно это касается плат ТУРБО), то рекомендую установить конденсатор на вывод 6 D61 (сигнал WD).

Как уберечься от неприятностей при ремонте

1. Аксиома — на жале любого включенного в сеть паяльника может появиться 220 В.

Последствия — средний ремонт, переходящий в капитальный.

Способы устранения:

- используйте низковольтный паяльник;

- заземляйте жало паяльника;

- при любых работах отключите компьютер не только от сети (выньте вилку из розетки), но и от всех периферийных устройств (TV, принтер, магнитофон и т.п.).

2. Аксиома — "не ковыряй" включенные блоки питания.

Последствия — печальные. Вы можете потерять не только плату, но и часть периферии вместе с "куском" здоровья.

Способы устранения:

- используйте качественные блоки питания;

- не "ковыряйте" блоки питания.

3. Аксиома — не все, что греется — неисправно.

Последствия — лишняя работа.

На плате могут довольно сильно нагреваться следующие элементы:

- ОЗУ, особенно на платах TURBO+ — импортные аналоги РУ7;

- ИМС серии 531— DD2 и в некоторых старых платах — DD63;

- 556РТ11 (в старых платах — РТ4).

Некоторые микросхемы могут греться по причине выхода из строя элементов, на которые они нагружены (пробой по входу). Но это можно выяснить только с помощью АККУРАТНОГО перерезания дорожек.

4. Аксиома — длительность сигнала INT составляет 9 мкс.

Последствия — "глюки" при работе с ПРОФ ПЗУ и с некоторыми программами.

Способы устранения:

- на старой плате — подбор RC-цепочки;

- на плате TURBO+ — проверить схему формирования.

Подготовил Е.Зарецкий (Zeg/Fenomen/Phycho), 213200, Могилевская обл., г. Чаусы, ул Фрунзе, 28.

ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ

("РЛ.ВК", №12/2000, С.36)

Расширения файлов TR-DOS

Расширение	Комментарий
#?	Вторая часть 255-секторного файла, который был преобразован в формат hobeta с помощью программы "Godzilla". Второй символ расширения показывает, каким было расширение исходного файла
b	Командный файл с программой для оверлея batproc к EMS
bin	Двоичный файл. В нем может быть, например, содержимое ПЗУ
Ckt	Таким становится расширение файла — почтового пакета (pkt) после его извлечения из архива с помощью старых версий программы pkunzip. Будьте внимательны — не каждая программа, поддерживающая трехсимвольные расширения файлов, правильно отобразит это расширение на экране. Так как в нем сочетаются прописные и строчные буквы, оно может быть сочтено недопустимым, и тогда вы увидите лишь первый его символ — "C".
D	Двухэкранная картинка
gif	Графический файл. Программа для обработки — GIF screen's viewer by DIS/XPJ. Насколько мне известно, эта программа доступна лишь в исходных текстах (в формате ассемблера ALASM), так что перед запуском придется вам ее откомпилировать
mo0, tu0, we0, th0, fr0, sb0, su0	Архивы с почтовыми пакетами, сформированные соответственно в понедельник, вторник, среду, черверг, пятницу, субботу и воскресенье
MPP	Список проигрываемых музыкальных модулей (playlist) для Mm<M Player
nfo	Текстовый файл с кратким описанием содержимого архива или подкаталога, в котором он расположен
O	Оверлей к EMS
pcx	Графический файл. Еще две программы для обработки — Brujeria 1.0 by ALFF/RD/CPU и PCX viewer 2.04 XM by S.F.D. (для работы со второй программой измените расширение файла с "pcx" на "X")
R	Текст в формате ассемблера STORM 1.2d
vi1	Дамп микросхемы CMOS. Создается программой "CMOS Commander"
WRD	Текстовый файл
X	Сжатый pcx-файл. Для просмотра используйте PCX viewer 2.04 XM by S.F.D.
Zxz	Архив ZXZIP (в MS-DOS и IS-DOS эти архивы часто хранятся именно с таким расширением; соответственно, перед раскрытием архива его расширение надо изменить на "ZIP").

Со времени опубликования статьи мне стали доступны дополнительные сведения о работе с различными файлами в среде TR-DOS.

Часть информации мне предоставили Dmitriy Nesmachny и Eugene Palenock. Благодарю их за помощь!

BV_CREATOR,
г.Москва,
Fido: 2:5020/689.53
ZXNet: 500:95/462.53
E-mail: asder_ffc@softhome.net
WWW: http://www.ivr.da.ru

КОМПЬЮТЕРНЫЕ ХРОНИКИ

ASCii DEMOPARTY

31 марта в Ижевске прошло спектрумовское demoparty под названием ASCii. Состоялись конкурсы музыки (copy/nocopy), графики, демо. Как можно судить по краткому пресс-релизу, распространявшемуся в компьютерных сетях вместе со сборниками работ, party прошло весьма успешно, даже лучше, чем ожидали организаторы.

Ниже приведены результаты конкурсов.

Copy music compo. 1 — "Zemfira/Iskala" (Eastman^X-Team); 2 — "Ruki-3" (VAD^ULG); 3 — "The Tale" (Ded Smimoff^PoS).

Nocopy music compo. 1 — "Valley" (VAD^ULG); 2 — "Paradoxal" (Dr.'S'^CIC); 3 — "Illusion" (Monohrom^TRG).

Gfx compo. 1 — "Lady" (Alex^X-Team); 2 — "Dinoz" (Psychotron); 3 — "Fuck Man" (Slider).

Demo compo. 1 — "Quarantine" (Jumble); 2 — "Fucking Life" (Brutal Creators); 3 — "Mad Pixels" (Power of Sound).

Более подробную информацию ищите на официальной странице demoparty, размещенной на сервере Power Of Sound Web Team (http://poswt.da.ru).

MILLENNIUM DEMOPARTY

5-6 мая в Минске прошло мультиплатформенное demoparty Millennium. Вот результаты конкурсов на платформе ZX Spectrum.

AY-музыка. 1 — "Karkusha in death fly" (Freeman/Freeart); 2. — "SoundStorm" (SERGant/FBN), это лучшая, на мой взгляд, мелодия; 3. — "Israel is my sweet home" (Mast/Freeart). В music compo также принимали участие и такие известные музыканты, как EA/Antares, Mm<M/Freeart/Sage.

Графика. 1 — "Magic Ball" (Monarch); 2 — "Avenger" (Monarch); 3 — "Dragon-2" (Fantom/Siberian Group).

Кстати, работ в номинациях "музыка" и "графика" в этом году оказалось больше, чем в прошлом.

Демо. 1 — "GC Return" (Phantom Digger, ZS); 2 — "Action of Savers" (Savers Alliance).

Игры. 1 — "3D-Roost" (Fenomen^Psycho) — логическая игрушка; 2 — "Tech Me" (PHG) — версия игры "Технодром", написанная специально для Millennium.

В номинациях ***512 bytes intro*** и ***16K intro*** было представлено всего по одной работе — "Blobs" (Imperio/PHG) и "16K Intro" (K.C.Soft). Так что, эти конкурсы не состоялись.

Работы с party можно скачать с сайтов http://scenergy.natm.ru, http://zx.da.ru и http://pos.izhnet.ru.

Хроники вел И.Рощин

ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ
("РЛ.ВК", №7/2000, С.40)

Вывод трехсимвольных расширений файлов в операционной системе TR-DOS

Как выяснилось, одно из требований к трехсимвольному расширению (ранее сформулированное так: все три символа — коды в диапазоне #20...#7F) нуждается в некотором уточнении.

Во-первых, символ с кодом 96 (обратная кавычка) не является допустимым и, следовательно, должен быть исключен из данного диапазона.

Во-вторых (об этом мне сообщил Max Vasilyev), трехсимвольное расширение не должно начинаться с пробела (например, "ab") или содержать вторым символом пробел ("a b"). Иначе говоря, пробелом может быть лишь последний символ расширения.

Соответственно, процедура определения длины расширения файла теперь должна выглядеть так:

```
ANALYS_EXT      PUSH    HL,DE,BC

                LD      BC,3*256+%00100000

;Проверка на BASIC:

                LD      A,(HL)
                CP      "B"
                JR      Z,ANALYS_NO

;Проверка: первый или второй символ - пробел?

                INC     HL
                LD      A,(HL)
                CP      C                       ;=CP #20
                JR      Z,ANALYS_NO

                DEC     HL
                LD      A,(HL)
                CP      C                       ;=CP #20
                JR      Z,ANALYS_NO

;Проверка на XAS, xAS, XaS, xaS:

                RES     5,A
                CP      "X"
                JR      NZ,ANALYS_EX1

                INC     HL
                LD      A,(HL)
                RES     5,A
                CP      "A"
                JR      NZ,ANALYS_EX2

                INC     HL
                LD      A,(HL)
                CP      "S"
                JR      Z,ANALYS_YES

                DEC     HL
ANALYS_EX2      DEC     HL

ANALYS_EX1      LD      DE,#FF00

ANALYS_EX3      LD      A,(HL)
                CP      C               ;=CP #20
                JR      C,ANALYS_NO
                CP      #80
                JR      NC,ANALYS_NO    ;не #20...#7F
                CP      96
                JR      Z,ANALYS_NO     ;"`"
                OR      C
                CP      "a"
                JR      C,ANALYS_EX4
                CP      "z"+1
                JR      NC,ANALYS_EX4
                LD      A,(HL)
                AND     D
                LD      D,A
                LD      A,(HL)
                OR      E
                LD      E,A

ANALYS_EX4      INC     HL
                DJNZ    ANALYS_EX3

                LD      A,D
                XOR     E
                AND     C
                JR      Z,ANALYS_YES

ANALYS_NO       OR      H

ANALYS_YES      POP     BC,DE,HL
                RET
```

В регистровой паре HL, напомню, должен задаваться адрес первого символа расширения (H<>0!), а результат работы процедуры возвращается во флаге Z (установлен — расширение трехсимвольное, сброшен — односимвольное).

И.РОЩИН,
г.Москва,
Fido: 2:5020/689.53
ZXNet: 500:95/462.53
E-mail: asder_ffc@softhome.net
WWW: http://www.ivr.da.ru

Оптимизация на примере intro "Start"

(Продолжение. Начало в №№7-8/2001)

И.РОЩИН,
г.Москва,
Fido: 2:5020/689.53
ZXNet: 500:95/462.53
E-mail: asder_ffc@softhome.net
WWW: http://www.ivr.da.ru

Оптимизация помещения констант в регистры и ячейки памяти

Загрузка констант в регистры или запись их в память выполняется в реальной программе довольно часто, и оптимизация этих действий может принести весьма ощутимые результаты. О некоторых приемах такой оптимизации, использованных при написании intro, я и расскажу. Когда необходимо обнулить аккумулятор, команду LD A,0 обычно заменяют на XOR A (см. строки 234, 311, 525, 647), что позволяет сэкономить 1 байт/3 такта на каждой такой замене. Праеда, эта команда, в отличие от LD, влияет на флаги, так что тут надо быть внимательным. Например, такой фрагмент (строки 522-526):

```
                BIT   5,(HL)
                LD    A,#23
                JR    Z,SET_COD
                XOR   A
SET_COD         LD    (ADR_CODE),A
```

ни в коем случае нельзя записывать так:

```
                BIT   5,(HL)
                XOR   A
                JR    NZ,SET_COD
```

```
        LD    A,#23
SET_COD LD    (ADR_CODE),A
```

потому что после команды XOR A флаг Z всегда будет установлен (он устанавливается по результату операции, а результат-то нулевой!), так что дальнейшая проверка его значения (JR NZ) теряет всякий смысл, и программа не будет правильно работать.

В строке 116 необходимо поместить 0 по адресу, заданному в регистровой паре HL. Можно было бы сделать это командой LD (HL), 0. Но известно, что к этому моменту регистр B обнулился после DJNZ. Таким образом, достаточно применить команду LD (HL),B. Экономия составляет 1 байт/3 такта.

Перед заполнением экрана текстурой (строки 121...140) необходимо поместить в HL адрес первого обрабатываемого байта видеопамяти — #57FF. Но к этому моменту в HL уже содержится это значение — оно оказалось там после процедуры CLS, выполняющей очистку экрана. Следовательно, команда LD HL,#57FF не нужна. Экономия составляет 3 байта/10 тактов.

В строке 147 необходимо загрузить в DE значение #AA00 (адрес таблицы TABLE_SIN). Но мы знаем, что D=#AA и A=0 (см. строки 142...143). Тогда вместо команды LD DE,#AA00 можно использовать команду LD E,A. Экономия составляет 2 байта/6 тактов. Адрес таблицы TABLE_SIN нарочно был выбран равным #AA00 (см. раздел "Расположение структур данных"), чтобы эта оптимизация стала возможной.

В строке 189 необходимо загрузить в DE адрес таблицы PALETTE. Известно, что перед этим в DE был установлен адрес таблицы TABLE_SIN, и эти 256-байтные таблицы располагаются друг за другом. Тогда достаточно использовать команду INC D вместо LD DE,PALETTE. Экономия составляет 2 байта/6 тактов.

Тот факт что таблицы PALETTE и TABLE_SIN расположены в смежных сегментах, используется и в дальнейшем. Так, в строке 252 команду LD H,PALETTE/256 удается заменить на INC H, так как известно, что в регистре H находится старший байт адреса таблицы TABLE_SIN. Аналогично, в строке 260 вместо команды LD H, TABLE_SIN/256 используется команда DEC H. И в том, и в другом случае экономия составляет 1 байт/3 такта.

Аналогичным образом используется и расположение в смежных сегментах таблицы POSITIONS и данных о паттернах (строки 478...489).

А то, что переменная A_AMP и таблица FREQ_TABLE расположены в одном сегменте, позволяет вообще обойтись без загрузки старшего байта адреса (строки 566...582).

В строках 291...294 необходимо увеличить SP на 4 (иначе говоря, снять со стека два запомненных там значения, теперь уже ненужных) и поместить 0 по адресу TRIGGER. Можно сделать это так — POP AF: POP AF: XOR A: LD (TRIGGER),A. Но известно, что старший байт значения, находящегося на вершине стека, равен нулю (строка 469). Тогда команда XOR A не нужна — достаточно записать по адресу TRIGGER значение, которое окажется в аккумуляторе после первой команды POP AF. Экономия составляет 1 байт/4 такта.

В строках 475...476, для того чтобы обнулить DE, используются две команды — LD D,A: LD E,A — при том что A=0. Это позволяет сэкономить 1 байт/2 такта по сравнению с применением команды LD DE,0.

Если вы хотите побольше узнать об оптимизации загрузки констант в регистры, рекомендую изучить [1].

Оптимизация циклов

Замена цикла с заранее известным количеством повторений ("for") на цикл с выходом по условию ("repeat-until") часто дает дополнительный выигрыш.

Рассмотрим, например, заполнение экрана текстурой (строки 129...140). Длина заполняемой области известна заранее — #1800 байтов. Если бы мы использовали цикл "for", то пришлось бы завести отдельный счетчик (и выделить для него регистровую пару), занести в него количество повторов (#1800), а затем в каждом проходе цикла уменьшать значение счетчика и проверять его на равенство нулю.

А как сделано в intro? Никакого специального счетчика там нет, а вместо этого после каждого прохода цикла производится проверка — не стал ли шестой бит регистра H равен нулю (строка 139)? Объясню смысл этой проверки. Заполнение видеопамяти (#4000...#57FF) происходит от старших адресов к младшим, и в регистровой паре HL хранится адрес текущей заполняемой ячейки. При каждом проходе цикла этот адрес уменьшается на единицу. Когда вся видеопамять оказывается заполненной, адрес уменьшается до значения #3FFF. А теперь обратите внимание на старший байт адреса — когда он лежит в диапазоне #40...#57 (%01000000...%01010111), его шестой бит установлен; но как только он становится равным #3F (%00111111), шестой бит сбрасывается. Таким образом, срабатывает условие выхода из цикла. Просто и эффективно, не правда ли?

Общий прием такой оптимизации заключается в следующем — специальная переменная под счетчик количества выполнений цикла не заводится, а вместо этого послв каждого прохода выполняется проверка определенного условия, связанного с адресом обрабатываемых в цикле данных.

Другие примеры оптимизированных подобным образом циклов вы можете увидеть в строках 149...180 (вычисление таблицы sin), 191...197 (формирование палитры), 238...264 (внутренний цикл эффекта "плазма"), 315...326 (очистка экрана), 667...677 (запись в регистры AY).

Еще один пример — главный цикл intro (строки 213...269). Прежде всего заметим, что организован он несколько необычно — проверка условия выхода выполняется в подпрограмме PLAY, отвечающей за музыкальное сопровождение (строки 478...485). Так сделано из-за того что продолжительность intro определявтся длиной мелодии. После такого выхода из цикла, естественно, необходимо снять со стека лишние значения (запомненное в подпрограмме содержимое AF и адрес возврата), что и выполняется в строках 291...294.

Выход из главного цикла должен произойти при достижении конца таблицы POSITIONS. Длина этой таблицы (строки 402...430) не превосходит 256 байтов. В этом случае следить за достижением конца можно очень просто — по младшему байту адреса текущего элемента. Как только он стал равен младшему байту адреса первой ячейки памяти, лежащей после таблицы — таблица обработана (строки 483...485). Как вы можете увидеть, такое сравнение с переходом занимает всего пять байтов. Если бы таблица POSITIONS была длиннее 256 байтов, то в ней оказались бы по крайней мере две ячейки с одинаковыми младшими байтами адраса. Тогда пришлось бы сравнивать полный адрес элемента с адресом первой следующей ячейки памяти:

```
LD   BC,PLAY
AND  A
SBC  HL,BC
JR   Z,END
```

А это занимает уже восемь байтов — на три байта длиннее.

Повышение скорости за счет табличного задания функций

Используемый в intro графический эффект "плазма" требует вычисления синуса при расчете каждого элемента изображения. Естествввно, чтобы успеть рассчитать все изображение за 1/50 секунды, вычисление синуса должно быть как можно более быстрым. Больше всего подходит для этой цели табличный способ. Сущность его заключается в следующем — имеется таблица, содержащая значения функции при всех возможных значениях аргумента, а когда в программе надо вычислить значение функции, из таблицы просто берется готовый результат.

Применение этого способа, конечно, увеличивает размер программы, но, вместе с тем, значительно повышает быстродействие. Взять число из таблицы можно за несколько тактов, а вычисление синуса с помощью калькулятора ПЗУ (фактически — вычисление суммы ряда) займет на несколько порядков больше времени. В этом нетрудно убедиться, если заметить, что формирование таблицы при запуске intro занимает целых 12 секунд, а вычисляются при этом всего-навсего 256 значений синуса.

Тем не менее, будьте внимательны при использовании табличного задания функции на процессорах, более быстрых по сравнению с Z80. Убедитесь сначала, что выигрыш действительно будет. Дело в том, что обычно процессор работает на большей частоте чем память, и вычисление функции может занять меньше времени, чем сравнительно медленное чтение из памяти.

Повышение скорости за счет раскрытия циклов

Когда надо оптимизировать программу прежде всего по длине, этот способ примвняют редко — ведь при раскрытии циклов длина программы возрастает. Тем не менее, в одном месте intro такой прием все-таки используется.

Обратите внимание на то, как работает процедура очистки экрана (строки 311...328). Сначала происходит ожидание импульса вертикальной синхронизации (команда HALT), затем устанавливается черный цвет бордюра и обнуляется буфер атрибутов (длиной 768 байтов), причем обнуление идет от старших адресов к младшим (так удобнее проверять условие выхода из цикла). Так вот, обнулить атрибуты желательно до того как луч начнет прорисовывать первую строку основного экрана, иначе получится не очень приятный эффект — в одном кадре экран будет не полностью очищен.

Подсчитаем общее время работы максимально оптимизированного по длине цикла (справа от каждой команды указано, за сколько тактов она выполняется):

```
CLS_1       LD      (HL),A    ;7
            DEC     HL        ;6
            BIT     3,H       ;8
            JR      NZ,CLS_1  ;12
```

Получается (7+6+8+12)*768=25344 такта. За это время луч прорисует более 1/3 экрана, а это нам не подходит.

Если бы ограничение на общее время работы цикла было не таким жестким (скажем, не более 24000 тактов), можно было бы сделать так — заменить команду JR на более быструю JP (она на байт длиннее), что обеспечило бы сокращение времени работы на 2*768=1536 тактов. Но в данном случае такое повышение быстродействия оказывается недостаточным.

А вот если раскрыть цикл, как сделано в intro:

```
CLS_1       LD      (HL),A    ;7
            DEC     L         ;4
            LD      (HL),A    ;7
            DEC     HL        ;6
            BIT     3,H       ;8
            JR      NZ,CLS_1  ;12
```

то длина возрастет всего на два байта, а время выполнения составит (7+4+7+6+8+12)*384=16896 тактов. За это время на экране успеет прорисоваться 75 строк (при 224 тактах на строку). Это уже вполне приемлемо — на "Пентагоне" (а именно на нем демонстрировались работы на CC 000) верхняя часть бордюра составляет 80 строк, так что к тому моменту, когда луч начнет прорисовывать первую строку основного экрана, атрибуты уже будут обнулены.

Да, кстати, заметьте, что при раскрытии цикла он не просто повторяется дважды (LD HL,A: DEC HL: LD HL,A: DEC HL). Первая команда DEC HL заменяется на более быструю DEC L. Это возможно, так как при ее выполнении старший байт HL заведомо не будет изменяться.

Самомодифицирующийся код

Как сделать программу очень простой, быстрой и короткой? Зачастую этого можно добиться, если сделать ее самомодифицирующвйся — изменяющей саму себя в процессе работы. Это один из самых мощных способов оптимизации.

В разделе "Выбор наиболее выгодного представления данных" я уже упоминал, что в каждом паттерне громкость либо увеличивается, либо уменьшается, либо остается постоянной, и что информация об этом находится в двух старших битах первого байта описателя паттерна. Так вот, в процедуре проигрывания музыки при переходе к очередному паттерну проверяется значение этих двух битов (строки 491...508), и в соответствии с одним из трех возможных случаев по адресу CODE_I_D_N заносится либо команда INC (HL), либо DEC (HL), либо NOP (строки 513...514). Как нетрудно догадаться, именно эта команда и будет изменять громкость во время проигрывания паттерна (строки 564...567). Если бы для информации о том, как изменяется громкость в текущем паттерне, была выделена отдельная переменная, то затем нужно было бы каждый раз проверять ее значение и в зависимости от него изменять громкость. Это, конечно, менее эффективно.

Далее, в том же разделе я упоминал, что паттерны, в которых повторяются две ноты, хранятся в упакованном виде, и что информация об упаковке паттерна находится в пятом бите первого байта его описателя. При переходе к очередному паттерну значение этого бита проверяется, и в соответствии с ним по адресу ADR_CODE помещается либо код команды NOP (если паттерн упакован), либо (в противном случае) код команды INC HL (строки 518...526). Что же происходит в дальнейшем?

В каждом байте паттерна хранится информация о двух нотах. Если паттерн упакован, то в нем находится всего один байт с информацией о нотах; если нет — то четыре байта. Когда очередные две ноты проиграны, выполняется команда, находящаяся по адресу ADR_CODE (строки 571...572) и изменяющая позицию в паттерне. Если там команда NOP, то позиция не изменится, а значит, далее будут проиграны те же самые две ноты (что и требуется). Если же там команда INC HL, то произойдет переход к следующему байту паттерна, содержащему информацию о следующих двух нотах. Проще и не сделать, не правда ли?

И последний пример. В строке 629 находится команда JR C_NORM — переход на обработку канала C. Когда мелодия заканчивается, в строке 292 во второй байт этой команды (где хранится смещение перехода) записывается 0. Нулевое смещение в команде JR означает просто переход к следующей команде. Таким образом, вместо того чтобы выводить в канале C три чередующихся звука разной частоты (как это было на протяжвнии всей мелодии), программа будет выводить такой же сэмпл, как и в каналах A и B, и в результате мы услышим финальный аккорд.

Размещение переменных непосредственно в командах загрузки

Смысл этого широко используемого приема оптимизации в следующем — вместо выделения отдельных ячеек памяти для хранения одно- и двухбайтных переменных и последующей загрузки их из памяти в регистр или регистровую пару, переменные размещаются непосредственно в командах загрузки. Вот характерный пример:

```
Было:   VAR       DB  15
        ......
        LD        A,(VAR)
        ......
       ----------------------
        4 байта,  13 тактов

Стало:  ......
        LD        A,15
        VAR       EQU $-1
        ......
       ----------------------
        2 байта, 7 тактов
```

Как видим, при этом удается сократить как размер программы, так и время ее выполнения. В табл.1 приведено сравнение команд загрузки и показано, каким получается выигрыш в зависимости от того, в какой регистр/регистровую пару загружаются данные. Здесь: L — длина (в байтах); T — время (в тактах); в столбце "Загрузка из памяти" длина указывается в виде "X(+Y)", где X — длина команды, а Y — длина загружаемых данных; экономия указывается с учетом длины данных.

Обратите внимание, что Z80 не имевт команд для загрузки байта из памяти в регистры B, C, D, E, H, L и в половинки индексных регистров. Из-за этого в программе приходится, например, сначала загружать байт из памяти в аккумулятор, а потом пересылать его в нужный регистр (при этом еще иногда приходится

Табл. 1

Непосредственная загрузка			Загрузка из памяти			Экономия	
Команда	L	T	Команда	L	T	L	T
LD A,N	2	7	LD A,(addr)	3(+1)	13	2	6

LD B,N
LD C,N
LD D,N
LD E,N
LD H,N
LD L,N
	2	7	Нет			>2	>6

LD XH,N
LD XL,N
LD YH,N
LD YL,N
	3	11	Нет			>2	>6

LD HL,NN	3	10	LD HL,(addr)	3(+2)	16	2	6
LD DE,NN	3	10	LD DE,(addr)	4(+2)	20	3	10
LD BC,NN			LD DE,(addr)				
LD IX,NN	4	14	LD IX,(addr) LD IY,(addr)	4(+2)	20	2	6
LD IY,NN							
LD SP,NN	3	10	LD SP,(addr)	4(+2)	20	3	10

запоминать значение аккумулятора в стеке, а затем восстанавливать). Преимущества непосредственной загрузки в этих случаях особенно ощутимы.

Примеры применения этого метода в intro вы можете увидеть в строках 465...466, 478...479, 551...552, 598...600, 602...603, 613...614, 621...623, 641...642 и 650...651.

Если загрузка какой-либо переменной из памяти осуществляется в нескольких местах программы, то необходимо выбрать, в каком именно месте она будет заменена на непосредственную загрузку. При выборе следует руководствоваться тем, чтобы экономия была максимальной. Скажем, если из нескольких возможных мест замены одно находится внутри цикла (а в остальном они равноценны), то надо выбрать именно его, так как при большом количестве повторов цикла выигрыш во времени также будет наибольшим.

Вот пример из intro:

```
(1)     448           LD    HL,(FREQ_A)
        449           LD    (FREQ_B),HL
        ..............................
(2)     598           LD    DE,9*256+(freq_g4-freq_shift)
        599 FREQ_A    EQU   $-2    ;E
        600 A_AMP     EQU   $-1    ;D
```

Почему непосредственная загрузка использована именно в (2), а не в (1)? Потому что в (2) загрузка выполняется в регистровую пару DE, а в (1) — в HL. Судя по приведенным в таблице данным, выигрыш будет наибольшим именно при замене загрузки в DE (3 байта/10 тактов против 2 байтов/6 тактов).

(Окончание листинга программы intro "Start". Начало — в №№7-8/2001. Полный листинг выложен на http://radio-mir.com)

```
584 NE_NOTE
586 ;Обработка канала A (центрального). В самом начале в канале
587 ;A стоит нота g4 с громкостью -9 (см. ниже). При первом
588 ;вызове PLAY эти значения будут скопированы для канала B,
589 ;и там будет тихо (амплитуда 2->1->0) звучать эта нота
590 ;(именно эта - так как и дальше она там используется).
591 ;Вообще-то в канале B в самом начале должна быть тишина,
592 ;но так сделать проще:
598             LD    DE,9*256+(freq_g4-freq_shift)
599 FREQ_A      EQU   $-2    ;E
600 A_AMP       EQU   $-1    ;D
602 NE_NOTE_F   LD    HL,0
603 POS_IN_S_A  EQU   $-2
604             LD    A,(HL)
605             INC   HL
606             LD    (POS_IN_S_A),HL
608             LD    L,8
609             CALL  IZM_FRQ
611 ;------------ Обработка канала B (левого): ------------
---
613             LD    HL,0
614 POS_IN_S_B  EQU   $-2
615             LD    A,(HL)
616             PUSH  AF ;для фин. аккорда
617             INC   HL
618             LD    (POS_IN_S_B),HL
620             LD    L,9
621             LD    DE,0
622 FREQ_B      EQU   $-2    ;E
623 B_AMP       EQU   $-1    ;D
624             CALL  IZM_FRQ
625             POP   AF
627 ;--------- Обработка канала C (правого): ------------
629             JR    C_NORM
630 TRIGGER     EQU   $-1
632 ;В конце мелодии (финальный аккорд) в TRIGGER
633 ;записывается 0, что означает переход к следующей команде.
636             LD    L,#A
637             LD    E,freq_g5-freq_shift
638             CALL  IZM_FRQ
639             JR    OUT_AY
641 C_NORM      LD    A,2              ;Прошлое смещение
642 SM_IN_ORN   EQU   $-1             ;в орнаменте.
644             INC   A
645             CP    3
646             JR    NZ,SAVE_SM
647             XOR   A
648 SAVE_SM     LD    (SM_IN_ORN),A
650             ADD   A,ornament_0     ;N текущего орнамента.
651 ORNAMENT    EQU   $-1
652             ADD   A,ORNAMENTS\256
654             LD    H,ORNAMENTS/256
655             LD    L,A
657 ;HL указывает на значение делителя частоты.
660             LD    A,(HL)
661             LD    HL,#8004
662             CALL  IZM_FRQ_2
664 ;--------------- Выводим дамп AY: -----------------
666 OUT_AY      LD    L,#0A      ;H=#80
667 OUT_AY_1    LD    BC,#FFFD
668             OUT   (C),L
669             LD    B,#BF
670             OUTD
672 ;После OUTD флаг C=0, если значение H не изменилось после
673 ;уменьшения, и C=1, если изменилось (и если в (HL) не
0).
674 ;A вы не знали? ;-)
677             JR    NC,OUT_AY_1
678             RET
680 ;-------------------------------------------------
682 IZM_FRQ     SUB   D
683             LD    H,#80
684             LD    (HL),A ;громкость
686 ;Сейчас в регистре L номер регистра AY для амплитуды
687 ;канала, преобразуем его в номер регистра AY
688 ;для младшего байта частоты того же канала:
691             SLA   L        ;*2
692             RES   4,L      ;-16
694             RLCA
695             RLCA
696             RLCA
697             AND   7
698             ADD   A,E
699             SUB   2
701 IZM_FRQ_2   ADD   A,freq_shift
703 ;После прибавления freq_shift в A будет младший байт
704 ;делителя, а во флаге C - старший байт.
707             LD    (HL),A
709             INC   L
710             LD    A,0    ;Получаем значение старшего байта
711             ADC   A,A    ;и выводим его.
713             LD    (HL),A
715 ;           RET (почему эта команда закомментарена - см.ниже)
716 ;-------------------------------------------------
717 ;Таблица частот (каждой ноте в таблице соответствует
```

```
718 ;значение делителя частоты минус смещение freq_shift, таким
719 ;образом, на одну ноту тратится один байт, а не два).
722 ;Значение freq_shift подобрано так, чтобы первый байт
723 ;таблицы был равен #C9 и одновременно служил кодом команды
724 ;RET, что сокращает общую длину программы.
731 ;Таблица частот совмещена с таблицей ORNAMENTS
732 ;(8 байтов), где хранятся значения freq. Для использования
733 ;нужно знать смещение в этой таблице, тогда ornament - это
734 ;3 байта, отсчитываемые от этого смещения.
742 FREQ_TABLE DB   freq_g4-freq_shift
743          DB   freq_b4-freq_shift
744 ORNAMENTS  DB   freq_a4-freq_shift
745          DB   freq_c5-freq_shift
746          DB   freq_e5-freq_shift
747          DB   freq_g5-freq_shift
748          DB   freq_b5-freq_shift
749          DB   freq_d5-freq_shift
750          DB   freq_f5-freq_shift
751          DB   freq_a5-freq_shift
753 ;------------ Таблица нот в каждом паттерне -----------
755 ;В первом байте паттерна кодируется изменение громкости,
756 ;признак "упакованный паттерн" и номер орнамента. В следую-
757 ;щих байтах (в одном или четырех) - ноты, по две в байте.
758 PATTERN_0  DB   code_dn+code_pack+ornament_0
759          DB   note_c5*16+note_g4
761 PATTERN_1  DB   code_dn+code_pack+ornament_1
762          DB   note_d5*16+note_a4
764 PATTERN_2  DB   code_dn+code_pack+ornament_0
765          DB   note_e5*16+note_b4
767 PATTERN_3  DB   code_up+code_pack+ornament_0
768          DB   note_e5*16+note_b4
770 PATTERN_4  DB   code_up+ornament_0
771          DB   note_e5*16+note_c5
772          DB   note_d5*16+note_e5
773          DB   note_a5*16+note_g5
774          DB   note_f5*16+note_e5
776 PATTERN_5  DB   code_dn+code_pack+ornament_1
777          DB   note_a5*16+note_d5
779 PATTERN_6  DB   code_max+code_pack+ornament_2
780          DB   note_g5*16+note_d5
782 PATTERN_7  DB   code_dn+code_pack+ornament_2
783          DB   note_b5*16+note_e5
785 PATTERN_8  DB   code_dn+code_pack+ornament_0
786          DB   note_f5*16+note_c5
788 PATTERN_9  DB   code_up+ornament_0
789          DB   note_g5*16+note_c5
790          DB   note_g5*16+note_c5
791          DB   note_a5*16+note_c5
792          DB   note_a5*16+note_c5
794 PATTERN_10 DB   code_max+ornament_1
795          DB   note_a5*16+note_d5
796          DB   note_f5*16+note_d5
797          DB   note_a5*16+note_d5
798          DB   note_f5*16+note_d5
800 PATTERN_11 DB   code_max+ornament_3
801          DB   note_e5*16+note_a4
802          DB   note_c5*16+note_a4
803          DB   note_e5*16+note_a4
804          DB   note_c5*16+note_a4
806 PATTERN_12 DB   code_max+ornament_0
807          DB   note_g5*16+note_c5
808          DB   note_e5*16+note_c5
809          DB   note_g5*16+note_c5
810          DB   note_e5*16+note_c5
812 END_PLAY
814          ORG  #81FD
816 ;Константы, прибавляемые к значению амплитуды в sample
817 ;и указывающие на смещение частоты:
820 up_2      EQU  #80
821 up_0      EQU  #40
822 dn_2      EQU  #00
824 ;Первые 4 байта таблицы sample:
826 SAMPLE    DB   #F+up_0
827          DB   #E+up_0
828          DB   #D+up_0
829          DB   #D+up_0
```

(Окончание следует)

А.ЛУКЬЯНОВ,
г.Туймазы, Башкортостан.

Король Мастеров

Хочу предложить читателям журнала свою программу — игру "Король Мастеров". В эту игру могут играть два игрока. Смысл игры заключается в том, чтобы делая ходы по очереди, поставить четыре кубика одного цвета в ряд по вертикали, по горизонтали, либо по диагонали.

Ходы осуществляются кнопками A, B, C, D, E, F (для первого игрока) и 1, 2, 3, 4, 5, 6 (для второго игрока).

Возможен вариант этой игры, когда вторым игроком является компьютер, который может заменить живого партнера.

```
  5 FOR F=USR"A" TO 65439: READ A: POKE F, A: NEXT F:
BORDER 6 INK 9: DIM a(6,6):DIM B(6,6):LET D=5:
FORF=1 TO6: LET D=11-D:FOR G=1 TO 6: LET D=11-D:
LET B(F,6)=d: LET A(F,6)=9: FOR U=-2 TO 0:
PRINT AT 0, F*3-1; CHR$(64+F); AT 6*3-1,0;
G; PAPER D; AT G*3+U, F*3-2;"     ":
NEXT U: NEXT G: NEXT F: LET N=0:
DATA 128,64,32,31,16,16,16,0,0,0,
255,0,91,0,0,1,2,4,250,10,200,232,
104,16,16,16,16,16,16,16,20,0,0,0,0,0,
0,0,0,40,40,8,40,8,40,8,40,20,18,
21,16,31,32,0,128,0,0,128,0,245,
0,0,0,8,8,8,8,88,0,2,77
  10 PRINT AT 0,23; "Игрок";n/4+1; AT 1,2,3; "Ваш ход";
AT 3,25; " "; FOR H=0 TO 48 STEP 48: FOR F=1 TO G:
IF INKEY$=CHR$ (48+F+H) THEN
PRINT AT 3,25; CHR$(48+F+H/3); AT 1,23;" ":
GO TO 30+H*110/48
  20 NEXT F: NEXT H: GO TO 10
  30 FOR x=6 TO 2 STEP-1: LET A(x,F)=a(x-1,F): NEXT x:
LET A(1,F)=N: FOR x=1 TO 6: PAPER a(x,F): LET I=1:
IF A (x,F)=9 THEN PAPER B(x,F): LET I=0
  40 BEEP 0.03,x: FOR G=-2 TO 0: FOR H=-2 TO 0:
PRINT AT F*3+H, G+x*3: CHR$( 32+(114+3*(H+2)+G)*I):
NEXT H: NEXT G: NEXT X
  50 PAPER 7: RESTORE 125: LET S3=0:
LET S2=0: READ A,B,C,D,E : LET S=D:
  55 FOR F=A TO B STEP C: LET D=D+e
  60 IF H=2 THEN LET U=F: LET F=D: LET D=U
  65 IF A(F,D)=9 THEN LET S1=0: LET S2=0
  70 IF A(F,D)=0 THEN LET S1=S1+1
  75 IF A(F,D)=4 THEN LET S2=S2+1
  80 IF S1=4 THEN LET S3=1
  85 IF S2=4 THEN LET S4=1
  90 IF H=2 THEN LET U=D: LET D=F: LET F=U
  95 IF H<=2 THEN IF D<6 THEN GO TO 55
  100 IF H<=2 THEN IF D>5 THEN LET D=S
  105 NEXT F: NEXT H
  110 IF S3=1
THEN IF S4=1 THEN PRINT AT 10,17; "Ничья": STOP
  115 IF S3=1
THEN PRINT AT 10,20; "Игрок 1"; AT 11,20; "Победа": STOP
  120 IF S4=1
THEN PRINT AT 10,20; "Игрок 2"; AT 11,20; "Победа": STOP
  125 LET n=4-n: GO TO 10:
DATA 1,6,1,0,1,1,6,1,0,1,1,6,1,0,1,6,
1,-1,0,1,4,1,-1,0,1,5,1,-1,0,1,
6,2,-1,1,1,1,6,3,-1,2,1,3,6,1,0,
1,2,6,1,0,1,1,5,1,1,1,1,4,1,2,1
  140 FOR x=6 TO 2 STEP-1: LET A(F,x)=A (F,x-1):
NEXT x: LET A(F,1)=n:
FOR x=1 TO 6: PAPER A(F,x): LET I=1:
IF A(F,x)=9 THEN PAPER B(F,x): LET I=0
  150 BEEP 0.03,x: FOR G=-2 TO 0: FOR H=0 TO 2:
PRINT AT x*3+G,F*3-H; CHR$ (32+(114+3*(G+2)-H)*I):
NEXT H: NEXT G: NEXT x: GO TO 50
```

А.ВЕНДИЛОВСКИЙ,
г.Минск.

Дальнобойщики 2

Понеслась душа в рай по кочкам...
Народная мудрость.

Глава первая. Лирическое отступление

Я не хотел браться за эту игрушку. Я доказывал себе, что из симуляторов — единственное, что согласен погонять, так это космические петалки или, в крайнем случае, NFS. Но меня не слушали. У меня отобрали мой любимый шотган и даже дарственный меч от самой королевы Катерины с надписью "За взятие Стедвика и героическую оборону Эрафии", не говоря уже о том, что куда-то спрятали форму GDI и шапку-ушанку с фонариком +20 ко всем параметрам. Но я продолжал сопротивляться, пока мое начальство с милой улыбкой (было в этой улыбка что-то такое, что заставляло нильских крокодилов нервно сглатывать, не веря своим глазам) не пообещало лишить меня не только зарплаты и премии, но и стереть с винчестера любимый Unreal и послать объяснять ламерам, где искать клавишу "anykey". Таким образом, все мои дальнейшие попытки сопротивления были в корне подавлены, мне был вручен фирменный джевел-диск с игрой, и меня попросили удалиться, прозрачно намекнув, что пора браться за работу...

Глава вторая. Тормоз-ручник, жадность и ложка дегтя

С установкой программы проблем не возникло — инсталляция прошла гладко и чисто, все запустилось с первого раза. Но вот потом... Нет, вы не подумайте ничего плохого, игра действительно нормально запустилась с первого раза, только вот на тех системных требованиях, что прописаны на обложке, играть невозможно — 64 Мб памяти для нее совершенно несерьезно, бедный винчестер шуршал пластинами за двоих, а моя виртуальная машина дергалась такими рывками, что не то что играть, в поворот вписаться было невозможно. Для решения проблемы пришлось срочно одолжить еще 64 Мб памяти, мысленно сочувствуя тем, у кого такой возможности нет. Движок игры меня впечатлил, но как потом оказалось, проблема была не в нем. Никому не надо объяснять, сколько потоков, памяти и других ресурсов уходит на проигрывание файлов в формате mp3. Разработчики использовали именно этот стандарт для кодирования звука и музыкальных треков, и это в программе, которая сама по себе нещадно эксплуатирует процессор и память! По скромным прикидкам, если бы музыкальные трвки были записаны в стандартном виде, а все остальные звуки — в привычном формате, то минимальных системных требований за глаза хватило бы для нормальной работы. Небольшое расследование показало — программисты старались ужать игру до одного компакта любой ценой, то ли боясь пиратских "урезок", то ли не веря, что это творение будет продаваться на двух CD. Это самый крупный недостаток "Дальнобойщиков 2".

Глава третья. "Эх, прокачу!"

Раздел обучения был краток — мол, смотри: там стоянка, тут заправка, а здесь склады. Поломаешь машину — звони технарям. И будь осторожен — копы строгие, да братки на дороге шалят. После таких напутствий захотелось поскорее в дорогу. Покрутив пустой кар по дороге, чтобы привыкнуть к управлению (тут тоже есть небольшой недостаток — когда смотришь на дорогу из кабины, кажется, что грузовик занимает минимум две трети всей дороги, и просто диву даешься, как ему поместиться на одной полосе или, тем более, идти на обгон. При переключении на внешнюю камеру все становится просто и понятно. Насколько я помню, та же проблема была и в первых "Дальнобойщиках"), зарулил на склад. В моей кассе денег было кот наплакал — на хороший кар не разоришься, да и крупный товар не доверят. Все взвесив и решив не рисковать, я выбрал апельсины, благо пункт конечного назначения был недалеко, и товар малобьющийся. С такими радостными мыслями вырулил на трассу, и тут все началось...

Все необходимые параметры были перед глазами — спидометр, обороты движка, повреждения машины и наличие топлива. Кокпит стандартный, но, как позже выяснилось, у каждого грузовика он свой. Заведя кар, потихоньку выполз на трассу, завершая разворот возле стоянки. И тут мне пришлось познакомиться с еще одной неудачной идеей разработчиков. Как только в моем поле зрения оказывался свободный кар, сразу же навязчиво выползала табличка с его характеристиками и ценой, напрочь закрывая полэкрана. Так что оставшаяся часть поворота выполнялась "на авось" и на честном слове, а я взял на заметку случившееся, чтобы не попасть в какую-нибудь неприятную историю в будущем.

Мотор взревел, но загруженный под завязку автомобиль тяжело набирал скорость. Разница по сравнению с пустым каром была огромна, и пришлось заново привыкать к ощущениям, а время на выполнение заказа тихо таяло. Быстро сориентировавшись по карте, я рванул по самому короткому маршруту. Солнышко светило, погода была хорошей, и тяжелейший удар по моему кару стал для меня полной неожиданностью. В зеркале заднего обзора четко отражался грузовик моего конкурента. И если в первый момент я и подумал о случайности, то второй и третий удары быстро развеяли мои заблуждения — этот негодяй очень настойчиво пытался прижать меня к обочине и вывести из игры. Сожалея об отсутствии шотгана или даже простой монтировки, после долгих маневров мне удалось немного оторваться от наглеца. Проценты повреждения груза навевали мысли о том, что на склад поступит уже натуральный апельсиновый сок, а не апельсины. А время таяло. Выжав педаль газа до упора, я не обратил внимания на две вещи — ограничитель скорости и копов в кустах. На финишной прямой внезапно за спиной раздался голос, столь же приятный, как звук, издаваемый гвоздем, царапающим стекло: "Водитель! Немедленно остановитесь!" — и штраф, очень хороший штраф, съедающий почти половину будущей прибыли. Пока суд да дело, мои конкуренты уже успели разгрузиться, а меня дожидались лишь квитанции о неустойке, просроченном времени и побитом грузе...

Load game. Please, wait...

Пока затишье, и твой багажник пуст, можно полюбоваться открывающимися видами. Хорошо сделано — вон ветер качает одиноко стоящее дерево, и можно различить шум листвы сквозь его порывы. Лесные массивы с виду мало отличаются от тех, что можно увидеть через лобовое стекло в настоящем автомобиле. Правда, одна моя попытка "срезать" маршрут через лес не удалась — при ближайшем рассмотрении он оказался сплошной глухой стеной. Видно это лишь когда упрешься в деревья своим каром, а тогда уже не лес рассматривать надо, а думать, как ты сюда попал, и что собираешься делать дальше. Прорисовка внутренней части кабины тоже на высшем уровне, не хватает только собственных рук на руле. Можно переключить камеру на внешний обзор и посмотреть, насколько сильно повреждена ваша машина. Система повреждений заслуживает похвал, хотя ожидалось нечто большее (нечто большее можно увидеть, но без второго GeForse не обойдешься :-( ). На соседней стоянке выбор машин невелик — пара тягачей и прицеп с цистерной. Правда, если есть желание, можно связаться с дилером, и он устроит любую модель в короткие сроки. Сами цены у него нормальные, но за доставку он берет просто бешеные деньги. Ага, вон на дороге одинокий кар, и тоже продается. Только очень не хочется рисковать — эти "жучки" запросто отдадут тебе краденный тягач, а с полицией разбираться уже тебе. И результат разборок всегда одинаков — конфискация тягача и солидный штраф, так что экономить на машинах не будем. А денег мало, очень мало, придется идти "ва-банк". Вот поэтому я и сижу здесь, ожидая самого крупного заказа в этой игре — заказа на алмазные шахты.

Заказ не заставил себя долго ждать — один взгляд на премиальные заставил плюнуть на всех гаишников и братков, осталось только пожелать конкурентам скользкой дороги и пробитых скатов. Лихо вырулив на трассу, я сразу стал выжимать скорость по максимуму (благо, дорогу я запомнил по истории с апельсинами, они еще долго будут мне в кошмарных снах являться, укоризненно глядя на меня из разбитых ящиков, облитые чистейшим калорийным апельсиновым соком. Брр!). "90-60-90!" — так учил меня держать скорость при встрече с копами один из асов автомобилевождения, но сейчас уже не до церемоний. На памятной развилке я вновь услышал противный вой сирен за спиной, но скорость не сбросил. Дорога узкая, по бокам лес, попытается подрезать — быстро отправлю на обочину хвою на ветках пересчитывать. Но погоня за водителем-лихачом, кажется, для них праздник — в зеркале заднего вида уже виднелись три полицейских машины, а с четвертой я едва не столкнулся лоб в лоб. Ну и ладненько, пусть там и болтаются, лишь бы не мешали, в их угрозы открыть огонь по мне я не верил. И тут я встретил пятого преследователя... Он не догонял меня, а просто выложил "ежа" на дороге и спокойненько ждал моего прибытия. Нв на тех напал! Я такие шипы столько раз в NFS объезжал! Вот только не на грузовом тягаче с полным прицепом. Каким образом это корыто на колесах не перевернулось, я до сих пор не понимаю, одно точно — в следующий раз мне так не повезет.

Черный джип, вынырнувший из ниоткудв, точно подтверждал мои догадки. "Эй, браток! Тормози! Базар есть!" — знаю уж, какие бывают "базары". И только поддал газу. Мои намерения уж точно не понравились владельцам джипа, а автоматная очередь по движку заставила похолодеть — не дай бог, остановят! Итак, сзади — копы, впереди — братки, и что-то быстро темнеть начало... Первые капли дождя по ветровому стеклу и вспышки молнии — обещанная гроза настигла меня. Стена воды, смахиваемая дворниками, мокрая дорога, но мне опять повезло — на встречной полосе шел тоже тягач. Выжав вновь газ, я протаранил несущийся впереди меня джип и мощным ударом выбросил его на встречную полосу. Грохот за спиной был как бальзам на раны. Но на этом везение закончилось раз и навсегда. Перекрывая шум грозы, над дорогой завис вертолет, сквозь удары грома я расслышал стук автоматной очереди, и огоньки пламени стали вырываться из-под капота. Только большая скорость и склон горы позволили мне проскочить его (кстати, позже я проезжал там, этот "стрекозел" по-прежнему висел над дорогой и требовал, чтобы я остановился, бедняга). Но машина уже не слушалась руля, колеса выписывали огромную восьмерку, а движок заглох в ста метрах от склада...

Load game. Please, walt...

Денег все меньше и меньше, на последнюю горсть "зелени" приобрел цистерну-полуприцеп и, заправив его под горлышко бензином, не спеша отправился к бензоколонке в поселке Горный. Деньги предлагали хорошие, а конкуренция небольшая. Товар опасный, и ни в коем случав не терпел повреждений. На мир тихо опускалась ночь, в свете фар придорожный лесок стал постепенно редеть, а обочина — все больше и больше покрываться снегом. Когда из тьмы проступили очертания гор, я с опозданием догадался, что название свое поселок получил неспроста. Дорога все больше и больше виляла, накручивая километры, и с каждым мгновением становилась все более и более скользкой. Удерживать машину становилось с каждым часом все труднее и труднее. Но я уже двигался вторым в этой гонке, правда, лидера с трудом можно было различить в этой тьме. Испытав на секунду легкий укор совести, я быстро набрал телефон братков и "заказал" лидера... Через минуту черный джип легко обогнал меня на повороте — вскоре грузовик фаворита был прижат к обочине, и разговоры там велись явно не задушевные. Я лидер! Но нужно и удержать свое лидерство, мои конкуренты так же легко "закажут" меня браткам, да и рискованно тут упражняться в скоростной езде — быстро вылетишь в пропасть. Мигающие фары в зеркале заднего вида заставили занервничать и непроизвольно нажать на газ. Многотонная махина заскользила по ледяному крошеву, уже абсолютно не слушаясь водителя. В доли секунды отстраненно проскочила мысль о том, что нужно было поставить на колеса цепи, за стеклом мелькнули обрыв и бетонная стена, машина замерла, перекрыв всю дорогу. Прежде чем я успел сделать один вдох и осознать, что остался жив, прямо в мою цистерну, отчаянно тормозя, врезался другой бензовоз. Сталь, алюминий и хром, сминаемые как бумага, тонны горючего, хлестнувшие на дорогу. "Повреждение груза ...%"... и яркий сполох огня, разорвавший ночь, который я уже не увидел...

Load game. Please, wait...

Но однажды я вырвал свой шанс. Фортуна была благосклонна, и я первым привез груз телевизоров. Кроме денежного вознаграждения, меня ожидал сюрприз — лицензия на найм водителей. Так я стал потихоньку создавать свою империю, разъезжая по стоянкам и выискивая способных водителей. Постепенно денежный ручеек стал полноценной рекой, все больше хороших каров закупалось моими служащими, и все меньше они делали ошибок. Я уже не рвзвозил грузы сам, как раньше, а если и делал это, то только ради удовольствия. Меня волновали более глобальные проблемы — у дилера появился не тягач, а чудо — сплошной хром, движок просто всесильный, такого еще не было в моей коллекции...

Системные требования

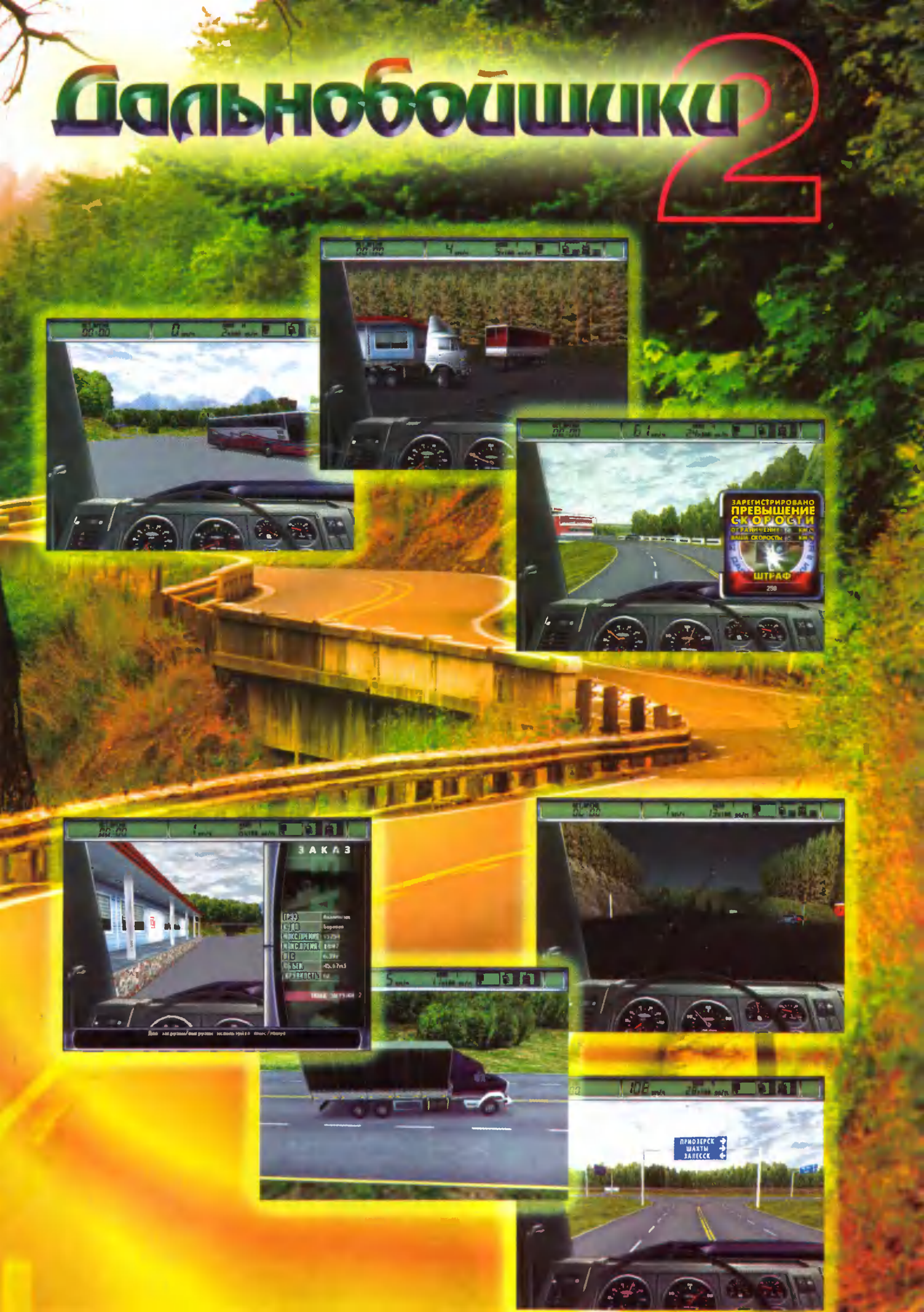
Обратите внимание — игра не работает под Windows NT 4.0 или младше. Поддерживаются Windows 95, Windows 98, Windows Me и Windows 2000. Игра требует графического акселератора.

Поддерживаемые разрешения экрана: 640x480, 800x600, 1024x768 и 1600x1200.

Минимальные требования:

- Pentium II 300 МГц (рекомендуется Pentium III);
- 64 Мб ОЗУ;
- Direct3D-совместимый графический акселератор;
- звуковая карта (рекомендуется);
- рулевое колесо и педали (рекомендуется);
- Windows 9X/Me, Windows 2000;
- DirectX 7.X.

Дальнобойщики 2




КОЛЛЕКЦИЯ ИГРУШЕК
ОПЕРАЦИЯ "ПЛАСТИЛИН"
ENGLISH VERSION
КОНАН
IMMORTAL SOVERIGNS
TIMEGA
ПОЛНЫЙ РУССКИЙ ПЕРЕВОД
TONY HAWK'S PRO SKATER 2
2В1
АНГЛИЙСКАЯ ВЕРСИЯ
ACTIVISION
ФАРГУС
НА РУССКОМ ЯЗЫКЕ
Summoner
ОНИ
BUNGIE
Бука
НА РУССКОМ И АНГЛИЙСКОМ ЯЗЫКЕ
Русский Проект
QUAKE III
+ QUAKE III TEAM ARENA
ARENA 2В1
PC
RUS
Игры для детей от компании Бука
ПЯТАЧОК
И Разные Звери
5-13 Лет
ПРОТИВОСТОЯНИЕ
ENGLISH VERSION
Fallout TACTICS
BROTHERHOOD OF STEEL
НОВЫЙ ДИСК
ДЕТСКАЯ БИБЛИОТЕКА
СУПЕР ИНТЕЛЛЕКТ
Суперголоволомки
ФАРГУС
НА РУССКОМ ЯЗЫКЕ
СЕРЬЕЗНЫЙ СЭМ
ПОЛНАЯ РУССКАЯ ВЕРСИЯ