http://radio-mir.com

индексы: 48925, 45995

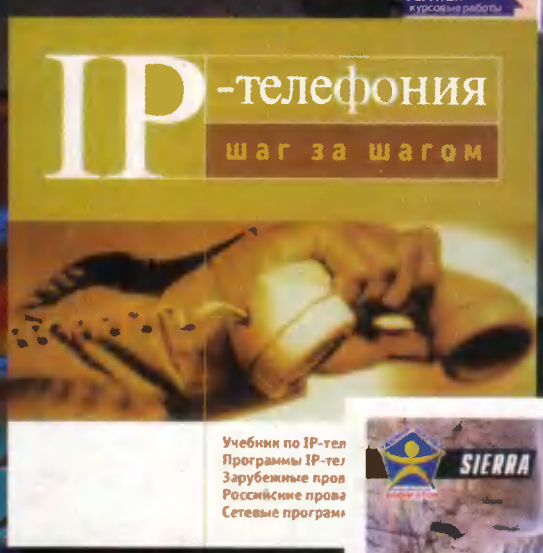
радиомир

5•2003

Ваш компьютер

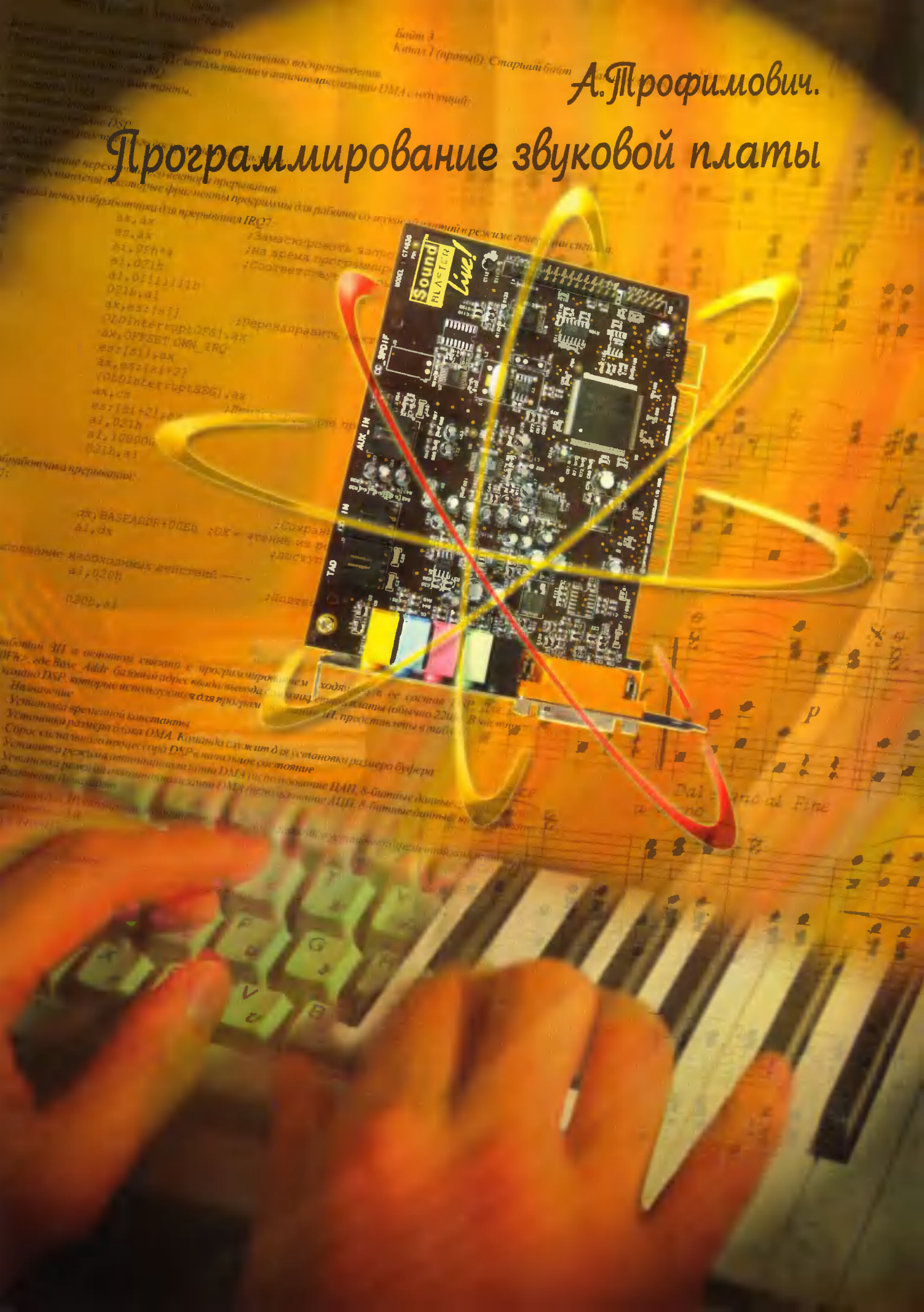
Microsoft Windows xp Edition

официальная редакция



А. Трофимович.

Программирование звуковой платы



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

Главный редактор
Ольга СТРЫЖАНКОВА

Зам. гл. редактора
Иван БЕЛЬСКИЙ

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

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

Компьютерная верстка —
Янина БЕЛЬСКАЯ

Техническая графика —
Наталья БЕЛЬСКАЯ,
Александр ГОРАЮТИН,
Мария КАЛАБУХОВА

Оформление обложки —
Наталья БЕЛЬСКАЯ,
Надежда БОГОМОЛОВА

Адреса для писем:
119454, РФ, г.Москва, а/я 37,
факс (095) 131-97-17;
220095, РБ, г.Минск-95, а/я 199

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

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

Материалы для публикации принимаются в рукописном, печатном и электронном вариантах

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

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

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

Подписано к печати 25.03.2003 г.
Формат 60 x 84 1/8.
Печать офсетная, 6 печ. л.
Цена свободная.

Отпечатано в типографии
ЗАО "Красногорская типография".
Заказ №762.

Радиомир
Ваш компьютер

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

№5/май/2003

ЧИТАЙТЕ В НОМЕРЕ

Дорогие друзья!

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

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

Редакция.

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

Компьютерная математическая система MATLAB

Б.КИСЕЛЕВ,
г.Минск,
БГАТУ, каф.ВТ.

Использование вычислительной техники для решения научно-технических задач развивается в нескольких направлениях. Отметим два крайних случая [1]. Один —использование универсальных языков программирования, а другой — специализированных программных продуктов для решения наиболее распространенных в той или иной области задач. Каждый из них имеет свои достоинства и недостатки. Первый требует профессионального владения методами и средствами программирования, а также значительных временных затрат, второй — профессионального владения методами и средствами той области знаний, к которой относится программный продукт. Например, пакетами прикладных программ по математической статистике в полной мере могут воспользоваться только профессиональные статистики, а пакетами программ по исследованию систем управления — только те, кто имеет достаточно хорошее представление о математических моделях таких систем и об основных методах их исследования.

В последние годы особый интерес представляет *компьютерная математика*. Под этим термином понимается "совокупность методов и средств, обеспечивающих максимально комфортную и быструю подготовку алгоритмов и программ для решения математических задач любой сложности... с высокой степенью визуализации всех этапов решения" [2]. При этом в подавляющем большинстве случаев предусматривается объединение возможностей текстовых редакторов (например, в формате Word) с собственно математическими системами. Это позволяет создавать электронные документы и книги с "живыми" примерами математических расчетов и высокой степенью графической визуализации всех этапов решения задачи.

Программные средства компьютерной математики реализованы в виде компьютерных математических систем. Существует большое число таких систем, например, Matematica, Mathcad, Maple и др. [2, 3]. Система MATLAB (MATrix LABoratory — матричная лаборатория) занимает среди них особое место как по степени универсальности, так и по сложности. По обилию функций и скорости вычислений MATLAB превосходит большинство подобных систем и является бесспорным лидером в области численных расчетов и математического моделирования различных систем и устройств [1...5].

Поток издаваемой литературы, посвященной системе MATLAB, быстро растет. Хотя все монографии имеют солидный объем и цену, каждая из них освещает лишь небольшую долю основных возможностей системы. "Меньшей кровью" можно познакомиться с системой через журнальные статьи. Отдельные статьи публиковались и в данном журнале.

Система MATLAB — это и операционная среда, и язык программирования, на котором могут быть написаны программы для многократного использования. На этом языке уже написано множество программ для решения самых разнообразных задач во многих областях науки и техники. Коллекции родственных программ, предназначенных для решения задач из той или иной области науки (или техники), объединяются в специальную папку, которую называют пакетом прикладных программ (ППП, MATLAB Application Toolboxes). Существует множество ППП, их семейство непрерывно растет. Постоянно расширяется и содержание каждого отдельного ППП. Насчитывается около 40 ППП, в их числе:

- пакеты для анализа и синтеза систем управления;
- пакеты для идентификации систем;
- пакеты для обработки сигналов и изображений;
- пакеты для вейвлетов;
- пакеты для финансово-экономических расчетов;
- пакет SIMULINK, предназначенный для математического моделирования динамических систем, представленных своей функциональной блок-схемой;
- пакет SimPowerSystem, предназначенный для математического моделирования электроэнергетических систем и устройств;
- пакет SimMechanics , предназначенный для математического моделирования систем и устройств механики и т.д.

Кроме того, из широкого спектра математических пакетов следует упомянуть пакет для построения нейронных сетей и пакет, относящийся к теории размытых множеств.

Понятно, что ни один пакет не может охватить все многообразие проблем и задач, поэтому необходимо владение базовыми программными средствами системы MATLAB как для решения конкретных задач, так и для понимания программ, входящих в ППП. Вместе с тем, некоторые ППП оказались настолько интегрированными с системой MATLAB, что стали составной ее частью. Это относится к ППП Notebook (интеграция с текстовым процессором Word) и Simulink (моделирование динамических систем) [1, 2, 6].

Система MATLAB использует командный режим работы. Имеются возможности компилирования, проблемно-ориентированного и визуального программирования.

Важным достоинством MATLAB является ее открытость и расширяемость. Большинство команд и функций данной системы оформлены в виде текстовых файлов (M-файлов) и файлов на языке C (C++). Пользователь может их модифицировать и создавать новые. MATLAB дает возможность писать самостоятельные программные модули на разных языках, однако штатными являются C++ и свой собственный M-язык

Имеется возможность объединения системы с пакетом символьной математики Maple, пакетом Excel и некоторыми другими.

Описанию системы автоматизации математических и научно-технических расчетов MATLAB посвящено достаточно большое число работ, однако ни один из известных литературных источников не может считаться исчерпывающим. Так или иначе, пользователю приходится обращаться к справочной документации в процессе практической работы.

Отметим, что система MATLAB поставляется с обширной документацией и развитой справочной системой, поэтому опытный пользователь может и без литературных источников освоить работу в данной системе. Определенным неудобством является только то, что документация и справочная система существуют пока только на английском языке. Однако этот пробел можно частично компенсировать источником [4], где имеются переводы по отдельным вопросам, ряд консультационных и учебных материалов, а также возможность участия в форуме.

В системе MATLAB имеется широкий спектр демонстрационных примеров, которые можно модифицировать и использовать в своих целях.

Таким образом, *MATLAB — универсальная интегрированная система, предлагаемая ее разработчиками как язык программирования высокого уровня для технических вычислений.* Этап компиляции полной программы отсутствует. Для выполнения программ необходимо находиться в среде MATLAB. Однако для программ на языке MATLAB созданы компиляторы, транслирующие программы на языке MATLAB в коды языков программирования C и C++. Это решает задачу разработки исполняемых программ, изначально создаваемых в среде MATLAB.

Следует упомянуть, что в MATLAB класс *array* — обобщенный класс объектов-массивов, является прародителем всех остальных встроенных классов. Матричная форма представления операций делает программирование в среде MATLAB очень лаконичным, что снижает трудоемкость работ и повышает надежность результатов.

Предлагаемая серия статей рассчитана на читателя, владеющего компьютерной и математической подготовкой в объеме первого курса технического вуза без расширенного изучения информатики, и ставит целью более или менее последовательно и, в то же время, кратко познакомить читателя с системой MATLAB. Данная публикация основана на лабораторном цикле по математическому моделированию в БГАТУ (Белорусском государственном аграрно-техническом университете) для студентов-механиков. Практика показала, что студенты в состоянии усвоить материал в достаточной степени, чтобы дальше успешно осваивать систему самостоятельно. Отметим, что к моменту изучения системы MATLAB студенты владеют ЭВМ в объеме курсов по программированию для специальностей без расширенного изучения информатики (в БГАТУ основу таких курсов составляют Pascal, Delphi, Excel, Windows).

Популярность системы MATLAB подтверждается тем, что она является базовой в курсах моделирования и численных расчетов таких учебных заведений, как МГУ, МИФИ, МГТУ им. Баумана, Российской экономической академии им. Г.В. Плеханова. Среди минских вузов она популярна в БГУИР.

Учитывая, что читатель имеет начальную компьютерную подготовку, в следующих статьях мы будем описывать лишь те элементы меню, инструментария и конструкций языка, которые необходимы для выполнения предлагаемых действий. Так как интерфейс среды MATLAB является стандартным для программных продуктов, работающих под операционной системой Windows, на его описании мы тоже специально останавливаться не будем.

Содержание следующих трех статей посвящено знакомству с вычислениями в среде MATLAB и элементами языка программирования. Следующие статьи будут содержать разбор простых прикладных задач с одновременным вводом в другие возможности системы.

Литература

1. Муха В.С. Введение в MATLAB: Методическое пособие для выполнения лабораторных работ по курсам "Статистические методы обработки данных" и "Теория автоматического управления" для студентов специальности 53 01 02. — Мн.: БГУИР, 2002, С.40.

2. Дьяконов В. П. Компьютерная математика. Теория и практика. — М.: Нолидж, 2001. — 1296 с.

3. http://exponenta.ru

4. http://matlab.ru

5. Мэтьюз Д.Г., Финк К.Д. Численные методы. Использование MATLAB. — М.: Издательский дом "Вильямс", 2001. — 720 с.

6. А.Гультяев. Визуальное моделирование в среде MATLAB: Учебный курс. — Питер, 2000.

7. Мартынов Н.Н. Введение в Matlab 6. — Кудиц-образ, 2002.

Троичная система счисления

В.ПОНОМАРЕВ,
г.Витебск, ВГУ.

Для описания двухуровненых квантовых систем уже стало привычным следующее представление кубита:

$$|\varphi>=\alpha|0>+\beta|1>. \quad (1)$$

Для описания трехуровневых квантовых систем используется представление в виде кутрита. Например:

$$|\varphi>=\alpha|0>+\beta|1>+j|2>. \quad (2)$$

Но кутрит можно представить и так:

$$|\varphi>=\alpha|0>+\beta|1>+j|-1. \quad (3)$$

Используя двухуровневые системы для построения квантовых ЭВМ, наиболее естественно использовать двоичный код. О недостатках и неудобствах применения двоичного кода говорилось в [1].

На мой взгляд, существует альтернативное решение, полностью свободное от недостатков, проблем и неудобств. Суть этого решения состоит в том, что надо продвинуться на один шаг вперед — перейти от систем с двумя цифрами к системам с тремя цифрами, от двоичного представления информации к троичному. В троичной системе счисления имеется возможность представления натурального кода для чисел со знаком.

Эта возможность реализуется выбором в качестве числовых значений трита (трехзначного аналога бита) чисел 0, 1, -1, т.е. добавлением к двоичным цифрам 0 и 1 "отрицательной цифры -1".

Числовое значение троичного слова длиной n (состоящего из n тритов) выражается при этом применительно к целым числам в виде, аналогичном выражению числа в натуральном двоичном коде, а именно:

$$A = A_{n-1}\cdot 3^{n-1} + A_{n-2}\cdot 3^{n-2} + \ldots + A_i\cdot 3^i + A_2\cdot 3^2 + A_1\cdot 3 + A_0, \quad (4)$$

где A_i — значение i-го трита, т.е. 0, 1 или -1.

В данной интерпретации, из 3^n значений, принимаемых троичным словом длиной n, мы имеем $(3^n - 1)/2$ положительных, ровно столько же отрицательных и одно значение, равное нулю. Знак числа A задается старшим из неравных нулю тритов — число является положительным, если этот трит равен 1; отрицательным, если он равен –1, и равно нулю, если все триты слова нулевые. Так, при n = 3 множество представимых чисел выглядит следующим образом:

-13	-12	...	-2	-1	0	1	2	...	12	13
$\bar{1}\bar{1}\bar{1}$	$\bar{1}\bar{1}0$	...	$0\bar{1}\bar{1}$	$00\bar{1}$	000	001	$01\bar{1}$	...	110	111

Единица с чертой сверху означает -1.

С достоинствами троичного кода читатель может самостоятельно ознакомиться в [1]. Для меня непонятно, почему в большинстве ЭВМ используется двоичный код, хотя, если помните, первый компьютер ENIAK работал в десятичной системе счисления. Я же хочу вернуться к тому, с чего начал.

Если мы из каких-то соображений будем конструировать квантовую ЭВМ, используя для ее построения троичный код, элементарные основы которого описаны выше, то кутрит лучше представлять в виде (3).

Какой именно код и какая именно система счисления получат наибольшее распространения в квантовых ЭВМ, покажет время.

Литература

1. Н.П.Брусенцов. Микрокомпьютеры. — М.: Наука, 1985.

Работа с редактором деловой графики Visio 2002

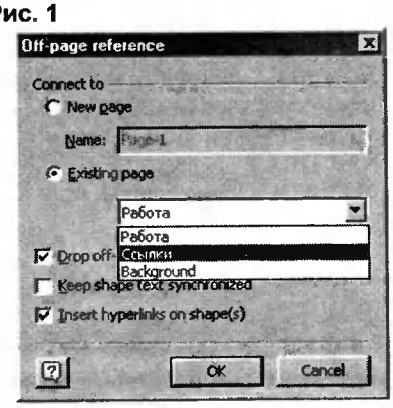
А.ГРИНЧУК,
gav@nihe.niks.by

(Продолжение. Начало в №№12/2002, 1-4/2003)

Работа с web-страницами

Развитие сети Интернет в последние годы заставило практически всех производителей программного обеспечения позаботиться о возможности работы с гиперссылками и публикации результатов работы в виде web-документа. Visio 2002 предлагает в определенном смысле "двусторонний" вариант действий. С одной стороны, имеются все типичные для семейства Microsoft средства разработки web-страниц, с другой — достаточно интересная возможность снятия карты сайта.

Рис. 1



Что касается создания документов для Интернет, то в первую очередь пользователю могут пригодиться графические средства Visio. Самый простой вариант — сохранение диаграмм в формате GIF или JPEG с последующей вставкой на web-страницу. Недостатки здесь достаточно очевидны — как рисунки, так и текст сохраняются в растровом графическом формате. Это значительно увеличивает размеры файлов при одновременном снижении их качества. Поэтому все-таки желательно пользоваться встроенными средствами конвертации в HTML-формат.

Вначале рассмотрим подготовку самого документа Visio. При работе с многостраничными документами, для облегчения навигации и быстрого перехода между страницами удобно использовать фигуру **OffPage Reference** (внестраничная ссылка). Найти ее можно на трафарете **Basic Flowchart Shapes** (**File/Stencils/Flowchart/Basic Flowchart Shapes**). При перетаскивании фигуры на рабочий лист на экране появляется диалоговое окно, в котором можно выбрать либо создание гиперссылки вместе с созданием новой страницы (переключатель в положении **New Page**), либо установить связь с существующей страницей (переключатель соответственно в **Existing Page** (**рис.1**), с возможностью выбора из списка существующих страниц). После этого автоматически осуществляется переход на указанную страницу, где Visio создает гиперссылку для обратного перехода. Если же в диалоговом окне выставить флажок **Keep shape text synchronized**, то подписи на прямой и обратной ссылках будут совпадать. Для создания надписей используется обычный прием — выбор на панели инструмента текста (Text Tool) и щелчок по фигуре. Естественно, эту фигуру можно растягивать и вращать, а вот для изменения формы возможностей меньше — по щелчку правой клавишей мыши доступен выбор только из четырех вариантов (исходящая, входящая, круг и стрелка — **рис.2**). Зато можно изменить "место назначения" (**Format/Behavior/Double Click/Go to page** — **рис.3**). А для правильной работы ссылок после конвертации в формат HTML устанавливается флажок в позиции **Insert hyperlinks on shape** (рис.1). После этих настроек вы получите в документе Visio фигуру, которая по двойному щелчку будет производить переход к другому листу документа.

Рис. 2

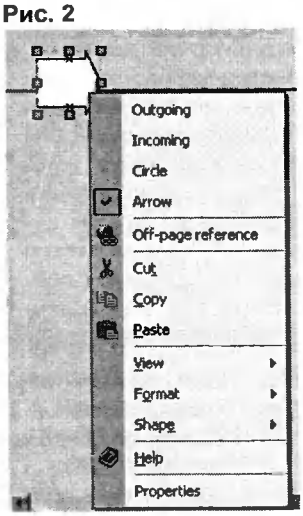
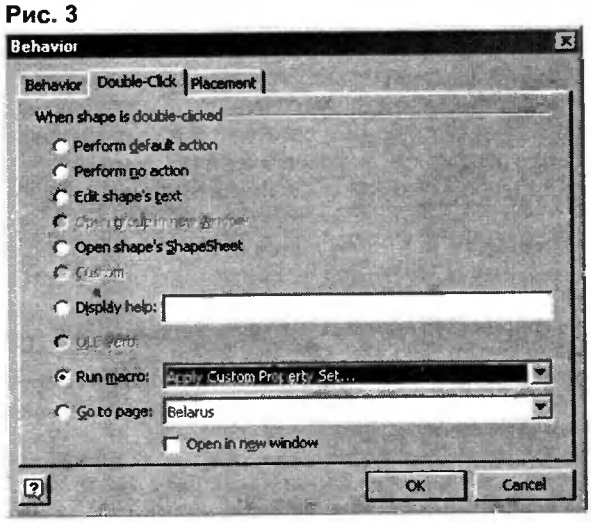


Рис. 3



Возможно также и создание "обычных" гиперссылок. Для этого выделяется объект (обычно это текст) и выбирается команда **Insert/Hyperlinks...**. В диалоговом окне (**рис.4**) указывается следующая информация:

- **Adress** — адрес сайта или путь к файлу;
- **Sub-adress** — адрес страницы на сайте или закладки, либо путь к странице документа Visio;
- **Description** — описание ресурса.

Однако необходимо учитывать некоторые тонкости. Во-первых, выделенный текст не приобретает автоматически "гипертекстовый" вид — синий цвет и подчеркивание ссылок приходится создавать дополнительно. Да-

Рис. 4

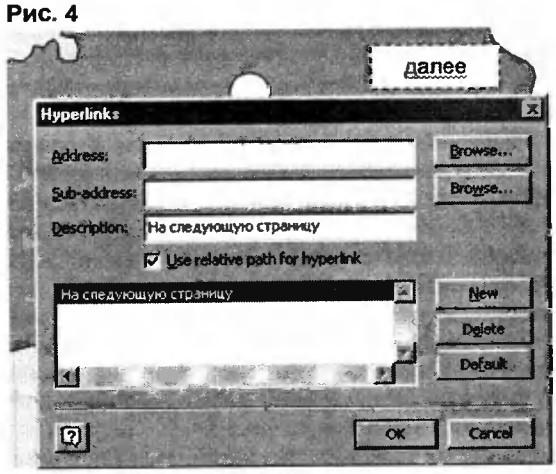


Рис. 5

лее, двойной щелчок мышью приводит не к переходу по ссылке, а к запуску режима редактирования текста. Для перехода используется щелчок правой клавишей мыши и выбор первой команды контекстного меню (**рис.5**), где содержится текст, введенный в поле **Description**. Для обеспечения привычного режима работы — переход по двойному щелчку — после выделения объекта-гиперссылки выполняется команда **Format/ Behavior/Double Click/Go to page** (рис.3). И, конечно, есть возможность задать тип гиперссылки — относительная или абсолютная — установкой или снятием флажка **Use relative path for hyperlink** (рис.4). Первый вариант обычно используется при создании сайтов, при этом файл со ссылкой и файл перехода должны перемещаться синхронно, иначе происходит разрыв связи. Второй вариант предпочтителен, когда во время работы вы перемещаете только документ Visio. Для корректной работы относительных ссылок необходимо сохранить документ Visio до их создания.

После создания необходимых гиперссылок можно приступать и к конвертации в полноценный web-документ. Для этой цели в меню **File** имеется специальная команда **Save as Web Page**. Настройка сохранения производится нажатием кнопки **Publish**. В диалоговом окне (**рис.6**) указываются размер изображения (на вкладке **Appearance**) и режим сохранения файлов — использование длинных имен и создание отдельной папки (на вкладке **Files**). Однако главное содержится на первой вкладке — **General**. Здесь можно выбрать создание окна свойств файла **Property Viewer** (из-за малой информативности для посетителей вашей страницы, от его создания рекомендуется отказаться) и создание ссылок на каждую страницу — **Page Tabs**. Кнопка **Format Options** позволяет задать формат сохраняемых рисунков. Если вам важно, чтобы страницы просматривались любым браузером, то следует предпочесть форматы GIF, JPG или PNG. Тогда все элементы, включая надписи, преобразуются в графический формат, будут работоспособны все ссылки, и внизу экрана появится удобная навигационная панель (**рис.7**). Однако наибольшие возможности открываются при работе с форматом VML — Vector Markup Language — язык векторной разметки. Здесь появляется достаточно редко реализуемая другими программами возможность масштабирования страницы с появлением дополнительного элемента на навигационной панели (**рис.8**). Увы, полноценный просмотр таких web-страниц возможен только в Internet Explorer версии 5.0 и выше.

Рис. 6

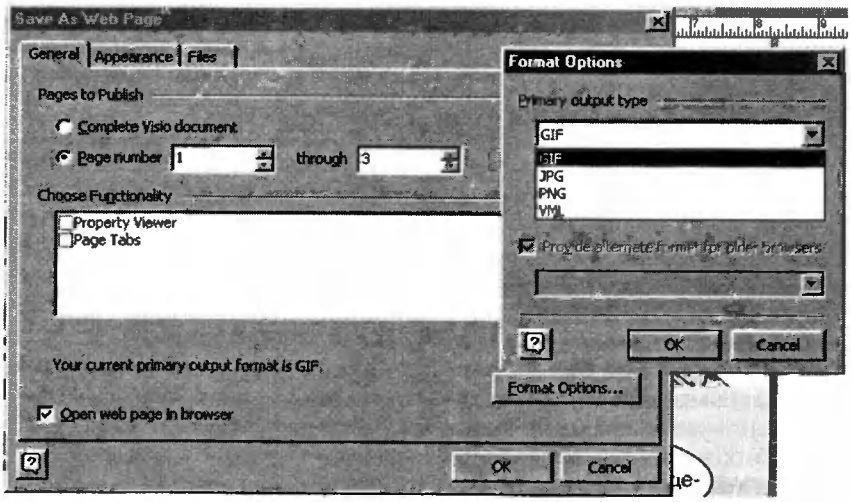
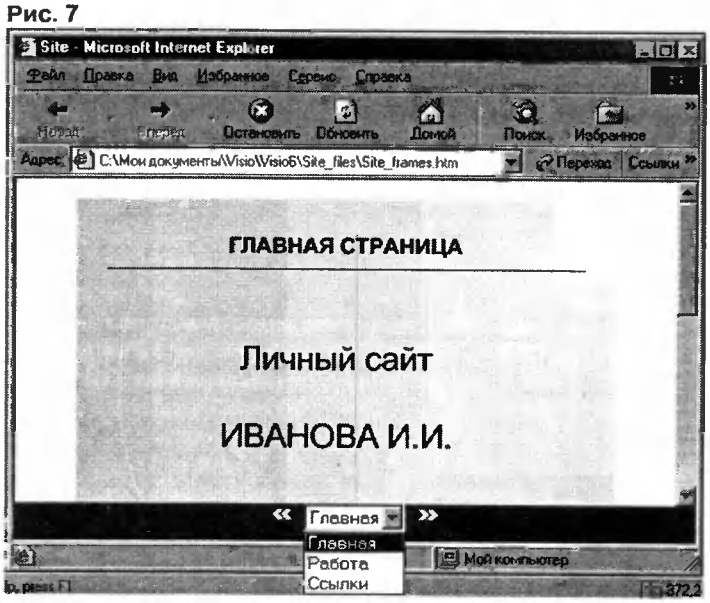


Рис. 7



И хотя такие возможности выходят за рамки привычных представлений о графическом редакторе, корпорация Microsoft не остановилась на достигнутом. Для работы с web-узлами предусмотрено создание карты сайта. В категории **Web Diagram** есть два шаблона: **Conceptual Web Site** для рисования структуры разрабатываемого сайта и, что гораздо интереснее, **Web Site Map** — для создания схемы уже готового и опубликованного сайта.

Рис. 8

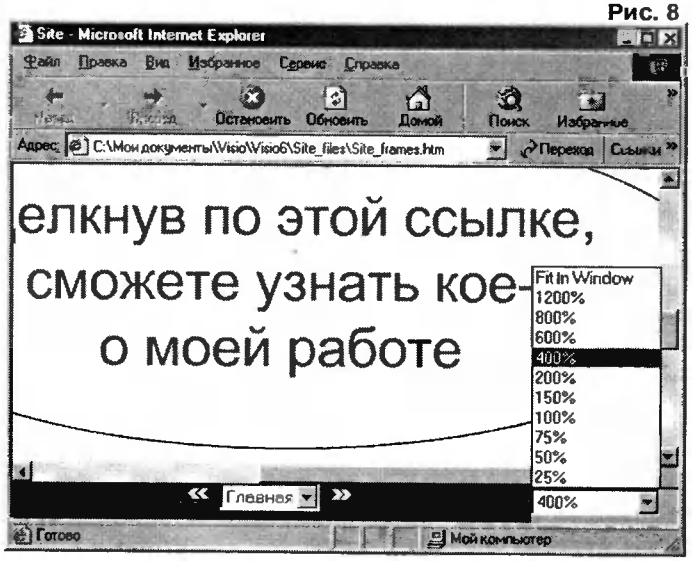
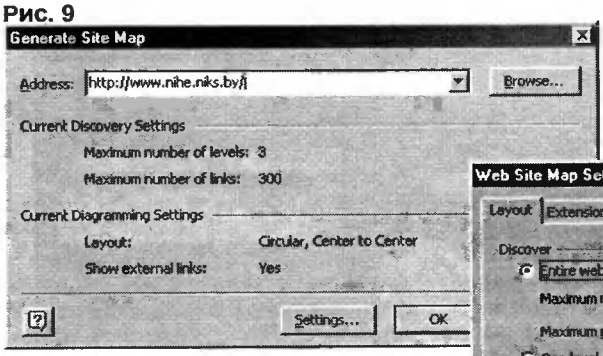
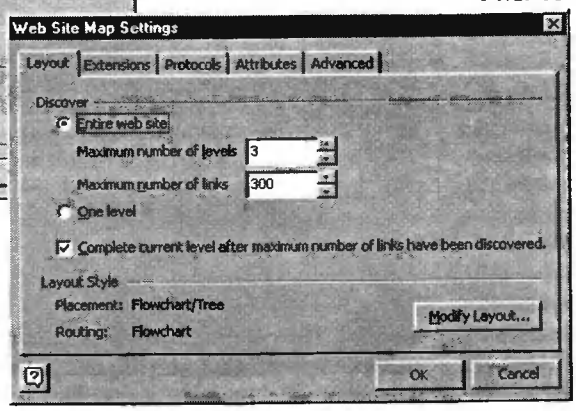


Рис. 9



После выбора **Web Site Map** запускается мастер генерирования карты сайта (**рис.9**). Для увеличения глубины исследуемых ссылок выбирается кнопка **Settings...** (**рис.10**), где указывается глубина ссылок (**Maximum number of levels**) и максимальное количество ссылок (**Maximum number of links**). На других вкладках указываются расширения файлов (**Extensions**), виды протоколов (**Protocols**) и атрибуты файлов (**Attributes**). При необходимости, на вкладке дополнительных настроек (**Advanced**) указывается имя пользователя и пароль для входа на сайт. Для изменения внешнего вида карты сайта (древовидная, радиальная и круговая диаграмма) на первой вкладке (**Layout**) предусмотрена кнопка **Modify Layout**. Затем, в зависимости от скорости соединения, Visio от минуты до получаса занимается сбором информации о сайте. На достаточно серьезном узле можно найти сотни файлов, диаграмма получается довольно громоздкой и часто занимает 5...10 страниц (**рис.11**). Для удобства, символы страниц сопровождаются кнопками со значками "+" и "-" (**рис.12**). Их назначение должно быть понятно всем работающим с Проводником Windows — щелчки по этим кнопкам приводят к сворачиванию-разворачиванию ветви дерева сайта.

Рис. 10



Таким образом, Visio 2002 проявляет себя не только как средство создания диаграмм и блок-схем и не только как средство создания интерактивных приложений. Программа может выступать в качестве средства разработки web-страниц, и более того, для получения информации о web-узлах. Наш обзор возможностей Visio 2002 уже приближается к завершению — в следующей, заключительной статье нам осталось ознакомиться с некоторыми дополнительными приемами работы.

(Окончание следует)

Рис. 11

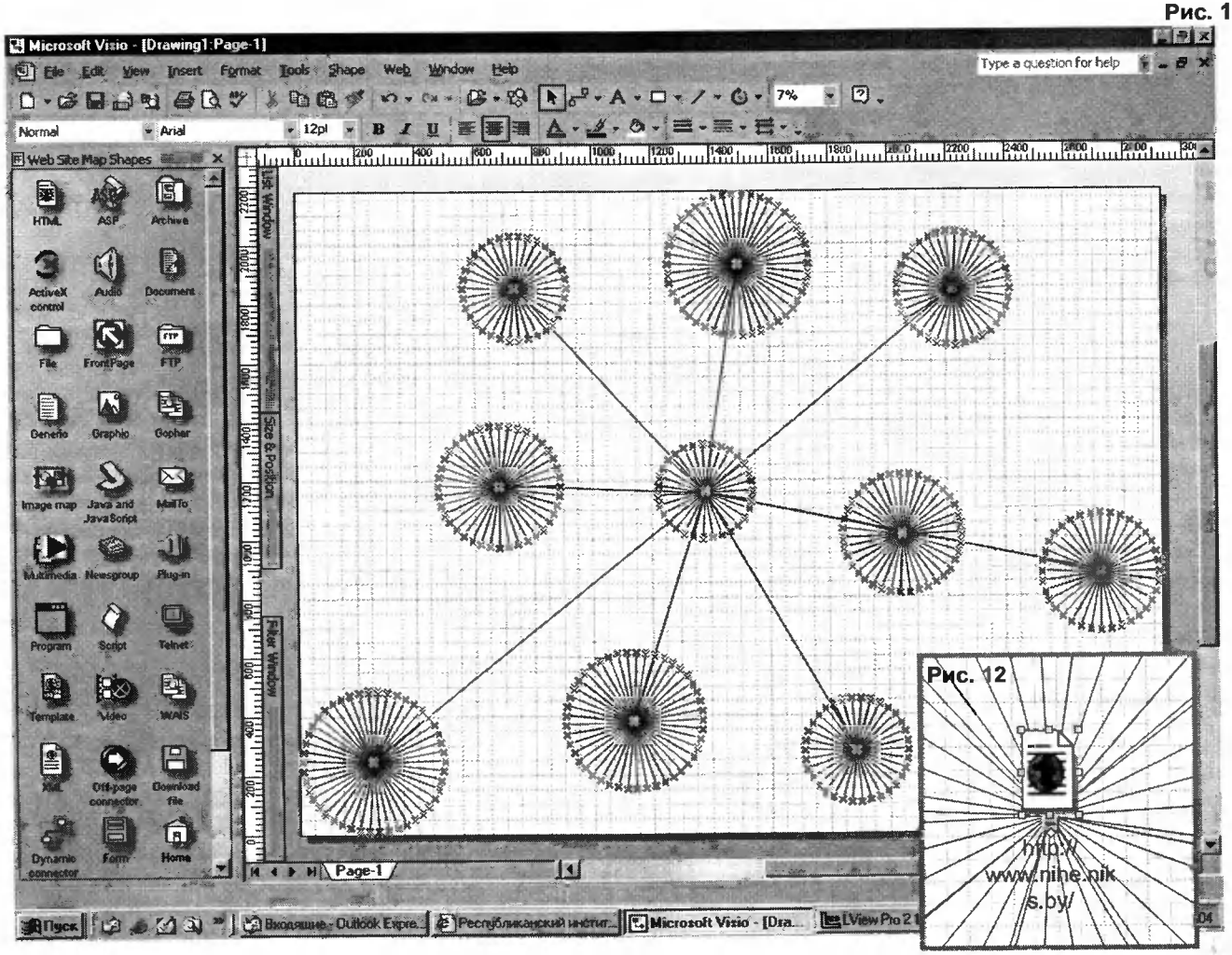


Рис. 12

Создание справочной системы

И.ШИРКО,
г. Минск, лицей №1,
E-mail: FDC@tut.by

Все! Гениальное творение не менее гениального программиста завершено. Программное детище отлажено, проверено и перепроверено. Все найденные друзьями-тестерами "баги" благополучно устранены. А значит, пришло время распространить программу среди пользователей, которые раньше как-то умудрялись без нее обходиться... И вдруг понимаешь, что, несмотря на интуитивно-понятный интерфейс, обязательно найдутся не интуитивно-понятливые "юзеры", которые не смогут насладиться всеми функциями данного произведения искусства, а то и вовсе (о, ужас!) удалят программу БЕЗВОЗВРАТНО. Смахнув со лба капли холодного пота и преисполнившись жалостью к таким "юзерам", всерьез задумываешься о сопроводительной документации. На ум сразу же приходят мысли о файле readme.txt или о небольшой html-страничке, но тут же отметаются — в солидном проекте все должно быть солидно. Так что придется делать общепринятый файл *.hlp, который станет гордо называться "справочной системой". Как создавать файлы такого формата, рассказывается в данной статье.

Прежде всего нам понадобится программа Microsoft Help Workshop. Найти ее можно по адресу http://www.realsoft.ru/lib/ms_help_workshop.zip, также она распространяется вместе с популярными средами программирования (Delphi, VC++). Помимо этой "софтины", нам потребуется текстовый процессор для создания файла в формате RTF (rich-text format), Word от все той же Microsoft вполне подойдет.

Теперь можно приступить к процессу создания справочной системы, который можно разбить на три важные составляющие:

- подготовка RTF-файла;
- создание и компилирование файлов справочной системы;
- создание содержания справки.

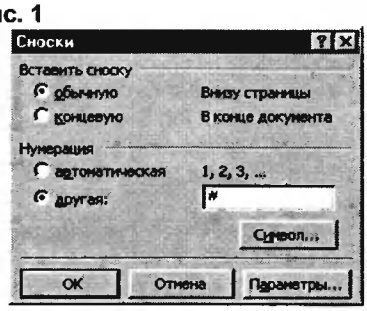
Создание RTF-файла

Подразумевается, что сам текст справочной системы вы уже сочинили, набрали и сохранили как текстовый файл.

Оформление разделов

Как известно, справка обычно разбивается на разделы. В RTF-файле каждый раздел должен начинаться заголовком и заканчиваться символом "разрыв страницы" ("Вставка/Разрыв..."). Кроме того, раздел должен содержать уникальный идентификатор. Для его установки поместите текстовый курсор перед первым символом заголовка, и из меню "Вставка" выберите пункт "Сноска...". В появившемся диалоговом окне (**рис.1**)

Рис. 1



в разделе "Нумерация" установите радиокнопку в положение "Другая" и введите символ диеза ("#"). После нажатия на кнопку "OK", Word предложит ввести текст сноски, что и необходимо сделать. Следует заметить, что если текст сноски начинается с префикса *IDH_*, то во время компиляции справочной системы будет проверена корректность всех ссылок данного раздела.

Ссылки на другие разделы

Для связывания разных разделов используются слова-ссылки, при нажатии на которые осуществляется переход к нужной ветке справки. Для того чтобы сделать слово ссылкой, нужно выделить его и подчеркнуть двойной линией ("Формат/Шрифт/Подчеркивание/Двойное"). После этого сразу за словом-ссылкой требуется поместить идентификатор нужного раздела (текст сноски). При запуске справки ссылка будет выделена цветом и одинарным подчеркиванием.

Комментарии

В документе можно использовать не только ссылки на другие разделы, но и ссылки на комментарии (например, для объяснения какого-нибудь заумного термина). Во время работы справочной системы такие ссылки выделяются цветом и подчеркиванием пунктирной линией, при нажатии на них появляется вплывающее окно с текстом комментария. В RTF-файле комментарии оформляются так же, как и разделы, но они не должны начинаться с заголовка. Ссылку на комментарий нужно подчеркнуть одной линией и сразу за ней написать его идентификатор скрытым текстом.

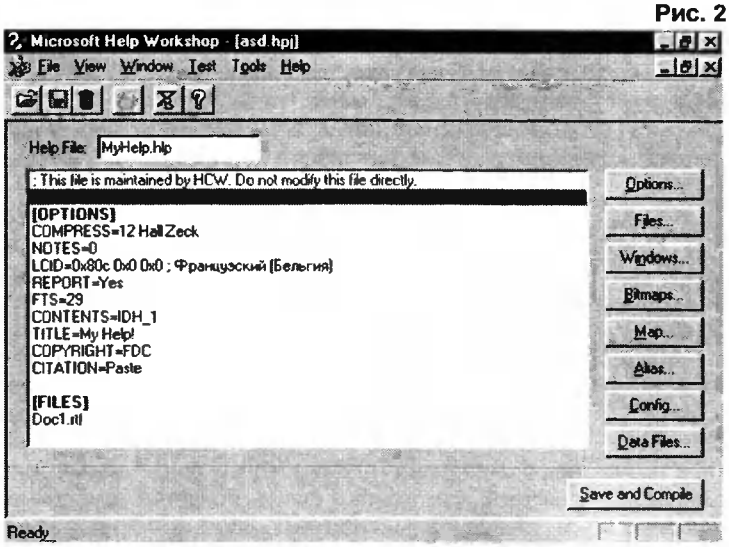
Ключевые слова и поиск по разделам

Для каждого раздела справки можно создать список ключевых слов. Для этого нужно перед заголовком раздела установить сноску *K*, а в текст сноски записать все ключевые слова, разделив их точкой с запятой. При работе справочной системы ключевые слова всех разделов будут отображены в закладке "Указатель". Рядом с ней находится закладка "Поиск", в которой осуществляется поиск по справке. Для того чтобы включить возможность поиска по какому-либо разделу, нужно перед его заголовком поставить сноску *$*, текстом которой должно служить название раздела в поисковой системе.

Вставка графики

Для вставки картинки в раздел, достаточно просто поместить ее туда в Word'е ("Вставка/Рисунок").

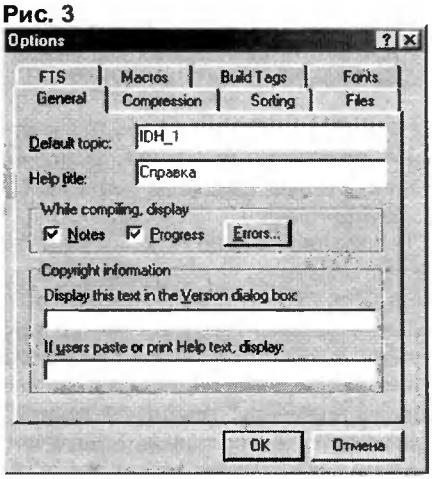
Рис. 2



Создание файла справочной системы

Теперь, когда RTF-файл набран, можно приступить к созданию справки на основе этого файла. Запустите программу Microsoft Help Workshop (**рис.2**) и создайте новый проект, выбрав пункт меню "File/New/Help Project". В правой части окна программы нажмите на кнопку "Files...", и в появившемся диалоговом окне при помощи кнопки "Add" добавьте к проекту RTF-файл. Теперь пройдемся по опциям справочной системы (кнопка "Options...", **рис.3**).

Рис. 3



General

Эта закладка содержит основные настройки справки:

- *Default topic* — идентификатор основного раздела справки. Этот раздел появляется при запуске справочной системы (если она не имеет содержания) и в случае, если пользователь нажал на ссылку на несуществующий раздел;

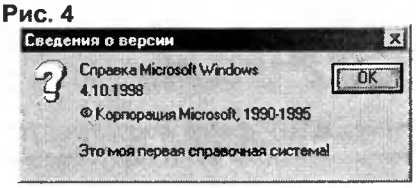
- *Help title* — заголовок справки, который отображается на заголовке окна;

- *Copyright information* — текст, отображаемый в диалоговом окне "Версия" (**рис.4**), и текст, который автоматически добавляется при печати и копировании справки (подпись).

Compression

Здесь находятся настройки сжатия

Рис. 4



справочной системы:

- *None* — не использовать сжатие;

- *Maximum* — максимальное сжатие (файл справки дольше компилируется, но занимает меньше места на диске);

- *Custom* — позволяет выбрать алгоритмы сжатия справочной системы.

Files

Файлы, составляющие справочную систему:

- *Help File* — имя файла справки (*.hlp);

- *Log File* — имя log-файла (отчет о компиляции справочной системы);

- *Rich Text Format (RTF) files* — RTF-файлы (*.rtf);

- *Contents file* — содержание справочной системы (*.cnt).

FTS (full-text search)

Настройки поисковой системы справки:

- *Generate full text search index* — генерировать содержание поиска по справке. Если этот параметр выбран, то при компиляции справочной системы сгенерируется файл *имя_справки.fts*, который нужен для текстового поиска. Как правило, этот файл занимает намного больше места на диске, чем сама справка (*.hlp).

Теперь осталось сформировать содержание для нашей справочной системы.

Создание содержания

Выберите "File/New/Help Contents" (рис.2) и введите имя и заголовок содержания. При помощи кнопок "Add Above" ("Добавить над") и "Add Below" ("Добавить под") создайте нужные папки (Heading) и пункты содержания (Topic). При добавлении пункта, введите его название в поле "Title"; в "Topic ID" — идентификатор раздела справки, на который ссылается этот пункт; в "Help File" — имя файла справки, в котором находится этот раздел. Кнопки "Move Right" и "Move Left" служат для изменения иерархии пунктов. После сохранения содержания, откройте проект справки (*.hpj), нажмите на кнопку "Options", активизируйте закладку "Files", и в поле "Contents file" введите имя файла содержания, либо выберите его при помощи кнопки "Browse". Откомпилируйте проект.

На этом создание полноценной справочной системы завершено. Наш маленький, но очень гордый "хелп" имеет удобное содержание и мощную поисковую систему, содержит как текстовую, так и графическую информацию, и даже подписывается при цитировании и распечатке!

Набор юного хакера "Сломай сам"

Хакерсни — не тормози.
Девиз юных хакеров

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

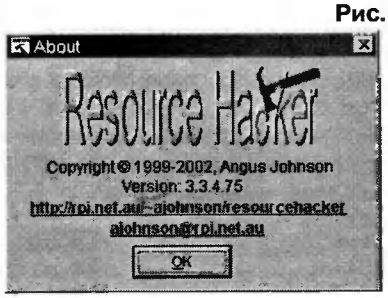
Многие пользователи встречались в своей практике с программами, в которых текст, содержащийся в меню и диалоговых окнах, имел досадные орфографические и грамматические ошибки. Особенно часто это приходится наблюдать при использовании программ, русифицированных непрофессионалами или же не русскоязычными программистами. Некоторые пользователи пытаются исправить эти ошибки, пытаясь в текстовом режиме отредактировать файлы. Но во многих случаях это не приносит желаемого результата.

Для решения подобных проблем на помощь пользователям приходит специальная утилита Resource Hacker (**рис.1**). Сайт программы в Интернете — http://rpi.net.au/~ajohnson/resourcehacker. Разработчик программы — австралийский программист Angus Johnson E-mail: ajohnson@rpi.net.au). Статус программы — freeware. Программа самодостаточна и не требует инсталляции. Размер zip-архива программы составляет 540 Кб. По заявлению разработчика, утилита Resource Hacker нормально функционирует под Windows 9x, Windows NT и Windows XP.

С помощью данной утилиты можно просматривать и редактировать файлы с расширениями ***.exe**, ***.dll**, ***.cpl** и ***.ocx**. Диалоговые окна, меню, типы String Tables и Message Tables, а также формы Borland, содержащиеся в указанных файлах, могут быть отредактированы и перекомпилированы заново. Иконки, битовые изображения, курсоры могут быть полностью извлечены из файлов и сохранены в соответствующих форматах. Интерфейс программы интуитивно понятен, прост и удобен.

Единственным ограничением при использовании утилиты Resource Hacker является невозможность редактирования упакованных и 16-разрядных приложений. Упакованные файлы придется предварительно распаковывать, используя при этом программы-упаковщики, которыми они были упакованы. Но не буду больше утомлять читателей сухой теорией о неограниченных возможностях утилиты Resource Hacker, а лучше перейдем к практическим занятиям.

Рис. 1

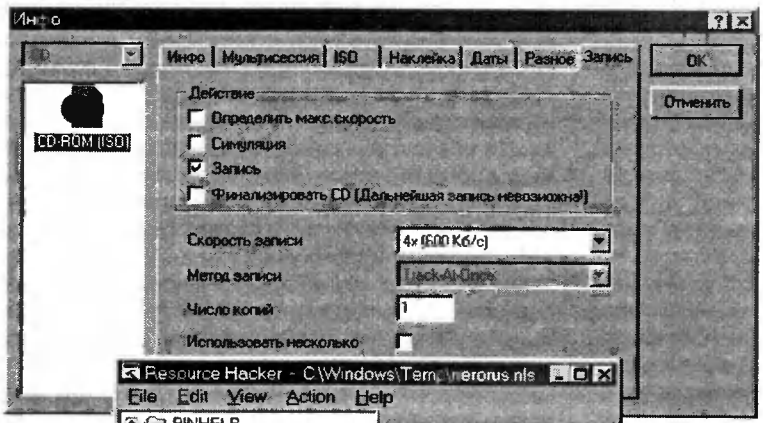


Практическое занятие №1
Тема занятия: Редактирование диалоговых окон

Как я уже упоминал в начале статьи, иногда возникает жгучее желание исправить чужие текстовые ошибки в диалоговых окнах какой-либо программы. Ярким примером этому служит то, как сделана русификация Nero v5.5.9.0. Эта

программа широко применяется пользователями для записи дисков CD-R/RW [1]. Русификация этой программы, мягко скажем, сделана сплошь и рядом с ошибками. Рассмотрим конкретный пример. Загружаем Nero, открываем меню "Файл/Информация проекта", далее переходим на последнюю вкладку "Запись". В опции "Финализировать CD" в глаза бросается ошибка в слове **невозиожна (рис.2)**.

Рис. 2



Начинаем исправление ошибки. Закрываем Nero, ищем файл **NeroRus.nls**, сохраняем оригинал в отдельном месте, а копию файла помещаем, например, в C:\Windows\Temp и загружаем программу Resource Hacker. В меню "File/Open ..." указываем тип файлов "All files" и в списке находим наш файл **NeroRus.nls**. После открытия файла **NeroRus.nls** перед нами в левом окне предстает четкая древовидная структура файла (**рис.3**). Ищем раздел "Dialog", открываем его щелчком по "плюсику" слева от названия. В этом окне откроются все подразделы с диалоговыми окнами. Находим нужное нам диалоговое окно и просматриваем его. Найти нужное диалоговое окно можно быстрее, если применить поиск текста в меню "View/Find Text ...".

Рис. 3

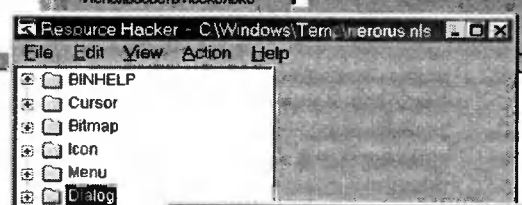
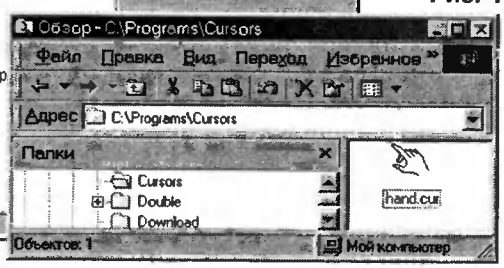


Рис. 4



В правой половине окна будет показано содержимое диалогового окна, а по нажатию на кнопку "Show Dialog" – само диалоговое окно в его графическом представлении. После редактирования текстовой информации нажимаем на кнопку "Compile Script", которая расположена сверху. Затем просматриваем результаты своей работы нажатием на кнопку "Show Dialog" и сохраняем изменения в файле **NeroRus.nls** командой "File"/"Save". Выходим из программы Resource Hacker, помещаем отредактированный файл **NeroRus.nls** обратно в папку программы Nero и наслаждаемся нарезанием болванок.

Совет дня. Чтобы каждый раз не делать одну и ту же работу по редактированию диалоговых окон, после установки программы Nero, а также других программ, желательно сохранить копии отредактированных файлов. Например, поместить их вместе с инсталляционными программами.

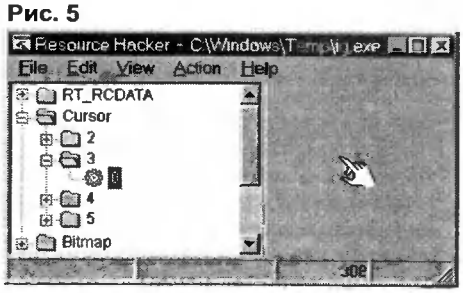
Практическое занятие №2

Тема занятия: Извлечение курсоров

Допустим, вам понравился оригинальный курсор в какой-нибудь игре, и вы хотели бы постоянно применять его вместо стандартных указателей мыши. Как и в первом случае, копируем файл, содержащий курсоры в C:\Windows\Temp. Загружаем Resource Hacker, открываем нужный файл и смотрим раздел "Cursor". В этом разделе будут находиться все программные курсоры, каждый в отдельном подразделе (**рис.4**).

Выбираем понравившийся нам курсор, и в меню "Action" даем команду "Save [Cursor:x]...". В указанном месте сохранится файл с расширением ***.cur (рис.5)**. Далее следуем по маршруту "Пуск/Настройка/Панель управления/Мышь". В окне "Свойства: Мышь" на вкладке "Указатели" нажимаем на кнопку "Обзор..." и применяем извлеченный с помощью программы Resource Hacker курсор.

Рис. 5



Практическое занятие №3

Тема занятия: Извлечение графических объектов

И вот, наконец, мы добрались до самого интересного — извлечения графических объектов. Есть у меня карточная игра "Adult Poker"1999". По правилам этой игры при прохождении каждого уровня на экране показываются не совсем одетые тетеньки из журнала "Playboy", причем каждый раз разные. Но поскольку я слабый игрок в покер, то, соответственно, первый уровень игры я пройти до конца так и не смог. Но посмотреть на девушек из любопытства я был не прочь. С помощью программы Resource Hacker картинки в формате JPEG предстали моему взору во всей своей красе.

Загружаем Resource Hacker, открываем файл **apoker99.exe**. В разделе "Bitmap" я ничего нужного не обнаружил, а вот в разделе "RCData", в подразделах "TFORM" было что-то интересное, но как оттуда все это извлечь, я догадался не сразу, пока в меню "Action" не дал команду "Save [RCData] resource...". Все картинки сохранились в файлах с расширением ***.bin**. Осталось только загрузить просмотрщик графических файлов ACDSee32. В нем обязательно должна быть включена опция "Reading image information". В этом случае браузер ACDSee32 будет прочитывать заголовки всех файлов в директории независимо от их расширения, но выводить листинг только графических файлов, которые можно просмотреть. В данном случае файлы с расширением ***.bin** были опознаны программой ACDSee32 как файлы формата JPEG. Их можно переименовать в файлы с расширением ***.jpg** непосредственно в браузере, используя в контекстном меню команду "Rename series...". Подобным образом можно извлекать графику из любых приложений.

Итак, мы на практике убедились в том, что утилита Resource Hacker действительно обладает широкими функциональными возможностями. Она позволяет редактировать текстовую информацию в диалоговых окнах, извлекать курсоры и графические объекты из приложений, написанных под Windows. Кроме того, программа способна удалять и перезамещать все имеющиеся ресурсы другими аналогичными данными. Я полагаю, что это только часть тех возможностей, которыми обладает эта прекрасная и удивительная утилита Resource Hacker.

Литература

1. Горячкин А. Зачем император Нерон поджег Рим? — Радиомир. Ваш компьютер, 2002, №10, С.6.

ЛИСТАЯ СТРАНИЦЫ

Об особенностях технологии Hyper-Threading, реализованной в процессоре Pentium 4 3,06 ГГц, рассказывает С.*Лахомов* в статье "Новый процессор Intel Pentium 4 с тактовой частотой 3,06 ГГц и поддержкой технологии Hyper-Threading" (**КомпьютерПресс, №12/2002, С.30...34).**

Новый чип — это не просто очередная версия процессора с большей тактовой частотой, а дальнейшее развитие микроархитектуры, позволяющее повысить производительность процессора без увеличения его тактовой частоты, подчеркивает автор. Повышение тактовой частоты позволяет повысить производительность микропроцессора, однако оно не может быть бесконечным. При этом рост производительности не прямо пропорционален росту тактовой частоты, наблюдается некая тенденция насыщаемости, когда дальнейшее увеличение тактовой частоты становится нерентабельным.

Другой способ повышения производительности заключается в увеличении количества исполнительных блоков внутри самого процессора. В этом случае возможно параллельное выполнение нескольких процессорных инструкций одновременно. Такая многозадачность реализована в том или ином виде во всех современных процессорах. Отход от последовательного выполнения команд, использование нескольких исполняющих блоков в одном процессоре позволяют одновременно обрабатывать несколько процессорных микрокоманд, то есть организовывать параллелизм на уровне инструкций, что, естественно, увеличивает общую производительность.

Впрочем, говорить о серьезном увеличении производительности за счет реализации параллелизма на уровне инструкций не приходится. Причин тому несколько. Прежде всего, ошибки в предсказании ветвлений и прерывания вызывают сброс длинного конвейера Pentium 4, что серьезно сказывается на общем быстродействии процессора.

Вторая причина заключается в том, что трудно найти программу, способную одновременно использовать все исполнительные блоки процессора. Pentium 4 имеет в общей сложности семь исполнительных устройств, два из которых могут работать с удвоенной скоростью — две операции за такт. Как правило, программы содержат достаточно часто повторяющиеся команды, то есть либо выполняют в основном целочисленные операции, либо, наоборот, загружают работой устройства для операций с плавающей точкой, поэтому одновременно задействовать все исполнительные блоки процессора просто невозможно.

В современных приложениях в любой момент времени, как правило, выполняется не одна, а несколько задач или несколько потоков одной задачи, называемых также нитями. Если бы оба потока исполнялись изолированно, то скорость выполнения приложения была бы максимальной. При одновременном исполнении обоих потоков процессор будет постоянно переключаться между обоими потоками так, что за один такт процессора выполняются только инструкции какого-либо одного из потоков. Скорость будет относительно невелика.

Самый простой способ повышения производительности заключается в том, чтобы использовать два процессора вместо одного. Такой подход типичен для серверных SMP-конфигураций. В этом случае рост производительности достигается за счет параллельного решения нескольких разных задач или нескольких потоков одной задачи на нескольких процессорах. Впрочем, ожидать пропорционального числу процессоров роста производительности и в данном случае не приходится — многое зависит и от типа решаемых задач, и от того, как реализована в серверной операционной системе поддержка SMP. Всегда можно найти такое приложение, которое в двухпроцессорной конфигурации будет показывать результаты хуже, чем в однопроцессорной. Но самый главный недостаток многопроцессорных систем — это их высокая стоимость.

В то же время, повысить производительность процессора, то есть снизить количество тактов для многопоточной задачи, можно и при использовании одного процессора. Для этого нужно по возможности не только выполнять параллельно независимые инструкции одного потока, но и совместить выполнение инструкций различных потоков. В таком параллельном выполнении двух потоков и заключается основная идея новой технологии Hyper-Threading. Эта технология является промежуточной между многопоточной обработкой, осуществляемой в мультипроцессорных системах, и параллелизмом на уровне инструкций, реализованным в однопроцессорных системах.

Однако и в этом случае есть определенные ограничения. Дело в том, что возможность одновременного выполнения на одном такте процессора инструкций от разных потоков ограничивается тем, что эти инструкции могут задействовать одни и те же исполнительные блоки процессора. В результате возникает конфликтная ситуация, и один из потоков должен ждать освобождения требуемого ресурса процессора.

Итак, от технологии Hyper-Threading можно ожидать максимального выигрыша при условии, что:

- оба потока используют разные блоки процессора;
- отдельные потоки были плохо оптимизированы под процессор Pentium 4;
- выполняемые потоки принадлежат разным приложениям.

С первым условием все понятно, а второе и третье нуждаются в пояснении. Дело в том, что если исполняемое приложение плохо оптимизировано под процессор Pentium 4, то доля использования ресурсов процессора за счет параллелизма на уровне инструкций также невелика. В результате на каждом такте процессора часть ресурсов не задействуется и может быть использована вторым потоком.

При выполнении потоков, принадлежащих различным приложениям, также может наблюдаться значительный прирост производительности процессора, так как в этом случае велика вероятность того, что будут использоваться различные ресурсы процессора.

Естественно, существуют приложения, в которых использование технологии Hyper-Threading не только не приводит к увеличению производительности процессора, но может даже вызывать ее снижение.

С конструктивной точки зрения, процессор с поддержкой технологии Hyper-Threading состоит из двух логических процессоров, каждый из которых имеет свои регистры и контроллер прерываний, то есть две параллельно исполняемые задачи работают со своими собственными независимыми регистрами и прерываниями, но при этом используют одни и те же ресурсы процессора для выполнения задач. После активации каждый из логических процессоров может самостоятельно и независимо от другого выполнять свою задачу, обрабатывать прерывания либо быть заблокированным. Таким образом, от реальной двухпроцессорной конфигурации новая технология отличается только тем, что оба логических процессора используют одни и те же исполняющие ресурсы, одну и ту же разделяемую между двумя потоками кэш-память и одну системную шину. Важно отметить, что никакого специализированного арбитража потоков, способного разделять ресурсы самого процессора между двумя исполняемыми задачами, не существует, однако это не препятствует эффективному использованию ресурсов процессора для каждого независимого потока.

Новый процессор с поддержкой технологии Hyper-Threading требует новых чипсетов, BIOS, материнских плат и даже новых систем охлаждения и, соответственно, корпусов. Поэтому можно говорить не просто о выходе очередной версии процессора, а о смене целой платформы, хотя привычное название Intel Pentium 4 все же остается.

Вместе с новым процессором были объявлены и новые чипсеты: Intel 845PE, Intel 845GE и Intel 845GV. Кроме того, новый процессор будет поддерживаться чипсетом Intel 850E. Все чипсеты рассчитаны на 533-мегагерцовую шину. Чипсеты Intel 845PE, Intel 845GE и Intel 845GV, кроме того, поддерживают память SDRAM DDR333 (PC2700), а чипсет Intel 850E — память RDRAM PC1066.

Оптическую радиомышь компании Defender опробовал *С.Самарин* и рассказал об этом в статье "Defender Wireless Optical 1440 — глазастая мышка без хвоста" **(ПЛ: компьютеры, №11/2002, С.36).**

Мышь поставляется со специальной подставкой, которая является еще и базой. При размещении в базе происходит подзарядка установленных внутри мыши двух Ni-MH аккумуляторов типа AA. Помимо функции зарядного устройства, база играет роль приемника сигналов и подключается к USB-порту или через специвльный переходник к PS/2-порту компьютера. Главной деталью электронного "организма" мыши является миниатюрный радиопередатчик, работающий на частоте 27 МГц. Его мощность невелика и рассчитана на радиус действия 1 м. Если мышке придется трудиться в загруженном офисе рядом с другими беспроводными устройствами, чтобы избежать наводок, производитель предусмотрел разделение частотного канала, а также систему кодирования сигнала. Переключатель канала находится в основании мыши. Подключить мышь к компьютеру так же просто, как и любое другое внешнее устройство. Однако производитель, заботясь о пользователе, вложил в упаковочную коробку небольшую, но подробную инструкцию по установке. Запуск в эксплуатацию прост. Сначала нужно установить внутрь мыши два аккумулятора из комплекта поставки и положить ее на базу, к которой подключается внешний блок питания. Время заряда аккумуляторов, как указано в инструкции, составляет порядка 8 ч. Далее инсталлируется драйвер, база подсоединяется к одному из портов (USB или PS/2) компьютера, и наше устройство автоматически определяется системой. После всех этих шагов остается лишь произвести идентификацию нового устройства нажатием соответствующих кнопок на базе и мыши. Подобную процедуру придется выполнять каждый раз при замене аккумуляторов, либо в случае разрыва связи из-за ухода из зоны действия передатчика.

Работать с мышкой автору понравилось. Благодаря продуманному эргономичному дизайну, рука не устает. Оптический сенсор точно позиционирует указатвль вне зависимости от поверхности, по которой перемещается мышка. Как и у всех беспроводных устройств, предусмотрен спящий режим, который продлевает время автономной работы устройства.

Defender Wireless Optical 1440, по мнению автора, идеально подходит для работы с Интернетом и офисными пакетами — колесо прокрутки, а также пять кнопок придутся как нельзя кстати во многих программах, а невысокая цена (30 USD) наверняка заставит обратить на себя внимание многих потенциальных покупателей, нуждающихся в недорогой, но надежной мышке.

Миниатюрный цифровой видеоплеер будет очень интересен тем, кто много путешествует. О новой разработке фирмы Intel рассказывается в статье "Карманный цифровой видеоплеер — уже не фантастика" **(КомпьютерПресс, №1/2003, С.82...83).**

Портативные цифровые устройства выходят на качественно новую ступень своего развития — недавно корпорация Intel представила прототип персонального видеоплеера (Personal Video Player, PVP), который легко умещается в кармане пиджака и позволяет воспроизводить видеопрограммы, записанные в цифровом формате. Это устройство имеет габариты 12,5x7,5x2,5 см, весит всего 350 г и базируется на высокопроизводительном процессоре с малым энергопотреблением и тактовой частотой 400 МГц, созданном по технологии Intel XScale. Для хранения информации в плеере используется жесткий диск емкостью 20 Гб — этого хватает для записи 70 часов высококачественного видео, а также тысяч песен и фотографий. Изображение выводится на полноцветный ЖК-экран с диагональю 10 см.

Как и многие другие концептуальные разработки корпорации Intel, данный плеер содержит сразу несколько технологических новинок. Специально для этого устройства были разработаны новые алгоритмы сжатия видеоинформации, повторного проигрывания последних кадров и передачи видеофильма с ПК на плеер. Например, разработанный для этого видеоплеера видеокодвк позволяет уменьшить объвм видеофайлов на 40% без видимой потери качества изображения.

В PVP используется большой набор аудио-, видео- и фотокодеков, а также программное обеспечение для воспроизведения аудио-, видеозаписей и фотографий, оптимизированное под процессоры на базе технологии Intel XScale. При помощи высокоскоростного интерфейса USB 2.0 плеер можно легко подключить к персональному компьютеру или к специализированному цифровому устройству для записи телевизионных и видеопрограмм по составленному пользователем расписанию и в течение нескольких минут обновить содержимое его винчестера.

Для обеспечения большей гибкости в PVP предусмотрена возможность беспроводного подключения по протоколам семейства 802.11 (bluetooth). В перспективе это позволит значительно расширить возможности плеера — можно будет закачивать фильмы и видеопрограммы не только из собственного компьютера, но и из общедоступных сетевых ресурсов.

Корпорация Intel не планирует заниматься самостоятельным производством персональных видеоплееров, однако намерена сотрудничать с производителями бытовой электроники, которые заинтересуются данным типом устройств. Вклад Intel в этот проект заключается в оснащении нового устройства важнейшими структурными компонентами, в частности, процессорами на базе технологии XScale и усовершенствованными видеокодеками, разработанными подразделением Intel Emerging Platforms Lab. Помимо этого, Intel предоставит эталонную методику организации производства портативных видеоплееров. Что касается цены на подобные устройства, то ее будут определять сами производители, исходя из конъюнктуры рынка, однако эксперты Intel полагают, что она не превысит 400 USD.

Даже очень хорошая мышь в сочетании с плохим ковриком может доставить много неприятностей в современных динамичных игрушках. **Хороший коврик** будет полезен и при работе с графикой. Поэтому рекомендуем обратить внимание на статью *В.Карапетянца* "Идеальное скольжение" **(CHIP, №12/2002, С.44...47),** посвященную тестированию достаточно дорогих ковриков.

Преимущества таких ковриков становятся очевидными с первых минут работы, отмечает автор. Изделия Everglide Giganta Optical Smoke, Steelpad 3S, ICEMAT (black) и fUnc sUrface 1030 тестировались с помощью оптических мышек Microsoft Explorer 3.0 и Logitech Dual Optical и шариковой мышки Logitech Mini Well Mouse. Для сравнения опробованы также и более дешевые коврики Warner Brothers, "МАКЦЕНТР", гелиевый коврик и Verbatim.

Коврик Everglide Giganta Optical Smoke выпускается известной американской компанией Everglide.com. Его рабочая поверхность сделана из вспененного твердого полимера; он настолько прочный и жесткий, что его практически невозможно сломать или разбить. К столу он крепится пятью резиновыми держателями и совершенно по нему не скользит, как бы вы ни усердствовали.

Первые опыты с использованием оптических мышек не произвели большого впечатления — конечно, разница по сравнению с тетрадью есть, но она, мягко говоря, стоимости коврика не окупает. С шариковой мышью все обстоит более или менее удовлетворительно, хотя этот коврик и создавался специально для оптики.

Все кардинально изменилось при использования специальных наклеек. С на-

клейками мышки Explorer и Dual Optical показали себя во всей красе. Следует также отметить, что продукт компании Everglide.com экономит батарейки беспроводных и совсем не пачкает шариковые мыши. К тому же, это самый долговечный коврик из всех испытанных.

Кстати, о наклейках. Наклейки фирмы Everglide.com дают возможность мыши очень плавно скользить по коврику. Рулон этих наклеек чем-то смахивает на советскую изоленту, но достаточно провести по ней пальцем, и потраченных долларов становится не жалко. Без наклеек ни один дорогой коврик (кроме fUnc) не сможет показать, на что он способен. Поэтому они входят в комплект каждого коврика, свои наклейки есть у ICEMAT и Steelpad. Однако на практике наклейки от Everglide оказываются лучшими даже в сочетании с ковриками других производителей, поэтому они использовались во всех тестах.

Существует еще одно неоспоримое доказательство необходимости этих наклеек. Дело в том, что при использовании дорогих ковриков у мыши довольно быстро стираются ножки, и она приходит в негодность, поэтому наклейки на скользящие поверхности желательно использовать.

Коврик fUnc sUrface 1030 продается в разобранном виде, и собрать его не составляет труда. Коврик двухсторонний, но очень тонкий (примерно 1/3 Giganta). Одна из сторон ворсистая — судя по всему, для шариковой мышки, а вторая — абсолютно гладкая пластмасса — для оптики. Этот коврик, в отличие от Everglide, сразу показал, на что способен. Безукоризненное плавное скольжение, точная передача движений — это плюсы, а минусов как таковых на оптике замечено не было. Все работало безупречно. Шарик тоже скользил хорошо.

К минусам можно отнести то, что коврик слишком тонкий и хрупкий. Создается впечатление, что при небрежном обращении он может довольно скоро потерять внешнюю привлекательность. Кроме того, небольшая толщина кому-то может не понравиться.

Коврик ICEMAT сделан из стекла. По размерам он меньше fUnc и Giganta, но тяжелее. Главный недостаток, заметный сразу — при работе издается просто ужасный скрежет, похожий на звук "ножа по стеклу". Использовать мышь без специальных наклеек смысла не имеет. Причем после пробы оказалось, что наклейки, которые предлагает ICEMAT, ликвидируют этот звук не полностью.

Играется с ковриком очень неплохо, идеально гладкая поверхность гарантирует легкость скольжения. В целом ковер оставляет очень хорошее впечатление, главным и единственным недостатком является неприятный звук, к которому придется привыкать. Не стоит также забывать, что сделан коврик из стекла, пусть даже и толстого, и может разбиться.

Коврик Steelpad 3S сделан из алюминия. Купив его, вы получите также наклейки на мышь, без которых коврик впечатления совсем не производит. Однако и с этим ковром автор рекомендует использовать наклейки от Everglide.com, поскольку они все же лучше. С этими наклейками Steelpad заслуживает всяческих похвал, смотрится он элегантно, служить, по-видимому, будет долго, если, конечно, его не гнуть. Единственное, что портит впечатление, — это аналогичный ICEMAT неприятный скрежет. И еще следует иметь в виду, что Logitech Dual Optical на Steelpad практически не работает.

Коврик от Wamer Brothers — достаточно качественный продукт, который можно купить за сравнительно небольшие деньги. Но движения мышью несколько "суховаты", она скользит недостаточно легко.

Ковер "МАКЦЕНТР" отлично подходит для так называемой "инвалидной" (очень маленькой) чувствительной мыши. При движении коврик создает серьезное сопротивление, что при небольшом давлении на мышь дает превосходный результат. С точки зрения автора, это лучший представитель "тряпки на резине". Но автор не рекомендует использовать его для работы или рисования. Для передвижения мыши по ковру требуются определенные усилия, ввиду чего после нескольких часов рука может серьезно устать. Его основное назначение — игры.

Гелиевые (силиконовые) коврики тоже заслуживают внимания. Некоторые из них стоят по 15...20 USD. Это очень приятные на ощупь, но не слишком удобные коврики. У образца, который использовал автор, в нижней части находится подушечка для запястья, но она жесткая и, что самое главное, очень толстая. Держать мышь (особенно небольшую) крайне неудобно. Катаются мышки тоже так себе. С наклейками ситуация, конечно, меняется в лучшую сторону, но все же автор не рекомендует покупать такой коврик.

Ковриками Verbatim завален весь рынок, и пользователь, ввиду его доступности, все-таки покупает подобный товар. Автор оценивает Verbatim лишь для сравнения, так как описывать его не имеет смысла. Он не обладает ни одним из достоинств описанных выше экземпляров, зато имеет все мыслимые и немыслимые недостатки — плохо держится на столе, сильно пачкает мышку, а оптика на нем постоянно проскальзывает. В общем, это коврик совсем другого уровня, который сделала популярным лишь цена, не превышающая трех долларов.

Подводя итоги, автор отмечает, что ни один коврик из первой четверки, при условии использования с наклейками, его не разочаровал. По качественным показателям он распределяет их в следующем порядке: fUnc sUrface 1030, Everglide Giganta Optical Smoke, Steelpad 3S и ICEMAT (black). С одной стороны, все они достаточно дороги, с другой — предоставляют владельцу качественно новый подход к управлению. Принимая во внимание тот факт что дорогие мыши все более и более востребованы, успех новых ковриков в России — это лишь вопрос времени. Большинство пользователей в конечном счете пересядут за них, так же как некоторое время назад предпочли оптические мыши шарику, ведь к хорошему гораздо легче привыкнуть, чем отказаться от него впоследствии.

В настоящее время идут интенсивные работы по формированию **нового стандарта на сменные оптические носители высокой емкости, которые должны прийти на смену DVD.** Об этом рассказывает *С.Асмаков* в статье "Восход сине-фиолетового диска" **(КомпьютерПресс, №1/2003, С.43...46).**

19 февраля 2002 года компании Hitachi Ltd, LG Electronics Inc., Matsushita Electric Industrial, Pioneer Corporation, Royal Philips Electronics, Samsung Electronics, Sharp Corporation, Sony Corporation и Thomson Multimedia объявили о выпуске базовой спецификации нового типа накопителей на оптических дисках, получивших название Blu-ray Disc. Эта спецификация предусматривает три типа (только для чтения, записываемый и перезаписываемый) односторонних однослойных дисков диаметром 12 см и емкостью 23,3; 25 и 27 Гб. Для записи и чтения дисков данного формата используется сине-фиолетовый лазер с длиной волны 405 нм. Использование лазера с меньшей, чем у применяемой в CD- и DVD-приводах, длиной волны позволило минимизировать рассеяние луча и увеличить числовую апертуру оптической системы до 0,85.

Физическая структура носителей Blu-ray Disc также отличается от их предшественников — толщина прозрачного защитного слоя уменьшена до 0,1 мм, что позволило минимизировать аберрации, вызываемые отклонениями поверхности диска относительно считывающего узла. Ширина дорожки этих носителей составляет всего 0,32 мкм (что вдвое меньше аналогичного параметра DVD-носителей), благодаря чему удалось обеспечить значительно большую емкость новых носителей, а также повысить скорость чтения/записи. Базовая скорость (1x) накопителей Blu-ray Disc составляет 36 Мбит/с (или 5580 Кб/с; для сравнения: у CD этот параметр составляет 150 Кб/с, у DVD — 1385 Кб/с). Для защиты от внешних воздействий диски Blu-ray упаковываются в защитные кар-

триджи размером 129x131x7 мм.

Благодаря использованию записи видеосигнала в формате MPEG-2 Transport Stream, накопители Blu-ray Disc хорошо подходят для записи телевизионных и видеопрограмм, транслируемых в цифровом формате. Емкость диска позволяет хранить более двух часов высококачественного видеосигнала (digital high definition video) и более 13 часов стандартного телевизионного сигнала с шириной потока 3,8 Мбит/с. В перспективе предусматривается создание двухслойных дисков емкостью свыше 50 Гб.

Параллельно с группой компаний, занятых подготовкой спецификации Blu-ray Disc, работу над собственным оптическим носителем, получившим рабочее название Advanced Optical Disk System, вели компании NEC и Toshiba. В накопителях этого формата, как и у Blu-ray Disc, используется сине-фиолетовый лазер с длиной волны 405 нм, однако, в отличие от последнего, применяется оптическая система с числовой апертурой 0.65, а носители имеют 0,6-миллиметровый прозрачный защитный слой — такой же, как и у ныне производимых DVD-носителей. Емкость однослойных односторонних носителей Advanced Optical Disk System составляет 20 Гб для записываемых и перезаписываемых дисков и 15 Гб для ROM-дисков.

Существующая ныне спецификация Advanced Optical Disk System предусматривает возможность создания как однослойных, так и двухслойных односторонних ROM-дисков емкостью соответственно 15 и 30 Гб. Что касается перезаписываемых дисков, то в будущем планируется внедрение односторонних двухслойных носителей емкостью 40 Гб. NEC и Toshiba делают акцент на необходимость обеспечения максимальной совместимости новых продуктов с ныне существующими DVD-носителями.

Недостатком Advanced Optical Disk System следует признать отставание по емкости этих носителей от их главного соперника — Blu-ray Disk. Правда, как утверждают разработчики, этот недостаток компенсируется определенными достоинствами. В частности, носители стандарта Advanced Optical Disk System не требуют использования защитного картриджа, что делает возможным уменьшение габаритов считывающих приводов, а это немаловажно для их использования в портативных устройствах. Кроме того, производство двухслойных дисков с защитным слоем толщиной 0,6 мм проще, чем изготовление носителей с 0,1-мм защитным слоем, и уже обкатано на двухслойных DVD.

Blu-ray Disc на данный момент является весьма раскрученным и быстро развивающимся форматом. Однако подготовка окончательной версии этой спецификации несколько затянулась, а кроме того, до сих пор не подана официальная заявка в основной стандартообразующий орган — DVD Forum — на рассмотрение формата Blu-ray Disc в качестве "преемника" DVD. Между тем, конкуренты в лице NEC и Toshiba оказались гораздо расторопнее, и уже в конце августа представили окончательную версию спецификации Advanced Optical Disk System на рассмотрение DVD Forum. А в начале ноября 2002 г. DVD Forum неожиданно для многих принял решение использовать в качестве базы для нового DVD-стандарта технологию записи, предложенную компаниями NEC и Toshiba. Конечно, весьма сомнительно, чтобы альянс компаний, разрабатывающих и продвигающих Blu-ray Disc, добровольно отказался от дальнейшей борьбы и уступил огромный перспективный сегмент рынка оптических устройств хранения данных своим конкурентам.

Третьим участником предстоящего соревнования может стать еще один формат — High-Definition DVD (HD-DVD), разработкой которого занимается тайваньский консорциум Advanced Optical Storage Research Consortium при поддержке правительственных структур. По предварительной информации, распространенной во второй половине октября 2002 г., в приводах HD-DVD будет использоваться лазер синего диапазона, а емкость носителей составит от 15 до 17 Гб. Одной из причин, побудивших тайваньцев взяться за разработку собственного формата оптических дисков высокой емкости, являются большие лицензионные выплаты, которых требуют держатели патентов за право использования их технологий. Поэтому существует вероятность того, что приводы и носители HD-DVD будут распространяться исключительно на внутренних рынках Китая и Тайваня.

Таким образом, ситуация, когда на рынке представлены накопители нескольких "родственных", но при этом несовместимых между собой форматов (которая наблюдается сейчас в стане записываемых и перезаписываемых DVD), вполне может повториться и на следующей стадии развития сменных оптических носителей. К сожалению, проигравшими в данном случае окажутся пользователи, вынужденные расплачиваться за амбиции лидеров рынка и решать проблемы несовместимости своими силами.

И.Разливин опробовал **последнюю версию ОС Lindows** и рассказал об этом в статье "LindowsOS 2.0: пингвин стучится к вам в окно" **(ПЛ: компьютеры, №11/2002, С.46).**

При создании новой ОС цель ставилась, по мнению автора, грандиозная — совместить надежность Linux с удобством Windows. Причем речь идет не просто о заимствовании отдельных элементов графического интерфейса или организации работы за компьютером в целом, а о полной интеграции любого Windows-приложения с платформой, построенной на ядре Linux.

LindowsOS распространяется исключительно через Интернет. Инсталляция осуществляется либо из-под Windows 98/Me/2000, либо на полностью очищенный жесткий диск, и только в первый логический раздел (впрочем, оставшиеся разделы распознаются вполне корректно). Этот процесс довольно утомителен, однако те, кому удастся установить систему на свой компьютер, увидят более или менее знакомый по Windows "Рабочий стол".

Ряд программ запустился под новой ОС без проблем, однако в целом ситуация с совместимостью оказалась не столь радужной. Гораздо проще перечислить программы, которые работают в новой ОС. Среди них вы точно не увидите популярных игр. Из наиболее известных в число счастливчиков попали IE, Outlook Express, Word, Excel, PageMaker, Acrobat, Mozilla, Netscape и некоторые другие. Судя по всему, пока придется обходиться тем, что руководители проекта называют технологией Click-N-Run. Это, пожалуй, наиболее примечательная особенность новой ОС. Она позволяет буквально одним щелчком мыши скачивать и устанавливать на машину реально работающие приложения (порядка 2000 наименований), включая OpenOffice, отдельные полезные утилиты и простенькие игры.

Итак, какие блага получит пользователь, скачав последний дистрибутив системы (около 350 Мб) — LindowsOS 2.0? По сравнению с первой версией, преобразился графический интерфейс. На смену грубым квадратным формам пришли аккуратные иконки и темы для Рабочего стола, более продуманной оказалась и организация различных меню. В панель управления добавлена функция смены разрешения монитора. Оптимизирован процесс загрузки приложений по технологии Click-N-Run. LindowsOS 2.0 поставляется с модернизированным почтовым клиентом и браузером, снабженным функцией блокировки всплывающих баннеров и окон. Проблемы с запуском Windows-приложений остались прежними.

Прочие нововведения касаются поддержки компонентов аппаратного обеспечения и работы в Сети. В результате тесного сотрудничества с Hewlett-Packard, LindowsOS научилась распознавать около 800 принтеров различных моделей. Кроме того, в ней реализована функция интеграции с сетями, построенными на Windows-платформе.

По всей видимости, пройдет еще немало времени, прежде чем станет ясно, насколько успешным оказался проект LindowsOS, заключает автор.

Олимпиадные задачи

Уважаемые читатели, как было обещано, предлагаем вашему вниманию решения задач Гомельской городской олимпиады школьников по информатике 2002 г. (условия задач — в РМ. ВК, №4/2003), предложенные их авторами. Данный материал любезно предоставлен М.С.Долинским (г.Гомель).

5...8 классы

Авторы задач — Владимир Миняйлов и Сергей Вераксич, ученики 7-го класса СШ №27 г.Гомеля.

Задача 1. "Лимоны"

Суммируем массив, состоящий из количества сока, который можно выдавить из лимонов, и сравниваем полученное значение с нужным количеством.

Полный текст решения:

```
program lemon;
var
  z,n,k,i,sm:longint;
begin
  sm:=0;
  readln(N,k);
  for i:=1 to N do
    begin readln(z); inc(sm,z); end;
  if sm>=k
    then writeln('Yes') else writeln('No');
end.
```

Задача 2. "Марс"

Считываем из входного файла оценки каждого из учеников и проверяем, хорошая ли эта оценка. Если хорошая, то увеличиваем результат на 1.

Полный текст решения:

```
program mars;
var
  x,y,n,i,cl,zz : longint;
begin
  readln(n,x,y);
  cl:=0;
  for i:=1 to N do
    begin
      readln(zz);
      if (zz>x) and (zz<y) then inc(cl);
    end;
  writeln(cl);
end.
```

Задача 3. "Programming"

Для каждого слова исходного текста программы находим количество каждой из букв и проверяем, чтобы количество каждой из гласных букв совпадало с количеством букв в программе на "гласной" версии языка. Аналогично поступаем и с согласными буквами.

Полный текст решения:

```
program p;
const
  glasn : set of char=['a','e','i','j','o','u','q'];
type
  tabc = array [0..255] of byte;

  function cmp(a,b:tabc):boolean;
    var
      i : word;
    begin
      for i:=0 to 255 do
       if a[i]<>b[i]
         then begin cmp:=false; exit; end;
      cmp:=true;
    end;

var
  a,b     : array[1..100] of tabc;
  z       : tabc;
  n       : word;
  i,j     : word;
  flg     : boolean;
  s,s1,s2 : string;
begin
  readln(N);
  for i:=1 to n do
    begin
      readln(s);
      s1:=''; s2:='';
      for j:=1 to length(s) do
        if s[j] in glasn
          then inc(a[i,byte(s[j])])
          else inc(b[i,byte(s[j])]);
    end;
  flg:=true;
  for i:=1 to n do
    begin
      readln(s);
      fillchar(z,sizeof(Z),0);
      for j:=1 to Length(s) do inc(z[byte(s[j])]);
      if not cmp (b[i],z) then flg:=false;
    end;
  if flg then writeln('Yes') else writeln('No');
  readln;
  flg:=true;
  for i:=1 to n do
    begin
      readln(s);
      fillchar(z,sizeof(Z),0);
      for j:=1 to Length(s) do inc(z[byte(s[j])]);
      if not cmp(a[i],z) then flg:=false;
    end;
  if flg then writeln('Yes') else writeln('No');
end.
```

Задача 4. "Эпидемия"

Для каждой пары городов проверяем, чтобы расстояние от одного города до первого было меньше радиуса заражения, а от другого до первого — больше. Затем, если значение расстояния между этими двумя городами меньше минимального, то оно принимается за минимальное. Первоначальное значение минимального расстояния следует инициализировать заведомо большим числом.

Полный текст решения:

```
Program epidemy;
  function rast(x1,y1,x2,y2:real):real;
    begin
      rast:=sqrt(sqr(x1-x2)+sqr(y1-y2));
    end;
var
  a         : array[1..1000,1..2] of real;
  b         : array[1..1000] of boolean;
  n,k       : word;
  min       : real;
  i,j,mi,mj : word;
begin
  assign(input,'input.txt');reset(input);
  assign(output,'output.txt');rewrite(output);
  readln(n,k);
  for i:=1 to N do readln(a[i,1],a[i,2]);
  b[1]:=true;
  for i:=2 to N do
    if rast(a[1,1],a[1,2],a[i,1],a[i,2])<=k then b[i]:=true;
  min:=1.7e38;
  for i:=1 to N do
    for j:=1 to N do
      if b[i] and (not b[j]) and
         (rast(a[i,1],a[i,2],a[j,1],a[j,2])<min)
        then
          begin
            min:=rast(a[i,1],a[i,2],a[j,1],a[j,2]);
```

```
            mi:=i;
            mj:=j;
          end;
  writeln(mi,' ',mj);
  close(output);
end.
```

Задача 5. "Лабиринт"

Это стандартная задача на очередь.

Полный текст решения:

```
program labyrint;
{$A+,B-,D+,E+,F-,G-,I+,L+,N-,O-,P-,Q+,R+,S+,T-,V+,X+}
{$M 16384,0,655360}
var
  a              : array[1..100,1..100] of byte;
  ocr            : array[1..15000,1..2] of byte;
  n,m,x,y,x1,y1  : longint;
  pw,pr          : word;

  procedure add(x,y:word);
    begin
      ocr[pw,1]:=x;
      ocr[pw,2]:=y;
      inc(pw);if pw>10000 then pw:=1;
    end;
  procedure Get(var x,y:word);
    begin
      x:=ocr[pr,1];
      y:=ocr[pr,2];
      inc(pr);if pr>10000 then pr:=1;
    end;
var
  i,j   : longint;
  tx,ty : word;
begin
  pw:=1;
  pr:=1;
  readln(n,m);
  readln(x,y);
  readln(x1,y1);
  for i:=1 to N do
    for j:=1 to N do read(a[i,j]);
  add(x,y);
  while pw<>pr do
    begin
      get(tx,ty);
      if (tx=x1) and (ty=y1)
        then begin writeln("Yes"); halt; end;
      if a[tx,ty]=1 then continue;
      a[tx,ty]:=1;
      if tx>1 then if a[tx-1,ty]=0 then add(tx-1,ty);
      if tx<m then if a[tx+1,ty]=0 then add(tx+1,ty);
      if ty>1 then if a[tx,ty-1]=0 then add(tx,ty-1);
      if ty<n then if a[tx,ty+1]=0 then add(tx,ty+1);
    end;
  writeln('No');
end.
```

Задача 6. "Уравнение"

Главным при решении этой задачи было догадаться, что минимальный из корней будет принимать максимальное значение, если все корни равны между собой. Для решения задачи суммируем коэффициенты и делим на полученную величину число M:

```
program equation;
var
  N       : word;
  i       : word;
  sum,z,k : extended;
begin
  readln(N,k);
  sum:=0;
  for i:=1 to N do
    begin
      read(z);
      sum:=sum+z;
    end;
  writeln((k/sum):0:6);
end.
```

9...11 классы

Задача 1. "Клетки"

Автор задачи — Иван Зайцев, студент 2-го курса математического факультета ГГУ им.Ф.Скорины.

1. Периметр заданной фигуры считается по ходу считывания входных данных.

2. Площадь заданной фигуры можно было считать алгоритмом нахождения площади произвольной фигуры, но можно и немного проще:

```
Координаты увеличиваем на 20000 (чтобы они были положительными).
В случае шага влево мы вычитаем столбик высотой Y.
В случае шага вправо прибавляем столбик высотой Y.
В завершение берем полученную величину по модулю, и в итоге
получаем площадь фигуры.
```

3. Основную проблему составляет вычисление количества сторон клеток, лежащих внутри фигуры. Легко заметить, что имеет место соотношение:

P+2*Q=4*S (здесь P — периметр, Q — количество клеток, S — площадь).

Действительно, данная фигура содержит S единичных квадратиков, которые имеют 4*s сторон. Но стороны, лежащие внутри фигуры, мы учитываем два раза, а на периметре — один. Отсюда и получаем — P+2Q=4S.

Полный текст решения:

```
program cell;
Var
  i,j,x,y : longint;
  p,q,s   : longint;
  d       : char;
  r       : real;
Begin
  assign(input,'input.txt');reset(input);
  assign(output,'output.txt');rewrite(output);
  readln(x,y);
  x:=x+20000;y:=y+20000;
  while not(eof) do
    begin
      read(d);
      if d='u' then begin inc(p); inc(y); end else
      if d='d' then begin inc(p); dec(y); end else
      if d='l' then begin inc(p); s:=s-y; dec(x); end else
      if d='r' then begin inc(p); s:=s+y; inc(x); end
    end;
  s:=abs(s);
  r:=(4.0*s-p)/2;
  q:=trunc(r);
  writeln(P,' ',S,' ',Q);
  close(output);
End.
```

Задача 2. "Автоугон"

Автор задачи — Михаил Сваричевский, студент 2-го курса математического факультета ГГУ им.Ф.Скорины.

Находим кратчайший путь Пети до песочницы, и значения времени, в которые он появится на каждом перекрестке. Значения времени храним в вещественном типе.

Затем для каждого полицейского находим кратчайшие расстояния до всех вершин и значения времени прибытия на каждый перекресток. Остается только проверить перекрестки Петиного пути в порядке их проезда Петей на то, чтобы там уже не было полицейского или он еще не подъезжал. Если ничего не нашли — выводим 1, иначе — номер вершины, в которой Петю задержат.

Полный текст решения:

```
Program auto;
{$A+,B-,D+,E+,F-,G+,I+,L+,N+,O-,P-,Q+,R+,S+,T-,V+,X+,Y+}
{$M 16384,0,655360}
const
  min : real = 0.1e-11;
var
  Sity,SityD            : array [0..99,0..99] of byte;
```


```
V                    : array [0..99] of integer;
Where                : array [0..99] of integer;
Pred,Usd,Dist        : array [0..99] of integer;
ArriveTime           : array [0..99] of real;
Time                 : real;
N,M,K,VB,i,j,l,out   : integer;

procedure InputData;
  begin
    assign(input,'input.txt'); reset (input);
    readln(N,M,K);
    read  (VB);
    for i:=0 to K-1 do read(V[i]);
    for i:=0 to M-1 do
      begin
        read(j,l);
        Sity[j-1,l-1]:=1;
        Sity[l-1,j-1]:=1;
      end;
    for i:=0 to K-1 do read(Where[i]);
    close(input);
  end;

procedure FloidAlgoritm;
  begin
    move(Sity,SityD,sizeof(Sity));
    for i:=0 to N-1 do
      for j:=0 to N-1 do
        if SityD[i,j]=0 then SityD[i,j]:=255;

    for l:=0 to N-1 do
      for i:=0 to N-1 do
        for j:=0 to N-1 do
          if SityD[i,j]>SityD[i,l]+SityD[l,j]
            then SityD[i,j]:=SityD[i,l]+SityD[l,j];
end;

procedure MakePoliceArriveTime;
  procedure Make;
    begin
      time:=SityD[Where[i]-1,j]/V[i];
      if (time<ArriveTime[j]) then ArriveTime[j]:=time;
    end;
  begin
    for i:=0 to N-1 do ArriveTime[i]:=1.7e38;
    for i:=0 to K-1 do
      begin
        l:=Where[i];
        ArriveTime[l-1]:=0.0;
        for j:=0 to l-1 do Make;
        for j:=l to N-1 do Make;
      end;
  end;

procedure DeicstraAlgoritm;
  var
    Cur : integer;
  begin
    for i:=0 to N-1 do
      begin
        Usd[i]:=0; Dist[i]:=MaxInt;
      end;
    Dist[0]:=0;
    Cur:=0;
    Pred[0]:=-1;
    while cur>=0 do
      begin
        Usd[cur]:=1;
        for i:=0 to Cur-1 do
          if Sity[Cur,i]<>0 then
            if Dist[i]>Dist[Cur]+1 then
              begin
                Dist[i]:=Dist[Cur]+1;
                Pred[i]:=Cur;
                Usd[i]:=0;
              end;
        for i:=Cur+1 to N-1 do
          if Sity[Cur,i]<>0 then
            if Dist[i]>Dist[Cur]+1 then
              begin
                Dist[i]:=Dist[Cur]+1;
                Pred[i]:=Cur;
                Usd[i]:=0;
              end;
        l:=-1;
        j:=MaxInt;
        for i:=0 to N-1 do
          if (Usd[i]=0) and (Dist[i]<j) then
            begin
              j:=Dist[i]; l:=i;
            end;
        Cur:=l;
      end;
  end;

  procedure MakePetiaArriveTime;
    begin
      out:=-1;
      l:=Dist[N-1];
      j:=N-1;
      while j>=0 do
        begin
          time:=l/VB;
          if (time > ArriveTime[j]) or
             (abs(time-ArriveTime[j])<min) then out:=j+1;
          j:=Pred[j];
          if j>=0 then l:=Dist[j];
        end;
    end;

  procedure OutputData;
    begin
      assign (output,'output.txt'); rewrite(output);
      writeln(out);
      close  (output);
    end;

begin
  InputData;
  FloidAlgoritm;
  MakePoliceArriveTime;
  DeicstraAlgoritm;
  MakePetiaArriveTime;
  OutputData;
end.
```

Задача 3. "Кодирование"

Автор задачи — Павел Сухорученко, студент 2-го курса математического факультета ГГУ им.Ф.Скорины.

Задача решается реализацией алгоритма, описанного в условиях задачи.

Полный текст решения:

```
program coding;
Var
  codel,freq    : array[#0..#255] of longint;
  code          : array[#0..#255,1..32] of byte;
  b             : array[0..225] of record
                          k : byte;
                          a : array[0..225] of char;
                          f : longint;
                          end;
  i,j           : integer;
  ch            : char;
  st            : string;
  n,m,t,k,min1,min2,nm1,nm2 : longint;
Begin
  assign(input,'input.txt');reset(input);
  assign(output,'output.txt');rewrite(output);
  while not(eof)do
    begin
      read(ch);
      if (ch<>#10)and(ch<>#13) then (freq[ch]);
    end;

  for ch:=#0 to #255 do
    if freq[ch]>0
      then
        begin
```

```
          inc(k);
          b[k].a[1]:=ch;
          b[k].f:=freq[ch];
          b[k].k:=1;
        end;
  if k>1
    then
      while k>1 do
        begin
          min1:=1000000;min2:=1000000;
          for j:=1 to k do
            begin
              if b[j].f<min1
                then
                  begin
                    min2:=min1;
                    nm2:=nm1;
                    min1:=b[j].f;
                    nm1:=j;
                  end
                else
                  if b[j].f<min2
                    then
                      begin
                        min2:=b[j].f;
                        nm2:=j;
                      end;
            end;
          for j:=1 to b[nm1].k do
            begin
              inc(codel[b[nm1].a[j]]);
              code[b[nm1].a[j],codel[b[nm1].a[j]]]:=0;
            end;
          for j:=1 to b[nm2].k do
            begin
              inc(codel[b[nm2].a[j]]);
              code[b[nm2].a[j],codel[b[nm2].a[j]]]:=1;
            end;
          if nm1<nm2
            then
              begin
                 for j:=1 to b[nm2].k do
                b[nm1].a[j+b[nm1].k]:=b[nm2].a[j];
                 b[nm1].f:=b[nm1].f+b[nm2].f;
                 b[nm1].k:=b[nm1].k+b[nm2].k;
                 for j:=nm2+1 to k do b[j-1]:=b[j];
                 dec(k);
              end
            else
              begin
                 for j:=1 to b[nm1].k do
                b[nm2].a[j+b[nm2].k]:=b[nm1].a[j];
                 b[nm2].f:=b[nm2].f+b[nm1].f;
                 b[nm2].k:=b[nm2].k+b[nm1].k;
                 for j:=nm1+1 to k do b[j-1]:=b[j];
                 dec(k);
              end
        end
    else
      for ch:=#0 to #255 do
       if freq[ch]>0 then begin codel[ch]:=1; break; end;
  reset(input);
  for ch:=#0 to #255 do
    begin
      if codel[ch]>0
        then
          begin
            write(ch,' ');
            for i:=codel[ch] downto 1 do write(code[ch][i]);
            writeln
          end;
    end;
  while not(eof) do
    begin
      read(ch);
      if (ch<>#10)and(ch<>#13)
        then for j:=codel[ch] downto 1 do write(code[ch][j]);
    end;
  close(output);
End.
```

А.КОРБИТ, А.ЩЕРБАКОВ,
г.Минск, БГУИР.

Программирование в среде Visual C++ 6.0

Программа "Калькулятор"

Цель этой статьи — изучить механизмы обмена и проверки данных на базе каркаса MFC и написать программу, реализующую создание и функционирование простейшего калькулятора.

1.Механизмы обмена и проверки данных диалога

В предыдушей статье [1] в коде обработчика события (щелчок на кнопке "Выполнить") вызывается метод библиотеки MFC UpdateData, который организует обмен данными между приложением и элементами управления. Рассмотрим этот механизм более подробно.

Механизм обмена данными диалога (DDX — Dialog Data eXchange) представляет собой простой способ инициализации элементов управления диалогового окна и получения приложением вводимых пользователем данных, с одной стороны, и их отображения в элементах управления как результатов работы приложения, с другой. Механизм проверки данных диалога (DDV — Dialog Data Validation) обеспечивает простой способ контроля данных, вводимых/выводимых в диалоговом окне.

Для использования механизма обмена и проверки данных нужно при помощи ClassWizard связать элементы управления с переменными-членами класса. Обмен данными между элементами управления и приложением осуществляет функция класса CWnd: DoDataExchange. Переопределяемая версия этой функции автоматически формируется средствами AppWizard при создании класса, производного от CDialog. Если связать элементы управления с переменными-членами класса, то ClassWizard автоматически добавляет в нее обращения к DDX-функциям.

Имеется несколько DDX-функций, соответствующих разным типам элементов управления. Наиболее часто используются две из них:

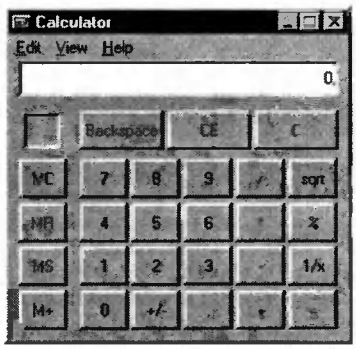
- DDX_Text() — для обмена данными между полем ввода/вывода и переменной-членом типа CString или числового типа;
- DDX_Check() — для установки или передачи состояния флажка в переменную-член типа BOOL.

В MFC также имеется несколько DDV-функций, соответствующих различным способам проверки, применяемых к строковым или числовым переменным, например, DDV_MaxChars().

К функции DoDataExchange() напрямую не обращаются. Вызов функции DoDataExchange() осуществляется функцией класса CWnd: UpdateData. Единственный параметр функции UpdateData имеет тип BOOL, который определяет направления передачи данных. Если этот параметр имеет значение TRUE, то передача происходит от элементов управления в переменные, если параметр установлен в FALSE, то передача данных идет в обратном направлении.

2. Пример написания программы

Требуется создать простой калькулятор наподобие стандартного калькулятора, приведенного на **рисунке**



(ввод первого операнда, ввод знака арифметической операции, ввод второго операнда и щелчок на кнопке "="). Рассмотрим алгоритм действий на примере калькулятора с кнопками "1", "2", "+", "-", "=" и "C".

Для решения задачи проделаем следующие действия.

1. Создайте при помощи MFCAppWizard(exe) новый проект типа Dialog based (предварительно выбрав путь и имя — например, Lab2).

Используя панель конструктора ресурсов "Controls", нанесите на Dialog based один элемент типа EditBox и шесть кнопок типа Button, установите нужные свойства и надписи. Для этого нужно сделать щелчок на соответствующем элементе управления, выбрать пункт меню Properties, в появившейся закладке General выбрать поле Caption и ввести его "название"("1", "2", "+", "-", "=", "C"). Затем в закладке Styles: установить нужные свойства, в свойствах диалоговой панели активизировать переключатели Minimize и Maximize Box, а в выпадающем меню Border выбрать пункт Resizing, т.е. включить возможность изменения размеров будущего калькулятора.

Поле Edit связываем с переменной m_L. Для этого необходимо сделать щелчок правой клавишей мыши в любом месте диалоговой панели, в меню выбирать ClassWizard, далее — закладку Member Variables. Затем делаем щелчок на IDC_EDIT1, щелчок на Add Variable..., в появившейся закладке в поле name устанавливаем m_L , в поле type — CString. И — завершающий щелчок на OK.

2. Теперь создадим код обработчика для кнопки "1". Делаем щелчок правой клавишей мыши в любом месте диалоговой панели, выбираем ClassWizard, закладку — Message Maps, идентификатор (Object Ids) — IDC_Button1, событие — BN_CLICKED. Затем последовательно выполняем щелчки на Add Function, OK, Edit Code и вводим код обработчика:

```
void CLab2Dlg::OnButton1()
{
        m_L+="1";
        UpdateData(FALSE);

}
```

Аналогично создаются обработчики для других кнопок.

Код обработчика щелчка на кнопке "2":

```
void CLab2Dlg::OnButton2()
{
        m_L+="2";
        UpdateData(FALSE);

}
```

Теперь для левого и правого операндов добавим в проект две вещественные переменные — reg_x и reg_y. Для этого в окне WorkSpace выбираем закладку ClassView, делаем щелчок правой кнопкой мыши на CLab2Dlg, выбираем пункт Add Member, в появившемся окне в поле Variable Type выбираем double, а в поле Name вводим reg_x. Завершает все щелчок на OK.

Аналогично в класс CLab2Dlg добавляем reg_y.

3. Создаем коды для кнопок "+" и "-". После создания обработчиков по методике пункта 2 вводим следующий код:

```
void CLab2Dlg::OnButton3()
{
        UpdateData(TRUE);
        reg_x=atof(m_L);
        m_L="";
        UpdateData(FALSE);
        code='+';
}
void CLab2Dlg::OnButton4()
{
        UpdateData(TRUE);
        reg_x=atof(m_L);
        m_L="";
        UpdateData(FALSE);
        code='-';
}
```

Здесь перед обработчиками необходимо объявить переменную code символьного типа, которая будет использована для выбора нужного действия над операндами. В кодах обработчиков, при присваивании программной переменной значения операнда из поля ввода/вывода, необходимо воспользоваться функцией atof, которая преобразует значение типа String в значение типа double. Также следует добавить в код файла подключение заголовочного файла **math.h**.

4. Теперь добавляем обработчик для кнопки "=":

```
void CLab2Dlg::OnButton5()
{
        UpdateData(TRUE);
        reg_y=atof(m_L);
        switch (code)
        {
                case '+': rez=reg_x+reg_y; break;
                case '-': rez=reg_x-reg_y; break;
        }
        m_L.Format("%lf",rez);
      UpdateData(FALSE);
}
```

Здесь переменная **rez** добавлена к классу таким же способом, как и переменные **reg_x** и **reg_y**.

5. И, наконец, добавим обработчик кнопки C(lear):

```
void CLab2Dlg::OnButton6()
{
        m_L="";
        UpdateData(FALSE);
}
```

3. Индивидуальные задания

Используя рассмотренные методы, согласно заданию создайте действующий калькулятор, предусмотрев при этом блокировку ошибочных действий:

- введение символов "не цифр";
- деление на ноль;
- дублирование десятичной точки и т.д.

1. Написать программу "Калькулятор", реализующую пять основных арифметических операций — "+", "-", "*", "/" и "%" (вычисление остатка от деления). Запись операндов должна производиться в алгебраической форме — ввод операнда 1, выбор операции, ввод операнда 2.

2. Написать программу "Калькулятор", реализующую пять основных арифметических операций — "+", "-", "*", "/" и "%" (вычисление остатка от деления). Запись операндов должна производиться в польской форме — ввод операнда 1, ввод операнда 2, выбор операции.

3. Написать программу "Калькулятор", реализующую вычисление четырех основных тригонометрических функций — sin(x), cos(x), tg(x), ctg(x). Для выбора соответствующей функции необходимо использовать элемент управления Radio Button.

4. Написать программу "Калькулятор", реализующую вычисление четырех основных тригонометрических функций — sin(x), cos(x), tg(x), ctg(x). Для выбора соответствующей

функции необходимо использовать элемент управления Button. Аргумент X вводится в градусах или в радианах. Автоматический перевод значения угла в радианы должен устанавливаться элементом управления CheckBox.

Литература

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

Решение задач на графах построением минимального остовного дерева

М.ДОЛИНСКИЙ, г.Гомель.

Введение

Ранее в [1] уже рассматривались следующие способы решения задач на графах:

- метод Флойда для поиска кратчайших расстояний между всеми парами вершин графа (одновременно можно построить и сами кратчайшие пути от каждой вершины до каждой), построения матрицы достижимости графа и построения множества истоков и множества стоков (исток — вершина, в которую нет входящих дуг; сток — вершина, из которой нет исходящих дуг);
- метод Дейкстра для поиска кратчайших расстояний от одной вершины графа до всех остальных и построения оптимальных маршрутов от одной вершины до всех остальных;
- поиск в глубину для: решения произвольных задач на графах, требующих соответствующего порядка обхода графа "в глубину" (раскрашивая пройденные вершины и/или дуги можно управлять порядком и способами обхода, например, применять однократное или многократное использование дуг и вершин); построения множества истоков и стоков (как истоков на транспонированном графе); построения сильносвязанных компонент (ССК) графа (ССК — это множество вершин, каждая из которых достижима из всех вершин ССК);
- поиск в ширину для решения произвольных задач на графах, требующих соответствующего порядка обхода графа ("в ширину").

В данной статье рассматривается еще один способ решения задач на графах — построение минимального остовного дерева. Далее приводятся условия реальных олимпиадных задач, которые сводятся к нахождению минимального остовного дерева, описываются два алгоритма построения минимального остовного дерева — алгоритм Прима и алгоритм Крускала, и показывается их применение для решения приведенных задач.

Прежде всего, вспомним некоторые определения теории графов [1].

Неформально граф можно определить как набор вершин (города, перекрестки, компьютеры, буквы, цифры кости домино, микросхемы, люди) и связей между ними (дороги между городами; улицы между перекрестками; проводные линии связи между компьютерами; слова, начинающиеся на одну букву и закачивающиеся на другую или эту же букву; проводники, соединяющие микросхемы; родственные отношения, например, "Алексей — сын Петра"). Двунаправленные связи, например, дороги с двусторонним движением, принято называть ребрами графа. Однонаправленные связи, например, дороги с односторонним движением, принято называть дугами графа. Граф, в котором вершины соединяются ребрами, принято называть неориентированным графом. Граф, в котором хотя бы некоторые вершины соединяются дугами, принято называть ориентированным графом.

Графы могут задаваться матрицей смежности. То есть G[i,j]=1, если есть дуга из вершины i в вершину j, и G[i,j]=0 в противном случае. Часто также представляет интерес и матрица достижимости графа, в которой G[i,j]=1, если существует путь по ребрам (дугам) исходного графа из вершины i в вершину j.

Граф называется взвешенным, если для его дуг вводятся веса, например, длины дорог между городами. Во взвешенном графе G[i,j] — вес дуги от вершины i к вершине j.

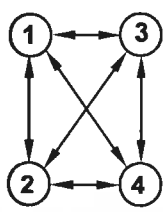
1. Простейшие примеры задач о минимальном остовном дереве

Взвешенный граф служит математической моделью широкого класса задач на нахождение минимального остовного дерева. Формально, минимальное остовное дерево определяется следующим образом.

Пусть имеется связанный неориентированный граф G=(V,E), где V — множество его вершин, а E — множество его ребер. И пусть для каждого ребра графа (u,v), соединяющего вершины u и v, задан его неотрицательный вес P(u,v). Задача нахождения минимального остовного дерева заключается в нахождении такого подмножества T ребер исходного графа, которое связывает все вершины графа G, и для которого суммарный вес всех ребер минимален.

Для примера рассмотрим граф с четырьмя вершинами, которые пронумерованы от 1 до 4 (**рис.1**):

Рис. 1



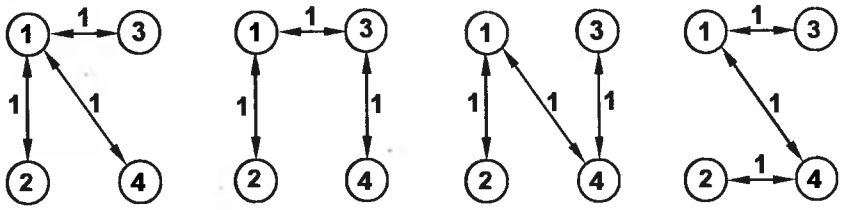
Веса C[i,j] дуг графа:

	1	2	3	4
1	0	1	1	1
2	1	0	1	1
3	1	1	0	1
4	1	1	1	0

И пусть веса всех ребер равны 1. Примерами минимального остовного

дерева для данного графа могут быть его подграфы, изображенные на **рис.2**.

Рис. 2



Если же веса ребер исходного графа заданы следующим образом:

	1	2	3	4
1	0	1	1	1
2	1	0	5	5
3	1	5	0	5
4	1	5	5	0

то, так как изменились веса некоторых входящих в остовные графы ребер, соответствующие остовные графы примут вид как на **рис.3**:

Рис. 3

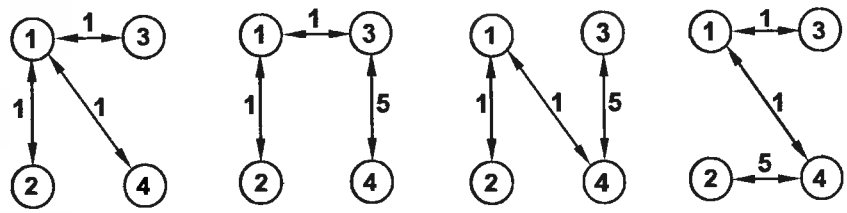
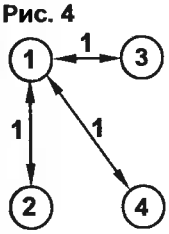


Рис. 4



Теперь суммы весов приведенных выше четырех деревьев будут 3, 7, 7, 7 соответственно, и минимальное остовное дерево будет единственным (**рис.4**):

Заметим также, что в остовном дереве для графа из N вершин всегда содержится N-1 соединяющих их ребер.

2. Задача "Веревочный телеграф"

(Гомельская городская олимпиада,1999 г.)

Тимур и его друзья, приехав летом на свои старые дачи, решили устроить на время своего отдыха игру. Они организовали команду, чтобы тайно помогать жителям дачного городка в их повседневных делах.

Дачный городок довольно большой, и дома, в которых живут друзья Тимура, расположены далеко друг от друга. Как быстро передавать друг другу сообщения? Как собирать ребят на совет?

Тимур решил проложить веревочный телеграф, который связал бы все домики, в которых живут ребята из его команды.

Всего имеется N домиков. По карте ребята вычислили координаты каждого домика (Xi, Yi) в целых числах и выписали их на бумаге. За единицу измерения координат они взяли один метр. Однако возник вопрос, какие домики нужно соединять веревочным телеграфом, чтобы связь была между всеми домиками (возможно, через другие домики), а общая длина всех веревок была как можно меньше?

Требуется написать программу, которая по координатам домиков определяла бы, какова минимальная общая длина всех веревок, соединяющих все домики между собой (возможно, через другие домики).

Порядок ввода	Порядок вывода
N X1 Y1 X2 Y2 : Xn Yn	X.xx (X.xx — минимальная общая длина веревки с точностью до двух знаков в дробной части)
Пример ввода	**Пример вывода**
5 100 200 200 200 300 400 400 200 400 100	623.61

Ограничения:

$0 < N \le 100$

$-32000 \le Xi, Yi \le 32000$

Рассмотрим домики тимуровцев как вершины графа, веревки между ними — как ребра графа, а длины веревок — как веса ребер. Теперь перед нами задача о минимальном остовном дереве. Тело программы, решающей эту задачу, может выглядеть следующим образом:

```
begin
  InputData;          {Ввод исходных данных}
  MinTree;            {Построение минимального
                       остовного дерева}
  OutputData;         {Вывод результата}
end.
```


В данном случае исходный граф — полный, т.е. существует ребро между любыми его двумя вершинами, поскольку по условиям задачи веревки можно протянуть между любыми двумя домиками.

В данной задаче удобно представлять граф в виде матрицы весов ребер. D[i,j] — расстояние между вершинами i и j. Ввод данных, обеспечивающий вычисление D[i,j] по исходным данным задачи, представлен ниже и не требует комментариев:

```
procedure InputData;
  begin
    assign (input,'input.txt'); reset(input);
    readln(N);
    for i:=1 to N do readln(x[i],y[i]);
    close(input);
    for i:=1 to N do
      for j:=1 to N do
        D[i,j]:=sqrt(sqr(x[i]-x[j])+sqr(y[i]-y[j]));
  end;
```

Результат работы алгоритма построения остовного дерева методом Прима будет представлен в виде массива Pred[1..N]:

Pred[i]=j,

т.е. предком вершины i в остовном графе будет вершина j. Или, другими словами, минимальное остовное дерево будет содержать ребро (i,j), а у вершины 1 предка не будет (Pred[1]=0).

Тогда ответ задачи по построенному остовному дереву получается следующим образом:

```
procedure OutputData;
  var
    Answer  : single;
  begin
    Answer:=0;
    for i:=2 to N do
      Answer:=Answer+D[i,Pred[i]];
    assign (output,'output.txt'); rewrite(output);
    writeln(Answer:0:2);
    close(output);
  end;
```

Рассмотрим теперь собственно алгоритм Прима для построения минимального остовного дерева. Идея алгоритма Прима заключается в следующем.

Пусть A — это множество ребер части остова, растущей от пустого множества ребер к минимальному остовному дереву. Формирование множества A может начинаться с произвольной вершины, называемой корневой. Для определенности, в реализации в качестве корневой выбирается вершина 1. На каждом шаге в множество (дерево) A добавляется ребро наименьшего веса среди ребер, соединяющих вершины из дерева A с вершинами не из дерева A.

После выполнения N-1 раз таких добавлений мы получаем минимальное остовное дерево.

Для эффективной организации выбора нужного для добавления ребра используется очередь с приоритетами, в которой хранятся все вершины, еще не попавшие в дерево A. При этом каждая вершина i снабжена также приоритетом — специальной характеристикой Key[i], которая равна минимальному весу ребер, соединяющих вершину i с вершинами из A.

Более формально алгоритм Прима может быть записан следующим образом:

```
Ставим в очередь Q все вершины исходного графа.
Устанавливаем для всех вершин Key[i] = бесконечность (MaxD).
Заносим в дерево A вершину 1:
   key[1] =0  - расстояние от нее  до вершин дерева A,
   Pred[1]=0  - у корневой вершины нет предка.
Пока очередь Q не пуста,
  Выбираем из очереди вершину U такую, для которой расстояние
          от нее до вершин из множества A минимально.
  Для каждой вершины V, соседней с U
      (т.е. в графе имеется ребро (U,V))
    если V находится в очереди и D[U,V]<key[V]
      то  Pred[V]=U
          key[V]=D[U,V]
```

Программная реализация алгоритма Прима имеет вид:

```
procedure MinTree;
  Const
    MaxD =1e16;    {"бесконечность"}
    Free =0;
    Used =1;
  var
    Lbl   : array [1..MaxN] of byte;
    Key   : array [1..MaxN] of single;
    u,v   : byte;
    Min   : single;
  begin
    for i:=1 to N do
      begin
        key[i]:=MaxD;
        Lbl[i]:=Free;                    {Все вершины в очереди}
      end;
    Key[1]:=0;          {Начнем построение с остова с вершины 1}
    Pred[1]:=0;                    {У первой вершины нет предка}
    for i:=1 to N Do                              {Для всех вершин}
      begin
        Min:=MaxD;
        for j:=1 to N Do                    {Находим  вершину,}
          if (Lbl[j]=Free) and (Key[j]<Min) {ближайшую к остову}
            then begin u:=j; Min:=Key[j] end;
        Lbl[u]:=Used;           {Помечаем ее как использованную}
        for v:=1 to N do  {Для всех оставшихся в очереди вершин}
          if (Lbl[v]=Free) and (D[u,v]<Key[v])
            then
              begin                    {Пересчитываем расстояние}
                Key[v]:=D[u,v];             {до растущего остова}
                Pred[v]:=u;          {Переустанавливаем предка}
              end;
      end;
  end;
```

Полное решение задачи "Веревочный телеграф" алгоритмом Прима имеет вид:

```
program gg99d1t2;
Const
  MaxN = 100;
var
  N,i,j : 1..MaxN;
  X,Y   : array [1..MaxN] of longint;
  D     : array [1..MaxN,1..MaxN] of single;
  Pred  : array [1..MaxN] of byte;

procedure InputData;
  begin
    assign (input,'input.txt'); reset(input);
    readln(N);
    for i:=1 to N do readln(x[i],y[i]);
    close(input);
    for i:=1 to N do
      for j:=1 to N do
        D[i,j]:=sqrt(sqr(x[i]-x[j])+sqr(y[i]-y[j]));
  end;

procedure OutputData;
  var
    Answer  : single;
  begin
    Answer:=0;
    for i:=2 to N do
      Answer:=Answer+D[i,Pred[i]];
    assign (output,'output.txt'); rewrite(output);
    writeln(Answer:0:2);
    close(output);
  end;

procedure MinTree;
  Const
    MaxD =1e16;
    Free =0;
    Used =1;
  var
    Lbl   : array [1..MaxN] of byte;
    Key   : array [1..MaxN] of single;
    u,v   : byte;
    Min   : single;
  begin
    for i:=1 to N do
      begin
        key[i]:=MaxD;
        Lbl[i]:=Free;
      end;
    Key[1]:=0;          {Начнем построение с остова с вершины 1}
    Pred[1]:=0;                    {У первой вершины нет предка}
    for i:=1 to N Do                              {Для всех вершин}
      begin
        Min:=MaxD;
        for j:=1 to N Do                    {Находим вершину,}
          if (Lbl[j]=Free) and (Key[j]<Min) {ближайшую к остову}
            then begin u:=j; Min:=Key[j] end;
        Lbl[u]:=Used;           {Помечаем ее как использованную}
        for v:=1 to N do          {Для всех оставшихся вершин}
          if (Lbl[v]=Free) and (D[u,v]<Key[v])
            {Пересчитываем}
            then
              begin
                Key[v]:=D[u,v];          {расстояние до остова}
                Pred[v]:=u;          {Переустанавливаем предка}
              end;
      end;
  end;

begin
  InputData;
  MinTree;
  OutputData;
end.
```

(Продолжение следует)

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

НИЗКОУРОВНЕВОЕ ПРОГРАММИРОВАНИЕ

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

Программирование под WIN32

12. "Заклинание" окна

#1. Воистину правы те, кто обожествляет **окно**, "объект, который дал Windows ее название, объект, который, как кажется, вскоре приобретет человеческие свойства, объект, который, может быть, будет вам мерещиться" — тысячу раз правы ;). Ибо окно — это не просто четырехугольник на экране, не просто "рабочая область" программы, содержащая кучу фенечек, (кнопочек, менюшек, полос прокрутки и пр.), призванных предоставить удобный пользовательский интерфейс. Окно —это целая объектно-ориентированная философия!

К счастью, вам не придется ее изучать при программировании на ассемблере. Ибо на самом деле нет никаких объектов, есть только регистры и байты, а всякие там полиморфизмы и наследования — происки "мракобесов" и ортодоксов. К сожалению, подобные ренегаты встречаются и среди ассемблерщиков — они пытаются создать ассебмлерную модель ООП. Но, к счастью, и в стане "идеологических врагов" также есть свои ренегаты — например, Кормак разрабатывает своихQuake'ов на Си (без ++).

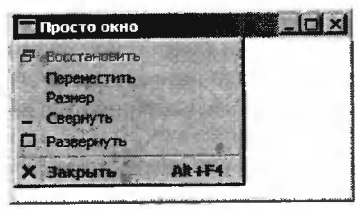
Насколько оконные приложения красивее консольных, этих пережитков черно-белого прошлого, насколько многозадачный режим удобнее однозадачного — во столько же раз возрастает и сложность программирования. И если для создания консольного приложения нам достаточно было щелкнуть разок пальцами — и готово, то оконное приложение — это "магия" на порядок выше. Тут придется и сложные "заклинания" составлять, и "жертвоприношения" устраивать, и бесконечно долго "скакать с бубном"...

Программы с графическим интерфейсом были и во времена ДОС'а, но в них отрисовкой разных графических элементов должна была заниматься сама программа. Под Windows же все окна как таковые "принадлежат" не прикладной программе, а операционной системе. Именно она их рисует, а задача программиста — всего лишь указать, где и в каком месте изобразить тот или иной графический объект.

Создатели "иксов" (графической оболочки в Линуксе) в развитии этой мысли пошли еще дальше. Они вообще разделили всю "графику" на x-клиент и x-сервер! Любая графическая программа запускается в контексте сервера, а x-клиенту (который вообще может находится на другой машине) всего лишь передается поток данных, наподобие "нарисовать в таких-то координатах четырехугольник с такой-то надписью и подождать, пока пользователь не кликнет мышью". Прикольно — то, что в Windows 2000 Advansed Server присутствует как дополнительная фича (требующая, естественно, нехилых дополнительных ресурсов), в XFree присутствует изначально.

От такого подхода выигрывают как пользователи, которые избавлены от необходимости изучать интерфейс каждой новой программы, так и программисты, так как в их подчинении уже есть куча стандартных, оттестированных и готовых для использования "графических" функций. Однако, как уже говорилось, есть и обратная сторона — возросшая сложность разработки приложений, в чем мы с вами скоро сможем убедиться. И будем надеяться, что "мерещиться", о котором говорилось выше, не станет для нас диагнозом.

#2. Для тех, кто "в танке" — здесь и далее под окном подразумевается нечто похожее не следующий рисунок:



С этим окном мы можем делать все, что угодно: сворачивать, разворачивать, максимизировать, перемещать, прятать, изменять размеры, закрывать... Все окна имеют рамку, которая определяет границы окна и используется для изменения его размеров. В левом верхнем углу находится *"значок системного меню"*, щелчок по которому приводит к отображению соответствующей менюшки. Рядом с ним — *"заголовок окна"*, еще правее — кнопки свертывания, развертывания и закрытия окна. Само "внутреннее содержимое" окна, этакий белый квадрат, называется *"клиентской областью"*, и именно в ней (области) происходит отображение признаков деятельности нашей программы.

Всю эту красотищу реализовывает следующий код (не забудьте изменить ключ линкера */SUBSYSTEM* с *CONSOLE* на *WINDOWS*):

```
.386

.model flat,stdcall
option casemap:none

include \masm32\include\windows.inc
include \masm32\include\user32.inc
include \masm32\include\kernel32.inc
include \masm32\include\gdi32.inc

includelib user32.lib
includelib kernel32.lib
includelib gdi32.lib

WinMain proto :DWORD,:DWORD,:DWORD,:DWORD

.data

  ClassName db "SimpleWinClass",0
  AppName   db "Просто окно",0

.data?

  hInstance   HINSTANCE ?
  CommandLine LPSTR ?

.code

;-------------------------------------
;Подготовка параметров WinMain
;-------------------------------------

start proc
  invoke GetModuleHandle, NULL
  mov    hInstance, eax
  invoke GetCommandLine
  mov    CommandLine, eax
  invoke WinMain, hInstance, NULL, CommandLine,\
         SW_SHOWDEFAULT
  Invoke ExitProcess, eax
start endp

;-------------------------------------------------------
;WinMain (написал бы я, что "главная", если бы так оно
;и было; так ведь нет!)
;-------------------------------------------------------
```

```
WinMain proc hInst:HINSTANCE,hPrevInst:HINSTANCE,
                    CmdLine:LPSTR,CmdShow:DWORD

  LOCAL  wc:WNDCLASSEX
  LOCAL  msg:MSG
  LOCAL  hwnd:HWND

  ;--- 1. описываем клас окна

  mov    wc.cbSize,SIZEOF WNDCLASSEX
  mov    wc.style, CS_HREDRAW or CS_VREDRAW
  mov    wc.lpfnWndProc, OFFSET WndProc
  mov    wc.cbClsExtra,NULL
  mov    wc.cbWndExtra,NULL
  push   hInstance
  pop    wc.hInstance
  invoke GetStockObject, WHITE_BRUSH
  mov    wc.hbrBackground, eax
  mov    wc.lpszMenuName,NULL
  mov    wc.lpszClassName,OFFSET ClassName
  invoke LoadIcon,NULL,IDI_APPLICATION
  mov    wc.hIcon,eax
  mov    wc.hIconSm,eax
  invoke LoadCursor,NULL,IDC_ARROW
  mov    wc.hCursor,eax

  ;--- 2. регистрируем класс окна

  invoke RegisterClassEx, addr wc

  ;--- 3. создаем окно

  INVOKE  CreateWindowEx,NULL,ADDR ClassName,ADDR AppName,\
          WS_OVERLAPPEDWINDOW,CW_USEDEFAULT,\
          CW_USEDEFAULT,CW_USEDEFAULT,CW_USEDEFAULT,NULL,NULL,\
          hInst,NULL
  mov     hwnd,eax

  ;--- 4. отображаем окно

  invoke ShowWindow, hwnd,SW_SHOWNORMAL
  invoke UpdateWindow, hwnd

  ;--- 5. запускаем цикл обработки сообщений

  .WHILE TRUE
    invoke     GetMessage, ADDR msg,NULL,0,0
  .BREAK .IF (!eax)
    invoke TranslateMessage, ADDR msg
    invoke DispatchMessage, ADDR msg
  .ENDW

  mov    eax,msg.wParam
  ret
WinMain endp

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

WndProc proc hWnd:HWND, uMsg:UINT, wParam:WPARAM,
lParam:LPARAM
  .IF uMsg==WM_DESTROY
  invoke PostQuitMessage,NULL
  .ELSE
    invoke DefWindowProc,hWnd,uMsg,wParam,lParam
    ret
  .ENDIF
  xor eax,eax
  ret
WndProc endp

end start
```

Как вы думаете, почему мы в свое время начали программировать под win32 "издалека", с консольных приложений? А чтобы не так страшно было! И не так много всего сразу ;). Можно сколько угодно ругать разработчиков мелкософта, но, тем не менее, факт имеет место быть: вышеприведенный код — далеко не маленький и не простой шаблон минимального оконного приложения. Не пугайтесь, ниже мы еще устроим ему детальную разборку.

#3. Разрабатывая программы под ДОС (консольной подсистемы win32 это тоже, в общем-то, касается), мы привыкли к тому, что взаимодействие с операционной системой инициируется самой программой. Так вот, в оконной программе не только программа вызывает функции операционной системы, но и *операционная система вызывает функции программы*. Программа находится в состоянии ожидания до тех пор, пока операционная система не пошлет ей *сообщение* посредством вызова специальной функции (еще раз повторюсь — *windows вызывает функцию программы*). А после того как адресованное ей сообщение будет принято программой, она может выполнить то либо иное запрограммированное действо. И хотя в ответ на один вызов операционной системой функции программы та может ответить парой-десятком вызовов функций операционной системы — тем не менее, именно последняя является инициатором всей этой "бурной" деятельности.

Операционная система вызывает эту функцию при: создании и удалении окна, изменении его размеров, перемещении, свертывании, выборе пункта меню, манипуляции с полосами прокрутки или с мышью... Вызывает, чтобы сообщить программе о необходимости перерисовать клиентскую область.

Идея функции, которая "находится" в программе, но вызывается операционной системой, не нова. Так, функция signal в C может перехватить комбинацию клавиш Ctrl+Break. С помощью конструкции ON (не говоря уж о языке ассемблера) в Visual Basic'е можно перехватывать аппаратные прерывания. Однако именно в Windows эта идея расширена и пронизывает всю систему.

Минимальное ***консольное*** приложение, если вы помните, содержит одну-единственную функцию *ExitProcess*, и никаких действий, кроме своего корректного завершения, не совершает.

Минимальное же ***оконное*** приложение вызывает не менее 14 апишных функций, а само состоит из трех процедур, причем одна из них постоянно вызывается самой операционной системой.

#4. Для большинства программистов на ЯВУ, даже если это Си без плюсов, и пишут они исключительно "на WinApi", оконное приложение начинается с процедуры *WinMain*, которую многие "высокоуровневые" товарищи ошибочно считают точкой входа в программу. MSDN же потворствует этому распространенному заблуждению:

```
WinMain
Функция WinMain вызывается операционной системой
как точка входа в Win32-приложение.

int WINAPI WinMain(
    HINSTANCE hInstance,     // хэндл данного экземпляра
    HINSTANCE hPrevInstance, // хэндл предыдущего экземпляра
    LPSTR lpCmdLine,         // указатель на командную строку
    int nCmdShow             // режим отображения
);
```

hInstance — это хэндл ("описатель", "идентификатор", "дескриптор" — кто еще какие синонимы знает?) ***данного экземпляра***, уникальное число, идентифицирующее каждую запущенную программу. Например, если мы запустили программу два раза, то мы получим два экземпляра программы, и каждый из них будет иметь собственный описатель. Это можно сравнить, например, с PID'ом процесса в диспетчере задач Windows2000.

hPrevInstance — вообще-то, это идентификатор "предыдущего экземпляра", но в настоящее время этот параметр не работает, он устарел, оставлен для совместимости и всегда равен нулю.

lpCmdLine — указатель на ноль-терминированную строку, в которой содержатся параметры, переданные программе при ее запуске из командной строки.

nCmdShow — константа, определяющая, каким именно образом будет показано окно программы при его запуске *другой программой*. Просто знайте, что такая возможность

существует ;). В нашем примере мы поступим мудро, напишем *SW_SHOWDEFAULT*.

Так вот, *MSDN* врет, говоря, что функция *WinMain* является *entery point* для win32-приложения. В этом несложно убедиться, открыв любую исполняемую программу в *IDA* и совершив прыжок на точку входа (*Ctrl+E*). Там совершенно ясно видно, что перед вызовом *WinMain* происходит куча всего интересного: выяснение версии операционной системы, переменных окружения и, как само собой разумеющееся — подготовка всех параметров, которые якобы, судя по прототипу, Windows должна передавать вызываемой программе.

— *Так что же это выходит, даже MSDN'у верить нельзя?* — спросите вы.

А что же вы хотели? Языку Си, хотя его и считают "высокоуровневым ассемблером", до истинного ассемблера все-таки далеко ;). Дело в том, что при сборке проекта компоновщиков Visual C++, в программу добавляется стартовая функция *_WinMainCRTStartup*, которая сначала "подготавливает почву" и только потом вызывает функцию *WinMain*.

Что из этого следует? А то, что мы сами должны позаботиться о стартовой функции и перед вызовом WinMain "вычислить" как минимум два ее параметра — хэндл модуля и указатель на командную строку.

Смотрим на исходник. Хэндл экземпляра получаем при помощи функции *GetModuleHandle* и сохраняем его в переменной *hInstance*. Истинное его значение — базовый адрес в памяти, указывающий на ту область в адресном пространстве процесса, где находится представление данного exe'шника. Например, если операционная система открывает исполняемый файл и загружает его содержимое по адресу 400000h, именно это значение и вернет нам функция *GetModuleHandle* (параметр NULL свидетельствует о том, что мы хотим получить хэндл текущего модуля, а не какого-то иного).

При помощи функции *GetCommandLine* находим указатель на командную строку и сохраняем его в переменной *CommandLine*.

Теперь, когда нам известны все аргументы *WinMain*, можно вызывать и саму функцию.

#5. Функция *WinMain* должна последовательно выполнить следующие действия:

- ***описать*** класс окна;
- ***зарегистрировать*** этот класс в операционной системе;
- ***создать*** окно данного класса;
- ***отобразить*** это окно;
- запустить выполнение ***цикла обработки сообщений***.

Функция *WinMain* начинается с декларирования трех локальных переменных.

Переменная *wc* — это структура *WNDCLASSEX*, определенная следующим образом (выдрано из *windows.inc*):

```
WNDCLASSEX STRUCT
  cbSize         DWORD ? ;размер структуры
  style          DWORD ? ;тип окна
  lpfnWndProc    DWORD ? ;указатель на процедуру окна
  cbClsExtra     DWORD ? ;дополнительная память класса
  cbWndExtra     DWORD ? ;дополнительная память окна
  hInstance      DWORD ? ;хэндл данного экземпляра
  hIcon          DWORD ? ;хэндл большой иконки
  hCursor        DWORD ? ;хэндл указателя мыши
  hbrBackground DWORD ? ;цвет фона
  lpszMenuName   DWORD ? ;указатель на имя главного меню
  lpszClassName DWORD ? ;указатель на имя класса
  hIconSm        DWORD ? ;хэндл маленькой иконки
WNDCLASSEX ENDS
```

Регистрацию класса окна выполняет функция *RegisterClassEx*, требующая в качестве аргумента указатель на эту структуру, естественно, должным образом "заполненную". Рассмотрим элементы этой структуры.

cbSize — размер структуры *WNDCLASSEX*. Ее мы определяем при помощи директивы *sizeof*.

style — стиль класса, определяет дополнительные элементы класса. Два или более элемента стиля могут комбинироваться при помощи оператора *or*. Описание этих констант легко найти в *MSDN*. В нашем же примере мы собираемся "делать окно" через *CS_HREDRAW* и *CS_VREDRAW* — это означает, что все окна, созданные на основе этого класса, будут перерисовываться при изменении их горизонтального и вертикального размеров. Искатели приключений могут добавить к этим двум элементам еще и флаг *CS_NOCLOSE* и попытаться закрыть программу естественным путем, не вызывая диспетчер задач ;).

lpfnWndProc — указатель на процедуру окна, о которой мы уже говорили в п.#3 как о функции, вызываемой операционной системой, т.е. о функции, в которой происходит обработка сообщений, адресуемых операционной системой окну программы.

cbClsExtra и *cbWndExtra* — число байтов, которые надо дополнительно зарезервировать в структуре класса и структуре окна соответственно. Позже мы еще разберемся, зачем нужно резервировать дополнительное место, а пока что просто ответим — *NULL*.

hInstance — описатель экземпляра, с которым мы уже разобрались (он такой же и для параметров *WinMain*). Парочка push/pop в соответствующем месте исходника — это обычное копирование из памяти в память, как вы уже, наверняка, догадались.

hIcon и *hIconSm* — хэндлы большой и маленькой иконок, которые будут отображаться в окне программы. В нашем случае при помощи функции *LoadIcon* мы загружаем стандартную иконку приложения (*IDI_APPLICATION*). Желающие поизвращаться могут попробовать следующие типы иконок (соответствующие константы определены все в том же *windows.inc*):

	IDI_APPLICATION	Пиктограмма по умолчанию
	IDI_ERROR	Символ ошибки
	IDI_INFORMATION	Символ информации
	IDI_QUESTION	Знак вопроса
	IDI_WARNING	Восклицательный знак
	IDI_WINLOGO	Логотип Windows

hCursor — хэндл курсора. Загружаем функцией *LoadCursor* стандартный рисунок курсора — стрелку (*IDC_ARROW*). Хотя могли бы взять любой рисунок из следующей простыни:

	DC_ARROW	Указатель стрелки по умолчанию
	IDC_CROSS	Перекрестие
	IDC_HAND	Рука
	C_IBEAM	Вертикальная двутавровая балка
	IDC_WAIT	Песочные часы

hbrBackground — хэндл кисти, которой мы хотим закрасить фон окна. При помощи функции *GetStockObject* получаем хэндл стандартной белой кисти (*WHITE_BRUSH*) и записываем этот хэндл в структуре. Можете разнообразия ради попробовать следующие константы:

BLACK_BRUSH	Черная кисть
DKGRAY_BRUSH	Темно-серая кисть
LTGRAY_BRUSH	Светло-серая кисть

Заметьте, что речь идет не о цветах, а о "кистях", которые представляют собой шаблон пикселов различных цветов, используемый для закрашивания заданной области. Впрочем, мы еще поговорим об этом более обстоятельно.

lpszMenuName — хэндл менюшки. Учитывая, что меню в

нашей программе нет, присваиваем этому элементу структуры значение *NULL*.

lpszClassName — указатель на имя класса. Подсовываем, естественно, *offset ClassName*.

Вот так, долго и утомительно мы подготавливали необходимую для регистрации класса окна структуру :(. И наконец, свершает главное на этом этапе — непосредственно регистрацию класса — следующая строка:

```
invoke RegisterClassEx, addr wc
```

И где-то там, в недрах операционной системы, соскочила волшебная собачка, освободив храповое колесо, завертелись зубчатые колеса и натянулась, готовая к работе, стальная пружина...

#6. Так дадим же этой пружине развернуться — при помощи функции *CreateWindiwEx* откроем окно, которое так долго готовили и настраивали! Но сначала посмотрим на описание этой функции:

```
HWND CreateWindowEx(
  DWORD dwExStyle,        // расширенный стиль окна
  LPCTSTR lpClassName,    // зарегистрированное имя класса
  LPCTSTR lpWindowName,   // имя окна
  DWORD dwStyle,          // стиль окна
  int x,                  // горизонтальная позиция окна
  int y,                  // вертикальная позиция окна
  int nWidth,             // ширина окна
  int nHeight,            // высота окна
  HWND hWndParent,        // хэндл родительского окна
  HMENU hMenu,            // хэндл меню
  HINSTANCE hInstance,    // хэндл экземпляра приложения
  LPVOID lpParam          // параметры создания
);
```

dwExStyle — в нашем случае мы не использовали никаких расширенных стилей.

lpClassName, *lpWindowName* — указатели на имя класса и имя окна, они у нас в глобальных переменных *ClassName* и *AppName* соответственно.

dwStyle (на префикс обратили внимание?) — стиль окна. Вот табличка наиболее часто употребляемых констант для задания стилей:

WS_OVERLAPPED	Перекрывающееся с обрамлением
WS_MAXIMIZEBOX	Кнопка максимизации
WS_MINIMIZEBOX	Кнопка минимизации
WS_SYSMENU	Системное меню
WS_HSCROLL	Горизонтальный скролл
WS_VSCROLL	Вертикальный скролл

Стиль *WS_OVERLAPPEDWINDOW* является комбинацией стилей *WS_OVERLAPPED*, *WS_CAPTION*, *WS_SYSMENU*, *WS_THICKFRAME*, *WS_MINIMIZEBOX*, и *WS_MAXIMIZEBOX*. Это стандартный стиль окна. Под "комбинацией" подразумевается, что эти константы можно объединять посредством оператора OR.

x, *y*, *nWidth*, *nHeight* — координаты: длина, ширина... Доверимся Windows, используя ее стандартные настройки, т.е. четыре раза скажем *CW_USEDEFAULT* ;).

hWndParent и *hMenu* — хэндл родительского окна и меню. Ни того, ни другого у нашей программы пока нет, а посему и NULL с ними, как и с дополнительными параметрами создания *lpParam*.

Хэндл экземпляра приложения у нас давным-давно в переменной *hInst*...

Windows 2000 абсолютно безразличен этот параметр, но для 9x он должен содержать хэндл экземпляра приложения.

Выполняем функцию... Думаете, появится окно? Ничего подобного! Оно появилось, все еще где-то там, в "желудке" операционной системы. А в качестве признаков жизни функция вернула нам хэндл окна, который мы благодарно сохраняем в переменной *hwnd*.

#7 Чтобы "желудок" Windows все-таки "выплюнул" наше долгожданное окошко, необходимо сделать еще два вызова.

Сначала:

```
invoke ShowWindow, hwnd, SW_SHOWNORMAL
```

чтобы показать окно и указать, каким именно образом мы хотим его показать:

SW_HIDE	Скрытым
SW_MINIMIZE	Минимизированным
SW_MAXIMIZE	Максимизированным
SW_RESTORE	Восстановленным

Затем:

```
invoke UpdateWindow, hwnd
```

чтобы перерисовать рабочую область.

Все, теперь окно на мониторе. Нам осталось сделать только одно — позаботиться о его взаимоотношениях с операционной системой.

#8. Последнее, что мы сделаем в процедуре WinMain — запустим *"цикл обработки сообщений"* (message loop). Для каждой загруженной в данный момент программы Windows поддерживает *"очередь сообщений"* (message queue). Во время работы прикладной программы ей постоянно посылаются разнообразные сообщения. Так вот, все эти сообщения сохраняются в "очереди сообщений приложения" и хранятся там до тех пор, пока не будут извлечены и обработаны. Давайте посмотрим на этот кусок кода:

```
.WHILE TRUE
  invoke      GetMessage, ADDR msg,NULL,0,0
.BREAK .IF (!eax)
  invoke TranslateMessage, ADDR msg
  invoke DispatchMessage, ADDR msg
.ENDW
```

Как нетрудно догадаться из ее названия, функция *GetMessage* "получает сообщение", т.е. извлекает его из очереди и "запихивает" в структуру *msg* (первый параметр — указатель на нее), о которой мы еще поговорим чуть ниже. Три последующих нулевых параметра расшифровываются как "все сообщения от всех окон, созданных этой программой".

Функция *TranslateMessage* транслирует виртуальные коды клавиш в символьные сообщения. В принципе, она здесь сто лет не нужна, мы просто написали ее для того, чтобы оставить "закладку на будущее".

А вот на функции *DispatchMessage* начинается самое интересное. Она отсылает сообщение *назад операционной системе*, чтобы та при первой возможности передала его *процедуре окна*. Т.е. таким образом мы постоянно напоминаем Windows о том, что она должна вызывать процедуру окна. "Вызови мою процедуру окна", "вызови мою процедуру окна" — и так до тех пор, пока функция *GetMessage* не вернет 0 (а ноль она вернет только в том случае, если извлеченным из очереди сообщением окажется *WM_QUIT*). Не правда ли, довольно извращенную схему придумали разработчики мелкософта?

Да, бесконечный цикл выполняется, но не стоит думать, что какой-то процент производительности процессора расходуется на этот цикл даже в том случае, если программа не активна, и никаких действий над ее окном не совершается — *GetMessage* никогда не вернет программе управление, если не получит от Windows никакого адресованного программе сообщения.

#9. В самом начале функции *WinMain* у нас задекларирована переменная *msg* типа *MSG*. Это структура, в которую операционная система заносит сообщение при обращении к ней функцией *GetMessage*.

```
MSG STRUCT
  hwnd    DWORD ? ;хэндл окна, для которого было послано сообщение
  message DWORD ? ;сообщение
  wParam  DWORD ? ;основная информация сообщения
  lParam  DWORD ? ;дополнительная информация, обусловленная сообщением
```

```
  time    DWORD ? ;время отправления
  pt      POINT <>;координаты мыши
MSG ENDS
```

hwnd — ясно, что это такое...

message — это идентификатор сообщения. Соответствующие ему константы можно найти в windows.inc, вот некоторые из них:

WM_CREATE	Создание окна
WM_SIZE	Изменение размеров
WM_VSCROLL	Вертикальный скроллинг
WM_PAINT	Отрисовка окна
WM_DESTROY	Разрушение окна

Т.е. когда мы, например, захватываем нижний правый угол окна и изменяем его размеры, то в результате этого при очередном проходе цикла обработки сообщений член message структуры msg примет значение *WM_SIZE*.

wParam и *lParam* — это основная и дополнительная информации сообщения. Если одной констатации факта "дык, событие" нам явно недостаточно, то можно и нужно анализировать и эти элементы структуры. Например, для того же WM_SIZE в wParam можно найти вот что:

SIZE_MAXIMIZED	Окно максимизированно
SIZE_MINIMIZED	Окно минимизированно
SIZE_RESTORED	Изменился размер, но не SIZE_MAXIMIZED и не SIZE_RESTORED

А в *lParam* содержатся новые высота-ширина клиентской области (младшее слово — ширина, старшее — высота).

Естественно, для других сообщений эти элементы будут содержать совершенно другие признаки.

time — время отправления сообщения (нужно ли говорить о том, что разница между временем отправления сообщения и временем его получения программой может порой достигать нескольких минут?).

pt — структура (ну да, "структура в структуре", это нормально) типа POINT (см. windows.inc), в которой содержатся координаты точки, с которой произошел постинг мессаги. В большинстве случаев это соответствует координатам мыши.

#10. Итак, вызвав функцию *DispatchMessage*, мы "отфутболиваем" вышерасписанную мессагу назад в Windows с "ценным" указанием вызвать процедуру окна. Где-то в недрах операционной системы эта функция "разбирается" на параметры hWnd:HWND, uMsg:UINT, wParam:WPARAM, lParam:LPARAM (смотрим на WndProc), которые передаются операционной системой при ее вызове. Сотый раз повторю — это Windows вызовет нашу процедуру окна (*WndProc*) и передаст ей параметры ("разобранную" на четыре полки структуру *msg*).

Именно в "процедуре окна" и происходит большая часть того, что называется "собственно программированием" оконного приложения. В этой части программы мы обыкновенной лестницей проверок отлавливаем необходимые нам сообщения и в ответ на них программируем те либо иные действия.

Например, единственное действие, которое предусмотрено нашим шаблоном — это обработка сообщения *WM_DESTROY*, в ответ на которое происходит посылка окну сообщения *WM_CLOSE*. В результате этого, при очередном вызове функции *GetMessage*, через eax возвращается 0, и программа завершает свою работу.

Все сообщения, на которые мы не удосужились написать собственные обработчики, по умолчанию обрабатываются функцией *DefWindowProc*, наверное, исходя из принципа, что все мессаги должны быть, тем или иным образом, но обязательно обработаны.

Вот такой вот он зверь — окно. Но, как говорится, медведя бояться — в лес не ходить. В следующих статьях мы с вами попробуем что-нибудь нарисовать на клиентской области.

(Продолжение следует)

А.ТРОФИМОВИЧ,
г.Могилев, МГУП,
E-mail; mti@mogilev.by

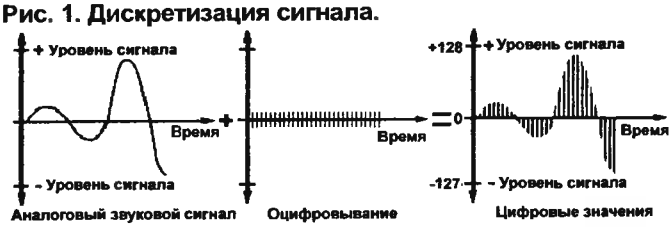
Программирование звуковой платы

Основные сведения о кодировании звука

Компьютер является цифровым устройством, и поэтому представляемый звуковой сигнал, являющийся по сути аналоговым, должен быть оцифрован.

Дискретизацией называется процесс превращения исходного звукового сигнала в цифровую форму, в которой он хранится. При этом сохраняются мгновенные значения звукового сигнала в определенные моменты времени (**рис.1**), называемые выборками (samples) [1].

Рис. 1. Дискретизация сигнала.



Дискретизация сигнала выполняется с помощью аналого-цифрового преобразователя (АЦП), обратный процесс осуществляется с помощью цифро-аналогового преобразователя (ЦАП).

К параметрам оцифровки аналогового сигнала относятся *частота дискретизации* и *количество уровней квантования* [2].

Частота дискретизации (sample rate) измеряется в герцах и показывает, сколько раз в секунду определяется значение амплитуды сигнала (амплитуда задается напряжением электрического сигнала, поэтому под амплитудой можно понимать величину напряжения). Полученное значение оцифровывается, т.е. определяется соответствующий уровень. Если используется стереозвук (два канала), то амплитуда измеряется по двум каналам одновременно, и образуются два значения.

При ***квантовании*** интервал допустимых амплитуд разбивается на N равных интервалов (N — число уровней квантования). Каждому уровню однозначно соответствует определенное значение. Если амплитуда сигнала попадает в некоторый интервал, то ее оцифрованным значением и является соответствующее интервалу число.

Разрядность звука характеризует количество битов, используемых для цифрового представления каждой выборки (для одного канала), и, очевидно, определяет число уровней квантования. По этому показателю различают 8-битный и 16-битный звук (256 и 65536 уровней соответственно).

Рис. 2. Различия при 8- и 16-битном представлении сигнала.

Чем выше число уровней квантования, тем точнее передается амплитуда звука, что сказывается на качестве воспроизведения сигнала (**рис.2**).

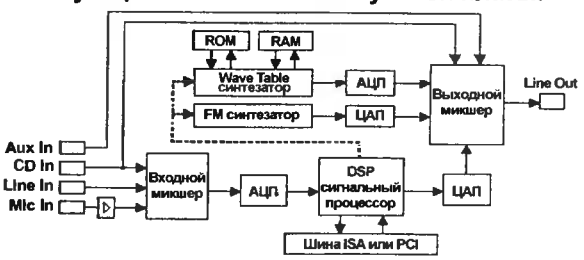
Таким образом, цифровой сигнал представляет собой набор чисел, характеризующих амплитуду сигнала в определенные промежутки времени [1].

Принципы функционирования звуковой платы

Каждая звуковая плата (ЗП) имеет несколько входов, которые расположены на металлической панели, выходящей на заднюю стенку системного блока компьютера. Основными входами являются (**рис.3**):

- Line In и Mic In — линейный и микрофонный входы. Отдельный вход Mic In предусмотрен из-за того, что выходной сигнал микрофонов имеет низкий уровень, и его нужно усиливать до нормального уровня (0 дБ), перед тем

Рис. 3. Функциональная схема звуковой платы.



как направлять на АЦП. Поэтому на микрофонных входах звуковой платы всегда установлен предусилитель — небольшая схема, повышающая уровень сигнала до нормального (линейного) уровня;

- CD In — вход проигрывателя компакт-дисков. Как правило, он расположен не на задней панели звуковой платы, а прямо на ней. Если в компьютере установлен привод CD-ROM, то можно связать его выход с этим входом звуковой платы. Такое соединение позволит слушать аудиокомпакт-диски и оцифровывать звук прямо с привода;

- Aux In — дополнительный вход, используемый на некоторых типах ЗП. Сигнал с этого входа минует основные устройства ЗП и поступает на выходной микшер, а оттуда — сразу на выход ЗП. Этот вход позволяет упростить коммутацию внешних устройств и использовать внутренний микшер ЗП для смешивания сигналов с внешнего и внутреннего источников. Например, если имеется автономный синтезатор, то можно его выход подключить к Aux In, и все, что играется, будет слышно в колонках, подключенных к ЗП.

Сигналы с внешних аудиоустройств подаются на входной микшер (рис.3), и после аналого-цифрового преобразования поступают в сигнальный процессор DSP (Digital Signal Processor) — "сердце" звуковой платы. Этот процессор управляет обменом данными со всеми остальными устройствами компьютера через шину ISA или PCI. Если центральный процессор выполняет программу записи звука, то цифровые данные поступают либо в оперативную память компьютера, либо прямо на жесткий диск (это зависит от выполняемой программы).

При воспроизведении звукового файла данные с жесткого диска через шину поступают в сигнальный процессор DSP, который направляет их на ЦАП. Электрический сигнал, получившийся в результате преобразования, в свою очередь, поступает на выходной микшер. Этот микшер практически идентичен входному, и управляется при помощи одной и той же служебной программы, у которой используются два разных "окна" для входных и выходных сигналов.

На любой универсальной мультимедийной ЗП есть синтезатор. В последнее время практически на всех платах устанавливается не один, а два синтезатора: FM — для сохранения совместимости с Sound Blaster и Ad Lib, и Wave Table — для получения более качественного звука. У Wave Table-синтезатора есть не только постоянная (ROM), но и оперативная (RAM) память, расширяющая творческие возможности ЗП. Каждый из синтезаторов, установленных на ЗП, имеет свой собственный ЦАП. После цифро-аналогового преобразования сигналы поступают на выходной микшер.

Формат представления звукового сигнала

Для представления звукового сигнала часто используется специальный формат PCM (Pulse Code Modulation — импульсно-кодовая модуляция), который задает кодирование моно, стерео, 8- и 16-битного звука.

В формате PCM сигнал задается набором последовательных сэмплов, каждый из которых, в зависимости от режима кодирования сигнала, располагается в памяти в формате ***ячейка памяти/значение амплитуды для i-го канала***:

- режим **8 бит, моно**

Байт1
Канал 0

- режим **16 бит, стерео**

Байт 1	Байт 2	Байт 3	Байт 4
Канал 0 (левый). Младший байт	Канал 1 (правый). Старший байт	Канал 0 (левый). Младший байт	Канал 1 (правый). Старший байт

Общие принципы программирования и работы ЗП в режиме генерации сигнала

Для хранения в формате PCM задающих сигнал данных, необходимо выделить буфер в памяти. Данные могут передаваться из буфера в ОЗУ на ЗП в режиме воспроизведения, а также от ЗП в память при работе в режиме записи. Контроллер DMA (прямого доступа к памяти) разбивает память на страницы по 64 Кб, поэтому необходимо, чтобы буфер не лежал на пересечении этих страниц. Это ограничивает размер буфера значением 64 Кб [2].

Необходимо настроить DMA-канал и ЗП для совместного функционирования. Одно из необходимых действий при этом — установка вектора прерывания. После настройки сигнал воспроизводится под управлением контроллера DMA. Данный процесс может быть циклическим — при достижении конца буфера воспроизведение начинается сначала.

В процессе работы вызывается настроенное ранее прерывание, которое сигнализирует о том, что половина буфера была уже воспроизведена, и звуковая плата начала воспроизводить следующую половину буфера. В это время программа может изменять данные в уже воспроизведенной половине и таким образом организовывать непрерывное воспроизведение динамического сигнала.

Затем в некоторый момент можно закончить воспроизведение сигнала, послав соответствующую команду на ЗП. После этого необходимо сбросить установки ЗП в исходное состояние и восстановить вектор прерывания [3].

Выполнение записи сигнала аналогично выполнению воспроизведения.

Порядок программирования ЗП с использованием автоинициализации DMA следующий:

- установка обработчика IRQ;
- установка временной константы;
- настройка DMA;
- включение динамиков;
- программирование DSP;
- процесс непосредственного воспроизведения сигнала;
- сброс DSP;
- восстановление перехваченного вектора прерывания.

Ниже представлены некоторые фрагменты программы для работы со звуковой платой в режиме генерации сигнала.

Установка нового обработчика для прерывания IRQ7:

```
xor     ax,ax              ;Замаскировать запросы прерывания IRQ7
mov     es,ax              ;на время программирования.
mov     si,0Fh*4           ;Соответствует прерыванию IRQ7.
in      al,021h
and     al,01111111b
out     021h,al
mov     ax,es:[si]         ;Перенаправить вектор на свой обработчик.
mov     [OLDInterruptOFS],ax
mov     ax,OFFSET OWN_IRQ
```

```
mov     es:[si],ax
mov     ax,es:[si+2]
mov     [OLDInterruptSEG],ax
mov     ax,cs           ;Демаскирование прерываний по IRQ7.
mov     es:[si+2],ax
in      al,021h
or      al,10000000b
out     021h,al
```

Пример обработчика прерывания:

```
OWN_IRQ:
pusha                   ;Сохранить регистры.
mov     dx,BASEADDR+00Eh ;DX = чтение из регистра состояния
in      al,dx           ;доступности данных (для оповещения
                        ;DSP о перехвате прерывания).
; — Выполнение необходимых действий —
mov     al,020h         ;Подтверждение завершения обработки
                        ;прерывания.
out     020h,al
popa
                        ;Восстановить регистры.
IRET
```

Управление работой ЗП, в основном, связано с программированием входящего в ее состав DSP. Для работы с ним используются порты <*Base_Addr*+06h> и <*Base_Addr*+0Ah> ... <*Base_Addr*+0Fh>, где *Base_Addr* — базовый адрес ввода/вывода для конкретной платы (обычно 220h). В частности, порт <*Base_Addr*+0Ch> используется для передачи команд и данных для DSP. Некоторые из команд DSP, которые используются для программирования ЗП, представлены в **табл.1**.

Табл. 1

Код	Назначение
040h	Установка временной константы
048h	Установка размера блока DMA. Команда служит для установки размера буфера
06h	Сброс сигнального процессора DSP в начальное состояние
090h	Установка режима автоинициализации DMA (использование ЦАП, 8-битные данные; инициирование передачи)
098h	Установка режима автоинициализации DMA (использование АЦП, 8-битные данные; инициирование передачи)
0D1h	Включение динамикое

Частота дискретизации для ЗП варьируется в диапазоне 4...44 kГц и задается установкой временной константы, которая определяется по формуле :

$$Time\_Const = 256 - \frac{1000000}{f_{req}},$$

где f_{req} — требуемая частота, Гц.

Например, для f_{req} = 44000 Гц,

$$Time\_Const = 256 - \frac{1000000}{44000} = 233.$$

Фрагмент кода для установки временной константы:

```
mov             dx,Base_Addr+00Ch       ;Базовый порт для DSP.
WAITWRITE
mov             al,040h                 ;Команда 40h - установка
out             dx,al                   ;временной константы.
WAITWRITE
mov             al,Time_Const           ;Непосредственно передача
out             dx,al                   ;в DSP значения
                                        ;Time_Const.
```

В этом фрагменте использован макрос WAITWRITE, необходимый для ожидания готовности DSP к приему данных. Для этого необходимо проверять 7-й бит прочитанного из порта <*Base_Addr*+0Ch> (порт состояния/команд) байта на 0 (готов к приему), ожидая выполнения этого условия:

```
WAITWRITE  MACRO                ;Аргументы: DX = (BASEADDR+0Ch).
LOCAL      loopWait,endloop
           push     cx
           push     ax
           xor      cx,cx       ;Задержка для старых моделей ЗП.
loopWait:  dec      cx
           jz       endloop
           in       al,dx       ;Чтение порта состояния.
           or       al,al
           js       loopWait    ;Переход, если бит 7 = 1
endloop:                        ;(запись не разрешена).
           pop      ax
           pop      cx
ENDM
```

Пример программирования канала 1 DMA (этот канал используется с большинством моделей ЗП семейства Sound Blaster):

```
mov     al,1            ;Маскирование канала 1 DMA для
add     al,4            ;предотвращения его использования на
out     0Ah,al          ;время процесса программирования.
mov     al,01010100b ;Установка режима работы канала 1
add     al,1            ;DMA: одиночная передача,
out     0Bh,al          ;автоинициализация, чтение.
mov     al,bl           ;Установка регистра страниц канала 1 DMA.
out     083h,al
mov     ax,si           ;Установка базового адреса страницы
out     02h,al          ;канала 1 DMA.
mov     al,ah
out     02h,al
mov     cx,SampleBufferLength      ;Установка размера буфера
dec     cx              ;со звуковыми данными (SampleBufferLength)
mov     al,cl           ;канала 1 DMA.
out     03h,al
mov     al,ch
out     03,al
mov     al,1            ;Демаскирование канала
out     0Ah,al
```

Фрагмент программы для включения динамиков:

```
mov     dx,BASEADDR+00Ch    ;DX = Команды /данные для DSP.
WAITWRITE
mov     al,0D1h             ;AL = Включить колонки.
out     dx,al
```

Фрагмент, производящий передачу размера буфера для DSP:

```
mov     dx,BASEADDR+00Ch ;DX = Команды/данные для DSP.
WAITWRITE
mov     al,048h          ;Команда 048h – установка размера
out     dx,al            ;блока DMA.
mov     cx,SAMPLEBUFFERLENGTH
dec     cx
WAITWRITE
mov     al,cl            ;Передача младшего байта размера блока.
out     dx,al
WAITWRITE
mov     al,ch            ;Передача старшего байта размера блока.
out     dx,al
```

Перед началом и после завершения работы с DSP необходимо сбросить его состояние, для чего используется порт <*Base_Addr*+06h> (Сброс DSP):

```
DSP_Reset:
     mov     dx,BASEADDR+06h
     mov     al,1
     out     dx,al           ;Начать сброс DSP.
     in      al,dx
     in      al,dx
     in      al,dx
     in      al,dx           ;Задержка 3 мкс.
     xor     al,al
     out     dx,al           ;Закончить сброс DSP.
     mov     dx,BASEADDR+0Eh ;DX = доступны данные из DSP.
WAITREAD
     mov     dx,BASEADDR+0Ah ;DX = чтение данных из DSP.
     IN      al,dx
     cmp     al,0aah         ;Если есть ЗП, возвращается 0AAh.
     je      SBthere
     jmp     RESET_ERROR     ;ЗП не найдена.
SBthere:                     ;Инициализация проведена успешно.
          ; ------ Выполнение необходимых действий -------
RESET_ERROR:                 ;Обработка ошибки.
          ; ------- Выполнение необходимых действий ------
```

В предыдущем фрагменте использован макрос WAITREAD для ожидания готовности данных в DSP для чтения:

```
WAITREAD MACRO          ;Аргументы: DX – порт состояния
LOCAL    loopWait,endloop;(обычно Base_Addr+0Eh).
     push ax
     push cx
     xor  cx,cx         ;Цикл ожидания для старых моделей ЗП.
loopWait:
     dec  cx
     jz   endloop
```

```
    in   al,dx          ;Проверка доступности данных.
    or   al,al
    jns  loopWait       ;Переход, если бит 7 = 1,
    endloop:
    pop  cx             ;данные не доступны.
    pop  ax
    ENDM
```

Можно осуществить генерацию заданного сигнала программным способом. В качестве примера ниже приведен фрагмент программы для генерации гармонического сигнала:

```
for f:=0 to samplebufferlength div 2 do begin
mem[seg(samplebuffer^):f*2]:= round((sin(2*pi*f/
  WaveLength1+Phase1)*Ampl));
mem[seg(samplebuffer^):f*2+1]:= round((sin(2*pi*f/
  WaveLength2+Phase2)*Amp2));
end;
```

Результат работы программы проверяется с помощью осциллографа.

Дальнейшим развитием использования звуковой платы в качестве аппаратно-программного комплекса может быть написание программного обеспечения для использования компьютера в качестве осциллографа, анализатора сигнала, универсального генератора сигнала, управляющего устройства в технической системе.

Литература

1. Гук М. Аппаратные средства IBM PC.— СПб.: Питер, 2000.
2. Мультимедиа. Под ред. А.И. Петренко.— Киев: BHV, 1994.
3. Andr B Baresel, Craig Jackson. SoundBlaster. Programming Information. Version 0.90 — 1995.

Преобразование псевдографических таблиц в формат HTML

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

Введение

При подготовке материалов для своей web-страницы мне понадобилось преобразовать в формат HTML текстовые файлы, многие из которых содержали таблицы, оформленные с помощью символов псевдографики. Преобразование таких таблиц вручную — дело долгое и скучное, поэтому я решил написать программу, которая бы этим занималась. Действительно, если компьютер может это делать — так пусть и делает :-)!

Объясню на примере, что за задачу надо решить. Пусть у нас имеется текстовый файл, в котором есть, например, таблица как на **рис.1**, а нам надо, чтобы эта таблица выглядела в итоговом HTML-документе как показано на **рис.2**.

Для этого нужно сформировать такой HTML-код:

```
<table align=center border=1>
<tr bgcolor=#ccffff align=center>
<th>Значение a</th>
<th>Значение b</th>
</tr>
<tr bgcolor=#ccffcc align=center>
<td>1</td>
<td>15</td>
</tr>
<tr bgcolor=#ccffff align=center>
<td>2</td>
<td>30</td>
</tr>
<tr bgcolor=#ccffcc align=center>
<td>3</td>
<td>45</td>
</tr>
<tr bgcolor=#ccffff align=center>
<td>4</td>
<td>60</td>
</tr>
</table>
```

Рис. 1

Значение a	Значение b
1	15
2	30
3	45
4	60

Рис. 2

Значение a	Значение b
1	15
2	30
3	45
4	60

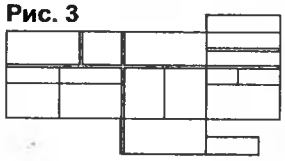
Формированием соответствующего HTML-кода на основе исходного текстового файла и должна заниматься программа.

Возможности программы

Мне хотелось сделать серьезную программу, мощную и универсальную. В результате всех усилий получилась программа со следующими возможностями:

- автоматическое распознавание и преобразование в формат HTML всех таблиц, содержащихся в исходном текстовом файле (в том числе и таблиц сложной формы, например, как на **рис.3**);
- к формируемым таблицам может автоматически добавляться заголовок, содержащий определенный пользователем текст и порядковый номер таблицы. Можно указать, с какого номера начнется нумерация таблиц;
- можно задать атрибуты для используемых при описании таблицы HTML-тегов (<table>, <caption>, <tr>, <th>, <td>);
- можно указать, сколько первых строк сверху и первых столбцов слева в каждой таблице будут содержать тег <th> вместо <td>;
- можно указать, какие теги должны добавляться перед текстом заголовка таблицы и после него, перед содержимым ячеек и после него (отдельно для <th>- и <td>-ячеек);
- можно указать, нужно ли добавлять символ   в пустые ячейки таблицы, чтобы вокруг этих ячеек рисовалась рамка;
- можно указать кодировку получаемого файла — ALT, WIN, KOI;
- можно указать, чтобы кавычки в тексте таблиц заменялись не на символ ", а так: первая кавычка в ячейке таблицы заменяется на «, вторая — на », следующая — опять на «, и так далее. Тогда кавычки будут выглядеть более правильно — не “ ”, а « »;
- параметры, определяющие внешний вид получаемых таблиц и кодировку выходного файла, считываются из ini-файла, который хранится в обычном текстовом формате (и может содержать комментарии). Можно создать несколько ini-файлов с различными параметрами, и при запуске программы указывать имя нужного в данный момент ini-файла.

Рис. 3



Используемый алгоритм

Сравнительно легко придумать, как распознавать и преобразовывать в HTML простые таблицы, форма которых полностью определяется путем задания всего двух параметров: количества строк и столбцов (как на рис.1). Значительно более трудной задачей является распознавание и преобразование в HTML таблиц сложной формы (как на рис.3). Решение этой задачи, по существу, и было самым сложным при создании программы.

Об используемом алгоритме, который уверенно справляется с таблицами самой сложной формы, я и собираюсь здесь рассказать.

Первый этап

На первом этапе происходит просмотр исходного текстового файла, определение строк текста, подлежащих дальнейшему анализу, и построение списка этих строк. В программе этим занимается функция def_text_hl.

Список строк текста (strings) представляет собой массив, каждый элемент которого описывает одну строку и содержит смещение от начала файла, по которому находится эта строка, и длину строки (с учетом символов табуляции).

При анализе текста строки, не содержащие псевдографических символов, используемых для изображения таблиц, отбрасываются. Когда в тексте встречается одна или несколько идущих друг за другом таких строк, то в список строк просто добавляется элемент, соответствующий пустой строке (длина которой равна нулю) — только для того чтобы отделить друг от друга строки с псевдографикой, расположенные до и после отброшенной группы строк.

Максимальное количество элементов в списке строк ограничено числом 5000 (определяется константой MAX_STRINGS). Но за счет того что строки текста, точно не относящиеся к таблицам, отбрасываются, в исходном тексте может быть и больше 5000 строк.

Итак, мы имеем список, в котором содержится информация о строках исходного текста, подлежащих дальнейшему анализу, и о добавленных пустых строках. Совокупность описанных в этом списке строк будет представлять собой текст, подлежащий дальнейшему анализу.

Для дальнейшей работы с этим текстом удобно представить его следующим образом. Во-первых, заменить все табуляции на соответствующее количество пробелов. Во-вторых, сделать все строки текста одинаковой длины, для чего дополнить короткие строки пробелами до длины самой длинной строки текста. Тогда текст будет иметь прямоугольную форму, и можно говорить о таких величинах как ширина текста (text_l) и высота текста или количество строк (text_h). В-третьих, введем систему координат: начало координат (0,0) соответствует символу, расположенному в левом верхнем углу текста, координата y показывает номер строки (от 0 до text_h-1), а координата x — позицию символа в этой строке (от 0 до text_l-1).

Реально в программе текст, подлежащий дальнейшему анализу, нигде специально не хранится, и никаких манипуляций с ним не производится. Вместо этого имеется функция getc_xy, позволяющая получить символ этого текста по его координатам x и y. Эта функция использует описанный выше список строк текста и исходный текстовый файл.

Ошибочные ситуации, которые могут возникнуть на первом этапе:

- длина строки превысила 65535 символов;
- количество строк с псевдографикой больше 5000 (MAX_STRINGS);
- размеры текста, подлежащего дальнейшему анализу, позволяют сделать вывод об отсутствии в нем таблиц (чтобы была самая маленькая таблица, текст должен иметь размеры хотя бы 2x2).

Табл. 1

Элемент ячейки	Псевдографические символы, которые могут использоваться для его изображения
Левый верхний угол	┬ ├ ┼ ╞ ╟ ╓ ╥ ╠ ╬ ╤ ╥ ┌ ╒ ╫ ╪ ┌
Верхняя сторона	┴ ─ ╨ ═ ╧ ╨
Правый верхний угол	┤ ╡ ╢ ╖ ╕ ╣ ╗ ┐ ┬ ┼ ╥ ╬ ╤ ╥ ╫ ╪
Правая сторона	│ ‖ ├ ╞ ╟ ╠
Правый нижний угол	┤ ╡ ╢ ╣ ╜ ╝ ┘ ┴ ┼ ╨ ╬ ╧ ╨ ╫ ╪ ┘
Нижняя сторона	┬ ─ ╥ ═ ╤ ╥
Левый нижний угол	└ ┴ ├ ┼ ╞ ╟ ╙ ╨ ╠ ╬ ╧ ╨ ╙ ╟ ╪ ╪
Левая сторона	│ ┤ ╡ ╢ ╣ ╟

Второй этап

Распознавать всю таблицу сразу — слишком сложно, так как таблица может быть самой "странной" формы. Поэтому распознавание таблиц начинается с более простой задачи — распознавания отдельных ячеек.

Что мы знаем о ячейках? Каждая ячейка ограничена псевдографической рамкой прямоугольной формы, причем для изображения каждого элемента ячейки могут использоваться только определенные псевдографические символы (**табл.1**).

Для каждого элемента ячейки в программе имеется своя функция (check_left_up, check_up, ..., check_left), определяющая, может ли символ, подаваемый на вход, быть использован для изображения этого элемента (т.е. содержится ли этот символ в соответствующей строке табл.1).

Распознаванием ячеек занимается функция find_all_cells. Для всех i от 0 до text_l-2 и для всех j от 0 до text_h-2 она проверяет (с помощью функции check_cell), имеется ли в тексте ячейка, левый верхний угол которой находится по координатам (i,j). Если да, то координаты этой ячейки (т.е. координаты ее левого верхнего и правого нижнего угла) заносятся в массив cells.

При распознавании ячеек текст просматривается слева направо и сверху вниз. Соответственно, в таком же порядке в массив cells помещается информация о ячейках.

Ошибочные ситуации, которые могут возникнуть на втором этапе:

- в тексте не обнаружено ни одной ячейки — значит, текст не содержит таблиц;
- количество обнаруженных ячеек оказалось больше максимально допустимого (5000 — определяется константой MAX_CELLS).

Если второй этап завершился успешно, количество обнаруженных ячеек печатается на экране.

(Продолжение следует)

А.СИЗЕНКО,
Украина, г.Луганск.
E-mail: pat_2000@mail.ru

Многотабличные документы в программе "1С: Бухгалтерия"

В среде "1С Бухгалтерия" документ может иметь только одну табличную часть или не иметь ее вообще. Однако на практике может возникнуть необходимость наличия в документе двух, трех, а то и более разномерных таблиц с произвольным доступом и типом обработки действия пользователя. Как это сделать косвенным путем и рассказывается в этой статье.

В документе используем поле ввода "ТаблицаЗначений" (не сохраняемый в информационной базе объект). Задаем идентификатор, а в закладке "Дополнительно" — имя функции при выборе таблицы значений. Таким образом рисуем N-е количество таблиц, при необходимости прибегая к дополнительным слоям.

При закрытии документа все данные из поля ввода не запоминаются, поэтому их нужно сохранять в табличной части документа при закрытии и восстанавливать при открытии.

Описываем табличную часть документа по максимальному количеству того или иного типа данных, иными словами, приводим данные таблиц "к общему знаменателю". Например, пусть у нас будут две таблицы — Т30 (от "ТаблицаЗначений") и Т31 с соответствующими функциями Т30() и Т31(), имеющие следующие типы данных (Т30 — **табл.1**; Т31 — **табл.2**):

Тогда табличная часть документа будет иметь четыре реквизита (**табл.3**) плюс "код" для различия.

Также желательно сделать невидимое поле ввода для имитации внесения изменений в документ при записи в таблицу.

Используем предопределенные процедуры в модуле "Фор-

Табл. 1

Идентификатор	Тип	Количество знаков	Количество знаков после запятой
ВидЗатрат0	Справочник.ВидыЗатрат	-	-
ВидЗатрат1	Справочник.ВидыЗатрат	-	-
Число0	Число	10	2

Табл. 2

Идентификатор	Тип	Количество знаков	Количество знаков после запятой
ВидЗатрат0	Справочник.ВидыЗатрат	-	-
Число0	Число	10	2
Число1	Число	10	2

Табл. 3

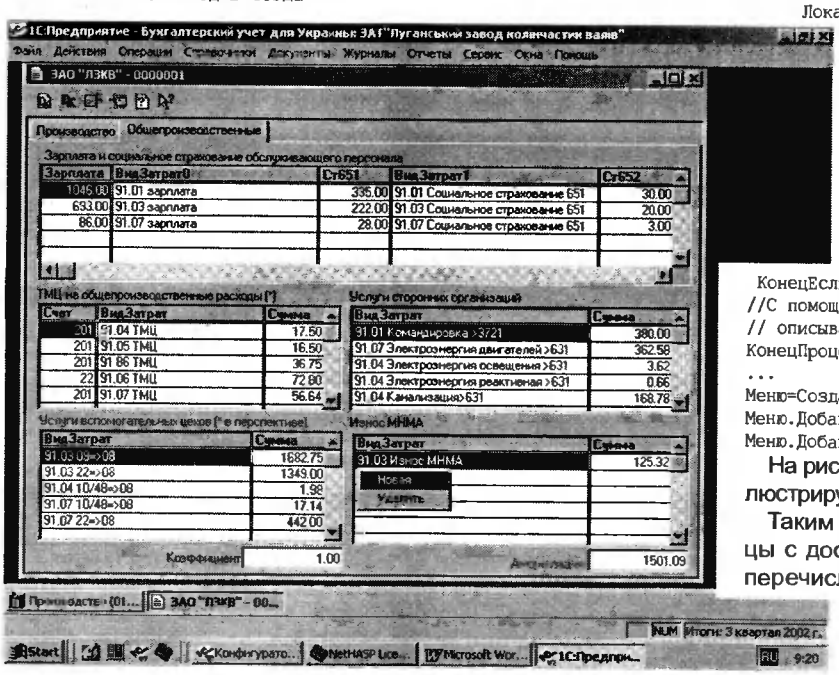
Идентификатор	Тип	Количество знаков	Количество знаков после запятой
ВидЗатрат0	Справочник.ВидыЗатрат	-	-
ВидЗатрат1	Справочник.ВидыЗатрат	-	-
Число0	Число	10	2
Число1	Число	10	2

маДокумента", т.е. процедуры, в которые система "ломится" при открытии и закрытии документа:

```
Процедура ПриОткрытии()
...
//Инициализируем колонки таблиц
ТЗ0.НоваяКолонка("ВидЗатрат0","Справочник.ВидыЗатрат",,
                 "Заголовок 0");
ТЗ0.НоваяКолонка("Число0","Число",10,2,"Заголовок 1");
ТЗ0.НоваяКолонка("Число1","Число",10,2,"Заголовок 2");
// Все параметры команды "НоваяКолонка" см.в документации

//Можно скрыть некоторые колонки: ВидимостьКолонки()
ТЗ1.НоваяКолонка("ВидЗатрат0","Справочник.ВидыЗатрат",,
                 "Заголовок 0");
ТЗ1.НоваяКолонка("ВидЗатрат1","Справочник.ВидыЗатрат",,
                 "Заголовок 1");
ТЗ1.НоваяКолонка("Число0","Число",10,2,"Заголовок 2");

//Выбираем табличную часть документа и восстанавливаем
ВыбратьСтроки()
Пока ПолучитьСтроку()=1 Цикл
        Если Код=1 Тогда
                ТЗ0.НоваяСтрока;
                ТЗ0.ВидЗатрат0=ВидЗатрат0;
                ТЗ0.Число0=Число0;
                ТЗ0.Число1=Число1;
        ИначеЕсли Код=2 Тогда
                ТЗ1.НоваяСтрока;
                ТЗ1.ВидЗатрат0=ВидЗатрат0;
                ТЗ1.ВидЗатрат1=ВидЗатрат1;
                ТЗ1.Число0=Число0;
        КонецЕсли;
КонецЦикла;
...
КонецПроцедуры

Процедура ПриЗакрытии()
...
//При записи выбираем последовательно строки "ТаблицЗначений"
// и заносим в табличную часть документа для сохранения.
//(при желании можно отсортировать, например:
// ТЗ0.Сортировать("+ВидЗатрат0,+Число0");
ТЗ0.ВыбратьСтроки;
Пока ТЗ0.ПолучитьСтроку()=1 Цикл
        Код=1;
        ВидЗатрат0=ТЗ0.ВидЗатрат0;
        Число0=ТЗ0.Число0;
        Число1=ТЗ0.Число1;
КонецЦикла;

ТЗ1.ВыбратьСтроки;
Пока ТЗ1.ПолучитьСтроку()=1 Цикл
        Код=2;
        ВидЗатрат0=ТЗ1.ВидЗатрат0;
        ВидЗатрат1=ТЗ1.ВидЗатрат1;
        Число0=ТЗ1.Число0;
КонецЦикла;
...
КонецПроцедуры
```

Далеее описываем функции обращения к "ТаблицеЗначений". В простейшем случае, для работы со строками таблицы, диалог с пользователем организуем в форме меню с двумя строчками — "Новая" и "Удалить":

```
ПеремМеню;
...
Процедура ТЗ0();
//Например, так
        ЛокальнаяПеременная=0;
        Если ТЗ0.КоличествоСтрок()<>0 Тогда
                Меню.ВыбратьЗначение(,,ЛокальнаяПеременная,,1);
        КонецЕсли;

        Если (ТЗ0.КоличествоСтрок()=0) или (ЛокальнаяПеременная =1)
        Тогда
// Флаг - невидимое поле ввода, необходимое для имитации
// внесения изменений в документ
          Флаг=0;
//
          ТЗ0.НоваяСтрока();
          ЛокальнаяПеременная =СоздатьОбъект("Справочник.ВидыЗатрат");
          ЛокальнаяПеременная.Выбрать("Здесь напишем
          поясняющий текст №1",);
          ТЗ0.ВидЗатрат0= ЛокальнаяПеременная.ТекущийЭлемент();
          ЛокальнаяПеременная =0;
          ВвестиЧисло(ЛокальнаяПеременная," Здесь напишем
          поясняющий текст №2",10,2);
          ТЗ0.Число0= ЛокальнаяПеременная;
          ВвестиЧисло(ЛокальнаяПеременная," Здесь напишем
          поясняющий текст №3",10,2);
          ТЗ0.Число1= ЛокальнаяПеременная;
          ИначеЕсли ЛокальнаяПеременная=2 Тогда
          ТЗ0.УдалитьСтроку();
        КонецЕсли;
//С помощью команд ТекущаяКолонка() и ТекущаяСтрока() можно
// описывать более сложные варианты
КонецПроцедуры
...
Меню=СоздатьОбъект("СписокЗначений");
Меню.ДобавитьЗначение("Новая");
Меню.ДобавитьЗначение("Удалить");
```



На рисунке приведен, так сказать, "пример из жизни", иллюстрирующий описанную методику.

Таким же образом можно создать: статические таблицы с доступом к определенным ячейкам; многомерные перечисления, динамически изменяющиеся таблицы, таблицы с вложенной визуализацией, т.е. когда выбор из одной влияет на отображение и выбор из другой или других таблиц, и т.п.

Замена микросхем в фирменном программаторе

С.РЮМИК,
г.Чернигов.

Каждому рассуждению противостоит равносильное.
Протагор

При выборе типа микроконтроллера для своей схемы разработчик в первую очередь интересуется доступностью средств отладки и программирования. В частности, сложно ли приобрести или самостоятельно изготовить программатор. Разные фирмы-изготовители по-разному решают этот вопрос — одни рекомендуют приобретать только фирменные программаторы у официальных дилеров, другие размещают в Интернете бесплатные схемы и программы к ним.

Разумеется, для обычного пользователя (радиолюбителя) последний вариант наиболее приемлем. "Бесплатные" программаторы, как правило, имеют незатейливый интерфейс управления, зато хорошую надежность в работе. Их несложные схемы легко выполняются на макетной плате в течение нескольких вечеров. Одна проблема — в конструкции могут использоваться дефицитные зарубежные микросхемы, не имеющие прямых отечественных аналогов. Кроме того, в схемах встречаются мелкие недочеты, затрудняющие наладку устройства.

В качестве примера рассмотрим программатор фирмы "Atmel" для популярных микроконтроллеров серии AT89C51, AT89LV51, AT89C52, AT89LV52, AT89C1051, AT89C2051, AT89S8252. Электрическая схема программатора распространяется бесплатно. Она размещена на фирменном сайте [1] вместе с пакетом обслуживающих программ ftp://www.atmel.com/pub/atmel/apcpgm.exe.

Программатор подключается к LPT-порту IBM-совместимого компьютера через гибкий ленточный кабель длиной не более трех футов (91 см), реально — вдвое больше. В компьютере должен быть установлен расширенный режим работы LPT-порта — "ECP/EPP" (выбирается в меню конфигурации компьютера при включении питания).

Программатор содержит девять зарубежных микросхем семи разных типов и два транзистора, причем не все из них имеют прямые аналоги, производимые в странах СНГ. Тем не менее, подобрать замену им можно, в подтверждение чему на **рисунке** приведена электрическая схема "самодельного" программатора.

Отличия предлагаемого устройства от фирменного:

1. Введены "антизвонные" резисторы R5...R16, обычно располагаемые внутри корпуса вилки XP1. Это стандартное решение при использовании в программаторе микросхем серии КР1533. Резисторы могут быть заменены перемычками.

2. Микросхемы DD6 (74LS541) и DD7 (74LS368) заменены их непрямыми аналогами — соответственно микросхемами К555АП5 и КР1533АП3 (см. **таблицу**).

3. Введен светодиодный индикатор HL1, по которому можно судить как о наличии напряжения питания (пониженная яркость), так и о начале процесса программирования 12 В (повышенная яркость).

4. Панелька XS1 (20 выводов с прижимом) заменена более распространенной на 24 вывода. Микроконтроллер устанавливается в эту панельку так, чтобы его вывод 1 совпал с первым контактом панельки XS1.

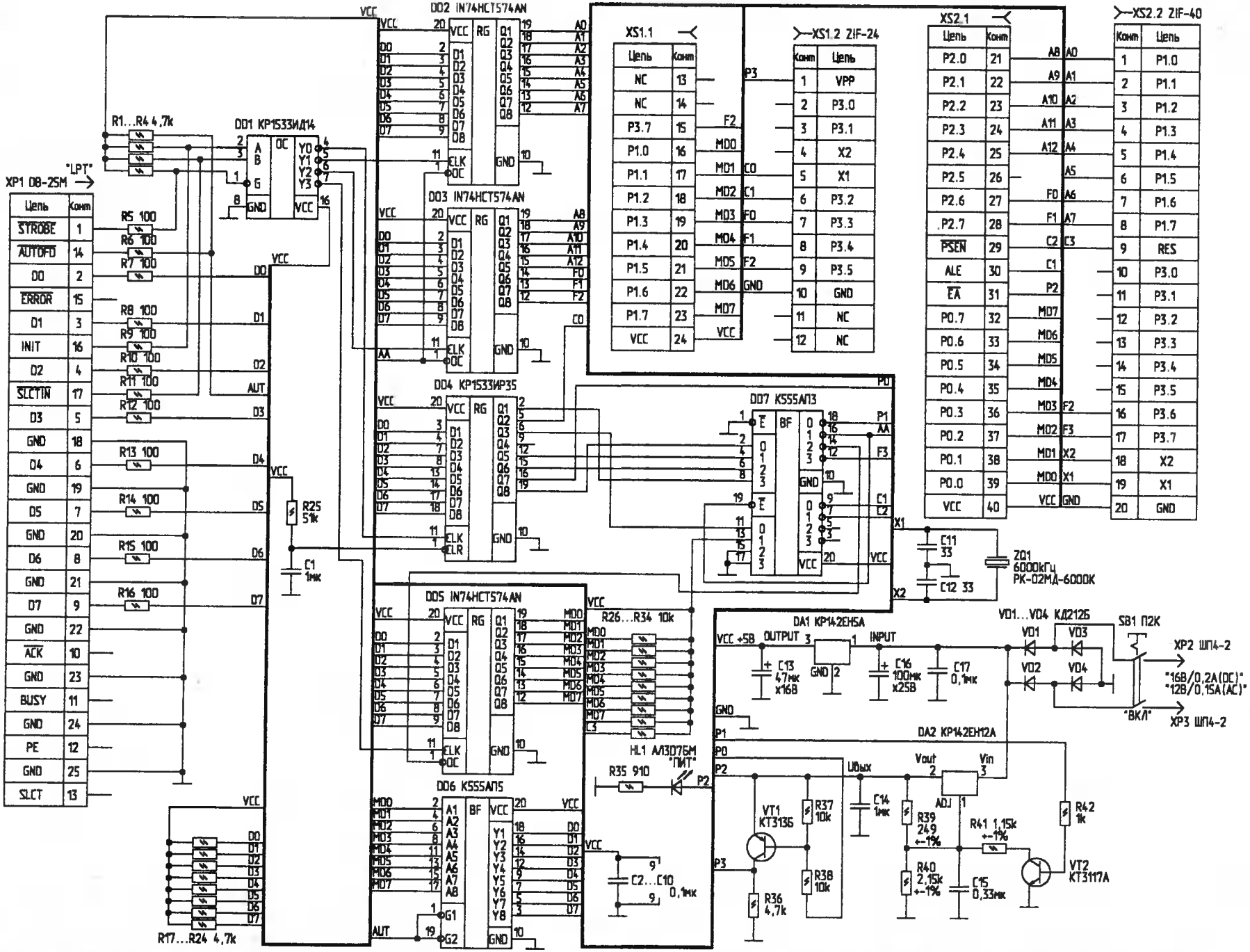
5. Резисторные сборки заменены обычными резисторами.

6. Уменьшен номинал конденсатора C15 с 10 мкФ до 0,33 мкФ.

На последнем изменении хотелось бы остановиться подробнее. Конденсатор C15 емкостью 10 мкФ входит в типовую схему включения стабилизатора DA2 LM317 [2]. Его установка уменьшает пульсации выходного напряжения, но, главное, снижает кратковременные "просадки" питания при переходных процессах. В схеме программатора стабилизатор DA2 обеспечивает выдачу напряжения программирования $U_{вых}$, которое для одних типов микроконтроллеров составляет 5 В, а для других — 12 В.

В ходе экспериментов с программатором было установлено, что на "быстрых" типах компьютеров запись в первую ячейку FLASH-памяти "12-вольтовых" микроконтроллеров блокируется, то есть код в ней постоянно равен FFh. Рабочая версия причины — низкая скорость нарастания напряжения $U_{вых}$ до величины 12 В, из-за которой программирование первых ячеек памяти происходит при пониженном питании, что недопустимо.

Обозначение	"Фирменное" название	Функция, параметры	Прямой аналог	Косвенный аналог (отличия)
DD1	74LS139 (ТТЛ)	Два дешифратора демультиплексора 2 в 4	-	КР1533ИД14 (другая серия ТТЛ)
DD2, DD3, DD5	74HCT574 (КМОП)	Восьмиразрядный регистр с тремя состояниями на выходв	IN74HCT574AN (НПО "Интеграл")	КР1533ИР37 (ТТЛ)
DD4	74LS273 (ТТЛ)	Восьмиразрядный регистр с установкой в "ноль"	К555ИР35	КР1533ИР35 (другая серия ТТЛ)
DD6	74LS541 (ТТЛ)	Восьмиразрядный буфер с тремя состояниями на выходе	К555АП13	К555АП5 (другая цоколевка выводов)
DD7	74HCT368 (КМОП)	Шесть инверторов с тремя состояниями на выходе	КР5564ЛН9, КР1564ЛН9	КР1533ЛН7 (ТТЛ), К555АП3 (ТТЛ, другая цоколевка выводов)
DA1	LM7805	Стабилизатор положительного напряжения +5 В	КР142ЕН5А	142ЕН5А (планарный корпус)
DA2	LM317	Регулируемый стабилизатор положительного напряжения	КР142ЕН12А, КР142ЕН12Б	К142ЕН12 (планарный корпус)
VT1	2N2222A (n-p-n)	$U_{кэ}$=40 В; $I_к$=0,8 А; $P_к$=0,5 Вт; $h_{21э}$>75; F_{max}=300 МГц	-	КТ608Б, КТ3117А, КТ3117Б, КТ3102Б, КТ315Г
VT2	2N2907 (p-n-p)	$U_{кэ}$=40 В; $I_к$=0,6 А; $P_к$=0,4 Вт; $h_{21э}$=100...300; F_{max}=200 МГц	-	КТ686А, КТ644В, КТ685В, КТ313Б, КТ3107Б, КТ361Г


XP1 DB-25M "LPT"
Цепь Конт
STROBE 1
AUTOFD 14
D0 2
ERROR 15
D1 3
INIT 16
D2 4
SLCTIN 17
D3 5
GND 18
D4 6
GND 19
D5 7
GND 20
D6 8
GND 21
D7 9
GND 22
ACK 10
GND 23
BUSY 11
GND 24
PE 12
GND 25
SLCT 13
R1...R4 4,7k
R5 100
R6 100
R7 100
R8 100
R9 100
R10 100
R11 100
R12 100
R13 100
R14 100
R15 100
R16 100
R17...R24 4,7k
R25 5k
C1 1мк
DD1 КР1533ИД14
DD2 IN74HCT574AN
DD3 IN74HCT574AN
DD4 КР1533ИР35
DD5 IN74HCT574AN
DD6 К555АП5
DD7 К555АП3
R26...R34 10k
HL1 АЛ307БМ "ПИТ"
R35 910
C2...C10 0,1мк
XS1.1
Цепь Конт
NC 13
NC 14
P3.7 15
P1.0 16
P1.1 17
P1.2 18
P1.3 19
P1.4 20
P1.5 21
P1.6 22
P1.7 23
VCC 24
XS1.2 ZIF-24
Конт Цепь
1 VPP
2 P3.0
3 P3.1
4 X2
5 X1
6 P3.2
7 P3.3
8 P3.4
9 P3.5
10 GND
11 NC
12 NC
XS2.1
Цепь Конт
P2.0 21
P2.1 22
P2.2 23
P2.3 24
P2.4 25
P2.5 26
P2.6 27
P2.7 28
PSEN 29
ALE 30
EA 31
P0.7 32
P0.6 33
P0.5 34
P0.4 35
P0.3 36
P0.2 37
P0.1 38
P0.0 39
VCC 40
XS2.2 ZIF-40
Конт Цепь
1 P1.0
2 P1.1
3 P1.2
4 P1.3
5 P1.4
6 P1.5
7 P1.6
8 P1.7
9 RES
10 P3.0
11 P3.1
12 P3.2
13 P3.3
14 P3.4
15 P3.5
16 P3.6
17 P3.7
18 X2
19 X1
20 GND
C11 33
C12 33
ZQ1 6000кГц РК-02МД-6000К
DA1 КР142ЕН5А
C13 47мк x16В
C16 100мк x25В
C17 0,1мк
VD1...VD4 КД212Б
SB1 П2К
XP2 ШП4-2
"16В/0,2А(DC)"
"12В/0,15А(AC)"
"ВКЛ"
XP3 ШП4-2
DA2 КР142ЕН12А
VT1 КТ3136
VT2 КТ3117А
R36 4,7k
R37 10k
R38 10k
R39 249 +-1%
R40 2,15k +-1%
R41 1,15k +-1%
R42 1k
C14 1мк
C15 0,33мк

Проведем расчет. В исходном состоянии транзистор VT2 открыт, выходное напряжение $U_{вых}$ микросхемы DA2 составляет 5 В

($U_{вых}$[В]=1,25[В]*(1+R41*R40/(R39*(R41+R40)))=1,25*(1+1,15*2,15/(0,249*(1,15+2,15)))=5 В).

При программировании транзистор VT2 закрывается, и напряжение $U_{вых}$ повышается до 12 В

($U_{вых}$[В]=1,25[В]*(1+R40/R39)=1,25*(1+2,15/0,249)=12 В).

Благодаря наличию конденсатора C15, напряжение $U_{вых}$ меняется не скачком, а плавно, в течение нескольких миллисекунд. Этого времени "быстрому" компьютеру достаточно, чтобы обратиться к одной или нескольким первым ячейкам памяти, и они оказываются незапрограммированными. Чтобы увеличить скорость нарастания напряжения $U_{вых}$, следует уменьшить номинал конденсатора C15. На практике такое изменение полностью восстанавливает программируемость всех ячеек памяти.

"Самодельный" программатор, собранный из исправных деталей, в наладке не нуждается. Панельки XS1, XS2 лучше использовать "многократные", со специальным прижимом. При небольшом числе программируемых микросхем можно применить обычные панельки, но желательно — цанговые, с круглыми гнездами.

Для питания программатора необходим источник постоянного напряжения 16 В/0,2 А или источник переменного напряжения 12 В/0,15 А (сетевой трансформатор). И в том, и в другом случае питание подается на разъемы XP2, XP3 без соблюдения полярности. Также можно применить блок питания от игровой приставки "Sega Mega Drive-II". В этом случае отпадает необходимость в диодном мосте VD1...VD4, а вместо XP2, XP3 следует поставить один разъем для подключения блока питания Если сетевое напряжение постоянно занижено (190 В), то питать устройство следует через автотрансформатор ЛАТР или телевизионный сетевой стабилизатор.

Микросхемы могут быть заменены аналогичными из **таблицы**, разумеется, с соответствующей коррекцией схемы в случае другой цоколевки выводов. Нет ограничений для применения смешанной комплектации, состоящей из зарубежных микросхем и их прямых и непрямых аналогов.

Транзисторы VT1, VT2 — высокочастотные, малой или средней мощности. Диоды VD1…VD4 должны быть рассчитаны на ток не менее 0,2 А. Если программатор предполагается использовать только для 40-выводных микроконтроллеров AT89C51, AT89C52, то панелька XS1, транзистор VT1 и резисторы R36…R38 могут отсутствовать.

Конденсаторы C11, C12 и кварцевый резонатор ZQ1 должны располагаться как можно ближе к контактам 18 и 19 панельки XS2.

Фирменное программное обеспечение распаковывается в отдельную директорию. Сначала из файла "apcpgm.exe" извлекаются файлы "read1.me!" и "pcpgm.exe", а затем из последнего — еще 32 файла. Среда функционирования — DOS, для пользователей Windows следует выбрать "Пуск — Завершение работы — Перезагрузить компьютер в режиме MS-DOS".

Поясняющая информация содержится в текстовых файлах *.txt и "исходниках" на языке C. Смена типов микросхем производится через выбор разных исполняемых файлов, например, для контроллеров серии 2051 — это файл "pc2051.exe", для контроллеров серии 89C51 — "pc51-3.exe" и т.д.. Применение данного пакета программ дает определенную гарантию того, что алгоритм программирования будет выдержан без отклонений от рекомендаций разработчиков фирмы "Atmel".

Для ускорения запуска программы можно создать *.bat-файл, который будет содержать одну строку, наподобие "pc52-3.exe 1", где "pc52-3.exe"— имя файла программы обслуживания микроконтроллера (в данном случае, AT89C52), а цифра "1" — номер LPT-порта, к которому подключается программатор.

Процесс программирования на удивление простой и занимает не более одной...двух минут. Все сводится к нажатию клавиш из высвечиваемого на экране меню. Вначале производится идентификация микросхемы (Read Signature). У каждой серии микросхем имеется свой набор сигнатурных кодов, указываемых в справочных данных Datasheet. При получении кодов, отличных от FFh, считается, что программатор и установленная в панельку микросхема в первом приближении исправны. На следующем этапе производится очистка FLASH-ПЗУ (Chip Erase), затем проверка на чистоту (Blank Check), далее программирование кодами из файла, находящегося в той же директории (Program from File), проверка правильности программирования (Verify against File), установка одного или нескольких битов защиты (Write Lock Bit). Из дополнительных возможностей — сохранение бинарного файла прошивки на диск (Save to File).

Литература

1. Using a Personal Computer to Program the AT89C51/C52/LV51/LV52/C1051/C2051. — Atmel, 1997…2000, C.5-6…5-8, http://www.atmel.com/atmel/acrobat/doc0285.pdf.

2. Интегральные микросхемы: Микросхемы для линейных источников питания и их применение. — М.: ДОДЭКА, 1998.

Ну очень удобная кнопка!

С.ШИРКО,
E-mail:S_Shirko@tut.by

Чтобы слова не расходились с делом, нужно молчать и ничего не делать

Лень — двигатель прогресса! Именно благодаря ей на свет появились кофеварка, автоматическая стиральная машина, пульт ДУ к телевизору… Не так давно создали робота, который таскается за вами и открывает бутылки с пивом. В общем, все направлено на то, чтобы при минимуме мускульной работы достичь поставленной цели. В нашем случае цель несложная — включение/выключение компьютера нажатием одной кнопки.

Согласитесь, что в большинстве случаев, помимо включения системного блока и монитора, требуется еще включать колонки, внешний модем, принтер, сканер и другие периферийные устройства. Не меньше усилий нужно затратить и на выключение ПК. Чтобы упростить задачу, многие покупают дешевые сетевые фильтры с множеством розеток и встроенной кнопкой "Вкл/Выкл". От помех они защищают слабо (чаще всего из-за отсутствия заземления), однако "переноска с кнопкой" из них получается неплохая. Хотя, с другой стороны, чтобы включать и выключать компьютер при помощи кнопки

на фильтре, нужно, чтобы последний был в легко доступном месте — на столе, к примеру. Соответственно, силовые кабели, идущие от "системника" и периферийных устройств, будут захламлять рабочее место. Кроме того, если корпус — стандарта ATX, то нажатие кнопки на фильтре не включит компьютер, а переведет его в спящий режим. В общем, это не слишком удобно, а потому перейдем к рассмотрению вопроса: "Как же все-таки включить (или выключить) компьютер и необходимые периферийные устройства нажатием одной кнопки?".

Проще всего поставленный вопрос может решаться владельцами корпусов стандарта AT. У большинства из корпусов на задней стенке имеется специальный разъем для подключения питания монитора. Разъем этот включен в сеть 220 В последовательно с выключателем питания "системника". Это позволяет нажатием кнопки "Power" включать/выключать как системный блок, так и монитор. Естественно, этим свойством можно воспользоваться и для включения/выключения других периферийных устройств — достаточно подключить параллельно разъему питания монитора несколько розеток и включить в них необходимые устройства. Для этих целей в магазине можно приобрести достаточно симпатичную, как бы ее назвать, "переноску без шнура". Оная представляет собой набор из 3...6 розеток, смонтированных в одном, чаще всего прямоугольной формы, корпусе. Стоимость этих розеток не превышает 1...2 у.е. Отыскав "переноску", перейдем к собственно процессу ее подключения. Исходя из соображений удобства и эстетики, рекомендуется поступить следующим образом:

1. Разбираем "переноску" и убеждаемся, что все ее розетки соединены параллельно.

2. Разрезаем шнур питания монитора пополам.

3. Подключаем, соблюдая меры безопасности, оба конца разрезанного шнура к крайним противоположным розеткам "переноски" (при необходимости в ее корпусе просверливаются соответствующие отверстия).

4. Собираем "переноску" и подключаем шнур к системному блоку и монитору.

Все. Теперь при нажатии кнопки "Power" на системном блоке, будут включены все периферийные устройства, подключенные к розеткам.

Владельцы корпусов стандарта ATX могут подключить дополнительные розетки к разъему питания монитора способом, аналогичным подключению к AT-корпусу. Задача осложняется тем, что в большинстве недорогих ATX-корпусов входной силовой разъем и разъем питания монитора просто соединены параллельно, т.е. на них всегда присутствует напряжение 220 В (включение монитора, а точнее вывод его из спящего режима, осуществляется по сигналу с видеокарты). Чтобы исправить это досадное недоразумение, нужно разрезать один из проводов, идущих к разъему питания монитора, и в разрыв подпаять контакты реле (для продления жизни контактов, параллельно им желательно включить конденсатор 0,01 мкФ на 1000 В). В свою очередь, обмотку реле подключают к 12-вольтовому выходу блока питания. Само реле крепится в любом удобном месте корпуса или блока питания. В выключенном состоянии контакты реле разомкнуты, ни на монитор, ни на другие периферийные устройства напряжение не подается. При включении системного блока (кнопкой "Power" либо с клавиатуры), на обмотку реле подается напряжение 12 В, контакты замыкаются, монитор и необходимые периферийные устройства включаются. Выбираем в Windows опцию "Завершение работы" — и компьютер со всеми периферийными устройствами выключается. Удобно до ужаса!

Теперь остановимся на выборе необходимого реле. Напряжение срабатывания должно быть в пределах 9...11 В. Допустимый ток через контакты — не менее 2 А (а лучше — 3 А) при напряжении 250 В. Лично я для этих целей нашел реле польского производства (название, к сожалению, привести не могу) с допустимым током 5 А. В принципе, должно подойти отечественное КУЦ-1, хотя в его надежности есть повод усомниться. Подробнее о выборе реле и вариантах его подключения можно прочитать на www.ixbt.com/peripheral/atxpsuupgrade.html.

Несколько слов о нагрузке. Суммарная мощность подключаемых к разъему питания монитора устройств не должна превышать 220 Вт (как для AT-, так и для ATX-корпусов). Поэтому, при включении в розетки различной периферийной техники, необходимо учитывать ее потребляемую мощность (**см. таблицу**).

Устройство	Потребляемая мощность, Вт
Монитор 15"	90...110
Монитор 17"	110...130
Монитор 19"	130...150
Матричный принтер	около 20
Струйный принтер	около 20
Лазерный принтер	600...850
Планшетный сканер	около 20
Внешний модем	менее 8
Две колонки без сабвуфера	8...30
Две колонки с сабвуфером	30...50
Многоканальная АС	более 50

Из приведенных в таблице данных видно, что, помимо монитора, к разъему вполне можно подключить небольшие активные колонки, матричный или струйный принтер, планшетный сканер и внешний модем одновременно. Категорически не рекомендуется подключать лазерный принтер — во время печати он потребляет до 700 Вт и более. Могут возникнуть проблемы и с подключением мощной многоканальной акустической системы.

Вот, в принципе, и все. Извините, если во время экспериментов вы умудритесь что-нибудь "спалить", или, еще хуже, получить удар электрическим током. Я за это ответственность с себя снимаю, потому как у вас своя голова на плечах, и в ее обязанности входит отвечать за действия ваших же рук.

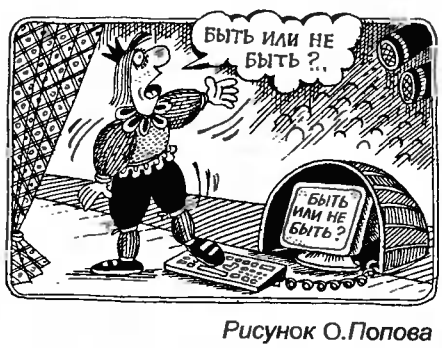


Рисунок О.Попова

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

Самая главная дискета, или что нужно для автономного плавания

В этой статье вернемся к вопросу, для чего нужна и какое значение для пользователя имеет загрузочная, или системная, дискета. При повреждении жесткого диска либо изменении его системных областей, компьютер может не загрузиться с жесткого диска, и все программное обеспечение, расположенное на нем, окажется недоступным пользователю. В таких случаях приходится загружать компьютер с загрузочной дискеты. Загрузочная дискета будет необходима и при заражении компьютера вирусами. В этом случае загружать операционную систему с жесткого диска и использовать расположенные на нем программы нельзя, так как при этом могут быть активированы вирусы. При заражении вирусами следует загрузиться с загрузочной дискеты и лечить компьютер с помощью записанных на дискеты или компакт-диски антивирусных программ. Загрузочная дискета будет нужна и при удалении старой или установке новой операционной системы. Только не надо жить по принципу "пока гром не грянет — мужик не перекрестится". Нужно создать дискету заблаговременно — тогда, когда все работает нормально. К любым неожиданностям надо быть подготовленным заранее и встретить их во всеоружии. Время, затраченное на создание загрузочной дискеты, несоизмеримо мало по сравнению с тем временем, которое уйдет на определение и устранение неисправностей и восстановление нормальной работоспособности компьютера.

Для того чтобы появилась возможность загрузки операционной системы с загрузочной дискеты, предварительно ее нужно создать. Для этого понадобится только один 3,5" гибкий диск 1,44 Мб, дистрибутив Windows 9x/Me и ваше желание. Создать загрузочный диск предлагается при установке операционной системы Windows 9x/Me, но, как правило, пользователи игнорируют эту возможность. Также создать загрузочную дискету в Windows 9x/Me можно, следуя по пути ***Пуск/ Настройка/Панель управления/Установка и удаление программ/Загрузочный диск***. Но лучше создать загрузочную дискету "вручную", самостоятельно, с учетом требований пользователя и с пониманием процесса создания загрузочной дискеты. И ваша дискета будет работать не хуже, а даже лучше стандартной.

Почему "ручной" вариант создания загрузочной дискеты лучше? Дело в том, что загрузившись со стандартной загрузочной дискеты, чувствуешь себя не очень комфортно — отсутствуют привычная файловая оболочка и курсор мыши. Возможна и такая ситуация, когда дополнительно к вышеперечисленным неудобствам добавляется и отсутствие доступа к приводу CD-ROM (это характерно для загрузочного диска OS Windows 95 и Windows 95 OSR2). Также отсутствует программное дисковое кеширование, нерационально используются ресурсы оперативной памяти. Все это и подвигло автора на создание загрузочной дискеты, которая была бы лишена этих недостатков.

Перед созданием загрузочного диска сначала нужно подготовить дискету. Под загрузочный диск надо использовать новую, отформатированную дискету, предварительно устроив ей полную проверку (с проверкой поверхности диска) Scandisk'ом. Ведь от ее надежности при хранении записанной информации будет зависеть, насколько успешно мы сможем произвести с нее загрузку операционной системы. Далее, загрузив файловый менеджер, например FAR, командой **sys C: A:** копируем системные файлы с жесткого диска на дискету. После этого копируем на дискету все необходимые файлы, которые будут нужны для загрузки операционной системы и могут понадобиться при "ремонтных работах". Список файлов, которые необходимо скопировать на загрузочную дискету, приводится ниже. Все файлы на дискете должны быть размещены в корневой директории. Заметьте, что для копирования с жесткого диска на загрузочную дискету системных файлов **io.sys** и **msdos.sys** нужно использовать команду **sys C: A:**, а все остальные файлы можно копировать обычной командой **copy**.

Для повышения производительности и ускорения обмена данными рекомендуется применять программное кеширование жестких дисков, приводов CD-ROM, а также гибких дисков, используя утилиту SmartDrive. Для этого нужно скопировать на загрузочную дискету файл **smartdrv.exe**, а потом в файле **autoexec.bat** включить команду запуска этой программы. Только необходимо учесть одно важное обстоятельство — для того чтобы произошло кеширование приводов CD-ROM, необходимо, чтобы утилита SmartDrive была запущена после программы MSCDEX.

Для удобства работы, на загрузочную дискету нужно дополнительно дописать следующие файлы:

- файловую оболочку Volkov Commander v4.0. Для минимальной конфигурации потребуются всего два файла — **vc.com** и **vc.ini** (вместе они занимают места чуть больше 64 Кб);
- DOS'осовский драйвер мыши **mouse.com**;
- драйверы, обеспечивающие физический доступ к приводу CD-ROM (можно применить драйверы от CD-ROM Samsung, они подходят ко многим приводам).

Если после загрузки компьютера с загрузочной дискеты вставить в дисковод A: другую дискету, например, с антивирусной программой, и попробовать ее запустить, то наверняка у вас из этого ничего не выйдет. На экране монитора появится сообщение, суть которого состоит в том, что операционная система не может найти файл **command.com** и просит вставить дискету с файлом **command.com** в дисковод A: Дело в том, что для выполнения какой-либо команды операционной системе требуется командный интерпретатор DOS **command.com**, и если система его не находит, она выдает соответствующее сообщение на экран. Для того, чтобы в дальнейшем избежать появления подобного сообщения, можно помещать копии файла **command.com** в корневой каталог дискет, которые вы будете использовать после загрузки компьютера с загрузочной дискеты (при наличии на них свободного места около 100 Кб). Есть и другой, более приемлемый вариант — скопировать командный интерпретатор DOS **command.com** на виртуальный электронный RAM-диск и указать операционной системе, какой файл использовать в качестве командного интерпретатора DOS, и где он размещен. Разумеется, можно и нужно сделать, чтобы все вышеперечисленные драйверы и программы загружались автоматически, без ввода их из командной строки. Для этого надо соответствующим образом скорректировать конфигурационные файлы **config.sys** и **autoexec.bat**.

Если на загрузочной дискете есть свободное место, то можно будет дописать на нее какие-либо другие дополнительные программы и драйверы, которые могут понадобиться в "автономном плавании", например, архиваторы. Вот перечень необходимого минимума файлов, который должен находиться на загрузочной дискете:

io.sys	himem.sys	fdisk.exe	country.sys	config.sys	mscdex.exe
msdos.sys	emm386.exe	format.com	smartdrv.exe	autoexec.bat	sscdrom.sys
command.com	keyb.com	scandisk.exe	mouse.com	vc.com	display.sys
sys.com	keybrd3.sys	scandisk.inl	ramdrive.sys	mode.com	ega3.cpi

Большинство вышеперечисленных файлов находятся в директориях C:\Windows и C:\Windows\Command.

Ниже приведены примеры файлов конфигурации **config.sys** и **autoexec.bat**, находящихся на загрузочной дискете.

Пример файла **config.sys**:

```
Device=himem.sys
DeviceHigh=emm386.exe ram
Dos=High,Umb
DeviceHigh=ramdrive.sys 2054 /e
DeviceHigh=display.sys con=(ega,,1)
Country=007,866,country.sys
DeviceHigh=sscdrom.sys /d:sscd000 /v
BuffersHigh=10
FilesHigh=20
StacksHigh=9,128
LastDriveHigh=F
```

Пример файла **autoexec.bat**:

```
@echo off
mode con cp prep=((866) ega3.cpi)>nul
mode con cp sel=866>nul
lh keyb ru,,keybrd3.sys
lh mscdex /d:sscd000
lh smartdrv A+ C+ /v
copy a:\command.com d:
set comspec=d:\command.com
lh mouse
lh vc
```

Если ваш жесткий диск уплотнен программой сжатия дисков DoubleSpace или DriveSpace, не забудьте включить в **config.sys** загрузку драйвера **DblSpace.sys** или **DrvSpace.sys** для обеспечения доступа к уплотненному диску и разместить на дискете файлы **DblSpace.bin** или **DrvSpace.bin**.

После того как все нужные файлы будут скопированы на загрузочную дискету, необходимо загрузиться с нее, и при необходимости откорректировать файлы конфигурации. После завершения создания загрузочного диска нужно произвести оптимизацию размещения файлов на дискете, т.е. выполнить ее проверку и дефрагментацию. Это ускорит загрузку операционной системы с дискеты и уменьшит излишний износ дисковода. Не забудьте подписать сверху на дискете "Загрузочная дискета", и указать имя диска, например: "System_disk". И в заключение хочу напомнить два важных правила:

1. На загрузочной дискете нужно обязательно установить защиту от записи. Это предохранит дискету от случайного изменения и удаления файлов, а также от заражения вирусами.

2. Надо иметь в виду, что дискеты — это очень ненадежное средство хранения информации. Поэтому загрузочную дискету лучше всего иметь в двух копиях.

О спросе на ПК, и не только...

Е.МУХУТДИНОВ,
Волгоградская обл.

В конце 2002 г. у меня была возможность пообщаться с продавцами компьютерных салонов и лицами, собирающими ПК на заказ в Волгоградской, Свердловской и Московской областях. Поэтому хочу поделиться информацией, которая может быть полезна желающим приобрести современный компьютер с тем условием, чтобы он не устарел слишком быстро (насколько это возможно).

Итак, среди современных компьютеров лидируют платформы на базе Pentium 4. Возникает впечатление, что интерес к линейке дешевого аналога Pentium 4 — процессорам Celeron — снижается. Лично у меня этому есть простое обьяснение — в настоящее время, благодаря рекламе, процессоры Pentium 4, в отличие от Celeron, "на слуху". В то же время, компьютеры все чаще покупают люди, мало искушенные в этой области, просто стало модным иметь ПК. Поэтому название Celeron не вызывает у них энтузиазма, несмотря на более низкую цену. Надо сказать, что если в процессе изготовления процессора Pentium не удается получить кристалл с необходимым размером L2-кеша, но можно получить процессор с кешем в два раза меньше, то так и поступают. В результате получается процессор Celeron.

Самое интересное, что у многих продавцов компьютеров дома имеются машины на базе процессоров не от Intel, а от AMD, таких как Athlon 1800XP+ и более мощных Athlon'ов. Действительно, фирма AMD дает возможность приобрести хороший функциональный аналог процессоров Pentium 4 по умеренной цене. Неискушенного пользователя настораживает лишь обеспечение теплового режима процессоров от AMD — ведь у них, в отличие от процессоров от Intel, отсутствует защита от перегрева, и остановка вентилятора или недостаточная его мощность приводят к выходу процессора из строя. Поэтому при покупке вентиляторов для Athlon'ов и Duron'ов ни в коем случае нельзя экономить на вентиляторе. Также надо сказать, что фирма AMD до сих пор использует P-rating, и частота, к примеру, процессора Athlon 1600 XP+ не 1600 МГц, а только 1400 МГц, хотя производительность, судя по знаку "+", несколько превышает производительность процессора типа Pentium 1600 (конечно, все зависит от применяемых тестов).

Что касается системных плат, то просматривается ориентация на чипсеты, поддерживающие память DDR SDRAM, как наиболее привлекательную по соотношению цена/качество.

Наиболее популярными видеоадаптерами являются карты на основе чипсета GeForce 2 MX400 с видеопамятью 64 Мб.

Особо надо сказать о винчестерах. Все продавцы компьютерных салонов отметили снижение качества жестких дисков от фирмы IBM (недавно новый винчестер IBM вышел из строя у моего знакомого). По поводу винчестеров Maxtor мнения разделились. Благозвучные слова были сказаны в адрес винчестеров Seagate.

Один из продавцов продемонстрировал мне жесткий диск Seagate, обратив особое внимание на наличие снизу металлической крышки, которая выполняет роль пассивного радиатора, отводя тепло от микросхем винчестера через изолирующую теплопроводящую прокладку. Это несомненно плюс, т.к. известно, что винчестеры чаще выходят из строя при высокой температуре окружающей среды в летнее время, и дополнительное охлаждение микросхем никогда не повредит. На данный момент больше всего приобретают винчестеры с интерфейсом IDE, объемом 40 Гб и скоростью вращения шпинделя 7200 об/мин.

Что касается приводов CD-дисков, то пока лидируют обычные CD-ROM несмотря на вполне доступные цены на приводы CD-RW, позволяющие не только читать диски CD-ROM, CD-R, но и записывать информацию на диски CD-R и CD-RW [1]. Свой привод CD-RW — "голый" SONY, китайской сборки, с характеристиками 24/10/40 и гарантией 6 месяцев,

я приобрел в августе 2002 г. в подмосковном Сергиевом Посаде всего за 65 USD.

Вполне доступны по цене и приводы DVD, но из-за дороговизны дисков DVD пока спрос на них невелик. Поэтому лучший выбор на данный момент — это CD-RW.

Если же вы приобретаете CD-ROM, то следует воспользоваться рекомендациями из [2] и не гнаться за слишком скоростными приводами. Исходя из того, что возможность чтения дисков CD-RW появляется при скорости примерно 16X, возможно, оптимальным будет выбор привода со скоростью 24X.

Что касается мониторов, то, по-прежнему, лидируют мониторы на электронно-лучевых трубках (ЭЛТ). Приобретение мониторов с диагональю 15 дюймов резко снизилось, лидируют 17-дюймовые мониторы. С приобретением жидкокристаллических мониторов продавцы компьютерных салонов советуют повременить пару лет, пока цены на них не станут более доступными.

От себя хочу заметить, что приобретение монитора — довольно ответственный шаг, и приобретать его самостоятельно, без соответствующего опыта, крайне нежелательно.

Исходя из вышесказанного, несмотря на то что все, связанное с компьютерами, устаревает очень быстро, можно утверждать, что основными "рабочими лошадками" в 2003 г. будут компьютеры на базе Pentium 4 и его эквивалентах — Celeron, Athlon, Duron. При покупке либо сборке нового современного компьютера следует избегать системных плат, не поддерживающих память DDR SDRAM, видеокарт с объемом видеопамяти менее 64 Мб, винчестеров объемом менее 40 Гб, слишком скоростных приводов CD-ROM, мониторов на основе ЭЛТ с диагональю экрана менее 17 дюймов.

Литература

1. А.Горячкин. Зачем император Нерон поджег Рим? — Радиомир. Ваш компьютер, 2002, №10, С.6.

2. С.Ширко. Взрыв CD-ROM'а. — Радиомир. Ваш компьютер, 2002, №9, С.37.

Кое-что о Windows 9x, и не только

Е.ЗАРЕЦКИЙ,
г.Чаусы, Могилевской обл.

(Продолжение. Начало в №4/2003)

При работе с Проводником нажатие комбинации клавиш <Shift>+<F10> — это то же самое, что и щелчок правой кнопкой мышки.

******

Если вы используете "детальный" вид в Проводнике (Explorer), то довольно часто видите только часть информации, так как остальное не помещается в колонке. Если одновременно нажать <Ctrl> и "серый плюс", то колонки автоматически настроятся на оптимальную ширину.

******

Если нажать кнопку "Пуск" (Start), выбрать пункт "Выполнить" (Run) и напечатать в строке точку, то откроется папка "Рабочий стол" (Desktop). А если напечатать две точки, то откроется папка Windows.

******

Если щелкнуть правой кнопкой по какому-нибудь exe-файлу и выбрать "Быстрый просмотр" (Quick View), то в разделе "Import Table" вы увидите, какие dll-файлы использует эта программа.

******

Если в командной строке 7-й DOS (той, которая ставится при инсталляции Windows 95) набрать dir/v, то вы увидите длинные имена файлов и папок.

******

Для того чтобы пользоваться длинными именами директорий и файлов в окне DOS, заключайте их в кавычки — C:/ "длинное имя директории".

******

Окно DOS "понимает" сетевые имена. Т.е. можно, например, напечатать `DIR//server/share` и получить список файлов. Команда CD не работает, но многие другие — работают. Например, `COPY, MOVE, REN, MD, RD ...`

******

Чтобы выделить в Word'е вертикальный блок текста, надо предварительно нажать комбинацию клавиш <Ctrl>+<Shift>+<F8>. Кроме того, выделить вертикальный блок можно с помощью мышки, удерживая клавишу <Alt>.

******

В Word'е, чтобы поставить дефис, нажмите клавишу <->, для короткого тире используйте комбинацию <Ctrl>+<Gray->, для длинного — <Ctrl>+<Alt>+<Gray-> (клавиша <Gray-> — это клавиша со знаком "-" на дополнительной цифровой панели клавиатуры). "Минус" в тексте лучше изображать в шрифте Symbol.

******

Проблемы с отображения русских шрифтов в Photo Shop? Исправляется мгновенно!

Проделайте следующее:

1. Откройте реестр (...Windows/Regedit.exe).
2. Откройте раздел `HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Nls\CodePage`.
3. Измените значение 1252: Для Windows NT — "C_1251.NLS", для Windows 9x —"cp_1251.nls".
4. Теперь перегрузите компьютер. Все шрифты будут работать нормально.

******

Пара интересных идей, вовремя "подброшенных" M00nk'ом, за что ему и спасибо.

Чтобы зарегистрировать воспроизведение видео-файлов через ActivMove, нужно:

1. В реестре в разделе HKEY_CLASSES_ROOT создать папки (разделы) с именами, соответствующими обрабатываемым расширениям (папки с именами "avi", ".mpg" и ".mpeg").
2. В том же разделе создать папку, например, с именем "Video Files". В ней установить в параметре "по умолчанию" название типа файлов, которое будет отображаться в выпадающем списке в окне ассоциации файлов Windows 98.
3. Создать в этой папке двоичный параметр "EditFlags" и заполнить его четырьмя нулевыми байтами.
4. Создать в этой папке новый раздел с именем "Shell", в котором создать раздел "Open".
5. В разделе "Open" создать двоичный параметр "EditFlags" и заполнить его значениями "01 00 00 00". В нем же создать раздел "command", в значение "по умолчанию" которого внести строку (без кавычек и с соблюдением пробелов!!!)

"rundll32.exe c:\windows\system\amovie.ocx,RunDll /play %1"

6. В разделы расширений (созданные в п.1) в значения "по умолчанию" внести название папки настроек (в нашем случае это "Vidoe Files") и добавить строковый параметр "Content Type" с описанием конкретного типа файла.
7. Машину можно не перезапускать :).

******

(Продолжение следует)

Архивация на Sinclair

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

(Окончание. Начало в №4/2003)

Далеко не у всех пользователей Sinclair-совместимой техники есть возможность эксплуатировать два дисковода. Поэтому было решено провести подобную группу тестов на одном накопителе. В качестве тестовой была определена выборка из файлов, участвовавших в предыдущем испытании, общим объемом около половины емкости дискеты. Меньший по сравнению с предыдущей группой тестов размер архивируемых данных был взят для того, чтобы можно было вести запись архива на исходный диск. Это, во-первых, должно было уменьшить погрешность тестирования, связанную с ручным "перевтыканием" дискет, а во-вторых — не все рассматриваемые архиваторы допускают функционирование в режиме сохранения архива на отдельном диске. Итак, **табл.2**.

Табл. 2. Графика (1026 секторов в 38 файлах, В → В)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	425	41,4	33:27	20:04	1,67
(fast)	486	47,4	5:18	3:44	1,42
(slow)	490	47,8	24:22	15:09	1,61
LZ-Compressor	473	46,1	19:15	12:23	1,55
Hrip	435	42,4	16:27	10:04	1,63

Относительные оценки эффективности остались, по большому счету, неизменными. Незначительное повышение плотности упаковки можно объяснить неоднородностью (большей "рыхлостью") использованной выборки файлов. Несколько хуже, по сравнению с использованием двух дисководов, показатели повышения производительности в режиме "turbo", но эти сотые доли вполне укладываются в погрешность, обусловленную, вероятнее всего, потерями времени на "катание" головок чтения-записи по разным областям одного и того же диска.

Еще одна группа тестов напросилась на реализацию буквально в самый последний момент. Дело в том, что из всевозможных технических новшеств для Sinclair-совместмых машин, которых за последние годы было придумано и реализовано немало, далеко не все выдержали проверку временем, и уж совсем немногие "перекочевали" в промышленные, серийные изделия. Одним из таких прижившихся на платформе "девайсов", на деле доказавшим свою полезность, является электронный квазидисковод.

Попытки использовать дополнительную оперативную память в качестве еще одного диска программным способом предпринимались довольно давно. Пожалуй, самый известный пример — файловая оболочка Honey Commander разных версий от фирмы MicroArt, изначально ориентированная на компьютеры ATM-turbo и ATM-turbo 2 с базовым объемом ОЗУ 512 Кб. Одна из функций этой очень даже неплохой оболочки позволяет организовать в дополнительной памяти ATM-клона RAM-диск объемом более 400 Кб. Существует также версия ассемблера GENS-4 по имени GENS-4 128K (автор — Wlodek Black, г.Москва), использующая эту расширенную память в качестве третьего диска емкостью 74 Кб. Одной из наиболее удачных попыток можно считать реализацию квазидисковода в файлере Real Commander (Real software, г.Брест, Беларусь), например, версии 1.96 — он грамотно определяет дополнительную память многих существующих клонов Sinclair и позволяет выбирать по усмотрению пользователя размер формируемого квазидиска в пределах доступного объема оперативной памяти.

В компьютере KAY-1024 возможность сформировать в расширенной памяти квазидисковод реализована программно-аппаратным способом — путем подачи прямой команды из TR-DOS. Такой виртуальный дисковод достаточно корректно эмулирует реальный "железный", если в программах используются только стандартные обращения к дисковой системе TR-DOS. Поэтому вызывает особый интерес тестирование работоспособности программ-архиваторов именно на квазидисководе. Практическую же пользу это может иметь в плане повышения производительности компьютера, так как а этом случае из процесса исключаются относительно медленные операции обмена данными с реальным дисководом.

Набор тестовых файлов — тот же, что и в предыдущем случае, поэтому результаты эффективности сжатия в **табл.3** не приводятся.

К большому сожалению, по итогам этой группы тестов "выпал в осадок" очень неплохой во всех других отношениях архиватор Hrip — ну "не видит" он в архитектуре компьютера квазидисковод, и все тут. Возможно, как уже говорилось, для повышения скорости работы с TR-DOS, в нем использованы какие-то не совсем стандартные обращения к дисковой системе, что и подвело его на этот раз.

Ну а главный вывод, следующий из результатов этого тестирования — реально подтвердился заявленный производителем компьютера KAY-1024 коэффициент турбирования в ОЗУ — в 1,7 раза. Некоторое недоумение, правда, вызвало ускорение работы ZXZIP "normal" — в 1,73 раза. Для исключения ошибки замеры были повторены по 4 (четыре!) раза. Основная рабочая версия на данный момент — скромность фирмы-производителя компьютера, просто округлившей коэффициент турбирования до десятых долей в меньшую сторону. Кстати, забегая вперед — похожий эффект наметился затем и в последующих сериях тестов, но там он не проявился столь наглядно.

Наверняка каждый синклерист, использующий свой компьютер не только как игровую приставку, имеет некоторую коллекцию текстов в электронном виде на дискетах. Это могут быть help'ы к системным и прикладным программам, описания прохождения сложных игр, заготовки рефератов и сочинений или какая-нибудь другая документация. Таким образом, задача экономичного хранения текстов выглядит более чем актуальной. Поэтому следующие проверки архиваторов были проведены с использованием как раз текстовой информации. В качестве тестовых были взяты файлы, содержащие почтовую переписку средней "рыхлости", в альтернативной кодировке, в формате iS-Editor. Архивировалась заполненная примерно на 90% дискета, результаты тестирования приведены в **табл.4**.

Итак, смотрим таблицу и диаграммы

Табл. 3. Графика (1026 секторов в 38 файлах, С → С)

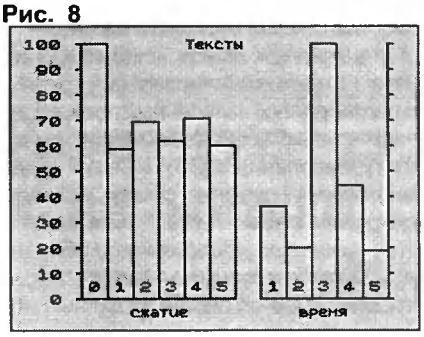
Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, свкторов	% от исходного			
ZXZIP (norm)	-//-	-//-	31:48	18:24	1,73
(fast)	-//-	-//-	3:38	2:09	1,69
(slow)	-//-	-//-	22:41	13:28	1,68
LZ-Compressor	-//-	-//-	16:18	9:39	1,69
Hrip	-	-	-	-	-

Табл. 4. Тексты (2351 сектор в 103 файлах, A → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	1385	58,9	26:45	17:12	1,56
(fast)	1642	69,8	15:08	10:25	1,45
(slow)	1467	62,4	74:25	45:47	1,62
LZ-Compressor	1661	70,6	33:29	22:08	1,51
Hrip	1419	60,4	13:56	9:09	1,52

на **рис.8**. Лучшие и довольно близкие друг к другу результаты по степени сжатия наблюдаются у архиватора ZXZIP в режимах "normal" и "slow", а также у Hrip. Несколько худшую, но тоже примерно одинаковую между собой эффективность упаковки с разницей менее одного процента демонстрируют другие программы и режимы. Самый лучший архиватор по степени сжатия — снова ZXZIP "normal", а самый быстрый по времени на этот раз — Hrip. Но если разница в упаковке между этими лидерами составляет всего полтора процента, то по скорости Hrip выигрывает у ZXZIP в два раза. LZ-Compressor выглядел посредственно как по степени упаковки, так и по времени работы. ZXZIP "slow" по причине запредельно большого времени, потраченного на архивацию, снова выглядел далеко не блестяще, замкнув собой цепочку конкурсантов.

Рис. 8



Оценки прироста производительности в турборежиме в основном близки к полученным при тестировании на графической информации, поэтому более подробно на них останавливаться не будем.

Теперь тестирование на одном дисководе, с выборкой части файлов из предыдущего тестрования (**табл.5**).

Итоги этой группы тестов в сравнении с полным набором текстовых файлов в основном близки или совпадают с аналогичными, полученными при испытаниях на графических файлах, с тем общим для этих типов информации отличием, что структура текстовых файлов по природе своей обычно менее "рыхлая", чем графических, и поэтому, как правило, тексты по сравнению с графикой сжимаются хуже.

Продолжаем "гонки" на квазидисководе (**табл.6**).

Как и ожидалось, в случае использования квазидисковода прирост производительности близок к предельному, заявленному фирмой (c)Nemo, и сходен с данными, полученными при испытаниях на графике.

Затем тестовыми послужили файлы, содержащие музыкальные фрагменты, написаные и откомпилированные с помощью различных музыкальных редакторов для звукового сопроцессора AY/YM. Общий объем этой звуковой информации составил около 94% от объема дискеты (**табл.7**).

По итогам этого тестирования (см. также диаграммы на **рис.9**) выделяются два ярко выраженных "чемпиона", показавших не только лучшую, но и практически одинаковую эффективность сжатия — ZXZIP "normal" и Hrip. Но Hrip при этом затратил вдвое меньше времени, с чем можно и поздравить Delirium Tremens Freedom. LZ-Compressor и в этот раз ничем особенным не выделился. Безусловный аутсайдер — ZXZIP "slow", для упаковки музыкальных данных его алгоритм работы оказался явно не из лучших как по степени архивирования, так и по использованному времени.

А теперь — тесты на одном дисководе (**табл.8**).

Разница в эффективности упаковки на одном дисководе здесь не такая однозначная, как при испытаниях с двумя дисководами, а также в предыдущих сериях тестов, но общая картина от этого практически не изменилась.

И снова квазидисковод (**табл.9**).

Комментарии, пожалуй, излишни.

И наконец, эффективность архивирующих программ была опробована на файлах, содержащих фрагменты реальных машинокодовых процедур (**табл.10**). "Не мудрствуя лукаво", их роль сыграли программы, содержащиеся на любимой системной дискете (примерно 84% от полного объема дискеты TR-DOS).

Гораздо более низкая эффективность архивации (табл.10 и диаграммы на **рис.10**), по сравнению с предыдущими тестированиями, обусловлена тем, что законченные программные продукты очень часто содержат в себе уже скомпрессированные кодовые блоки и прочие данные, что, естественно, не самым лучшим образом влияет на конечные

Табл. 5. Тексты (1277 секторов в 59 файлах, B → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	759	59,4	14:56	9:47	1,53
(fast)	914	71,6	8:58	6:19	1,42
(slow)	806	63,1	41:03	25:27	1,61
LZ-Compressor	910	71,3	18:35	12:20	1,51
Hrip	777	60,8	8:17	5:46	1,44

Табл. 6. Тексты (1277 секторов в 59 файлах, C → C)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	-//-	-//-	12:15	7:09	1,71
(fast)	-//-	-//-	6:13	3:39	1,70
(slow)	-//-	-//-	38:15	22:39	1,69
LZ-Compressor	-//-	-//-	13:57	8:16	1,69
Hrip	-	-	-	-	-

Рис. 9

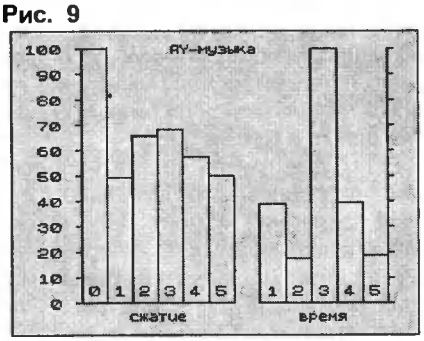


Табл. 7. AY-музыка (2386 секторов в 70 файлах, A → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	1191	49,9	31:27	19:34	1,61
(fast)	1576	66,0	14:06	9:36	1,47
(slow)	1634	68,5	79:39	48:31	1,64
LZ-Compressor	1375	57,6	31:52	20:38	1,54
Hrip	1194	50,0	14:59	9:29	1,58

Табл. 8. AY-Музыка (1190 секторов в 44 файлах, B → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	582	48,9	11:13	7:24	1,52
(fast)	806	67,7	7:47	5:27	1,43
(slow)	843	70,8	39:28	24:17	1,62
LZ-Compressor	670	56,3	15:34	10:13	1,52
Hrip	586	49,2	5:42	3:55	1,46

Табл. 9. AY-музыка (1190 секторов в 44 файлах, C → C)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	-//-	-//-	9:11	5:22	1,71
(fast)	-//-	-//-	5:40	3:20	1,70
(slow)	-//-	-//-	37:18	22:05	1,69
LZ-Compressor	-//-	-//-	1:58	7:06	1,68
Hrip	-	-	-	-	-

результаты работы архиваторов.

Здесь лучшие, как и в предыдущей серии — ZXZIP "normal" и Hrip (разница менее процента). И так же, как и в предыдущей серии, Hrip затратил на упаковку в два раза меньше времени, чем его напарник-соперник. ZXZIP "fast" отличился превосходным показателем времени и неважным архивирования. ZXZIP "slow" не блеснул ни тем, ни другим. LZ-Compressor по совокупности показателей оказался примерно на уровне ZXZIP "fast".

Результаты для системы с одним дисководом приведены в **табл.11**.

В выборке, использованной в ходе теста, есть только одно, хотя и немаловажное обстоятельство, мешающее однозначно поставить Hrip на первое место — не совсем корректная работа с дисковой системой TR-DOS, исключающая возможность использования электронного квазидисковода. Здесь уместно еще раз напомнить, что тестировалась версия архиватора 1.03. Вполне возможно, что более новые версии свободны от этого недостатка. К сожалению, за неимением оных, проверить это предположение у автора не было возможности.

Есть из этого монопольного сравнения и исключение. На файлах с графи-

Табл. 10. Машинные коды (2141 сектор в 52 файлах, A → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	1616	75,5	24:48	15:30	1,60
(fast)	1797	83,9	13:19	8:54	1,50
(slow)	1760	82,2	98:21	59:16	1,66
LZ-Compressor	1737	81,1	27:06	18:21	1,48
Hrip	1601	74,8	11:48	7:45	1,52

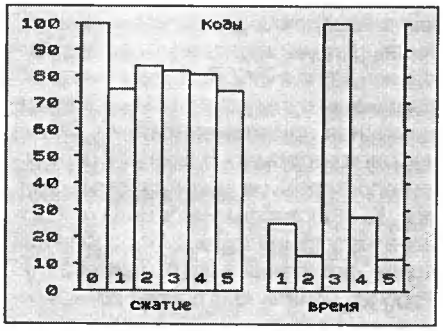
стирования на одном дисководе, содержалась более значительная доля уже скомпрессированных программ, что повлияло в худшую сторону на эффективность работы всех рассматриваемых программ, по сравнению с полномасштабным тестированием. Относительные же оценки эффективности — без изменений.

И снова квазидисковод (**табл.12**).

И вот, наконец, общие итоги и выводы.

Во всех рассмотренных случаях наилучшие результаты по степени сжатия демонстрировали архиваторы ZXZIP "normal" и Hrip — их показатели шли, что называется, "ноздря в ноздрю", очень незначительно отличаясь друг от друга. Но Hrip при этом постоянно выделялся экономией времени, затрачивая его, как правило, примерно в два раза меньше, чем ZXZIP. По большому счету, архиватор ZXZIP "fast", не блеснув эффективностью упаковки, при этом очень даже порадовал скоростью работы, в полной мере оправдав свое название. В целом ряде случаев этот фактор может иметь решающее значение.

Рис. 10



Сравнительные оценки архивирующих программ будут неполными, если упустить из виду еще один параметр — оперативность при использовании уже готовых архивных файлов. Ведь, по большому счету, архивы именно для этого и создаются. Какие-нибудь масштабные исследования на эту тему, подобные вышеописанным, видимо, затевать не обязательно. Но некоторые сведения, могущие дополнить уже почти завершенный обзор, все-таки привести стоит.

Для проведения короткого экспресс-тестирования были взяты для распаковки архивы, полученные в ходе одного из последних исследований (табл.11). Сам тест состоял из двух частей. В обоих случаях разархивация велась на тот же диск, с которого считывался архив. Но в первой части теста дисковод был реальный, что должно было приблизить результаты исследования к обычным для большинства синклеристов условиям работы, а во второй — виртуальный, электронный дисковод.

Для "разворачивания" архивов в стандарте ZXZIP используется отдельная

Табл. 11. Машинные коды (1062 сектора в 18 файлах, B → B)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	868	81,7	11:25	7:10	1,59
(fast)	959	90,3	6:44	4:27	1,51
(slow)	934	87,9	53:46	32:17	1,66
LZ-Compressor	927	87,3	14:33	10:06	1,44
Hrip	855	80,5	5:33	3:48	1,46

Табл. 12. Коды (1062 сектора в 18 файлах, C → C)

Архиватор	Размер		Время "норма", мин	Время "турбо", мин	Коэффициент турбирования
	Сжатый, секторов	% от исходного			
ZXZIP (norm)	-//-	-//-	10:10	5:56	1,71
(fast)	-//-	-//-	5:29	3:13	1,70
(slow)	-//-	-//-	52:28	31:01	1,69
LZ-Compressor	-//-	-//-	9:57	5:56	1,68
Hrip	-	-	-	-	-

утилита — ZXUNZIP, обычно имеющаяся в комплекте с собственно архиватором. Дизайн и управление в ней такие же, как и в ZXZIP. В ходе процесса распаковки отслеживается заранее подсчитанная при архивировании циклическая контрольная сумма для проверки целостности распаковываемых файлов. К недостаткам утилиты можно отнести отсутствие каких-либо сообщений после окончания разархивирования. Рабочая панель программы перерисовывается только после нажатия любой клавиши, и о том, в какой именно момент нужно нажать эту самую "any key", можно догадаться только по окончательному погасанию сигнального светодиода на дисководе. Гораздо меньше шансов вовремя "поймать" момент истины, если вся работа происходит на электронном квазидисководе.

Архивы в формате LZ-Compressor также раскрываются с помощью отдельной программы под названием LZ-Depacker. Управление в этой утилите несколько иное, чем в самом архиваторе, что нарушает принцип единообразия и немного затрудняет процесс разархивирования. Во время работы LZ-Depacker также проверяет контрольную сумму (CRC) распаковываемых файлов.

Гораздо удобнее в этом отношении работать с Hrip-архивами, поскольку процедуры распаковки уже встроены непосредственно в сам архиватор и вызываются при выборе соответствующей опции в главном меню программы (см. также рис.5).

Таким образом (**табл.13**), можно видеть, что оперативность при распаковке уже готовых архивов практически у всех

Табл. 13. Распаковка архивов

Архиватор	B → B		C→ C	
	"норма"	"турбо"	"норма"	"турбо"
ZXZIP (norm)	2:36	2:14	0:57	0:35
(fast)	2:52	2:24	1:08	0:42
(slow)	28:29	17:34	26:44	15:50
LZ-Compressor	3:03	2:28	1:27	0:54
Hrip	2:51	2:13	-	-

рассмотренных программ вполне приемлемая. У всех, кроме ZXZIP slow. Если еще учесть, что и сам процесс архивирования в режиме ZXZIP slow выполняется в большинстве случаев крайне медленно, то использование этого режима для практических целей не выглядит оправданным ни с какой точки зрения.

Для других режимов и архиваторов (за исключением, конечно же, Hrip), использование квазидисковода дополнительно сокращает в два-три раза время готовности при распаковке архивных файлов.

И в заключение — о системном и прикладном программном обеспечении, которое было использовано при подготовке статьи, помимо уже названого:

- операционная система iS-DOS'99;
- текстовый редактор iS-Editor;
- графический редактор iS-DOS ArtStudio;
- iS-DOS Calculator;
- iS-DOS Basic;
- iS-Basic Line Diagramm.

Остается добавить, что для системы iS-DOS, в которой непрерывное операционное пространство достигает 16 Мб, необходимость в мощных средствах архивирования файлов является еще более насущной, чем в TR-DOS. Но, хотя утилиты-компрессоры для "сжимания" разных типов данных в iS-DOS представлены довольно широко, какие-либо универсальные программные продукты, подобные вышеописанным TR-DOS'ным архиваторам, отсутствуют как класс.

Несколько снижает остроту проблемы архивного хранения файлов распаковщик ISUNZJR Mini Extract Utility ver.1.2, (c)1997 (автор — все тот же М.Кондратьев, Санкт-Петербург). Эта утилита позволяет "разворачивать" архивы, упакованные по стандарту архивного формата ZIP, разработанного и выпущенного в свет фирмой PKWARE в 1989 г. [3]. Работает она довольно эффективно, средняя скорость распаковки — около 1,8 Кб "зипа" в секунду на RAM-диске в режиме "turbo". Но вот только "заготавливать" такие архивы приходится на компьютерах других платформ (в частности, IBM PC), поскольку "своего" ZIP-архиватора в iS-DOS на Sinclair, к сожалению, нет и по сей день.

Литература

1. (c) Diver/CRG, (c) Елена Гладкова/CRG. Алгоритмы сжатия информации. — Электронный журнал Deja vu #6, г.Кемерово, декабрь 1998 г.

2. (c) С.Симонович. Архивация, компрессия, сжатие. — ZX-Ревю, 1995, №1, С.16, МКП "Инфорком", Москва, 1995 г.

3. Экслер А. Б., Архиваторы (Программы для хранения и обработки информации в сжатом виде). — М.: МП "Алекс", 1992 г.

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

О программировании арифметических операций

В [1], рассматривая различные способы быстрого умножения целых чисел, я приводил две формулы, взятые из [2]:

$$xy = \frac{1}{2}((x+y)^2 - x^2 - y^2), \quad (1)$$

$$xy = \frac{1}{4}((x+y)^2 - (x-y)^2). \quad (2)$$

Далее я утверждал: "Чтобы избавиться от деления на 2 в первой формуле и от деления на 4 во второй, достаточно хранить в таблице квадратов уже поделенные на 2 или на 4 значения. Умножение будет выполняться быстрее, но станет неточным, особенно при малых значениях сомножителей". Иначе говоря, имелось в виду следующее:

$$xy \cong \left[\frac{(x+y)^2}{2}\right] - \left[\frac{x^2}{2}\right] - \left[\frac{y^2}{2}\right], \quad (3)$$

$$xy \cong \left[\frac{(x+y)^2}{4}\right] - \left[\frac{(x-y)^2}{4}\right]. \quad (4)$$

Из чего следовали мои выводы о неточности умножения по этим формулам? Так как при хранении в таблице уже поделенных на 2 (или 4) значений (т.е. целых частей этих значений) происходит потеря точности, логично предположить, что вычисления будут происходить с логрешностью. Так как абсолютнное значение погрешности ограничено, при умножении больших значений относительная погрешность будет невелика, а при уменьшении сомножителей она будет увеличиваться.

Вроде бы, все очевидно, да? Мне тоже так казалось :-). В действительности, даже очевидные на первый взгляд утверждения нуждаются в тщательной проверке.

Просматривая [3], я обнаружил формулу:

$$xy = \left[\frac{(x+y)^2}{4}\right] - \left[\frac{(x-y)^2}{4}\right], \quad (5)$$

эквивалентную формуле (4), только равенство оказалось не приближенным (как я считал), а точным. Почему?

В [3] утверждается, что дробные части выражений $(x+y)^2/4$ и $(x-y)^2/4$ равны. Действительно, произведение целых чисел x и y — также целое число, а разность двух чисел $(x+y)^2/4$ и $(x-y)^2/4$ может быть целой только тогда, когда их

дробные части равны. Таким образом, отбрасывание дробных частей в данном случае не влияет на получаемый результат, и умножение по формуле (5) действительно будет точным.

Кстати, в [3] также упоминается, что эта формула вместе с таблицей значений функции $[z^2/4]$ при натуральных значениях z использовалась для облегчения умножения многозначных чисел еще до изобретения таблиц логарифмов.

Может быть, и формула (3) обеспечивает точное умножение? Проверим это. Так как x и y могут быть четными или нечетными, возможны четыре различных случая.

1. x — четное, y — четное. Тогда $(x+y)^2$, x^2, y^2 будут четными, и дробные части всех трех выражений в скобках будут равны 0.

2. x — четное, y — нечетное. Тогда x+y — нечетное; $(x+y)^2$ — также нечетное (дробная часть $(x+y)^2/2$ равна 1/2); x^2 — четное (дробная часть $x^2/2$ равна 0); y^2 — нечетное (дробная часть $y^2/2$ равна 1/2). В результате, после вычитания дробные части взаимно уничтожатся, а значит, от их отбрасывания результат не изменится.

3. x — нечетное, y — четное, т.е. ситуация аналогична предыдущей, т.к. формула симметрична относительно x и y.

4. x — нечетное, y — нечетное. Тогда x+y — четное; $(x+y)^2$ — также четное (дробная часть $(x+y)^2/2$ равна 0); x^2 и y^2 — нечетны (дробные части $x^2/2$ и $y^2/2$ равны 1/2). Следовательно, при отбрасывании дробных частей результат окажется на 1 больше истинного.

Как видим, умножение по формуле (3) действительно является неточным — в том случае, когда оба сомножителя нечетные.

С теоретическими вопросами разобрались :-). Перейдем теперь к практическому использованию формулы (5). В [1] я приводил основанную на формуле (2) процедуру умножения двух целых 8-битных чисел с ограничением "сумма сомножителей не превосходит 255". Когда в таблице квадратов хранятся значения, уже поделенные на 4, из этой процедуры надо убрать команды деления результата на 4. Соответственно, получим следующее:

```
; Вход : D - первый сомножитель,
;        E - второй сомножитель.
; Выход: DE - произведение.
; Длина: 22 байта (сама процедура) + 512 байтов
;        (таблица квадратов).
; Время: 101 или 104 такта - в зависимости
;        от того, первый сомножитель
;        больше второго или наоборот
;        (если заранее известно, что один
;        всегда больше другого, процедуру
;        можно упростить).

        LD      H,TABL_SQR/256
        LD      B,H

        LD      A,D
        ADD     A,E
        LD      C,A     ;C=x+y

        LD      A,D
        SUB     E
        JR      NC,M1
        NEG
M1      LD      L,A     ;L=|x-y|

        LD      A,(BC)
        SUB     (HL)
        LD      E,A
        INC     H
        INC     B
        LD      A,(BC)
        SBC     A,(HL)
        LD      D,A     ;DE=x*y

        RET
```

Теперь умножение выполняется на 24 такта (~20%) быстрее по сравнению с исходным вариантом процедуры.

Ниже приведен текст процедуры построения таблицы квадратов, деленных на 4. При ее написании я взял за основу приведенную в [1] процедуру построения таблицы квадратов (которая, в свою очередь, является результатом оптимизации процедуры, приведенной в [4]).

```
        XOR     A
        LD      H,A
        LD      L,A
        LD      E,A

M2      LD      D,TABL_SQR/256+1
        EX      DE,HL
        LD      (HL),D
        LD      A,E
        SRL     (HL)
        RRA
        SRL     (HL)
        RRA
        DEC     H
        LD      (HL),A
        EX      DE,HL
        LD      D,0
        ADD     HL,DE
        INC     E
        ADD     HL,DE   ;На флаг Z не влияет.
        JR      NZ,M2

        RET
```

Литература

1. И.Рощин. Еще о программировании арифметических операций. — Радиолюбитель. Ваш компьютер, 2000, №12; 2001, №№1-4.

2. М.А.Карцев. Арифметика цифровых машин. — М.: Наука, 1969.

3. А.П.Доморяд. Математические игры и развлечения. — М.: Государственное издательство физико-математической литературы, 1961.

4. Card!nal/PGC/BD. Кодить хочу!. — Электронный журнал "Deja Vu #5".

Какие бывают Z80

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

Немного истории

Микропроцессор Z80 был разработан фирмой Zilog в 1980 г. (о чем свидетельствует его название), тогда же начался его массовый выпуск. В этом же году фирма "Sinclair" британца Клайва Синклера выпустила на рынок первый общедоступный компьютер ZX80. Равная по своим возможностям многим "продвинутым" компьютерам того времени, эта машина ввела в мир вычислительной техники многих и многих людей.

На смену ZX80 в 1981 г. пришел компьютер ZX81, обладавший улучшенными характеристиками (например, блочной графикой) и строившийся на базе процессора ZX81, не получившего такого широкого распространения как Z80.

Но огромную популярность и распространение МП Z80 получил с выпуском в 1982 г. той же фирмой микрокомпьютера "ZX-Spectrum". В дальнейшем выпуском МП Z80 или его аналогов занялись десятки фирм, в том числе и отечественных. Наиболее известными аналогами являются МП U880 (Германия) и КР1858.

Модели Z80

Большинство разновидностей микропроцессоров Z80 имеют тактовую частоту 4 МГц. Главное их различие состоит в количестве "ножек", а также команд микропроцессора. Общепринятый и самый распространенный конструктивный стандарт — 40 выводов — был заложен фирмой-разработчиком Zilog и перенят производителями процессоров серии U880. Однако не все Z80 (U880) выполнены по такому конструктиву. Например, есть 28-"ножечный" U857 (Z80CTC).

В серию МП U880 (Z80) входят и такие как U855 (Z80PIO) и U856 (Z80SIO). Но все-таки наиболее "продвинутым" следует считать Z80A, обладающий конструктивной совместимостью и набором команд большинства Z80 (U880), а также самый производительный МП данного класса — Z80B с тактовой частотой 7 МГц.

Перспективы

На мой взгляд, компьютеры, построенные на базе МП Z80 и ему аналогичных, являются прекрасным инструментом для непрофессионалов и новичков, желающих познакомиться с вычислительной техникой, а также для любителей-программистов. Если "дать ума", т.е. немного усовершенствовать такие компьютеры, они наверняка приобретут огромную популярность у средних слоев населения, ведь не все мы вынуждены быть богатыми, а как известно, ZX стоят очень дешево по сравнению с другими ПК.

Многие говорят, что все МП Z80 являются 8-разрядными. Но как известно, истины не скроешь, поэтому хочется сказать, что в конце 80-х годов были разработаны 16-разрядный МП U8800 (Z800), а также 32-разрядный МП U88000 (Z8000), к сожалению, не получившие такого широкого распространения как 8-разрядный Z80.

А.ВЕНДИЛОВСКИЙ,
г.Минск.

Unreal 2

2335 г. Военная база землян в системе Ориона.

Маленький "челнок" легко опустился на посадочную площадку перед главным корпусом штаба. Впереди виднелись сторожевые башни космодрома и массивное здание Центра координации Армии в этом регионе космоса. Через несколько минут на бетон площадки молодцевато спрыгнул лейтенант, по виду очень молоденький, явно получивший свои нашивки весьма недавно, и очень ими гордящийся. Он практически бегом устремился к эскалатору, на ходу показывая охране свой значок и удостоверение.

"Еще один "желторотик" на нашу голову, — грустно вздохнул генерал, наблюдая эту картину — ну, ничего не поделаешь..."

Едва двери лифта открылись, как лейтенант сразу попытался доложить генералу, но тот нетерпеливо прервал его взмахом руки. На мгновение в воздухе повисло молчание

— Слушайте меня очень внимательно, лейтенант, — генерал посмотрел в свои бумаги и еще раз вздохнул. — Вам вверяется контроль участка космоса, код 12-М-45. Вы должны прибыть туда немедленно. На вас ложится задача по патрулированию космоса, а также по выявлению действий предположительно враждебных нам цивилизаций в этом секторе. Район большой, 12 солнечных систем, это не шутка!

На лице лейтенанта отразилась вся гамма чувств человека, который пришел на службу ради сражений и героических подвигов, а оказался на занюханной штабной работе в роли участкового инспектора.

— И еще, экипаж уже ждет вас на орбите, с личными делами ознакомьтесь по дороге. Вам достались лучшие люди нашей армии, и не только земной! Поэтому...

— Простите, сэр! — лейтенант нашел в себе силы побороть возмущение и некоторую робость по отношению к вышестоящему начальству, — но я надеялся, что мне поручат настоящую работу...

— А эту вы не считаете настоящей!? — генерал гневно взглянул на новичка, и, тихо свирепея, продолжил, — от этой работы зависит целостность нашей земной коалиции, прочность наших границ! У нас много врагов, и они не дремлют! — он бросил на стол папку с мини-диском, — вот очередной инцидент, пришло сообщение с планеты... э..., координаты тут указаны, просят помощи, говорят, что у них проблемы. Вот и разберитесь с этим эпизодом.

— Но...

— Никаких но! Кругом!

Когда двери лифта закрылись за лейтенантом, он тихо буркнул про себя в адрес генерала, — Старый козел! — И вряд ли он услышал, как генерал, провожая взглядом опускающийся лифт, тихо пробурчал: "Пацан...".

Взлет был полностью автоматизирован, так что у лейтенанта было много времени, чтобы ознакомиться с личными делами своего будущего экипажа. Чтобы не умереть от скуки, он зачитывал их вслух.

Инопланетянин Ne'Ban — ваш пилот и навигатор, инженер Isaak отвечает за двигатель и снаряжение, Aida — офицер разведки.

Aida прошла серьезную подготовку (она была руководителем команды Nightwing Corps).

Isaak был первоклассным инженером на крупном корабле, пока не заработал посттравматический стресс в одной серьезной битве. Прошел лечение и перешел в колониальную службу в надежде на меньшие стрессы в работе.

Ne'Ban попал на Atlantis в результате выполнения программы по обмену военными офицерами между Землей и недавно разведанной внеземной расой. Программа обмена была выдумана кем-то из дипломатов, но среди военных была встречена без энтузиазма — никто не горел желанием заполучить инопланетянина в качестве офицера.

В этот момент челнок хорошенько тряхнуло, и стыковка с новым домом состоялась. Первое, во что уперся взгляд лейтенанта, был шикарный бюст офицера разведки. Вот чего он не мог себе представить, так это такую красавицу на этом космическом корыте.

— Добро пожаловать на борт "Атлантиса", командир. Прошу вас пройти за мной в рубку.

В рубке столпотворения не было, как мог бы предположить лейтенант, экипаж весьма индифферентно отнесся к смене командира.

— Пока вы добирались, ситуация несколько изменилась — с планеты пришел еще один сигнал с просьбой о помощи из этой колонии, к сожалению, очень обрывистый и слабый. Через пару минут мы выйдем из гиперпространства, вы спуститесь на поверхность и произведете разведку. Попытайтесь спасти всех, кого возможно. Командование ждет от вас информации, стоит ли выдвигать сюда основные силы, или этот инцидент не стоит выеденного яйца. Прежде чем вернуться в шаттл, советую спуститься к нашему оружейнику и на всякий случай набрать как можно больше боеприпасов и вооружения...

... Корабль вынырнул из пустоты и словно завис над поверхностью голубой планеты. Через пару мгновений маленький силуэт оторвался от него и заскользил вниз, превращаясь в гигантский огненный шар. Ворвавшись в облака, огненный шар исчез, и к земле стремительно летел маленький шаттл. Лейтенант очень внимательно рассматривал окрестности земной базы. Людей не наблюдалось, что было само по себе странно, внезапно в его сторону ударило несколько энергетических лучей.

— Что за дьявольщина! — аппарат едва не рухнул на посадочную площадку, но в последний момент пилот справился с управлением, и агрегат грузно приземлился. Не задерживаясь ни на секунду, лейтенант выскочил из него, готовый в любой момент открыть огонь по любому, что может показаться угрожающим. Но странные твари, которые попытались подбить челнок, уже сбежали. И только пронзительное верещание доносилось издалека.

— Бог мой, ... "Атлантис", вы тоже это видите?... — двор был просто залит кровью и завален останками людей, по окраваленным ошметкам с трудом можно было понять, что совсем недавно это были техники.

— Аида! Нам тут одним не справиться! Вызывай подкрепление! Вызывай скорее! — в этот момент тяжелый энергетический снаряд просвистел над головой лейтенанта, превращая в крошево бетонную стену.

— У нас проблема, командир... на планете находится что-то, что глушит нашу связь!

В этот момент настала тишина, лейтенант осторожно заглянул в огромный зал. На первый взгляд, там никого не было, только из-за дальней колонны была видна фигура человека,

совершавшая странные конвульсивные движения. Присмотревшись, он увидел, что грудь бедняги пробита подобием вил, на которых он и висел.

— Думаете, я клюну на такой дешевый трюк? — лейтенант ухмыльнулся, и короткими перебежками от колонны к колонне стал подбираться к твари, стараясь остаться незамеченным. Когда зеленая рожа этой образины аккурат попала в прицел винтовки, он ухмыльнулся и нажал на спусковой крючок...

В этот миг комната превратилась в ад, никто не мог ожидать, что эти твари могут становиться невидимыми. Он кричал, прыгал, выпуская обойму за обоймой, энергетическое оружие пришельцев давало страшный рикошет, и уже было трудно понять, откуда действительно ведется огонь. Их прыжки в невидимость создавали впечатление, что их тут тысячи. В тот момент, когда бой практически подошел к концу, и последняя тварь бросилась на него, винтовка тихо щелкнула, и лейтенант с ужасом понял, что только что расстрелял последнюю обойму. Времени на то, чтобы достать другое оружие, просто не оставалось, он отбросил ненужную сейчас винтовку и рефлекторно направил на противника валявшиеся под его ногами "вилы". Рука, скользнув по стальной поверхности, зацепила небольшую скобу, и огромный шаровой заряд энергии ударил прямо по оскаленной морде твари, превращая ее в кровавое месиво.

— Приятного аппетита, — буркнул лейтенант, тяжело поднимаясь на ноги. В этот момент в коридоре раздались гулкие шаги. Не желая испытывать судьбу, он быстро прыгнул в трубу вентиляции и затаился. Огромная, в полтора раза больше человеческого роста фигура выросла в дверях. На ее руках, помимо двух ручных гранатометов, крепились страшные лезвия для ближнего боя. Несмотря на свои габариты, тварь передвигалась с грацией и скоростью, которым позавидовали бы кошки. Монстр просто излучал смертельную угрозу, оглядев зал, он быстро побежал по коридору вглубь.

— "Атлантис", ..., это что за ублюдок? — лейтенант был просто взбешен, по-прежнему ощущая страх.

— Командир, кажется я знаю, кто это такой. — Лейтенант готов был поклясться, что Аида тоже испугана, и очень испугана. — Я была допущена к документам с высшим грифом секретности. В одном из них, почти полуторастолетней давности, говорилось об инциденте на планете На-Пали. Боевой крейсер неизвестной расы, настроенной очень враждебно, высадился там и захватил жителей планеты в рабство. На свою беду, на этой планете разбился тюремный корабль землян, в живых смог остаться только один заключенный. Никто не знает его имени, он проходил под номером 6. Деваться ему было некуда, и он в одиночку перебил штурмовой корпус этих тварей. А потом попытался бежать на спасательной шлюпке пришельцев. Мы знаем об этом, потому что в том секторе космоса проводили испытания своего нового боевого корабля, который был сбит пришельцами. Когда наша доблестная армия спустилась на планету, ей просто надрали задницу. На беду или на удачу, в спасательной шлюпке Шестого не оказалось топлива, и наши доблестные десантники подобрали ее на орбите. Командование предложило ему сделку — он находит им корабль, а они ему дают полное помилование. Шестой справился с заданием, но наши командиры совсем не собирались выполнять свою часть сделки. Когда обман раскрылся, Шестой, опять-таки в одиночку, сделал то же, что и пришельцы — надрал задницу нашей героической армии и скрылся в неизвестном направлении. У нас остались только образцы некоторого оружия и знание, что этих тварей называют Скаарджи. Откуда они пришли, нам тоже установить не удалось.

— Этот ..., что заходил сюда, был скаардж?

— Боюсь, что да... И еще, мы не можем передать сигнал на базу и не можем прыгнуть в гиперпространство, что-то или кто-то блокирует нас. На маневровых двигателях мы не сможем покинуть зону воздействия, наш единственный шанс — найти местный центр связи и послать сигнал о помощи!

Лейтенант поднялся, в его глазах заблестел дьявольский огонек.

— Еще не вечер! Если уж какой-то заключенный смог это сделать, то мы уж точно сделаем!

И он стал осторожно пробираться по коридору...

Как всегда, выхода такой игры все ждали, и, как всегда, он для всех стал полной неожиданностью. Нам снова придется окунуться в нереальный мир, но теперь вместо робы заключенного примерим костюм космического шерифа.

Как всегда, начнем наши "разборки" с игрой с графики. Все помнят, на какой уровень была поднята графика в первой части, ветераны до сих пор взахлеб рассказывают об этом. Потомок тоже не подвел. Красота и буйство красок насытят экран вашего монитора, движения персонажей, совершенно живую анимацию: движения головы, глаз, мимику и движения губ. Монстры и другие персонажи будут обладать аналогичным типом персонификации. Каждый ведет себя как личность. Как-нибудь обратите внимание на поведение скаарджа, когда ему удается победить игрока. А так как разработкой игры занималась "Legend Ent.", которая хорошо известна по игре "Колесо Времени", то дизайн уровней поражает своей фантазией и... дотошностью в мелочах, которую вы редко встретите в других играх. Посмотрите на скриншоты игры. Вы сможете посетить руины города инопланетян, пройти по подземным катакомбам, обследовать поверхность вулканического происхождения. Каждый мир будет по-своему интересен, новые технологии придадут максимальную реалистичность ландшафтам, струящейся воде, огню, дыму, бьющемуся стеклу и погодным эффектам. Количество полигонов на объектах вызывает немое благоговение. И тут же нас поджидает разочарование — всей этой прелестью смогут насладиться лишь владельцы третьих и четвертых GeForce. GeForce 2MX, получившая у нас сейчас наибольшее распространение, считается... минимальной карточкой для запуска игры. У меня была возможность сравнить картинку как на второй, так и на третьей GeForce, и с того момента мысль об апгрейде стала весьма навязчивой :).

Что касается прохождений миссий, то должен отметить, как до этого делали мои коллеги, что разработчики пошли путем "half-life", стараясь, чтобы сюжет был тесно переплетен с геймплеем, и игроку не приходилось бездумно выносить толпы монстров, а можно было связать все это единой идеей. Вам придется вести переговоры с собственным экипажем и даже с начальством, чтобы получить какие-нибудь бонусы или перевести сюжет в новое русло. Не волнуйтесь, это не станет надоедливо-скучной обязанностью — диалоги коротки, и больше помогают расслабиться от адреналинового драйва и глубже проникнуться атмосферой игры.

Все действо будет происходить в пределах семи различных планет, где придется пройти 13 миссий, различных по поставленным задачам (да-да, придется даже покомандовать целым отрядом!), что в сумме составит порядка тридцати уровней.

Как и любого любителя хорошей стрелялки, меня всегда интересовало, сколько мне встретится видов враждебных личностей, и чем я их буду истреблять. Создатели населили этот мир 24 представителями враждебной цивилизации и еще тремя сотнями весьма злобных представителей "местной" флоры и фауны. А для истребления изначально выдали стандартный земной штурмовой набор — пистолет, дробовик, автомат, рокет, снайперскую винтовку, огнемет, гранатомет, снаряжаемый токсичными, усыпляющими, фрагментирующимися и разрывающимися гранатами. Сверх того, более экзотические типы оружия будут доступны в виде изделий инопланетян. Эти сумасшедшие штуки будут включать в себя Takkra, производящую на свет рой летающих объектов, способных атаковать противника либо защищать вас, летая вокруг. Будет еще Leech Gun (пиявочная пушка), выстреливающая пиявками, которые будут накрывать противника, валить на землю и высасывать его энергию или жизнь. Скучать не придется, а любой поединок со скаарджем превратится в смертоносную дуэль, в которой победа вам не гарантирована, даже если у вам самое мощное оружие.

Осталось сказать только о мультиплеере. В отличие от UT, он будет... стратегическим!

Вот как об этом пишут сами создатели: "В новой, расширенной версии игры, команды игроков различных классов будут сражаться за контроль над поверхностью планеты и артефакты пришельцев, с использованием нанотехнологических репликаторов для создания более мощного оружия. Каждый из 32 максимальных игроков, будучи морским пехотинцем, наемником или скаарджем, сможет выбрать для себя один из трех различных классов. Доступны будут: солдаты, которые смогут использовать все виды тяжелого оружия в игре; "привидение" — легкий класс, который сможет очень быстро двигаться и использовать все виды более экзотического оружия; техник-солдат, который сможет контролировать установку, производящую все оружие, амуницию, генераторы силовых полей, ракетницы, станции лечения и бронезащиты, сенсоры, всстанавливающих роботов и любые другие предметы, которые вы сможете создавать для модификации своей базы. Если вы захотите сыграть один на один с другим игроком, вы сможете выбрать тип "Commando Superclass", который способен воплотить в себе все три роли. Если игроков больше одного, вы можете выбрать для себя любую группу, но вам необходимо будет иметь одного техника. Вы сможете комбинировать и дополнять игроков-людей искусственными персонажами — играть одному с одними ботами или использовать ботов в командной игре."

Вывод — полномасштабный хит, который на годы установит новую планку уровня качества экшена.

Диск с игрой предоставлен интернет-магазином "MediaCraft"

К.КЛИЩЕНКО,
г.Минск.

Шароварры

На безрыбье и рак – рыба.
Народная мудрость

Голь на аыдумки хитра.
Не менее народная мудрость,
придуманная самой голью

Застой... Настоящий застой... Полный застой... Программисты совсем обленились, а ведь солнце еще высоко, и им бы работать и работать. Приходится обращаться за помощью к нераскрученным, хотя от этого отнюдь не менее замечательным народным умельцам и к их shareware-продуктам. Выдаю справку для неосведомленных: shareware — это программные продукты, распространяемые по Интернету. Вы бесплатно получаете часть программы или даже всю ее (на ограниченное время или с урезанными функциями) и пользуетесь этим "обрубком" в свое удовольствие. Но когда вы в порыве экстаза захотите использовать ее "по полной", то вступает в силу золотое правило — за все нужно платить. И только после пересылки произвольной суммы на счет разработчиков вы получаете "остаток" (кусок кода или пароль для активации программы). И вот перед пиратствующим всю жизнь племенем геймеров встает просто неразрешимый вопрос — послать Jagged Gaming 15 USD или найти пароль в Интернете. Ни за что не признаюсь, как именно я поступил. С одной стороны, не хочется получить нагоняй со стороны антипиратских законов, а с другой — обвинения в "закушенности" лица мне также не нужны. После такой справки стоит сказать, что же собственно мне попало в руки. Оба (а мне довелось опробовать две игры) продукта выпущены одной и той же конторой — Spider Web. Честь ей и хвала (кстати, не только за то, что она помогла мне скоротать сессию). Ведь игры-то получились на славу, и я сильно жалею, что эти проекты не были выполнены на более высоком технологическом уровне. Ведь тогда они вполне могли бы попортить кровь грандам (грандам жанра РПГ).

Две игры — такие разные и, естественно (ведь "руки", их творившие — одни и те же) до боли похожие во всех частях тела. И обе примечательны. Первая —Avernum 3 — является одновременно и третьей (это ясно из номера), и шестой частью серии (это следует из истории). Существовавшая ранее серия Exile, на первый взгляд, отличается от Avernum лишь переходом из чистой и незамутненной плоскости в псевдотрехмерную изометрию (типа сверху-сбоку). Но изменение некоторых заклинаний и переработка системы персонажа существенно меняет характер и механику игры. Так что считать, что Avernum — это лишь пересмотренный под другим углом Exile, можно с большим трудом. Если же говорить о собственно механике, то здесь наши герои не открывают ничего особо нового. Ранние Ultima, Gold Box AD&D, в чем-то Dark Sun (т.е. игры не из последних) явно подействовали на умы создателей. Ваши герои (четыре персонажа) бороздят безграничные просторы верхнего мира и не менее бесконечные подземелья и башни. Все эти прогулки нужны для выполнения великой миссии. Сотни лет ваш народ томился в титанических подземных пещерах Avernum'a. И если поначалу эти пещеры заполнялись преступниками, то теперь Империя проводит "охоту на ведьм" и засылает туда "неправильно" мыслящих и просто попавших под горячую руку. Именно вам и предстоит обеспечить возможность выхода своего народа на поверхность. Пошаговость всего на свете играет на ваших ностальгических чувствах. Цельной группой вы бродите по поверхности,

иногда встречая дружелюбно (чаще недружелюбно) настроенных жителей. Условия несколько изменяются, когда вы заходите в город или в подземелье. Теперь ваши персонажи становятся в колонну и ходят друг за другом по пятам (видимо, чтобы оставлять поменьше следов). Только в этом режиме можно осуществлять мирные действия (разговоры и манипуляции с объектами). Но когда выбор уже невелик (т.е. меч приставлен к горлу), необходимо переходить в боевой режим. Свобода перемещений — полная, а вот действий — поменьше. Ударить кого-то, колдануть или выпить, конечно, можно, но не более того. Кстати, именно боевой режим вызвал у меня больше всего нареканий. Хотя он и проверен годами, но очарования некоторых других ему достичь не удалось. Неоправданная сложность в одних местах и простота в других оставляют неприятный осадок. К тому же, излишняя частота сражений способна многих утомить (а бой все-таки не Startrail, Fallout или Wizardry). К тому же, если воины исполнены довольно приемлемо, то маги и жрецы (и им подобные) немного не проработаны. Нет ни одного сильного атакующего заклятия, рассчитанного на поражение одного субъекта. Также налицо явный дисбаланс в сторону заклинаний вызова. А потому бой на поздней стадии игры идет 40 на 40 (хотя вначале он был 4 на 4). Особенно примечательны в этом плане некие паучки — они не только дерутся, но и умеют вызывать существ той же силы, что и вы, т.е. и себе подобных (лично мне доводилось наблюдать подряд три клонирования этих арахнид). Создателям игры "повезло" сделать этот вызов неуправляемым (хоть это иногда и раздражает, но зато и не предоставляет мир паукам).

Прошедшись таким образом по бою, все же стоит сказать, что игр без недостатков не бывает (к сожалению), а здесь, к счастью, недостатки не ведут к сильному дисбалансу. Мир радует гораздо больше, хочется лишь напустить на него толпу безумных железных дровосеков. Деревья особенно раздражают не по причине своей уродливости, а потому что в ваших долгих поисках и хождениях вам столько раз предстоит искать путь в непроходимой чаще, что кроме гнева этот поиск ничего не вызывает. Локаций много, и если бы не скудность руки штатного художника, то они были бы красивы (а так выделяются оригинальностью лишь главные "нехорошие"). Города интересны не только количеством зданий и их расположением, но и населением. Жаль, что убрали возможность задавать вопросы на произвольные темы (как в Exile). А то у меня с этой возможностью связаны интересные воспоминания. Довелось мне как-то беседовать с американским проповедником. Так тот мне поведал, что играет лишь в одну компьютерную игру — Exile 3. А играет он в нее только потому, что ему нравиться ходить по миру и беседовать с людьми о религии (хотя он мог и перепутать с Ultima Online). Так вот, несмотря на то что побеседовать с гражданами о Боге не удастся, с ними можно поговорить о многом другом. Об их проблемах (квестах), слухах (интересных локациях), и на некоторые общие (для этого мира) темы. Персонажи, может, не столь запоминающиеся (кроме немногих), но зато создают очень интересную и объемную (в отличие от графики) картину. Да, между прочим, Анаксимандр теперь для меня не философ, а имя моего главного координатора в этом мире. Спасибо авторам за в меру интересное повествование, в разных местах игры мера разная. Но вот желание сделать игру интереснее и сложнее заставило авторов перегнуть палку в процессе воплощения планов. Им хотелось не только сделать бои сложнее, но и заставить ваши мозги работать на более высоком уровне. Не всегда это получается (прошедший башню големов, да поймет меня). И вот с этими недостатками (не слишком отвращающими от игры) действо все равно захватывает и вызывает острейшую потребность забросить все дела (в моем случае квантовую механику и лазерную физику) и пройтись незримой пятой по ее полям.

Второй продукт еще более примечателен. И это не только мое мнение. Обозревателей одного из ведущих сайтов, посвященных РПГ, попросили назвать по три лучшие игры года в этом жанре. Только одну игру назвали все специалисты — богоподобный Morrowind. Но только один из них упомянул ее на первом месте, остальные поставили на второе, сразу вслед за игрой, которую я и хочу представить. Это — Geneforge. Непонятно, зачем тратить миллионы долларов на третьесортный клон Diablo, ведь тихо работая маленькой командой, можно создать такой же маленький (40,7 Мб), но, тем не менее, шедевр. Два слова о том, что объединяет эти игры кроме разработчика. Текстуры словно бы перенесены из одной игры в другую. Только в Geneforge они крупнее. А так как и в Avernum они не особенно оригинальны, то дежа вю и легкое подташнивание сопровождают вас на протяжении всего процесса прохождения.

Теперь о не слишком важных отличиях. Все, кроме боя, идет в реальном времени, что иногда доставляет мелкие неудобства, но зато смотрится и играется гораздо приятнее. Нет огромной внешней карты, и перемещения между локациями происходят скучновато (зато без головной боли). Сюжет, может, и не сильно закручен и лихо завинчен, но не банален, не слишком линеен, и оставляет о себе приятные воспоминания. Фабула его напрямую связана с самым главным в этой игре — с Shaper'ами (это вовсе не специалисты по поддержанию тела в форме). Это самые интересные и, по-моему, единственные персонажи в игре. Так вот, вы попадаете на заброшенный остров, где эти шэйперы ранее проводили важные и опасные исследования. Теперь он занят их собственными созданиями и случайно приплывшими людьми (которые почему-то носят исключительно русские имена и фамилии). Что же, собственно, происходит на острове, вам и предстоит разобраться. Ну вот, я и подошел к этим самым Shaper'ам. Плюньте на агента (боевого мага), забудьте о страже (воине) — это игра о и для Shaper'ов. Эти персонажи — самое новое, что было в РПГ за последние годы. Теперь вы по своему желанию можете создавать своих существ на базе примерно 30-ти видов, а главное, по потребностям. Такое уже было в стратегиях, но там вы были рабом ресурсов, здесь же вы вольны на каждый случай подобрать себе команду по своим возможностям и потребностям — много маленьких ребят перебьют парочку сильных. Иногда нужны танки. Не нравятся танки, сделайте себе один, закройте проход, а задние ряды заплюйте кислотой, пусть умирают. В дороге встретились мины, положите на амбразуры несколько своих созданий. Восполнить ряды просто. И можно забыть о росте существ в уровне. Для шэйпера это не актуально. Возможности огромны. Мир сразу становится привлекательным и захватывающим.

В общем, стоит попробовать эти невинные (судя по размеру на винчестере) игрушки. И, могу вас заверить, помнить о них вы будете не меньше чем о ... (к сожалению, мне нельзя писать конкретные названия, чтобы не обвинили в антирекламе).

P.S. Jagged Gaming — скорее всего, выдуманное название.

КУПЛЮ, ПРОДАМ, ОБМЕНЯЮ

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

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

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

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

Приму в дар компьютер и комплектующие.
Тел. в г.Минске 257-91-46, Евгений.

Куплю: любые выпуски сборника "Радиодизайн", журналы "Телеспутник" за 2002-2003 гг.; панель к лампе ГУ-34Б: диод 6Д13Д; ПК, совместимый с "ZX-Spectrum" с контроллером дисковода и пакет программ к нему.
40000, Украина, г.Сумы, а/я 9, Александр.

Ученик 8-го класса с большой благодарностью **примет в дар** или **дешево купит** подержанный компьютер.
428022, г.Чебоксары,
ул.Декабристов, 20 — 1 — 100, Морозов А.

Куплю запрограммированные ПЗУ для контроллера TR-DOS (версия 5.03 или 5.04) или готовый контроллер со схемой.
385765, Р. Адыгея, Майкопский р-н,
ст.Кужорская, ул.Чапаева,1, Ветров И.П.

Ищу контроллер дисковода к ПК "Байт" (можно без ВГ-93) с дисководом 5,25" и принципиальную схему к ПК "Балтик".
223082, Минская обл, п/о Бровки, 31,
Кривощеков Б.Г.
Тел. 504-35-16, Борис.

Ученик 6-го класса **примет в дар** компьютер.
220017, Республика Беларусь, г.Минск,
ул.Матусевича, 61 — 320, Каяшев Саша
Тел. 215-36-49.

Куплю контроллер жесткого диска для ПК "Истра 4816.03" (1991 г.) и программное обеспечение для ПК "Искра-1030М".
453500, Башкортостан, г.Белорецк, а/я 21,
Сафонов С.Н.
Тел. (34792) 4-45-67.

Ищу владельцев "Вектор-06Ц".
Куплю АОН-модем, винчестер и его контроллер в рабочем состоянии.
Продам "Мощный музыкальный автомобильный сигнал своими руками" — подробное описание сборки, платы, подключение к любому типу автомобиля. Приложить конверт с обратным адресом.
353866, Краснодарский край, г.Приморско-Ахтарск, Авиагородок, 4/24, Тышечко Р.В.
Тел. (861-43) 2-76-89.

Приму в дар компьютер и комплектующие.
460502, Оренбургская обл., с. Нижняя Павловка,
п.Геологов, ул.2-я Центральная, 11 — 4,
Маркин С.В.

Приглашаются к сотрудничеству пользователи и программисты в iS-DOS с целью создания информационного сборника по обмену опытом работы в этой ОС. Подробности в газете "Абз@ц" N13 или в конверте с обратным адресом.
412302, Саратовская обл., г. Балашов,
ул. Красина, д.82., Илясов Е. В.

Продаю ПК "Электроника БК 0011" с блоком питания.
E-mail: mdemenev@yandex.ru. Михаил.

Ученик 11-го класса с благодарностью **примет в дар** компьютер.
211970, Витебская область, г.Браслав,
ул. Тургенева, 23, Олечнович А.

Студент профессионального лицея с благодарностью **примет в дар** портативный компьютер или электронную записную книжку.
694400, Сахалинская обл, п.Тымовское,
ул,Библиотечная, 6а — 53, Фисуненко М.

Продам ПК "KAY-1024" (с документацией) + 5,25" FDD "Robotron" + кучу дисков.
230001, г.Гродно, ул.Суворова, 27б – 19.
Тел. 8-029-6-20-82-86.

Ученик 11-го класса с благодарностью **примет в дар** комплектующие для сборки компьютера или сам компьютер (можно без монитора).
Тел. в Минске: 289-33-68, Павел.

Ищу любое программное обеспечение для "ZX-Spectrum 48 (128)K" на дискетах 5,25" (TR-DOS 5.03); ищу схемы подключения периферии (программатор, принтер и т.д.) с программным обеспечением; книгу Ларченко, Родионов. "ZX-Spectrum и TR-DOS для пользователей и программистов".
385765, р.Адыгея, Майкопский р-н, ст.Кужорская, ул.Чапаева, 1, Ветров И.П.

Меняю на ПК или спутниковый ресивер (с ант. Ø0,9 м) радиоэлементы: транзистры 2П306А, 2П103Г, ГТ329, КП305Б, КТ603, 2Т321, 2П350Б, 2П302Б, КТ608, КТ312, 2Т602, МП105, МП114, МП103, МП106, МП101Б, 2Т920А, П307, КТ332, КТ801, П702А, 2Т610, 2Т610Б, КТ841А, П210Ш, КТ808, КТ803А, П701А, КТ903Б, КТ809, КТ908, КТ301, ГТ311, КТ208, КТ606, КП303, 1Т320, КТ325, ГТ404, 2П305Б, КТ203Б, 2Т919Б, ГТ403, ГТ313; диоды Д9, Д504, Д311, Д310, Д18, Д223, 2Д504, Д245, Д232, Д229А, Д234Б, Д245, 2С107А, 2С113А, КС119А, КС170, КС175, 2Ц106А, КЦ402Б, 2Н102В, Д238, Д235Б, КУ101, КУ102, КУ202И, КУ201К, КУ202Л, КУ201Л; кварцы 11800 кГц, 1575 кГц, 10700 кГц, 7410 кГц, 1312,5 кГц.
Тел (902) 468-81-36 (после 18.00., дом.).
(902) 466-68-53 (до 18.00, раб).

Уважаемые читатели!

Подписаться на наши журналы (индексы: **48996** —"Радиомир", **48924** — "Радиомир. КВ и УКВ", **48925** — "Радиомир. Ваш компьютер") во II полугодии 2003 г. можно по каталогу Агентства "Роспечать" или по каталогу РО "Белпочта" "Газеты и журналы Республики Беларусь" (раздел "Издания РФ, распространяемые по прямым договорам").

Те, у кого возникли проблемы с подпиской, могут получить их из редакции. Там же можно заказать имеющиеся в наличии отдельные номера журналов за предыдущие годы.

Год	Можно заказать следующие номера журналов			Расценки на 1 экз. (с учетом пересылки)		
	Радиолюбитель	Радиолюбитель. Ваш компьютер	Радиолюбитель. КВ и УКВ	По России и СНГ (рос. руб.)	По Беларуси (бел. руб.)	По Украине (гривны)
1998	3, 4, 5, 8	1 — 5, 8, 9, 11, 12	1, 6, 7	20	700	5
1999	3, 9, 10, 11	1 — 6, 10, 11, 12	3, 5, 6	25	800	5
2000	2 — 6, 8 — 12	2 — 8, 10 — 12	4, 6, 7, 9, 11, 12	25	900	6
2001	2 — 6	1 — 6	2 — 6	30	1000	6
	Радиомир	Радиомир. Ваш компьютер	Радиомир. КВ и УКВ	По России и СНГ (рос. руб)	По Беларуси (бел. руб.)	По Украине (гривны)
2001	7 — 12	7 — 12	7 — 12	35	1300	7
2002	1 — 12	1 — 12	1 — 10, 12	40	1400	8
2003	1 — 12	1 — 12	1, 2, 4 — 12	45	1700	9

Наши платежные реквизиты:
- для жителей России и стран СНГ (кроме Беларуси) —
получатель: ООО "Радиомир Пресс", ИНН 7729404741,
р/с 40702810900010000084 в ООО КБ "Агропромкредит", ф-л Центральный, г.Москва,
корр. счет 30101810500000000109, БИК 044525109
Адрес банка: 113054, г.Москва, ул.Зацепа, 43Б;
- для жителей Беларуси —
получатель: УП "РЛД", УНН 190218688
р/с 3012000004882 в ф-ле №524 АСБ "Беларусбанк" в г.Минске код 121.
Адрес банка: 220028, г.Минск, ул.Физкультурная, 31.

На бланке почтового перевода необходимо очень четко написать свой почтовый индекс, полный адрес, фамилию, имя и отчество полностью. В графе "Для письма" необходимо точно перечислить, какие конкретно номера какого из журналов Вы заказываете.
При оплате платежным поручением эти сведения необходимо указать а графе "назначение платежа".

Вся информация по E-mail: rm-sales@radio-mir.com
или по тел. в г.Минске (017) 221-01-10, (017) 505-13-65.

Приобретение отдельных номеров журналов

В РОССИИ:

В **ЗАО "Интерпресс — Экспрвсс"**: тел. в Москве (095) 208-85-50, 208-67-55, e-mail: inter@aif.ru

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

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

В **ООО "ТРЭНТЭКС"**: 111024, г.Москва, 4-я Кабельная, 2А (ст. метро "Авиамоторная"), тел. (095) 258-91-94, 258-91-95. E-mail: abook@mail.ru, abook@inbox.ru
На радиорынках: "Царицыно" (место 13/А), "Митино" (ряд 1, контейнер 17 и место Т-8); в ТК "Савеловский" (ст. метро "Савеловская", места А4, А5); на книжной ярмарке на "Тульской" (ст. метро "Тульская", место 412-38).

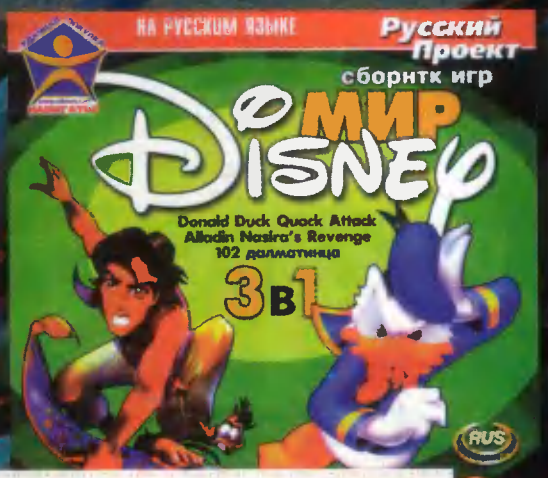
На УКРАИНЕ:

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

В БЕЛАРУСИ:

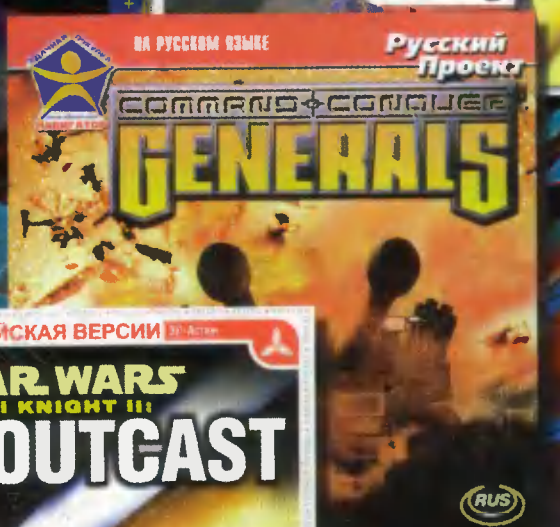
В Минске в магазинах **"Книга XXI век"**, пр.Ф.Скорины, д.92, тел. (017) 264-27-97 (ст.метро "Московская") и **"Глобус"**, ул.Володарского, д.16, тел. (017) 227-30-67 (ст.метро "Площадь Независимости").


Unreal II
THE AWAKENING


НА РУССКОМ ЯЗЫКЕ
Русский Проект
сборнтк игр
МИР Disney
Donald Duck Quack Attack
Aladin Nasira's Revenge
102 далматинца
3 в 1
RUS


РУССКАЯ ВЕРСИЯ
Donald Trump's
REAL-ESTATE
TYCOON
+АНГЛИЙСКАЯ ВЕРСИЯ
2 в 1


РУССКАЯ И АНГЛИЙСКАЯ ВЕРСИИ
3D Action
BATTLEFIELD
1942

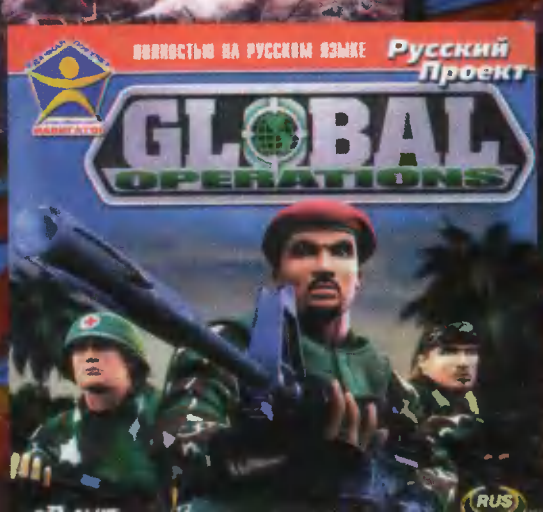

НА РУССКОМ ЯЗЫКЕ
Русский Проект
COMMAND & CONQUER
GENERALS
RUS


РУССКАЯ И АНГЛИЙСКАЯ ВЕРСИИ
STAR WARS
JEDI KNIGHT II:
JEDI OUTCAST
PC CD ROM


2 в 1
РУССКАЯ + АНГЛИЙСКАЯ ВЕРСИЯ
RUSSIAN & ENGLISH VERSIONS
AIRPORT
TYCOON 2
3D MANAGEMENT SIMULATION
TAKE-TWO COMPANY
TEEN T
EVERYONE E


КАЗАКИ
СНОВА ВОЙНА


СБОРНИК
ФУТБОЛЬНЫЕ
менеджеры 2003
CHELSEA OFFICIAL
LAZIO OFFICIAL
MANCHESTER CITY
TOTAL CLUB
VIALLI EUROPEAN
ОБЗОРЫ ИГР
ССЫЛКИ
PC CD ROM
5 в 1


НОВОСТЬМИ НА РУССКОМ ЯЗЫКЕ
Русский Проект
GLOBAL
OPERATIONS
RUS