РЕВЮ ZX

1-2 1997

121019,
Москва,
а/я 16,
МКП
"Инфорком"

ZXРевю '97

Выпуск 1-2

СОДЕРЖАНИЕ Стр

ОТ РЕДАКЦИИ	3
СПЕКТРУМ В ШКОЛЕ	5
А. Пугачев. Метод Зайделя решения СЛАУ	5
НОВЫЕ ПРОГРАММЫ	7
И. Рощин. ABCDcopy v. 1.1	7
И. Рощин. FLOPPY FORMAT v.1.00	8
RST7. TASM4	9
ВИЗИТНАЯ КАРТОЧКА.	10
Представляем известные электронные журналы	
группа STEP. "SPECTROFON" - 21 номер	10
группа XL DESIGN Inc. "ZX FORMAT" - 6 номеров	19
Новый электронный журнал	
UNATED CODERS SOFTWARE&ENIGMA GROUP. "ZX POWER"	21
TR-DOS ДЛЯ НАЧИНАЮЩИХ	23
В. Сироткин. Глава 5. Интегральная микросхема КР 1818ВГ93.	23
ЧИТАТЕЛЬ-ЧИТАТЕЛЮ	36
П. Федин. О железе, софте и прочем...	36
А. Шуравин. Восстановление утерянной информации.	40
КОМПЬЮТЕРЫ, КОТОРЫЕ МЫ ВЫБИРАЕМ	44
А. Соков. Доработка компьютера "ATM-TURBO".	44
ЭТЮДЫ	47
AmiS. Простейшие приемы для написания грамотных программ.	47
М. Лупан. Загрузчик с индикатором и процедура SPRITE OUT.	48
Василий. Несколько полезных советов для написания оптимальных процедур.	51
А. Лавров. Процедуры быстрого скроллинга экрана.	53
Е. и А. Федяевы. Процедура рисования круга без использования вычислений с плавающей точкой.	54
Тимоти. Несколько процедур для системных программистов.	56
Д. Хороших. Процедура заполнения экрана.	57
И. Командин. Процедуры проявления экрана.	60
С. Колотов. Примеры улучшения процедур.	64
Е. Куликаев. Процедура оригинальной очистки экрана.	70
Процедура вывода картинки.	72
Процедура увеличения изображения.	74
П. Старков. Процедура загрузки/выгрузки кодового блока без использования системных переменных.	75
В. Мишуков. Процедура печати 42 символов в строке.	78

РЕТРО	80
Дж. Хардман, Э. Хьюзон. 40 лучших процедур (пер. Н.А. Тихоновой)	80
ФОРУМ	104
С. Колотов. Единый стандарт программирования на ассемблере.	104
И. Рощин.. Доступ к порту #1F в TR-DOS 5.03.	107
И. Рощин.. О сокращении времени форматирования.	112
Д. Анисимов. Еще раз опрограмме "ЖИЗНЬ".	113
А. Колесов. Изменения в мини-драйвере дисковых операций.	118
С. Бандура. Вызов TR-DOS из меню Бейсика-128.	119
ФОРУМ-ИГРЫ	120
В разделе рассмотрены: **Dun Darach, Murray Mause (Supercop), Wild West Seymour, Sim City, Hero Quest, Dizzy Y, Apollo, Cliffhanger, Laser Squad, Movie, Elite, Where Time Stood Still, Sceptre, Carrier Command, Match Day.**	120
ПЕРЕКРЕСТОК	129
В разделе рассмотрены: **Matt Lucas Florida Detective, Human Torch and The Thing, Экзаменационное Расследование, Витязь Светогор, Snowball**	129
ПЕРЕКРЕСТОК ДРАКОНОВ (Автор В.Л. Овечкин)	131
Dangeons&Dragons. Adventure 1	136
СОВЕТЫ ЭКСПЕРТОВ	158
И. Командин. Wellington At Waterloo.	158
И. Командин. Napoleon.	162
А. Мизинов. Star Fox	165
КОМПЬЮТЕРНАЯ НОВЕЛЛА	170
J.C.L. from "L.A." Безумный Зверь.	170
А. Березинов. Русалка.	174
ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ	177
СПРАВОЧНАЯ ИНФОРМАЦИЯ	192

ЗДРАВСТВУЙТЕ, ДОРОГИЕ ДРУЗЬЯ !

Приветствуем и поздравляем с наступившим Новым Годом наших постоянных читателей и тех, кто держит в руках ZX РЕВЮ впервые!

Вступая в седьмой год своего существования, мы, как водится, начнем рассказ о наших планах со "СТАРОЙ ПЕСНИ О ГЛАВНОМ 2" Да... В прошлом году было полтора, а теперь вот ДВА. Догадались? Правильно принято решение об увеличении объема ZX РЕВЮ'97 в 2 РАЗА! Мы выпустим шесть номеров двойного объема. Это судьбоносное решение, которое, как мы надеемся, вас обрадовало и которое определяет все наши планы, связанные как с содержанием ZX РЕВЮ, так и с его распространением.

Приглашаем всех, кто может написать в журнал что-то интересное и полезное для остальных, активизировать свои усилия в этом направлении, по крайней мере, в 2 раза. Помните, в конце прошлого года мы объявили очередной конкурс на лучшие статьи? В конце 1997-го будем подводить итоги с хорошими призами!

Удвоение объема позволило нам одновременно с сохранением прежних рубрик ввести новые - и этот процесс будет продолжаться. Останутся, как и прежде, ваши любимые разделы: Новые программы, Читатель-читателю, Форум, Перекресток, Советы экспертов и другие. Новое развитие получил раздел "TR DOS для начинающих" теперь он очень пригодится и опытным программистам. Но не называть же его "TR DOS для кончающих"? Название пришлось оставить. В разделе "Этюды" начат мозговой штурм по поводу создания "жемчужин" программистской мысли, т.е. если у Вас в голове есть жемчужины, отдайте их нам.

Любители "умных" игр получат в этом году подарок в виде новой рубрики "Перекресток драконов", содержащей в себе великолепную книгу В.Овечкина с одноименным названием.

Несколько слов хочется сказать специально для начинающих пользователей SPECCY. Если этот номер ZX РЕВЮ показался Вам слишком сложным для понимания, не расстраивайтесь. Ведь если Вы поднимите весь пласт информации предыдущих номеров ZX РЕВЮ и

других наших книг, сможете его постепенно освоить, не ленясь и не пропуская трудных мест быть Вам самым крутым кодером от Москвы до самых до окраин.

Шесть лет тому назад начали мы печатать самые разнообразные материалы по ZX SPECTRUM. Многие из них весьма актуальны и сегодня, особенно это касается образовательных статей. Мы много раз перепечатывали сборники ZX РЕВЮ за 1991 и 1992 гг. их суммарные тиражи превысили уже все пределы... Но наступил момент, когда они в своем прежнем виде устарели как физически (пришли в негодность типографские пленки), так и морально (многие тогдашние статьи уже неактуальны). Поскольку лучшие материалы прошлых лет были бы очень полезны тем читателям, которые стали ими недавно, мы ввели рубрику "Ретро", в которой начинаем печатать книгу Хардмана и Хьюзон "40 лучших процедур", опубликованную ранее в ZX РЕВЮ/92.

Теперь о распространении. Тех, кто подписался на весь год (с N1 по N 6), мы призываем оформить дополнительную подписку с N 7 по N 12. Что поделаешь сами ведь просили: "Нам бы потолще, да почаще"

Особое сообщение для тех, кто подписался на первое полугодие через почтовый каталог "Книга-Сервис" По этой подписке Вы должны были получить 3 выпуска ZX РЕВЮ, а получите 4 (1+2, 3+4). Для Вас специально организована подписка до конца года с N 5 по N 12.

В нашем Прайс-листе, который прилагается к этому номеру, Вы найдете ответ на все ваши вопросы.

На этом разрешите откланяться до следующего номера и еще раз пожелать Вам всего доброго!

ИФК.: Пугачев Андрей Николаевич из г.Ковров, Владимирской области, предлагает на суд читателей ZX РЕВЮ свой материал, посвященный численным методам.

КОРР: в ZX РЕВЮ 1993 года было напечатано несколько статей о численных методах. Но потом эта очень интересная тема была забыта. Чтобы как то вернуть справедливость, я специально для журнала написал три программы на бейсике по численным методам. Это программа решения СЛАУ "методом Зейделя" (очень полезная вещь является модификацией метода простых итераций), и две программы по поиску минимума функций: метод золотого сечения и метод дихотомии.

МЕТОД ЗЕЙДЕЛЯ РЕШЕНИЯ СЛАУ

Позволяет получить решение системы с заданной погрешностью за ограниченное количество итераций. Метод сходится при условии, что диагональные коэффициенты системы по абсолютной величине равны или больше любого другого коэффициента в своей строке.

$$\left|\begin{array}{cccc||c} A11 & A12 & A13 & A1n & B1 \\ A21 & A22 & A23 & A2n & B2 \\ A31 & A32 & A33 & A3n & B3 \\ - & - & - & - & - \\ An1 & An2 & An3 & Ann & Bn \end{array}\right|$$

Программа на Бейсике:

```
5 REM Ввод
10 DIM A(30,30)  DIM B(30)  DIM Z(30)  DIM D(30)  DIM X(30)
20 INPUT "Количество уравнений (<=30)";KOLUR
40 INPUT "Погрешность метода";EPS
50 FOR I=1 TO KOLUR
60 FOR J=1 TO KOLUR
70 INPUT "Введите A[i,j]";A(I,J)
80 PRINT "A[";I;",";J;"]=";A(I,J)
90 NEXT J
100 INPUT "Введите B[i]";B(I)
110 LET Z(I)=B(I)/A(I,I)
120 PRINT "B[";I;"]=";B(I)
130 NEXT I
135 REM Analiz
140 FOR I=1 TO KOLUR
150 FOR J=1 NJ KOLUR
160 IF ABS (A(I,I)) < ABS (A(I,J)) THEN PRINT "Нет сходимости!": STOP
170 NEXT J
180 NEXT I
185 Solution
190 LET S=0  REM Счетчик итераций
200 LET K=0  REM Признак точности решения
210 FOR I=1 TO KOLUR
220 LET D(I)=-B(I)
230 FOR J=1 TO KOLUR
```

```
240 LET D(I)=D(I)+A(I,J)*Z(J)   REM
Вычисление "невязок"
250 NEXT J
260 IF ABS (D(I)/A(I,I)) > EPS THEN
LET K=1: REM Точность не достигну-
та
270 LET X(I)=Z(I)-D(I)/A(I,I)   REM
Корректировка решения
280 LET Z(I)=X(I)   REM Заполнение
для использования на данной итера-
ции
290 NEXT I
300 LET S=S+1
310 PRINT "Итерация номер ";S
320 FOR I=1 TO KOLUR
330 PRINT "X[";I;"]=";X(I);" delta_X=";-
D(I)/D(I,I)
340 NEXT I
350 PRINT "Для продолжения на-
жмите Enter"
355 PAUSE 0
360 IF K<>0 THEN GO TO 200   REM
Точность достигнута
365 REM Result
380 PRINT "Результаты:
390 PRINT "Количество итераций =
"; S
400 FOR I=1 TO KOLUR
410 PRINT "X[";I;"]=";X(I)
420 NEXT I
```

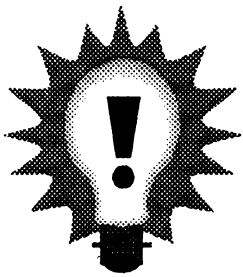
Метод "Золотого сечения" поиска минимума функции y=F(x)

```
10 LET C=0.618   LET D=0.382
13 INPUT "Начало отрезка = ";A
INPUT "Конец отрезка = "; B   IF
A>=B THEN GO TO 13
15 INPUT "Допустимая погрешность
= "; EPS   GOSUB 58   GOSUB 70
21 IF Y1>Y2 THEN GO TO 36
26 LET B=X2   LET X2=X1   LET
Y2=Y1
27 GOSUB 58   GO TO 43
36 LET A=X1   LET X1=X2   LET
Y1=Y2
37 GOSUB 70
43 IF ABS (B-A) > EPS THEN GO TO
21
50 PRINT "Результат:
52 LET X=(A+B)/2   PRINT
"X_min=";X   STOP
58 LET X1=C*A+D*B   LET X=X1
GOSUB 82   LET Y1=Y   RETURN
70 LET X2=D*A+C*B   LET X=X2
GOSUB 82   LET Y2=Y   RETURN
82 LET Y=..............   REM Здесь
вводится исследуемое уравнение
```

Поиск минимума одномерной функции методом "дихотомии"

```
10 DEF FN F(X)=...............   REM
Здесь вводится функция
20 INPUT "Начало отрезка "; A
30 INPUT "Конец отрезка "; B
40 INPUT "Допустимая погрешность
; EPS
50 REM Metod
60 LET I=0
70 LET C=(A+B)/2
80 IF FN F(C-EPS/10) < FN
F(C+EPS/10) THEN GO TO 100
90 LET A=C-EPS/10   GO TO 110
100 LET B=C+EPS/10
110 LET I=I+1
120 PRINT "C=";C;"Итерация номер:
"; I
130 IF (B-A) > EPS THEN GO TO 70
140 PRINT "Результат: "
150 PRINT "X_min=";C;" Количество
итераций = "; I
```

НОВЫЕ ПРОГРАММЫ



ИФК: Представляем сегодня работы молодого автора из Москвы Ивана Рощина. Его программы, которые мы включили в сборник системных программ СД20 (см.Прайс-лист) показывают в своей "весовой категории" наилучшие результаты.

Достаточно сказать, что мы в Инфоркоме и Формаке перешли на использование копировщика И.Рощина для своих производственных нужд. Сами понимаете, что для нас крайне важна достоверность копируемой информации, да и каждая секунда на учете. Итак, слово автору.

(с) И.Рощин, г.Москва. ABCDcopy v.1.1

Предлагаемая вашему вниманию программа предназначена для быстрого и удобного копирования или тиражирования дисков.

Основные достоинства программы:

- повышенная устойчивость в работе. Программа не зависает и не сбрасывается, корректно обрабатывает все ошибки при работе с диском;
- возможность использования от одного до четырех дисководов;
- по выбору пользователя можно включать или отключать проверку при копировании;
- можно настроить быстродействие пролграммы в зависимости от качества используемого дисковода;
- по выбору пользователя возможно как копирование целого диска, так и только занятой файлами области;
- наглядное изображение процесса копирования;
- - можно копировать в прямом (0 - 159) или обратном (159 - 0) направлении;
- имеется возможность сохранения выбранных параметров копирования (Save setup);
- использование специальных алгоритмов с целью обеспечения максимальной скорости копирования;
- можно указать количество попыток повторения для операций чтения или записи;
- автоматическое определение необходимости форматирования диска копии (впрочем, при желании можно его отключить);
- возможность использования дополнительной памяти в качестве буфера для копирования. Программа может работать на компьютерах с памятью 48, 128, 256 и 512 килобайт (расширение с помощью порта #7FFD). Если написать соответствующий драйвер (как это сделать указано в описании),

то можно использовать любую дополнительную память;

- - если в памяти умещается вся копируемая информация, то она считывается с диска-оригинала за один раз, а затем можно записывать любое количество копий;
- предусмотрен временный выход из программы в TR DOS, чтобы, например, посмотреть каталог диска;
- имеется подробное описание в формате ZX WORD (93 сектора).

FLOPPY FORMAT v.1.00

Эта программа для форматирования гибких дисков не только универсальна, но она еще и самая безопасная! FLOPPY FORMAT реализует следующие возможности:

- форматипрование дискет для трех операционных систем: TR DOS, IS DOS, MS DOS;
- поддержка стандартных и нестандартных форматов дискет (всего 25 форматов);
- естественно, форматирование производится с проверкой качества;
- - проверка содержимого дискеты перед форматированием. Программа всегда предупреждает пользователя о том, что форматируемая дискета содержит какую-либо информацию;
- возможность, так называемого, "фиктивного" форматирования (очищается таблица размещения файлов и корневой каталог) для уже ранее отформатированных дискет. FLOPPY FORMAT сам распознает ситуации, когда возможно "фиктивное" форматирование;
- - при "фиктивном" форматировании TR DOS дисков, информация, которая была на нулевой дорожке, переписывается на 160-ю, так что содержимое дискеты всегда можно восстановить;
- перед форматированием TR DOS диска, на котором уже было что-то записано, программа считывает и хранит в памяти содержимое нулевой дорожки. Если во время форматирования Вы вдруг вспомнили, что на диске была записана ценная информация, достаточно нажать BREAK, и нулевая дорожка будет восстановлена, если, конечно, нужная область диска не была уже отформатирована;
- отформатированные с помощью FF диски нормально читаются на эмуляторах;
- на дискеты MS DOS при форматировании записывается оригинальный загрузчик (BOOT-сектор), как в аналогичной программе на PC;
- - как минимум, в 1,5 раза увеличивается производительность MS DOS дискет за счет форматирования со смещением секторов;
- - имеется Screen Saver;
- программа устойчива в работе, имеет возможность временного выхода в TR-DOS, имеет подробное описание в формате ZX WORD (104 сектора);
- имеется контекстная справочная система (доступна в

любой момент работы программы при нажатии клавиши "Н"

RST7/CBS, г. Харьков
Интегрированный комплекс разработчика для программирования на ассемблере TASM4.

"Впервые на Спектруме сделана НОРМАЛЬНАЯ ИНТЕГРИРОВАННАЯ СИСТЕМА: Вы пишете текст в редакторе, ассемблируетесь, и если получаете кучу ошибок, возвращаетесь в редактор и нажатием пары клавиш переходите по строкам, в которых найдены ошибки. Если не помните, как работает та или иная команда, давите другую пару клавиш и попадаете в гипертекстовый Help, прямо в описание того слова, на которое указывает курсор. Пока Help сделан только по директивам ассемблера, но в ближайших планах посадить туда еще и полное описание команд Z-80... такими словами характеризует уровень интеграции в своей новой разработке сам автор.

По сравнению с предыдущим версиями в интегрированный комплекс внесены ряд изменеий и дополнений выгодно отличающих его от существующих аналогичных программных продуктов.

Редактор:

- -значительно увеличена скорость работы,
- -изменен формат текста (для уменьшения размеров исходного файла),
- -введены удобные команды перемещения по тексту (например, на последнюю измененную строку),
- -добавлена команда поиска меток.
- -исправлены ошибки.

Ассемблер:

- -увеличена скорость,
- -сняты ограничения по длине меток,
- -сделаны НОРМАЛЬНЫЕ макроопределения, с передачей параметров на уровне текста,
- -введены локальные метки,
- -добавлена возможность условного ассемблирования,
- -сделаны LX/LY/HX/HY,
- -поправлены замеченные ошибки.

ВИЗИТНАЯ КАРТОЧКА

ИФК: По многочисленным просьбам читателей мы сегодня представим Вашему вниманию электронные средства массовой информации. В отличие от Радио и Телевидения (которые обычно и попадают под термин ЭСМИ), речь пойдет об электронных журналах по ZX SPECTRUM, распространяемых на дискетах.

В последнее время электронные журналы плодятся с неимоверной скоростью. Действительно, что может быть проще: собрался с парочкой друзей, слепил немудреную оболочку, набил туда любых текстов (что накипело) и объявил всем, что в городе N впервые появился замечательный журнал...!

Мы в редакции получаем огромное количество таких произведений, и поэтому есть возможность выбрать и предложить Вашему вниманию действительно достойные продукты.

Начнем с самого первого и самого лучшего (по мнению читателей) до сегодняшнего дня журнала SPECTROFON, выпускаемого группой STEP, г.Москва.

К данному моменту вышел в свет уже 21 номер журнала, и в связи с этим ЮБИЛЕЕМ мы объявляем акцию расширенной продажи журнала под названием "ОЧКО" (см. наш Прайс-Лист). Каждый номер журнала SPECTROFON - это веха в развитии Спектрумовского движения. Судите сами.

SPECTROFON N1

ЭКСПЕРТИЗА. Рассмотрена игра "ENIGMA FORCE", выпущенная фирмой Beyond в 1985 г.

ДЕБЮТ. Программы DRILLER (INCENTIVE, 1987) и KAYLETH (ADVENTURE SOFT, 1987).

СИСТЕМА. Программа "ADVENTURE BUILDER SYSTEM" переведенная на русский язык.

ОБЗОР. Рассказывается о новинках, не так давно появившихся в продаже в г.Москве.

ЭКЗАМЕН. Предлагается решить задачу, в основу которой положен эпизод из игры "DRILLER"

ПРИЛОЖЕНИЕ. Представлены все игры, рассмотренные на страницах журнала и программа для разработки адвентюрных игр "ABS" с примером ее использования.

SPECTROFON N2

ЭКСПЕРТИЗА. Игра "SHADOWFIRE", которая является продолжением игры "ENIGMA FORCE"

ДЕБЮТ. Описания двух интересных игр: "DEACTIVATORS" и "VENDETTA"

ОБЗОР. Новые подробности о новосибирской версии игры "ELITE" В статье "Не надо нам такого порядка" дискуссия на тему: "IS-DOS: проблемы развития"

СИСТЕМА. Музыкальный редактор ASM А.Сендецкого. Приводится авторская инструкция по работе с редактором.

ЭКЗАМЕН. Вам предлагается решить игровой этюд по программе "DEACTIVATORS"

ПРИЛОЖЕНИЕ. Приводятся все игры, рассмотренные на страницах журнала и последняя версия музыкального редактора ASM.

SPECTROFON N3

ЭКСПЕРТИЗА. Игра "GENGHIS KHAN" Вам представляется возможность повторить завоевания татаро-монгол.

ДЕБЮТ. "ZZZZ" оригинальная адвентюрная юмористическая игра. "CASTLE MASTER" - игра, которую отличает трехмерная закрашенная графика и обилие головоломок.

ОБЗОР. Рассказ о новых играх для "ZX-SPECTRUM" Двадцатка игр, пользующихся наиболее высоким спросом.

АРХИВ. Рассмотрена игра "GREAT BRITAIN Ltd", в которой Вы можете попробовать себя на посту премьер-министра.

С МИРУ ПО БИТУ Обзор почты. Ответы на вопросы читателей журнала.

СИСТЕМА. Полезные советы по работе в TR-DOS.

ПРИЛОЖЕНИЕ. Приводятся дисковые версии всех игр, рассмотренных на страницах журнала.

SPECTROFON N4

ЭКСПЕРТИЗА. Подробно рассмотрена игра "HOSTAGES" Вы участник операции "Юпитер"

ДЕБЮТ. Представлены две игры: "ESPIONAGE ISLAND" адвентюра и "CRYPT" продолжение игры "CASTLE MASTER"

ЭКЗАМЕН. Подводятся итоги по задачам, заданным в предыдущих номерах.

ФАНТАЗИЯ. Новелла по игре "Viaje at centre of Terra"

ОБЗОР. Статья "Субъективные заметки независимого эксперта"посвещена новосибирской версии "ELITE-3"

С МИРУ ПО БИТУ Обзор почты журнала, ответы на вопросы, редакционное обращение ко всем коллекционерам игр и немного фантастики.

СИСТЕМА. Музыкальные редакторы "WHAM" и "WHAM+AY" - новые версии этих программ вошли в ПРИЛОЖЕНИЕ.

ПРИЛОЖЕНИЕ. Представлены дисковые версии игр из разделов ЭКСПЕРТИЗА, ДЕБЮТ и ФАНТАЗИЯ.

SPECTROFON N5

ЭКСПЕРТИЗА. Представлена экспертная проработка стратегической игры "THEATRE EUROPE"

ДЕБЮТ. Подробно рассматривается одна из самых увлекательных адвентюр "ROBIN OF SHERWOOD"

ОБЗОР. И вновь разговор о новосибирской версии игры "ELITE-3"

АРХИВ. Подробно рассмотрена стратегическая игра "1812"

ФАНТАЗИЯ. Компьютерная новелла по игре "NIGHT RUN"

СИСТЕМА. Полная дисковая версия полупрофессиональной инструментальной системы для создания адвентюрных игр "GAC"

ПРИЛОЖЕНИЕ. Дисковые версии всех рассмотренных игр.

SPECTROFON N6

ЭКСПЕРТИЗА. Рассмотрена игра жанра Dungeons & Dragons "HEROES OF THE LANCE"

ДЕБЮТ. Рассмотрены дебютные проблемы двух игр: "THE MUNSTERS" и "STAR PAWS"

ОБЗОР. Рассказ о новых играх, появившихся на рынках Москвы, а также хит-парад бестселлеров.

С МИРУ ПО БИТУ. Обзор читательской почты: вопросы стандартизации, проблемы IS-DOS и др.

СИСТЕМА. К вопросу о нестабильности шины данных ваших компьютеров. Публикуется схема подключения музыкального сопроцессора.

ПРИЛОЖЕНИЕ. Дисковые версии игр: "HEROES OF THE LANCE", "THE MUNSTERS", "STAR PAWS 48K" и "STAR PAWS 128K"

SPECTROFON N7

ЭКСПЕРТИЗА. Рассмотрена игра "BATTLE COMMAND" (128K). Вы в роли командира современного танка.

ДЕБЮТ. Представлена игра "THE SENTINEL" ("СТРАЖ"). Любители трехмерных игр по достоинству оценят идею этой игры.

ОБЗОР. Вопросы дистрибуции авторских программ.

С МИРУ ПО БИТУ Письма читателей. Главные темы музыка, "ELITE", ответы на вопросы и др.

ТАЙНИК. Новый раздел для тех, кто любит устанавливать в играх "бессмертие", "бесконечную" энергию и др.

ГОРЯЧИЙ ПРИВЕТ. Юмористический рассказ на компьютерную тематику.

ПРИЛОЖЕНИЕ. Игры "BATTLE COMMAND"(128K)+10 levels и "THE SENTINEL"

SPECTROFON N8

ЭКСПЕРТИЗА. Представлены две части ролевой игры "HERO QUEST" с отгрузкой на диск.

ДЕБЮТ. Рассмотрены дебютные проблемы игры: "THEY STOLE A MILLION" шедевра в своем жанре.

ОБЗОР. Новые игры и традиционный хит-парад.

С МИРУ ПО БИТУ Большой блок посвящен письмам читателей. Статья-пародия на различные виды описаний игр.

ПРИЛОЖЕНИЕ. Вошли дисковые версии игр "HERO QUEST 1,2" (128K), "HERO QUEST" (48K) и "THEY STOLE A MILLION"

SPECTROFON N9

ЭКСПЕРТИЗА. Сетевая версия стратегической игры "LASER SQUAD" Позволяет играть с партнером по телефону. Включен четвертый, дополнительный сценарий.

ДЕБЮТ. Фантастическая адвентюрная игра "MIND-

FIGHTER" В этом же разделе представлена игра "MARSPORT"

ЭКЗАМЕН. Разбор предыдущих заданий. Полные характеристики персонажей игры "THEY STOLE A MILLION"

ОБЗОР. Новые игры и анонс программы "Звездное наследие"

СИСТЕМА. Подробно освещается работа с принтером. Статья для начинающих пользователей.

ПРИЛОЖЕНИЕ. Вошли полные дисковые версии игр: "LASER SQUAD"(128K), "MINDFIGHTER", "MARSPORT"

SPECTROFON N10

ЭКСПЕРТИЗА. Великолепная игра "HIJACK" (Electric Dreams), 1986 г. В роли госсекретаря США Вам нужно ликвидировать банду, захватившую школьный автобус.

ДЕБЮТ. Раздел полностью посвящен игре "MARSPORT"

ПРЕЗЕНТАЦИЯ. Фантастическая детективная игра "ЗВЕЗДНОЕ НАСЛЕДИЕ" Игра относится к адвентюрному жанру с элементами ролевых игр. В ПРИЛОЖЕНИЕ включена демонстрационная версия первой части игры.

ОБЗОР. В разделе рассматриваются авторские разработки Н.Величутина, обучающие программы для детей, программа "ЗВУК" позволяющая оцифровывать аналоговые сигналы.

С МИРУ ПО БИТУ. Письма читателей, дискуссия по проблемам программного обеспечения для "ZX-SPECTRUM" Заявки на публикацию авторских программ.

СИСТЕМА. Статья "Турбирование ВГ93"

ПРИЛОЖЕНИЕ. Программы: "ЗВЕЗДНОЕ НАСЛЕДИЕ" (ДЕМО-верс.),"HIJACK","G.A.C.D.", "ЗВУК" и др.

SPECTROFON N11

ЭКСПЕРТИЗА. "SWORDS & SORCERY" Предшественница популярных ныне ролевых игр. Она приятно удивит Вас обширным игровым пространством, обилием монстров, чудовищ и колдунов.

ШТУРМ. Новый раздел посвящен разбору трудных игр, рассмотренных в предыдущих номерах. В этом номере читатели журнала штурмуют загадочный "MINDFIGHTER"

ПРЕМЬЕРА. Представлены две авторские разработки: 1."NOTEBOOK" "ЗАПИСНАЯ КНИЖКА" 2. "COLOR LINES" логическая игра, аналог игры для IBM PC. Здесь же комикс "СМЕРТЕЛЬНОЕ КАСАНИЕ", о боевом искусстве ниндзя.

ОБЗОР. Шахматные программы для "ZX-SPECTRUM"

С МИРУ ПО БИТУ. Письма читателей, статья-пародия на "бегущие строчки"

СИСТЕМА. "Новое о Z-80 или кое-что о недокументированных командах процессора и архитектуре "ZX-SPECTRUM"

ПРИЛОЖЕНИЕ. Дисковые версии игр "SWORDS & SORCERY", "COLOR LINES" и программа "NOTEBOOK"

SPECTROFON N12

ЭКСПЕРТИЗА. Новинка сезона фантастическая игра стратегического жанра "SPACE CRUSADE"(128K). Для владельцев компьютеров 48K в в

ПРИЛОЖЕНИЕ включена упрощенная версия программы.

ЭКЗАМЕН. Разбор предыдущих заданий.

ПРЕМЬЕРА. Представлены две авторские программы: 1."СЧИТАЛКА" - обучение устному счету. 2.LIFE позволяет исследовать математическую модель деления клеток. Здесь же кое-что о вирусных играх.

ОБЗОР. Новые отечественные и зарубежные игровые программы. Новое слово в игровых программах - вирусные игры.

С МИРУ ПО БИТУ Обсуждение демоверсии игры "ЗВЕЗДНОЕ НАСЛЕДИЕ", разговор о новосибирской "ELITE" и др.

СИСТЕМА. Представлена исправленная и доработанная схема турбирования ВГ-93.

ПРИЛОЖЕНИЕ. Дисковые версии игр "SPACE CRUSADE" (128), "SPACE CRUSADE"(48), "СЧИТАЛКА" и программа моделирования "LIFE"

SPECTROFON N13

ЭКСПЕРТИЗА. Рассмотрена уникальная игра "TAU CETI 3", одна из лучших в своем жанре.

ДЕБЮТ. Раздел посвящен адвентюрной фантастической игре "WORM IN PARADISE", которую написала известная фирма LEVEL 9.

ПРЕМЬЕРА. Речь пойдет об еще одной версии программы "COLOR LINES", которую написали московские программисты.

ШТУРМ. Как все-таки "прикончить" террористов, захвативших автобус в игре "HIJACK"?

ОБЗОР. Вы узнаете о новых зарубежных программах, познакомитесь с готовящейся к выходу в свет стратегической программой "SPECIAL FORCES" (STEP CREATIVE GROUP, 1995), продолжающей славные традиции незабываемого "LASER SQUAD" Большой блок раздела посвящен игре "UFO. ENEMY UNKNOWN" (НЛО. ВРАГ НЕИЗВЕСТЕН), которую адаптировал под "SPECCY" В.Медноногов из Санкт-Петербурга. Приведено авторское описание спектрумовской версии игры.

С МИРУ ПО БИТУ Обзор писем читателей, кое-что о программе "VIRUS" Здесь Вы можете прочитать об условиях проведения и участия в Открытом Чемпионате Вирусов, который пройдет под эгидой редакции журнала "SPECTROFON"

СИСТЕМА. В разделе подробно рассмотрен музыкальный редактор для AY "SOUND TRACKER"

ПРИЛОЖЕНИЕ. Программы "TAU CETI 3"(рус),"WORM IN PARADISE", "COLOR LINES", "ПОЛЕ ЧУДЕС" (демо), "SOUND TRACKER" и "SOUND TRACKER COMPILER"

SPECTROFON N14

ЭКСПЕРТИЗА. Рассмотрена известнейшая игра Пита Кука "АКАДЕМИЯ" В приложении Вы найдете полностью русифицированную и восстановленную версию игры плюс один экстра-уровень (звездная карта), в котором можно назвать звезду своим именем после успешного завершения обучения в Академии Гая Корна.

ДЕБЮТ. Редакция журнала продолжает рассматривать игры фирмы Level 9. На этот раз Вашему вниманию предложена игра

"EMERALD ISLE" Вам предстоит преодолеть все невзгоды и препятствия и стать королем загадочного острова.

АРХИВ. В этом разделе подробно рассмотрена игра "ARCHON" Вам предлагается сразиться с компьютером или с живым противником в сказочные шахматы, где вместо традиционных шахматных фигур участвуют драконы, ведьмы и т.д.

ПРЕМЬЕРА. Рассмотрены две отечественные программы. Первая музыкальный редактор "PRO SOUND MAKER (DEMO)" великолепная программа, сочетающая в себе достоинства "S.T." и "ASM", а по удобству в работе превосходящая любые отечественные и зарубежные аналоги. Вторая программа "VISUAL DECOMPRESSOR" позволит Вам использовать различные эффекты при декомпрессии экранных файлов.

ЧЕМПИОНАТ. Репортаж с проходящего под эгидой журнала Открытого Чемпионата Вирусов.

СИСТЕМА. В этом разделе рассматриваются вопросы, касающиеся совместимости и модификации отечественных SPECCY. Кроме этого, традиционные "С МИРУ ПО БИТУ", "ОБЗОР", "РЕКЛАМА" и "ПРИЛОЖЕНИЕ", в которое вошли программы "АКАДЕМИЯ"(рус), "ARCHON", "EMERALD ISLE", "PRO SOUND MAKER", "VISUAL DECOMPRESSOR"

SPECTROFON N15

ЭКСПЕРТИЗА. Рассмотрена уникальная игра "CAPTAIN BLOOD" представляющая из себя некий сплав космического квеста и детектива.

ДЕБЮТ. Любители графических адвентюр найдут здесь описание интересной и несложной игры "SIDEWALK"

СИСТЕМА. Недокументированные команды процессора Z-80.

ЧЕМПИОНАТ. В разделе освещается прохождение Открытого Чемпионата Вирусов.

ГОРЯЧИЙ ПРИВЕТ. Крах оригинального проекта "ELITE-3" Рассказ о скандальной истории, связанной с этой игрой.

ПРЕМЬЕРА. В разделе представлены несколько отечественных программ: утилита для определения истинности штрихового кода на товаре, две новые отечественные игры, а также прикладные программы по астрономии и новый мощный ассемблер (демо-версия).

ПРИЛОЖЕНИЕ. Вошли: "CAPTAIN BLOOD", "КАЛЕНДАРИ ПЛАНЕТ, СОЛНЦА И ЛУНЫ", "ZANNI-1", "MONTANA JONES II", "SIDEWALK", "EAN-13" и "MASTER ASSEMBLER" (demo).

SPECTROFON N16

ЭКСПЕРТИЗА. В разделе подробно разобрана игра "HACKER" Вам необходимо разрушить коварные планы могущественной корпорации "МАГМА"

С МИРУ ПО БИТУ Ответы на вопросы читателей. Интервью с творческой группой "STEP"

СИСТЕМА. Кое-что о проблемах совместимости компьютеров. Основы программирования музыкального процессора AY-8910/12.

ЧЕМПИОНАТ. Финал чемпионата вирусов по игре "ВИРУС-1"

КОНСТРУКТОР. Универсальная схема дешифрации порта расширения для машин с дополнительной памятью. Сравнение компьютеров` "PROFI" и "SCORPION"

ФАНТАЗИЯ.
"Невыполнимое задание" компьютерная новелла по мотивам игры "IMPOSSIBLE MISSION"

Кроме этого, ОБЗОР, ПРЕМЬЕРА, РЕКЛАМА и ПРИЛОЖЕНИЕ, в которое вошли программы: "HACKER", "SEA ACTION", "REBEL STAR EDITOR", "IMPOSSIBLE MISSION", "LASER SQUAD EDITOR DEMO", русифицированная версия игры "THEY STOLE A MILLION"

SPECTROFON N17

ЭКСПЕРТИЗА. В разделе подробно рассмотрена вторая часть игры "HACKER" Вам предстоит стать участником захватывающей детективной истории.

ОБЗОР. Обзор новых поступлений на рынки Москвы, а также английский хит-парад лучших игр конца 80-х - начала 90-х годов.

СИСТЕМА. В разделе обсуждаются насущные вопросы стандартизации Спектрум-совместимых компьютеров.

ПРЕМЬЕРА. Раздел посвящен двум программам: "ESPERANTO-48" (обучающая программа) и "Gemmini Commander v.3.2" (нортон-оболочка).

КОНСТРУКТОР. В разделе рассматриваются проблемы, связанные с портом #FD, а также вопросы расширения ОЗУ до 512 кб и реализации ПСЕВДО-ПЗУ

ФАНТАЗИЯ. Компьютерная новелла по игре "LORDS OF CHAOS"

ГОРЯЧИЙ ПРИВЕТ. Приведено интервью с фирмой Muthos Games, известной по таким играм, как "LASER SQUAD", "REBEL STAR", "LORDS OF CHAOS" и др.

КРОМЕ ЭТОГО: Популярный раздел С МИРУ ПО БИТУ, РЕКЛАМА и ПРИЛОЖЕНИЕ, в котором Вы найдете пропрограммы: "LORDS OF CHAOS", "ESPERANTO-48", "HACKER-2" и "GemminiCommander"

SPECTROFON N18

ЭКСПЕРТИЗА. Впервые в нашей стране представлена и рассмотрена "THE BARDS TALES" полноценная семнадцатиуровневая ролевая игра.

ПРЕМЬЕРА. В этот раз в разделе всего одна программа исправленная и переработанная демонстрационная версия игры "PRINCE OF PERSIA" Конверсия с IBM российского программиста Никодима.

ФАНТАЗИЯ. Компьютерная новелла по игре "CAPTAIN BLOOD" Полная исправленная и дискофицированная версия игры включена в приложение к журналу.

СИСТЕМА. Советы начинающим программистам.

КОНСТРУКТОР. Несколько полезных разработок для Вашего компьютера, а также оригинальная схема АЦП. Для владельцев модемов статья о способах улучшения связи.

С МИРУ ПО БИТУ. Почта журнала и продолжение дискуссии о СПЕКТРУМЕ и вокруг него.

ПОМИМО ЭТОГО: ОБЗОР программного обеспечения, РЕКЛАМА и ПРИЛОЖЕНИЕ, в которое вошли: "THE BARDS TALES", "CAPTAIN BLOOD" "PRINCE OF PERSIA.Demo"

SPECTROFON N19

АРХИВ. Раздел посвящен игре "THE DOUBLE", которая относится к жанру business / management. Вам предоставлена возможность стать менеджером английского футбольного клуба.

ДЕБЮТ. Здесь Вы найдете подробную статью о проблемах дебюта в игре "REBEL PLANET", а также описание отдельных миссий к игре "LASER SQUAD"

ФАНТАЗИЯ. "Конец великой русификации" - компьютерная новелла по многим спектрумовским играм.

СИСТЕМА. "Быстрая графика" статья, которая включена в этот раздел.

С МИРУ ПО БИТУ Как и всегда, письма читателей, но помимо этого Вы найдете здесь полезную информацию по играм "BARDS TALE" и "WORM IN PARADIZE"

А также, разделы ОБЗОР, ПРЕМЬЕРА, РЕКЛАМА и ПРИЛОЖЕНИЕ,в котором Вы найдете следующие программы: "THE DOUBLE 128K", "REBEL PLANET", "LASER SQUAD"+7 LEVELS, "ПУТЬ КЛАНОВ"(demo), "RETURN TO HOME"(demo) и свежайший релиз музыкального редактора "SAMPLE TRACKER"

SPECTROFON N20

ЭКСПЕРТИЗА. В разделе подробно рассмотрена увлекательнейшая игра жанра arcade-adventure "SHADOW OF THE BEAST" Изумительная графика, интересный сценарий, превосходная музыка. Тайны этой игры долгое время оставались нераскрытыми, но теперь каждый может насладиться этой великолепной игрой, благодаря новой дисковой версии и подробнейшему описанию, подготовленному С. Новиковым.

ДЕБЮТ. Многим, наверное, известна игра "COLONY" фирмы MASTERTRONIC. Подробный анализ быстрого старта читайте в статье А. Школьникова.

ИНТЕРВЬЮ. Сегодня гость этого раздела сотрудник редакции Спектрофона Д. Григорьев (OLDMAN).

ШТУРМ. На этот раз разговор пойдет о недавно представленной в нашем журнале игре "BARDS TALE" "Подсказки из подземелья" заинтересуют многих поклонников этой игры.

ОБЗОР. Новые игры тема этого раздела.

С МИРУ ПО БИТУ В разделе представлен хит-парад игр, составленный на основе опроса одесских синклеристов. Здесь же читайте большое аналитическое письмо с продолжением дискуссии, начатой в письме хаккерской группы "STARS OF KELADAN"

КОНСТРУКТОР. KEMPSTON MOUSE. В статье В. Ларькова рассказывается о том, как подключить к Спектруму мышь.

ENLIGHT 96. Обзор событий года. Интервью, репортажи... Рассказ о творческом форуме синклеристов.

ГОРЯЧИЙ ПРИВЕТ. Взлет и падение детища сэра Клайва Синклера. История ZX Spectrum. Взгляд со стороны. Печатается с продолжением.

Помимо этого разделы ПРЕМЬЕРА, РЕКЛАМА и ПРИЛОЖЕНИЕ, в которое вошли "SHADOW OF THE BEAST", "COLONY", "DUNA 2", playable demo и другое.

SPECTROFON N21

ДЕБЮТ. В разделе представлена игра "DARK SCEPETRE" представитель жанра RPG, но весьма своеобразный. В игре изумительная графика, крупные, тщательно прорисованные спрайты героев, превосходные звуковые эффекты, великое многообразие команд, удобный интерфейс. Ваша задача - с помощью своего отряда найти и обезвредить Темный Скипетр символ зла. Но это не так просто сделать...

ЗЕРКАЛО. Интервью с Майком Сингльтоном, автором таких известных игр, как "Lords of Midnight", "Doomdark Revenge", "War in the middle earth", "Dark Scepetre" и др.

ОБЗОР. Новые игры на рынках Москвы.

С МИРУ ПО БИТУ В этом разделе, как всегда, письма читателей, а также две небольшие аналитические статьи о спектрумовских играх.

МАСТЕРСКАЯ. Новый раздел. Создан специально для тех, кто любит рисовать. Раздел о художниках и для художников. Представлено несколько работ.

ТАЙНИК. Специальный раздел для любителей "вечностей" Разговор пойдет о тайнах и секретах игр. Маленькие хитрости, POKES, пароли и секретные коды.

СИСТЕМА. В разделе представлен большой материал по портам компьютера PROFI.

КОНСТРУКТОР. Универсальный контроллер периферии SMUC. Так называется большая статья, подготовленная специально для журнала А.Ларченко.

ГОРЯЧИЙ ПРИВЕТ. В гостях у журнала - сэр Клайв Синклер. Разговор с К.Синклером о Спектруме и вокруг него.

ПОМИМО ЭТОГО: ПРЕМЬЕРА, РЕКЛАМА и ПРИЛОЖЕНИЕ, в которое вошли дисковые версии рассмотренных в разделах программ, демо-версия очередной (уже омской) DUNE и другие программы.

Заканчивая обзор трудов группы STEP, нельзя не упоминуть об их программе ЗВЕЗДНОЕ НАСЛЕДИЕ. Впервые отечественная программа оказалась на мировом уровне. От сценария до готовой реализации - это (в отличие от всех других "перепевов" игр с других платформ) абсолютно оригинальное авторское произведение.

Выход этой программы на рынок ознаменовался таким ажиотажем, которого раньше не было. Потоки писем в редакцию ZX РЕВЮ были сравнимы только с обсуждением программы ELITE в прошлые годы. На радиорынках и других синклеровских "тусовках" на территории бывшего СССР лю-

ди платили десятки тысяч рублей только за то, чтобы узнать как пройти трудное место или соответствующий пароль...

Советуем всем, у кого этой программы нет, приобрести ее и получить удовольствие. Эту программу мы также включили в проект "ОЧКО"

****

ZX FORMAT (группа XL DESIGH Inc.)

Теперь представим сравнительно молодой журнал из Санкт-Петербурга, в котором, поначалу очень сильно чувствовалось влияние SPECTROFON'а, но который, похоже, смог найти свой стиль и свою индивидуальность.

ZX FORMAT N1

ИГРУШКИ. 10 популярных игр месяца, обзор новинок, игра "НЛО 1. ВРАГ НЕИЗВЕСТЕН", новелла по игре "KRISTAL KINGDOM DIZZY" (DIZZY 7).

ПРОГРАММИСТАМ. Начальный курс Бейсика. Ассемблер для начинающих, редактор миссий к игре "LASER SQUAD"

IS-DOS. Первое знакомство с системой. Путеводитель пользователя по программам базового комплекта. Для программистов: программа "gmen.com", оконная система IS-DOS. Новинки ISKRASOFT'а: "arzt.com", IS-DOS, v.4.0, монитор-отладчик для IS-DOS, контроллер винчестера.

ЖЕЛЕЗО. Тормоз для компьютера. Блокировка порта #1FFD для SCORPION'а. Как избавиться от устаревших компьютеров? Особенности компьютера KAY-256/35L TURBO. SCORPION ZS256 - что новенького?

ИНТЕРВЬЮ. Рассказ С.Медноногова о своей деятельности программиста.

ПРЕМЬЕРА."COLOR LINES" - новая версия игры. г. С.-Петербург.

ОТДОХНЕМ. Одни день из жизни программиста. Демонстрации: "ZX STAG", "НЛО-2", "OPEN IT", "КОЛОБОК"

ПОЧТОВЫЙ ЯЩИК. Бессмертие в играх. Реклама и объявления.

КОНКУРС. Конкурс на уровни к игре "LASER SQUAD" Конкурс на тему для нового конкурса.

РАЗНОЕ. История развития компьютера АМИГА.

Перспективы: "КОЛОБОК", "НЛО-2", "OPEN IT", "ПЯТАЧОК-СУПЕРАГЕНТ", "СТРАНА ДРАКОНОВ", "THE TURN" "PCX VIEWER", "THE CITY"

ПРИЛОЖЕНИЕ."DIZZY 7", "LASER SQUAD EDITIR", "COLOR LINES", "БЫКИ И КОРОВЫ"

ZX FORMAT N2

ИГРУШКИ. 10 популярных игр месяца, обзор новинок: игра "CARRIER COMAND", новелла по игре "48 УТЮГОВ"

ПРОГРАММИСТАМ.Продолжение курса по Бейсику и ассемблеру для начинающих. Работа с музыкальным редактором "INSTRUMENT 3.01"

IS-DOS. Начинающим системные утилиты; пользователям описание программы "eliminat.com"; программистам оконная система IS-DOS. "Как это сделано?" монитор командной строки. Компьютерные классы на базе ZX Spectrum в среде IS-DOS.

ЖЕЛЕЗО. О новой музыкальной приставке "GENERAL SOUND" Переключатель "TURBO/NORMAL" для SCORPION`a. Турбирование Spectrum-машин.

ИНТЕРВЬЮ. Интервью с С.Зоновым и А.Ларченко.

ПРЕМЬЕРА. "DIGITAL STUDIO" - описание редактора.

ОТДОХНЕМ. "Нервная работа" рассказ-эпизод. Демонстрации: "MICROSTORM", "MINISTAG"

ПОЧТОВЫЙ ЯЩИК. Письма читателей. Реклама и объявления.

РАЗНОЕ. Дальнейшая история развития компьютера АМИГА.

ZX FORMAT N3

ИГРУШКИ. 10 популярных игр месяца, обзор новинок. Новеллы по играм "BATMAN 2.1", "LAST BATTLE" "48 УТЮГОВ" (level 2).

ПРОГРАММИСТАМ.Продолжение курса по Бейсику и ассемблеру для для начинающих. Советы по адаптации игр (дискетирование, защита, бессмертия).

IS-DOS. Начинающим продолжение знакомства с системой; программистам оконная система IS-DOS, формат объектных модулей *.obj и таблицы локальных символов IS-DOS ассемблера.

ЖЕЛЕЗО. Музыкальная карта "GENERAL SOUND" Шинная архитектура Spectrum`a. SCORPION TURBO+. Модемы. "Глюки" TR-DOS.

ПРЕМЬЕРА. "ВИННИ ПУХ" "TASM 4.0 (XLD)"

ОТДОХНЕМ. Глюкодром. Ну, юзер, погоди!

ПОЧТОВЫЙ ЯЩИК. Письма читателей. Доска объявлений.

КОНКУРС. Два новых конкурса.

РАЗНОЕ. АМИГА: вопросы и ответы, новости, перспективы развития программного обеспечения.

ПРИЛОЖЕНИЕ. "ENLIGHT`96 INFO", "UFO-2 demo", "TASM 4.0 (XLD)", "BATMAN 2.1", "ВИННИ ПУХ demo", "СТРАНА ДРАКОНОВ demo"

ZX FORMAT N4

ИГРУШКИ. Новеллы и подсказки по играм "LA ABADIA", "ВИННИ ПУХ (level 2), "48 УТЮГОВ" (level 3).

ПРОГРАММИСТАМ.Продолжение курса по Бейсику, советы по адаптации игр (дискетирование, "вечная жизнь"). Музыкальные редакторы. Теневой монитор ZS-256.

IS-DOS. Начинающим продолжение знакомства с системой. Работа с электронным диском.

ЖЕЛЕЗО. Интервью с фирмой Nemo. Компьютер KAY-256.

ПРЕМЬЕРА. Ассемблеры. Описание программы "CONVER-Commander, v. 4.50 pro" НЛО-2 новые подробности.

ИНТЕРВЬЮ. Капитан Nemo, Н.Родионов, Д.Михайлов о себе и ZX Spectrum.

ОТДОХНЕМ. HAL 9000. Виртуальная реальность.

ПОЧТОВЫЙ ЯЩИК. Письма читателей. Доска объявлений.

РАЗНОЕ. Перспективы ПО, AMIGACLUB, AMIGA-RULEZ.

ZX FORMAT, выпуск N4 1/2

ENLIGHT'96. Отчет о прошедшей встрече, некоторые работы участников в разделах GRAPHIC, MUSIC, DEMO.

ИГРУШКИ. Кое-что из невошедшего в прошлые номера.

ИНТЕРВЬЮ. Интервью с участниками INLIGHT'96: KSA, STALKER, KANO, MAX IWAMOTO и др.

ЖЕЛЕЗО. Этот раздел состоит из двух частей "KAY..." и "PETERS"

ПОЧТОВЫЙ ЯЩИК. Письма читателей, частные объявления, как выбрать домашний компьютер, реклама...

РАЗНОЕ. Программа "Macro-Modem", v.2.1, компьютер "Amiga" и др.

ZX FORMAT N5

ИГРУШКИ. Новелла по игре "48 УТЮГОВ #4", разбор игры "CASTLE MASTER, part one", новелла по игре "ROBIN OF SHERLOCK"

ПРОГРАММИСТАМ.Продолжение курса по Бейсику (расширения Бейсика), "искусственный интеллект" в играх, tr-dos для программистов, программа "MEM HELP", создание эффектов на бордюре, программа определения конфигурации компьютера, адвентюрные игры.

IS-DOS. Начинающим главы из новой книги: справочные данные по основным командам и утилитам базового комплекса is-dos. Программистам: программирование в среде is-dos. Новые программы, подключение винчестера в среде is-dos.

ЖЕЛЕЗО. CPU Z80180 "второе дыхание" Спектрума!?, музыкальный комплекс "GENERAL SOUND", новый модем для Спектрума, схема подключения KEMPSTON MOUSE, доработка мыши и др.

ПРЕМЬЕРА. Программа "CATALOGUE MASTER" текстовый редактор MED, программа "TEXT DESIGNER"

А также: разделы ИНТЕРВЬЮ, ОТДОХНЕМ, ПОЧТА, РАЗНОЕ

ПРИЛОЖЕНИЕ. Программы "MED", "TEXT DESIGNER", "CAT MASTER", "ROBIN OF SHERLOCK, part 1,2,3" и "COVOX TEST"

****

И вот, наконец, совершенно новый журнал из еще одной "синклеровской столицы" Харькова. Пожелаем новому журналу удачи.

UNITED CODERS SOFTWARE совместно с ENIGMA GROUP Представляют новый электронный журнал ZX POWER N 1

Выносим на Ваш суд первый номер нового украинского общественного электронного журнала спектрумовской тематики. Мы надеемся, что наши читатели примут активное участие в его создании,\и совместными усилиями мы сделаем именно то, что нужно.

Ну, а теперь немного подробней о том, что Вы сможете прочитать в первом номере ZX POWER.

В рубрике "РАБОЧИЙ СТОЛ"·

"Прямое предназначение этой утилиты - оцифровка звука с

помощью АЦП для ЦАП (или COVOX)"

"Теневой микроассемблер-отладчик ALASM v.2.8. Краткое описание функциональных возможностей..."

В рубрике "ЛИКБЕЗ":

"В среде начинающих программистов постоянно циркулируют слухи, что кому-то удалось перебросить экран быстрее, чем за одно прерывание. Разберемся, возможно ли это?..."

В рубрике "ЖЕЛЕЗО":

"Речь идет о подключении к Спектруму IBM-овской периферии: мультикарты, внутреннего модема, звуковой карты и т.д. Через мультикарту, в свою очередь, подключаются IBM-овские: винчестер, мышь, принтер, COVOX, внешний модем, 1,44 и 1,2 МБ дисководы, CD ROM, сетевой адаптер и т.д...."

В рубрике "РАЗНОЕ":

"Дожили уже и на Спектруме начали появляться "вирусы"! Во всяком случае, один уже есть, а написали его "кодеры-герои" из группы..."

Но это еще не все! Много другой интересной информации, фантастические и юмористические рассказы Вы прочитаете в этих и других рубриках харьковского журнала ZX POWER. А в приложении сможете найти игру Doom Mania, демоверсию игры "Citadel", которую сейчас дописывает Stalker и еще несколько системных программ.

В дальнейшем мы намерены улучшать оболочку (ИФК: обязательно надо улучшить русский язык), появятся новые рубрики. И "гвоздем" второго номера, скорее всего, станет "ZX ТУРБО ДИЗАССЕМБЛЕР", написанный украинскими программистами. Ждем Ваших писем!

Бугай Алексей
гл. редактор журнала ZX POWER

ВНИМАНИЕ!

В Москве действует официальное представительство фирмы "SCORPION".

У нас Вы можете приобрести, как собранные компьютеры последней модификации, так и комплектующие, для самостоятельной сборки. Мы готовы предоставить Вам любую информацию, которая может понадобиться при сборке Вашего компьютера или при написании программы под него. Здесь Вы можете заменить старые версии ПрофПЗУ, на самые новейшие. Мы предоставляем возможность замены ПЗУ от контроллера IBM клавиатуры и мыши. Возможна сборка компьютеров под заказ и многое другое.

По всем интересующим вопросам Вы можете обращаться по телефону: 412-4705 в вечернее время. Спросить Алексея.

Имеется возможность общения через FIDO: 2:5020/659.51
Alexey Ivanov
Subj: SCORPION
обязательно NetMail'ом

TR-DOS ДЛЯ НАЧИНАЮЩИХ

ИФК: В наступившем году мы продолжаем публикацию книги В.Сироткина "Общие сведения по дисковой системе ТР ДОС" Нужно сказать, что публикуемые далее материалы представляют интерес не только для начинающих, но и для более подготовленных пользователей и программистов. Поэтому название рубрики уже не в полной мере отражает ее содержание.

Продолжение. Начало см. в ZX РЕВЮ 1996 NN 1-2, 4-5, 6, 7-8

ГЛАВА 5

ИНТЕГРАЛЬНАЯ МИКРОСХЕМА КР 1818 ВГ93.

Прежде всего нам надо разобраться с самой главной микросхемой контроллера дисковода ТРДОС (BETA DISK интерфейса): с микросхемой КР 1818 ВГ93. Вся логика работы системы зависит, прежде всего от нее. Эта микросхема представляет собой программный микроавтомат с внутренней логикой работы, который обеспечивает управление дисководом, обмен данными между процессором и дисководом, программирование номеров дорожки, сектора, стороны диска, длины сектора и т.д. и т.п.

В общем, он обеспечивает все то ,что необходимо для нормальной работы с дисководом. Обмен данными между микросхемой ВГ93 и процессором копьютера происходит по 8-ми разрядной шине данных. По этой же шине происходит программирование микросхемы на различные режимы и подача команд контроллеру.

Графическое изображение микросхемы показано ниже на рисунке 9.

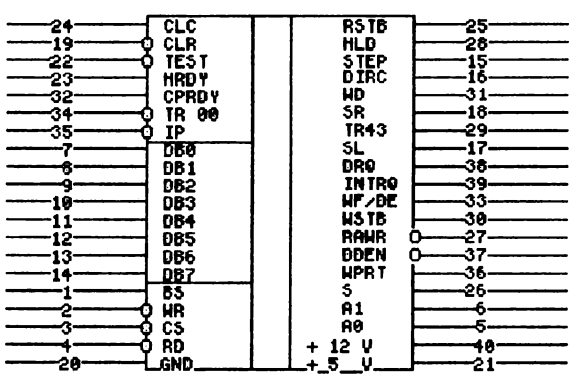


РИС 9.

Назначение выводов микросхемы.

ВЫВОД	МНЕМОН.	НАЗНАЧЕНИЕ ВЫВОДА
1	BS	Вывод микросхемы контроль подложки; в схемах не подключается.
2	W	Вход ЗАПИСЬ.При W=0 - Разрешение записи в выбранный регистр.
3	CS	ВХОД. При 0 - микросхема выбрана
4	R	Вход ЧТЕНИЕ.При = 0 - микросхема выдает на Ш.Д. содержимое выбранного регистра .
5,6	A0,A1	Вход Адреса выбора внутренних регистров ВГ93 для операций чтения/записи. A0 A1 ЧТЕНИЕ ЗАПИСЬ 0 0 РЕГ.СОСТ. РЕГ.КОМАНД 1 0 РЕГ.ДОРОЖ. РЕГ.ДОРОЖ. 0 1 РЕГ.СЕКТОР. РЕГ.СЕКТР. 1 1 РЕГ.ДАННЫХ РЕГ.ДАННЫХ
7-14	D0-D7	Вход/Выход данных или команд
15	STEP	Выход на дисковод импульса перемещения головки на 1 шаг (цилиндр)
16	DIRC	Выход для дисковода. При = 0 - перемещение головки от ЦЕТРА к краю ; При = 1 - от края к центру
17	SL	Выход для внешнего регистра .Сдвиг импульса данных с выхода WD ВЛЕВО.
18	SR	Выход для внешнего регистра.Сдвиг импульса данных с выхода WD ВПРАВО.
19	CLR	Вход СБРОСА. При =0 - микросхема выполняет команду ВОСТАНОВЛЕНИЕ. РЕГ.СЕКТОРА сбрасывается в #01. Выход 39 (INTRQ) = 0.
20	GND	Корпусной вывод.
21	Ucc1	Питание + 5 вольт
22	TEST	Вход- Скорость перемещения головки. При =1 на выходе 'STEP' импульсы перемещения идит с удвоенной частотой.
23	HRDY	Вход ГОТОВНОСТЬ головки. При = 1 указание на готовность к работе.
24	CLC	Тактовая частота
25	RSTB	Выход СТРОБ ЧТЕНИЯ. Устанавливается в 1 после приема 2х байтов нулей при одинарной плотности диска ; или 4х байтов

		нулей (единиц) при удвоенной плотности диска.
26	S	Вход синхронизации. Сигнал вырабатывается из сигналов RAWR.
27	RAWR	Вход ДАННЫХ от дисковода.
28	HLD	Выход ГОТОВНОСТЬ для дисковода. (При =1 - готов)
29	TR43	Выход ПОЛОЖЕНИЕ головки. При = 1 указывает,что головка на цилиндре с номером большим 43. (Сигнал вырабатывается только в процессе ЗАПИСИ или ЧТЕНИЯ)
30	WSTB	Выход. При = 1 включает дисковод на запись.
31	WD	Выход данных для записи на диск.
32	CPRDY	Вход Готовность дисковода для ЗАПИСИ или ЧТЕНИЯ. При = 0 команды ЗАПИСЬ/СЧИТЫВАНИЕ НЕ ВЫПОЛНЯЮТСЯ, а вырабатывается СИГНАЛ INTRQ. (Не влияет на остальные вспомогательные команды).
33	WF / DE	ВХОД / ВЫХОД : ОШИБКА / ДАННЫЕ При WSTB = 1 WF ВХОД; (если WF =0 то, запись любой команды прекращается). При WSTB 0 DE ВЫХОД; (на DE устанавливается 0 при ЧТЕНИИ после загрузки головки и HRDY = 1).
34	TR00	ВХОД - НОЛЬ ДОРОЖКА. При = 0,головки дисковода находятся на 0 дорожке.
35	JP	Вход ИНДЕКСЫЙ ИМПУЛЬС. При = 0 индексный импульс считан и диск начал очередной оборот.
36	WPRT	Вход ЗАПРЕТ ЗАПИСИ . (При = 0)
37	DDEN	Вход ПЛОТНОСТЬ ДИСКА. (При = 1 - плотность ДВОЙНАЯ)
38	DRQ	Выход СТРОБ ДАННЫХ. При DRQ = 1 ; Во время ЧТЕНИЯ указание на то, что РЕГИСТР ДАННЫХ содержит информацию для передачи.

		Во время ЗАПИСИ - ГОТОВНОСТЬ регистра ВГ 93 принять данные от Ш.Д. При DRQ=0 ; При ЧТЕНИИ указание,что данные из регистра ВГ93 считаны процессором. При ЗАПИСИ указание ,что регистр ВГ93 принял данные с Ш.Д.
39	INTRQ	Выход ГОТОВНОСТЬ микросхемы. При = 1 - микросхема готова принять команду ; При = 0 - микросхема ВЫПОЛНЯЕТ команду ИЛИ был считан регистр СОСТОЯНИЯ.
40	Ucc 2	Напряжение питания +12 вольт.

Для нас с вами не потребуется досконального знания функционирования всех входов-выходов микросхемы и режимов ее работы.

С точки зрения программиста, нам потребуется знание логики работы всего лишь нескольких выводов, подключенных непосредственно в адресное пространство портов контроллера ТР ДОС.

Микросхема ВГ93 имеет 5 внутренних регистров, доступных программисту, через которые идет управление работой ВГ93, и еще один внешний регистр, выполненный, как правило, на микросхеме 555ТМ9 (триггере-защелке), для управления дисководом и для опроса готовности.

Все регистры подключены к определенным адресам в адресном пространстве компьютера, как ПОРТЫ ввода/вывода, и, значит, обращатьсяк к ним надлежит командами IN или OUT. Но простой подачей в эти порты каких-либо значений или при попытке считать что-то из этих портов, Вы ничего не добьетесь, т.к. эти порты во время работы SOS Бейсика недоступны!

В ПЗУ ТРДОС есть все необходимые команды обращения к этим портам, и проблема заключается только в том, как выйти на ПЗУ ТРДОС. Вот об этом мы и поговорим чуть позднее...

РАСПРЕДЕЛЕНИЕ ПОРТОВ ТРДОС - ВГ93.

РЕГИСТР ДОРОЖКИ - ПОРТ #3F (ЧТЕНИЕ/ЗАПИСЬ)

Служит для ЗАПИСИ номера дорожки при операциях или для СЧИТЫВАНИЯ номера текущей дорожки.

РЕГИСТР СЕКТОРА - ПОРТ #5F (ЧТЕНИЕ/ЗАПИСЬ)

Служит для ЗАПИСИ номера сектора при операциях или для СЧИТЫВАНИЯ номера текущего сектора.

РЕГИСТР ДАННЫХ - ПОРТ #7F (ЧТЕНИЕ/ЗАПИСЬ)

Служит для обмена данными между диском и компьютером.

РЕГИСТР КОМАНД - ПОРТ #1F (ЗАПИСЬ)

Служит для подачи команд контроллеру.

РЕГИСТР СОСТОЯНИЯ - ПОРТ #1F (СЧИТЫВАНИЕ)

Служит для определения текущего состояния выполнения команд.

РЕГИСТР УПРАВЛЕНИЯ - ПОРТ #FF (СЧИТЫВАНИЕ/ЗАПИСЬ)

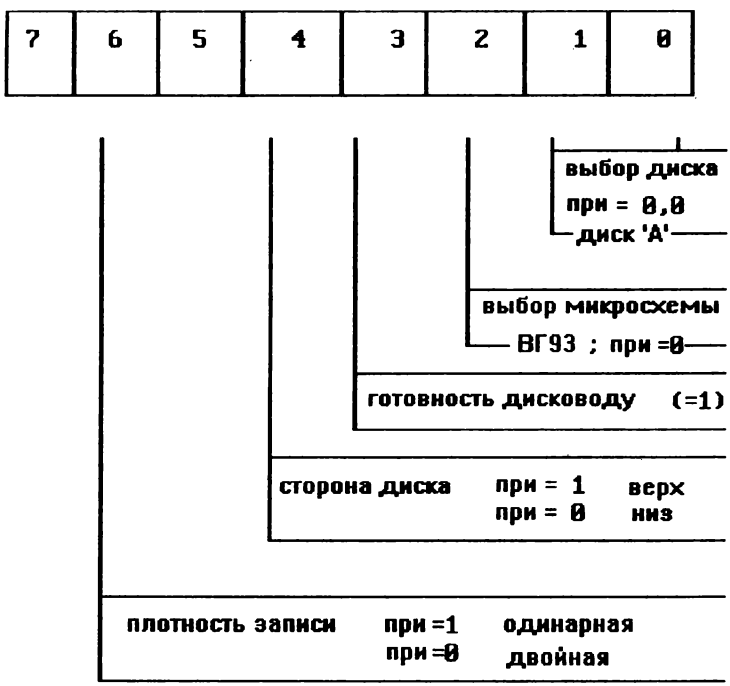
Служит для управления дисководом и для считывания готовности микросхемы ВГ93 при работе или при обмене данными. Этот регистр выполнен на микросхеме ТМ9 триггере защелке. При подаче в порт байта, он сохраняется - защелкивается.

Рассмотрим подробнее РЕГИСТР УПРАВЛЕНИЯ (#FF).

При ЗАПИСИ в этот регистр можно выбрать номер дисковода, сбросить микросхему ВГ93, выбрать сторону диска, дать готовность дисководу и выбрать плотность записи.

Ниже на рисунке представлены биты этого регистра и за что они отвечают при ЗАПИСИ в него.

Режим записи в регистр управления (#FF).



БИТЫ 0 и 1 с выхода триггера ТМ9 напрямую связаны со входами дисковода "DISK A,B,C"

БИТ 2 это вход CLR микросхемы ВГ93

БИТ 3 — это вход HRDY микросхемы ВГ93 и одновременно разрешение прохождения сигнала IP от дисковода к микросхеме ВГ93

БИТ 4 — напрямую связан со входом дисковода "SIDE"

BIT 6 — это вход DDEN микросхемы ВГ93

В Режиме ЧТЕНИЕ регистра #FF использованы всего 2 бита, (смотри ниже).

Режим чтения регистра управления (#FF).

БИТ 6 — это выход DRQ микросхемы ВГ93

БИТ 7 — это выход INTRQ микросхемы ВГ93

Микросхема ВГ93 обеспечивает прием и выполнение 11 команд, поданных в Регистр КОМАНД. Каждая команда,из одиннадцати, имеет свой КОД-МОДИФИКАТОР, который изменяет некоторые условия выполнения команды.

Условно команды подразделяются на четыре типа:

1 - команды перемещения головок дисковода и поиска: эти команды загружают (прижимают) головки.

2 - команды чтения/записи секторов диска.

3 - команды чтения/записи дорожки и адресной метки.

4 - команды принудительного прерывания операций.

Результаты выполнения или невыполнения команд отображаются в Регистре СОСТОЯНИЯ, каждый бит которого несет информацию о ходе выполнения команды. Разберем подробнее каждую команду.

Команды ПЕРВОГО ТИПА.

Данные команды обеспечивают загрузку головок, т.е. прижатие их к поверхности диска.

Команда ВОСТАНОВЛЕНИЕ.

Переход головок дисковода на НУЛЕВОЙ ТРЕК (ЦИЛИНДР). Если на входе TR00 микросхемы не появится подтверждение о выходе головок на 0 ,то через 256 импульсов действие команды прекращается.

КОД команды.

7	6	5	4	3	2	1	0
0	0	0	0	G	V	F1	F2
				МОДИФИКАТОР			

При:

G=0 головка поднята.
G=1 головка опущена (в работе).
V=0 местонахождение головки не проверяется.
V=1 читается и проверяется номер цилиндра под головкой.
F1,F2 коды времени перемещения головок (см.таблицу).

КОМАНДА ПОИСК ЦИЛИНДРА.

КОД команды.

7	6	5	4	3	2	1	0
0	0	0	1	G	V	F1	F2
				МОДИФИКАТОР			

При:

G=0 головка поднята.
G=1 головка опущена (в работе).
V=0 местонахождение головки не проверяется.
V=1 читается и проверяется номер цилиндра под головкой.
F1,F2 коды времени перемещения головок (см.таблицу).

Перед командой Регистр ДОРОЖКИ должен содержать номер ТЕКУЩЕГО ЦИЛИНДРА, а Регистр ДАННЫХ номер ТРЕБУЕМОГО. Поиск продолжается до тех пор, пока регистр ДОРОЖКИ не сравняется с Регистром ДАННЫХ.

Поиск выполняется при модификаторе V=1 !!

КОМАНДА ШАГ.

Перемещение головки на ОДИН шаг,(на 1 цилиндр). Направление перемещения остается прежним.

КОД команды.

7	6	5	4	3	2	1	0
0	0	1	J	G	V	F1	F2
			МОДИФИКАТОР				

При:

G=0 головка поднята.
G=1 головка опущена (в работе).
V=0 местонахождение головки не проверяется.
V=1 читается и проверяется номер цилиндра под головкой
F1,F2 коды времени перемещения головок (см.таблицу).
J=0 содержимое Регистра ДОРОЖКИ не меняется.
J=1 содержимое Р.ДОРОЖ. меняется на 1 (+/-).

КОМАНДА ШАГ ВПЕРЕД (к центру диска)

Перемещение головки на ОДИН шаг ВПЕРЕД (на 1 цилиндр).

КОД команды.

7	6	5	4	3	2	1	0
0	1	0	J	G	V	F1	F2
			МОДИФИКАТОР				

При:

G=0 головка поднята.
G=1 головка опущена (в работе).
V=0 местонахождение головки не проверяется.
V=1 читается и проверяется номер цилиндра под головкой.
F1,F2 коды времени перемещения головок (см.таблицу).
J=0 содержимое Регистра ДОРОЖКИ не меняется.
J=1 содержимое Р.ДОРОЖ. увеличивается на 1.

КОМАНДА ШАГ НАЗАД (к краю диска)

Перемещение головки на ОДИН шаг НАЗАД (на 1 цилиндр).

КОД команды.

7	6	5	4	3	2	1	0
0	1	1	J	G	V	F1	F2
			МОДИФИКАТОР				

При:

G=0 головка поднята.
G=1 головка опущена (в работе).
V=0 местонахождение головки не проверяется.
V=1 читается и проверяется номер цилиндра под головкой.

F1,F2 коды времени перемещения головок (см.таблицу).
J=0 содержимое Регистра ДОРОЖКИ не меняется.
J=1 содержимое Р.ДОРОЖ. уменьшается на 1.

КОМАНДЫ ВТОРОГО ТИПА.

КОМАНДА ЧТЕНИЕ СЕКТОРА ,(секторов).

Команда выполняется после того, как микросхемой ВГ93 идентифицирована область ИАМ сектора (индексная адресная метка) и подсчитана контрольная сумма (все это происходит автоматически после подачи команды и не зависит от программиста).

Перед командой необходимо:

1. Установить головку на нужный ЦИЛИНДР
2. в Регистр СЕКТОРА занести номер требуемого СЕКТОРА.

КОД команды.

7	6	5	4	3	2	1	0
1	0	0	M	S	E	C	0
			МОДИФИКАТОР				

При:

M=0 чтение ОДНОГО сектора, содержимое Регистра СЕКТОРА увеличивается на 1.

M=1 после чтения одного сектора, содержимое Регистра СЕКТОРА увеличивается на 1, и начинается чтение следующего, и так до последнего сектора на ДОРОЖКЕ.

S=0 сторона диска НИЗ.
S=1 сторона диска ВЕРХ.
E=0 нет задержки для установки головки дисковода.
E=1 задержка в 15 мс. для установки головки после сигнала HLD микросхемы (готовность).
C=0 сторона диска не проверяется.
C=1 проверка стороны диска в процессе идентификации.

Если в процессе идентификации микросхема не находит в ИАМ-области НОМЕР нужного СЕКТОРА, или же вообще не находит ИАМ, то в Регистре СОСТОЯНИЯ записывается признак "МАССИВ ЧТЕНИЯ НЕ НАЙДЕН"

Если по окончании считывания сектора не совпали Контрольные суммы, то в Регистр СОСТОЯНИЯ записывается признак ОШИБКА, и выполнение команды ОБРЫВАЕТСЯ.

При чтении массива данных из поля сектора в Регистр ДАННЫХ микросхемы, КАЖДЫЙ байт сопровождается сигналом СТРОБ ДАННЫХ (вывод DRQ микросхемы; 6-й бит порта #FF, контроллера ТРДОС).

Если текущий байт не был считан из Регистра ДАННЫХ по сигналу СТРОБ ДАННЫХ до прихода следующего байта, то в регистр

ДАННЫХ записывается следующий байт, а в Регистре СОСТОЯНИЯ записывается признак ПОТЕРЯ ДАННЫХ.

КОМАНДА ЗАПИСЬ СЕКТОРА ,(секторов).

Команда выполняется после того, как микросхемой ВГ93 идентифицирована область ИАМ сектора, (индексная адресная метка) и подсчитана контрольная сумма (все это происходит автоматически после подачи команды и не зависит от программиста).

Перед командой необходимо:

1. Установить головку на нужный ЦИЛИНДР.
2. в Регистр СЕКТОРА занести номер требуемого СЕКТОРА.

КОД команды.

7	6	5	4	3	2	1	0
1	0	1	M	S	E	C	A
			МОДИФИКАТОР				

При:

M=0 запись ОДНОГО сектора, содержимое Регистра СЕКТОРА увеличивается на 1.

M=1 после записи одного сектора,содержимое Регистра СЕКТОРА увеличивается на 1 и начинается запись следующего, и так до последнего сектора на ДОРОЖКЕ.

S=0 сторона диска НИЗ.

S=1 сторона диска ВЕРХ.

E=0 нет задержки для установки головки дисковода.

E=1 задержка в 15 мс. для установки головки после сигнала HLD микросхемы (готовность).

C=0 сторона диска не проверяется.

C=1 проверка стороны диска в процессе идентификации.

A=0 запись в область ИАМ признака "сохранения данных (#FB) (без стирания)"

A=1 запись в ИАМ признака "стирать данные разрешено (#F8)"

Если в процессе идентификации микросхема не находит в ИАМ области НОМЕР нужного СЕКТОРА, или же вообще не находит ИАМ, то в Регистре СОСТОЯНИЯ записывается признак МАССИВ ЧТЕНИЯ НЕ НАЙДЕН.

После того, как ИАМ найдена, следует запрос данных сигнал DRQ. Микросхемой записываются (автоматически) нужные пробелы в секторе (12 нулей) перед областью данных, затем записывается "Адресная метка данных" и только затем сами данные.

Перед каждым запросом следующего байта данных микросхемой выставляется Строб ДАННЫХ, (выход DRQ).

Если на запрос DRQ в регистр данных не будет помещен следующий байт для записи, то в регистре СОСТОЯНИЯ выставится признак ПОТЕРЯ ДАННЫХ, а на диск запишется байт 00 !

После записи всего массива данных микросхемой автоматически записывается контрольная сумма в 2 байта и байт #FF.

КОМАНДЫ ТРЕТЬЕГО ТИПА.

КОМАНДА ЧТЕНИЯ АДРЕСНОЙ МЕТКИ

Команда выполняется тогда, когда головка дисковода находится в рабочем положении, т.е. прижата к диску (сигнал HLD=1). Считываются ШЕСТЬ байтов ИНДЕКСНОЙ области диска, включая Контрольную сумму (Номер цилиндра, сторона диска, тип сектора, номер сектора, 2 байта контр.суммы).

Все байты сопровождаются СТРОБОМ чтения БАЙТА (вывод DRQ или 6-й бит порта #FF). При выполнении команды содержимое Регистра ДОРОЖКИ пересылается в Регистр СЕКТОРА и запоминается. По окончании выполнения команды генерируется сигнал INTRQ (7 бит порта #FF).

КОД команды.

7	6	5	4	3	2	1	0
1	1	0	0	0	E	0	0
					МОДИФИКАТОР		

При:

E=0 - нет задержки для установки головки дисковода.

E=1 - задержка в 15 мс. для установки головки после сигнала HLD микросхемы (готовность).

КОМАНДА: ЧТЕНИЕ ДОРОЖКИ ЦЕЛИКОМ.

По этой команде считывается ВСЯ информация с дорожки, вместе с индексным массивом,пробелами и областью данных. Во время чтения НЕ выдается СТРОБ чтения БАЙТА и не проверяются контрольные суммы. Применяется, в основном, эта команда в диагностических целях, например, для юстировки головок дисковода или в программах типа "ТРЕК КОПИР"

КОД команды.

7	6	5	4	3	2	1	0
1	1	1	0	0	E	0	0
					МОДИФИКАТОР		

При:

E=0 - нет задержки для установки головки дисковода.

E=1 - задержка в 15 мс. для установки головки после сигнала HLD микросхемы (готовность).

КОМАНДА: ЗАПИСЬ ДОРОЖКИ ЦЕЛИКОМ.

Эта команда применяется для ФОРМАТИРОВАНИЯ диска. Запись информации происходит за ОДИН РАЗ на целую дорожку. При форматировании вся необходимая информация должна находиться в ОЗУ/ПЗУ и содержать все пробелы и индексные метки.

КОД команды.

7	6	5	4	3	2	1	0
1	1	1	1	0	E	0	0
					МОДИФИКАТОР		

При:

E=0 - нет задержки для установки головки дисковода.

E=1 - задержка в 15 мс. для установки головки после сигнала HLD микросхемы (готовность).

При выполнении команды ЛЮБАЯ информация запишется на дорожку, кроме некоторых байтов. А именно: появление байтов #F5...#FE интерпретируются как начало СЛУЖЕБНЫХ МЕТОК.

БАЙТ #F5	запись кода #A1, инициализация подсчета контрольной суммы.
БАЙТ #F6	запись кода C2 - индексная метка.
БАЙТ #F7	запись контрольной суммы.
БАЙТ #FB	адресная метка данных.
БАЙТ #FE	адресная метка ИНДЕКСНЫХ данных.

Все это справедливо для записи на диск ДВОЙНОЙ плотности (с модифицированной частотной мадуляцией). Для одинарной плотности байты меток будут немножко другими, но такие дисководы стали уже редкостью и рассматривать одинарную плотность записи мы не будем.

В процессе форматирования эти байты меток не должны записываться в область пробелов и в область для данных! Подробнее массив данных для форматирования дисков мы рассмотрим в отдельной главе.

КОМАНДЫ ЧЕТВЕРТОГО ТИПА.

КОМАНДА "ПРИНУДИТЕЛЬНОЕ ПРЕРЫВАНИЕ"

Команда предназначена для завершения выполняемой операции и может быть записана в Регистр КОМАНД в любой момент времени.

КОД команды.

7	6	5	4	3	2	1	0
1	1	0	1	m3	m2	m1	m0
				МОДИФИКАТОР			

При:

- m3=m2=m1=m0 = 0 команда прекращается, но сигнал JNTRQ не вырабатывается (порт #FF, 7 байт).
- m0 = 1 прерывание происходит при переходе от 0 на 1 на входе CPRDY.
- m1 = 1 прерывание происходит при переходе от 1 на 0 на входе CPRDY.
- m2 = 1 прерывание по первому индексному сигналу (началу нового оборота диска).
- m3 = 1 немедленное прерывание и установка на выходе JNTRQ сигнала ГОТОВНОСТЬ (бит 7, порта #FF).

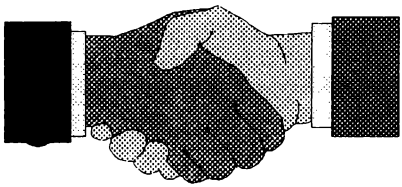
Время установки времени перемещения головки в командах ПЕРВОГО типа зависит, прежде всего, от уровня на входе 'TEST' микросхемы ВГ93, тактовых импульсов на входе CLK, а также от состояния модификаторов команд #F0,#F1.

TEST	F1	F0	Время перемещения на 1 шаг	
			при CLK=1 МГЦ	при CLK=2
1	0	0	6	3
1	0	1	12	6
1	1	0	20	10
1	1	1	30	15
0	-	-	400	200

Любая команда в процессе своего исполнения обновляет регистр СОСТОЯНИЯ микросхемы (порт #1F при чтении). С помощью этого регистра как раз и ведется "диалог" между компьютером и контроллером о выполнении или невыполнении команд.

Ниже дана таблица значения битов этого регистра при выполнении различных команд. А следом за ней помещена таблица, в которой показаны все команды ВГ93.

ЧИТАТЕЛЬ – ЧИТАТЕЛЮ



П.Федин, г.Москва

О ЖЕЛЕЗЕ, СОФТЕ И ПРОЧЕМ...

Слава богу, наступила пора, когда все уже поняли, что с неразберихой в железе и программах, оставшейся с давних времен, пора заканчивать. И начались разговоры о стандартизации. Я читаю об этом в разных изданиях, но, увы, вижу, что иногда авторы статей на эту тему упускают некоторые вещи. Я считаю, что до принятия стандарта нужно собрать информацию обо всех существующих в данное время моделях, чтобы постараться учесть нюансы каждой из них.

Я вношу свой вклад в это дело и предоставляю информацию о компьютере PROFI.

Хочу предложить вам выкладку из результатов проведенных мной исследований. Первым рассмотренным вопросом будет адресация дополнительных устройств этого компьютера. Итак, здесь мы имеем следующую картину:

Порты обычных устройств. Доступны всегда.

#XX1F - порт А микросхемы 580ВВ55, обычно - Kempston джойстик.
#XX3F - порт В 580ВВ55. Обычно - данные на принтер.
#XX5F - порт С 580ВВ55. Обычно - управление принтером (строб и BUSY).
#XX7F - регистр управления 580ВВ55.
#XXFE - бордюр, магнитофон, клавиатура и т.д.
#7FFD - первый регистр конфигурации. Все как у Speccy-128.
#DFFD второй регистр конфигурации. Структура рассмотрена ниже.
#BFFD - регистр данных AY.
#FFFD - регистр адреса AY.

Порты контроллера IDE винчестера. Эти и нижеследующие порты доступны только при снятой защелке (бит 5 порта #DFFD равен 1).

#06AB - системный регистр контроллера Только запись.
#07EB - регистр команд винчестера. Только запись.
#07CB - регистр состояния винчестера. Только чтение.
#06EB - регистр номера головки на запись.
#06CB - регистр номера головки на чтение.
#05EB - старший байт номера дорожки на запись.
#05CB - старший байт номера дорожки на чтение.
#04EB - младший байт номера головки на запись.
#04CB - младший байт номера головки на чтение.
#03EB - регистр номера сектора на запись.
#03CB - регистр номера головки на чтение.
#02EB - счетчик секторов на запись.

#02CB - счетчик секторов на чтение.

#01EB регистр стартового цилиндра предкомпенсации. Только запись.

#01CB - регистр ошибок. Только чтение.

Порты компорта.

#XXB3 регистр управления компортом и контроллером прерываний.

#XXD3 - регистр данных 580ВВ51.

#XXF3 - регистр управления/состояния 580ВВ51.

#XX8F регистр чтения и загрузки канала 0 580ВИ53. Задает скорость работы последовательного порта.

#XXAF - регистр чтения и загрузка канала 1 580ВИ53. Используется для генерации -12 В для питания 580ВВ51.

#XXCF регистр чтения и загрузки канала 2 580ВИ53. Не используется.

#XXEF - регистр управления 580ВИ53.

Порты CMOS-часов.

#XXDF - регистр данных 580ВИ51.

#XXFF регистр адреса 580ВИ51. Только запись, поэтому на порт атрибутов не влияет. Правда, у меня почему-то этот порт не работает, хотя говорят, что он все-таки есть.

Я специально обьединил порты в группы по устройствам. В этих группах ясно прослеживается принцип построения адресов:

XXXXXXXX	0AAXXXXX	- 580ВВ55.
XXXXXXXX	XXXXXXX0	- порт #FE.
0XXXXXXX	XXXXXX0X	- порт #7FFD.
XX0XXXXX	XXXXXX0X	- порт #DFFD.
X0XXXXXX	XXXXXX0X	- порт #BFFD
111XXXXX	XXXXXX0X	- порт #FFFD.
XXXXXAAA	1CI01011	- контроллер винчестера.
XXXXXXXX	1CA10011	- компорт (580ВВ51).
XXXXXXXX	1AA01111	- 580ВИ53.
XXXXXXXX	1XA11111	- CMOS-часы.

A - биты локальной адресации (номер регистра в устройстве)

C - если этот бит сброшен, то выбирается дополнительный регистр, не входящий в микросхему, а биты адресации игнорируются.

I - в контроллере винчестера указывает направление передачи данных: 0 - ввод, 1 - вывод.

Мы видим, что дешифрация стандартных портов построена стандартно, а за дополнительные устройства отвечают биты 0 - 4 и 7, причем каждому устройству соответствует строго определенная их комбинация. Таким образом, для подключения дополнительных устройств можно использовать оставшиеся комбинации этих битов плюс все возможные комбинации битов локальной адресации. В этом случае возможно использование следующих адресов:

XXXXXXXX	10000011	- #83.
XXXXXXXX	10100011	- #A3.
XXXXXXXX	11000011	- #C3.
XXXXXXXX	11100011	- #E3.
XXXXXXXX	10000111	- #87. 1
XXXXXXXX	10100111	- #A7. 12
XXXXXXXX	11000111	- #C7. 1
XXXXXXXX	11100111	- #E7. 12
XXXXXXXX	10010111	- #97. 1
XXXXXXXX	10110111	- #B7. 123
XXXXXXXX	11010111	- #D7. 1
XXXXXXXX	11110111	- #F7. 123
XXXXXXXX	10011011	- #9B.
XXXXXXXX	10111011	- #BB.
XXXXXXXX	11011011	- #DB.
XXXXXXXX	11111011	- #FB.

Вот полный список портов, свободных для использования любыми устройствами. Теперь, для совместимости с остальными моделями компьютера, выкинем те адреса, которые конфликтуют с устройствами, не имеющимися в Profi, но присутствующими в других машинах. Из нерассмотренных здесь интерфейсов известны:

XXXXXXXX	SXXXX0XX	- ZX-LPRINT.
XXXXXXXX	XX0XXXXX	- Kempston mouse.
XXXXXXXX	XXX0XXXX	- модем.

Отбросив из вышеприведенного списка адреса, конфликтующие с этими, останутся #B7 и #F7, дешифрацию которых необходимо производить по следующей схеме:

XXXXXXXX	X0X10111	- #B7.
XXXXXXXX	X1X10111	- #F7.

При этом мы не рассматриваем старший байт адреса, а если применить и его, то получим ну очень много устройств. Только НЕ НАДО стремиться к упрощению адресации. Здесь я упростил дешифрацию до предела, опустив 2 бита, но ЛЮБОЕ дальнейшее упрощение приведет к конфликтам. А вот и пример того, до чего может довести упрощение адресации. Если бы стандартные порты Speccy-128 дешифрировались полностью, то и адресов было бы намного больше. Вспомните поговорку скупой платит дважды. Так что не гоняйтесь за дешевизной пара корпусов никого не устроит.

Рассматривая приведенную карту портов Profi, Вы не могли не заметить один печальный факт ни Kempston mouse, ни модем НЕ ВПИСЫВАЮТСЯ в нее. Вот Вам печальный пример того, что разработчики не всегда учитывают особенности других существующих устройств. Вот и С. Шмелев в ZX РЕВЮ 96/9 предлагал перепахать интерфейс ВВ55. Конечно, на Кванте ничего не стоит это сделать, но вот на Профи... Нет, я не говорю, что все авторы подобных разработок неправы, а

наоборот, неправы были разработчики Profi. Они стали лепить новые порты, не ознакомившись с тем, что уже есть. И теперь владельцы этих машин лишены возможности использовать Kempston Mouse, а если они пользуются винчестером и компортом, то во время работы с ним они вынуждены отключать еще и модем. Вот к чему мы пришли.

Поэтому заявляю всем разработчикам аппаратуры ***ПРЕКРАТИТЕ ВЫДУМЫВАТЬ НОВЫЕ СТАНДАРТЫ! ХВАТИТ!!!*** Но и впадать в другую крайность призывать всех повыкидывать машины, подобные моей, и покупать Scorpion, тоже не стоит, ибо этих несчастных мутантов уже слишком много, и в одночасье сменить парк машин невозможно. Уже есть какие-то стандарты, пусть не совсем сложившиеся, но все-таки с ними надо считаться. Иначе это не прекратится. Мы должны выбрать какой-то один стандарт и следовать ему, но так, чтобы его можно было внедрять на ВСЕХ машинах. Я очень хотел бы, чтобы с моей статьей ознакомились в редакции журнала ZX-FORMAT, пропагандирующей Kempston Mouse и призывающей вынести на свалку все остальное, а также уважаемые члены студии X-TRADE, готовящие к выпуску свой General Sound. Как бы с ним не вышло такой же истории, как и с Kempston Mouse на Profi...

И напоследок скажу немного на тему новой операционной системы. Эта тема напрямую связана с той, которую я рассмотрел выше. Ведь для обеспечения функционирования нового оборудования нужна новая ОС. Только такая система должна быть гибкой и наращиваемой, чтобы обеспечить ее подстройку под любое новое оборудование. В ZX РЕВЮ 96/7-8 прозвучали мысли о том, что надо пересаживаться на Profi под CP/M. Повторяю ВЫ НИКОГО НИКОГДА НЕ ЗАСТАВИТЕ ВЫКИНУТЬ СВОЙ СКОРПИОН И КУПИТЬ PROFI (как, впрочем, и наоборот примечание для редакции ZX-FORMAT).

К тому же базовой моделью остается машина со 128К, следовательно ОС ОБЯЗАНА ФУНКЦИОНИРОВАТЬ С ЭТИМ ОБЬЕМОМ ПАМЯТИ. Дальше загружаемая ОС подвержена атаке вирусов, а ПЗУ'шная нет. Тем более, что никому не нужен Бейсик-128.

Пару слов по поводу четвертой страницы ПЗУ Не используется она далеко не везде. Например, на Scorpione в ней находится теневой монитор, на GRMe системное меню, на Profi загрузчик CP/M и всевозможные тесты. К тому же на многих машинах потребуются аппаратные доработки для обеспечения программного доступа к ней, а это увеличивает стоимость ОС. Вот на чем надо экономить, а не на ЛЛ'ках в дешифраторах портов. К тому же CP/M жутко тормозная. "Но для этого есть TURBO!"- скажете Вы. А на IBM PC тоже было TURBO, потом его сняли, потому что его перестали выключать, а потом и этого стало мало... Нет, товарищи, так не годится... К тому же не у всех оно есть. Отсю-

да еще одно условие ОС ДОЛЖНА РАБОТАТЬ С НОРМАЛЬНОЙ СКОРОСТЬЮ НА 3.5 МГЦ. И современное ПО как раз в TR DOS'е намного превосходит CP/M'овское не по количеству ненужных наворотов, а по удобству работы. В общем, на эту тему мы еще поспорим, а сейчас придется закругляться, а то если я начну разбирать по косточкам этот вопрос, то статья вдвое длиннее станет.

****

А.Шуравин, г.Воткинск

ВОССТАНОВЛЕНИЕ УТЕРЯННОЙ ИНФОРМАЦИИ.

Хотя и не часто, но случается, что ценные данные, доставшиеся пользователям ЭВМ с большим трудом, утеряны. Обидно! Но все-же, так ли уж безвыходно положение? Рассмотрим несколько случаев повреждения информации.

1. Случайное стирание файла.

Это самое безобидное явление, если Вы, конечно, не сделали команду MOVE. Дело в том, что в отличие от системы MS-DOS, в TR-DOS файлы не фрагментированы, и поэтому стоят строго друг за другом. Так что, при удалении файла на диске останется незанятое место, которое заполнено содержимым стертого файла. Файл никуда не девался. Однако, по команде CAT или LIST Вы его не увидите.

Что же произошло? Сотрем файл, например, t13++ <C>. Просмотрите в каком-нибудь диск-докторе содержание первых восьми секторов системного (нулевого) трека. Вы сможете увидать примерно такую картину:

54	4C	57	2E	52	42	54	20	TLW.RBT
42	AE	02	92	02	03	0E	03	B
01	6C	33	2B	2B	20	20	20	13++
43	86	BB	7A	44	45	0F	03	C zDE

Как видим, первый символ файла t13++ оказался замененный кодом 01 (если файл последний, то символ заменяется кодом 00 - конец каталога). То есть, чтобы восстановить его, нужно просто исправить этот код. И байт номер 244 восьмого логического сектора системного трека уменьшить на единицу (здесь хранится информация о количестве удаленных файлов).

В случае, когда файл оказался последним, сложнее. Тогда придется исправить байты 225, 226 - номер первого свободного сектора дорожки; 228 - количество файлов на диске, и 229, 230 - количество свободных секторов. Для этого необходимо знать структуру заголовка файла.

Структура заголовка файла

БАЙТЫ	НАЗНАЧЕНИЕ
0...7	ИМЯ ФАЙЛА (8 СИМВОЛОВ)
8	ТИП ФАЙЛА (B, C, D, #)
9 - 10	АДРЕС НАЧАЛА ФАЙЛА

11 - 12	ДЛИНА ФАЙЛА
13	КОЛИЧЕСТВО СЕКТОРОВ
14 - 15	СЕКТОР / ТРЕК НАЧАЛА ФАЙЛА

Есть и более быстрый способ восстановить стертый файл - воспользоваться программой CONVER COMMANDER, которая позволяет видеть удаленные файлы в файловой панели. Для их восстановления просто переименуйте файл, подведя к нему курсор и нажав вместе CAPS SHIFT и SYMBOL SHIFT, выбрать пункт меню "переименовать" При включенном режиме "+del файлы" стертые файлы начинаются с красного знака вопроса.

2. Порча системной дорожки.

В этом случае можно поступить двумя способами. Или найти где на диске размещены данные файлы и скопировать их, или попытаться восстановить системный трек. Последнее, к сожалению, не всегда возможно.

Несколько советов тем, кто пользуется первым способом.

1. Начните с текстовых файлов. Их найти гораздо легче.
2. Файлы формата GENS обычно начинаются с ORG.
3. Если Вы ищите BASIC - файлы, то обращайте внимание на номера строк. которые имеют такой вид:
   - 1 байт - номер строки/256
   - 2 байт - остаток от деления на 256.
4. Скопировать файл Вам поможет следующая программа:

```
LD      DE,#NNNN;     NNNN - НОМЕР ТРЕКА (СТАРШИЙ
                      РАЗРЯД), НОМЕР СЕКТОРА НАЧАЛА
                      ФАЙЛА
LD      HL,50000
LD      BC,#NN05;     ГДЕ NN - КОЛИЧЕСТВО СЕКТОРОВ
                      ФАЙЛА
LD      A,255
CALL    15635
RET
```

После ее запуска Вам следует войти в TR DOS и дать следующую команду: SAVE "FILENAME"CODE START, LENGHT,

где FILENAME - имя вновь созданного файла,

START - адрес начала файла (в данном случае 50000),

LENGHT - длина файла.

Если Вам необходимо восстановить файл BASICа, данных или ВВОДА/ВЫВОДА, то поменяйте расширение при помощи диск-доктора. Если Вам не понятен смысл этих закорючек, то наберите в BASICе следующие строки:

```
10 DATA 17,N1,N2,33.80,195,1,5,N3,62,255,205,19,61,201
20 FOR I=0 TO 14: READ A: POKE (50000 + I),A:NEXT I
30 RANDOMIZE USR 50000
```

и запустите программу командой RUN. Вместо N1 следует подставить номер сектора, N2 - номер трека, N3 - количество секторов в файле.

Восстановить саму системную дорожку зачастую бывает сложнее. Если имеет место порча ДЕВЯТОГО физического сектора, то при его восстановлении можно воспользоваться таблицей:

Структура 9-го физического сектора системного трека.

Номер байта	Назначение
225	номер первого свободного сектора
226	номер первой свободной дорожки
227	тип диска #16 - двусторонний, #18 - односторонний
228	количество файлов на диске
229,230	количество свободных секторов
231	количество секторов в дорожке, всегда 16
232,233	#00
234...242	#20
243	#00
244	количество стертых файлов
245...252	имя диска (8 символов)

Впрочем, системная дорожка занимает всего 2 килобайта, так что не будет расточительстом сохранить ее где-нибудь в свбодном месте, например, на 160 физическом треке. Это можно сделать, набрав в каком-нибудь ассемблере такую программу:

```
LD      HL,50000
LD      DE,0
LD      BC,#905
LD      A,255
CALL    15635
LD      HL,50000
LD      DE,#9F07
LD      BC,#906
XOR     A
CALL    15635
RET
```

Восстановить сохраненную системную дорожку можно при помощи такой программы:

```
LD      HL,50000
LD      DE,#907
LD      BC,#906
XOR     A
CALL    15635
LD      HL,50000
LD      DE,0
LD      BC,#905
LD      A,255
CALL    15635
RET
```

На бейсике они будут выглядеть вот так:

```
10 DATA 33, 80, 195, 17, 0, 0, 1,
5, 9, 62, 255, 205, 19, 61, 33, 80,
195, 17
20 DATA 7,159,1,6,9,175,201
20 FOR I=0 TO 24: READ A:
POKE (50000+I),A: NEXT I
30 RANDOMIZE USR 15619:
REM: SAVE "MIRROR" CODE
50000,100
```

```
10 DATA 33, 80, 195, 17, 7, 159,
1, 6, 9, 175, 205, 19, 61, 33, 80, 195,
17
20 DATA 0,0,1,5,9,62,255,201
20 FOR I=0 TO 24: READ A:
POKE (50000+I),A: NEXT I
30 RANDOMIZE USR 50000
```

Чтобы автоматически сохранять системную дорожку, создайте файл объектного кода MIRROR, и переименовав фойл boot в BOOT, создайте новый файл автозагрузки:

```
10 RANDOMIZE USR 15619:
REM: LOAD "MIRROR" CODE
50000
20 RANDOMIZE USR 50000
30 RANDOMIZE USR 15619:
REM: RUN "BOOT" набрав в TR
DOS команду SAVE "boot"
```

3. Порча отдельного сектора.

Здесь Вас может выручить разве что специальная программа восстановления размагниченных секторов (принцип: считать, отформатировать, записать). Такую возможность имеют утилиты типа FUT, ADS, DCU.

Это, к сожалению, не всегда помогает, поэтому всегда желательно делать хотя бы раз в неделю копию файлов, над которыми Вы работаете.

Иногда такое копирование лучше делать несколько раз в день: никто не гарантирует, что в процессе работы не может произойти сбой, в результате которого испортится диск.

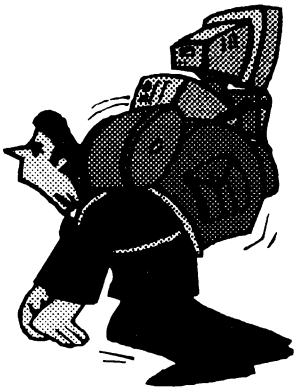
Бывают случаи, когда не читается сектор системного трека. Тогда, если Вы воспользовались рекомендациями n2, сохранив системную дорожку на 160 физическом треке, просто скопируйте весь диск посекторным копировщиком типа D-D FULL COPY, игнорируя копирование запорченного сектора, а затем восстановите системный трек.

И, самое главное, не ленитесь делать резервные копии!

Это сэкономит Ваши время и нервы.

****

КОМПЬЮТЕРЫ, КОТОРЫЕ МЫ ВЫБИРАЕМ



(С) Соков Андрей (KONEX). SCORPION CLUB. Москва,1996

ДОРАБОТКА КОМПЬЮТЕРА "ATM-TURBO"

Хочу предложить вашему вниманию мою доработку,после которой на "ATM-TURBO" будут работать все программы, использующие команду OUT (#FD),A.

В ZX РЕВЮ 95/3 была помещена статья о повышении совместимости компьютера "ATM-TURBO 2" К сожалению, те аппаратные изменения, которые там описаны, для "ATM-TURBO" не годятся, несмотря на схожесть схем этих двух компьютеров. Поэтому мне пришлось разработать свою схему доработки именно для "ATM-TURBO"

Но прежде,чем описывать ее, приведу доработку из описания "ATM-TURBO"

1.Параллельно триггеру D50 напаяйте еще один триггер TM9 и пропустите через него сигналы SOUND (D50/12) и TAPE-OUT (D50/10), пред-варительно отрезав их от D50. Тактовый вход нового триггера подсоедините к сигналу BRDWR.

2.Отрежте сигналы BRDWR от D50/9 и M1 от D17/6 и, напаяв сверху D17 микросхему 555КП12, соедините провода как показано на рис.1

Итак, теперь количество работаю-

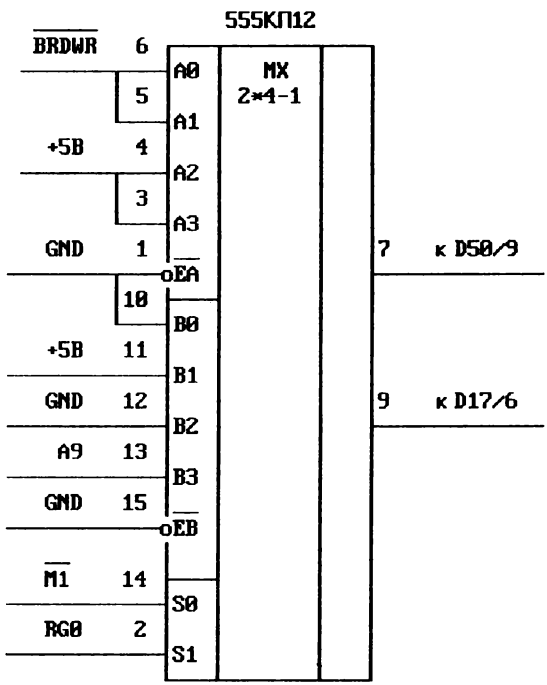


Рис.1

щих программ повысилось, в основном, за счет тех, над которыми жестоко надругался VASILYEV ANTON, "прославившийся" тем, что портил спрайты и лишал игры некоторых подзагружаемых этапов (INDIANA JONES AND THE LAST CRUSADE, THE UNTOUCHABLES, THE HUNT FOR RED OCTOBER, ASTRO MARINE CORPS.1&2, P-47 и т.д. и т.п.).

Но теперь в программе "Honey Commander" Вы узнаете, что у Вашего компьютера не 512K памяти, а 128K, т.е. доступ ко всем 512K возможен только из режима CP/M.

Поэтому использование "Honey Commander" практически теряет смысл, т.к. максимальный RAM DISK теперь в нем всего лишь 32K.

Более того, как "не шли", так и "не идут" такие программы, как: INSULT MEGADEMO, ECSTASY MEGADEMO, MIRACLE DEMO, PSG-WINS, некоторые системные программы и игры, в интро которых используется команда OUT (#FD),A.

Поэтому, если Вы хотите, чтобы у Вас работали все эти и многие другие программы, а также в SPECTRUM'овском режиме был доступ ко всем 512K, то нужно припаять еще 2 микросхемы и один тумблер.

Итак, все по порядку:

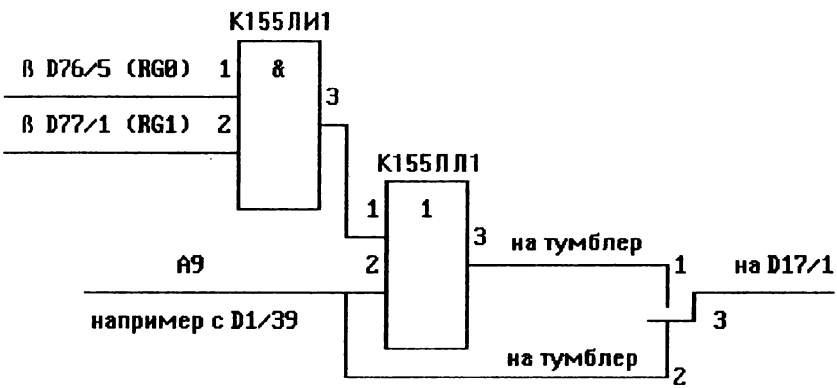


Рис.2

1. Вместо сигнала A9 на 13 ногу 555КП12, показанную на рис.1, нужно подать +5В.

2. Спаять схему,показанную на рис. 2

Нужно взять 2 микросхемы К155ЛИ1 и К155ЛЛ1. На 1 ножку К155ЛИ1 нужно подать сигнал RG0 с 5 ножки D76 (К555ЛН1). На 2 ножку К155ЛИ1 надо подать сигнал RG1 с 1 ножки D77 (К555КП11). На 1 ножку К155ЛЛ1 нужно подать сигнал с 3 ножки К155ЛИ1. На 2 ножку К155ЛЛ1 надо подать сигнал A9, взятый, например, с 39 ножки D1 (Z-80).

3. Отрезать от D17/1 сигнал A9 и вместо него припаять к D17/1 провод. Другой конец этого провода нужно припаять к тому контакту тумблера, который может коммутироваться в одном положении с одним контактом, а в другом с другим контактом тумблера (т.е. либо замкнута цепь 3-2, как на РИС.2, либо цепь 3-1).

4. На один из двух оставшихся контактов тумблера нужно припаять сигнал с 3 ножки К155ЛЛ1, а на другой - подать сигнал A9.

Наиболее удобно использовать обычный двухпозиционный тумблер.

Теперь, если Вы установите тумблер в то положение, как показано на РИС.2, т.е. замкнете контакты тумблера 3 и 2, то Вы будете иметь доступ к 512K в SPECTRUM'овском режиме (а не только в CP/M), и в "Honey Commander" Вы увидите это. Теперь можно полноценно работать с этой замечательной программой (особенно хороша ее четвертая версия).

Если же Вы замкнете контакты тумблера 3 и 1 (см. РИС.2), то у Вас будут идти все программы, которые шли раньше плюс те, которые используют команду OUT (#FD),A (INSULT MD, ECSTASY MD и многие другие).

ВНИМАНИЕ !!! Не забудьте к 7-ым ножкам К155ЛИ1 и К155ЛЛ1 припаять землю, а к 14-ым- +5В.

Вот и все. Море удовольствия ждет Вас после 20-30 минут работы !

ЭТЮДЫ

(c) AmiS (095) 412-4705 (Алексей)

Здравствуйте, уважаемые поклонники ZX-РЕВЮ! Хочу обратиться к тем, кто присылает свои процедуры для публикации на страницах журнала.

Многие уже писали о том, что пора бы покончить с сырыми и непродуманными процедурами. Но, сами при этом делают в своих работах недопустимые, с моей точки зрения, просчеты, как в алгоритмах и синтаксисе, так и в реализации. На это и хочу обратить Ваше внимание. Далее я приведу несколько простейших примеров и объясню зачем это нужно.

Перед нами задача: написать самую короткую процедуру очистки экрана (естественно без использования ПЗУ).

Многие начинающие сделают нечто похожее на:

```
ld      hl,#4000
ld      de,#4001
ld      bc,#1aff
ld      (hl),l
ldir
```

Посидев некоторое время можно сократить процедуру:

```
ld      hl,#5b00
dec     hl
xor     a
ld      (hl),a
or      h
jr      nz,$-6
```

или

```
ld      h,#23
ld      (hl),0
ld      a,h
cp      #5b
jr      nz,$-6
```

что работает несколько быстрее, но портит ячейку #5b00, и следовательно не удовлетворяет нашим требованиям.

Таких простейших приемов можно придумать огромное множество. Большинство из них применять даже нецелесообразно. Тогда спрашивается, зачем это нужно? Отвечу. Эти упражнения нужны не только для применения в своих программах (хотя, как правило, почти все они становятся для Вас стандартом), а для "тренировки мозгов" Если данная "тренировка" станет для Вас обыденным делом, то Вы вскоре поймете, что уже не можете писать свои программы без нее. Данная тренировка должна работать автоматически и только тогда о Вас можно будет сказать, что Вы стали настоящим "крутым" кодером.

Посмотрите на следующий пример. Он родился за несколько ЧАСОВ упорного штурма. Зато теперь можно смело применять эту идею в своих программах:

Задача: "разворот" байта (т.е. обмен битов 0-7,1-6,2-5,3-4)

было 76543210 %(01100001)

стало 01234567 %(10000110)

```
ld      a,#80
add     hl,hl
rra
jr      nc,$-2
ld      h,a
```

Байт приходит в h, в нем же и уходит.

Разберите подробно, что и как делает данная процедура, и Вам станет понятен ход моих мыслей и ранее изложенных доводов.

Особенно обратите внимание на то, как тут используется регистр А.

Для "домашнего задания" могу предложить Вам оптимизировать простейшие процедуры умножения, деления, взятие факториала, возведение в степень, ВЫЧИСЛЕНИЕ sin и cos. Попробуйте оптимизировать данные процедуры как по размеру, так и по скорости исполнения. Т.е. получить два разных варианта одной процедуры. Для тех, кто давно знаком с этим приемом, предлагаю на страницах ZX РЕВЮ опубликовать свои наработки. В последующих выпусках я предложу на Ваше рассмотрение доведенные до совершенства процедуры лучших кодеров, таких как KSA, MAX IWAMOTO, RST 7, STALKER, TITUS и т.д., естественно, с их согласия.

Если у Вас есть модем и Вы живете в Москве, то можете присылать свои работы по вышеуказанному телефону. Или приносить/присылать их в ИНФОРКОМ с пометкой "ОПТИМИ-ЗАЦИЯ"

Внимание! На данную тему объявляется конкурс.
AmigaNet: 39:241/17.8
FidoNet: 2:5020/659.51
****

ИФК: Максим Лупан из Адлера прислал материал, который должен вызвать интерес не только у начинающих, но и у опытных программистов.

КОРР· Процедуры две: первая - загрузчик с индикатором.

В ZX РЕВЮ 96/4-5 появились два отличных загрузчика Сергея Колотова и Богдана Артемьева. Не в силах что-либо противопоставить первому, я решил сделать свой вариант второго. Наблюдая как-то за инстоляцией игрушки на IBM в оболочке WINDOWS 95, я подумал: а почему-бы не сделать подобный загрузчик для SPECCY? Для дисковода нет смысла, т.к. загрузка не занимает более 10 секунд. А вот для магнитофона получилось вполне подходяще. Правда, пришлось распахнуть шкалу на весь экран, но суть (скорость заполнения в зависимости от длины файла) остались неизменной.

Вообще, в последнее время наметился ощутимый прорыв в области магнитофонной загрузки. Я думаю, это естественно, ведь все начинали с магнитафона (по крайней мере большинство) и теперь, когда дисковод стал обычной вещью, многих (в том числе и меня) снова потянуло к ленточным загрузчикам где возможностей развернуться гораздо больше.

Вторая процедура это SPRITE OUT для спрайтов с маской с точностью до пикселя. Она очень быстрая, хотя и учитывает переход через границу экрана. Если Вам этого не надо, ее можно ускорить. Даже применяя таблицы, увеличить ее скорость нельзя.

И еще у меня обращение к авторам "Этюдов" Если Вы начинающий, то не спешите в ZX РЕВЮ с первым же вариантом Вашей процедуры, оптимизируйте ее, думайте над ней так, чтобы голова трещала. Только так Вы станете человеком, который пишет отличные программы.

И еще пара слов об Ассемблерах. В частности, о MASM 1.1. Это очень удобный, очень быстрый любительский ассемблер, но не профессиональный. Профессио-

нальным является GENS (может быть GENIUS, но я не видел его), а MASM это лишь нечто вроде ZEUS для 128 кб. Профессиональный ассемблер должен иметь следующие качества:

- макросы (полноценные, а не как в MASM);
- условия трансляции;
- работа с принтером;
- обработка текста в один проход;
- и безусловно, родственный монитор с возможностью получения исходного текста.

Теперь сама процедура:

```
; (c) 1996 MAXIM LUPAN
LOADER  DI
        CALL    3435
 очистка экрана
        LD      HL,MES
; текст по каналу #FD
PRLOOP  LD      A,(HL)
        RST     #10
        INC     HL
        LD      A,(HL)
        INC     A
        JR      NZ,PRLOOP
        LD      DE,17
; загрузка заголовка
        LD      IX,LOADBF
        SCF
        INC     D
        EX      AF,AF'
        DEC     D
        CALL    #562
        JR      NC,ERROR
 можно изменить
        LD      IX,(LOADBF+13)
        LD      DE,(LOADBF+11)
        LD      A,D
        EXX
        LD      E,A
 в E - длина/256
        XOR     A
        LD      B,A
 в B - счетчик на 256 блоков
        LD      HL,#52E0
; адрес линейки в экране
        LD      C,#80
        EXX
        LD      D,A
; в DE - остаток от
        SUB     1
; деления длины/256
        EX      AF,AF'
        CALL    #562
        EXX
LOADL   LD      A,(HL)
 заполнение линейки
        OR      C
        LD      (HL),A
        INC     H
        LD      (HL),A
        INC     H
        LD      (HL),A
        INC     H
        LD      (HL),A
        RRC     C
        JR      NC,LNEXT
; следующее знакоместо
        INC     L
LNEXT   LD      H,#52
        LD      A,E
        EXX
        LD      E,A
        LD      (IX),L
 сокращенный вариант ПЗУ
        INC     IX
        DEC     DE
        LD      A,(#5C7)
; коррекция задержки,
        INC     A
; может быть и излишняя осторожность?
        LD      B,A
        CALL    #5C8
        EXX
        DJNZ    LOADL
ERROR   LD      HL,10072
; выход в BASIC
        EXX
        EI
        RET
MES     DB      22,0,12,"LOADIN
                G",13,6,6,255
```

```
LOADBF    DS      17

LENGTH OF OBJECT: #0082 (130)

 SPROUT (c) 1996 MAXIM LUPAN
 в HL - адрес спрайта с чередовани-
ем маски данных
 в B - ширина
 в C - высота
 в D - Y (в пикселях)
 в E - X (в пикселях)
SPROUT    PUSH    HL
          CALL    ADRPIX
 вычесления адреса
          PUSH    HL
 в экране
          LD      HL,0
 очистка участков
          LD      (SOROL1),HL
 ролирования
          LD      (SOROL1+2),HL
          LD      (SOROL2),HL
          LD      (SOROL2+2),HL
SOFORM1   SCF
 создание маски
          RR      L
          INC     H
          RLCA
          JR      NC,SOFORM1
          DEC     H
          LD      A,L
          OR      A
          RLA
          LD      (SOMAS1+1),A
          LD      (SODAT2+1),A
          CPL
          LD      (SOMAS2+1),A
          LD      (SODAT1+1),A
          LD      A,#0F
 код команды RRCA
          BIT     2,H
          JR      Z,SORI6H
          LD      A,H
          XOR     7
          INC     A
          LD      H,A
          LD      A,7
; код команды RLCA
SORI6H    PUSH    BC
; создание роллирующих
          LD      DE,SOROL1
 участков
          LD      BC,SOROL2
          INC     H
SOFORM2   DEC     H
          JR      Z,SOEXIT
          LD      (BC),A
          LD      (DE),A
          INC     BC
          INC     DE
          JR      SOFORM2
SOEXIT    POP     BC
          POP     HL
          POP     DE
SOEI6H    PUSH    BC
          PUSH    HL
SOLOOP    LD      A,(DE)
SOROL1    NOP
 ролирование маски
          NOP
          NOP
          NOP
          LD      C,A
          PUSH    HL
SOMAS1    OR      0
          AND     (HL)
; левая часть маски
          LD      (HL),A
          LD      A,#1F
          INC     L
          AND     L
          JR      NZ,SOCONT1
 оценка перехода через
          LD      A,L
 границу экрана с
          SUB     #20
 возвратом с другой
          LD      L,A
; стороны
SOCONT1   LD      A,C
SOMAS2    OR      0
          AND     (HL)
 правая часть маски
          LD      (HL),A
          INC     DE
          POP     HL
          LD      A,(DE)
SOROL2    NOP
```

```
 ролирование данных
          NOP
          NOP
          NOP
          LD      C,A
SODAT1    AND     0
          OR      (HL)
          LD      (HL),A
          LD      A,#1F
          INC     L
          AND     L
          JR      NZ,SOCONT2
          LD      A,L
          SUB     #20
          LD      L,A
SOCONT2   LD      A,C
SODAT2    AND     0
          OR      (HL)
          LD      (HL),A
          INC     DE
          DJNZ    SOLOOP
 переход к следующему
          POP     HL
; знакоместу
          POP     BC
          CALL    DOWNHL
          LD      A,H
; если вышел из экрана
          CP      #58
, возврат
          RET     NC
          DEC     C
          JR      NZ,SOEI6H
; цикл по высоте
          RET
ADRPIX    LD      A,E
 вычесление по
          RLCA
 координатам D-y E-x
          RLCA
; адреса HL
          RLCA
 и бита его положение
          LD      C,A
 в байте) в A
          CPL
          OR      #C7
          LD      (MODPIX+1),A
          LD      A,L
          XOR     D
          AND     #C7
          XOR     D
          RLCA
          RLCA
          LD      L,A
          LD      A,D
          RRA
          RRA
          RRA
          AND     #18
          XOR     D
          AND     #F8
          XOR     D
          ADD     A,#40
          LD      H,A
          XOR     A
MODPIX    SET     0,A
          RET
; подпрграмма DOWNHL стандартная

длина SPROUT #A1 (161)
длина ADRPIX #24 (36)
****************************************
```

(c) Василий.

Hello, Inforcom. Я хочу на страницах вашего журнала обратиться к вашим читателям, посылающим свои процедуры в раздел Этюды.

Уважаемые "производители" этюдов! Прежде чем посылать в журнал свои процедуры постарайтесь сначала оптимизировать их, сделать проще и эффективнее. Не торопитесь! Пусть Ваша процедура прекрасно работает, пусть даже в ней заложена хорошая и нужная идея, но если ее можно сделать короче в 1,5 раза, и если она ужасна по своей структуре, то какой же это "перл" и, тем более, "алмаз" Этим самым Вы портите замечательную идею А.А.Иванова. Я конечно же имею в виду не все про-

цедуры - некоторые из них воистину шедевры, но речь идет не о них.

В качестве примера (я мог взять любой другой) привожу на ваш суд свое собственное решение процедуры Redefine Keys (аналогичную процедуре А.Шокало и написанную мной еще полгода назад). Она занимает всего 14 байт (у А.Шокало 23 байта). Вот она:

```
REDEF   LD      BC,#7FFE
C1      RRC     B
        IN      A,(C)
        CPL
        AND     31
        JR      Z,C1
        LD      C,A
        RET
```

Сравните ее с процедурой А.Шокало (ZX РЕВЮ 96/4-5). Как Вы видите она гораздо короче, проще и понятнее. Старший байт порта в регистре B, бит нажатой клавиши установлен в C.

Предлагаю вам также переделанную мною процедуру TRANSF (30 байт против 46).

```
TRANSF  CALL    CO
        RLCA
        RLCA
        RLCA
        LD      E,A
        LD      A,B
        CPL
        CALL    CO
        ADD     A,E
        LD      E,A
        LD      A,44
        SUB     E
        LD      L,A
        LD      H,2
        LD      A,(HL)
        RET
CO      LD      H,#FF
CC      INC     H
        RRA
        JR      NC,CC
        LD      A,H
        RET
```

И в заключение даю в таблице пару советов, посвященных оптимизации программ (уж это то должен знать каждый, кто работает в ассемблере):

Команда	Команда на которую можно заменить	Примечание. Следует учитывать при замене
xor #ff	cpl	Инверсия регистра A. CPL не изменяет, в отличии от XOR, флаги CY, Z, P/V, S
cp 0	and a (or a)	При проверке на 0, CP 0 можно заменить на AND A. Флаг Z устанавливается точно также.
and x cp 0 jr z,lll	and x jr z,lll, где x=n,A,B,C,D,E,H,L,(hl),(ix+d),(iy+d)	cp 0 делать не обязательно. см 2.
ld a,0	xor a, sub a	При SUB A и XOR A портятся флаги, поэтому использование их не всегда возможно. (Команда LD флаги не портит)
scf ccf	and a	Обнуление флага CY. AND A портит другие флаги и это следует учитывать

ret z jr mmm	jr z,mmm ret	Выигрыша в размере нет, но Вашу процедуру будет легче читать(видно конец процедуры)

P.S. Предлагаю конкурс. Придумайте процедуру календарь. В регистре DE задается год, в регистре A - число. Нужно получить в регистре A день недели (0...6 или 1...7, где 0 и 7 - воскресенье).

****

(c) Александр Лавров г.Тольятти.

В ZX РЕВЮ 96/3 были даны два скроллинга экрана (автор Вадим Куликов). Эти процедуры можно ускорить, если заменить внутренний цикл (строки с 90 по 130) на LDIR и поменять расчет адреса следующей линии.

Идея использовать LDIR в процедурах скроллинга взята из книги "Как написать игру для ZX Spectrum" (часть 2-ая) фирмы "Питер"

```
SCROLL UP
10  (c) A. Lavrov from
20  NEW STANDARD
30          ORG     40000
40          LD      DE,#4000
50          LD      HL,#4100
60          LD      A,191
70 UP1      LD      BC,32
80          PUSH    HL
90          LDIR
100         POP     DE
110         LD      H,D
120         LD      L,E
130         INC     H
140         EX      AF,AF'
150         LD      A,H
160         AND     #07
170         JR      NZ,NXTU
180         LD      A,H
190         SUB     8
200         LD      H,A
210         LD      A,#20
220         ADD     A,L
230         LD      L,A
240         JR      C,UP2
250 NXTU    EX      AF,AF'
260         DEC     A
270         JR      NZ,UP1
280         RET
290 UP2     LD      A,8
300         ADD     A,H
310         LD      H,A
320         JR      NXTU
```

```
SCROLL DOWN
10  (c) A.Lavrov
20  from NEW STANDARD
30          ORG     40000
40          LD      DE,#57E0
50          LD      HL,#56E0
60          LD      A,191
70 DW1      LD      BC,32
80          PUSH    HL
90          LDIR
100         POP     DE
110         LD      H,D
120         LD      L,E
130         DEC     H
140         EX      AF,AF'
150         LD      A,H
160         AND     #07
170         XOR     #07
180         JR      Z,DW2
190 NXTD    EX      AF,AF'
200         DEC     A
210         JR      NZ,DW1
220         RET
230 DW2     LD      A,H
240         ADD     A,8
250         LD      H,A
260         LD      A,L
270         SUB     #20
280         LD      L,A
290         JR      NC,NXTD
300         LD      L,#E0
310         LD      A,H
320         SUB     8
330         LD      H,A
340         JR      NXTD
```

Скорость скроллинга должна увеличится почти вдвое. Правда, нарушена синхронность, если использовать процедуры в паре. Т.е. процедура SCROLL DOWN работает медленнее чем SCROLL UP А именно, в каждой установке новой линии на 2 такта, новой строки на 10 тактов и нового сегмента на 2 такта дольше.

Для скроллинга окна надо изменить часть процедуры SCROLL_DOWN (для SCROLL_UP это не требуется), вставив такие строки:

```
280        JR    C,DW3
285        LD    L,A
290        JR    NXTD
295 DW3    LD    A,L
300        OR    #E0
305        LD    L,A
```

Количество перемещаемых строк (пиксельных) задается в строке 60, в А. Адрес верхнего левого или нижнего левого углов, как и раньше в DE, строка 40. Ширина окна (в знакоместах) в BC, строка 70.

Чтобы процедуры сами рассчитывали начальный адрес в HL, надо заменить строки 50 и 60 в обоих процедурах, и поставить метку в строку 110:

```
50     LD    A,n
  количество пиксельных ли-
ний + 1
60     JR    s
; метка в строке 110.
```

****

ИФК.: Евгений и Алексей Федяевы представили на рассмотрение читателей ZX РЕВЮ процедуру, взятую из boot REAL MASTER, и свою процедуру рисования круга.

КОРР.: В boote красиво горит надпись в верхней части экрана. Формат картинки: справо побайтно, сверху вниз по 1 пиксельной строчке, без атрибутов.

Если IX указывает на пустую область памяти, то вся заданная область экрана сгорает.

```
        ORG    40000
GGG     CALL   FIRE
        CALL   OG1
        LD     A,127
        IN     A,(#FE)
        BIT    0,A
        ,RET   Z
        JR     GGG

OG1     LD     A,R
; п.п. случайных
        LD     L,A
 чисел
        LD     A,R
        AND    7
        LD     H,A
        PUSH   HL
        POP    IY
        RET

FIRE    LD     IX,50000
; адрес картинки
        LD     C,64
; ее высота в PIX
        LD     HL,#4000
; адрес вывода в экране
        LD     E,L
        LD     D,H
OG2     PUSH   HL
        LD      B,14
; ширина в байтах
OG3     XOR    (IY+0)
        OR      (IY+1)
 если убрать эту команду,
        AND    (HL)
 то высота огня уменьшится.
        OR     (IX+1)
 если заменить на OR 0,
        LD     (DE),A
 то все по точкам сгорит.
        INC    L
        INC    E
        INC    IY
```

```
        INC     IX
        DJNZ    OG3
        POP     HL
        LD      E,L
        LD      D,H
        CALL    DAD1
        DEC     C
        JR      NZ,OG2
        RET

DAD1    INC     H
; п.п. расчета адреса в
        LD      A,H
; дисплейном файле
        AND     7
 на PIX ниже
        RET     NZ
        LD      A,L
        ADD     A,32
        LD      L,A
        RET     C
        LD      A,H
        SUB     8
        LD      H,A
        RET
```

Теперь программа рисования круга без использования вычислений с плавающей точкой.

Алгоритм рисования круга взят из книги Роджерса Д. "Алгоритмические основы машинной графики" Перевод с англ. Москва, издательство "Мир", 1989.

В этой реализации алгоритма Брезенхема радиус круга ограничен 49-ю точками. При желании можно доработать программу как для работы с большим радиусом, так и для одновременного рисования восьми частей круга (здесь реализован вариант рисования одновременно 4 четвертей круга).

Теперь описание алгоритма. Основные положения:

1) круг находится в своей системе отсчета - его центр в координатах (0,0);
2) строится он от координат (0,-R) до (R,0), т.е. дуга;
3) остальные 3 дуги получаются отражением от координатных осей;
4) при расчетах следует выбрать 1 из 3 случаев: увеличивается только X, только Y, увеличиваются и X и Y;
5) этот алгоритм основывается на приближенности точки к идеалу. Для этого введем величину C. Возьмем, к примеру, смещение точки по диагонали, тогда: C=(X+1)^2+(Y+1)^2-R^2.
6) начальное значение: C=(0+1)^2+(Y+1)^2-R^2=2*(1-R)
7) величины для горизонтального и вертикального смещения: CG=(X+1)^2+Y^2-R^2=C-(2*Y+1); (Y+1)^2-R^2=C-(2*X+1)
8) выбирается, какая из них ближе к 0, т.е. наименьшую по абсолютной величине, т.к. по построению X>0, Y<0 видно, что при C<=Y ближе будет CG, т.е. выбирается горизонтальное направление C>X ближе CV, т.е. вертикальное, в остальных случаях диагональное.

Теперь сам алгоритм:

1) X=0,Y=-R,C=2*(1-R)=2*(Y+1)
2) PLOT (+-X,+-Y)
3) ЕСЛИ Y=0 ТО ВЫХОД
4) C1=C
5) ЕСЛИ C1>X ТО НА ПУНКТ 8
6) X=X+1, C=C+2*X+1
7) ЕСЛИ C1<=Y ТО НА ПУНКТ 2
8) Y=Y+1, C=C+2*Y+1
9) НА ПУНКТ 2

Существует подобный алгоритм рисования эллипса, но об этом в следующий раз.
****

Тимоти из Санкт-Петербурга предлагает несколько процедур для системных программистов.

1. Процедура для быстрой остановки дисковода (но без сброса ВГ):

```
STOP_D    LD      A,(#5D16)
 копия системного регистра
          PUSH    AF
          XOR     3
 меняем дисковод
          CALL    STOPD1
          POP     AF
, восстанавливаем дисковод
STOPD1    LD      HL,#02B9
 для TR-DOS 5.03
          PUSH    HL
          JP      #3D2F
```

2. Информация для владельцев Scorpion 256. Отладчик можно отключить, оставив остальной сервис-монитор рабочим:

```
DI
LD        A,#10
LD        BC,#7FFD
OUT       C,(A)
LD        B,#1F
OUT       (C),A
          , RAM #08
LD        A,#FF
LD        (#C063),A
; отключаем монитор.
SUB       A
OUT       (C),A
; RAM #00
EI
RET
```

Этот способ лучше, чем описанный А.Ларченко вариант отключения через системный рестарт, т.к. в последнем случае произойдет остановка по ошибке, если машина не Scorpion. К сожалению, я не смог проверить это на ранних версиях монитора.

ИФК.: и нет гарантии, что будет работать на новых версиях. Данный способ является, мягко говоря, некорректным т.к. не обеспечивает совместимость с различными версиями (хотя наверняка данная переменная имеет фиксированное положение, но кто знает...). А.Ларченко не зря указал именно тот способ отключения монитора, от которого отказался Тимоти. Для обеспечения совместимости с не Scorpion'ами можем порекомендовать следующий прием:

```
LD        HL,ADR
PUSH      HL
LD        (#5C3D),SP
RST       8
DB        #85
ADR
```

Данный способ позволит Вашей программе корректно работать на любой машине. Кстати, рекомендуем использовать этот вариант и для остальных рестартов, используемых в Scorpione.

Но вернемся к письму...

3. Проверка версии TR-DOS:

```
LD        HL,#0368
LD        C,#13
CALL      #3D13
LD        A,(#5CDD)
CP        #44
JR        Z,TR-DOS 5.03
CP        #31
JR        Z,TR-DOS 5.01
```

4. По поводу TURBO-LOADER из ZX-РЕВЮ 95/5. Не знаю, у кого как, а у меня для достижения "турбости" нужно в процедуре позиционирования команду "поиск и проверка" (LD A,#1C) заменить на просто "поиск" (LD A,#18).

****

ИФК.: О своих успехах в освоении ассемблера решил написать Хороших Дмитрий из города Сосенский, Калужской области.

КОРР: Совсем недавно, после прочтения книги "ИНФОРКОМА" о программировании в машинных кодах (КИ1), я начал писать программы на Ассемблере (и что самое интересное - они даже начали работать). Я хочу предложить свою процедуру для заполнения экрана. Такой эффект я видел в игре "CJ in the USA" Я не стал взламывать программу, а решил написать свою процедуру, которая бы воспроизводила такой эффект. Идея алгоритма пришла мне в голову во время просматривания в книге Инфоркома "Элементарная графика" процедуры для закраски части экрана. Там используется конвейерная организация стека, которую я и применил в своей программе.

Процедура занимает всего 188 байт (без буфера). Экран обязательно должен располагаться с адреса 32768. Программе также требуется некоторая область памяти (100 байт в большинстве случаев будет предостаточно), которая определяется в строках 1070 и 1080. STK_B_F это физическая верхняя граница (конец) стека. В листинге под стек отдана область памяти с 41000 по 41100. Благодаря строкам 760-900 при "доезжании" конвейером до конца отданной ему области памяти, он автоматически копируется в начало, и таким образом использует отданную ему область памяти циклически. Программа может заполнять экран параллельно с любого количества мест. Это определяется заполнением конвейера: нужно занести в STK_BEG, которая является началом буфера, 41000, а в область памяти, начиная с 41000, координаты тех точек, с которых программа должна заполнять экран: сначала X, потом Y.

Например, для трех точек (10,6), (16,16) и (22,6) область памяти с 41000 по 41005 будет содержать значения 10,6,16,16,22,6. В переменной STK_END в таком случае должно быть 41006, т.е. она указывает на адрес на единицу больший, чем конец буфера. Единственным ограничением при использовании процедуры является то, что на экране не может быть знакоместа с атрибутом 0 (т.е. черный цвет на черном фоне). Я не думаю, что это серьезное ограничение, а процедура использует это для распознования тех мест, где она еще не закрасила экран.

Кстати, этот факт может использоваться для усложнения эффекта: например, если между строками 110 и 120 изменить цвет различных частей экрана на отличный от 0, то заполнение не коснется этих частей экрана, а если в каждом знакоместе объединить вывод цветовой и графической информации, то можно будет в один экран "впечатывать" фрагмент любого другого экрана, причем фрагмент произвольной формы. Достаточно только отметить этот участок атрибутом 0. Это может делать, например, процедура PRINT, вызов которой вставляется между строками 530 и 540. В таком случае строки 80-110 не нужны вообще, а строки 30-70 не нужны в том случае, когда пользова-

тель сам заполнит нужный участок атрибутом 0.

Процедура заполнения экрана:

```
FILL SCREEN
10          ORG   40000
20          ENT
30          LD    HL,#5800        ; экран
40          LD    DE,#5801        ; заполняется
50          LD    BC,#2FF         ; атрибутом 0
60          LD    (HL),L
70          LDIR
80          LD    HL,#8000        ; копирование
90          LD    DE,#4000        ; графической
100         LD    BC,#1800        ; информации
110         LDIR                  ; на экран
120 BEGIN   LD    HL,(STK_BEG)    ; снятие координат
130         LD    D,(HL)          ; с вершины конвейера
140         INC   HL              ; и занесение в DE
150         LD    E,(HL)          ; X -> DE, Y -> DE
160         INC   HL
170         LD    (STK_BEG)
180         PUSH  DE              ; сохраним DE
190         LD    A,E             ; если Y=0, то на
200         OR    A               ; CONT_1
210         JR    Z,CONT_1
220         DEC   E               ; иначе проверка
230         CALL  VERIFY          ; точки сверху
240 CONT_1  POP   DE              ; восстановим DE
250         PUSH  DE              ; снова сохраним
260         LD    A,E             ; если Y=28, то на
270         CP    28              ; CONT_2
280         JR    Z,CONT_2
290         INC   E               ; иначе проверка
300         CALL  VERIFY            точки снизу
310 CONT_2  POP   DE
320         PUSH  DE              ; аналогично
330         LD    A,D             ; слева
340         OR    A
350         JR    Z,CONT_3
360         DEC   D
370         CALL  VERIFY
380 CONT_3  POP   DE
390         PUSH  DE              ; аналогично справа
400         LD    A,D
410         CP    31
420         JR    Z,CONT_4
430         INC   D
440         CALL  VERIFY
450 CONT_4  POP   DE              ; снимаем DE со стека
460         CALL  TRANS           ; вызов процедуры расчета ад-
```

```
                                        реса в экранной области
470          LD      E,L                ; преобразование
480          LD      A,H                  старшего байта
490          AND     #03                  адреса для буферного
500          OR      #98                  экрана
510          LD      D,A
520          LD      A,(DE)               копирование атрибута
530          LD      (HL),A               из буфера в экран
540          LD      DE,(STK_BEG)       ; проверка, не кончились ли
550          LD      HL,(STK_END)       ; значения на конвейере
560          SBC     HL,DE                если да то конец
570          RET     Z
580          LD      B,#FF
590 PAUSE    DJNZ    PAUSE                пауза
600          JR      BEGIN                все сначала

610 VERIFY   CALL    TRANS              ; пересчет координат
620          LD      A,(HL)             ; проверка атрибута
630          OR      A                    на экране
640          RET     NZ                 ; если не ноль - конец
650          LD      A,63                 иначе атрибут 63 (белый
660          LD      (HL),A               по белому) и
670          LD      HL,(STK_END)         занесение координат
680          LD      (HL),D               на конвейер
690          INC     HL
700          LD      (HL),E
710          INC     HL
720          LD      (STK_END),HL         проверка - не достиг ли
730          LD      DE,(STK_E_F)         конец конвейера
740          SBC     HL,DE                физического конца
750          RET     NZ                   если нет, то конец
760          LD      HL,(STK_END)         иначе вычисляется
770          LD      DE,(STK_BEG)       ; длина конвейера...
780          SBC     HL,DE
790          LD      B,H
800          LD      C,L
810          PUSH    HL                     сохраняется
820          LD      HL,(STK_BEG)         и конвейер копируется
830          LD      DE,(STK_E_F)         в начало отведенной
840          PUSH    DE                 ; ему области памяти
850          LDIR
860          POP     DE                   высчитываются новые
870          LD      (STK_BEG),DE         начало и конец
880          POP     HL                   конвейера
890          ADD     HL,DE
900          LD      (STK_END),HL
910          RET

920 TRANS    LD      A,E      ; на входе в DE - координата
```

```
930          AND    #18     в знакоместах
940          RRA            в D - X, E - Y
950          RRA            на выходе в HL - адрес
960          RRA          ; в файле атрибутов
970          OR     #58
980          LD     H,A
990          LD     A,E
1000         AND    #07
1010         RRCA
1020         RRCA
1030         RRCA
1040         ADD    A,D
1050         LD     L,A
1060         RET

1070 STK_B_F    DEFW    41000
1080 STK_E_F    DEFW    41100
1090 STK_BEG    DEFW    41000
1100 STK_END    DEFW    41006

1110     ORG    41000     занесение координат
1120     DEFB   10,6      на конвейер
1130     DEFB   16,16
1140     DEFB   22,6
```

Текст процедуры PRINT:

```
PRINT
10 PRINT     LD     A,H        пересчет адреса из файла
20           AND    #03      ; атрибутов в дисплейный файл
30           RLCA              изменяется только старший байт
40           RLCA
50           RLCA
60           OR     #40
70           LD     H,A      ; в HL - адрес на экране
80           AND    #18
90           OR     #80
100          LD     D,A        в DE - адрес в буферном экране
110          LD     B,#08
120 REP_1    LD     A,(DE)     копирование восьми
130          LD     (HL),A   ; пиксельных линий
140          INC    D        ; знакоместа
150          INC    H
160          DJNZ   REP_1
170          RET
```

****

Командин Илья (ZX-MANIACS) из г.Калуга.

КОРР: Предлагаю одну довольно интересную процедуру проявления экрана - картинка выезжает из-под нижней кромки. Самое любопытное в этой процедуре то, что, изменив всего 4 стро-

ки, Вы заставите картинку выезжать сверху. Но об этом позже. Вот листинг процедуры.

```
1 ; ZX-MANIACS'1996
2 ; SPECIAL FOR INFORCOM
10          ORG   60000
20 ADDR     EQU   40000
30          ENT   $
40          LD    HL,ADR
50          LD    DE,16384
60          SBC   HL,DE
70          LD    (DIFFER),HL
80          LD    (DIFF),HL
90          LD    B,192
100 LOOP    LD    HL,0
110         LD    (DIFFER),HL
120         PUSH  BC
130         CALL  SCR
140         LD    HL,ADR
150         LD    DE,22496
160         LD    BC,32
170         LDIR
180         LD    HL,(DIFF)
190         LD    (DIFFER),HL
200         CALL  SCR
210         POP   BC
220         DJNZ  LOOP
230         LD    HL,ADDR+6144
240         LD    DE,22528
250         LD    BC,768
260         LDIR
270         RET
280 SCR     LD    A,0
290         LD    C,0
300         PUSH  AF
310         CALL  8880
320         POP   AF
330         LD    DE,(DIFFER)
340         ADD   HL,DE
350         LD    BC,48896
360 SCR1    PUSH  BC
370         INC   A
380         PUSH  AF
390         PUSH  HL
400         CALL  8880
410         LD    DE,(DIFFER)
420         ADD   HL,DE
430         POP   DE
440         PUSH  HL
450         LD    BC,32
460         LDIR
470         POP   HL
480         POP   AF
490         POP   BC
500         DJNZ  SCR1
510         RET
520DIFFER   DEFW  0
530 DIFF    DEFW  0
```

Длина процедуры 107 байт.

В данной процедуре сначала выезжает монохромное изображение, а затем на него накладываются атрибуты. Чтобы все происходило наоборот (а такая потребность появляется часто) необходимо просто перенести строки 230-260 в начало процедуры.

Теперь, как обещал, расскажу о том, что нужно сделать, чтобы картинка выезжала не снизу, а сверху. Для этого, повторюсь, достаточно заменить всего 4 строки:

```
140         LD    HL,ADDR+6112
150         LD    DE,16384
280 SCR     LD    A,191
370         DEC   A
```

КОРР.: Хочу предложить вам еще одну процедуру проявления экрана: довольно медленная, но очень эффектная (картинка выезжает одновременно с двух сторон и постепенно сливается в одно целое). Подобный эффект можно наблюдать в игре HUNT FOR THE RED OCTOBER, но туда я не заглядывл, так что программа полностью авторская.

```
1  (c) ZX-MANIACS'1996
2  SPECIAL FOR INFORCOM
10          ORG   60000
20 ADDR     EQU   40000
30          ENT   $
40          LD    HL,ADDR
50          LD    DE,16384
60          SBC   HL,DE
```

```
70          LD      (DIFFER),HL
80          LD      B,0
90 LOOP     PUSH    BC
100         LD      A,1
110         LD      BC,24826
120 SCRL1   PUSH    BC
130         PUSH    AF
140         CALL    8880
150         PUSH    HL
160         LD      DE,(DIFFER)
170         ADD     HL,DE
180         LD      B,32
190         XOR     A
200 SCRL2   RL      (HL)
210         DEC     HL
220         DJNZ    SCRL2
230         POP     HL
240         LD      B,32
250 SCRL3   RL      (HL)
260         DEC     HL
270         DJNZ    SCRL3
280         POP     AF
290         INC     A
300         INC     A
310         POP     BC
320         DJNZ    SCRL1
330         LD      A,0
340         LD      BC,24577
350 SCRR1   PUSH    BC
360         PUSH    AF
370         CALL    8880
380         PUSH    HL
390         LD      DE,(DIFFER)
400         ADD     HL,DE
410         LD      B,32
420         XOR     A
430 SCRR2   RR      (HL)
440         INC     HL
450         DJNZ    SCRR2
460         POP     HL
470         LD      B,32
480 SCRR3   RR      (HL)
490         INC     HL
500         DJNZ    SCRR3
510         POP     AF
520         INC     A
530         INC     A
540         POP     BC
550         DJNZ    SCRR1
560         POP     BC
570         DJNZ    LOOP
580         LD      HL,ADDR +
                    6144
590         LD      DE,22528
600         LD      BC,768
610         LDIR
620         RET
630 DIFFER  DEFW    0
```

Длина процедуры 107 байт.

Теперь расскажу немного об этих процедурах. Во-первых, у них есть один недостаток они окончательно и бесповоротно портят картинку, загруженную в ADDR. Не забывайте об этом, если захотите использовать их в своих программах. Во-вторых, надо все же немного рассказать, на каком принципе обе эти программки построенны. В обеих процедурах активно используется подпрограмма ПЗУ с точкой входа 8880. Эта подпрограмма вычисляет адрес байта в видеобуфере по координатам точки, заданным в пикселях. Началом отсчета считается левый верхний угол экрана. На входе в подпрограмму в аккумулятор загружается вертикальная координата точки, а в регистр С горизонтальная. На выходе: в HL вычисленный адрес в видеобуфере, в А - величина смещения точки в пикселах от левого края вычисленного байта. Об этой подпрограмме я узнал из книги издательства "Питер" "Как написать игру на ассемблере" Советую всем приобрести эту книгу. Вместе с небезызвестной книгой "Инфоркома" она является отличным пособием и справочником для тех, кто программирует на ассемблере.

Вообще, подпрограмма 8880 довольно интересна и может с успехом использоваться в процеду-

рах, работающих с экраном. Так например, широкоизвестную процедуру "Штора" с помощью этой подпрограммы удалось сократить до 49 байтов:

```
1; (c) ZX-MANIACS'1996
2; SPECIAL FOR INFORCOM
10        ORG   60000
20        ENT   $
30        LD    HL,ADDR
40        LD    DE,16384
50        SBC   HL,DE
60        LD    B,H
70        LD    C,E
80        LD    A,191
90 LOOP   PUSH  AF
100       PUSH  BC
110       LD    C,0
120       CALL  8880
130       POP   BC
140       POP   AF
150       PUSH  BC
160       LD    D,H
170       LD    E,L
180       ADD   HL,BC
190       LD    BC,32
200       LDIR
210       HALT
220       POP   BC
230       DEC   A
240       CP    255
250       JR    NZ,LOOP
260       LD    HL,ADDR +
                6144
270       LD    DE,22528
280       LD    BC,768
290       LDIR
300       RET
```

Вполне возможно, что можно сократить программу еще больше - я не очень старался сделать это, т.к. моя основная задача была несколько иной показать преимущества подпрограммы 8880.

Также, очень даже хорошо сократилась (до 92 байтов) процедура Николая Губина (ZX РЕВЮ 96/1-2, стр. 77). Правда я сделал ее без IM2, и моя процедура отлично работает на компьютерах с ОЗУ 48К. Я надеюсь, что уважаемый автор не обвинит меня в плагиате, т.к. в наших программах общая только идея (да и то она не наша):

```
1; (c) ZX-MANIACS'1996
2; SPECIAL FOR INFORCOM
10        ORG   60000
20        ENT   $
30        LD    A,95
40        LD    (SCRUP),A
50        INC   A
60        LD    (SCRDN),A
70        LD    HL,ADDR
80        LD    DE,16384
90        SBC   HL,DE
100       LD    (DIFF),HL
110       LD    B,96
120 LOOP  PUSH  BC
130       LD    A,(SCRUP)
140       PUSH  AF
150       CALL  8880
160       POP   AF
170       PUSH  HL
180       LD    DE,(DIFF)
190       ADD   HL,DE
200       POP   DE
210       LD    BC,32
220       LDIR
230       DEC   A
240       LD    (SCRUP),A
250       LD    A,(SCRDN)
260       PUSH  AF
270       CALL  8880
280       POP   AF
290       PUSH  HL
300       LD    DE,(DIFF)
310       ADD   HL,DE
320       POP   DE
330       LD    BC,32
340       LDIR
350       IMC   A
360       LD    (SCRDN),A
370       HALT
380       HALT
390       POP   BC
400       DJNJZ LOOP
```

```
410          LD      HL,ADDR +
                     6144
420          LD      DE,22528
430          LD      BC,768
440          LDIR
450          RET
460 SCRUP    DEFB    0
470 SCRDN    DEFB    0
480 DIFF     DEFW    0
```

Еще одна положительная особенность этой программы - она многоразовая.

Ну и на последок я отвлекусь от темы проявления экрана и предложу процедурку, которая поможет Вам в поиске символьного набора и шаблонов спрайтов. В верхней трети экрана она выводит столбцами по 8 знакомест содержимое памяти компьютера. За один раз она выводит 2048 байт.

```
 1; (c) ZX-MANIACS'1996
 2; SPECIAL FOR INFORCOM
10          ORG     64000
20          ENT     $
30          LD      HL,16384
40          LD      DE,(23296)
50          LD      B,32
60 M3       PUSH    BC
70          PUSH    HL
80          LD      B,8
90 M2       PUSH    BC
100         PUSH    HL
110         LD      B,8
120 M1      LD      A,(DE)
130         LD      (HL),A
140         INC     DE
150         INC     H
160         DJNZ    M1
170         POP     HL
180         LD      BC,32
190         ADD     HL,BC
200         POP     BC
210         DJNZ    M2
220         POP     HL
230         INC     HL
240         POP     BC
250         DJNZ    M3
260         RET
```

Перед запуском процедуры надо занести в ячейки 23296/97 необходимый адрес.

****

(c) Колотов Сергей, SerzhSoft from Shadrinsk, X.1996

КОРР: Добрый день! В ZX РЕВЮ 96/4-5 в ЭТЮДАХ на стр. 80 была опубликована процедура Прокопенко А.В. LOAD-MERGE (дисковый вариант). Задумка, конечно, неплохая, но во время набора и тестирования данной процедуры был обнаружен ряд досадных ошибок.

1. При установке числа байт, по которому происходит поиск имени в каталоге вместо переменной #5D06 изменяется ячейка памяти по адресу #5C06. Но это ведь системная переменная KSTATE6, используемая при ОПРОСЕ КЛАВИШ!

2. Непонятно, зачем по адресу #5DA0 заносится 0. Эта ячейка памяти не является ни системной переменной, ни переменной TR DOS! Скорей всего в листинг закралась опечатка, причем, двойная и в ассемблерной команде, и в комментарии. На самом же деле, вместо #5DA0 должно быть #5D10 - адрес переменной TR DOS, которая обязательно обнуляется перед чтением BASIC-файла (функция #0E для точки входа в TR DOS #3D13).

3. При печати на экране сообщения O.K. после загрузки файла, выдается не номер строки автостарта (как в магнитофонном варианте), а номер последней выполненной BASIC-строки. Следовательно, строки LD HL,(#5C42): LD (#5C45),HL можно безболезненно из программы выкинуть.

Номер строки автостарта узнавать довольно-таки громоздко, да и не очень-то нужно. Главное просто загрузить программу, не дав ей запуститься.

Новый вариант процедуры приведен ниже. Все указанные ошибки в нем исправлены, и вдобавок переделан алгоритм формирования имени файла. Вместо двух проходов (заполнение пробелами, перенос имени) реализован один. То есть вначале переносится имя, и только затем оставшаяся часть буфера заполняется пробелами.

Процедура релоцируема и занимает 64 байта.

```
LD_MRG
        LD      C,#18
        CALL    #3D13
        LD      A,#09
        LD      (#5D06),A
        XOR     A
        LD      (#5CF9),A
        LD      DE,(#5C5D)
        INC     DE
        LD      HL,#5CDD
        LD      B,#08
LP_LM1  INC     DE
        LD      A,(DE)
        CP      #0D
        JR      NZ,GO_LM1
LP_LM2  LD      (HL)," "
        INC     HL
        DJNZ    LP_LM2
        JR      GO_LM2
GO_LM1  LD      (HL),A
        INC     HL
        DJNZ    LP_LM1
GO_LM2  LD      (HL),"B"
        LD      C,#0A
        CALL    #3D13
        INC     C
        JR      Z,ER_LM1
        XOR     A
        LD      (#5D10),A
        LD      C,#0E
        CALL    #3D13
        RST     #08
        DB      #FF
ER_LM1  RST     #08
        DB      #0E
```

Следующие две процедуры - улучшенные аналоги программ Куликова Вадима, которые были опубликованы в ZX РЕВЮ 96/3, стр. 34. Теперь процедуры сдвига экрана вверх и вниз на один пиксел занимают по 33 байта каждая (вместо 46 и 44 соответственно). Строки листинга, заданные как комментарии, отвечают за очистку нужной линии изображения после сдвига (нижняя для SRL_UP и верхняя - для SRL_DN). Если очистка необходима, то просто удалите; в началах этих строк. В этом случае процедуры будут занимать 38 и 39 байт соответственно.

```
SRL_UP
        LD      DE,#4000
LP_SU1  LD      H,D
        LD      L,E
        LD      BC,#0020
        INC     H
        LD      A,H
        AND     #07
        JR      NZ,GO_SU1
        LD      A,L
        ADD     A,C
        LD      L,A
        JR      C,GO_SU1
        LD      A,H
        SUB     #08
        LD      H,A
GO_SU1  PUSH    HL
        LDIR
        POP     DE
        LD      A,H
        CP      #58
        JR      C,LP_SU1
;       XOR     A
;LP_SU2 LD      (DE),A
        INC     E
        JR      NZ,LP_SU2
        RET
```

```
;
SRL_DN
        LD      DE,#57FF
LP_SD1  LD      H,D
        LD      L,E
        LD      BC,#0020
        LD      A,H
        DEC     H
        AND     #07
        JR      NZ,GO_SD1
        LD      A,L
        SUB     C
        LD      L,A
        JR      C,GO_SD1
        LD      A,H
        ADD     A,#08
        LD      H,A
GO_SD1  PUSH    HL
        LDDR
        POP     DE
        LD      A,H
        CP      #40
        JR      NC,LP_SD1
        XOR     A
;LP_SD2 LD      (DE),A
        DEC     E
        JR      NZ,LP_SD2
        LD      (DE),A
        RET
```

В ZX РЕВЮ 96/6 на стр. 40 приведена процедура осыпания экрана. Мне удалось немного упростить структуру программы и уменьшить объем занимаемой памяти (с 52 до 47 байт). Изменяемые константы:

- XRWIND ширина окна осыпания в знакоместах;
- YRWIND высота окна осыпания в пикселях;
- BEG_LN - адрес нижнего правого угла окна в экране.

Итак, вот новая процедура:

```
XRWIND  EQU     32
YRWIND  EQU     191
BEG_LN  EQU     #57E0
;
FALLER
        LD      C,YRWIND
LPFLR1  LD      HL,BEG_LN
        LD      B,C
LPFLR2  PUSH    BC
        LD      D,H
        LD      E,L
        LD      A,H
        DEC     H
        AND     #07
        JR      NZ,GOFLR1
        LD      A,L
        SUB     #20
        LD      L,A
        JR      C,GOFLR1
        LD      A,H
        ADD     A,#08
        LD      H,A
GOFLR1  LD      C,L
        LD      B,XRWIND
LPFLR3  LD      A,(DE)
        PUSH    AF
        OR      (HL)
        LD      (DE),A
        POP     AF
        AND     (HL)
        LD      (HL),A
        INC     E
        INC     L
        DJNZ    LPFLR3
        LD      L,C
        POP     BC
        DJNZ    LPFLR2
        DEC     C
        JR      NZ,LPFLR1
        RET
```

В одном из украинских РЕВЮ я увидел процедуру "переворачивания" экрана. Кажется, ее длина была 56 байт не помню. Так вот, чтобы украинские кодеры не расслаблялись и не поплевывали свысока на наших, я написал свою процедуру, которая занимает всего 36 (!) байт. Итак, УКРАИНА vs РОССИЯ:

```
TRNSCR
        LD      HL,#597F
        LD      DE,#5980
LP_TS1  LD      A,(DE)
```

```
        LDI
        DEC     HL
        LD      (HL),A
        DEC     HL
        BIT     3,H
        JR      NZ,LP_TS1
        LD      D,#40
LP_TS2  LD      A,(DE)
        LD      B,#08
LP_TS3  RLA
        RR      (HL)
        DJNZ    LP_TS3
        RLA
        LD      (DE),A
        INC     DE
        DEC     HL
        LD      A,D
        CP      #4C
        JR      NZ,LP_TS2
        RET
```

Недавно, один из моих друзей попросил меня написать процедуру для эффектного вывода картинки на экран. В чем же оригинальность? Дело в том, что картинка появляется на экране постепенно раскручивающейся спиралью по часовой стрелке. Причем, центр раскрутки задается произвольно и даже может находиться за пределами экрана. Итак, автор идеи - Андреев Андрей (LONE WOLF, Шадринск), ну а программу написал ваш покорный слуга.

Перед вызовом CALL OUTSCR необходимо в регисте E установить координату X центра спирали, а в регистре D - координату Y. В регистре A (аккумуляторе) должен находиться старший байт адреса картинки. А сама картинка должна располагаться в памяти с адреса, кратного 256.

Подпрограмма PUT_ZN выводит на экран одно знакоместо из картинки, если координаты его лежат внутри экрана.

Длина процедуры 118 байт.

```
        ORG     #8000
;
EXAMPL
        LD      E,15 ;X -
                KOOРДИНАТА
        LD      D,11 ;Y
                KOOРДИНАТА
;старший байт картинки, выравненной на сегмент (адрес, кратный 256):

        LD      A,SCREEN/256
;
OUTSCR
        SUB     #40
        EX      AF,AF'
        LD      C,#01
LP_OS0  LD      B,C
LP_OS1  CALL    PUT_ZN
        INC     E
;ВПРАВО
        DJNZ    LP_OS1
        LD      B,C
LP_OS2  CALL    PUT_ZN
        INC     D
;ВНИЗ
        DJNZ    LP_OS2
        INC     C
        LD      B,C
LP_OS3  CALL    PUT_ZN
        DEC     E
;ВЛЕВО
        DJNZ    LP_OS3
        LD      B,C
LP_OS4  CALL    PUT_ZN
        DEC     D
;ВВЕРХ
        DJNZ    LP_OS4
        INC     C
        LD      A,E
        CP      #20
        JR      C,LP_OS0
;X - в экране
        ADD     A,C
        CP      #20
        JR      C,LP_OS0
```

```
;X - в экране
        LD      A,D
        CP      #18
        JR      C,LP_OS0
;Y - в экране
        ADD     A,C
        CP      #18
        JR      C,LP_OS0
;Y - в экране
;иначе - повсюду вышли за экран!
        RET
;выход из процедуры
;
PUT_ZN
        PUSH    BC
        LD      A,30
;величина задержки
LPPZN1  EX      AF,AF'
        LD      C,A
        PUSH    DE
        LD      A,E
        CP      #20
        JR      NC,GOPZN1
;X вне экрана
        LD      A,D
        CP      #18
        JR      NC,GOPZN1
;Y вне экрана
        AND     #07
        RRCA
        RRCA
        RRCA
        ADD     A,E
        LD      E,A
        LD      A,D
        AND     #18
        OR      #40
        LD      D,A
        ADD     A,C
        LD      H,A
        LD      L,E
        LD      B,#08
LPPZN2  LD      A,(HL)
;перенос
        LD      (DE),A
 графики
        INC     H
        INC     D
        DJNZ    LPPZN2
        DEC     D
        LD      A,D
        RRA
        RRA
        RRA
        AND     #03
        OR      #58
        LD      D,A
        ADD     A,C
        LD      H,A
        LD      A,(HL)
;перенос
        LD      (DE),A
; атрибута
GOPZN1  POP     DE
        LD      A,C
        EX      AF,AF'
        DEC     A
        JR      NZ,LPPZN1
        POP     BC
        RET
;выход из подпрограммы

;выравнивание на сегмент:
        DS      $/256*256 +
                256-$
SCREEN  INSERT  "myface.$"
;КАРТИНКА
```

В одном из номеров ZX РЕВЮ читатели подали очень хорошую мысль насчет публикации адресов различных подпрограмм TR DOS для различных версий ПЗУ. Но это так и не решает проблемы при появлении новых версий ROM, в которых по каким-либо причинам процедуры расположены по другим адресам. Мне пришла в голову мысль:

А ПОЧЕМУ БЫ НЕ ИСКАТЬ НЕОБХОДИМЫЕ ПОДПРОГРАММЫ В ТОМ ПЗУ, КОТОРОЕ СТОИТ В КОНКРЕТНОМ КОМПЬЮТЕРЕ?!

Сказано сделано... В результате получилась система процедур именно для этих целей. Теперь можно в любом ПЗУ TR DOS

искать именно те процедуры, которые нужны для работы конкретной программы.

Ну, а если необходимая процедура все-же так и не найдена, то ее можно заменять другими, менее эффективными (как, например, чтение регистра состояния). В итоге получим как бы самонастраивающийся дисковый интерфейс.

Процедуры:

TRCOPY аналог команды LDIR, но копирует данные из ПЗУ TR-DOS. Длина копируемого блока должна быть кратна 16. Регистры, задаваемые при вызове:

HL адрес в ПЗУ (откуда копировать)

DE - адрес буфера (куда копировать)

BC длина копируемого блока

ABSEEK - поиск нужной последовательности байтов в памяти и выдача смещения найденной последовательности относительно начала проверяемого блока памяти.

На входе:

HL адрес проверяемого блока

DE адрес последовательности, которую ищем

BC длина проверяемого блока

A длина последовательности

На выходе:

Z_flag=0 (NZ) - нужная последовательность не найдена.

Z_flag=1 (Z) последовательность найдена и находится со смещением HL от начала проверяемого блока.

SEEKER - аналогична предыдущей точке входа, но при найденной последовательности выдает не адрес смещения от начала блока, а реальный адрес расположения последовательности в памяти.

Процедуры ABSEEK и SEEKER можно использовать и для других целей в своих программах (с указанием авторства, естественно).

В приведенном ниже примере в ПЗУ TR DOS ищется процедура вывода в порт BC регистра A. При вызове из бейсика командой PRINT USR 32768 программа напечатает адрес найденной процедуры или выдаст сообщение об ошибке VARIABLE NOT FOUND, если данная процедура отсутствует.

```
        ORG     #8000
;
EXAMPL
        LD      HL,#0000
        LD      DE,BUFFER
        LD      BC,#4000
        PUSH    DE
        PUSH    BC
        CALL    TRCOPY
        POP     BC
        POP     HL
        LD      DE,FORFND
        LD      A,FNDLEN
        CALL    ABSEEK
        JR      NZ,ER_EX1
        LD      B,H
        LD      C,L
        RET
ER_EX1  RST     #08
        DB      #01
;VARIABLE NOT FOUND
;
BUFFER  EQU     #C000
;
FORFND
        OUT     (C),A
        RET
```

```
;
FNDLEN  EQU     $-FORFND
;
TRCOPY
        PUSH    BC
        PUSH    HL
        PUSH    DE
        LD      C,#13
        CALL    #3D13
        POP     DE
        LD      HL,#5CDD
        LD      BC,#0010
        LDIR
        POP     HL
        LD      C,#10
        ADD     HL,BC
        POP     BC
        LD      A,C
        SUB     #10
        LD      C,A
        JR      NC,TRCOPY
        LD      A,B
        SUB     #01
        LD      B,A
        JR      NC,TRCOPY
        RET
;
ABSEEK
        PUSH    HL
        CALL    SEEKER
        POP     DE
        RET     NZ
;Fz=0 -> not found
        AND     A
        SBC     HL,DE
        XOR     A
        RET
;Fz=1 -> found at HL
;
SEEKER
        LD      (LB_SK1+1),
                A
LP_SK1  EX      AF,AF'
        LD      A,(DE)
        CPIR
        RET     NZ
;Fz=0 -> not found
        PUSH    HL
        PUSH    DE
        PUSH    BC
LP_SK2  EX      AF,AF'
        DEC     A
        JR      Z,EX_SK1
;Fz=1
        EX      AF,AF'
        INC     DE
        LD      A,(DE)
        CPI
        JR      NZ,GO_SK1
        JP      PE,LP_SK2
        EX      AF,AF'
        DEC     A
;Fz=(A=0)
EX_SK1  POP     BC
        POP     DE
        POP     HL
        DEC     HL
        RET
GO_SK1  POP     BC
        POP     DE
        POP     HL
LB_SK1  LD      A,0
        JR      LP_SK1
```

В заключение хочется добавить, что все приведенные здесь программы были написаны и отлажены на великолепном ассемблере ZX ASM 3.0 (данная статья тоже набиралась на нем) и пожелать всем дочитавшим до этого места ВСЕГО НАИЛУЧШЕГО!

****

(c) Евгений Куликаев, р.п. Муромцево,'96

Представляю на суд читателей ZX РЕВЮ три свои программы.

Первая производит очистку экрана оригинальным способом, как бы расщепляя его, задвигая при этом вверх и вниз. Сам экран располагается с адреса BUFF и переносится самой программой. После появления картинки на экране нужно нажать SPACE для продолжения. Для изменения скорости расползания необходимо

изменить шаг, т.е. число линий, на которое раздвигается экран за один проход. Все строки, которые нужно изменить, выделены в листинге. Длину я старался сделать как можно меньше, но получилась 171 байт.

```
(C) Program 'Crawl' by E.Kulikaev '96.
1          ORG   40000
2          ENT   $
3 BUFF     EQU   30000
4          LD    HL,BUFF
5          LD    DE,16384
6          LD    BC,6144
7          LDIR
8 PAU      LD    A,#7F
9          IN    A,(#FE)
10         RRA
11         JR    C,PAU
12 SCR1    LD    B,24/2
```

Сколько раз происходит скролинг Рассчитываемое по формуле 24/(шаг/8) число, должно быть целым.

```
;-------------------------
13 LSCR    PUSH  BC
14         LD    HL,22527-95
```

, Адрес 16-ой снизу линии экрана.

```
;-------------------------
15         LD    DE,22527-31
16         LD    B,192-16
```

;Кол-во скроллируемых вниз линий.

```
;-------------------------
17 SD      PUSH  DE
18         PUSH  HL
19         LD    C,32
20 SD1     LD    A,(DE)
21         XOR   (HL)
22         AND   %10101010
23         XOR   (HL)
24         LD    (DE),A
25         INC   DE
26         INC   HL
27         DEC   C
28         JR    NZ,SD1
29         POP   DE
30         CALL  L_UP
31         EX    DE,HL
32         POP   DE
33         CALL  L_UP
34         DJNZ  SD
35         LD    HL,16384
36         LD    C,16
```

Кол-во очищаемых сверху линий.

```
;-------------------------
37 SD3     PUSH  HL
38         LD    B,32
39 SD2     LD    A,(HL)
40         AND   %10101010
41         LD    (HL),A
42         INC   HL
43         DJNZ  SD2
44         POP   DE
45         CALL  L_DE
46         EX    DE,HL
47         DEC   C
48         JR    NZ,SD3
49         LD    HL,16384+6
           4
```

Адрес 16-ой линии экрана сверху.

```
;-------------------------
50         LD    DE,16384
51         LD    B,192-16
;-------------------------
```

;Кол-во скроллируемых вверх линий экрана.

```
;-------------------------
52 SU      PUSH  DE
53         PUSH  HL
54         LD    C,32
55 SU1     LD    A,(DE)
56         XOR   (HL)
57         AND   %01010101
58         XOR   (HL)
59         LD    (DE),A
60         INC   DE
61         INC   HL
62         DEC   C
63         JR    NZ,SU1
```

```
64          POP   DE
65          CALL  L_DE
66          EX    DE,HL
67          POP   DE
68          CALL  L_DE
69          DJNZ  SU
70          LD    HL,22527-95

; Адрес 16-ой линии экрана
снизу.
;--------------------------
71          LD    C,17
;--------------------------
;Кол-во очищаемых внизу ли-
ний.
;--------------------------
72 SU3      PUSH  HL
73          LD    B,32
74 SU2      LD    A,(HL)
75          AND   %01010101
76          LD    (HL),A
77          INC   HL
78          DJNZ  SU2
79          POP   DE
80          CALL  L_DE
81          EX    DE,HL
82          DEC   C
83          JR    NZ,SU3
84          POP   BC
85          DJNZ  LSCR
86          RET
87 L_DE     INC   D
88          LD    A,D
89          AND   7
90          RET   NZ
91          LD    A,E
92          ADD   A,32
93          LD    E,A
94          RET   C
95          LD    A,D
96          SUB   8
97          LD    D,A
98          RET
99 L_UP     DEC   D
100         LD    A,D
101         AND   7
102         CP    7
103         RET   NZ
104         LD    A,E
105         SUB   32
106         LD    E,A
107         RET   C
108         LD    A,D
109         ADD   A,8
110         LD    D,A
111         RET
```

Вторая программа выводит на экран картинку, размещенную с адреса BUFF. Она прорисовывается линией атрибутов, закручивающейся по часовой стрелке. Процедуру печати линии я взял из книги 'Элементарная графика',изд."ИНФОРКОМ", немного ее переделав. Все регулируемые параметры показаны в листинге программы. Ее длинна 278 байт, но эффект довольно интересный, и может применяться (с небольшими переделками) в других целях.

```
(C) Program 'Clock' by E.Kulikaev '96.
1           ORG   40000
2           ENT   $
3 BUFF      EQU   30000
4           LD    HL,22528
5           LD    DE,22529
6           LD    BC,767
7           LD    (HL),L
8           LDIR
9           LD    HL,BUFF
10          LD    DE,16384
11          LD    BC,6144
12          LDIR
13          LD    DE,0
14          LD    B,31
15 L1       CALL  LINE
16          INC   D
17          CALL  LINE1
18          DJNZ  L1
19          LD    B,23
20 L2       CALL  LINE
21          INC   E
22          CALL  LINE1
23          DJNZ  L2
24          LD    B,31
25 L3       CALL  LINE
26          DEC   D
```

```
27          CALL  LINE1
28          DJNZ  L3
29          LD    B,23
30 L4       CALL  LINE
31          DEC   E
32          CALL  LINE1
33          DJNZ  L4
34          CALL  LINE
35          RET
36 LINE     PUSH  DE
37          PUSH  BC
38          PUSH  DE
39 ZAMP     LD    BC,3000
40 PAU      DEC   BC
41          LD    A,B
42          OR    C
43          JR    NZ,PAU
44          LD    DE,#100C
45          LD    (COORD),DE
46          CALL  PLOT
47          POP   DE
48          LD    HL,(COORD)
49          EXX
50          PUSH  HL
51          EXX
52          LD    BC,#0101
53          LD    A,D
54          SUB   H
55          JP    NC,CONT_1
56          LD    B,#FF
57          NEG
58 CONT_1   LD    D,A
59          LD    A,E
60          SUB   L
61          JR    NC,CONT_2
62          LD    C,#FF
63          NEG
64 CONT_2   LD    E,A
65          LD    A,D
66          CP    E
67          JR    NC,CONT_3
68          LD    L,D
69          LD    (SIGN),BC
70          XOR   A
71          LD    B,A
72          JR    CONT_4
73 CONT_3   OR    D
74          JR    Z,FINISH
75          LD    L,E
76          LD    E,D
77          LD    (SIGN),BC
78          LD    C,0
79 CONT_4   LD    H,E
80          LD    A,E
81          RRA
82 REPEAT   ADD   A,L
83          JR    C,CONT_5
84          CP    H
85          JR    C,CONT_6
86 CONT_5   SUB   H
87          LD    D,A
88          EXX
89          LD    DE,(SIGN)
90          JR    CONT_7
91 CONT_6   LD    D,A
92          PUSH  BC
93          EXX
94          POP   DE
95 CONT_7   LD    HL,(COORD)
96          LD    A,E
97          ADD   A,L
98          LD    E,A
99          LD    A,D
100         INC   A
101         ADD   A,H
102         JR    C,CONT_8
103         JR    Z,FINISH
104 CONT_9  DEC   A
105         LD    D,A
106         LD    (COORD),DE
107         CALL  PLOT
108         EXX
109         LD    A,D
110         DEC   E
111         JR    NZ,REPEAT
112         JR    FINISH
113 CONT_8  JR    Z,CONT_9
114 FINISH  EXX
115         POP   HL
116         EXX
117         POP   BC
118         POP   DE
119         RET
120 SIGN    DEFW  0
121 COORD   DEFW  0
122 PLOT    LD    A,E
123         AND   #18
124         SRL   A
```

```
125          SRL    A
126          SRL    A
127          OR     #58
128          LD     H,A
129          LD     A,E
130          AND    7
131          OR     A
132          RRA
133          RRA
134          RRA
135          RRA
136          ADD    A,D
137          LD     L,A
138          PUSH   HL
139          LD     DE,22528
140          OR     A
141          SBC    HL,DE
142          LD     DE,BUFF +
                    6144
143          ADD    HL,DE
```

В строках 139-143 происходит расчет адреса в атрибутах выводимой картинки.

```
;--------------------------
144          LD     A,(HL)
145          POP    HL
146 ZAM      NOP
147          NOP
```

Сюда программа вставляет команду LD A,атрибут, нужную для прорисовки выводящей линии.

```
;--------------------------
148          LD     (HL),A
149          RET
150 LINE1    LD     HL,256*63+6
                    2
```

Если вы хотите изменить цвет линии, поменяйте число 63 на нужный атрибут.

```
;--------------------------
151          LD     (ZAM),HL
152          LD     HL,1
153          LD     (ZAMP+1),HL
154          CALL   LINE
155          LD     HL,0
156          LD     (ZAM),HL
157          LD     HL,3000
```

Изменить скорость вывода можно изменив величину паузы.

```
;--------------------------
158          LD     (ZAMP+1),HL
159          RET
```

Третья программа увеличивает изображение в центре экрана на весь экран. Я не стал учитывать и атрибуты, так как программа бы выросла почти в два раза, а она и так 133 байта. Программа не использует дополнительных буферов, поэтому увеличение проходит в два этапа, так как иначе затирается еще не увеличенная часть центральной области.

(C) Program 'Zoom' by E.kulikaev '96.

```
1            ORG    40000
2            ENT    $
3            LD     HL,16384
4            LD     DE,16384 +
                    201
```

В DE адрес левого верхнего угла центральной области экрана.

```
;--------------------------
5            LD     B,48
```

Высота увеличиваемой обл.

```
;--------------------------
6 L1         PUSH   DE
7            CALL   DOUB
8            CALL   L_DE
9            PUSH   DE
10           PUSH   BC
11           LD     BC,32
12           LDIR
13           POP    BC
14           POP    DE
15           CALL   L_DE
16           EX     DE,HL
17           POP    DE
18           CALL   L_DE
19           DJNZ   L1
20           LD     HL,22527- 31
```

```
21          LD      DE,22527- 214

 Адрес левого нижнего угла цен-
центральной обл.
;---------------------------
22          LD      B,48

 Высота нижней части обл.
;---------------------------
23 R1       PUSH    DE
24          CALL    DOUB
25          CALL    L_UP
26          PUSH    DE
27          PUSH    BC
28          LD      BC,32
29          LDIR
30          POP     BC
31          POP     DE
32          CALL    L_UP
33          EX      DE,HL
34          POP     DE
35          CALL    L_UP
36          DJNZ    R1
37          RET
38 DOUB     PUSH    HL
39          LD      C,16
40 L2       PUSH    BC
41          LD      A,(DE)
42          LD      BC,#0402
43 L3       PUSH    BC
44 L4       RLA
45          PUSH    AF
46          RL      (HL)
47          POP     AF
48          RL      (HL)
49          DJNZ    L4
50          INC     HL
51          POP     BC
52          DEC     C
53          JR      NZ,L3
54          INC     DE
55          POP     BC
56          DEC     C
57          JR      NZ,L2
58          POP     HL
59          PUSH    HL
60          POP     DE
61          RET
62 L_DE     INC     D
63          LD      A,D
64          AND     7
65          RET     NZ
66          LD      A,E
67          ADD     A,32
68          LD      E,A
69          RET     C
70          LD      A,D
71          SUB     8
72          LD      D,A
73          RET
74 L_UP     DEC     D
75          LD      A,D
76          AND     7
77          CP      7
78          RET     NZ
79          LD      A,E
80          SUB     32
81          LD      E,A
82          RET     C
83          LD      A,D
84          ADD     A,8
85          LD      D,A
86          RET
```

****

(c) Старков Павел, 1996.

Предлагаю вниманию читателей процедуру выгрузки/загрузки, не использующую системные переменные. С ее помощью на диск можно записать кодовый блок в виде файла. Драйвер записи/чтения сектора был взят из ZX РЕВЮ 95/4.

Чтобы записать кодовый блок, нужно вызвать процедуру SAVE. При этом в регистровой паре HL адрес начала файла в памяти, в DE длина файла, а с метки DATA должно находиться имя файла (8 байтов) и байт расширения файла.

При загрузке нужно вызвать процедуру LOAD. При этом, с метки DATA должны находиться аналогичные данные, что и при записи. Если при вызове процедуры загрузки в регистре A - 0, то

стартовый адрес и длина файла будет взята из каталога, а если A не равен 0, то в регистре HL - адрес, в DE - длина файла.

Если при выходе из SAVE или LOAD, в регистре A содержится 0, то процедура отработала успешно, если не равно 0 - ошибка.

При работе с модулем существует несколько проблем:

1. Если в записываемый блок попадают ячейки #5C00-5C01 и #5CFE-5D01, то их содержимое будет записано неверно, т.к. процедура записи во время работы использует эти ячейки, восстанавливая их при выходе. Чтобы эти ячейки записать правильно, надо сначала перебросить записываемый блок в другую область памяти и записывать его оттуда.

2. Если при записи диск будет неотформатирован или защищен от записи, программа попытается выдать сообщение Retry, Abort, Ignore, но если системные переменные уничтожены, то последствия будут печальны.

3. Если при загрузке не будет читаться какой-нибудь сектор, то программа зависнет.

В остальном программа работает устойчиво. Весь модуль вместе с переменными занимает 476 байт + буфер длиной 256 байт, расположенный следом за программой.

```
10          ORG   50000
20 SAVE     LD    (START),HL
30          LD    (LEN),DE
40          CALL  ON_DSK
50          LD    DE,8
60          CALL  LOADL
70          LD    A,(BUFF+228)
80          CP    128
90          RET   NC
100         LD    HL,(BUFF+225)
110         LD    (SECTRK),HL
120         LD    HL,(LEN)
130         PUSH  AF
140         LD    C,H
150         INC   L
160         DEC   L
170         JR    Z,S1
180         INC   C
190 S1      LD    B,0
200         LD    A,C
210         LD    (LENSEC),A
220         LD    HL,(BUFF+229)
230         POP   AF
240         AND   A
250         SBC   HL,BC
260         RET   C
270         PUSH  HL
280         LD    (NUM1+1),A
290         LD    D,B
300         LD    B,A
310         AND   15
320         ADD   A,A
330         ADD   A,A
340         ADD   A,A
350         ADD   A,A
360         LD    E,A
370         LD    HL,BUFF
380         ADD   HL,DE
390         PUSH  HL
400         LD    A,B
410         AND   240
420         RRA
430         RRA
440         RRA
450         RRA
460         LD    E,A
470         LD    (SYSTRK+1),DE
480         CALL  LOADL
490         POP   DE
500         LD    HL,DATA
510         LD    BC,16
520         LDIR
530         LD    A,(NUM1+1)
540         CPL
550         AND   15
560         PUSH  AF
570         JR    Z,S9
580         XOR   A
590         LD    (DE),A
```

```
600 S9        LD    DE,(SECTRK)
610           LD    HL,(START)
620           LD    A,(LENSEC)
630           LD    B,A
640           LD    C,6
650           CALL  DRIVER
660           POP   AF
670           PUSH  DE
680           PUSH  AF
690           LD    DE,(SYSTRK+1)
700           CALL  SAVEL
710           POP   AF
720           JR    NZ,S3
730 SYSTRK    LD    DE,0
740           INC   E
750           CALL  LOADL
760           XOR   A
770           LD    (BUFF),A
780           DEC   E
790           CALL  SAVEL
800 S3        LD    DE,8
810           CALL  LOADL
820           POP   HL
830           LD    (BUFF+225),HL
840           POP   HL
850           LD    (BUFF+229),HL
860 NUM1      LD    A,0
870           INC   A
880           LD    (BUFF+228),A
890           LD    DE,8
900           CALL  SAVEL
910           XOR   A
920           RET
930 LOAD      LD    (LENSEC),A
940           LD    (START),HL
950           LD    (LEN),DE
960           CALL  ON_DSK
970           LD    DE,8
980           CALL  LOADL
990           LD    A,(BUFF+228)
1000          LD    (NUM1+1),A
1010          LD    DE,0
1020 LOAD0    CALL  LOADL
1030          LD    DE,BUFF
1040          LD    IX,NUM1+1
1050          LD    B,16
1060 LOAD1    PUSH  DE
1070          LD    HL,DATA
1080          LD    C,9
1090 LOAD2    LD    A,(DE)
1100          CP    (HL)
1110          JR    Z,LOAD3
1120          POP   DE
1130          DEC   (IX+0)
1140          RET   Z
1150          LD    HL,16
1160          ADD   HL,DE
1170          EX    DE,HL
1180          DJNZ  LOAD1
1190 IS_TRK   LD    DE,0
1200          JR    LOAD0
1210 LOAD3    INC   DE
1220          INC   HL
1230          DEC   C
1240          JR    NZ,LOAD2
1250          POP   IX
1260          LD    HL,(START)
1270          LD    BC,(LEN)
1280          LD    A,(LENSEC)
1290          AND   A
1300          JR    NZ,LOAD6
1310          LD    L,(IX+9)
1320          LD    H,(IX+10)
1330          LD    C,(IX+11)
1340          LD    B,(IX+12)
1350 LOAD6    LD    E,(IX+14)
1360          LD    D,(IX+15)
1370 LOAD5    INC   B
1380          DEC   B
1390          JR    Z,LOAD8
1400          PUSH  BC
1410 LOAD7    LD    C,5
1420          CALL  DRIVER
1430          POP   BC
1440          INC   C
1450          DEC   C
1460          RET   Z
1470          LD    DE,(IS_TRK+1)
1480 LOAD8    PUSH  HL
1490          CALL  LOADL
1500          POP   DE
1510          LD    BC,(LEN)
1520          LD    B,0
1530          LD    HL,BUFF
1540          LDIR
1550          LD    A,B
1560          RET
1570 LOADL    LD    C,5
```

```
1580          JR    CONT2
1590 SAVEL    LD    C,6
1600 CONT2    LD    B,1
1610          LD    HL,BUFF
1620 DRIVER   BIT   0,C
1630          JR    NZ,M1
1640          EXX
1650          PUSH  HL
1660          LD    HL,(#5C00)
1670          LD    DE,(#5CFE)
1680          LD    BC,(#5D00)
1690          EXX
1700          XOR   A
1710          LD    (#5C00),A
1720 M1       DI
1730 L1       PUSH  DE
1740          PUSH  BC
1750          LD    A,D
1760          OR    A
1770          RRA
1780          LD    C,A
1790          LD    A,#3C
1800          JR    NC,L2
1810          LD    A,#2C
1820 L2       LD    IX,#2F4D
1830          CALL  DOS
1840          POP   BC
1850          POP   DE
1860 L3       BIT   0,C
1870          JR    NZ,M2
1880          LD    A,E
1890          LD    (#5CFF),A
1900          LD    (#5D00),HL
1910 M2       PUSH  HL
1920          PUSH  DE
1930          PUSH  BC
1940          LD    IX,#2F1B
1950          BIT   0,C
1960          JR    NZ,M3
1970          LD    IX,#3F0A
1980 M3       CALL  DOS
1990          POP   BC
2000          POP   DE
2010          POP   HL
2020          INC   H
2030          INC   E
2040          BIT   4,E
2050          JR    NZ,L5
2060          BIT   0,C
2070          JR    NZ,M4
2080          LD    A,E
2090          LD    (#5CFF),A
2100 M4       DJNZ  L3
2110          JR    EXIT
2120 L5       LD    E,0
2130          INC   D
2140          DJNZ  L1
2150 EXIT     LD    (IS_TRK+1),DE
2160          BIT   0,C
2170          RET   NZ
2180          EXX
2190          LD    (#5D00),BC
2200          LD    (#5CFE),DE
2210          LD    (#5C00),HL
2220          POP   HL
2230          EXX
2240          RET
2250 ON_DSK   DI
2260          LD    IX,#2F65
2270 DOS      PUSH  IX
2280          JP    #3D2F
2290 DATA     DEFM  "FileNameC"
2300 START    DEFW  0
2310 LEN      DEFW  0
2320 LENSEC   DEFB  0
2330 SECTRK   DEFW  0
2340 BUFF     DEFS  256
```

****

(c) Вячеслав Мишуков, г.Северодвинск, 1996.

Процедура печати 42 символов в строке.

В ZX-РЕВЮ N5/95 на стр. 29 опубликована процедура печати 42 символов в строке (автор Дмитрий Сергеев).

Приведенная процедура обладает недостатком: выключаются пикселы, которые при работе процедуры не должны быть задействованы. Поскольку размер символа по горизонтали 6 точек, то справа от него в знакоместе 8x8 могут остаться 2,4 или 6 незанятых пиксельных столбцов, которые выключаются процедурой. Кроме того, если символ полностью по-

местился в знакоместе 8x8 (прижат к левому или правому его краю), то следующее знакоместо 8x8 также очищается.

Во всем виновата процедура SCRL, расположенная в строках 137-167. Процедура печати будет работать корректно, если переписать подпрограмму SCRL в соответствии с листингом. Изменения коснулись фрагмента, формирующего изображение символа.

Восемь строк символа помещаются в экранную область памяти таким образом, чтобы изображение справа и слева от него не пострадало. Это достигается последовательным использованием операций XOR, AND, XOR. Такой метод подробно описан в книге ИНФОРКОМА "Элементарная графика" на стр. 94.

Применение операций XOR, AND, OR позволяет заменить отдельные биты в байте A на соответствующие биты байта B. Заменены будут только те биты, которые замаскированы в операции AND нулями.

В измененном варианте подпрограммы, параллельно со сдвигом байта шаблона символа, сдвигается маска для операции AND.

После внесения изменений длина процедуры увеличится. В какой-то степени увеличение объема можно скомпенсировать, удалив в исходном листинге строки 113, 114, 120, 121, 124, 125, 126, раскрыв подпрограмму SCRL. Но время, затрачиваемое на вывод одного символа все же возрастет, поскольку увеличился главный цикл процедуры.

Листинг подпрограммы SCRL с внесенными изменениями:

```
SCRL   LD    A,(BC)
       PUSH  BC
       PUSH  AF
       LD    DE,#FF03
       XOR   A
       LD    C,A
       LD    A,(NUM)
       LD    B,A
       POP   AF
SHIFT  AND   A
       RLA
       RL    C
       SCF
       RL    E
       RL    D
       DJNZ  SHIFT
       LD    B,A
       LD    A,(HL)
       XOR   C
       AND   D
       XOR   C
       LD    (HL),A
       INC   HL
       LD    A,(HL)
       XOR   B
       AND   E
       XOR   B
       LD    (HL),A
       DEC   HL
       INC   H
       POP   BC
       INC   BC
       LD    A,(ST)
       DEC   A
       LD    (ST),A
       JR    NZ,SCRL
       LD    A,#08
       LD    (ST),A
       RET
```

РЕТРО

Итак, мы начинаем новую рубрику "РЕТРО" В нее мы будем включать лучшие материалы из выпусков ZX РЕВЮ и других изданий прошлых лет, тиражи которых уже закончились и полное переиздание которых нецелесообразно в силу как экономических, так и чисто творческих причин.

Основными критериями отбора материалов для новой рубрики, кроме качества и актуальности, нами выбрана еще и образовательная направленность. Надеемся, что эта рубрика понравится нашим новым читателям, а старые не будут в обиде, т.к. ее объем не будет слишком большим.

Работа, представленная ниже, была опубликована в ZX РЕВЮ 1992 года и вызвала в свое время самые восторженные отклики читателей.

Дж.Хардман, Э.Хьюзон Перевод Н.А.Тихоновой

40 ЛУЧШИХ ПРОЦЕДУР

ОТ ПЕРЕВОДЧИКА

Данная книга является сокращенным переводом книги "40 Best Machine Code Routines For The ZX Spectrum With Explanatory Text", J.Hardman & A.Hewson, содержащей в себе набор программ в машинных кодах с весьма подробными разъяснениями принципов их работы.

РАЗДЕЛ А

1. ВВЕДЕНИЕ

Цель этой книги обеспечить как начинающего, так и опытного пользователя компьютера ZX SPECTRUM полезными, интересными и развлекательными программами в машинных кодах.

Книга имеет два раздела. Раздел А описывает те особенности ZX SPECTRUM'а, которые важны при программировании в машинных кодах, некоторые процедуры системного ПЗУ, а также структуру машинного языка.

Раздел В представляет сами программы. Они представлены в стандартном формате, который детально описан в начале раздела. Программы полностью закончены, т.е. они могут быть загружены и использованы индивидуально, без обращений к другим программам.

Предлагаемые программы могут быть загружены с помощью простого загрузчика машинных кодов (MC LOADER - маш. ЗАГРУЗЧИК), описанного в начале раздела В.

ОБЩИЕ СВЕДЕНИЯ О БЕЙСИКЕ И МАШИННЫХ КОДАХ

Микропроцессор Z80A, на базе которого сделан ZX SPECTRUM, не понимает непосредственно слова Бейсика, такие как PRINT, IF, TAB и т.д. Вместо этого он выполняет приказы специального языка - своего внутреннего машинного кода. Процедуры ПЗУ, которые придают ZX SPECTRUM'у его индивидуальность, написаны на этом специальном языке и состоят из большого количества стандартных подпрограмм для ввода-вывода листинга, интерпретирования и выполнения команд BASIC'а и др.

Например, стандартные подпрограммы говорят процессору Z80A "ЧТО ДЕЛАТЬ, ЕСЛИ... Если, например, команда BASIC'а - слово PRINT, то что делать, если следующий элемент - имя переменной; или что делать, если следующий элемент - запятая и т.д.

Машинный код состоит из последовательности положительных целых чисел (от 0 до 255), которые диктуют действия для Z80A. Хотя машина использует двоичную форму представления чисел, нет необходимости для человека изучать команды в такой форме. Мы будем использовать десятичную форму, которая обрабатывается МС ЗАГРУЗЧИК'ом из раздела В.

Однако даже однообразную строку десятичных чисел трудно интерпретировать, и поэтому десятичные числа обычно преобразовывают в специальный язык (ассемблер), который представляет собой определенные аббревиатуры. Язык ассемблера называется так потому, что специальная программа, называемая АССЕМБЛЕРом, используется для обработки ("сбора" или ассемблирования) команд в машинных кодах при написании (формировании) программы.

Требуется только одно число, чтобы точно определить простую команду Z80A. Например, команда СКОПИРОВАТЬ содержимое регистра С в регистр D - это десятичное число 81 (термин "регистр" более детально описан в главе 3, а пока достаточно если Вы будете воспринимать С и D как переменные BASIC'а). Для таких команд есть точное соответствие между десятичным числом и командой: 81, например, записывается на языке АССЕМБЛЕРа как LD D,C ("LD", кстати, сокращение слова "load" - загрузить). Многие команды ассемблера состоят из подобных простых аббревиатур. По этой причине они часто называются мнемониками. Более сложные команды требуют 2, 3 или 4 числа, но все равно для представления их используется одна команда ассемб-

лера. Таблица 1.1 показывает список нескольких чисел, их мнемоник и краткое объяснение действия Z80A.

Таблица 1.1. Некоторые примеры машинных команд Z80A

Ссылка	Десятичное число	Мнемоника	Комментарий
(a)	81	LD D,C	Загрузить в D содержимое C
(b)	14 27	LD C,27	Поместить число 27 в C
(c)	14 13	LD C,13	Поместить число 13 в C
(d)	33 27 52	LD HL,13339	Поместить 13339 в пару регистров HL. Обратите внимание: 27 + 256 * 52 = 13339; 27 поместить в L; 52 поместить в H
(e)	221 33 27 52	LD IX,13339	Поместить 13339 в пару регистров IX

Строка (a) таблицы пример LD D,C - рассматривался выше. Строки (b) и (c) показывают, как положительное число может быть загружено в регистр (используются два числа: первое определяет действие, которое должно быть выполнено, а второе определяет число, которое должно быть загружено). Строка (d) показывает, как большое целое число может быть загружено в два регистра (H и L) вместе. Здесь второе и третье числа определяют, какие числа должны быть загружены. Последний пример в строке (e) иллюстрирует четырехбайтовый код для загрузки большого целого числа в пару регистров IX. Обратите внимание, что три из четырех чисел такие же, как в строке (d). А дополнительное первое число определяет пару регистров IX вместо HL.

Структура машинного языка объясняется более подробно в главе 3, а полный список мнемоник ассемблера Z80A можно найти в литературе по микропроцессорной технике и программированию.

Итак, наиболее насущный вопрос: ЗАЧЕМ ИСПОЛЬЗУЮТ МАШИННЫЙ КОД?

На некоторых компьютерах это делается потому, что задачи, которые пользователь желает выполнить, слишком медленно выполняются. В этом отношении ZX SPECTRUM не исключение. Рассмотрим, например, проблему сохранения полного экрана в RAM (ОЗУ) или копирования его обратно на экран с целью создания эффекта мультипликации.

Файл изображения и цветовые атрибуты занимают 6912 байт. Следующая программа BASIC'а сохранит отображение экрана, но это займет много времени - около 70 секунд:

```
5 CLEAR 58623
10 FOR 1=0 T0 6911
20 POKE 58624+1, PEEK (16384+1)
30 NEXT 1
```

Причина такой медленной работы в том, что ZX SPECTRUM тратит больше всего времени на декодирование команд BASIC'а перед их выполнением. Некоторое количество времени также тратится на преобразование чисел в двухбайтную форму (которую понимает Z80A) из десятичных чисел в пятибайтной форме (с которой оперирует BASIC), а также на выполнение пятибайтовой арифметики.

Так, в нашем примере по переброске экрана должны быть выполнены следующий шаги:

1. Прибавить 1 к 16384.
2. Преобразовать результат в форму двух байтов.
3. Восстановить содержимое адреса PEEK.
4. Прибавить 1 к 58624.
5. Преобразовать результат в форму 2-х байтов.
6. Сохранить полученное (PEEK) значение по заданному адресу (POKE).
7. Прибавить 1 к значению 1 и сохранить результат.
8. Вычесть 1 из 6911. Если результат положительный, то идти на пункт 1.

Во время прохождения цикла ZX SPECTRUM должен декодировать каждую команду снова, так как в данном случае память не используется для сохранения последовательности предыдущих действий. Легко увидеть, что компьютер тратит более 99% времени на подготовку к выполнению задачи, а не на выполнение самой задачи. Аналогичная программа в машинных кодах для сохранения экрана выполняется практически мгновенно. Пример такой программы дан в разделе В.

2. ВНУТРЕННЯЯ СТРУКТУРА ZX SPECTRUM

Компьютер это машина, которая способна запомнить последовательность команд и затем выполнить их. Конечно, чтобы сделать так, требуется память, в которой команды могут быть сохранены. ZX SPECTRUM имеет два вида памяти.

Первый тип - ПЗУ (ROM), в котором содержится фиксированная последовательность инструкций, введенных в машину ее изготовителем.

Второй тип - ОЗУ (RAM). RAM - это "блокнот для записей" компьютера.Когда компьютер выполняет задачу, он непрерывно просматривает, что находится в RAM ("чтение" из памяти) и обновляет содержимое RAM ("запись" в память). ZX SPECTRUM не использует свой "блокнот" для записей случайно. Различные части RAM используются для хранения различных видов информации. Например, программа BASIC'a, введенная пользователем, хранится в одной части RAM, в то время как переменные, используемые этой программой, хранятся в другом месте. Размер "блокнота" для записей ограничен и потому машина точна при распределении пространства для информации, которую она хранит. Свободное пространство собрано в одном месте и если пользователь хочет добавить строку в свою программу, информация в RAM должна быть "перетасована" по всей длине, используя некоторую свободную область для вставки дополнительной строки.

В значительной степени эта глава посвящена объяснению организации RAM ZX SPECTRUM'a, так как многие программы радела B предназначены для манипуляций с RAM. Глава содержит в себе описание дисплейного файла, атрибутов, буфера принтера, системных переменных, программной области и области программых переменных. В конце раздела описываются стандартные подпрограммы из ROM, к которым ссылаются программы в разделе B.

КАРТА ПАМЯТИ

RAM имеет 49152 ячейки памяти. Каждая ячейка может хранить одиночное целое число от 0 до 255 включительно и задается адресом, который является положительным целым числом от 0 до 65535. Адреса от 0 до 16383 зафиксированы для постоянной памяти ROM. Первый адрес RAM (ОЗУ) 16384. Таблица 2.1 упрощенная карта памяти ZX-SPECTRUM'a, которая показывает как используется RAM с адреса 16384.

Таблица 2.1. Карта памяти. Указатель стека SP хранится не в RAM, а в SP-регистре микропроцессора Z80A.

Стартовый адрес или имя системной переменной	Ячейка системной переменной	Содержание памяти
16384	-	Дисплейный файл

22528	-	Атрибуты
23296	-	Буфер принтера
23552	-	Системные переменные
23734	-	Карта микродрайва
CHANS	23631	Область информации о каналах
PROG	23635	Адрес начала программы на BASIC'е
VARS	23627	Адрес начала области программных переменных
E-LINE	23641	Адрес области редактирования
WORKSP	23649	Буфер INPUT
STKBOT	23651	Стек калькулятора
STKEND	23653	Свободная область
SP		Машинный стек и стек GOSUB
RAMTOP	23730	Пользовательские подпрограммы в машинном коде
UDG	23675	Графика пользователя
P-AMT	23732	Физическая вершина ОЗУ

Дисплейный файл, который хранит отображаемую на экране информацию, занимает ячейки от 16384 до 22527. Атрибуты, которые определяют цвет, яркость и т.д. для каждого знакоместа экрана, следуют непосредственно дальше в ячейках 22528 - 23295.

Первые 5 адресов в колонке 1 таблицы 2.1 являются фиксированными, т.к. дисплейный файл, атрибуты и т.д. занимают фиксированное положение. Пятая область назначена для карты памяти микродрайва. Это небольшое периферийное устройство представляет собой нечто среднее между магнитофоном и дисководом. Грубо говоря это дешевая альтернатива дисководу. Высокая скорость его работы достигается за счет того, что с одной стороны лента движется с очень большой скоростью, а с другой за счет организации прямого доступа к файлам, как на диске, а не последовательного, как на магнитофонной кассете. Вот для того, чтобы иметь в некоем месте хранилище с данными о том, что записано на каждом секторе микродрайва, в памяти компьютера и выделена область карты памяти микродрайва. Если микродрайв (а точнее INTERFACE-1, через который он подключается) подсоединен к ZX SPECTRUM'у, эта область содержит информацию

о его секторах. Если же не подсоединен, эта область не нужна, и в этом случае шестая область (информация о каналах) размещена непосредственно за четвертой областью, системными переменными, т.е. стартовый адрес области информации о каналах и всех последующих областей не фиксирован, а может "плавать" вверх-вниз в RAM.

ZX SPECTRUM хранит стартовый адрес всех этих областей в системных переменных. Область системных переменных находится перед картой микродрайва в ячейках 23552 23733 включительно. Эти адреса являются все время строго фиксированными.

Адреса ячеек, которые хранят стартовые адреса всех "плавающих" областей, даны в колонке 2 таблицы 2.1. Адрес области начала программы BASIC'а, например, хранится в ячейках 23635 и 23636 в области системных переменных.

Примечание: Ссылка к системной переменной с помощью адреса, по которому она находится, довольно неудобна. По этой причине в таблице 2.1 в 1-й колонке даны условные имена системных переменных. Эти имена удобны только для пользователей, в то время как ZX SPECTRUM'ом они, естественно, не распознаются. Например, введенная строка: PRINT PROG даст сообщение об ошибке "2: Variable not found" ("Переменная не найдена").

PEEK и POKE

Карта памяти это ключ для понимания того, как компьютером используется RAM-память. Для непосредственного управления RAM используются ключевые слова BASIC'а PEEK и POKE, которые позволяют просмотреть и изменить содержание любой ячейки памяти.

PEEK - функция вида: PEEK адрес

Адрес это целое положительное число от 0 до 65535 или арифметическое выражение, которое выполняясь дает положительное число. Важно заключить арифметическое выражение в скобки, т.к.

PEEK 16384 + 2

интерпретируется как (2 + результат PEEK 16384), тогда как

PEEK (16384 + 2)

интерпретируется как

PEEK 16386

Значение, выдаваемое функцией PEEK это число, хранящееся по данному адресу в текущий момент. Оно всегда будет положительным от 0 до 255.

Ваше объяснялось, что системная переменная PROG хранится по адресу 23635, но значение PROG (т.е. адрес в RAM) всегда больше числа 255. Следовательно, для его хранения необходимы две смежных ячейки с адресами 23635 и 23636. Значение PROG может быть получено с помощью командной строки:

PRINT "PROG="; PEEK 23635 + 256*PEEK 23636

Все адреса хранятся в 2-х смежных ячейках в такой форме и могут быть получены вводом:

PRINT PEEK 1-я ячейка + 256*PEEK 2-я ячейка

Например, если ZX SPECTRUM используется без подсоединенного микродрайва, область карты микродрайва не будет существовать и информация о каналах будет располагаться непосредственно после области системных переменных. Т.о. системная переменная CHANS должна быть такой же, как стартовый адрес карты микродрайва, когда он существует, т.е. 23734. CHANS хранится в 23631 и 23632 и, следовательно, после ввода:

PRINT PEEK 23631 + 256*PEEK 23632

будет получено значение 23734.

Функция PEEK может быть использована также для просмотра содержимого любой из ячеек ПЗУ Это очень полезно. Просмотр любой ячейки не приводит к разрушению или искажению программы или переменных. Иногда результаты PEEK могут быть обманчивыми, т.к. содержимое ячейки, которая просматривалась, может изменяться в течение или непосредственно после выполнения команды просмотра.

Например, если просматривается содержимое ячеек, которые связаны с левым верхним углом экрана дисплей, то результаты, распечатанные в верхнем левом углу экрана будут уже устаревшими, т.к. в момент вывода изображения мы уже изменили эти ячейки.

Команда POKE более рискованная, чем функция PEEK, т.к. задавая ее, пользователь вмешивается в функционирование компьютера. Использование этой команды может быть причиной сбоя машины или ее останова.

Формат команды:

POKE адрес, число

Адрес - это положительное целое число от 0 до 65535 включительно или арифметическое выражение, которое дает такое

число. В этом случае нет необходимости заключать арифметическое выражение в скобки, т.к. POKE это команда, а не функция (хотя есть негласная заповедь, что лишними скобками программу не испортишь). Число помещаемое в ячейку, может быть от 0 до 255 включительно.

ДИСПЛЕЙНЫЙ ФАЙЛ

Обычно дисплей содержит 24 строки по 32 символа. Дисплейный файл занимает ячейки от 16384 до 22527, т.е. 6144 ячейки в общей сложности. Следовательно, количество ячеек, используемое для символа:

6144/(24*32)=8

Эти 8 ячеек формируют изображение символа на экране, называемое знакоместом.

Наиболее легкий путь получения общего впечатления о том, как организован дисплейный файл - это печать (PRINT) картинки на экране, сброс (SAVE) экрана на ленту, очистка экрана (CLS) и загрузка (LOAD) картинки экрана вновь. Программа P2.1 сохраняет (SAVE) и загружает (LOAD) экран, используя графический символ на клавише 5 для создания оригинальной картинки.

Программа P2.1

```
100 FOR 1=0 TO 703
110 PRINT " ";   REM Символ, находящийся на клавише
"5" в G- режиме
120 NEXT 1
130 SAVE "Picture" SCREEN$
140 CLS
150 INPUT "Перемотайте ленту и включите воспроизве-
дение";z$
160 LOAD "Picture" SCREEN$
```

Когда картинка загружается с ленты, видно, что дисплей разделен на три зоны по 8 символьных строк в каждой, а каждая строка разделяется на восемь пиксельных линий. ZX-SPECTRUM загружает сначала верхние пиксельные линии для первых восьми строк, затем следующие пиксельные линии тех же восьми строк и т.д. Таким же образом формируются средняя и нижняя части дисплея.

Другой путь понимания формирования дисплея это рассмотреть. где хранятся 8 байтов, которые используются для формирования символа в верхнем левом углу экрана. Первый байт

формирует самую верхнюю 1/8 часть символа и размещается в начале дисплейного файла по адресу 16384.

Команде POKE 16384,0 очистит верхнюю линию пикселей самого верхнего левого знакоместа, в то время, как POKE 16384,255 закрасит всю эту линию. При помещении в эту ячейку числа от 0 до 255 мы получим в этом месте экрана различные штрихи. Вторая сверху линия в первом знакоместе на экране не сформирована числом, хранящимся в ячейке 16385 эта ячейка используется для верхней линии пикселей в соседнем символе. Вторая линия сверху в первом знакоместе формируется числом, хранящимся в ячейке

16384+32*8=16640

Подобным же образом вычисляются адреса оставшихся шести линий этого знакоместа.

Следовательно, образ символа в верхнем левом углу экрана определяется содержимым адресов 16384, 16640, 16896, 17152, 17408, 17664, 17920, 18176.

Программа P2.2 позволяет Вам экспериментировать, помещая различные числа в эти восемь ячеек. Каждая ячейка в дисплейном файле определяет восемь пикселей на экране. При этом число, которое хранится в данной ячейке, преобразуется в двоичную форму и затем устанавливаются восемь пикселей соответственно двоичным цифрам. Например, 240 после преобразования в двоичное число дает: 11110000

Следовательно, если ячейка содержит число 240, четыре из восьми пикселей, соответствующие единицам, будут высвечены, а оставшиеся - нет.

Программа P2.2. Программа для создания символа в верхнем левом углу экрана

```
10 REM Подпрограмма установки символа в верхнем углу экрана
20 INPUT "Символ состоит из восьми байтов, каждый из которых - число в диапазоне от 0 до 255. Введите номер байта (от 0 до 7)";n
30 IF n<0 OR n>7 OR n<>INT n THEN BEEP 0.2, 24: GO TO 20
40 INPUT "Ввести содержимое байта";m
50 IF m<0 OR m>255 OR m<>INT m THEN BEEP 0.2, 24: GO TO 40
60 POKE 16384+8*32*n,m
```

Таким образом, дисплейный файл состоит из 6144 ячеек памяти по 8 ячеек для одного знакоместа. Каждая ячейка определяет горизонтальную полосу из восьми пикселей. Ячейки, относящиеся к данному знакоместу, не расположены последовательно, одна за другой.

АТРИБУТЫ

Содержание дисплейного файла определяет, какие пиксели высвечиваются на экране. Цвет фона (PAPER), символа (INK),яркость (BRIGHT) и мерцание (FLASH) определяются с помощью атрибутов. Область атрибутов занимает ячейки 22528 23295 - по одной ячейке для каждого из 768 знакомест.

Соответствие между содержимым ячеек памяти файла атрибутов и самими атрибутами следующее:

Значение атрибута = 128*FLASH + 64*BRIGHT + 8*PAPER + INK

FLASH и BRIGHT принимают значение 1, если соответствующее условие установлено, а PAPER и INK принимают значение требуемого цвета, как показано на клавиатуре (красный, например, 2).

Программа P2.3 декодирует атрибуты, т.е. данное значение атрибута распечатывает с соответствующим цветом PAPER и INK.

Программа P2.3. Программа декодирования атрибута

```
 10 REM Декодер атрибутов
 20 DATA "Black", "Blue", "Red", "Magenta",
"Green","Cyan","Yellow", "White", "Bright", "Flash"
 30 DIM C$(8,7)
 40 FOR 1=1 TO 8
 50 READ C$(1)
 60 NEXT 1
100 REM Декодер атрибутов
110 INPUT "Ввести число от 0 до 255. Эта программа де-
кодирования интерпретирует его в файл атрибутов";n
120 IF n<0 OR n>255 OR n<> INT n THEN BEEP .2,24:GO
TO 110
200 PRINT "Цвет символа - "; C$(1+n-8*INT(n/8))
210 PRINT "Цвет фона - "; C$(1+INT(n/8)-8*INT(n/64))
220 IF INT (n/64)=1 OR INT (n/64)=3 THEN PRINT "Символ
  BRIGHT"
```

```
230 IF n>127 THEN PRINT "Символ будет мерцать
(FLASH)"
300 PRINT AT 6,0; "||||||||||||||||||||||||||||||"
310 FOR 1=22720 TO 22751
320 POKE 1,n
330 NEXT 1
500 INPUT "Для повторения - ENTER";Z$
510 CLS
520 GO TO 110
```

БУФЕР ПРИНТЕРА

256 ячеек ОЗУ, следующие за областью атрибутов, используются как буфер для хранения строки символов, которая должна быть передана на принтер.

Многие программы в разделе В будут использовать буфер принтера для передачи данных BASIC'а или от клавиатуры в подпрограммы. Буфер подходит для этой цели, т.к. его ячейки фиксированы и маловероятно, что пользователь пожелает его использовать для других целей.

Единственно важное ограничение в этом случае не использовать команды BASIC'а, которые требуют работы с принтером - LLIST, LPRINT, COPY.

Есть еще и ограничение для владельцев 128-килобайтных машин. У них нет буфера принтера. Дело в том, что буфер принтера предназначался в оригинальной модели для поддержки недорогого специализированного узкопечатного ZX-принтера. В фирменной модели 128-килобайтных машин есть порт подключения полноценного матричного принтера, обладающего собственным буфером, и необходимость в этом буфере отпала. Зато в этих моделях необходимо больше системных переменных и под них отдали область буфера ZX-принтера. Теперь, если в режиме 128К пользователь что-либо зашлет в эту область, то нарушив системные переменные он выведет программу из строя.

ОБЛАСТЬ ПРОГРАММ НА BASIC'е

Обычно эта область начинается с адреса 23755, и на нее указывает содержимое системной переменной PROG (23635,23636), но есть и исключения за счет некоторых видов периферийных устройств.

Если, например, к компьютеру подсоединен микродрайв, то начало этой области сдвигается. В этом самом общем случае начало области программ на BASIC'е и определяют с помощью сис-

темной переменной PROG. Ниже предполагается, что такая периферия не подсоединена.

Программа P2.4 распечатывает содержимое 18-ти ячеек в начале программной области, как показано на рис.2.1. В этих 18 ячейках хранится первая строка данной программы.

Программа P2.4 Программа для просмотра содержимого первых 18-ти ячеек в программной области

```
10 REM Peek program
20 FOR 1=23755 TO 23772
30 PRINT 1,PEEK 1
40 NEXT 1
```

23755	0
23756	10
23757	14
23758	0
23759	234
23760	80
23761	101
23762	101
23763	107
23764	32
23765	112
23766	114
23767	111
23768	103
23769	114
23770	97
23771	109
23772	13

Рис. 2.1. Формат, в которой строка 10 REM peek program хранится в программной области

Номер строки (10) хранится в первых двух ячейках в форме:

Номер строки = 256*PEEK первый адрес + PEEK второй адрес.

Следующие две ячейки 23757 и 23758 хранят длину оставшейся части строки, начинающейся в ячейке 23759. В нашем случае:

14+256*0=14

Таким образом, следующая строка начинается с ячейки:

23759+14=23773

Ячейка 23759 хранит в себе число 234, которое является кодом ключевого слова (токена) REM. Следующие 12 ячеек хранят коды символов одиннадцати букв и пробела, составляющих фразу Peek program. Последняя ячейка хранит число 13, которое является кодом для ENTER, определяющим конец строки.

Таблица 2.2 показывает метод кодирования программы в программной области.

Таблица 2.2. Этот метод используется для кодирования строк программы

Ячейки	Содержимое
1 и 2	номер строки
3 и 4	длина строки, исключая первые 4 ячейки
5	код команды
Конец	символ ENTER, число 13

Момент, который опущен в таблице - это описание метода хранения числовых значений, имеющих место в программе. Этот метод может быть исследован на примере строки:

10 LET a=1443

Введите ее в программу P2.4. На рис. 2.2 показан результат работы программы в этом случае.

23755	0
23756	10
23757	14
23758	0
23759	241
23760	97
23761	61
23762	49
23763	52
23764	52
23765	51
23766	14
23767	0
23768	0
23769	163

23770	5
23771	0
23772	13

Рис. 2.2. Формат, в которой строка 10 LET a=1443 хранится в программной области

Ячейки 23755-23758 такие же, как в прошлом примере. затем следуют коды для LET, a, = и четыре кода цифр, которые вместе формируют число 1443. Следующий элемент, находящийся в ячейке 23766, - это код 14. Этот код указывает, что следующие 5 ячеек хранят число в специальном пятибайтном формате. Линия заканчивается в ячейке 23772 вводом символа ENTER, как и ранее.

* * *

Примечание ИНФОРКОМа

Получается так, что в одной строке число как бы записано дважды. Первый раз своими символами, а второй раз в пятибайтной форме после кода 14. Зачем это нужно?

Дело в том, что первое представление используется для того, чтобы Бейсик-интерпретатор знал, что Вам показать на экране, а второе - для расчетов во внутреннем калькуляторе компьютера. (О калькуляторе читайте в нашей книге "Программирование в машинных кодах").

Самое интересное, что они могут и не совпадать. В этом случае на экране Вы будете видеть одно, а программа будет обрабатывать совсем другое число. И этим нередко пользуются в защите программ. Например, Вы видите на экране

LET a=1257: GO TO a

и пытаетесь проследить работу программы, а на самом деле там было записано нечто совсем другое и компьютер делает переход не к строке 1257, а туда, куда надо.

Желающие могут посмотреть загрузчик программы BOMBJACK. В череде относительно несложных вывертов этот там стоит одним из первых, но если на него "клюнуть", атака на программу никак не получится.

* *

ЦИФРОВОЙ ПЯТИБАЙТНЫЙ ФОРМАТ

Пять ячеек памяти используется для хранения чисел в программе на BASIC'е (исключая номера строк). Целые числа в диапазоне от -65535 до +65535 хранятся таким же образом, как в

формате Z80A. Для этих чисел первые две ячейки и последняя содержат 0, а третья и четвертая хранят число в двухбайтной форме:

число = PEEK 3-я ячейка + 256*PEEK 4-я ячейка

Таким образом 16533 хранится в пяти ячейках как

0 0 169 64 0

потому, что

169+256*64=16553

Дробные числа хранятся в формате с плавающей запятой таким образом: экспонента в первой ячейке, а мантисса в следующих четырех ячейках, то есть:

число = мантисса * 2^экспонента

Первая ячейка мантиссы используется также для определения знака числа. Если ячейка содержит значение в пределах от 0 до 127, число положительное, если нет - отрицательное.

Программа P2.5 может быть использована для восстановления дробного числа из составляющих его пяти компонентов.

Программа P2.5. Эта программа восстанавливает дробное число

```
10 PRINT "Ввести экспоненту и четыре байта
мантиссы. Все числа должны находиться между
0 и 255 включительно."
20 INPUT e,a,b,c,d
30 PRINT ,, " Exponent= ";e
40 PRINT "Mantissa= ";a,,b,,c,,d
50 PRINT ,, "The nember= ";(2*(a<128)-1)*2^(e-
160) * (((256*(a+128*(a<128))+b)*256+c)*256+d)
```

ОБЛАСТЬ ПЕРЕМЕННЫХ

Область переменных начинается в ячейке, адрес которой хранится в системной переменной VARS (ячейка 23627). Как бы ни была объявлена новая переменная, т.е. в программе или непосредственным вводом с клавиатуры, для нее резервируется соответствующее количество свободного пространства в этой области.

Все имена переменных начинаются с буквы. Различий между верхним и нижним регистром нет. Эти ограничения позволяют ZX-SPECTRUM'у манипулировать с кодом первого символа каждой переменной и таким образом он может различить шесть типов переменных просмотром диапазона, в котором находится код.

Все цифровые переменные с односимвольными именами, например, имеют коды в пределах от 97 до 122; буква a 97; b

98; c 99 и т.д. Подобным же образом цифровые массивы имеют коды в пределах 129 - 153, т.е. a - 129; b - 130; c - 131 и т.д. Диапазоны кодов представлены в таблице 2.3. Длина каждого типа переменной также показана в таблице 2.3.

Таблица 2.3. Переменные, диапазон кодов и длина переменных

Тип переменной	Диапазон символьного кода	Длина в области переменных
Цифровой (односимвольное имя)	97 122	6
Цифровой (многосимвольное имя)	161 186	5 + длина имени
Цифровой массив	129 - 154	4 + 2*размерность + 5*общее количество элементов
Управляющая переменная цикла FOR - NEXT	225 - 250	18
Символьная строка	65 - 90	3 + длина строки
Символьный массив	193 - 218	4 + 2*размерность + общее число элементов

ПОДПРОГРАММЫ ПЗУ

Некоторые из представленных в книге программ (раздел B) используют стандартные подпрограммы ПЗУ:

- RST 16 - распечатывает содержимое аккумулятора,
- CALL 3976 вставляет символ, хранящийся в аккумуляторе, по адресу, хранящемуся в паре регистров HL,
- CALL 6326 если аккумулятор хранит код 14, устанавливается нулевой флаг и увеличивается пара регистров HL в пять раз,
- CALL 6510 возврат в HL адреса в ОЗУ той строки, номер которой был передан в эту подпрограмму через HL.

3. МАШИННЫЙ ЯЗЫК Z80

РЕГИСТРЫ Z80A

Во время выполнения программы компьютер не обновляет непосредственно содержимое памяти. Он копирует содержимое

ячейки памяти в регистр и оперирует содержимым регистра. Регистры в машинном языке имеют функцию, схожую с переменными в BASIC'е, т.е. используются для хранения чисел. Они отличаются от переменных BASIC'а тем, что число их ограничено и они существуют в процессоре, а не в RAM. Кроме того, они могут хранить только один байт или два байта в паре регистров.

Z80A имеет несколько регистров и потому может хранить несколько чисел одновременно. Благодаря этому снижается время при обмене информацией между процессором и памятью.

РЕГИСТР "А" (АККУМУЛЯТОР)

Аккумулятор это наиболее важный регистр, потому что больше всего арифметических и логических команд выполняется с содержимым именно этого регистра. Этот регистр называется аккумулятором, т.к. результат последовательных операций накапливается ("аккумулируется") в нем. Некоторые команды, которые обращаются к аккумулятору, используют другой регистр или адрес памяти в качестве источника данных. Например, инструкция

ADD A,B

дает процессору команду прибавить содержимое регистра B к содержимому регистра A, поместив результат в A.

ФЛАГ-РЕГИСТР "F" (РЕГИСТР СОСТОЯНИЙ)

Регистр F довольно значительно отличается от всех остальных, т.к. его содержимое не рассматривают как один байт, а рассматривают как 8 индивидуальных битов, что конечно же одно и то же. Эти биты используются в качестве так называемых флагов для управления последовательностью выполнения программы. Каждый флаг используется для определения того, какое из двух логически противоположных условий имело место при выполнении предыдущей операции. Например, флаг нуля определяет был ли равен нулю результат последней операции сложения, вычитания и т.п. Только четыре из восьми флагов наиболее интересны для пользователей. Их свойства кратко изложены в таблице 3.2.

Таблица 3.2. Четыре флага, которые контролируют наибольшее количество операций Z80A

Флаг	Мнемоника		Использование
Знак	N	P	Включается, когда результат последней операции отрицательный

Нуль	Z	NZ	Включается, когда результат последней операции равен нулю или имело место совпадение
Перенос	C	NC	Включается, когда происходит переполнение регистра, т.е. последний результат больше того, что может быть записанным в один байт (или в два байта для операций с парой регистров)
Четность/Переполнение	PE	PO	Флаг включается, если в байте результата предыдущей операции количество включенных битов есть величина четная. В некоторых операциях этот флаг свидетельствует о переполнении

Флаг знака наиболее простой. Кроме того, условлено, что в операциях со знаковыми числами седьмой бит (старший бит в байте) тоже используется для хранения знака. Он включается, когда число отрицательно, иначе снимается. Этот флаг отражает знак последнего результата.

Флаг нуля устанавливается, если результат последней операции равен нулю. Он также используется инструкциями сравнения, которые фактически аналогичны вычитанию, при котором результат игнорируется.

Флаг переноса регистрирует переполнение, которое имеет место, если результат сложения длиннее чем то, что может быть записано в регистр, либо имеет место заем при вычитании. Имеется также несколько инструкций сдвига, в которых биты в регистре А сдвигаются влево или вправо циклически через флаг переноса.

Флаг четности/переполнения это два флага в одном. Он используется как флаг переполнения при выполнении арифметических операций для определения был ли бит 7 получен в результате переноса или сгенерирован битом 6 при "взятии займа" Это используется в случае, если бит знака был искажен. Логические инструкции используют тот же самый флаг для определения четности результата.

Примечание: четность двоичного числа - это четность количества битов, установленных в единицу. Если количество четно, то говорят, что имеет место четность, если оно нечетно, то гово-

рят, что имеет место нечетность. Флаг устанавливается, если имеет место четность.

Результат некоторых инструкций зависит от текущих установок отдельных флагов. Например, инструкция

JR Z,D

позволяет Z80A "перепрыгнуть" через несколько инструкций, если флаг нуля установлен (т.е. сделать условный переход типа IF...THEN GO TO). Если флаг нуля не установлен, процессор выполняет следующую инструкцию в обычной последовательности. Т.о. флаговый регистр очень важен, т.к. он позволяет процессору принимать решения и переходить к другой части программы.

РЕГИСТРЫ-СЧЕТЧИКИ "B" и "C"

Регистр B и, в некоторой степени, регистр C, с которым он может использоваться в паре, доступен для нескольких применений. Но наиболее важно использование его в качестве счетчика. Мы уже видели, как выполнение программы может управляться использованием флага 0 в инструкции JR Z,n.

Другая инструкция DJNZ n использует флаг 0 для создания циклов, используя регистр B в качестве счетчика аналогично циклам FOR-NEXT в BASIC'е. Когда встречается эта инструкция, Z80A уменьшает содержимое регистра B на 1. Если результат равен нулю, то выполняется следующая инструкция в нормальной последовательности. Если результат не равен нулю, подпрограмма "перескакивает" n инструкций. Если программист использует отрицательное значение для n, "прыжок" осуществляется в программе назад и, если здесь нет других переходов, процессор в конце концов встретит ту же самую инструкцию вновь.

Таким образом загрузка регистра B определенным числом и установкой соответствующего смещения n определенная часть кодов может быть выполнена заданное количество раз. Регистр B содержит только один байт и поэтому может устанавливаться на любое число от 0 до 255, т.е. используя описанный механизм через определенный блок кодов может быть сделано максимум 255 проходов. Для осуществления более 255 проходов в цикле нет эквивалентных инструкций, но имеются очень мощные инструкции, которые используют все 16 битов регистровой пары BC как счетчик с максимальным значением 65535. Например, инструкция CPDR. Когда Z80A выполняет ее, то:

1) содержимое BC уменьшается на 1;
2) уменьшается содержимое HL (о регистровой паре HL см. ниже);

3) сравнивается содержимое аккумулятора A с содержимым ячейки памяти, адрес которой находится в паре регистров HL.

Процессор повторяет эти действия до тех пор, пока не будет совпадения между содержимым A и ячейки памяти, либо пока не обнулится BC. Таким образом эта инструкция может быть использована для поиска ячейки, содержащей определенное число.

АДРЕСНЫЕ РЕГИСТРЫ "DE" и "HL"

Регистры D и E не имеют особых функций и, в основном, используются в качестве временной быстродоступной памяти. Они также используются в паре для хранения адреса ячейки, которая интересует нас в текущий момент.

Основная функция регистров H и L хранить в паре адрес ячейки памяти. Мы уже видели, как определенная инструкция использует пару HL с этой целью. H хранит старший байт, L младший. Полный адрес доступен в форме:

адрес = 256*H + L,

что дает максимум 65536 возможных адресов.

ИНДЕКСНЫЕ РЕГИСТРЫ "IX" и "IV"

Индексные регистры IX и IV являются 16-битовыми регистрами и могут быть использованы только так в отличие от регистров BC, DE и HL, которые могут использоваться и в паре, как 16-битовые регистры, и индивадуально, как 8-битовые. IX и IV используются, главным образом, подобно паре HL.

IX и IV, кроме того, имеют одно свойство, которое недоступно для HL они могут быть использованы со смещением s. Имеется в виду, что инструкция, которая ссылается к (IX+s), использует не ячейку памяти, адрес которой хранится в IX, а сначала смещение s прибавляется к значению в IX, чтобы получить новый адрес, и с ним и работает инструкция.

УКАЗАТЕЛЬ СТЕКА "SP"

Стек - это область в верхней части ОЗУ, которая используется для временного хранения содержимого пар регистров. Стек растет вниз, когда идет заполнение и поднимается вверх при его опустошении. Начало стека фиксировано в ZX-SPECTRUM - оно находится непосредственно ниже ячейки, на которую указывает системная переменная RAMTOP. Вершина стека находится ниже нижней границы стека, т.к. стек увеличивается вниз, а уменьшается вверх. Адрес текущей ячейки верхней границы стека хра-

нится в SP-регистре. Передача в стек или из стека происходит с помощью инструкций PUSH и POP. Например:

PUSH HL

В этом случае процессор:

1) уменьшает SP;
2) копирует содержимое регистра H в ячейку, указанную указателем стека SP;
3) уменьшает SP;
4) копирует содержимое регистра L в ячейку, указанную указателем стека SP.

Инструкция POP выполняет противоположное действие. По этому методу последняя поступившая пара чисел, помещенная в стек, является всегда первой парой, которая снимается со стека. Это обеспечивает простой и удобный способ временного хранения содержимого регистров во время вызова подпрограмм.

ПРОГРАММНЫЙ СЧЕТЧИК "PC"

Программный счетчик PC очень важный 16-битовый регистр, т.к. он хранит адрес следующей инструкции, которая должна быть выполнена. Нормальный ход работы когда инструкция выполнятся следующим образом:

1) Скопировать содержимое ячейки, указанной программным счетчиком PC в специальный регистр процессора.
2) Если инструкция содержится в нескольких байтах, увеличить PC и скопировать содержимое следующей ячейки во второй специальный регистр.
3) Увеличить PC таким образом, чтобы он указывал на следующую инструкцию, которая должна быть выполнена.
4) Выполнить инструкцию, которая только что была прочитана.

Команда JUMP так же, как и DJNZ n или JR Z,n, изменяет нормальный ход выполнения программы с помощью изменения PC во время выполнения шага 4. Заметим, что это изменение имеет место после увеличения PC. Таким образом, значение смещения n всегда будет отсчитываться относительно позиции инструкции следующей за инструкцией, содержащей смещение.

АЛЬТЕРНАТИВНЫЕ РЕГИСТРЫ AF', BC', DE', HL'

Z80A имеет копию каждого из A, B, C, D, E, H и L регистров. Отличаются копии использованием знака "'" возле обозначе-

ния регистра, например, A' это копия регистра A. Инструкции не работают с этими регистрами непосредственно, но инструкции обмена позволяют вывести из использования 2 или более регистров и ввести в использование вместо них их копии.

Команды обмена выполняются очень быстро, т.к. содержимое регистра физически не копируется из одного регистра в другой, а вместо этого установка внутренних коммуникаций процессора изменяется так, что дублирующий регистр становится основным, а оригинальный регистр - запасным.

О СИСТЕМЕ КОМАНД ПРОЦЕССОРА Z80

В системе команд Z80 около 700 инструкций. Поскольку возможны только 256 различных комбинаций из 8 битов (2^8=256), то лишь менее половины команд может быть выражено одним байтом. Оставшиеся инструкции хранятся в двух или трех байтах. Некоторые команды следуют с однобайтовым смещением s, или однобайтовым числом n, или двухбайтовым числом (адресом) nn, к которым инструкция ссылается. Всего же одна инструкция может занять максимум 4 байта.

* * *

Примечание ИНФОРКОМа

ZX SPECTRUM располагает еще встроенным программируемым калькулятором, имеющим свою систему команд. Доступ к ним возможен только из машинного кода. Команды калькулятора могут быть более длинными, чем команды процессора и занимать более 4-х байт.

* * *

Для удобства работы на ассемблере все команды записываются с помощью мнемоник (OP CODE). Мнемоника это сокращенное описание каждой команды. Правила задания команд в ассемблере:

1) Одиночные регистры описываются буквой, например, B. Пара регистров именуется в алфавитном порядке, например, BC.

2) Смещение s положительно, если оно находится в пределах от 0 до 127, и отрицательно, если оно находится в пределах от 128 до 255. Большие или меньшие числа недопустимы.

Отрицательное значение подсчитывается вычитанием s из 256. Например, команда относительного безусловного перехода

JR s

вызывает переход вперед на 8 байтов, если s=8 и переход назад на 8 байтов, если s=248, т.к. 256-8=248.

Запомните при подсчете смещения, что переход осуществляется с адреса первого байта следующей команды.

3) Однобайтовое число n лежит в пределах от 0 до 255 включительно.

4) Двухбайтовое число или адрес представляется как nn и лежит в пределах от 0 до 65535 включительно.

5) nn, заключенное в скобки, т.е. (nn), подразумевает "содержимое ячейки по адресу nn", тогда как nn без скобок подразумевает "число nn" Таким образом, LD HL,(23627) подразумевает загрузку пары регистров HL содержимым ячеек 23627 и 23628, тогда как:

```
LD    HL,23627
```

подразумевает загрузку HL числом 23627.

Таким же образом (HL) подразумевает "содержимое ячейки с адресом, хранящимся в HL", тогда как HL без скобок означает "число, хранящееся в HL"

Такой вид адресации называется косвенной адресацией.

6) Место назначения результата операции всегда задается первым. Например:

```
ADD  A,B
```

означает "прибавить содержимое регистра B к содержимому регистра A и результат оставить в регистре A"

ПРОДОЛЖЕНИЕ СЛЕДУЕТ

(c) Колотов Сергей, г. Шадринск, SerzhSoft, XI, 1996

ЕДИНЫЙ СТАНДАРТ ПРОГРАММИРОВАНИЯ НА АССЕМБЛЕРЕ.

В последнее время на страницах ZX-РЕВЮ все чаще и чаще можно встретить публикации, посвященные стандартизации. Наступление идет по всем фронтам: начиная от проекта ROM и заканчивая моделями компьютеров. Но пока еще не встречалось ни одной статьи, направленной на стандартизацию программирования. Предчувствуя запах тухлых яиц и гнилых овощей, уже приготовленных "свободными" кодерами и программистами для упражнений по метанию этих признаков уважения в автора данной статьи, сразу хотел бы предостеречь: остановите карающую длань! Ведь никто не заставляет именно Вас следовать нижеприведенным правилам и загонять в очень узкие рамки. Если Вы хотите ощущать полную свободу в программировании так ощущайте ее (и варитесь в собственном соку!)!

Программирование, а тем более на ассемблере, сопряжено с одной большой проблемой: программы, написанные одним, совсем необязательно будут понятны другому (в большинстве случаев это именно так!).

Даже если оба кодера и "съели в ассемблере не по одной собаке"! И это вполне естественно: один любит писать метки на русском языке английскими буквами, другой - наоборот, а третий - вообще вместо меток использует номера, ставя перед ними какую-либо букву, чтобы ассемблер это все нормально переваривал... Да и сам автор, написавший программу, через некоторое продолжительное время будет тупо смотреть на свое собственное "творение", не понимая как оно еще может работать!

Хорошо ли это? Нет, очень и очень плохо! В то время, как старина Speccy все больше и больше "загибается" от невиданного нашествия "амиг" и IBM, нужно не брести по одиночке многочисленными запутанными тропами, а "вместе весело шагать" одной проторенной дорогой, где все понимают друг-друга с полуслова. То есть предлагается "единый стандарт программирования на ассемблере", о котором далее и пой-

дет речь. Автор, конечно, понимает, что данная работа не закончена, и у кого-либо может вызвать протест, но главная задача этой статьи состоит не в публикации кучи непогрешимых заповедей, а предполагает возникновение диалога на страницах ZX РЕВЮ. Присылайте свои дополнения, предложения и просто высказывайте мнение, которое считаете наиболее правильным. Итак, вот некоторые пункты из моей "Библии Кодера":

1. Чем меньше в программе директив ORG - тем лучше.
2. Процедур же, наоборот, должно быть побольше.
3. В больших программах обязательно должен быть головной блок, в котором производятся вызовы рабочих процедур.
4. Чем больше в программе комментариев, тем лучше. Здесь срабатывает принцип "Маслом каши не испортишь."
5. Все процедуры отделяются друг от друга символом ";" (точка с запятой) в отдельной строке (пустой комментарий).
6. Каждая процедура начинается с уникального имени, являющегося сокращением английских слов, характеризующих действие данной процедуры. В тех ассемблерах, где это предусмотрено, имя процедуры заканчивается символом ":" (двоеточие). В строке с именем процедуры больше не должно быть никаких команд (возможен комментарий).
7. Метки внутри процедур.

Два первых символа любой метки должны однозначно указывать на тип этой метки:

LP (LOOP) - цикл
GO (GO TO) - переход
IF (IF) - переход по условию (расширение GO)
ER (ERROR) - ошибка
EX (EXIT) - выход, завершающие операции
LB (LABEL) - команда, изменяемая при выполнении
CH (CHANGE) - аналог LB
SP,HL,DE,BC,AF,IX,IY - изменяемая команда загрузки этих регистров (или их половинок). Например:

```
        LD      (SP_GKY+1),SP
;изменение

SP_GKY  LD      SP,NULL
;изменяемая команда
```

Для того, чтобы сразу было однозначно ясно и понятно, что данная команда изменяется во время выполнения программы, необходимо в команде загрузки регистра указывать в качестве операнда специально опеделенную в начале программы метку NULL. Она задается так: NULL EQU 0. Ее необходимо использовать во всех изменяемых командах.

Итак, после двух символов типа в метке далеее следуют 2-3 символа (для удобства могут отделяться символом подчеркивания "_"), характеризующие принадлежность к опеделенной процедуре. Эти символы берутся из имени процедуры, обычно являясь начальными буквами английских слов (сокращений), образующих имя процедуры.

Если в процедуре несколько однотипных меток, то далее ставится номер метки: 0..9 или 00..99.

Примеры меток для процедур:
CLRSCR LPCS1, GOCS04, EX_CS9, LB_CS...
GETKEY GOGKY1, LP_GKY, IF_GKY и т.д.

8. Переменные, массивы, таблицы как и процедуры должны быть "говорящими" Вот часто используемые элементы, входящие в состав имен переменных, процедур: см. Listing 1.

Итак, все эти сокращения (плюс сокращения для меток внутри процедур) удобно применять в программировании. Их также можно включать в имена файлов и т.д.

На этом позвольте раскланяться. Надеюсь, что читатели продолжат данное начинание и напишут отзывы о необходимости такой стандартизации программирования на ассемблере.

Listing 1.

ADR (ADDRESS)	АДРЕС
CNT (COUNTER)	СЧЕТЧИК
SPD (SPEED)	СКОРОСТЬ
LVL (LEVEL)	УРОВЕНЬ
LEN (LENGTH)	ДЛИНА
SZE (SIZE)	РАЗМЕР
HGT (HEIGHT)	ВЫСОТА
WDT (WIDTH)	ШИРИНА
VLM (VOLUME)	ОБЪЕМ
CHR (CHAR)	СИМВОЛ
TXT (TEXT)	ТЕКСТ
STR (STRING)	СТРОКА
SRL (SCROLL)	СКРОЛЛЕР
FLD (FIELD)	ПОЛЕ
BRD (BORDER)	ГРАНИЦА
PAP (PAPER)	БУМАГА
INK	ЧЕРНИЛА
ARW (ARROW)	СТРЕЛКА
SPR (SPRITE)	СПРАЙТ
XR,YR (X,Y-RANGE)	РАЗМЕРЫ
X,Y,Z	КООРДИНАТЫ
AY	МУЗ. ПРОЦЕССОР
AYA,AYB,AYC	КАНАЛЫ AY
IN	ВХОДНОЙ
OUT	ВЫХОДНОЙ
BUF (BUFFER)	БУФЕР
BIN (BINARY)	ДВОИЧНЫЙ
HEX	ШЕСТНАДЦАТЕРИЧНЫЙ
DEC (DECIMAL)	ДЕСЯТИЧНЫЙ
BCD	ДВОИЧНО-ДЕСЯТИЧНЫЙ
TRB (TRURBO)	БЫСТРЫЙ
POS (POSITION)	ПОЛОЖЕНИЕ
CR (CARRAGE)	КАРЕТКА
LST (LAST)	ПОСЛЕДНИЙ
LST (LISTING)	ЛИСТИНГ
DAT (DATA)	ДАННЫЕ
ARR (ARRAY)	МАССИВ
ERR (ERROR)	ОШИБКА
NEW	НОВЫЙ
MDE (MODE)	РЕЖИМ
FL,FLG (FLAG)	ФЛАГ
TBL (TABLE)	ТАБЛИЦА
CHN (CHANNEL)	КАНАЛ
NXT (NEXT)	СЛЕДУЮЩИЙ
LN (LINE)	ЛИНИЯ
STK (STACK)	СТЕК
MEM (MEMORY)	ПАМЯТЬ
BEG,BGN (BEGIN)	НАЧАЛО
END	КОНЕЦ
PTR (POINTER)	УКАЗАТЕЛЬ
TOP	ВЫСОТА
BOT (BOTTOM)	ДНО
OLD	СТАРЫЙ
UDG	СИМВОЛЫ UDG
NO,NOT	ОТРИЦАНИЕ
YES	ДА
SCR (SCREEN)	ЭКРАН
SHD (SHADOW)	ТЕНЕВОЙ
MSK (MASK)	МАСКА
INT (INTERRUPT)	ПРЕРЫВАНИЕ
PRN (PRINT)	ПЕЧАТЬ
GET	ВЗЯТЬ
PUT	ПОЛОЖИТЬ
KEY	КЛАВИША, КЛЮЧ
ON	ВКЛЮЧЕНО
OFF	ВЫКЛЮЧЕНО
SYS (SYSTEM)	СИСТЕМНЫЙ
USR (USER)	ПОЛЬЗОВАТЕЛЬСКИЙ
FLE (FILE)	ФАЙЛ
CDE (CODE)	КОДЫ
OBJ (OBJECT)	ОБЪЕКТ
MAP	КАРТА
ROM	ПЗУ
RAM	ОЗУ
DRV (DRIVE)	ДИСКОВОД
DVR (DRIVER)	ДРАЙВЕР
DVC (DEVICE)	УСТРОЙСТВО
HDR (HEADER)	ЗАГОЛОВОК
BCK (BACK)	НАЗАД
TR,TRD	TR-DOS

DOS	ДОС
UP	ВВЕРХ
DN (DOWN)	ВНИЗ
LF,LFT (LEFT)	ВЛЕВО
RG,RGH (RIGHT)	ВПРАВО
FR,FIR (FIRE)	ОГОНЬ
CLR (CLEAR)	ОЧИСТКА
CLR (COLOR)	ЦВЕТ
CLS	ОЧИСТКА ЭКРАНА
LO (LOW)	МЛАДШИЙ
HI (HIGH)	СТАРШИЙ
MAX,MX	МАКСИМАЛЬНЫЙ
MIN,MN	МИНИМАЛЬНЫЙ
SET	УСТАНОВИТЬ
RES (RESET)	ВЫКЛЮЧИТЬ
VAR (VARIABLE)	ПЕРЕМЕННАЯ
NUM (NUMBER)	ЧИСЛО
SGN (SIGN)	ЗНАК
MUS (MUSIC)	МУЗЫКА
SND (SOUND)	ЗВУК
FNT (FONT)	ШРИФТ
CPR (COMPRESS)	КОМПРЕССОР
DCP (DECOMPRESS)	ДЕКОМПРЕССОР
CDR (CODER)	КОДИРОВЩИК
DCD (DECODER)	ДЕКОДЕР
DCR (DECRUNCHER)	ДЕКОМПРЕССОР
FOR	ДЛЯ
TO	К, ДО
SPC (SPACE)	ПРОСТРАНСТВО
PLC (PLACE)	МЕСТО
ARC (ARCHIVE)	АРХИВ
RAR (REARCHIVATOR)	РАЗАРХИВАТОР
FRM (FORMAT)	ФОРМАТ
FRM (FRAME)	КАДР
SCN (SCANNER)	СКАНЕР
DEM (DEMO)	ДЕМОНСТРАЦИЯ
EX,EXT (EXIT)	ВЫХОД
EXT (EXTENTION)	РАСШИРЕНИЕ
INI	ИНИЦИАЛИЗАЦИЯ
WRK (WORK)	РАБОТА
TST (TEST)	ТЕСТ
DGT (DIGIT)	ЦИФРА

****

(С) Иван Рощин, Москва, 1996г.

ДОСТУП К ПОРТУ #1F В TR-DOS 5.03.

Программируя дисковые операции на уровне портов ввода-вывода, Вы неизбежно столкнетесь с необходимостью получить значение регистра состояния из порта #1F. Как же это сделать? Известно, что адреса регистров микроконтроллера появляются в адресном пространстве только в момент работы дисковой системы. Из бейсик-системы регистры недоступны. Не сделано, также, никаких попыток обеспечить прямой доступ к ним из дисковой системы. Несмотря на это, "достать" микроконтроллер все-таки возможно, для этого используются "обрывки" программ ОС. По другому, к сожалению, не получится.

Опубликованная в "ZX-РЕВЮ" 96/4-5 информация о том, что в ПЗУ TR-DOS 5.03 по адресу #09BF стоит последовательность команд IN A,(#1F) и RET, не подтвердилась. В фирменной версии ПЗУ по этому адресу Вы можете найти лишь коды #FF. Еще один экзотический способ доступа к портам (передаем управление в ПЗУ TR-DOS за несколько тактов до прерывания 2-го рода, а процедура обработки прерывания получает считанное из порта значение) теоретически неплох, но практически вряд ли осуществим (а если промежуток между двумя прерываниями непостоянен, этот метод вообще неприменим).

В книге А.Ларченко и Н.Родионова "ZX Spectrum & TR-DOS для пользователей и программистов" приводится следующий алгоритм считывания регистра состояний микроконтроллера:

- -записать 0 в регистр дорожки;
- -записать #0A в регистр сектора;
- -записать 1 в регистр D;
- -вызвать подпрограмму по адресу #3F33;
- -взять из регистра B значение регистра состояний;

- -восстановить в регистрах дорожки и сектора исходные значения.

Я пользовался этим способом, и все вроде бы было нормально. Но я стал замечать, что в некоторых случаях при обращении к процедуре считывания регистра состояния компьютер зависал или сбрасывался. Пытаясь найти причину, я выяснил следующее:

АЛГОРИТМ РАБОТАЕТ НЕПРАВИЛЬНО, ЕСЛИ 0-Й БИТ РЕГИСТРА СОСТОЯНИЙ УСТАНОВЛЕН В 1 (КОНТРОЛЛЕР ЗАНЯТ ВЫПОЛНЕНИЕМ КАКОЙ-ЛИБО КОМАНДЫ).

Действительно, рассмотрим все возможные случаи работы подпрограммы TR-DOS по адресу #3F33 в зависимости от считанного из порта #1F числа, а заодно разберемся, почему перед ее вызовом необходимо устанавливать описанные выше параметры.

1) Биты 0-6 равны нулю (X0000000).

```
#3F33:  IN      A,(#1F)
        LD      B,A
        AND     #7F
;выделяем биты 0-6
        RET     Z
;выход
```

2) 6-й бит равен 1, 0-й бит равен 0 (X1XXXXX0).

```
#3F33:  IN      A,(#1F)
        LD      B,A
        AND     #7F
        RET     Z
;не выходим
        LD      HL,#29D8
        AND     #40
;проверяем 4-й бит
        JR      NZ,#3F4B
;условие выполняется
#3F4B:  LD      A,#D0
        OUT     (#1F),A
        LD      A,B
        AND     #1
;проверяем 0-й бит
        JP      NZ,#3EE7
;условие не выполняется
        IN      A,(#3F)
;регистр дорожки равен 0
        OR      A
        JR      NZ,#3F5F
;условие не выполняется
        IN      A,(#5F)
;регистр сектора равен #0A
        CP      #0A
        RET     Z
;условие выполняется
```

3) 6-й бит равен 0, 2-й бит равен 0-й бит равен 0, какой-то из бит 1,3,4,5 равен 1. (X0XXX0X0).

```
#3F33:  IN      A,(#1F)
        LD      B,A
        AND     #7F
        RET     Z
;условие не выполняется
        LD      HL,#29D8
        AND     #40
;проверяем 4-й бит
        JR      NZ,#3F4B
;условие не выполняется
        LD      A,B
        AND     #4
        JR      Z,#3FA0
;условие выполняется

#3FA0:  DEC     D
;получили 0
        JP      Z,#3F48
;условие выполняется

#3F48:  LD      HL,#29E2
        LD      A,#D0
        OUT     (#1F),A
        LD      A,B
        AND     #1
;проверяем 0-й бит
        JP      NZ,#3EE7
;условие не выполняется
        IN      A,(#3F)
```

```
;регистр дорожки равен 0
        OR      A
        JR      NZ,#3F5F
;условие не выполняется
        IN      A,(#5F)
;регистр сектора равен #0A
        CP      #0A
        RET     Z
;условие выполняется
```

4) 0-й бит равен 1 (XXXXXXX1).

В этом случае, независимо от значения других битов, управление будет передано по адресу #3EE7. Там расположена процедура, портящая значение некоторых переменных TR-DOS и завершающая свою работу так:

```
#01F3:  LD      SP,(#5D1C)
        LD      HL,(#5D1A)
        LD      BC,(#5D0F)
        LD      B,0
        JP      (HL)
```

Так как в ячейках #5D1A-#5D1D, вообще говоря, могут содержаться произвольные значения, дальнейшее выполнение программы непредсказуемо. Вдобавок, значение, считанное из порта #1F и хранящееся в регистре B (ради которого все и затевалось), будет потеряно.

Теперь я приведу написанную мной процедуру, которая правильно (всегда!) определяет значение регистра состояния. При своей работе она не изменяет значение ни одной системной переменной TR-DOS.

```
;*****************************************
;ПРОЦЕДУРА STATUS
ВОЗВРАЩАЕТ СОДЕРЖИМОЕ
;РЕГИСТРА СОСТОЯНИЯ.
;ВХОД: A -СОДЕРЖИМОЕ
РЕГИСТРА ДОРОЖКИ,
; B -СОДЕРЖИМОЕ РЕГИСТРА
СЕКТОРА,
; КОТОРЫЕ БУДУТ
УСТАНОВЛЕНЫ ПОСЛЕ
; ВЫХОДА ИЗ ПРОЦЕДУРЫ.
;ВЫХОД: A -ЗНАЧЕНИЕ,
СЧИТАННОЕ ИЗ ПОРТА #1F.
;ПРЕРЫВАНИЯ ПОСЛЕ
ВЫХОДА ЗАПРЕЩЕНЫ!

STATUS DI
        LD      (RG_D+1),A
;ДОРОЖКА
        LD      A,B
        LD      (RG_S+1),A
;СЕКТОР

;СОХРАНЯЕМ СОДЕРЖИМОЕ
ЯЧЕЕК, КОТОРЫЕ
;МОГУТ БЫТЬ ИСПОРЧЕНЫ:

LD      A,(#5D0E)
LD      (ST1+1),A
LD      A,(#5D0C)
LD      (ST2+1),A
LD      A,(#5CB6)
LD      (ST3+1),A
LD      A,(#5D1F)
LD      (ST4+1),A
LD      A,(#5C3A)
LD      (ST5+1),A
LD      A,(#5D17)
LD      (ST6+1),A
LD      HL,(#5D1A)
LD      (ST7+1),HL
LD      HL,(#5D1C)
LD      (ST8+1),HL
LD      HL,(#5CF8)
LD      (ST9+1),HL

;УСТАНАВЛИВАЕМ
СОДЕРЖИМОЕ НЕКОТОРЫХ
;ЯЧЕЕК ДЛЯ ПРАВИЛЬНОЙ
РАБОТЫ:

LD      A,#FF
LD      (#5D0C),A
LD      (#5D1F),A
DEC     A
LD      (#5D0E),A
```

```
LD      A,#F4
LD      (#5CB6),A

LD      HL,S_SPEC
LD      (#5D1A),HL
LD      HL,0
ADD     HL,SP
LD      DE,-12
ADD     HL,DE
LD      (#5D1C),HL

LD      A,0
;0 В РЕГИСТР
LD      C,#3F
;ДОРОЖКИ
CALL    TO_WG93
LD      A,#0A
;#A В РЕГИСТР
LD      C,#5F
;СЕКТОРА
CALL    TO_WG93
LD      D,1
LD      IX,16179
CALL    TO_DOS
;ОПРЕДЕЛИЛИ #1F
```

;ТЕПЕРЬ ВОССТАНАВЛИВАЕМ СОДЕРЖИМОЕ
;РЕГИСТРОВ ДОРОЖКИ И СЕКТОРА:

```
RG_D    LD      A,0
        LD      C,#3F
        CALL    TO_WG93
RG_S    LD      A,0
        LD      C,#5F
        CALL    TO_WG93
```

;ВОССТАНАВЛИВАЕМ РАНЕЕ ЗАПОМНЕННОЕ
;СОДЕРЖИМОЕ ЯЧЕЕК:

```
ST1     LD      A,0
        LD      (#5D0E),A
ST2     LD      A,0
        LD      (#5D0C),A
ST3     LD      A,0
        LD      (#5CB6),A
ST4     LD      A,0
        LD      (#5D1F),A
ST5     LD      A,0
        LD      (#5C3A),A
ST6     LD      A,0
        LD      (#5D17),A
ST7     LD      HL,0
        LD      (#5D1A),HL
ST8     LD      HL,0
        LD      (#5D1C),HL
ST9     LD      HL,0
        LD      (#5CF8),HL

        LD      A,B
        RET
```

;СЮДА БУДЕТ ПЕРЕДАНО УПРАВЛЕНИЕ, ЕСЛИ
;0-Й БИТ РЕГИСТРА СОСТОЯНИЙ РАВЕН 1:

```
S_SPEC  POP     BC
;СОДЕРЖИМОЕ ПОРТА
        LD      HL,(#5D1C)
        LD      DE,12
;ВОССТАНАВЛИВАЕМ
        ADD     HL,DE
;УКАЗАТЕЛЬ
        LD      SP,HL
;СТЕКА
        JR      RG_D

TO_WG93 LD      IX,#2A53
TO_DOS  PUSH    IX
        JP      #3D2F
```

Я думаю, нет лучшего способа объяснить, как работает эта процедура, чем показать процесс ее выполнения, начиная с момента вызова подпрограммы по адресу #3F33. При этом выберем именно такой случай, когда 0-й бит регистра состояния равен 1:

```
#3F33:  IN      A,(#1F)
;допустим, считано #FF
        LD      B,A
        AND     #7F
        RET     Z
```

```
;не выходим
        LD      HL,#29D8
        AND     #40
;проверяем 4-й бит
        JR      NZ,#3F4B
;условие выполняется

#3F4B:  LD      A,#D0
        OUT     (#1F),A
        LD      A,B
        AND     #1
;проверяем 0-й бит
        JP      NZ,#3EE7
;условие выполняется

#3EE7:  CALL    #272B

#272B:  LD      A,#1A
        JR      #2731

#2731:  LD      (#5C3A),A
        RET

#3EEA:  LD      A,#FF
        LD      (#5D17),A
        JP      #271B

#271B:  LD      HL,#27FC
        LD      A,#06
        JP      #1C4A

#1C4A:  CALL    #03C3

#03C3:  PUSH    AF
        LD      A,(#5D0E)
;в эту ячейку
;перед запуском
;занесли #FE
        CP      #FE
        JR      NZ,#03CD
;условие не выполняется
        POP     AF
        RET

#1C4D:  JP      #01D3

#01D3:  LD      HL,0
        LD      (#5CF8),HL
        CALL    #20E5

#20E5:  PUSH    AF
        LD      A,(#5CF8)
;там 0
        CP      #FF
        JR      Z,#211C
;условие не выполняется
        POP     AF
        CALL    #2970

#2970:  PUSH    HL
        PUSH    DE
        PUSH    BC
```

Как видим, здесь содержимое BC (где находится столь нужный нам регистр состояния) заносится в стек. В моей процедуре этот факт используется для последующего восстановления BC.

```
        PUSH    AF
        LD      HL,#5D0C
        ;в эту ячейку
        ;перед запуском
        ;занесли #FF
        LD      A,(HL)
        OR      A
        JR      NZ,#2992
        ;условие выполняется

#2992:  POP     AF
        POP     BC
```

Содержимое BC извлечено из стека, но в памяти, отведенной под стек, оно осталось. Его можно извлечь оттуда, лишь бы оно не было затерто записываемыми в стек значениями. Процедура обработки прерываний затрет значение BC, а потому прерывания запрещены.

```
        POP     DE
        POP     HL
        RET

#20F1:  PUSH    AF
        LD      A,(#5CB6)
;в эту ячейку
;перед запуском
```

```
;занесли #F4
            CP      #F4
            JR      Z,#211C
;условие выполняется

#211C:      POP     AF
            RET

#01DC:      CALL    #1D63

#1D63:      LD      HL,#5D0E
;в эту ячейку
;перед запуском
;занесли #FE
            LD      A,(HL)
            CP      #FF
            LD      (HL),0
            RET     NZ
;условие выполняется

#01DF:      LD      HL,#5D17
            LD      (HL),#AA
            LD      HL,#5D1F
;в эту ячейку
;перед запуском
;занесли #FF
            LD      A,(HL)
            OR      A
            LD      (HL),0
            JR      NZ,#01F3
;условие выполняется

#01F3:      LD      SP,(#5D1C)
;значения этих ячеек
            LD      HL,(#5D1A)
;подготовлены заранее
            LD      BC,(#5D0F)
;теперь BC осталось
;только в стеке...
            LD      B,0
            JP      (HL)
```

При выходе управление передается на метку S_SPEC, а SP указывает на то самое значение системного регистра, которое когда-то было сохранено в стеке.

****

(C) Иван Рощин, 1996.

О СОКРАЩЕНИИ ВРЕМЕНИ ФОРМАТИРОВАНИЯ.

Известно, что при копировании дисков с форматированием диска-копии обычно применяется следующий алгоритм: за один оборот диска происходит форматирование, а при следующем обороте происходит запись секторов. Я хочу показать, как можно осуществить форматирование и запись за один оборот диска.

Вспомним, что при форматировании дорожки в поле данных сектора могут быть, вообще говоря, любые байты (исключая служебные). Обычно в качестве данных применяют коды #00 или #FF, но можно записывать и непосредственно информацию, содержащуюся в соответствующем секторе на диске-оригинале. За счет этого уменьшается время копирования. Естественно, этот способ не позволяет записывать сектора, в которых встречаются служебные байты #F5, #F6 и #F7 (однако запись служебных байтов #FC и #FE возможна).

Для практического применения привожу пример алгоритма записи дорожки:

- - записать в регистр команд ВГ93 код команды форматирования;
- - выдавать на ВГ байты, соответствующие выбранному формату, до тех пор, пока не начнется поле данных сектора;
- если в этом секторе содержатся служебные байты #F5, #F6 или #F7, его запись при форматировании невозможна, и на ВГ в качестве данных

выдаются байты #00 или #FF. В противном случае в качестве данных выдаются байты, которые должны быть записаны в сектор.

- - аналогичным образом поступаем и с остальными секторами;
- - после окончания форматирования дорожки записываем на нее сектора, которые не удалось записать при форматировании. При практической проверке этого метода выяснилось, что общее время записи уменьшается не так значительно, как предполагалось (не в два раза). Чтобы понять, почему это происходит, подробно разберем, на что же уходит "лишнее" время:
- форматирование автоматически начинается по приходу индексного импульса, то есть с начала дорожки. Таким образом, на ожидание индексного импульса тратится некоторое время;
- форматирование заканчивается по приходу следующего индексного импульса, т.е. когда записана вся дорожка.

Итак, мы видим, что после форматирования дорожки на одной стороне диска теряется целый оборот, прежде чем может начаться форматирование дорожки на другой стороне диска. Если две последовательно форматируемые дорожки относятся к одному физическому цилиндру, этот оборот диска теряется зря, и никакого уменьшения времени копирования не будет. Но если эти дорожки относятся к разным цилиндрам, время оборота диска можно использовать для того, чтобы передвинуть магнитные головки на следующий цилиндр, так что за счет этого общее время копирования уменьшится.

Возникает вопрос - неужели нельзя избежать потери оборота диска при форматировании дорожки? Оказывается, можно. Для этого нужно заканчивать форматирование не по приходу индексного импульса, а немного раньше, использовав команду принудительного прерывания. Правда, я сам не проверял этот способ и не знаю, как его использование отразится на качестве разметки. Но при желании вы можете проверить это сами.

****

ПРОГРАММА "ЖИЗНЬ"

ИФК: На крик о помощи по программе "LIFE" (ZX РЕВЮ 96/1-2), откликнулся Анисимов Денис и прислал свой вариант этой программы. Максимальное время сканирования всего экрана составляет около 4.5 секунд.

КОРР· Программа написана в формате GENS'а, и владельцам других ассемблеров придется изменить текст, особенно в области макросов. Макрос DWHL я переписал из описания MASM-128 (ИФК: ZX РЕВЮ 96/1-2), DWDE является переделкой DWHL, а UPHL я сделал сам. Счастливые пользователи MASM'а могут эти макросы не определять, а воспользоваться готовыми DOWN HL, DOWN DE и UP HL из этого ассемблера.

Изобразите на экране необходимую комбинацию точек и на-

чинайте вызывать программу на исполнение, например так:

```
10 randomize usr adr : go to 10,
```

где adr - это адрес ассемблирования. Если запускать не через randomize..., а через let a=... то в переменной будет находится количество включенных пикселов. Это можно испльзовать так:

```
10 let a=usr adr : if a>0 then go to
   10
```

или проще:

```
10 if usr adr then go to 10
```

Если какой-нибудь индивидуум в процессе передвижения подойдет к краю экрана, то он появится с другой стороны. И еще: у программы есть один недостаток для ее работы необходим буфер с адреса #C000 (49152) длиной #1800 (6144).

Текст программы "Жизнь":

```
 Program 'Life'
 (C) Anisimow Den 1996
;*********************************
DWHL    MAC
        INC     H
        LD      A,H
        AND     #07
        JR      NZ,$+12
        LD      A,L
        SUB     #E0
        LD      L,A
        JR      NC,$+6
        LD      A,H
        SUB     #08
        LD      H,A
        ENDM
DWDE    MAC
        INC     D
        LD      A,D
        AND     #07
        JR      NZ,$+12
        LD      A,E
        SUB     #E0
        LD      E,A
        JR      NC,$+6
        LD      A,D
        SUB     #08
        LD      D,A
        ENDM
UPHL    MAC
        DEC     H
        LD      A,H
        CPL
        AND     #07
        JR      NZ,$+12
        LD      A,L
        SUB     #20
        LD      L,A
        JR      C,$+6
        LD      A,H
        ADD     A,#08
        LD      H,A
        ENDM
TS1     MAC
        RRA
        JR      NC,$+3
        INC     B
        ENDM
TS2     MAC
        RLA
        JR      NC,$+3
        INC     C
        ENDM
TS3     MAC
        OR      C
        INC     A
        JR      NZ,$+3
        INC     B
        ENDM

        ORG     40000
        ENT     $
        DI
        EXX
        PUSH    HL
        PUSH    IX
        LD      HL,#C000
        LD      DE,#C001
        LD      BC,#17FF
        LD      (HL),L
        LDIR
        LD      HL,#4000
        LD      B,#20
L1      LD      A,(HL)
        OR      A
```

```
        JR      NZ,L8
        INC     L
        DJNZ    L1
        LD      HL,#57E0
        LD      B,#20
L2      LD      A,(HL)
        OR      A
        JR      NZ,L8
        INC     L
        DJNZ    L2
        LD      HL,#56E0
        LD      E,#BE
L3      LD      B,#20
        LD      C,L
L4      LD      A,(HL)
        OR      A
        JR      NZ,L5
        INC     L
        DJNZ    L4
        LD      L,C
        UPHL
        DEC     E
        JR      NZ,L3
        LD      C,B
        EXX
        EI
        JP      L34
L5      LD      HL,#4000
        PUSH    HL
        INC     H
        INC     E
        INC     E
L6      LD      B,#20
        LD      C,L
L7      LD      A,(HL)
        OR      A
        JR      NZ,L9
        INC     L
        DJNZ    L7
        LD      L,C
        POP     AF
        PUSH    HL
        DWHL
        DEC     E
        JR      L6
L8      LD      HL,#4000
        LD      E,#C0
        PUSH    HL
L9      POP     HL
        PUSH    HL
        LD      BC,#1F00
        PUSH    DE
L10     LD      D,#20
L11     LD      A,(HL)
        RLCA
        JR      C,L12
        OR      A
        JR      NZ,L13
        INC     L
        DEC     D
        JR      NZ,L11
        LD      A,L
        SUB     #20
        JR      L21
L12     DEC     L
        LD      A,L
        CPL
        AND     #1F
        JR      Z,L17
L13     LD      A,L
        AND     #1F
        CP      B
        JR      NC,L14
        LD      B,A
L14     LD      A,L
        OR      #1F
        LD      L,A
L15     LD      A,(HL)
        RRCA
        JR      C,L16
        OR      A
        JR      NZ,L18
        DEC     L
        JR      L15
L16     INC     L
        LD      A,L
        AND     #1F
        JR      NZ,L19
L17     POP     DE
        LD      D,#20
        EXX
        POP     HL
        JR      L22
L18     LD      A,L
        AND     #1F
L19     CP      C
        JR      C,L20
        LD      C,A
```

```
L20     XOR     L
        LD      L,A
        DWHL
        DEC     E
        JR      NZ,L10
        POP     DE
        LD      A,C
        SUB     B
        INC     A
        LD      D,A
        LD      A,B
        EXX
        POP     HL
        OR      L
        LD      L,A
L22     PUSH    HL
        UPHL
        LD      A,H
        CP      #40
        JR      NC,L23
        LD      H,#57
L23     EX      (SP),HL
        POP     IX
        EXX
        LD      BC,#0000
L24     PUSH    DE
        EXX
        LD      D,H
        LD      E,L
        DWDE
        LD      A,D
        CP      #58
        JR      C,L25
        LD      D,#40
L25     LD      BC,#0000
        EXX
        BIT     5,D
        EXX
        JR      Z,L26
        PUSH    HL
        PUSH    DE
        LD      A,L
        OR      #1F
        LD      L,A
        LD      A,E
        OR      #1F
        LD      E,A
        LD      A,(IX+#1F)
        TS1
        LD      A,(HL)
        TS1
        LD      A,(DE)
        TS1
        POP     DE
        POP     HL
L26     LD      A,(IX)
        TS2
        LD      A,(HL)
        TS2
        LD      A,(DE)
        TS2
        PUSH    BC
        EXX
        POP     HL
        EXX
        LD      A,C
        LD      (DAT+1),A
        PUSH    HL
        PUSH    DE
        LD      C,#7F
L27     BIT     6,C
        JR      NZ,L28
        LD      A,(DE)
        OR      (HL)
        OR      (IX)
        JR      NZ,L28
        LD      C,#FE
L28     PUSH    HL
        RRC     C
        JR      C,L30
        LD      A,L
        OR      #E0
        INC     A
        JR      NZ,L29
DAT     LD      A,#00
        JR      L31
L29     DEFB    #DD
        INC     L
        INC     L
        INC     E
L30     LD      B,#00
        LD      A,(IX)
        TS3
        LD      A,(HL)
        TS3
        LD      A,(DE)
        TS3
        LD      A,B
```

```
L31     EX      AF,AF'
        EXX
        LD      A,H
        LD      H,L
        ADD     A,L
        EX      AF,AF'
        LD      L,A
        EX      AF,AF'
        ADD     A,L
        EXX
        LD      B,A
        EX      (SP),HL
        CP      #05
        JR      NC,L33
        CP      #03
        JR      Z,L32
        LD      A,(HL)
        RRCA
        OR      C
        INC     A
        JR      NZ,L33
        LD      A,B
        CP      #04
        JR      NZ,L33
L32     LD      A,C
        RLCA
        CPL
        SET     7,H
        OR      (HL)
        LD      (HL),A
        EXX
        INC     BC
        EXX
L33     POP     HL
        BIT     7,C
        JR      NZ,L27
        EXX
        DEC     D
        EXX
        JR      NZ,L27
        POP     HL
        POP     IX
        EXX
        POP     DE
        DEC     E
        JP      NZ,L24
        EXX
        LD      HL,#C000
        LD      DE,#4000
        LD      BC,#1800
        EI
        HALT
        LDIR
L34     POP     IX
        POP     HL
        EXX
        RET
```

****

ИФК: Дмитрий прислал еще одну процедуру.

КОРР: Программа не является полностью моей, т.к. идею я взял из программы Кривцова А.Н. в ZX-РЕВЮ 94/4, стр 34.

Процедура выводит на экран стандартную картинку (6912), которую необходимо загрузить по адресу #C000 (49152).

```
(C) Anisimov Den 1996
        ORG     40000
        ENT     $
        LD      DE,#0000
S1      PUSH    DE
        HALT
S2      PUSH    DE
        LD      A,#7F
S3      LD      C,A
        LD      A,D
        CP      #20
        JR      NC,S6
        LD      A,E
        ADD     A,A
        ADD     A,A
        ADD     A,A
        LD      L,A
        LD      H,#16
        ADD     HL,HL
        ADD     HL,HL
        LD      A,L
        ADD     A,D
        LD      L,A
        LD      A,C
        CP      #40
        JR      NZ,S5
        PUSH    DE
        PUSH    HL
```

```
        LD      A,H
        ADD     A,A
        ADD     A,A
        ADD     A,A
        AND     #7F
        LD      H,A
        LD      D,H
        LD      E,L
        SET     7,H
        LD      B,#08
S4      LD      A,(HL)
        LD      (DE),A
        INC     H
        INC     D
        DJNZ    S4
        POP     HL
        LD      B,H
        LD      DE,#8000
        ADD     HL,DE
        LD      A,(HL)
        LD      H,B
        POP     DE
S5      LD      (HL),A
S6      DEC     D
        LD      A,C
        SUB     #09
        CP      #37
        JR      NZ,S3
        POP     DE
        DEC     D
        INC     E
        LD      A,E
        CP      #18
        JR      NZ,S2
        POP     DE
        INC     D
        LD      A,D
        CP      #3F
        JR      NZ,S1
        RET
```

****

ИФК: Колесов Александр из Ростова на Дону советует всем, кто пользуется Мини-драйвером дисковых операций (ZX-РЕВЮ 95/4), внести в него несколько изменений.

КОРР: Предлагаю доработку, позволяющую получить релоцируемый вариант Мини-драйвера.

Для начала строки 25, 36 и 51 (здесь и далее ссылки на исходный вариант А.Алексеева) переделываем следующим образом:

```
25      CALL    #007C   CD 7C 00
        JR      DOS     18 64

36      CALL    #007C   CD 7C 00
        JR      DOS     18 4F

51 M3:  CALL    #007C   CD 7C 00
        JR      DOS     18 2E
```

Как видим, все сводится к замене команды CALL DOS на CALL #007C и относительному переходу на метку DOS.

Далее, ввиду того, что после внесенных изменений длина программы увеличилась на шесть байтов, исправляем следующие строки:

```
63 M4:  DJNZ    L3      10 CE

67      DJNZ    L1      10 B0
```

И, наконец, самое главное. Переделываем программу (это еще шесть дополнительных байтов), начиная со строки 78 (метка DOS):

```
78 DOS: DEC     SP          3B
        DEC     SP          3B
        EX      (SP),HL     E3
        INC     HL          23
        INC     HL          23
        EX      (SP),HL     E3
        PUSH    IX          DD E5
        JP      #3D2F       C3 2F 3D
        END
```

Из всего вышесказанного ясно, что изменения каснулись только способа вызова подпрограмм из ПЗУ TR-DOS. Теперь коментарии.

После того, как отработает всем известная (см. например, ZX РЕВЮ 95/1) конструкция CALL #007C, по команде JR DOS попадаем на метку DOS и получаем на стеке адрес следующей за CALL #007C команды, в нашем случае - это JR DOS:

```
DEC     SP
DEC     SP
```

Помещаем содержимое вершины стека (т.е. адрес возврата) в HL, а HL сохраняем на стеке:

```
EX      (SP),HL
```

Увеличиваем значение адреса возврата, чтобы после выхода из подпрограммы ПЗУ TR-DOS "проскочить" JR DOS:

```
INC     HL
INC     HL
```

Возвращаем на стек новый адрес возврата и восстанавливаем HL:

```
EX      (SP),HL
```

А теперь, как обычно, помещаем на стек адрес подпрограммы из ПЗУ TR-DOS и переходим на ее исполнение:

```
PUSH    IX
JP      #3D2F
```

Вот и все. Смело набирайте и пользуйтесь: пашет как зверь!

****

Бандура Сергей Иванович г. Николаев.

ВЫЗОВ TR-DOS ИЗ МЕНЮ БЕЙСИКА-128

КОРР: Как справедливо заметил Paul Smith (ZX РЕВЮ 96/1-2, стр.48), от Бейсика-128 "одна морока с загрузкой программ" Кроме того, занят буфер принтера, невозмжно использовать LPRINT-III, есть проблемы с некорректной обработкой редактором Бейсика-128 управляющих кодов, вставляемых для затруднения просмотра текста программ.

Ниже предлагается доработка ПЗУ Бейсика-128 (ROM0), при вызове TR-DOS переводящая машину в режим, который можно назвать "48K с открытым портом"

Идею и саму программу перевода я подсмотрел в загрузчике к TASM2.0 (сам ассемблер не работает). Режим отличается тем, что устанавливаются стандартные каналы ввода-вывода, работает редактор Бейсик-48, но сегменты ОЗУ и экран можно переключать при помощи OUT 32765,N, так что тестирующие программы определяют режим как 128K. При этом в любой момент возможно OUT 32765,48 - закрыть порт и установить стандартный режим 48K.

Поскольку ПЗУ Бейсика-48 (ROM1) не изменяется, не возникает проблем с совместимостью.

Дальнейшее описание сделано с допущением, что копия ПЗУ Бейсик-128 загружена в ОЗУ с адреса #8000.

Доработки сводятся к следующему:

1. Адрес #A75E: сообщение Tape loader заменяем на TR DOS; сообщение Tape tester заменяем на Tape loader. Старший бит в последнем символе каждого байта должен быть установлен в 1 (+#80).
2. По адресу #A746 заносим #16; по адресу #A752 заносим #31 (меняем местами).
3. По адресу #B853 заносим #84; по адресу #B857 заносим #5E (тоже меняем местами).

4. С адреса #BBE9, на месте блока Tape tester, размещаем такую программу:

```
ORG     #BBE9
DI
LD      DE,#5B00
LD      HL,#3BF8
LD      BC,#002F
LDIR
JP      #5B00
LD      A,#10
LD      BC,#7FFD
OUT     (C),A
;включаем ПЗУ 48K
LD      SP,#FF54
LD      HL,#1303
PUSH    HL
LD      (#5C3D),SP
;стандартная
LD      HL,#1BB0
;процедура об
PUSH    HL
;работки ошибок
LD      HL,(#5C4F)
;стандартный
LD      DE,#000F
;канал принтера
ADD     HL,DE
LD      DE,#1B5E
EX      DE,HL
LD      BC,#0004
LDIR
RES     4,(IY+1)
;признак стандартной
;конфигурации

EI
JP      #3D00
;RANDOMIZE USR 15616
```

Теперь для запуска TR-DOS не нужно по буквам набирать RANDOMIZE USR 15616, а достаточно нажать клавишу ENTER. Пункт Tape tester исключен, ведь если есть дисковод, тестирующая программа может быть загружена с диска.

****

ФОРУМ - ИГРЫ

ИФК: Роман Сандаевский из Москвы был необычайно удивлен, обнаружив в игре **DUN DARACH** целое море текста. Он просит рассказать об игре подробнее. Роман, мы советуем Вам обратиться к нашим старым номерам, например почитайте ZX РЕВЮ 94/2 - там Вы найдете ответы на все Ваши вопросы. (Раздел "ADVENTURE GAMES", стр 62).

****

ИФК: Также Роман прислал нам описание своих впечатлений от игры **MURRAYMOUSE (SUPERCOP)**, созданую фирмой CODEMASTERS в 1992 году.

Корр: Эта привлекательная игрушка появилась сравнительно недавно и является одной из последних работ фирмы CODEMASTERS. Она должна понравиться и взрослым и детям, так как в ней соблюдается логическая цепочка событий, сдобренных изрядной долей юмора. Внешне игра чем-то напоминает всеми любимого Dizzy с совершенно новым уровнем графики. На этот раз главным героем будет симпатичный мышонок-полицейский в огромных кроссовках. Зовут мышонка Мюррей.

Сюжет игры таков: Храброго мыша засылают на опасную миссию по спасению сырных миров. Ему противостоит мафия, расположившая свою базу на сырной луне. По ходу игры, параллельно с выполнением определенных действий, Вы должны еще арестовать коварных заговорщиков с невинными мордашками.

Слева от игрового экрана, по вертикали сверху вниз, расположена надпись SUPER+COP. Во время ареста одного члена банды, одна из букв закрашивается в желтый цвет (отдаленно напоминает сбор монет в Dizzy, не так ли?). Когда все буквы пожелтеют, игра закончится. В игре существует масса ловушек и секретов. В некоторых комнатах, чтобы освободить себе путь, нужно аккуратно задеть за рукоятку факела. В игре также предусмотрены подсказки и намеки на какие-нибудь действия. Ваши собратья летучие мыши - иногда не прочь поделиться с Вами полезной, а иногда и бесполезной информацией.

Я не стану описывать здесь прохождение игры, чтобы остальным тоже было интересно. Скажу лишь, что бандиты появляются только в некоторых, строго определенных местах.

****

ИФК: На ошибку в другой CODEMASTER'овской игре, **WILD WEST SEYMOUR**, указывает Максим Бугрий из г.Старый Оскол.

Корр: В той самой первой части, которую не прошел S.O.N., посадка бобов в горшок ничего не дает, а чтобы пройти эту часть, нужно знать ее цель, которую Вам сообщит водитель автобуса. Не поленитесь, и переведите все, что он скажет. А вообще-то все остальные задачки в этой игре просты и линейны, сложность сюжета явно уступает хорошей графике и юмору.

****

Корр: В **SIM CITY** не надо забивать нулями цены и тарифы. Имеющиеся у Вас деньги находятся по адресу #8F97 (только для обладателей теневых мониторов).

****

Корр: HERO QUEST. Для того, чтобы поправить свое состояние, отправляйтесь в THE STONE HUNTER и там используйте заклинания, чтобы восстановить жизни персонажей. Кроме того, из-за промаха авторов 100 монет каждому дадут даже в том случае, если Вы сразу уйдете из лабиринта, не выполнив своей миссии.

ИФК: Тему **HERO QUEST** продолжает Павел Шарыгин из Екатеринбурга. Он спрашивает, зачем нужна святая вода и набор инструментов. Сразу отвечаем, что святая вода никакого видимого эффекта ни на противников ни на героев не оказывает. Что же касается набора инструментов (TOOLKIT), то он необходим для разборки завалов и отпирания замков.

Корр: Почему у Wizard'a в оружейном зале выбираются только STAFF и TOOLKIT.

ИФК: Колдуны опираются в своих поисках только на магическую силу, из оружия им доступен только посох. (В некоторых RPG набор оружия расширяется, но ненамного).

****

ИФК: Также Павел задает вопрос по игре **DIZZY Y**.

Корр: В стране фэнтези я попал в подземелье, а выбраться оттуда не могу. С одной стороны огонь, с другой тупик, и наверх не запрыгнешь - как быть?

ИФК: Мы уже отвечали на этот вопрос на страницах нашего журнала, но читатели вновь и

вновь задают его. Поэтому мы отвечаем еще раз. Чтобы выбраться из подземелья, Вам надо встать на булыжник. Булыжник лежит в пещере, где летает Драконид, в углублении рядом с крестом. Трудность заключается в том, что обычным способом в углубление попасть нельзя туда надо запрыгнуть с определенного расстояния.

****

ИФК: Об отечественной графической адвентюре **APOLLO** фирмы JOKERSOFT рассказывает Андрей Кисилев из Новосибирска.

Корр: Игра на русском языке, она содержит предысторию и инструкцию. Хочу дать несколько советов по прохождению игры.

Чтобы убить первого киборга, нужно лечь и выстрелить ему в шею, правда, с первого раза его не убьешь. Осмотрев убитого врага, мы сможем взять ключ, бронепанцирь и бластер. Хочу предупредить без ключей Вы никуда не пройдете, забирайте их у каждого киборга.

Второй киборг имеет тот же набор предметов, что и первый. Вернитесь к Сэму и дайте ему панцирь и бластер. Теперь Вы можете не переживать за его жизнь. Если первую разновидность киборгов можно убить достаточно легко, то чтобы убить другой вид, нужно выстрелить сначала в руку, затем в ногу.

Скоро Вы убьете киборга, у которого не окажется ключа. Нужно его два раза перевернуть - и ключик у Вас. Имея этот ключ, можно пройти глубже в лабиринт корабля, где Вам встретится третий, самый мощный вид противников. Выстрелами из бластера их не убить, придется использовать гранату. Дырку в скафандре можно залепить пластырем из аптечки. Хотя гранату можно взять дважды (после смерти первого киборга вернитесь на склад - она опять будет лежать в ящике), но на второго врага ее использовать не удается. Как с ним поступить, я не знаю. Помогите!

****

ИФК: Наш давний автор, Илья Командин из Калуги (ZX-MANIACS) рассказывает об игре **CLIFFHANGER**.

Корр: Жанр программы я бы определил как смесь теста на глазомер и реакцию с логической игрой. Суть в следующем: некий ковбой в белом, которым управляете Вы, должен уровень за уровнем, задание за заданием (на каждом уровне несколько заданий) уничтожать ковбоя в черном. Делает он это с помощью всевозможных камней, грузов бомб и т.п. (действие, по всей видимости, происходит на съемках трюкового фильма). Задания варьируются от абсолютно тривиальных (Вы стоите на горе рядом с большим валуном, ковбой бежит по дороге под Вами, и Ваша задача - столкнуть на него валун) до самых сложных (Вы находитесь на экран выше противника и о его передвижениях можете судить только по радару. Вы должны столкнуть бомбу в вагонетку, та подъедет к отверстию в центре экрана. Теперь Вам надо в строго определенной момент нажать на рычаг, и бомба упадет прямо на голову черного ковбоя).

Заговорил же я об этой игре вот по какой причине: мне интересно, проходил ли кто-нибудь дальше шестого уровня? Некоторые задание на нем мне кажутся заведомо непроходимыми. Например, в задании SEESAW'S REVENGE, когда Вы прыгаете на рычаг, камень, вместо того, чтобы лететь в ковбоя, падает прямо на Вас. В задании же GETTING YOUR OWN BACK, бумеранг, возвращаясь, убивает Вас независимо от того, попали Вы в черного ковбоя или нет. Может быть, все-таки что-то можно сделать?

****

ИФК: Далее Илья Командин вместе со своим другом, Андреем Чеботаревым, освещают новую версию игры **LASER SQUAD** с семью уровнями (АИ 16)

Корр: В отличие от всех предыдущих бесплодных попыток создать более-менее приличные дополнительные уровни, эта увенчалась успехом. Качество новых четырех уровней достойно всяческих похвал. Сейчас мы расскажем о новшествах.

CYBER HORDES

В этой миссии Вы управляете отрядом из 8 человек, который должен защищать базу от вторжения роботов. Цель роботов - уничтожить все генераторы, тут и там разбросанные по базе. Вы же должны помешать им сделать это, то есть Ваша задача - уничтожить всех роботов. Всего на базу вторгается пятеро врагов. Среди них: 2 PATROL DROID'a (довольно слабые и очень быстро перемещающиеся), 2 CYBER DROID'a (по силе примерно равны Вашим людям) и один COMBAT DROID, который и является основной ударной силой врагов. У этого дроида броня колоссальной мощности. Спереди его не берут ни серия прямых попаданий из HEAVY LASER'a, ни меткая очередь из AUTOCANNON. Уничтожить его можно, либо долго стреляя в спину из какого-нибудь мощного оружия, либо попав в него или на клетку позади него гранатой. (Да и то, с первого взрыва он может не погибнуть). Если же граната взорвется перед роботом, то это даже не покоробит его броню.

Если Вы уже испугались, то смею Вас огорчить это еще не самое страшное. Хуже всего то, что сильные роботы обладают способностью производить более слабых (вместо уничтоженных). Если же Вам удастся уничтожить COMBAT DROID'a, то новый экземпляр появится где-то ходов через 10 у места старта. Так что, Вам придется уничтожить всех более слабых роботов за эти 10 ходов.

PARADISE VALLEY

Эта миссия - самая замороченная. Ваш отряд должен пробраться через джунгли неизвестной планеты и установить на некий бакен (на реке) защитное устройство. Сразу скажем, что найти этот бакен нам так и не удалось, так что цель миссии мы знаем только по слухам.

Вся карта разделена на две части: джунгли и шахты, причем пройти из одной части в другую нельзя. По карте разбросаны сундуки зеленого цвета. Если открыть сундук, расположенный на самом верху поля, недалеко от места

старта, то Вы обнаружите в нем голубой ключ. Он нужен, чтобы открыть голубые ящики в запертом бункере рядом с рекой. Но вот незадача дверь в бункер никак не хочет открываться.

В игре существует, система телепортов. Только с помощью этой системы можно попасть в шахты. Точка телепортации ничем не отличается от обычной земли. Первые два телепорта находятся прямо перед стартом это два квадрата из кустов размером 3x3 клетки с пустым центром. Еще два телепорта можно найти в углублениях скал в центре карты. Один из этих телепортов ведет в комнату с закрытой дверью, находящуюся в шахтах. В этой комнате стоит сундук с пурпурным ключом, которым можно отпирать любые двери.

А шахтах также есть 2 телепорта (углубления в стенах). Один из них, находящийся радом с комнатой, где лежал пурпурный ключ, ведет в комнату с голубыми сундуками. Одному или нескольким Вашим солдатам стоит отправиться туда (обязательно взяв с собой оба ключа). В сундуках Вы найдете неплохой боекомплект: 4 MS AUTOCANNON и 4 мощных гранаты AP-75.

Второй телепорт ведет на крохотный островок посреди реки.

В шахтах противостоять Вам будут два SECTOID'a, которые, в принципе, особенной опасности не представляют. А вот на реке живут громадные лягушки, которые не только очень крепкие и точно стреляют (непонятно только, из чего), но еще и постоянно нарождаются взамен убитых.

STAR DRIVE

Группа инженеров захватила ценное устройство STAR DRIVE (часть межзвездного двигателя) и забаррикадировалась в базе с бронированными дверями и стенами. Ваша группа была послана с целью отнять устройство.

При выборе оружия обязательно выдайте одному из своих солдат лазерный резак (LAS-CUTTER), так как только с его помощью можно вскрыть бронированные двери базы.

У верхнего и нижнего краев карты есть как бы два выхода за ее пределы. Оттуда появляются два боевых дроида (как в миссии ASSASSINS). Если их уничтожить, то через 3-4 хода они появятся вновь. Лучше всего оставить по одному из своих солдат с каким-нибудь многозарядным оружием (например, HEAVY LASER'ом) возле мест появления дроидов, чтобы они отстреливали врагов по мере их появления.

Пробравшись в базу, надо действовать осторожно. Убив одного из инженеров (он отличается мощной защитой), Вы найдете в его карманах STAR DRIVE. Взяв его, Вы должны вернуться к месту Вашей высадки и подойти вплотную к левому краю карты - миссия выполнена.

LASER PLATOON

Идет война между войсками повстанцев и федерации. Вы руководите битвой за космическую базу важную стратегическую точку для обеих сторон.

База поделена ровно на две части: верхняя сторона Ваша, нижняя противника. В левом верхнем и правом нижнем углах

расположены коридоры, уходящие за пределы карты. Оттуда к Вам и к врагу будет приходить подкрепление по мере уничтожения солдат начальной группы. К сожалению, Вы играете за повстанцев, и у солдат федерации есть преимущество - в их распоряжении находится один боевой дроид (как в миссии CYBER HORDES). Вашей задачей является уничтожение всех солдат противника так быстро, чтобы новые просто не успели появиться. Это не так просто сделать, учитывая, что новый солдат на смену убитому появляется ходов через 4-5, а новый робот - через 10-15.

При расстановке солдат поставьте некоторых из них в коридоры, соединяющие верхнюю и нижнюю половины базы и, особо не дергаясь, за 2 хода Вы уничтожите 3-4 федералов. Постарайтесь постепенно сжимать кольцо вокруг места появления вражеских бойцов (но будьте осторожны, не оставьте солдат из начальной группы противника у себя в тылу). Подозреваю, что боевой робот федерации сможет пробиться через Ваши кордоны, но, если Вам все-же удасться его остановить, то всеми силами захлопывайте "котел", и обращая внимание в основном на время, а не на потери, уничтожайте оставшиеся силы врага.

****

ИФК: О программе **MOVIE** спрашивает А.В.Мунгалов из г.Тулун.

Корр: Вот что мне удалось обнаружить в этой игре. Все персонажи разговаривают стандартными фразами, что наводит на мысль посмотреть в мониторе словарь игры.

Попугай в основном повторяет сказанное Вами, но иногда говорит что-то свое. На вопрос "кто ты?" большинство отвечает, что мне надо найти пароль (YOU WILL NEED A PASSWORD). На вопрос, где взять пароль, некоторые говорят, что это часть головоломки или извиняются, и говорят, что не могут помочь.

Долго пробродив по городу, я обнаружил две странные улицы. В конце одной из них (если идти наверх), я нашел комнату, похожую на церковь или на какой-то зал заседаний. Там множество стульев. Здесь лежит какой-то непонятный предмет.

Если погулять по городу, можно обнаружить девушку, которая поддерживает с главным героем разговор и, как бы между прочим, говорит, что она замужем. Потом она начинает идти из комнаты в комнату. Мне ничего не оставалось, как пойти за ней. Неожиданно, в комнате с двумя громилами, она начинает кричать "kill, kill, kill", а потом настойчиво просит, чтобы я их убрал. Дальше из комнаты она никуда не идет, повторяя одно и то же "ELIMINATE THEM!"

Предметы. Из предметов мне встретились пистолет и пузырек с какой-то жидкостью. Если бросить последний о стену, то он разобьется с громким хлопком. Еще была найдена коробка, перевязанная лентой и пара совсем непонятных предметов.

Вопросы по игре:

1. Какова главная цель?

2. Что это за женщина, и что с ней делать?
3. Где можно применить предметы и где взять патроны к пистолету?
4. Как достать предметы, лежащие высоко?
5. Для чего в меню нужны две последние пиктограммы - Н и А?

****

ИФК: Новую возможность в Кладовской версии **ELITE** обнаружил Е.В.Селиверстов из г.Старый Оскол.

Корр: Наверное те, кто занимался разведением минеральной жизни знают, что гоняться за астероидами по всему космосу не очень приятно, гораздо лучше было бы подлететь к здоровенной куче и набить полные трюмы минералов. Так вот, это не плод воспаленного воображения шизофреника, это вполне возможно при условии, что Вы обладаете COBRA MK-4 и не боитесь стать FUGITIVE. Итак, Вы имеете на борту минеральную жизнь, и Вам нужны только астероиды и ничего кроме астероидов. Хорошо. Вылетаем из станции и волочем ее за собой в пояс астероидов до тех пор, пока планета не превратится в точку. Подходим к станции со стороны входа и делаем по ней пару залпов. И тут происходит самое интересное вместо VIPER'ов со станции стартуют... астероиды. Вуаля, набирай сколько хочешь.

А вот второй жучок. Когда я только начинал играть, меня перехватили таргоны. У меня был только расширенный грузовой отсек и почти 1000 кредов. Естественно, против шести таргонов я ничего не успевал сделать. Ладно, перегружаюсь, жму CAPS SHIFT + F, делаю гиперпереход и опять та же картина. и все-таки, помучившись, я нашел выход. При гиперпереходе жмем J и на выходе по инерции проскакиваем таргонский кордон - впереди открытое межзвездное пространство, абсолютно пустое. Ориентируясь по компасу, влетаем в ближайшую звездную систему. Все. Спасен.

****

ИФК: Способ прохождения игры **WHERE TIME STOOD STILL** описывает Вячеслав Ситников, проходящий сейчас службу в г.Комсомольск-на-Амуре.

Корр: На втором плато (там, где деревня индейцев), лежат инструменты и динамит. Не знаю как инструменты, но динамит понадобится, так что возьмите его.

Одно найденное мною случайно обстоятельство. Зеркальце (или пудреницу) надо отдавать индейцу следующим образом.

Подойдите к нему всей группой, но близко не надо. Тот, у кого зеркальце, должен стоять так, чтобы между ним и индейцем стоял еще кто-нибудь. Затем выложите зеркальце. Если Вы стоите не слишком близко, то индеец попробует подойти и взять его, но Вы тоже сделайте то же самое. То есть, Вам надо выложить зеркальце и тут же взять его. Затем возьмите подарок вождя мешочек. Походите немного. Заметили? Вождь ходит за Вами как собачка, то есть одним Вашим игроком стало больше, вот только переключиться на него нельзя, да это и не надо.

Теперь насчет руки, которая высовывается из скалы. Когда Вы дадите ей мешочек, она исчезнет на 1-2 секунды. За это время можно провести Джаррета, но если за ним идет еще кто-то, то пройдет и он и тот, кто идет за ним. Главное, чтобы вся группа шла вместе. Можете остановиться, постоять - рука не высунется, пока не уйдет последний.

Дальше, пройдя водопад, Вы попадете на третье плато. Вот тут-то Вам и пригодится вождь. Надо подойти Джарретом к краю спуска так, чтобы вождь был впереди него, а все остальные - сзади. Как известно, неуправляемые персонажи живут в игре своей жизнью - бегают, прыгают, спят и т.п., но ни один из них не будет прыгать с обрыва, если там высоко.

Потихоньку спускайтесь. Когда индеец сочтет, что расстояние до земли достаточное, он спрыгнет вниз, а Вы поднимитесь чуть выше - чтобы его примеру не последовали остальные. Внизу индейца будет поджидать противное пресмыкающееся. Оно захочет плотно пообедать, но индейца так просто не съесть он начнет бегать вокруг и забрасывать динозавра копьями. Еще один меткий бросок, и о страшном чудовище напоминают только белеющие кости, а индеец помчался к реке. Все. Больше Вы его не увидите, а если побежите за ним, то не заметите ничего интересного он просто стоит у берега. Если Вы уйдете с экрана, то он исчезнет.

Дальше решайте сами, что делать, скажу только, что Вы встретите и что сделал я. Вы увидите стойбище диких индейцев, кладбище, какой-то мост с индейцем, кстати, настроенным мирно, в отличие от всех остальных. Можете снять с моста сияющий шар, зачем только - я не знаю.

В правом верхнем углу есть тропинка наверх. Поднимайтесь по ней, придерживаясь *левого края, потом идите в ответвление слева - на следующее плато. Там пройдите вверх (осторожно, там иногда водятся динозавры) - Вы увидите подъем, который перегораживают два валуна. Вот здесь Вам и пригодится динамит (после использования отбегите подальше и прижмитесь к стене камни прокатятся мимо и не заденут Вас). Теперь поднимитесь наверх и зайдите в пещеру.

****

ИФК: Дополнительную информацию об игре **THE SCEPTRE** прислал нам Александр Феклистов из г.Челябинск.

Корр: Чтобы выиграть, Вам надо как можно быстрее нейтрализовать вражеских мистиков. Если же Вашего мистика заколдовали, то выиграть, я считаю, вообще невозмозможно. Если Вы превратите колдуна противника в камень, его можно потом начать "уговаривать" присоединиться к Вам. Если он согласится, то расколдовать. Предметы в начале игры расположены статично, а чужие персонажи их практически никогда не берут. Таким образом, найдя в каком-то месте предмет, Вы найдете его там и в следующий раз. каждый предмет обладает свойствами, влияющими на способности воинов (сила, ловкость и т.д.). Как изменить характеристики рабов я не знаю, а вот вои-

нам надо дать команду "USE название предмета" Если этот предмет у него был, то изменится или появится одно из свойств воина. Так, например, почти все кристаллы делают воинов неутомимыми, мечи сильными, а DARK SWORD - безжалостными. Однако, есть и более сложные комбинации. Например, если использовать SHING или GOLDEN CRYSTAL'ы, воин станет неутомимым (TIRELESS). Если же использовать сначала SHINING, а затем GOLDEN, то получите свойство OBEDENT. При использовании предметы остаются, и их можно передать другому персонажу. Если у Вашего воина уже есть предмет, а Вы даете ему еще один, то старый он бросит где-нибудь неподалеку. Некоторые предметы добавляют сразу два свойства.

Интересна команда GRAB <полное название предмета>. У нее три варианта.

1. Если предмет находится на земле, то воин просто его подбирает, и команда снимается.

2. Если предмет находится у кого-то из людей (пусть даже у Вашего), то воин находит его и вступает в схватку за право обладания данным предметом.

3. Если предмет уже находится у этого воина, то загорается надпись "в битве"; воин делает несколько шагов назад, останавливается, надпись гаснет и все повторяется сначала. Это очень удобно, когда надо изменить направление движения на противоположное, что можно сделать только на ограниченном числе локаций.

****

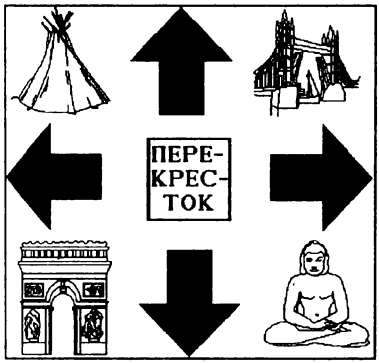
ИФК: Советы по игре **CARRIER COMMAND** прислал нам, забывший подписаться в этом разделе, читатель.

Корр: Не следует улетать на самолете или уплывать на катере слишком далеко от авианосца, так как по экрану начинают идти помехи (по-видимому, теряется связь с авианосцем), но управление при этом не ухудшается. Также не следует пытаться посадить самолет на авианосец, когдапоследний движется на большой скорости самолет либо разобьется при посадке, либо упадет, либо топливо кончится.

****

ИФК: Инересный "жучок" нашел в программе **MATCH DAY** Михаил Букреев из г. Озерск Челябинской обл.

Корр: Из-за ошибки программистов можно легко пройти всю игру. Дело в том, что когда Вы поворачиваетесь спиной к воротам компьютера, то игрок противника, который только что хотел забрать у Вас мяч, отойдет от Вас на определенное расстояние и так и останется стоять. Можно простоять в таком положении всю игру, и никто на Вас не нападет, но для выигрыша нужно забить хотя бы один гол. Развернитесь в сторону ворот противника. Игрок компьютера начнет подходить к Вам, а Вы сделайте маленький шаг к нему и снова отвернитесь. Компьютерный игрок опять отойдет, а Вы сможете вновь развернуться и немного подойти. Таким образом, можно потихоньку дойти до ворот и забить гол.



Евгений из г.Новочебоксарск описывает свой опыт в игре **MATT LUCAS FLORIDA DETECTIVE**. Евгений пишет, что игра не слишком сложная, а основные трудности возникают из-за бедного словаря программы. Ему удалось сделать следующее: в одной из локаций, изучив письменный стол, мы обнаружим ящик (DRAWER). Если его открыть, найдем шпильку (HAIRPIN). Возможно, с ее помощью можно открыть сейф или запертую дверь. Чтобы войти в квартиру к Joe, надо постучать (KNOCK). Внутри, повернув картину, Вы найдете ключ зажигания (IGNITION KEY), применить его не удалось. В одной из локаций есть неуказанный проход на север, пройдя туда, Матт оказывается в баре и становится свидетелем интересного разговора. Кроме того, на складе (WAREHOUSE) необходимо поспать (SLEEP). В игре есть еще много предметов, но применить их не удалось. Требуется помощь.

Евгений спрашивал, как можно перевести слово PUSHER. К сожалению, мы не знаем всего предложения с этим словом, поэтому не можем подобрать точный перевод. Вообще-то PUSHER это толкатель, или эжектор. Также называют самоуверенных, наглых людей, а в авиации PUSHER - самолет с толкающим винтом.

В игре **HUMAN TORCH AND THE THING** Евгений попробовал сделать следующее: когда THING залез в пушку, он бросил пушку в яму. При этом THING не погибнет, а так и будет сидеть внутри. Однако, при команде вылезти из пушки, игра сбрасывается.

****

В.Индюков из г.Новокузнецк, описывает российскую адвентюру **ЭКЗАМЕНАЦИОННОЕ РАССЛЕДОВАНИЕ** (АИ 19).

Итак, Вы оказываетесь на улице, у Вас ничего нет с собой. Если тут же залезть на столб и хорошенько осмотреться, то можно стать обладателем электрической лампочки. Если перелезть через забор, мы окажемся во дворе склада, где надо взять ящик. Вернувшись к месту старта и пройдя на юг, снова перелезаем через забор, осматриваем груду кирпичей и берем оттуда один из них. На север от места старта даем следующие команды: бросить ящик, залезть на ящик, разбить окно, открыть окно. После этого смело идем на запад и попадаем на кухню, где среди кухонной утвари находим нож. На юге дверь на улицу (ведет к месту старта) и вход в спальню. На кровати в спальне находим подушку, которую потрошим командой "разрезать подушку" В доме и окрестностях можно найти

еще и молоток, отвертку и антенну. С помощью отвертки можно вывернуть винты сливного бачка в унитазе внутри мы найдем дневник с изображением птицы.

Открыв раму, попадаем во двор, где в мусорном баке находим мясо для собаки. С мясом можно спокойно проходить мимо нее. На клумбе растет тюльпан, что с ним делать и вообще, что делать дальше непонятно. Наш корреспондент попытался открыть ворота склада ключом, вынутым из подушки и наткнулся на сообщение "ворота заперты" К сожалению, он не написал, какой именно фразой он пользовался. Возможно, стоило поэкспериментировать с глаголом "отпереть" и не забыть "с помощью ключа"

Продолжает тему отечественных программ игра **ВИТЯЗЬ СВЕТОГОР** (АИ 19). Здесь наш корреспондент только начал свои поиски. На востоке от места старта я нашел камень с колдовскими рунами, прочесть которые не удалось. На западе роща. Если войти в нее и пройти несколько экранов на юг, а потом два на запад, то мы увидим родник со скрытой в нем богатырской силой. Попив из родника, не получаем абсолютно ничего. На севере от места старта дракон, охраняющий вход в царство Карачуна. Если перед этим экраном пойти на запад, увидим цепных псов. Ничего с этими персонажами сделать не удалось.

****

Исследование гениального **SNOWBALL**'а развивает Александр Рыбин из г.Саранск, Мордовской республики. Он пишет, что в локации 8200 (рядом со скальпелем), нужно лечь (LIE) на кушетку и отдохнуть. После этого Вам сделают массаж, прибавят 50 очков, и Ваша грузоподъемность значительно возрастет.

Банку с краской нужно положить в TOOLBOX, а затем его закрыть. Теперь краске вакуум не страшен. После этого идем в локацию с WALLDROID'ом (11400) и открываем TOOLBOX. Краска испаряется и WALLDROID испаряется вместе с ней.

Поднимаемся наверх, отыскиваем BOMB. Кот-пылесос очищает лестницу от грязи (DROP CAT). Но дальше Т-образного коридора (локация 12800) официально пройти не удалось. Что делать с лазером - непонятно.

По-видимому, все дело в бульдозере, иначе зачем его ремонтировать и заводить - ведь до локации 12800 можно дойти и так. Изменив байт в отгрузочном блоке, Александр "прошел" лазер. Оказалось, что до конца игры рукой подать. Осталось пройти только одно препятствие, где Вам пригодятся огнетушитель и электрофлейта.

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

Дорогой Читатель!

Книгу, которую мы начинаем публиковать в данном номере ZX РЕВЮ, автор прислал нам давно. Очень трудно нам было найти возможность ее публикации в виде отдельного издания довольно большого объема. С другой стороны, тема книги, ее прекрасный стиль, разносторонняя эрудиция автора, его увлеченность... не давали нам спать спокойно - эта книга была обязана увидеть свет!

И вот, наконец, настал момент, когда принято принципиальное решение о существенном увеличении объема ZX РЕВЮ 1997 года!

С большим удовольствием представляем Вам автора: ВЛАДИМИР ЛЕОНИДОВИЧ ОВЕЧКИН и его книгу: ПЕРЕКРЁСТОК ДРАКОНОВ, которую мы будем публиковать в течение 1997 года в ZX РЕВЮ и которая, как нам кажется, станет украшением его игрового раздела.

Итак, отложите все деле, устраивайтесь поудобнее... мы начинаем.

ВВЕДЕНИЕ

Случилось это в те давние-давние времена, когда программы были маленькими-премаленькими, а компьютеры для которых они предназначались большими-пребольшими и назывались ЭВМ (электронно-вычислительными машинами). Были эти самые ЭВМ не только огромными, но еще и жутко дорогими, а потому встречались они крайне редко, работали круглыми сутками, использовались для решения самых-самых серьезных задач и были окружены завесой тайн и легенд.

Вот так и случилось, что изрядно устав от всевозможных заумных формул и расчетов, один чудак-программист в свое свободное время "не славы и денег ради, а удовольствия для", взял и придумал первую на свете игровую программу, назвав ее немудреным словом "***Adventure***" ("Приключение").

Люди, обслуживающие эти машины и составляющие для них программы на никому кроме них неведомых языках, были умными-преумными и серьезными-пресерьезными. Они очень гордились своей редкой профессией и были буквально влюблены в своих электронных монстров, порой не желая расставаться с ними даже на уик-энд.

Так как никакой графики ЭВМ еще не знала, вместо рисунков этот человек применнил краткие словесные описания различных мест (потом их стали называть локациями), по которым можно было "перемещаться", давая машине ту или иную команду. А чтобы скитания не были бесцельными он "спрятал" в программе множество всевозможных реальных и фантастических предметов, которые поставил охранять мифических стражей-монстров. Свое произведение программист не стал хранить в тайне (честь и хвала ему за это!), а предложил на суд своим коллегам, и те в часы досуга с увлечением принялись за поиски.

Идея настолько понравилась, что ей поделились с другимию Так оно и пошло, и пошло, и пошло...

Давно уже забыто имя того программиста. Давно ЭВМ превратились из огромных электронных монстров в маленькие персональные и даже домашние компьютеры, умеющие не только считать с невероятной скоростью но даже рисовать, говорить и петь. И давно где-то затерялась самая первая версия "***Adventure***" Но сама идея текстовой игры не только не умерла, а даже наоборот набрала огромную силу, воплотившись в сотни и тысячи разнообразных программ, причем самое огромное количество таких программ написано для нашего с вами маленького, уютного и домашнего-предомашнего ***Speccy***.

Тематика программ жанра ***Adventure*** невероятно разнообразна. Тут и традиционные поиски сокровищ; и спасение людей, стран и даже планет; всевозможные детективные и фантастические истории... Все даже перечислить сложно, ну а о том чтобы обо всем этом рассказать в один прием и думать не приходится. Именно поэтому в той книге, что вы сейчас держите в руках, речь пойдет только о самых традиционных для "адвентюр"[1] вещах: о поисках сокровищ и различных легендарных предметов, о спасении сказочных персонажей и фантастических государств, то есть о том, что частенько называют термином **D&D** (***Dungeons & Dragons = Подземелья и Драконы***). Благодаря этому термину появилось и название самой книги - "**Перекресток Драконов**"

Во всех программах **D&D** действует один главный герой, которым вы и управляете. Цель игры всегда конкретна, а путь ее достижения целиком зависит от игрока. И никогда дело не обходится без магии. Интересует вас это? Если - да - то, очень надеюсь, что эта книга хоть чуть-чуть вам пригодится, а кому-то, быть может, поможет открыть для себя новый, полный загадок и приключений мир, где в вечной борьбе добра и зла, вы всегда будете на стороне добра. Ну а кто победит в этой борьбе, будет зависеть от вас, и только от вас!

Не надейтесь найти в описаниях ответы на все вопросы поставленные той или иной программой для этого пришлось бы выпустить не одну весьма солидную книгу. Вот что, например, писал о программе "***The Wrath of Magra***" журнал "***Sinclair User***": "Это огромная игра, и не хватит целой жизни, чтобы полностью пройти ее." Можно оспаривать это мнение, но один из моих знакомых (весьма и весьма компетентный человек во всевозможных компьютерных играх) потратил вот уже несколько месяцев на игру "***Doomdark`s Revenge***" При этом он все свои скитания не ленился заносить на составляемую по ходу дела карту. К настоящему

---

[1] Не удивляйтесь, что слова "адвентюра" и "адвентюрный" в тексте книги будут браться в кавычки. Именно в такой транскрипции этот термин вошел в нашу жизнь со страниц журнала "***ZX-Ревю***", но на самом деле английское слово "***Adventure***" звучит несколько иначе, и чтобы наша терминология не резала слух сторонникам чистого английского, пусть "адвентюра" и "адвентюрный" будут в кавычках.

времени его карта занимает по площади более одного квадратного метра (каждая локация на ней нанесена квадратиком в 1 см), а конца игре пока еще не видно. Представьте теперь, что все свои странствия он решил подробно описать, добавьте к этому рассказ о самых важных загадках и их решениях, десяток-другой полезных советов, небольшой словарь - вот вам и впечатляющий фолиант обемом в несколько сотен страниц. Только стоит ли писать такой труд и найдутся ли у него читатели?

Вообще, вопрос необходимости описаний к "адвентюрным" программам является довольно спорным. Во всяком случае, все тот же мой знакомый очень категорично заявил, что описания для адвентюрных игр не только не нужны, но даже вредны! Вряд ли он прав на 100%. Если обратиться к западным журналам или к нашей литературе по компьютерным играм, мы найдем много доказательств обратного. Так давайте не будем слишком категоричны: описания, пусть даже весьма краткие нужны хотя бы потому, что далеко не все программы имеют встроенные инструкции, а играть не зная конечной цели, занятие не слишком интересное. Нашим западным коллегам было попроще - большинство программных продуктов попадавших к ним в руки имели в своем составе красочные буклеты, в которых авторы программ не только объясняли конечную цель игры, но и давали некоторые минимальные подсказки, а порой, если находили это нужным, печатали даже карты. Увы, но в России рынок программного обеспечения для ***Spectrum*** развивался совсем по другим законам, а точнее, вне всяких законов об охране авторских прав. Именно поэтому, мы с вами оказались лишенными не только буклетов, но даже самих программ в их первоначальном виде. Очень часто игры, которыми мы с вами пользуемся прошли через столько рук, и столько хэкеров "потрудилось" над ними, что больно бывает смотреть на то, что осталось от некогда превосходных программ. Чаще же всего от рук хэкеров страдали заставки программ и инструкции (!). Вот вам и еще один, увы не слишком веселый, довод в пользу описаний.

Однако, слишком подробные описания, состоящие из сплошных: пойдите туда, возьмите это, сделайте то, наверное, действительно не нужны. Подумайте сами, ведь они не только отнимают у игрока какую бы то ни было инициативу, но и просто-напросто убивают всякий интерес к игре, лишая множества интереснейших открытий.

Именно в силу всех этих соображений, описания игр, помещенные в этой книге носят характер рекомендаций, а не догм. Чем-то они могут показаться сродни "***Дебютам***" электронного журнала "***Спектрофон***" Но это ни в коей мере не копирование стиля этого, столь любимого многими синклеристами, журнала. Наверное, просто так уж сложилось исторически, что мы с ними разными путями пришли к одному и тому же выводу - да, описания игр нужны, но нужны они как руководство к действию, как краткий справочник, но ни в коем случае не как длинное и нудное смакование открытий и достижений автора описания!

Поэтому, читая книгу вы найдете еще множество неразгаданных загадок и нерешенных задач. Многие достижения и открытия, сделанные

автором, могут показаться вам спорными, а в некоторых случаях даже ведущими в тупик. Ну и что с того? Не нравится? Так выберите свой путь! Каждая "адвентюра" - это прежде всего ваше собственное приключение и вы вправе пройти его так, как сочтете нужным, опровергая, если того потребует игра, мнения любых авторитетов. Так что ищите свои решения, ошибайтесь, отступайте и снова идите вперед - в этом и только в этом смысл и ценность любой "адвентюрной" игры.

Даже если вы самостоятельно не справитесь с какими-то затруднениями, ответов на которые не будет на этих страницах, всегда есть возможность посоветоваться со своими друзьями, а в самом крайнем случае обратиться в раздел "***Перекресток***" журнала "***ZX-Ревю***" Кстати, нечто подобное "***Перекрестку***" было и у западных поклонников ***Speccy***: например, в рубрике "***At The Sign of The Dancing Ogre***" журнал "***Sinclair User***" регулярно публиковал советы по выходу из самых затруднительных ситуаций программ ***Adventure***, открытия наиболее упорных исследователей игр этого жанра, вопросы, ставившие в тупик не только игроков, но и авторов журнала. Теперь возможность колективной борьбы с проблемами ***Adventure*** есть и у нас, так почему бы ей не воспользоваться, если не придумать никакого другого выхода?

Кстати, пока писалась эта книга (а она задумывалась как плод коллективного творчества, а отнюдь не как сольное произведение), и пока автор "сражался" где с обилием информации, а где с ее нехваткой, родилась еще одна идея: а почему бы не сделать книгу подобную по своей сути уже упоминавшемуся "***Перекрестку***"? Если эта мысль найдет у вас отклик, попробуйте поддержать ее конкретными действиями - присылайте в "Инфорком" (адрес, если вы его еще не знаете, можно найти в конце книги) письма с вашими "адвентюрными" открытиями, проблемами, нерешенными загадками. Пригодятся и карты ваших странствий, и словари игр и все то, что на ваш взгляд сможет пригодиться остальным "адвентюрщикам" А чтобы ваша информация поскорее попала по нужному адресу не забудьте сделать на конверте пометку -"Перекресток Драконов" Тогда, возможно, общими стараниями увидит свет какой-нибудь "Звездный Перекресток", "Перекресток Драконов 2" или нечто похожее и, надеюсь, гораздо лучшее!

Кроме описаний конкретных игровых программ, в книге вы найдете страницы посвященные таким общим для всех "адвентюр" вопросам, как необходимость и способы рисования карт игрового пространства; поиски словарей программ; "бессмертие" и т.п.

А для тех, кому уже стали тесны рамки чужих приключений и хочется создать нечто свое, особенное и прекрасное, предназначен небольшой раздел о современных прикладных программах для создания "адвентюрных" игр. Там же вы найдете несколько, надеюсь, не бесполезных для начинающих авторов советов.

В заключительной части книги приведен список "адвентюрных" программ с указанием (где это оказалось возможным) выпустивших их фирм, примерной оценкой игр по пятибалльной шкале и указанием из-

данной в России литературы (в том числе и электронных журналов), в которой можно отыскать информацию по этим программам. Список литературы дан отдельной таблицей, и порядковые номера в ней соответствуют цифрам, указанным в колонке "Литература" таблицы "***Adventure Software***"

Несколько слов об оценке программ Впервые попробовал как-то справиться с этой проблемой английский журнал "***Sinclair User***", введя в публикуемый ежемесячно на его страницах каталог программного обеспечения "***Software Directory***" колонку "***Gilbert Factor***"("Гилберт Фактор"). Эта оценка программ базировалась на следующем: наличие четкого описания; скорость работы; удобство управления; оригинальность идеи; возможность сохранения текущего состояния игры; качество графики и звука; фиксация лучших результатов. Свое название Гилберт Фактор получил по имени обозревателя журнала Джона Гилберта (***John Gilbert***[2]), который и придумал эту оценку. В этой книге сделана попытка оценить "адвентюрные" программы по тем же критериям. Автор старался быть объектиным, а где это возможно сохранил оценки данные когда-то Джоном Гилбертом. В тех же случаях, когда информация о той или иной программе была недостаточной или противоречивой, оценка не выставлялась. Для большей наглядности оценок можно воспользоваться такой вот небольшой табличкой:

☆ ☆ ☆ ☆ ☆	*Прекрасная "адвентюрная" игра*
☆ ☆ ☆ ☆	*Хорошая игра и неплохо бы в нее сыграть*
☆ ☆ ☆	*Обычная, добротно сделанная, "адвентюра"*
☆ ☆	*Игра особого внимания не заслуживает*
☆	*Без этой игры можно прекрасно обойтись*

Пусть вас не смущает возраст программ[3], о которых рассказывается в книге - устаревших игр не бывает, так же, как не бывает устаревших книг. Вспомните "прятки", "салочки" в них играли ваши дедушки и бабушки, а вы разве не играли? Ну а книги: кто из вас не читал "Колобка" или "Золотой ключик"? С компьютерными "адвентюрами" дело обстоит очень-очень похоже: дедушки и бабушки в них, правда, не играли, но пока жив ***Speccy***, король Артур, викинг Эрик, хоббит Бильбо

---

[2] Постарайтесь не путать обозревателя журнала "***Sinclair User***" Джона Гилберта с "самым известным" взломщиком программ для ***Spectrum*** Биллом Гилбертом (***Bill Gilbert***). С "творениями" последнего вы наверняка уже сталкивались, и еще не раз столкнетесь, каждый раз поминая его не самыми лучшими словами за безжалостно исковерканные программы.

[3] То, что большинство программ, описываемых в книге, вышли в свет в 1984-1986г.г. скорее закономерность, чем случайность. Во-первых, именно в этот период ***Spectrum*** переживал пик своей популярности на Западе, и новые программы для него появлялись почти каждый день. Во-вторых, описания программ, выходивших начиная с 1987г., уже не раз встречались в отечественной компьютерной литературе, так как к тому времени ***Spectrum*** уже перебрался в Россию, а авторы описаний всегда предпочитали программы-новиноки. Так вот и получилось, что самыми малоизвестными у нас оказались программы 1982-1986г.

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

Бэггинс и другие их персонажи будут жить не старея на экранах наших видавших виды мониторов!

DUNGEONS & DRAGONS

ADVENTURE 1	
Приключение 1	
☆ ☆ ☆ ☆	
1982	Abersoft

"Возможно есть рай, и уж точно - ад.
А нам место здесь. Не так ли брат?"
Рэдьярд Киплинг "Конец пути"

К этой "древней" программе уже не раз обращались в книгах, но увы, дальше перевода встроенных в игру инструкции (**INST**), информации (**INFO**) и помощи (**HELP**) дело так и не зашло, а жаль! Авторам программы удалось создать в ней на удивление гармоничный фантастический мир, однажды соприкоснувшись с которым трудно уже будет оторваться. А если вы, впридачу ко всему, еще и поклонник такого жанра литературы как ***Fantasy***, то судьба ваша предрешена - вновь и вновь вы будете пускаться в путь по таинственным подземельям ***Adventure 1*** и, пожалуй, не столько в поисках сокровищ, нахождение которых является конечной целью игры, сколько в стремлении разрешить загадки этого невероятного сказочного мира.

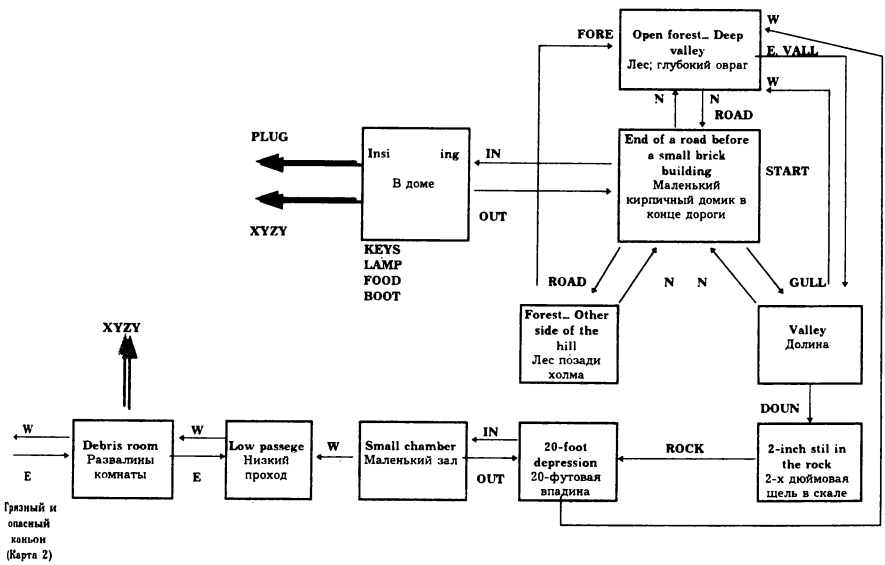
КАРТА I

И так, вы стоите в конце дороги (***End of a road***)[4]. Вокруг лес, а перед вами маленький кирпичный домик, рядом с которым, устремляясь в глубокий овраг, течет быстрый ручей. Где-то совсем рядом находятся несметные сокровища, спрятанные в огромной пещере, но как разыскать эту пещеру - великая тайна, ибо, кто ни отправлялся на поиски подземных богатств, назад уже никогда не возвращался.

Чтобы не повторить судьбу предшественников, давайте возьмемся за дело основательно и, самое главное, не будем спешить. Для начала осуществим первое и вполне естественное желание - войдем в дом. Так как он не заперт, сделать это совсем несложно: наберите **IN**, затем нажмите **ENTER** - и вот вы уже внутри (***Inside building***). А вот и первые находки: большая связка ключей (***a large set of KEYS***), блестящая медная лампа (***a shiny brass LAMP***), немного вкусной пищи (***some tasty FOOD***) и бутылка с водой (***a BOTTle of water***). Не пытайтесь сразу схватить все (**TAKE ALL**): во первых - из этого ничего не получится, во вторых "грузоподъемность" ваша весьма ограничена (7 предметов), и обязательно придется что-то бросать в самом неподходящем месте.

---

[4] Здесь и далее в описании игры и на карте используются сокращенные наименования локаций, поэтому, если вы не хотите упустить чего-либо важного, внимательно читайте и переводите то, что будет сообщать компьютер.



ADVENTURE 1
Карта 1

Возьмем только самое необходимое - ключи (**TAKE KEYS**) и лампу (**TAKE LAMP**)[5]. Теперь можно выходить из дома (**OUT**) и отправляться в путь, но прежде чем сделать это, обратите внимание, что в описании локации комната названа превосходным местом для "большого прыжка"(***Large spring***). Пока эти слова для вас ничего не значат, но будьте уверены, очень скоро смысл их станет понятен, и вы по достоинству оцените подсказку.

Вновь оказавшись перед домом, можно идти на все четыре стороны: пройтись по дороге, зайти в лес, спуститься в овраг и снова очутиться перед домом. Вы даже представить себе не можете, как много путешественников так и не смогло вырваться из этого замкнутого круга. Но взгляните на **первую карту**, и вы без труда найдете верный путь. Прямо от дома надо спуститься в овраг (**GULLy**) и идти вниз по ручью (**DOWN**), пока путь не преградят скалы. Ручей здесь теряется, ныряя в 2-дюймовую щель меж камней (***2-inch slit in the rock***), но он уже вывел вас на верный путь. Забирайтесь на скалу (**ROCK**), и вы окажетесь в мрачной впадине окруженной 20-футовыми скалами (***20-foot depression***). Под ногами чавкает грязь, и прямо в нее упирается крепкая стальная решетка вмонтированная в бетонное основание и запертая на надежный замок. Пускайте в дело ключи (**OPEN**) и, когда решетка откроется, входите в образовавшийся проход (**IN**).

Вы окажетесь в маленьком зале (***Small chamber***), откуда низким каменным лазом идущим на запад начинается подземелье. Несколько шагов вперед (**W**)[6], и вы оказываетесь в кромешной темноте. Двигаться на ощупь не стоит: с вероятностью 100% грохнетесь в первую же яму и переломаете все кости. На первый раз (и даже на второй, но не больше!) программа простит вам эту оплошность и вернет жизнь, но путь придется начинать с самого начала. Чтобы избежать таких последствий достаточно включить лампу (**ON**), однако на будущее рекомендуем перед тем, как совершать какой-либо рискованный поступок, сохранять текущее состояние игры с помощью команды **SAVE**.

Включив лампу и оглядевшись по сторонам (**LOOK**), вы обнаружите себя ползущим по булыжникам низкого прохода (***Low passage***). Обя-

---

[5] Напомним о различии в обозначениях. Жирным курсивом набраны фрагменты описания локаций и названия предметов (например, ***A large set of keys***) а то, что набрано жирными заглавными буквами (например, **TAKE LAMP**) это команды которые нужно набирать с помощью клавиатуры и запускать в действие клавишей **ENTER**.

[6] Вашими глазами, руками и т.д. в игре будет Spectrum. Управлять им можно командами из одного или двух слов, причем слова эти можно не набирать полностью, так как программа обращает внимание только на первые четыре буквы каждого слова. А для наиболее часто встречающихся команд (например, направлений движения) достаточно указать всего 1+2 буквы. Допустимые варианты сокращений приведены в словаре, который находится в конце описания.

зательно возьмите лежащую неподалеку маленькую клетку (**TAKE CAGE**) и продолжайте двигаться на запад (**W**).

Следующая локация развалины комнаты (***Debris room***). Вот и настал момент раскрыть тайну "большого прыжка": магическое слово **XYZY**, написанное на стене, будучи набранным с клавиатуры и введенным нажатием **ENTER**, позволяет мгновенно переместиться из развалин комнаты в моленький кирпичный домик или обратно. Правда пока нужды в таком перемещении еще нет, но самые нетерпеливые и недоверчивые могут проверить надежность действия магии.

Занимаясь экспериментами с телепортацией, не забудьте взять лежащий в этой же комнате черный трехфутовый жезл с ржавой звездой на конце (**TAKE ROD**) и только потом отправляйтесь дальше на запад (**W**). И еще одна деталь, которая может пригодиться в будущем: в описании ***Debris room*** упоминается низкий широкий проход вымощенный булыжниками и закупоренный пробкой из грязи и обломков - может вы найдете способ разобрать эту пробку?

КАРТА 2

Теперь путь лежит в грязном и опасном каньоне (см. **карту 2**), где вас ждет встреча с первым из подземных обитателей. Увы, гном, гулявший по каньону, совсем не обрадовался встрече - он метнул в вас свой маленький топор, промахнулся , грязно выругался и поспешил скрыться в неизвестном направлении. Ну что ж, берите топор и продолжайте путь на запад (**W**).

Каньон скоро кончится, и вы попадете в великолепный высокий зал, тридцатифутовые стены которого напоминают застывшие оранжевые реки. Здесь надо обязательно поймать весело поющую маленькую птичку. Тем, кто следовал нашим советам, сделать это будет совсем нетрудно: достаточно извлечь из своей поклажи жезл (**DROP ROD**) и птицу можно брать голыми руками (**TAKE BIRD**). Все иные способы ловли этого пернатого создания обречены на неудачу.

Справившись с птицей, и убедившись с помощью команды **INVE**, что она благополучно помещена в клетку, идите дальше на запад (**W**) и вы окажетесь в конце прохода, где наконец-то сможете изменить направление движения, спустившись вниз (**DOWN** или просто **D**) через небольшой пролом, выдыхающий навстречу путникам клочья тумана.

Перед вами огромный зал, тянущийся далеко на запад, но попробовав пойти в этом направлении (**W**), вы сразу же наткнетесь на широкую расщелину, которую не удастся ни обойти ни перепрыгнуть. Можете попробовать изобрести какой-либо оригинальный способ переправы, а если не сумеете, то хотя бы возьмите с собой, лежащую около расщелины, маленькую восточную флейту (**TAKE FLUTe**) и вернитесь немного назад (**E**) и проверьте еще два возможных варианта пути.

Через расщелину в южной стене (**S**) можно попасть в низкую комнату, где лежит первое из сокровищ огромный золотой самородок (а

large gold NUGGet). Корявая надпись на стене предупреждает, что с самородком вы не сделаете ни шага. И действительно, взяв золото (**TAKE NUGGet**) и попробовав вернуться назад (**N**), вы потерпите неудачу. С такими ситуациями придется сталкиваться еще не раз, поэтому запомните раз и навсегда: **в запутанном подземном мире *Adventure 1*, если вы вошли куда-то с севера, то это еще не значит, что выход будет на юге**. Поэтому не падайте духом, ищите выход, и рано или поздно он найдется. В данном случае, возвращение в огромный зал (причем, вместе с самородком) обеспечивает команда **LEAVe** (покинуть).

Но вот новая напасть - втащить золото вверх по лестнице (**UP**) не удастся, хотя без него сделать это - нет проблем. Придется исследовать последний оставшийся путь - на север (**N** или **STAIr**). Решившись на этот шаг, вы попадете в зал Короля Горы, откуда во все стороны расходится множество проходов, но все их (кроме обратного) преграждает, разбуженный вышим появлением гигантский зеленый змей. Можете не пытаться поразить змея топором или околдовать его игрой на флейте, лучше выпустите на свободу птицу (**DROP BIRD**): она как вихрь налетит на чудовище и обратит его в бегство.

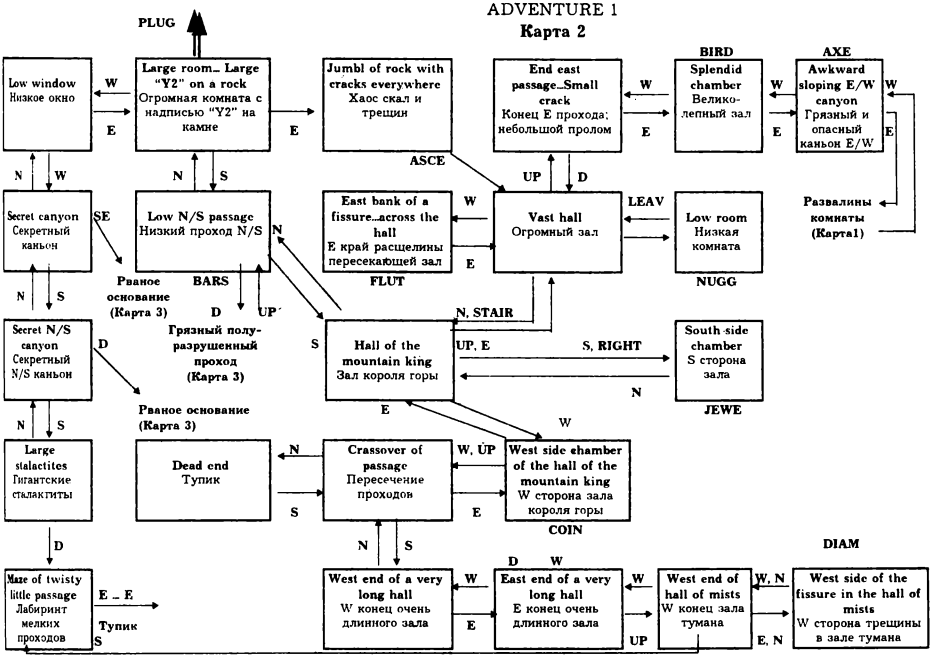
Теперь советуем пойти на север (**N**). Дважды повторив этот ход вы окажетесь сначала в низком проходе север/юг а затем в огромной комнате. Комната эта примечательна огромными знаками ***Y2***, начертанными на скале в ее центре и идущим непонятно откуда голосом, который произносит всего одно слово: "***PLUGH***" Если вы осмелитесь повторить это слово (т.е. ввести его с клавиатуры), то в мгновение ока перенесетесь в хорошо знакомый вам дом (помните, вы брали там снаряжение перед началом похода?). Там можно оставить золото и те из предметов, которые на ваш взгляд пока что не нужны, а затем, введя **XYZY** или **PLUGh**, без труда вернуться в соответствующее этим волшебным словам место пещеры.

Ну вот, первые сокровища найдены и доставлены в безопасное место. Не кажется ли вам, что пора начинать действовать самостоятельно? Не забывайте, программа называется ***Adventure1*** (Приключение 1), а какое же это будет приключение, если идти все время держась за чью-то, пусть даже и весьма надежную, руку. Именно по этим соображениям, описание игры, начиная с этого места, меняет свой характер: больше не будет подробных рекомендаций по каждому шагу игрока, но карты расположения локаций[7] останутся в вашем распоряжении, также как и краткие описания самых заковыристых мест и некоторых разгаданных головоломок.

---

[7] Эти карты включают в себя максимум известной на сегодняшний день информации о лабиринтах ***Adventure 1***, но много еще осталось такого, над чем придется поломать голову: найдены далеко не все сокровища, не все загадки разгаданы, много скратых от глаз проходов осталось незамеченными и т.п.

ADVENTURE 1
Карта 2



Low window
Низкое окно

Низкое окно с высоты в 50 футов смотрит на гигантскую впадину, простирающуюся до самого горизонта. Дно впадины окутывают клочья белого тумана, сгущающегося справа. Следы на пыльном полу около окна свидетельствуют о том, что недавно здесь кто-то побывал. Прямо за окном, на расстоянии каких-нибудь 25 футов, светится еще одно, очень похожее на ваше, окно какой-то комнаты, и оттуда пристально смотрит на вас колышущаяся фигура призрака.

В эту локацию можно попасть не только из огромной комнаты с надписью "Y2" на камне, но и из секретного каньона, однако, в секретный каньон из комнаты с низким окном пройти удается далеко не всегда. Пока это получилось только один раз: тогда, попав случайно из секретного каньона к низкому окну, я случайно дал команду **W**, и вдруг снова очутился в секретном каньоне, - повторить этот трюк при других обстоятельствах не удалось.

Через окно наверняка можно выбраться в долину или добраться до окна напротив, но как это сделать пока неизвестно.

Jumble of rock with cracks everywhere
Хаос скал и трещин

Это место примечательно, пожалуй, лишь тем, что войти в него очень просто, а вот выбраться можно лишь с помощью команды **ASCE**. При этом вы окажетесь вовсе не там, где были до того, как неосторожно забрели в хаос скал и трещин.

Secret canyon
Секретный каньон

Попасть сюда можно тремя путями:
1 - из секретного каньона N/S;
2 - случайным образом из локации ***Bedquilt*** (Рваное основание - см. карту 3);
3 - если повезет, из локации ***Low window*** (Низкое окно - см. выше).

Low N/S passage
Низкий проход N/S

Здесь находится одно из сокровищ бруски из серебра (**silver BARS**). И отсюда можно через дыру в полу перебраться в грязный полуразрушенный проход (***Dirty broken passage*** - см. карту 3).

South side chamber
Þæíàÿ ñòîðîíà çàëà

Здесь находится одно из сокровищ - очень красивые драгоценности (**Precious JEWEllery**).

Large stalactites
Гигантские сталактиты

Очень своеобразное место: гигантские сталактиты протянулись от свода до самого пола, и если вы умудритесь спрыгнуть или как-то по

другому спуститься с них вниз то окажетесь в совершенно гиблом месте - лабиринте мелких проходов (***Maze of twisty little passages***) и выбраться оттуда вряд ли сумеете.

West side chamber of the hall of the mountain King
Западная сторона зала Короля горы

Заберите отсюда редкие монеты (**lots of rare COINs**).

Maze of twisty little passages
Лабиринт мелких проходов

Если судьба заведет вас в лабиринт мелких проходов, выбраться из него будет практически невозможно: куда ни попытаетесь свернуть, всюду увидите одно и то же - похожие как две капли воды мелкие проходы, переплетенные немыслимым образом.

West end of hall of mists
Западный конец зала тумана

Отсюда ни в коем случае не ходите на юг (S) - попадете в лабиринт мелких проходов (***Maze of twisty little passages***) и останетесь там навсегда!

West side of the fissure in the hall of mists
Западная сторона трещины в зале тумана

Возьмите находящиеся здесь алмазы (**several DIAMonds**) и, если сможете, переправляйтесь через трещину, а не сумеете - возвращайтесь назад, так как другого пути здесь нет.

КАРТА 3

Dirty broken passage
Грязный полуразрушенный проход

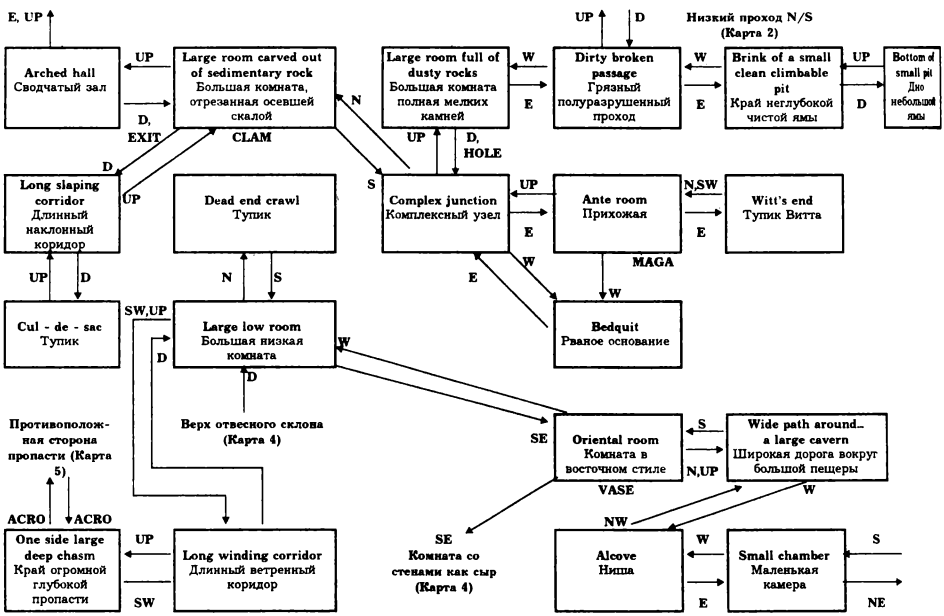
Здесь начинается карта 3, и отсюда можно вернуться в карту 2 (**UP**).

Arched hall
Сводчатый зал

Вверх на восток отсюда ведет коралловый проход (***coral passage***), который хоть и засыпан обломками (постарайтесь найти способ раскопать этот проход!), но не в силах удержать запах моря, пропитавший весь воздух зала.

Large room carved out of sedimentary rock
Большая комната, отрезанная осевшей скалой

В полу и стенах комнаты множество вросших в камень обломков ракушек, а посередине ее возвышается огромный моллюск в плотно закрытой раковине (**an enormous CLAM with its shell tightly closed**). А не скрывается ли за створками гигантской раковины какое-либо из сокровищ, ну хотя бы жемчужина? Не исключено, что и за обломки ракушек, вкрапленных в пол и стены, может оказаться что-то интересное.



ADVENTURE 1
Карта 3

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

Мы пытались сломать раковину моллюска (BREAK) ничего не получилось, не хватило сил. Попытка взять моллюска и вынести его из комнаты тоже оказалась безуспешной - размеры раковины (5 футов) не позволили протиснуть ее сквозь узкий проход.

Large room full of dusty rocks
Большая комната полная мелких камней

Тем кто придет сюда обычным путем или внезапно перенесется из локации ***Bedquilt***, возможно следует повнимательнее обследовать трещины (***cracks***), которые встречаются тут на каждом шагу.

Bottom of small pit
Дно небольшой ямы

В яме протекает крошечный ручеек, появляющийся и исчезающий через щели не шире двух дюймов. Пролезть в такие мелкие отверстия вам вряд ли удастся, но вода ручейка может быть когда-нибудь и пригодится.

Complex junction
Комплексный узел

Странно, но факт - несмотря на сходящееся здесь большое количество проходов, воздух в комнате отличается неподвижностью и затхлостью.

Ante room
Прихожая

В эту локацию можно войти из комплексного узла или случайно попасть из ***Bedquilt***. В том и другом случае вы увидите следы недавних земляных работ и парящую в воздухе надпись:"Пещера ведущая под основные сооружения находится далеко отсюда. Действуйте на свой страх и риск. (Компания "Witt Construction")"

Здесь же вы сможете разжиться несколькими номерами журнала "Spelunker News" (***a few issues of "Spelunker News" MAGAzine***), но вот для чего они нужны, никто не подскажет.

Large low room
Большая низкая комната

Попасть сюда довольно сложно, но крайне необходимо, так как именно здесь начинаются весьма интересные участки подземного лабиринта. Проще всего для входа в большую низкую комнату попытаться использовать особенности локации ***Bedquilt***.

Bedquilt
Рваное основание

Очень и очень странное помещение представляющее собой длинный пронизанный дырами проход вытянувшийся с запада на восток. Чтобы выбраться отсюда надо беспорядочно давать команды **N**, **S**, **D** и **UP**. Но никогда нельзя знать заранее, где вы окажетесь после этого - все подчинено воле случая. Хорошо хоть, что число локаций, в которых вы можете очутиться ограничено. На карте эти локации помечены знаком *.

Странствуя по этой локации, не отчаивайтесь, когда компьютер раз за разом будет сообщать: "Вы ползете в обход по довольно узкому

лазу и вновь оказываетесь в главном проходе",- рано или поздно куда-нибудь да доползете.

One side large deep chasm
Край огромной глубокой пропасти

Из каньона поднимается плотный белый туман, скрывая все, что находится на другой стороне. Над пропастью протянут шаткий деревянный мост, и у входа на него стоит тролль требуя плату за проход. Законность его требований подтверждает табличка:" Стой! Заплати троллю!"

В качестве платы[8] подойдет любое из сокровищ, но при этом вы лишитесь не только самого сокровища, но и очков, которые начисляются за него. Поэтому лучше всего для оплаты прохода использовать золотые яйца (**golden EGGS**) из локации ***Giant room*** (Исполинская комната - см. карту 4). Главное достоинство этих яиц заключается в том, что волшебными словами - **fee fie foe foo** - их можно вернуть в исполинскую комнату. К стати, не спешите на мост, лучше попытайтесь снова раздобыть золотые яйца, ведь за обратный проход по мосту тоже придется платить. Махинацию с волшебными словами при возвращении обязательно повторите, и только тогда можно будет взять яйца в третий раз и,наконец-то, в качестве сокровища отнести в дом.

Все это требует большого терпения, но другие варианты переправы несут с собой утрату каких-либо сокровищ или, того хуже, опасны для жизни. Пытаться расправиться со стражами моста не советуем, они намного хладнокровнее, сильнее и опытнее любого любителя бесплатных переправ. Не скрою, однажды я попытался запуститьть топором в тролля, но тот ловко поймал его, внимательно осмотрел и швырнул обратно, небрежно заметив, что вещь эта сделана добротно, однако для оплаты прохода по мосту ее явно недостаточно.

Oriental room
Комната в восточном стиле

Стены этой пещеры покрывает древняя восточная роспись, но не она должна интересовать вас в первую очередь, а стоящая в комнате драгоценная ваза Минга (**precious Ming VASE**), являющаяся еще одним из тех сокровищ, которые вы ищете.

Wide path around a large cavern
Широкая дорога вокруг большой каверны

Широкая дорога идет вдоль внешнего края огромной каверны. Снизу сквозь плотный белый туман доносится подозрительный плеск (что бы это могло значить?). Туман, поднимаясь снизу, уходит через расщелину в потолке.

---

[8] Плата за переход пропасти взымается два раза за всю игру, т.е. когда вы входите на мост первый раз, и когда первый раз возвращаетесь. Все остальные переправы через каньон не будут вам стоить ничего.

Чтобы оплатить проход (например золотыми яйцами), дайте команду **THROw EGGs**, тролль ловко поймает ваше сокровище и навсегда скроется с глаз.

Alcove
Ниша

Из ниши на восток ведет очень узкий туннель и кажется, что дальше он становится еще уже. Но советуем рискнуть и попробовать протиснуться вперед (**E**), хотя бы для того, чтобы установить источник пробивающегося оттуда мрачного света.

Увы, если у вас с собой есть какие-то предметы, почти все их придется оставить в нише, и даже с лампой следует временно расстаться, иначе в туннель не пролезть.

Small chamber
Маленькая камера

Маленькая камера освещена мрачным зеленым светом идущим непонятно откуда. На северо-восток (**NE**) уходит темный коридор, но что скрывается за ним можно узнать лишь имея какой-либо истачник света, а лампа осталась в нише около дороги - вот вам и еще одна задачка на сообразительность. Если не справитесь с ней, то, может быть, хоть чуть-чуть вас утешит, находящееся в камере, очередное сокровище - изумруд величиной с крупное яйцо (**EMERald the size of provers egg**), - берите его и возвращайтесь назад.

КАРТА 4

Room whose walls resemble swiss cheese
Комната со стенами похожими на швейцарский сыр

Часть помещения занята гигантским скальным блоком, а стены напоминают швейцарский сыр, но как можно использовать то и другое?

Soft room
Тихая комната

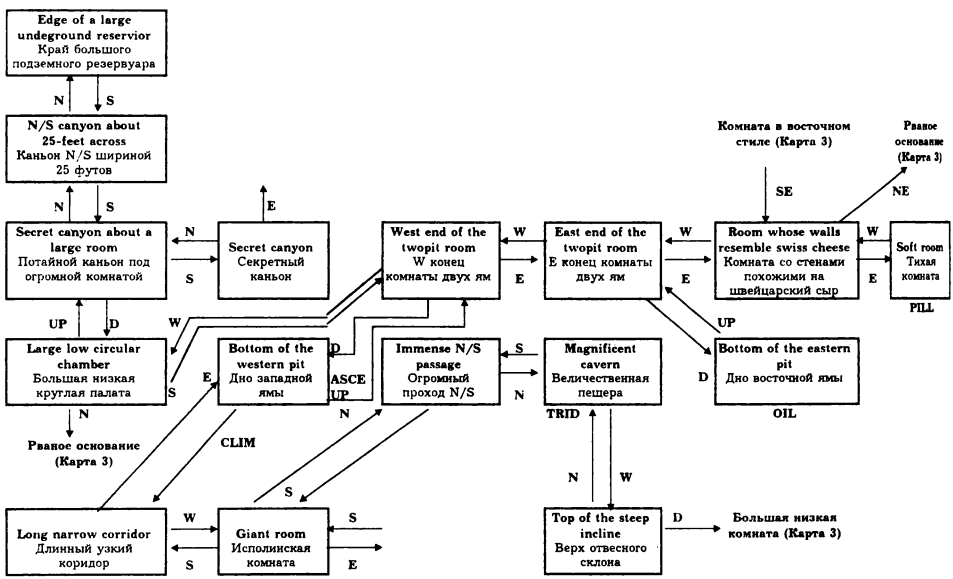
Стены комнаты скрыты за толстыми занавесками, пол покрывает ворсистый ковер, а с потолка свисает мох. Здесь царит невероятная тишина. На память об этом месте, уходя, можете прихватить с собой маленькую вельветовую подушку (**a small velvet PILLow**) ее вряд ли можно считать сокровищем (хотя не вредно проверить и на этот счет), но пригодиться она вполне может.

Не лишним будет исследовать стены, пол и потолок комнаты а вдруг что-то скрывается за мягкими покровами.

West end of the twopit room
Западный конец комнаты двух ям

В стене над ямой здесь есть довольно большая дыра, но дотянуться до нее невозможно. Зато можно спуститься в яму, где вы обнаружите нечто весьма интересное (см. описание локации ***Bottom of the western pit***).

Bottom of the western pit
Дно западной ямы



ADVENTURE 1
Карта 4

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

Крошечный росток (**a tiny little plant**), пробившийся сквозь камни на дне ямы, жалобно шепчет: "Воды, Воды,..."

Если вы оказались достаточно предусмотрительны и прихватили с собой из дома бутылку с водой, попробуйте полить растение (**DROP WATEr**), и прямо на ваших глазах из крошечного ростка оно превратится в 12-футовый бобовый стебель требовательно ревущий: "Воды!! Воды!!!"

Но бутылка уже пуста - ничего не поделаешь, придется сходить за водой к подземному резервуару (локация ***Edge of a large undergrounds reservoir***) и еще раз полить растение. После этого гигантский стебель дотянется до дыры в стене, и вы сможете забраться наверх (**CLIMb**).

East end of the twopit room
Восточный конец комнаты двух ям

Пол завален мелкими каменными обломками, что позволяет без труда спускаться в ямы и столь же просто выбираться из них. Но можно и не делать этого - все ямы легко обходятся. В стенах множество дыр, однако только одна из них достаточно велика, чтобы через нее можно было пройти. Эта дыра расположена над ямой в западном конце комнаты, но, увы, на такой высоте, что до нее невозможно дотянуться.

Обратите внимание на яму около которой вы стоите (см. описание локации ***Bottom of the eastern pit***).

Bottom of the eastern pit
Дно восточной ямы

В углу на дне ямы есть маленькая лужица масла (**a small pool of OIL**).

Large low circular chamber
Большая низкая круглая палата

Пол завален рухнувшими с потолка каменными плитами. На восток и на запад отсюда идут огромные коридоры, но они завалены крупными валунами. Низкие маленькие проходы ведут на север и на юг, причем южный очень скоро сворачивает на запад, огибая валуны.

Обратите внимание, что в этой комнате имеется потайной ход, который не упоминается в описании локации: команда **UP** (вверх) позволит вам воспользоваться этим ходом.

Secret canyon
Секретный каньон

Каньон имеет выходы на север и восток. С севера вы пришли и без труда можете вернуться туда, а вот путь на восток загораживает громадный зеленый дракон. Он вальяжно растянулся на великолепном персидском ковре (**a persian RUG**) и лучше не пытайтесь незаметно проскользнуть мимо ничего не получится. Топор тоже не поможет справиться с отвратительной рептилией он отскакивает от толстой чешуи дракона, не причиняя тому никакого вреда. Если же вы рискнете сыграть на флейте, то дракон, ненавидящий музыку, выдохнет струю пламени и сожжет ковер, который наверняка можно было бы причислить к числу подземных сокровищ.

Попробуйте поэкспериментировать с уже испытанным оружием против драконов - маленькой отважной птичкой. У нас из этого ничего не вышло, но может быть у вас получится. Ну а если нет, придется изобретать что-то новое, уж очень соблазнителен проход, загораживаемый драконом.

N/S canyon about 25 feet across
Каньон север/юг шириной 25 футов

Пол укутан текущим с севера белым туманом. Стены уходят вверх более чем на сотню футов. Посередине каньона, параллельно его стенам, висит устрашающих размеров двухстороннее зеркало, закрепленное где-то в недосягаемой взгляду вышине. Зеркало это, скорее всего, принадлежит гномам, отличающимся непомерным тщеславием. В дальнем конце каньона, на высоте около 40 футов виднеется небольшое оконце.

Edge of a large underground reservoir
Край большого подземного резервуара

От воды поднимаются плотные облака белого тумана, заполняя собой всю огромную пещеру и быстро поднимаясь вверх. С высоты около десяти футов, прямо над вашей головой, в озеро падает водный поток. Он с шумом разбивается брызгами где-то внизу среди тумана.

Giant room
Исполинская комната

Размеры комнаты настолько велики, что лампа не в состоянии осветить потолок, но на полу можно разглядеть гнездо с золотыми яйцами (**a nest of golden EGGS**), а на западной стене нацарапанные кем-то магические слова: **"FEE FIE FOE FOO"** Берите золотые яйца, но не спешите относить их домой, так как сначала их можно и нужно использовать для оплаты переправы через пропасть (см.описание локации ***One side large deep chasm***).

Immense N/S passage
Огромный проход север/юг

Путь на север преграждает огромная, ржавая железная дверь, которую невозможно открыть не смазав предварительно ее петли. Поэтому, если вы еще не выбросили где-то бутылку из-под воды, сходите в комнату двух ям и на дне восточной ямы (***Bottom of the eastern pit***) наберите в нее масла (**TAKE OIL**), а затем смажьте дверные петли (**DROP OIL**) - проход на север будет открыт..

Manificent cavern
Величественная пещера

ыстрый поток бежит через пещеру и искрящимся водопадом низвергается в ревущий водоворот, уходящий через дыру в полу.

Здесь находится одно из разыскиваемых сокровищ - украшенный самоцветами трезубец (**a jewel-encrusted TRIDent**). Возможно эту вещь можно использовать и для других целей - например, для борьбы с дра-

коном в локации ***Secret canyon*** (карта 4) или медведем в ***Barren room*** (карта 5).

Top of a steep incline
Верх отвесного склона

Отсюда вниз (**D**) идет кратчайший путь в большую низкую комнату (карта 3, локация ***Large low room***), т.е. в конечном счете к дому тому месту, где вы собираете найденные сокровища. Но на всякий случай запомните, что назад тем же путем пройти не удастся.

КАРТА 6

Far side of the chasm
Противоположная сторона пропасти

С этой стороны пропасти у моста тоже стоит тролль взымающий плату за проход - надеюсь, вы не забыли второй раз прихватить из локации ***Giant room*** (см. карту 4) золотые яйца (**golden EGGS**) для оплаты обратного прохода.

Long E/W corridor
Длинный коридор восток/запад

Ничего примечательного в этом коридоре нет, но издали слышится слабый грохот не предвещающий ничего хорошего.

The path forks here
Развилка

По мере удалении от моста подозрительный грохот усиливается, и идет он, похоже, с северо-востока (**NE**).

The walls are quit warm here
Теплые стены

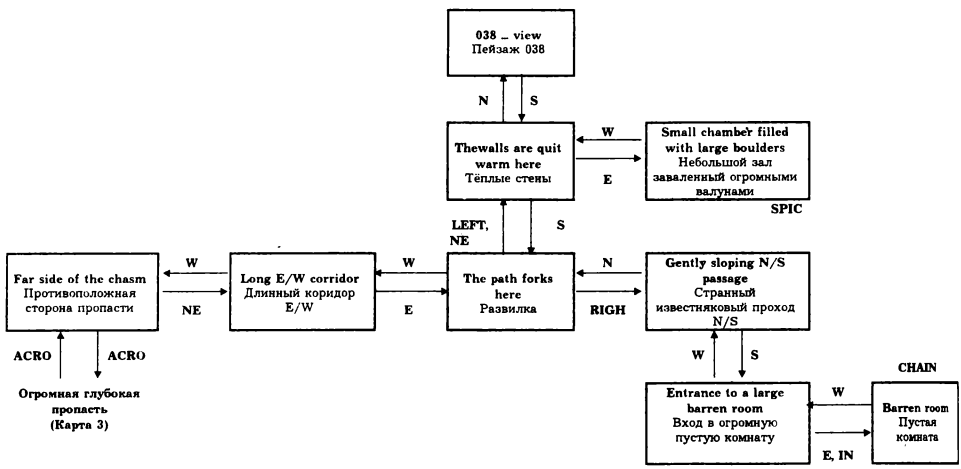
Стены в этой пещере очень теплые, а грохот идущий с севера достигает такой силы, что все вокруг содрогается от него. Вряд ли следует идти в ту сторону, хоть туда и ведет довольно широкий проход. Лучше через низкий лаз проберитесь на восток (**E**).

Small chamber filled with large boulders
Небольшой зал заваленный огромными валунами

Весь зал завален валунами. Воздух в комнате нестерпимо горячий и душный от раскаленных стен. Поскорее берите находящийся здесь набор очень редких пряностей (**some very rare SPICes**) и выбирайтесь назад.

038 view
Пейзаж 038

Далеко внизу виден вулкан, из кратера которого, низвергаясь в пучину, текут потоки раскаленной лавы. Справа от вас, из серного озера в центре бесплодного острова, в облаках обжигающего пара непрерывно извергается гейзер. Дальняя правая стена гигантского зала ослепительно светится сама по себе, придавая дополнительный дьявольский колорит этой и без того адской картине.



ADVENTURE 1
Карта 5

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

Нигде не видно никаких проходов, кроме того, по которому вы пришли сюда.

Gently sloping N/S passage
Странный известняковый проход север/юг

В известняковых стенах прохода имеется множество весьма странных вкраплений, которые не вредно было бы изучить повнимательнее.

Entrance to a large barren room
Вход в огромную пустую комнату

Можно ли считать пустой комнату в которой обитает пещерный медведь? О присутствии этого зверя оповещает табличка над входом, и медведь в следующей локации действительно есть, но факт остается фактом: в описании комната почему-то названа пустой.

Barren room
Пустая комната

Центр комнаты действительно пуст, если не считать пыли, на которой отчетливо видны следы ведущие в дальний угол, откуда на вас глазеет свирепый пещерный медведь, прикованный к стене золотой цепью (**golden CHAIn**).Нам не удалось добыть золотую цепь:

топор запущенный в медведя не попал в цель и упал настолько неудачно, что достать его оказалось невозможным;

прсто схватить цеть (TAKE CHAIn) и разомкнуть ее а потом взять (OPEN CHAIn затем TAKE CHAIn) тоже не получилось, причем в первом случае ноги стали подозрительно мокрыми.

DICTIONARY

A

ACROss	ЧЕРЕЗ (например - перейти через пропасть)	**AWKWard**	ОПАСНЫЙ
ASCE	ПОДНИМАТЬСЯ	**AXE**	ТОПОР

B

BACK	НАЗАД	**BIRD**	ПТИЦА
BARRen	ПУСТОЙ	BOTTle BOTTom	БУТЫЛКА ДНО
BARS	БРУСКИ	**BREAk**	ЛОМАТЬ
BATTle BATTer	БОЙ, сражение БИТЬ, колотить	**BROKen**	СЛОМАННЫЙ, РАЗРУШЕН-НЫЙ
BEAR	МЕДВЕДЬ	**BUILding**	ЗДАНИЕ, ДОМ
BED	РУСЛО		

C

CAGE	КЛЕТКА	**CLIMb**	ВЗБИРАТЬСЯ
CANYon	КАНЬОН	**COBBle**	БУЛЫЖНИК

CARPet	КОВЕР	**COIN**	МОНЕТА
CAVE	ПЕЩЕРА	**CONTinue**	ПРОДОЛЖАТЬ-(СЯ)
CHAIn	ЦЕПЬ	**CRACk**	ТРЕЩИНА
CHASm	ПРОПАСТЬ	**CRAWl**	ЛАЗ, ПОЛЗТИ
CHESt	СУНДУК	**CROSsover**	ПЕРЕСЕЧЕНИЕ
CLAM	МОЛЛЮСК		

D

DARK	ТЕМНОТА	**DOOR**	ДВЕРЬ
DEBRis	РАЗВАЛИНЫ, ОБЛОМКИ	**Down**	ВНИЗ
DEPRession	ВПАДИНА	**DRAGon**	ДРАКОН
DESCend	СПУСКАТЬ-СЯ	**DRAWing**	РОСПИСЬ
DIAMond	АЛМАЗ	**DROP**	ОСТАВИТЬ, БРОСИТЬ
DOME	КУПОЛ, СВОД	**DWARf**	ГНОМ

E

East	двигаться на ВОСТОК	**ENTEr**	ВОЙТИ
EGG	ЯЙЦО	**ENTRance**	ВХОД
EMERald	ИЗУМРУД	**EXIT**	ВЫХОД

F

FIGUre	ФИГУРА	**FOOD**	ПИЩА
FISSure	РАСЩЕЛИНА	**FOREst**	ЛЕС
FLOOr	ПОЛ	**FORK**	РАЗВЕТВЛЕНИЕ
FLUTe	ФЛЕЙТА	**FORWard**	ВПЕРЕД,ВПЕРЕДИ

G

GIANt	ГИГАНТС-КИЙ	**GRATe**	РЕШЕТКА
GEYSer	ГЕЙЗЕР	**GULLy**	ДОЛИНА,ОВРАГ
GOLD	ЗОЛОТО		

H

HALL	ЗАЛ	**HOLE**	ДЫРА,ОТВЕРСТИЕ
HELP	вызов ПОДСКАЗКИ	**HOUSe**	ДОМ
HILL	ХОЛМ		

I

INFOrmation	вызов ИНФОРМАЦИИ об игре	**INto**	ВОЙТИ
INSTruction	вызов краткой ИНСТРУКЦИИ	**INVEntory**	вывести на экран СПИСОК ПРЕДМЕТОВ

J

JEWEllery	ДРАГОЦЕННОСТИ	**JUG**	КУВШИН
JUMP	ПРЫГАТЬ,ПРЫЖОК		

K

KEY	КЛЮЧ	**KNIFe (KNIVes)**	НОЖ (НОЖИ)

L

LAMP	ЛАМПА	**LOOK**	ОСМОТРЕТЬ(ся)
LANTern	ФОНАРЬ	**LOW**	НИЗКИЙ
LEAVe LEAVes	ПОКИНУТЬ ЛИСТЬЯ		
LEFT	ВЛЕВО		

M

MACHine	МЕХАНИЗМ МАШИНА	**MING VASE**	ВАЗА Минга
MAGAzine	ЖУРНАЛ	**MIRRor**	ЗЕРКАЛО
MAIN	ГЛАВНЫЙ	**MOSS**	МОХ
MESS MESSage MESSenger	БЕСПОРЯДОК СООБЩЕНИЕ ВЕСТНИК, ПОСЫЛЬНЫЙ		

N

NEST	ГНЕЗДО	**NUGGet**	САМОРОДОК
North	двигаться на СЕВЕР; промежуточн. направления - NW, NE	**NULLah**	ОВРАГ,РУСЛО РЕКИ, РУЧЕЙ
NOWHere	НИГДЕ, НИКУДА		

O

OFF	ВЫКЛЮЧИТЬ	**ORIEntal**	ВОСТОЧНЫЙ
OIL	МАСЛО	**OUT**	ВЫЙТИ
ON	ВКЛЮЧИТЬ (лампу и т.п.)	**OVER**	СВЕРХ(у), НАД
ONWA	ДАЛЬШЕ, ВПЕРЕД	**OYSTer**	УСТРИЦА
OPEN	ОТКРЫВАТЬ		

P

PASSage	ПРОХОД	**PLANt**	РОСТОК
PEARl	ЖЕМЧУГ	PLAT PLATe	КАРТА,УЧАСТОК земли ПОСУДА, ТАРЕЛКА
PILLow	ПОДУШКА	**PLUGh**	волшебное слово для БОЛЬШОГО ПРЫЖКА
PIRAte	ПИРАТ	**PYRAmid**	ПИРАМИДА
PIT	ЯМА		

R

RESErvoir RESEmble	РЕЗЕРВУАР ПОХОДИТЬ, напоминать	**ROAD**	ДОРОГА
RESTore	ВОССТАНОВИТЬ неоконч. игру	**ROCK**	СКАЛА
RETRograde	ОТОЙТИ назад	**ROD**	ЖЕЗЛ
RETUrn	ВЕРНУТЬСЯ	**ROOM**	КОМНАТА
RIGHt	ВПРАВО	**RUG**	КОВЕР

S

SAVE	СОХРАНИТЬ текущее. состояние игры	**SNAKe**	ЗМЕЙ
SCORe	просмотр РЕЗУЛЬТАТОВ игры	**South**	Двигаться на ЮГ; Промежуточн. направления: SW, SE
SECRet	СЕКРЕТНЫЙ, ТАЙНЫЙ	**SPICes**	ПРЯНОСТИ

ADVENTURE PROJECT – ПЕРЕКРЕСТОК ДРАКОНОВ

SHADowy figure	ПРИЗРАК	**STAIrway**	ЛЕСТНИЦА
SHELl	РАКОВИНА	STALactite STALk	СТАЛАКТИТ СТЕБЕЛЬ
SILVer	СЕРЕБРО	**STEP**	ШАГ
SLAB	КАМЕННАЯ ПЛИТА	**STREam**	РУЧЕЙ, ПОТОК
SLIT	ЩЕЛЬ	**SURFace**	ПОВЕРХНОСТЬ

T

TABLe TABLet	СТОЛ, ТАБЛИЧКА ТАБЛЕТКА	**TRIDent**	ТРЕЗУБЕЦ
TAKE	БРАТЬ	**TROLl**	ТРОЛЛЬ
THROw	МЕТНУТЬ	**TUNNel**	ТУННЕЛЬ

U

Up	ВВЕРХ		

V

VALLey	ДОЛИНА	**VIEW**	ОСМОТРЕТЬ, ВИД, ПЕЙЗАЖ
VASE	ВАЗА	**VOLCano**	ВУЛКАН
VENDer	ТОРГОВЕЦ		

W

WALL	СТЕНА	**West**	двигаться на ЗАПАД
WATEr (H2O)	ВОДА	**WINDow**	ОКНО

X

XYZY	волшебное слово для БОЛЬШОГО ПРЫЖКА		

* Все команды(слова) в словаре получены при просмотре блока машинных кодов программы с помощью **MONS4B**.

** Со смещением вправо даны слова, значения которых пока еще не удалось проверить при работе с **ADVENTURE 1**.

WELLINGTON AT WATERLOO

(c) CCS, 1989 by R.T.Smith

Эксперт: Илья Командин, г.Калуга.

Стратегическая игра Wellington at Waterloo, выпущенная в 1989 году фирмой CCS одной из ведущих фирм по производству стратегических игр, повторяет события 1815 г. Тогда, близ местечка Ватерлоо произошли сражения, положившие конец так называемым "Ста дням" Наполеона - важному историческому периоду в жизни всей Европы. На всякий случай напомню, в какое время разворачивались события:

ВВЕДЕНИЕ

В октябре 1813 года около города Лейпциг произошло решающее трехдневное сражение войск Наполеона с объединенными войсками России, Пруссии, Австрии и Швеции. Армия Наполеона была разгромлена, и военные действия переместились на территорию Франции.

31 марта 1814 года в Париж вошли войска коалиции. Наполеона заставили подписать акт отречения, а затем отправили в почетную ссылку на маленький остров Элбу у берегов Италии.

В течении одиннадцати месяцев Бонопарт пристально следил за происходящим во Франции. Понимая, что народ не приемлет власть Бурбонов, он решился на смелый шаг. 1 марта 1815 г. бывший император с преданными ему солдатами и близкими людьми высадился на юге Франции и двинулся на Париж.

Людовик XVIII направил против Бонопарта тридцатитысячную армию, но вся она, в полном составе, перешла на сторону Наполеона. На всем пути в столицу войска переходили на его сторону. Армия императора вошла в Париж, его на руках внесли во дворец, из которого накануне сбежал Людовик XVIII. Но продержаться у власти Наполеону удалось всего 100 дней.

Коалиция европейских стран вновь выступила против Бонопарта и 18 июня 1815 года в сражении под Ватерлоо англо-голландские войска А.Веллингтона и прусские вой-

ска Г.Л.Блюхера разгромили наполеоновскую армию. 22 июня Бонопарта вновь заставили отречься от престола. На этот раз он был сослан на крошечный остров Св.Елены в Атлантическом океане, где через несколько лет скончался.

В WELLINGTON AT WATERLOO Вам предстоит попытаться повторить историческую победу союзных войск и разгромить армию Наполеона. Но не думайте, что это будет просто в игре войска Бонопарта гораздо опаснее, чем были в реальной жизни.

ИГРА

Те, кто знаком с другими играми фирмы CCS (в особенности, со STALINGRAD'ом) найдут здесь очень много знакомого, хотя мне, по ряду причин, WATERLOO нравится больше, чем другие игры такого типа.

Сразу же после загрузки программы Вы должны будете выбрать уровень сложности (1-3), а затем Вам предложат загрузить отложенную игру (LOAD SAVED GAME - L) или же начать новую. Хочу предупредить, что загрузить отложенную игру можно только один раз - в самом начале, так что у Вас не получится сначала запустить новую игру, а затем продолжить старую.

Также, перед каждым ходом Вам будет предложено сохранить состояние игры (SAVE GAME - S).

Итак, разобравшись с настройкой и работой с лентой или диском, перейдем к основной части.

ОСНОВНАЯ ЧАСТЬ

Структура игры основана на пошаговом методе - сначала Вы отдаете приказы, затем перемещаются и атакуют неприятеля Ваши войска, потом то же самое делают вражеские части. Управляется игра, как это принято, с помощью разветвленного меню.

Первое меню, в которое Вы попадете, будет состоять всего из двух пунктов - перейти в режим отдачи приказов (ARMY ORDERS A) или в режим передвижения (MOVERMENT - 0).

Второе выбирать пока не советую, так как в этом режиме войска выполняют Ваши приказы, а затем передают ход врагу.

Войдя в режим отдачи приказов Вы увидите следующее меню:

- ORDERS - O. Собственно, отдача приказов.
- DETAILS - D. Просмотр информации о войсках.
- TERRAIN - T. Просмотр расположения армий.
- REORGANIZATION - R. Реорганизация войск.

Более подробное рассмотрение этих опций, я, для удобства, начну с конца.

REORGANIZATION R. В отличие от STALINGRAD'а, в WATERLOO нет возможности производить пополнение войск из специального резерва. Вместо этого здесь предусмотрен режим реорганизации частей внутри армии.

Итак, нажав клавишу "R" и выбрав клавишами (1-7) армию, в которой Вы хотите произвести реорганизацию, Вы попадете в основное меню этого режима. Здесь клавишами вверх/вниз Вы должны будете выбрать часть, которую собираетесь реорганизовать, а затем выбрать один из видов реорганизации:

DISBANDMENT D. Расформирование выбранной части и, за счет этого, увеличение размеров остальных частей армии.

AMALGAMATION A. Восстановление размеров выбранной части за счет урезания остальных частей.

К сожалению, реорганизовать армию удается далеко не всегда (выдается сообщение NOT ALLOWED). Вот некоторые из причин, почему так происходит.

1. Реорганизуемая часть является единственной частью такого типа в данной армии. К примеру, Вы не можете реорганизовать артиллерийскую часть в армии, где все остальные войска - пехота.

2. В части, которую Вы хотите расформировать больше солдат, чем суммарная нехватка живой силы в остальных частях армии.

3. В части, которую Вы хотите пополнить, нехватка солдат не превышает 500-700 человек.

4. Хотя бы одна из частей армии находится вблизи врага.

TERRAIN - T. С помощью этой опции Вы можете просмотреть расположение как своих, так и вражеских армий. Сначала Вы должны будете выбрать сторону (союзные войска A, французские F), а затем номер армии. Части армии при этом показаны не будут, просто квадраты, на которых они расположены, будут выделены инверсией.

DETAILS - D. Воспользовавшись этой опцией, Вы сможете просмотреть информацию о видимых частях. На экране появится курсор, наведя который на любую часть, Вы увидите ее название и количество человек в ней. Если эта часть принадлежит Вам, то Вы получите также данные о моральном духе солдат.

ORDERS O. Это основная опция в игре - опция отдачи приказов. Выбрав одну из армий, Вы сможете задать ка-

ждой из ее частей рубеж, который она должна занять. Пехотным войскам Вы также можете задать тип построения: в линию (LINE L), колонной (COLUMN C) или каре (SQUARE S). Каждый из типов имеет свои преимущества и недостатки:

Линия - часть передвигается на два шага за ход, хорошо выдерживает артиллерийские удары и может стрелять по вражеским частям,находящимся от нее на расстоянии не более двух шагов.

Колонна часть перемещается на три шага за ход, хорошо бьется в ближнем бою, но сильно страдает под огнем артиллерии и не может стрелять.

Каре часть хорошо сражается в ближнем бою (особенно, с кавалерией), но не может двигаться, стрелять и очень сильно страдает под артиллерийскими ударами.

БИТВА

Когда Вы закончите отдавать приказы, выходите в первое меню и нажимайте 0. Ваши войска двинутся соответствии с приказами, а затем перейдут в режим битвы. По просьбе Алексея Семенова из Рязани, я освещу этот режим подробнее. В битве участвуют следующие части:

- Любые части, вошедшие в прямой контакт с врагом.
- Пехотные части, построенные в линию и находящиеся на расстоянии двух шагов от противника.
- Артиллерийские части, находящиеся на расстоянии 6-8 шагов от врага.

При столкновении двух частей противоборствующих армий битва происходит следующем образом: сначала мигает атакуемая часть и рядом с ней появляется число, показывающее количество убитых в ней солдат. Затем мигает атакующая часть и показывается, сколько солдат погибло при атаке. Если битва произошла после хода Ваших войск, то атаковать будет Ваша часть; если после хода французов - то наоборот.

При стрельбе по врагу сначала мигает часть, ведущая огонь, а затем часть, попавшая под обстрел, и показывается число убитых в ней солдат.Аналогичная система используется и в игре STALINGRAD, но там потери показываются в процентах от размера дивизии и нет стрельбы издалека.

ОБЩИЕ ЗАМЕЧАНИЯ

1. В игре присутствует всего три вида войск: пехота, кавалерия и артиллерия. Каждый из родов войск надо использовать по назначению.

Так, артиллерию лучше держать по краям основной

группы войск, чтобы обстреливать вражеские части на подходе. Кавалерию, учитывая ее большую скорость, лучше всего использовать для уничтожения вражеской артиллерии. Пехота - самое многоцелевое подразделение. Сначала лучше всего строить ее в линию, но когда Вы видите, что столкновения с врагами не избежать, срочно переводите все части, которые могут вступить в ближний бой, в режим колонны. При этом свободными частями окружайте войска, чтобы можно было стрелять.

2. При прямом контакте с врагом перестраивать пехотные части невозможно, так что всегда делайте это заранее.

3. Если Вы уже играли в WELLINGTON AT WATERLOO, то, наверное, заметили, что не можете работать с последней, седьмой армией. Эта армия появится гораздо позже, когда силы обеих сторон будут ощутимо подорваны, и сильно поможет Вам.

УДАЧИ!

****

NAPOLEON

(C) CSS, 1986

Эксперт: Илья Командин, г. Калуга.

NAPOLEON это еще одна игра фирмы CCS о противостоянии русских и французских войск. Игра, как всегда, жутко непатриотична Вы играете на стороне французов. Если Вы играли в Wellington at Waterloo или в Death in the Snow, то найдете в этих играх много общего. Это и неудивительно, ведь у них один и тот же автор - хорошо всем знакомый R.T.Smith. Да, кстати, по поводу Death in the Snow. Эта игра отличается от Napoleon'a только картой и начальным расположением армий, так что все нижеследующее можно считать и ее описанием тоже.

ИГРА

Вся игра, как обычно, разделена на несколько частей. Вы отдаете приказы, Ваши части надвигаются и атакуют противника, затем то же самое делают соединения врага. Основной частью игры является режим отдачи приказов. Он, в свою очередь, разделен на три этапа: отдача приказов основной группе войск, отдача приказов артиллерии, реорганизация.

Этап 1

У основной группе войск относятся пехота (изображение ружья) и кавалерия (две скрещенные сабли). Все части сгруппированы в армии (соединения, помеченные одинаковыми цифрами), но на Ваши действия это никакого влияния не оказывает.

Итак, после того, как в главном меню вы нажимаете клавишу "А", игра переходит к первому этапу. (Надо отметить, что это меню состоит всего из двух пунктов: "А" - отдача приказов и "0" - передвижения войск. В первом этапе Вам предстоит оперировать со следующем меню:

- ORDERS - отдача приказа;
- DETAILS узнать информацию о части;

- TERRAIN - посмотреть территорию, на которой находится часть;
- DELETE ORDERS отменить отданный приказ;
- EXIT - переход к следующему этапу.

Теперь обо всем чуть-чуть подробнее.

DETAILS - наведя курсор на любую часть, и нажав D, можно посмотреть количество человек в ней и их настроения. Можно узнать информацию и о вражеской части, но только если она находится в пределах видимости Ваших войск.

TERRAIN выбрав эту опцию, Вы сможете увидеть в инвертированном виде квадрат карты, на котором стоит соединение.

ORDERS этой опцией Вы можете выбрать рубеж, которого должна достичь часть. Компьютер попросит Вас навести курсор на нужное место и нажать клавишу "T" Отменить приказ можно, нажав клавишу "U"

Но Вы, должно быть, заметили, что иногда, при попытке отдать приказ, появляется новое меню, абсолютно не соответствующее тому, что я описал выше. Все дело в том, что в игре присутствуют два вида частей: обычные соединения, на которых, после подачи сигнала появляется буква "U" (для них справедливо все вышесказанное), и командные (на них появляется буква "C"). Вот именно для командных частей и предназначено это расширенное меню (на мой взгляд - абсолютно ненужное):

- MOVE полное повторение режима передвижения для обычных войск за исключением того, что в конце Вас попросят подтвердить отданный приказ.
- FULL COMMAND - Вы отдаете часть под полное командование ее командира. Если возле части будет находится враг, то командир сам примет решение, напасть ли ему на врага или отступить (предварительно, конечно, спросив Вашего разрешения).
- ENGANGE напасть на ближайшего врага.
- WITHDRAW - отступить, если рядом есть враг.
- REGROUP аналог MOVE (разницы я не заметил).

После того, как Вы отдадите все необходимые приказы, нажимайте "E", и переходите к другому этапу.

Этап 2

Первоначально компьютер спросит Вас, хотите ли Вы отдать приказы артиллерии. Если хотите, то нажимайте "A", и Вы попадете в меню, идентичное меню первого этапа. Здесь все функции, кроме ORDERS, работают абсолютно так же, как и в первом меню, но направлены они на артиллерию. Опция же ORDERS несколько изменена. Сначала Вы должны будете выбрать тип приказа: приказ передвигаться (M) или стрелять (F). При выборе первого Вы должны будете задать рубеж для части (как и в первом этапе). При выборе же второго, Вам достаточно будет определить вид войск, который должна обстрелять данная часть пехоту (I) или артиллерию (A).

Этап 3

Покончив с отдачей приказов артиллерии, Вы перейдете в режим реорганизации, а попросту говоря - пополнения. Выбор действий здесь не очень большой. R реорганизация части, на которую наведен курсор. E - выход в основное меню. Реорганизация не может быть совершена по следующим причинам:

- NOT YET ALLOWABLE реорганизация не дозволена по неизвестным причинам.
- ENEMY IN PROXIMITY враг в пределах видимости. При появлении данного сообщения советую, если позволяют обстоятельства, отвести часть подальше от вражеских войск и произвести реорганизацию.
- UNIT HAS BEEN ALREADY REORGANIZED часть уже была реорганизована. Как видите, пополнить соединение можно только один раз за игру.

Итак, наконец-то Вы отдали все приказы. Теперь нажимайте "0" (MOVERMENT) и смотрите, что получилось.

Ваши войска начнут двигаться к намеченным целям. Иногда их командиры предлагают изменить полученный приказ. Это выражается в двух видах докладов:

Мне кажется, что позиция части довольно уязвима. Хотите ли Вы, чтобы я отступил?

Действительно ил Вы хотите, чтобы я следовал приказу? В противном случае я атакую врага.

На мой взгляд, если Вы уже имеете опыт в стратегических играх, то Вы никогда не станете соглашаться с командирами и будете нажимать в первом случае клавишу "N", а во втором - "Y" (как это делаю я).

Когда передвижение Ваших частей завершится, игра перейдет в режим битвы. В сражении принимают участие любые части, вошедшие в прямой контакт с соединениями противника и артиллерийские части, в зоне досягаемости которых находятся вражеские войска. С системой битвы в игре есть небольшие странности. При обстреле войск артиллерией все просто: сначала мигает стреляющая част, затем соединение, подвергнувшееся обстрелу и показывается, сколько человек при этом было уничтожено.

При прямом же контакте обычно получают повреждения и атакуемая и атакующая части (как и должно быть), но иногда страдает только атакуемая (это еще куда ни шло), а иногда только атакующая! Но, к счастью, в любом случае легко понять, какая часть получила повреждения. Соединение мигает, а затем на его месте появляется число погибших в нем человек. В процессе сражения с войсками могут происходить две вещи:

- UNIT RETREATS часть отступает, и при этом получает довольно большие повреждения.
- UNIT ROUTS часть полностью уничтожена.

После завершения битвы ход будет передан русским войскам, затем опять начнется сражение, только нападать будут уже русские.

СОВЕТЫ:

1. Помимо обычных объектов, таких, как реки и леса, на карте встречаются странные овальные фигуры. Существуют два мнения на их счет: это насыпные валы, увеличивающие силу находящихся на них частей или это центры снабжения, дающие ресурсы для реорганизации. Я четко не придерживаюсь ни одного из этих мнений, и, честно говоря, не особо стараюсь захватить эти объекты.

2. Советую встречать противника, выстроив войска ровным строем, и выставив вперед артиллерию (но не забудьте оставить ей возможность для отхода).

3. При малейшей возможности, старайтесь уничтожить вражескую артиллерию (лучше всего - кавалерией). Артиллерийская часть уничтожается очень быстро, правда, и вред наносит большой.

4. Если поле первого же хода программа "вываливается" в Бейсик с сообщением VARIABLE NOT FOUND, то замените в указанной, и следующей за ней строке переменную GT на GL и давайте команду CONTINUE.

Вот и все. Удачи Вам!

****

STAR FOX

Copyright Ariolasoft and Realtime Games Software 1987

Эксперт: Антон Мизинов (GIP), г.Дзержинск Нижегородской обл.

Программа STAR FOX обычно упоминается в сравнении с ELITE, но она достаточно самостоятельна и интересна, чтобы остановиться на ней подробнее. Прекрасная трехмерная теневая графика, захватывающие моменты боев и стыковок - вот визитка этой игры. Можно с уверенностью сказать, что, однажды записав эту игру, Вы никогда ее не сотрете, если, конечно, жанр SPACE SIMALOTORS не оставляет Вас равнодушным.

1. Начало работы.

После запуска программы Вы можете выключить звук, выбрать управление или переназначить клавиши:

- SPACE - огонь (FIRE)
- Q - вверх (UP)
- A - вниз (DOWN)
- O - влево (LEFT)
- P - вправо (RIGHT)
- X - ускорение (ACCELERATE)
- Z - замедление (DECELERATE)
- S - выбор оружия (SELECT WEAPON)
- T - турбо-режим двигателя (TURBO)
- H - голографическая карта (HOLOCUBE)
- 2 - автопилот (PLANETARY LOG)
- 1 - бортовой компьютер (GENERAL LOG)
- 3 - пауза (PAUSE)

Внимание! После начала игры переназначить клавиши будет невозможно.

2. Цель игры.

Вы выступаете в роли пилота боевого патрульного корабля STAR FOX. Ваш "Звездный Лис" оборудован по последнему слову техники. Ваша цель уничтожать военные конвои, которых полным-полно в отведенном Вам участке.

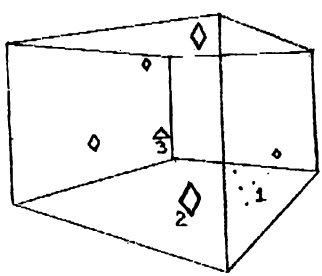
3. Навигация.

Под Ваше патрулирование отведен небольшой участок космо-

са в форме куба с восемью планетами. При вылете за его пределы Вам выдадут сообщение "LEVEL INCOMPLETE" и Ваш корабль автоматически развернется на 180 градусов.

Ваш STAR FOX оборудован голографической картой этого участка космоса, но в начале игры по ней можно определить только свое местонахождение. По мере обнаруживания планет и конвоев, они будут отмечаться на карте и можно будет следить за передвижениями обнаруженных врагов.

Трехмерная карта включается клавишей "Н" и выглядит как каркас куба. Клавишами управления куб можно всячески вращать. Для удобства Вы можете его увеличить или уменьшить клавишами X, Z. Обозначения на карте такие:



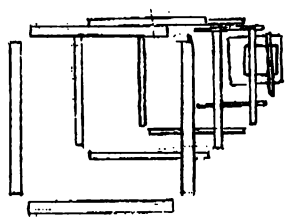
Точки - это вражеские конвои.

1. Ромбики - это планеты.
2. Треугольник Ваш корабль.

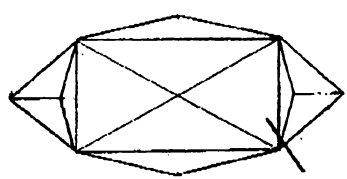
Для того, чтобы направить Ваш корабль на вражеский конвой или планету, нужно совместить цель и треугольник так, чтобы Ваша цель находилась за треугольником. Затем просто набирайте скорость.

4. Планеты и орбитальные станции

При близком приближении к неотмеченной на карте планете появится сообщение "PLANET DETECTED" (ОБНАРУЖЕНА ПЛАНЕТА). Воспользовавшись автопилотом (об этом будет сказано ниже) найдите вход в защитное поле.



Планету из космоса не видно, она окружена десятком защитных полей, но в них существуют окна для взлета и посадки кораблей. Вы должны пролететь сквозь все окна, не вылетая за их пределы, иначе защитные поля планеты будут забирать энергию защитных

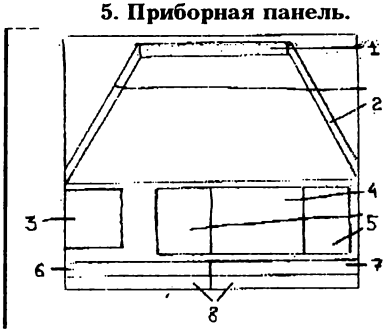


ЛЮК АНГАРА

полей Вашего корабля. Пролетев этот коридор, дождитесь, пока из поля зрения не исчезнут все звезды, и Вы опуститесь на орбиту. На орбите каждой планеты есть стан-

ция, Вам надо ее найти. При приближении станции на расстояние видимости, Вам выдадут сообщение "MOTHER SHIP IN VISUAL RANGE" (станция на расстоянии видимости). Чтобы пришвартоваться, Вам надо развернуться на 180 градусов. В монитор заднего обзора Вы можете понаблюдать процесс швартовки. При этом на экране будет сообщение TRACTOR BEAMS LOCKED ON (включена система стыковки). Маневрируйте около станции аккуратно, если Вы слишком близко приблизитесь, то появится сообщение COLLISION WARNING! MOTHER SHIP TOO CLOSE (опасность столкновения, станция слишком близко). Теперь любое неверное движение и Вам конец. На станции Вас отремонтируют, заправят топливом и предложат новое оружие.

5. Приборная панель.



На панели обозначены цифрами:

1. Индикатор Ваших координат.
2. Индикатор энергии защитных полей.
3. Указатель задействованного оружия.
4. Монитор заднего обзора.
5. Мониторы бокового обзора.
6. Индикатор количества топлива.
7. Количество премиальных очков.
8. Мощность работы двигателя.

6. Список планет.

Для удобства, предлагаю список существующих планет и их координаты:

1. REHTONA -23-82-15-
2. VEGTAR -10-57-47-
3. JANTUS -32-08-93-
4. MYSTO -22-42-46-
5. PHALBA -81-70-23-
6. DRAYGON -51-10-39-
7. BOLOS -43-18-19-
8. PSYLON -49-12-03-

По данным координатам найти планеты достаточно легко, воспользовавшись картой.

7. Автопилот и перелеты.

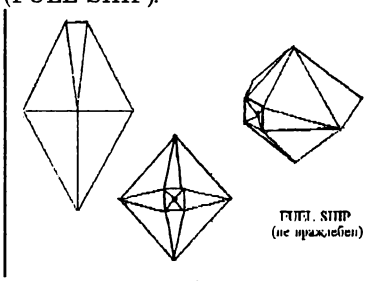
Клавиша "2" переводит Вас в режим программирования автопилота. При этом появляется список планет и, если Вы уже встречали эту планету, то будут указаны ее координаты. Клавишами "Q" и "A" можно выбрать любую из планет, при условии, что компьютеру известны ее координаты. Настройка автопилота осуществляется клавишей "огонь" Внизу экрана можно увидеть, на какую планету автопилот настроен в настоящее время. Чтобы выйти из этого режима, выберите "EXIT", "FIRE"

После описанных операций корабль развернется в нужную сторону, а Вам останется только прибавить скорость. Если в свободном полете автопилот обнаружит поблизости планету, то на экране появится сообщение "Planet Detected"

8. Заправка.

Самое главное в этой игре не остаться с пустыми баками. При стыковке со станциями Вас за-

правляют автоматически, но зачастую, топливо кончается еще вдали от планет. В этом случае часто помогают заправочные корабли (FUEL SHIP).



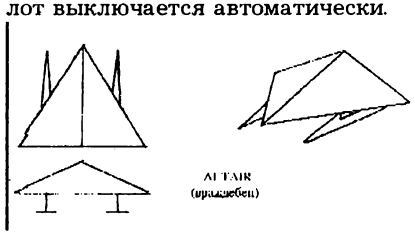
Эти корабли курсируют туда-сюда по всему пространству, но увидеть их довольно сложно, так как они снабжены защитными полями, делающими их невидимыми. В то же время, эти корабли сами дают о себе знать при приближении сообщением "FUEL SHIP DETECTED" Если у Вас наблюдается нехватка топлива, то смело нажимайте F, и можете наблюдать процесс заправки.

9. Боевые действия.

Все корабли, кроме заправщика и станции, враждебны. В течении всей игры Вам предстоит методично уничтожать один вражеский конвой за другим. По мере накопления премиальных очков, будут выдаваться сообщения о переходе на следующий уровень. Уровни отличаются новыми кораблями и оружием, которое Вы можете приобрести на планетах. При обнаружении вражеского конвоя на экране появится сообщение: "Convoy Detected"

Единственный совет по поводу уничтожения конвоев: пытайтесь уничтожить как можно больше кораблей, пока они на большом расстоянии от Вас. В ближнем бою это сделать довольно сложно. Здесь нам поможет еще одно техническое новшество система самонаведения, включающаяся по клавише "L"

Стоит нажать "L", и компьютер будет разворачивать корабль так, чтобы Ваша цель постоянно была в пределах экрана. О включении и выключении автопилота говорят сообщения "LOCKED ON" и "LOCKED OFF" Если конвой полностью уничтожен, то автопилот выключается автоматически.



10. Бортовой компьютер.

Он предназначен для отслеживания информации о вражеских конвоях и магнитных штормах. Если Вы обнаружили конвой и нажали кнопку "1", Вам будет выдана следующая информация:

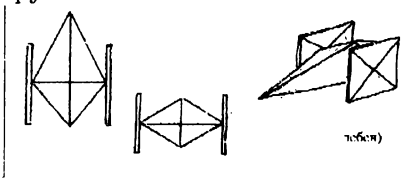
1. Из какого типа кораблей состоит конвой.
2. Количество кораблей.
3. Тип оружия.
4. Настоящие координаты.

Наблюдать движение конвоев по космосу можно и по карте. Точно также бортовой компьютер

отслеживает движение магнитных штормов, сообщая их координаты.

11. Вооружение.

В начале игры у Вас имеется только один "LASER MK1" но по прибытии на какую-нибудь планету, Вам предложат более мощное оружие.



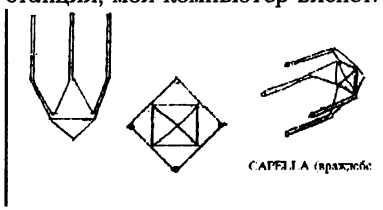
Всего Вы можете установить на борту три вида вооружения, два из которых Вы можете заменять. На более высоких уровнях и оружие, которое Вы приобретаете, становится более мощным. Повышается огневая мощь и у врагов. Вот лишь малая часть встреченного мною оружия:

1. Laser Mk1
2. Laser Mk2
3. Laser Mk3
4. Di-Laser
5. Tri-Laser
6. Shockers
7. Plasma Beam
8. Energy Bolt
9. ACID
10. Attack Shield и др.

Несколько слов от себя.

К сожалению, я не могу дать полностью исчерпывающего описания. И у меня есть на то свои причины мою версию игры невозможно пройти до конца, если он, конечно, есть... А дело вот в чем:

После перехода на третий или четвертый уровень может случиться, что на какой-нибудь станции Вас предупредят об электромагнитном шторме и сообщат его координаты. После вылета из этой станции, мой компьютер виснет.



Если не считать этой ошибки, наверняка возникшей из-за некорректного вскрытия защиты, впечатление от игры осталось отличное. Жутко интересно, что же это за шторм, из-за которого у меня даже компьютер виснет.

БЕЗУМНЫЙ ЗВЕРЬ
По игре " The Saga"

(c) J.C.L. from "L.A." 1996, г.Владивосток.

В тот ясный солнечный день я стоял посреди улицы и разглядывал медные дверные ручки, прикрепленные к дверям экстравагантно раскрашенных домов. Странное времяпрепровождение? Не судите слишком строго, ведь у каждого из нас есть свои маленькие странности. Кстати, меня зовут Мароник Безумный Зверь и я странствующий варвар, еще покруче, чем Конан. Не верите что "круче"? Это вы зря! В конце концов, ни одной адвентюры о Конане еще не написано, а обо мне вот она: "THE SAGA"

Приключение это началось в городке, который назывался Анвас. Сами горожане прозвали свой город: "город отбросов", что дает некоторое представление о тех краях, да и о тех временах. Раннее средневековье или что-то типа того.

Впереди у меня был целый день и никаких дел, которые необходимо было сделать. Я решил просто прогуляться. Первым местом, куда я заглянул, был музей. Музей Анваса учреждение уникальное. Во всем королевстве Кренан нет ничего подобного. Анвасцы очень гордятся своим музеем, но посещать его избегают.

Первым экспонатом, который привлек внимание, была редкая нефритовая статуя. Еще бы! Ведь она была: во-первых, редкая, во-вторых - нефритовая, в третьих статуя! Полюбовавшись этим произведением искусства, я решил, что мне оно ни к чему. Больно уж она тяжелая, да и таскать такую уродливую штуковину с собой хлопот не оберешься.

Могут ведь неправильно понять.

Вторым экспонатом в музее был уличный сорванец. Да. Да. Обычный уличный сорванец. Помню сколько шума было пару лет назад, когда сей мумми... мимеди... му..., ну, в общем, высохший такой и сморщенный труп вырубили из какого-то ледника ученые. С восторженными криками внесли они его в музей, крича что-то о великих антропопологичесих открытиях, о диссертациях и премиях, и обо всем прочем, о чем в таких случаях говорят ученые. Но труп - это только труп. Он мне не нужен. Последним экспонатом музея оказался изумительной работы гобелен, на котором был изображен я сам. Воистину, великое произведение искусства! Такие вещи следует бережно хранить и лелеять для будущих поколений... А еще, должен заметить, в этот гобелен можно увязать какие-нибудь тяжелые вещи, или он может сгодиться в качестве заплечного мешка.

Прихватив гобелен, я решил отправиться в таверну "Совращенный Пони" - тоже своего рода достопримечательность Анваса. Здесь было как всегда - сумрачно и людно. Пьяные экс-герои валялись на полу, за стойкой бара стоял неприветливый тролль, а рядом лежал маленький гномий трупик. В подобных заведениях трупы не редкость. Осмотрев несчастного, я увидел, что у него перерезано горло. Бедняге не поможешь. Наверняка, это работа Спада Резателя Глоток. Есть тут такой полубезумный желтый гном любит всем

глотки резать. Бросив беглый взгляд на стойку бара, я заметил свежевырезанную (вчера ее еще не было) надпись: "Я люблю Зеленую Соню" Воистину, любовь великое чувство!

Выйдя из бара, я зашел в магазинчик к Этель. Эта самая Этель продает здесь самый известный По - Эту - Сторону - Гор деликатес яйца. Она просто помешана на яйцах. (Нет, в хорошем смысле). Любые размеры, формы, цвета. Вкус - потрясающий! Какие угодно яйца Вы можете найти в ее магазине. Ну и, к тому же, она приторговывает разными сексуальными штучками. В магазинчике я проигнорировал неизменную банку с яйцами (не люблю яичницу), но захватил небольшой железный ключик, который, очевидно, забыл кто-то из покупателей.

Выйдя из магазинчика Этель, я пошел дальше, наслаждаясь прекрасным летним днем, но тут показался шагающий мне навстречу Спад Резатель Глоток. Этот гном был явно чем-то раздражен и жадным взглядом впился мне в шею. Я поспешно юркнул в первую попавшуюся дверь и оказался в оружейной лавке. Здесь были образцы самого современного вооружения и всяких сопутствующих штучек типа маскировочных пингвиньих и овечьих шкур. Мое внимание привлек меч, лежавший отдельно от всего остального. Осмотрев его, я понял, что мне невероятно повезло. Это было взаправдашнее, неломающееся магическое оружие. К тому же, одушевленное! Меч постоянно отпускал какие-то шуточки о кровавых битвах и жажде убийства... Немного садистский уклон. Ну да ладно. Магическому оружию в душу не смотрят - особенно когда нет выбора.

Прихватив меч, я решил, что неплохо бы прогуляться за городом. Ознакомиться с сельской жизнью, и все такое. Городские ворота оказались заперты, но ключ, который у меня был, прекрасно к ним подошел. Ворота отворились, и я оказался на широкой удобной дороге. Ею обычно пользовались (пользуются и теперь) работорговцы. Они погоняют своих мулов, тянущих тяжелые фургоны со стонущими несчастными рабами. Увы! Эту позорную практику еще не искоренили.

Никаких фургонов и рабов видно не было, и освобождать мне было не кого, поэтому я отправился по дороге на восток. Она лениво изгибалась, постоянно меняя направление и петляя между невысоких холмов с изяществом мангуста-астматика, изнывающего от жары. Ежик, бегущий по обочине, остановился и с любопытством уставился на меня. Мне почудилось что на его морде появилась усмешка, когда я, тяжело дыша, остановился и вытер пот со лба своим любимым розовым платочком. У меня появилось сильное желание запустить чем-нибудь тяжелым в эту милую, усмехающуюся зверюгу, но под рукой ничего не было, и к тому же я являюсь членом общества охраны диких животных. Добравшись до очередного поворота, я увидел указатель с многообещающими надписями: "ТУДА" и "СЮДА" Уверен, что мне это очень помогло, а когда я заметил маленькое объявление о том, что недалеко на западе живет старая

цыганка по имени Хаггис, то решил посетить милую старушку. Она сидела на крыльце своей аляповато разукрашенной хижины и бормотала что-то о превращении непрошенных гостей в послушных вьючных животных. Я почувствовал, что начинаю трястись от страха, и поспешил удалиться, не забыв захватить приятную на вид вещицу - хрустальный шар.

Улизнув от цыганки, я шагал дальше по дороге, все больше удаляясь от Анваса и все больше беспокоясь. Меня начали одолевать подозрения, что дорога может привести меня к логову какого-нибудь жуткого дракона, который спалит меня в одно мгновение своим огненным дыханием... Тут я заметил валяющуюся на дороге говорящую репу. Говорящие репы сами по себе не редкость, а эта вообразила себя комиком и все время хохмила. После нкоторого размышления я решил не брать ее с собой. Знаете, в неподходящий момент шуточка, какой бы безобидной она ни была, может стоить головы.

Я свернул с дороги на тропинку, ведущую к поместью Аранома. Мне рассказывали, что здесь живут необычные люди, любящие все зеленое и имеющие странное представление о чести. Мне было любопытно. Первой, кого я встретил в поместье, была Убийственная Вики. Ее внешность просто... убивала. Очарованный, я что-то невнятно пробормотал, и с неуклюжим поклоном протянул ей хрустальный шар. Как я и думал, вещица Вики понравилась. Она издала радостный вопль, взяла меня за руку и провела в уединенную комнату поместья. Я был брошен на кровать и с интересом наблюдал, как Вики стаскивает с себя одежду. (Странно, но на кассете с игрой не было надписи "До 16 лет" Наверное, мне попалась пиратская копия). Что-то твердое давило мне под лопатку, и я обнаружил, что это - забытая кем-то отмычка. Не долго думая, я сунул ее в карман и вновь обратил внимание на Вики... По вполне понятным причинам я не буду описывать следующий непосредственно за этим определенный промежуток времени. Скажу только, что расстались мы довольные собой и друг другом.

Я бодро зашагал дальше по дороге и вскоре добрался до перекрестка, где обнаружил зеленый запах. Вы знаете, что запахи бывают приятные и довольно отвратительные, но оказывается они бывают еще и цветные. Кроме того, здесь мне повстречался Гарри-Хиппи, который нечленораздельно пробормотал что-то доброжелательное и тут же удалился. В пещере (к западу от перекрестка) я обнаружил немного клея (чего только не оставляют после себя эти спелеологи!). На востоке же оказался лагерь Робина Пуда и Его-Не-Совсем-Женатых-Мужчин. Сам Робин лежал на траве и флегматично жевал какую-то соломинку. Его приятелей не было видно, и я решил, что сейчас самый подходящий момент, чтобы грабануть знаменитого разбойника. В конце концов, он ведь обещал отдавать награбленное бедным, а у меня (клянусь!) не было в карманах ни монетки. Я размахнулся своим магическим мечом и разумеется, промахнулся.

Заговоренный, "неломающийся" меч сломался с оглушительным "ЗВЯК!", и я обнаружил, что стою перед Робином с обломками рукояти в руках. Разбойник подозрительно уставился на меня, и я, со значением повертев рукоятью перед его носом, произнес что-то вроде: "А знаешь, Роб, ей удобно щелкать всякие вещи! Как орехи, понимаешь? Щелк, Щелк" Робин, очевидно, об этом и не подозревал и был сильно удивлен. Я поспешил удалиться - мало ли что взбредет ему в голову!

Вновь, стоя на перекрестке, я рассматривал большую гору, возвышавшуюся на севере. Несколько минут ходьбы, и вот я у подножия горы. Здесь я увидел очень интересную каменную плиту. За такими КАМЕННЫМИ ПЛИТАМИ обычно скрываются ТАЙНЫЕ ТОННЕЛИ, которые ведут в МРАЧНЫЕ ПЕЩЕРЫ, где ПОЛНЫМ ПОЛНО всяческих СОКРОВИЩ. Мысли о сокровищах меня сильно взволновали, и я попытался "щелкнуть" плиту. Как орех, понимаете? Щелк. Щелк. С жутким скрипом плита отодвинулась, и пройдя по тайному магическому тоннелю, я оказался в пещере внутри горы.

Со стен свисала паутина, что наводило на неприятные мысли о громадных пауках. Но тут я увидел ТАКОЕ, что у меня перехватило дыхание! Это были Потерянные Жесткие Доспехи! Старые и потертые, порядком изношеные и измятые, они, тем не менее, являлись священной реликвией нашей страны. Давным-давно один из наших доблестных королей-воителей сражался в этих доспехах в битве против орд злобных кочевников вентритардианцев. Каким-то чудом ему удалось победить. Так он спас нашу страну и покрыл себя неувядаемой славой, после чего отправился в поход, дабы сразить гигантского паука кровопийцу по имени Склоб... С тех пор его больше никто не видел. Ходят слухи, что вентритардианцы вновь собираются напасть на нашу страну. Если доставить эти доспехи нынешнему правителю - наши шансы на победу неизмеримо вырастут размышлял я.

Тут я сообразил, что стою посреди пещеры самого паука Склоба и громко разговариваю сам с собой. Мысль о том, что Склоб может появиться с минуты на минуту меня так взволновала, что я схватил доспехи и бежал из пещеры со всей быстротой, на которую был способен. Пусть другие сражаются с гигантскими пауками! Вот так. Я оказался в прекрасной долине. Благоухали цветы. Чудесные деревья... Мимо меня, громко икая и слегка покачиваясь, прошел Пипкин - всем известный волшебник. Он был закутан в ярко-алый плащ и черные сатиновые одежды, и от него несло перегаром. Дорога вела меня на восток и вскоре привела к месту, где валялся позабытый кем-то сундук с золотом. Рядом сидела лягушка. Она задумчиво посмотрела на меня и громко квакнула: "Ква! Ква!" Лягушки не очень-то разговорчивы. На мгновенье у меня промелькнула мысль о заколдованной принцессе, но целоваться с этим земноводным... Нет уж, спасибо! Придется ей побыть лягушкой еще немного. Я подхватил сундук и поспешно отправился

на юг. Лягушка призывно заквакала мне вслед, но я притворился, что не слышу и ускорил шаг...

Дорога внезапно оборвалась перед замерзшим озером. Я озадаченно поглядел на небо. Жаркое летнее солнце сияло в вышине, но, тем не менее, воды озера были скованы льдом. Это было подозрительно. Дело явно не обошлось без магии. Я припомнил пьяного Пипкина и содрогнулся. Эти маги способны на все, что угодно, особенно если напьются. Пересекать озеро, замороженное вдрызг пьяным волшебником, казалось очень опасным. Я задумался и, балуясь с тюбиком клея, капнул немного себе на ботинки. Сдобренные магическим клеем ботинки, мгновенно перенесли меня через озеро. Я оказался у ворот Дрос Дельнокс - величайшей крепости страны Кренай (мост страны), которой правили ныне: Пек, граф Жести (это его дедуле принадлежали Потертые Жестяные Доспехи) и Пусс Легендарный (резвый девяностодвухлетний старикан).

Ворота оказались запертыми, но моя отмычка легко справилась с замком. В главном зале Дрос Дельнокс я удостоился чести увидеть графа Пека и Пусса Легендарного. "Пьяные в доску" правители отплясывали что-то несуразное под громкие звуки безумной джазовой мелодии, несущейся из скрытых динамиков.

Я протянул доспехи графу. Ух! - сказал Пек. - Древние доспехи моих предков! Как раз то, что нужно!

С этими словами граф пал на пол и громко захрапел. Ты спас Кренай! сказал достойнейший Пусс Легендарный, тыча мне пальцем в живот, и последовал примеру графа. Конечно, Пек и Пусс не могли больше ничего сказать (пока не отоспались), но их многочисленные родственники, советники и придворные от души поздравили меня и заверили, что я освобождаюсь от уплаты налогов (пожизненно) и получу солидную награду за то, что вернул Потертые Жестяные Доспехи.

Ну что ж, награду я заслужил и приму с благодарностью. Еще бы! Ведь я, Моронни Безумный Зверь, спас свою страну!

****

РУСАЛКА

По игре (MERMAID MADNESS)

(с) Алексей Березин, г.Колпашево, Томской обл.

Мигель Мартинес был настоящим мужчиной. От далекого предка конкистадора ему достались в наследство глубокие синие глаза под черными бровями, черная же вечно взлохмаченная шевелюра, густые усы под ястребиным носом и худое острое лицо, обтянутое смуглой кожей короче говоря, все, что безоговорочно нравится женщинам. Сеньору Мартинесу женщины тоже нравились, однако, дожив до тридцати лет, он так и не женился, ибо будучи настоящим мужчиной, умел в нужный момент твердо сказать "нет"

Впрочем, был у сеньора Мартинеса и один недостаток совершенно незначительный он любил приврать. Друзья, конечно, прощали ему эту слабость, уж очень занятно было слушать рассказы Мигеля о его невероятных любовных похождениях; ну, а подруги этой слабости и вовсе не за-

мечали поскольку им о своих приключениях он не рассказывал. И вот сидел однажды этот замечательный сеньор Мартинес в своем маленьком дворике на зеленой травке и закачивал воздух в газовые баллоны, потому что был он никем иным, как водолазом. Он как раз подсоединил последний баллон к компрессору, когда калитка в заборе вдруг отворилась и вошел... Как вы думаете, кто? Впрочем, не пытайтесь угадать, все равно не получится. Это был мистер Рокуэлл из Калифорнии, старый друг сеньора Мартинеса. Друзья не виделись уже довольно давно, и новостей у них накопилось порядком. Поэтому, крепко пожав друг другу руки, они прошли в дом сеньора Мартинеса и, расположившись на веранде, занялись распитием бутылки виски. Уже после второго стаканчика язык сеньора Мартинеса развязался, и он поведал другу дивную историю о якобы влюбившейся в него русалке, и еще какую-то чушь. Когда же мистер Рокуэлл выразил, мягко выражаясь, сомнения в правдивости рассказа, Мигель поклялся ему на библии (специально для таких случаев лежавшей у него всегда на столе)... Однако мистера Рокуэлла не стоило причислять к легковерам. И вот тогда Впрочем, я сильно забежал вперед. Начну по порядку. А знаешь, Джек, какие тут у нас девушки! - восторженно говорил Мигель, сочно чмокая свои сложенные щепотью пальцы. - Между прочим, я на той неделе встретил такую чикиту, ты просто упадешь... Она русалка!

Н-да? приподняв брови, вопросил Джек, пытаясь на глаз оценить, не пора ли Мигелю завязывать с виски.

Ну да! радостно ответил Мигель, и хитро прищурив левый глаз, громким шепотом добавил: С ней у меня такое приключилось!.. Расскажу - не поверишь.

Джек утвердительно кивнул.

Так вот, продолжил Мигель уже в полный голос, - сижу я как-то на пирсе, ну, знаешь, там, где палаточный городок? Так вот, сижу и надеваю свою амуницию готовлюсь к погружению. Люблю поплавать утречком в одиночестве... И вдруг прямо на меня из-за кустов выскакивает кто-то лохматый, здоровущий... Знаешь, я не трус, но от неожиданности так и поскакал по пирсу. На бегу оглянулся - а она за мной чешет!..

Кто? поинтересовался Джек. - Русалка?

Ага, - кивнул Мигель.

С рыбьим хвостом? А? Нет, без хвоста... Да что без хвоста совсем без одежды! Тут мне совсем страшно стало. Много, конечно, за мной женщин бегало, но чтоб такой комплекции!..

А что с ней? В теле? У нее рука как моя нога!.. Ну, я и решил - отсижусь под водой, баллона мне на час хватит, залягу на дно, уж там-то она меня не достанет. Ныряю с пирса и что ты думаешь? Эта сумасшедшая ныряет следом! Я, не оглядываясь, заныриваю поглубже - там под пирсом года три назад затонул паровой катер и собираюсь спрятаться в трюме. Внутрь-то проплыл да зацепил какую-то железяку и обрушил старый трап. И упал он аккурат поперек прохода, что ни проплыть, ни проехать... Ну, Джек,

представь мое положение совершенно один в старом затопленном корыте, и ни одна живая душа не знает, что со мной...

Кроме русалки ехидно заметил Джек.

Ну да. Так ведь она меня и выручила. Я уже последнюю надежду потерял, как вдруг трап хрясь! - и пополам! И она вплывает... Да ты не смейся Джек. Ну она, конечно, не фотомодель, но тоже ничего. Полненькая, рыженькая... А хвост у нее какой шикарный, с серебристой чешуей, между прочим!

Ну да, вроде как у селедки? Ох и силен ты врать, дружище Мигель! Ты же десять минут назад говорил, что у нее нет хвоста! усмехнулся Джек.

А... Ну, понимаешь, в замешательстве протянул Мигель, она, вроде, как по суше - так у нее ноги, а как по воде - так вместо ног хвост вырастает... Ты что мне не веришь? Ну что ты, Мигель, конечно верю! - воскликнул Джек, и его интонация ясно говорила: о, боже, конечно - нет!

Ну хочешь, я тебе на библии поклянусь? - и не дожидаясь ответа, Мигель возложил одну руку на библию, а другую поднял вверх.

Клянусь честью, что все так и было!

Мистер Рокуэлл собирался было сказать еще что-то язвительное, но тут в дверь постучали, и сеньор Мартинес, уронив стул, побежал открывать. За дверью стояла розовощекая толстушка в розовом платье, рыжая и курносая. "Около двадцати пяти", определил Джек наметанным глазом.

Мигель перекинулся с деввицей парой слов и повернулся к Джеку.

Слушай, мы тут собираемся на пляж... Пойдешь с нами?

Джек замялся.

Да нет, дела, знаешь ли, и вообще... Как-нибудь в другой раз.

Как знаешь - пожал плечами Мигель, - До встречи, Джек.

И обняв девушку за плечи, он неторопливо удалился. Джек какое-то время разглядывал этикетку на недопитой бутылке виски, а затем встал и направился следом за Мигелем и его подружкой. Спроси его сейчас кто-нибудь, зачем он это сделал - едва ли он смог бы ответить. На пляже было почти пусто десяток одиноких купальщиков. Окинув взглядом залив, Джек сразу заметил ту парочку, за которой шел. Они плескались метрах в ста от берега. Вон Мигель поднырнул под свою девушку и испугал, внезапно схватив ее под водой. Вот он вынырнул, сердитая толстушка осыпает его брызгами, хлопая рукой по воде. А вот Мигель обхватил ее и повалил на спину. Широкий серебристый рыбий хвост поднялся в воздух и шлепнул по воде, подняв тучу брызг.

...Впрочем, может это были лишь солнечные блики на воде?..

****

ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ

В недавно выпущенной нами книге П.Федина "Полное описание и полный дизассемблер TR DOS 5.04 (5.03)" из-за технологической ошибки не был напечатан последний файл полного дизассемблера. Приносим автору и читателям свои извинения и исправляем допущенную оплошность. Итак:

Адрес 15361. *То же, что и 15665.*

15361 JR 15366

Адрес 15364. *То же, что и 15642*

15364 JR 15369

Адрес 15366. *То же, что и 15665.*

15366 JP 15616

Адрес 15369. *То же, что и 15642.*

15369 JP 15619

Адрес 15610. *То же, что и 8433.*

15610 JP 8433

Адрес 15613. *То же, что и 15635, но не переключает ПЗУ*

15613 JP 10300

Адрес 15616. *Вход в TR-DOS.*

⇒ Переключает ПЗУ

15616 NOP
JR 15665

Адрес 15619. *Выполнение команд TR-DOS из бейсика.*

⇒ Установите CH_ADD на префикс :REM: перед командой. Возвращает в BC код ошибки. Переключает ПЗУ

15619 NOP
JR 15642

Адрес 15622. *Подпрограмма ввода из файла данных.*

⇒ Переключает ПЗУ. При вызове из редактора вводит всю строку и производит выход из редактора, иначе возвращает в A символ из файла.

15622 NOP
JP 9711

Адрес 15626. *То же, что и 15629, но не переключает ПЗУ*

15626 JP 9290

Адрес 15629. *Подпрограмма вывода в файл данных.*

⇒ Символ поместите в аккумулятор. Используется также с адреса 15630. Обе точки входа переключают ПЗУ

15629 NOP
15630 JR 15626

Адрес 15632. *То же, что и 8433.*

⇒ Переключает ПЗУ.

```
15632   NOP
        JR      15610
```

Адрес 15635. *Переход на подпрограмму вызова подпрограмм TR-DOS из машинного кода.*

⇒ Адрес является переключателем.

```
15635   NOP
        JR      15613
```

Адрес 15638. *Переход на подпрограмму обработки ошибок.*

⇒ Адрес является переключателем.

```
15638   NOP
        JP      12137
```

Адрес 15642. *То же, что и 15619, но не переключает ПЗУ*

```
15642   CALL    15649   ;создание системных переменных TR-
                        DOS
        PUSH    HL      ;помещение на стек адреса переключа-
                        теля ПЗУ на бейсик
        JP      364     ;выполнение команды
```

Адрес 15649. *Создание области системных переменных TR-DOS, если ее нет.*

⇒ В HL помещается 23746.

```
15649   CALL    12787       ;проверка наличия области систем-
                            ных переменных
        NOP
        NOP
        CALL    C,15692     ;если ее нет, то создание ее
        LD      HL,23746    ;загрузка в HL адреса переключателя
                            ПЗУ на бейсик
        RET                 ;возврат
```

Адрес 15663. *Переключение пзу на TR-DOS и переход на любой адрес.*

⇒ Адрес поместите на стек.

```
15663   NOP
        RET
```

Адрес 15665. *Вход в TR-DOS.*

```
15665   CALL    15649   ;создание области системных перемен-
                        ных
        PUSH    HL      ;помещение на стек адреса переключа-
                        теля ПЗУ на бейсик
        JP      569     ;вход в TR-DOS
```

Адрес 15672. *Проверка наличия интерфейса-1.*

⇒ В случае его обнаружения происходит создание его системных переменных и в 23793 помещается 1. Для работы подпрограмма должна быть перемещена в любое место ОЗУ.

```
15672  XOR   A          ;интерфейс-1 есть ?
       OUT   (247),A
       IN    A,(247)
       CP    30
       JR    Z,15684    ;если да, то создание его системных
                        переменных
       CP    31
       RET   NZ         ;если нет, то возврат
15684  RST   8          ;создание системных переменных
                        интерфейса-1
       DEFB  #31
       LD    A,1        ;по команде SAVE записывать 1 ко-
                        пию
       LD    (23791),A
       RET              ;возврат
```

Адрес 15692. *Создание системных переменных TR-DOS.*

⇒ При наличии интерфейса-1 перед созданием переменных TR-DOS создаются его переменные и в 23791 помещается 1. В переменные TR-DOS помещаются:

- в 23802 - 23804, 23752 - 23754, 23610, 23830 и 23820 - 255;
- в 23831, 23833, 23832, 23823 и 23839 - 0;
- в 23746 - 201.

Также в системный регистр помещается 255, а на ВГ93 выдается команда прерывания. Подпрограмма портит ячейки памяти калькулятора.

```
15692  XOR   A          ;выбор дисковода A: и сброс ВГ93
       OUT   (255),A
       IN    A,(246)    ;это ничего не дает
       LD    HL,15672   ;адрес подпрограммы проверки на-
                        личия интерфейса-1
       LD    DE,23698   ; любой свободный адрес, ОЗУ
       LD    BC,20      ;длина подпрограммы
       LDIR             ;перенос подпрограммы в ОЗУ
       LD    HL,15719   ;адрес возврата из подпрограммы
       PUSH  HL         ;помещение его на стек
       LD    HL,15663   ;адрес переключателя ПЗУ на TR-
                        DOS
       PUSH  HL         ;помещение его на стек
```

```
        JP      23698       ;запуск подпрограммы
```

Адрес 15719. *Создание системных переменных TR-DOS.*

⇒ В переменные помещаются:

- в 23802 - 23805, 23752 - 23755, 23610, 23830 и 23820 - 255;
- в 23831, 23833, 23832, 23823 и 23839 - 0;
- в 23746 - 201.

Также в системный регистр помещается 255, а на ВГ93 выдается команда прерывания.

```
15719   LD      HL,12176    ;адрес возврата из пзу бейсика
        PUSH    HL          ;помещение его на стек
        LD      HL,15663    ;адрес переключателя ПЗУ на TR-
                            DOS
        PUSH    HL          ;помещение его на стек
        LD      HL,5717     ;адрес подпрограммы MAKE_ROOM
                            из ПЗУ бейсика
        PUSH    HL          ;помещение его на стек
        LD      HL,23551    ;адрес переключателя ПЗУ на бейсик
        PUSH    HL          ;помещение его на стек
        LD      (HL),201    ;помещение в озу команды RET
        LD      HL,23773    ;начальный адрес области системных
                            переменных
        LD      BC,112      ;длина ее
        RET                 ;запуск подпрограммы MAKE_ROOM
```

Адрес 15744. *Перевод строки.*

⇒ Используется также с адреса 15746 (печать символа из аккумулятора). Также эта подпрограмма ошибочно используется с адреса 15755 подпрограммой 1270.

```
15744   LD      A,13        ;символ ENTER
15746   PUSH    HL          ;сохранение регистров
        PUSH    BC
        PUSH    DE
        PUSH    AF          ;сохранение символа
        CALL    8433        ;изменение памяти
        POP     AF          ;восстановление символа
15754   CALL    15764       ;печать символа
        CALL    8433        ;восстановление памяти
        POP     DE          ;восстановление регистров
        POP     BC
        POP     HL
        RET                 ;возврат
```

Адрес 15764. *Печать символа без изменения памяти.*

⇒ Символ поместите в аккумулятор.

```
15764   RST     32
        DEFW    #0010
        RET
```

Адрес 15768. *Восстановление с ожиданием INTRQ.*

⇒ Используется также с адреса 15770 (отдача команды из рег. A с ожиданием INTRQ).

```
15768   LD      A,8          ;восстановление
15770   OUT     (31),A
15772   PUSH    HL           ;сохранение HL
        RST     32           ;проверка нажатия BREAK
        DEFW    #1F54
        JR      C,15781      ;если нажато, то сообщение
                             *BREAK*
        RST     32
        DEFW    #1B7B
15781   POP     HL           ;восстановление HL
        IN      A,(255)      ;INTRQ поступил ?
        AND     128
        JR      Z,15772      ;если нет, то повтор
        RET                  ;возврат
```

Адрес 15789. *Проверка наличия диска в дисководе.*

⇒ Никаких установок не требуется.

```
15789   LD      A,8          ;команда восстановления
        CALL    15770        ;выдача команды и ожидание INTRQ
        LD      DE,0         ;за 65536 проходов индексный сигнал
                             должен измениться
        IN      A,(31)       ;берем индексный сигнал для образ-
                             ца
        AND     2
        LD      B,A          ;запоминаем его
15802   IN      A,(31)       ;берем его снова
        AND     2
        CP      B            ;он изменился ?
        RET     NZ           ;если да, то возврат
        INC     DE           ;следующая попытка
        LD      A,E          ;попытки кончились
        OR      D
        JR      NZ,15802     ;если нет, то повтор
        JP      16103        ;обработка ошибки NO DISC
```

Адрес 15816. *Выбор дисковода, определенного по умолчанию.*

⇒ Вход: можно установить время перемещения головки и режим дисковода согласно описанию команды #01.

⇒ Выход: эти переменные могут быть изменены согласно этому описанию. Используется также с адресов:

- 15819 выбор дисковода. Номер задайте в рег. A, остальное как и для предыдущей подпрограммы.
- 15869 - задержка в 725779 тактов.
- 15871 - задержка любой нужной длительности. В рег. A установите число, определяющее длительность задержки.

```
15816  LD     A,(23833)   ;берем номер дисковода по умолчанию
15819  LD     (23798),A   ;установка номера выбранного дисковода
       LD     HL,23830    ;выбор дисковода
       LD     C,A
       LD     A,60
       OR     C
       OUT    (255),A
       LD     (HL),A
       CALL   15880       ;берем время перемещения головки дисковода
       AND    128         ;бит 7 выключен ?
       JR     Z,15866     ;если да, то инициализацию не проводим
       CALL   15789       ;проверка наличия диска в дисководе
       CALL   15894       ;определение времени перемещения головки дисковода
       CALL   15889       ;берем тип дисковода
       CP     255         ;в этой переменной 255 ?
       JR     Z,15866     ;если да, то не определяем его
       PUSH   HL          ;сохраняем адрес переменной с типом дисковода
       CALL   8138        ;определение количества дорожек дисковода
       POP    HL          ;восстановление адреса переменной типа
       CP     80          ;установка этой переменной
       LD     A,0
       JR     NZ,15865
       LD     A,128
15865  LD     (HL),A
```

Внимание, ошибка!!! Не проверяется количество сторон дисковода и предполагается, что он односторонний.

```
15866  CALL   7734        ;проверка индексной области диска
15869  LD     A,0         ;задержка в 725779 тактов
```

```
15871   LD      C,255
15873   DEC     C
        JR      NZ,15873
        DEC     A
        JR      NZ,15871
        RET                     ;возврат
```

Адрес 15880. *Выборка времени перемещения головки текущего дисковода.*

⇒ Установите 23798. На выходе в A будет время перемещения головки дисковода, а в HL - адрес, по которому оно находится. Используется также с адреса 15883 (выборка параметра дисковода; в DE поместите адрес таблицы параметров и установите 23798; На выходе в A будет параметр, в HL - адрес его ячейки).

```
15880   LD      DE,23802        ;берем адрес таблицы скоростей пе-
                                 ремещения головок
15883   LD      HL,(23798)      ;смещение равно номеру дисковода
        ADD     HL,DE           ;вычисление адреса ячейки с пара-
                                 метром
        LD      A,(HL)          ;берем оттуда параметр
        RET                     ;возврат
```

Адрес 15889. *Выборка типа дисковода.*

⇒ Вход: установите 23798.

⇒ Выход: в A тип дисковода, в HL - адрес его ячейки.

```
15889   LD      DE,23752        ;берем адрес таблицы типов дисково-
                                 дов
        JR      15883           ;выборка параметра и возврат
```

Адрес 15894. *Определение времени перемещения головки дисковода.*

```
15894   CALL    15880           ;берем адрес ячейки с временем пе-
                                 ремещения головки
        LD      B,8             ;начнем с наименьшего
        LD      C,4             ;4 попытки
15901   LD      (HL),B          ;установка времени перемещения
                                 головки
        LD      A,8             ;восстановление
        CALL    15770
        LD      A,32            ;позиционирование на дорожку 32
        LD      B,11
        CALL    15940
        LD      B,(HL)          ;берем время перемещения головки
        LD      A,1             ;позиционирование на дорожку 1 с
                                 заданным временем
        CALL    15940
```

```
        IN      A,(31)      ;головка на дорожке 0 ?
        AND     4
        JR      NZ,15935    ;если да, то ошибка
        XOR     A           ;позиционирование на дорожку 0
        IN      A,(31)      ;головка на дорожке 0 ?
        AND     4
        RET     NZ          ;если да, то возврат
15935   INC     B           ;увеличение времени перемещения
                            головки на 1
        DEC     C           ;следующая попытка
        RET     Z           ;если попытки кончились, то возврат
        JR      15901       ;повтор
```

Адрес 15940. *Позиционирование.*

⇒ В А поместите физический номер дорожки, в В - время перемещения головки и выберите сторону.

```
15940   OUT     (127),A     ;помещение номера дорожки в регистр
                            данных
        LD      A,B         ;получение команды позиционирования
        OR      24
        JP      2048        ;позиционирование с быстрой скоро-
                            стью
```

Внимание!!! Здесь нужно JP 15070, т.к. 2048 гасит биты скорости перемещения головки дисковода, следовательно позиционирование работает только с быстрой скоростью.

Адрес 15948. *Позиционирование.*

⇒ На входе в А должен быть физический номер дорожки, а в В - время перемещения головки дисковода.

```
15948   OUT     (127),A     ;загрузка номера дорожки в регистр
                            данных
        PUSH    BC          ;сохранение времени перемещения
                            головки дисковода
        LD      B,A         ;головка уже спозиционирована ?
        IN      A,(63)
        CP      B
        POP     BC          ;восстановление времени перемеще-
                            ния головки
        PUSH    AF          ;сохранение результата проверки
        LD      A,B         ;получение команды позиционирова-
                            ния
        OR      24
        CALL    2048        ;преобразование и выдача команды и
                            ожидание INTRQ
```

ВОЗВРАЩАЯСЬ К НАПЕЧАТАННОМУ

Внимание!!! Позиционирование осуществляется только с быстрой скоростью перемещения головки, т.к. в 2048 используется AND 252. Для исправления ситуации здесь нужно поставить JP 15770.

```
POP     AF      ;восстановление результата проверки поло-
                 жения головки
RET     Z       ;если головка была спозиционирована, то
                 возврат
PUSH    BC      ;сохранение времени перемещения головки ?
CALL    15869   ;задержка в 725779 тактов
POP     BC      ;восстановление времени перемещения го-
                 ловки
RET             ;возврат
```

Адрес 15971. *Позиционирование головки дисковода.*

⇒ Вход: в рег. A установите логический номер требуемой дорожки и установите 23757. На выходе 23757 обнуляется. Используется также с адреса 16032 (задержка в 2168937 тактов).

```
15971   LD      C,A         ;загрузка номера дорожки в регистр
                             C
        CALL    8171        ;выбор верхней стороны диска
        CALL    15889       ;берем тип дисковода
        AND     2           ;дисковод двухсторонний ?
        CALL    NZ,16042    ;если да, то выбор стороны
        PUSH    BC          ;сохраним номер дорожки
        BIT     7,(HL)      ;дисковод 40-дорожечный ?
        JR      Z,16003     ;если да, то номера дорожек не ме-
                             няем
        BIT     0,(HL)      ;используем только 40 дорожек ?
        JR      NZ,16003    ;если нет, то номера дорожек не
                             меняем
        IN      A,(63)      ;берем номер текущей дорожки
        CP      C           ;головка уже спозиционирована ?
        JR      Z,16002     ;если да, то номера дорожек не ме-
                             няем
        RLCA                ;делим номер текущей дорожки на 2
        OUT     (63),A
        LD      A,C         ;делим номер требуемой дорожки на
                             2
        RLCA
16002   LD      C,A
16003   CALL    15880       ;берем время перемещения головки
                             дисковода
        LD      B,A
        IN      A,(63)      ;головка уже спозиционирована ?
```

```
        CP      C
        PUSH    BC          ;сохраним номер требуемой дорож-
                            ки
        CALL    NZ,15869    ;если головка не спозиционирована,
                            то задержка в 725779 тактов
        POP     BC          ;восстановление номера требуемой
                            дорожки
        LD      A,C
        CALL    15948       ;позиционирование
        POP     BC          ;восстановление номера требуемой
                            дорожки
        LD      A,C         ;установка регистра дорожки
        OUT     (63),A
        LD      A,(23757)   ;нужна задержка после операции ?
        OR      A
        RET     Z           ;если нет, то возврат
        XOR     A           ;в следующий раз задержка не по-
                            надобится
        LD      (23757),A
16032   LD      B,3         ;задержка в 2168937 тактов
16034   LD      A,255
        CALL    15871
        DJNZ    16034
        RET                 ;возврат
```

Адрес 16042. *Выбор стороны при позиционировании.*

⇒ Вход: в C поместите логический номер дорожки и выберите верхнюю сторону диска.

⇒ Выход: будет выбрана сторона, а в C будет физический номер дорожки.

```
16042   LD      A,C         ;деление номера дорожки на 2
        OR      A
        RRA
        LD      C,A
        RET     NC          ;если номер дорожки был четный, то воз-
                            врат
        JP      8182        ;иначе выбор нижней стороны диска
```

Адрес 16050. *Проверка индексной области дорожки.*

⇒ Установите 23761 и поместите в B время перемещения головки дисковода. Выбирается верхняя сторона и при ошибке в 23831 помещается 255. В регистр H помещается номер текущей дорожки. Используется также с адреса 16103 (обработка ошибки NO DISC).

```
16050   CALL    8171        ;выбор верхней стороны диска
        IN      A,(31)      ;берем состояние контроллера
```

```
        AND     128         ;сохранение флага готовности
        LD      (23757),A
        IN      A,(63)      ;берем номер текущей дорожки
        LD      H,A         ;на случай возврата
        CALL    15940       ;преобразование и выполнение ко-
                            ;манды из рег. B
```

Внимание!!! Перед CALL 15940 в регистр B надо поместить время перемещения головки дисковода, иначе команда, которая выполнится перед проверкой зависит от содержимого регистра B на входе в подпрограмму и последствия могут быть весьма плачевными.

```
        LD      C,127       ;адрес регистра данных
        LD      D,1         ;число,определяющее количество
                            ;попыток
        DI                  ;запрет прерываний
        LD      A,192       ;чтение индексной области
        OUT     (31),A
        PUSH    BC          ;сохранение адреса регистра дан-
                            ;ных
        LD      B,3         ;определяем количество попыток
16078   IN      A,(255)     ;берем состояние сигналов DRQ и
                            ;INTRQ
        AND     192
        JR      NZ,16144    ;если какой-нибудь сигнал поступил,
                            ;то начало чтения
        INC     DE          ;следующая попытка
        LD      A,E
        OR      D
        JR      NZ,16078
        DJNZ    16078       ;повтор до конца попыток
        POP     BC          ;восстановление адреса регистра
                            ;данных
        EI                  ;разрешение прерываний
        LD      A,208       ;остановка контроллера
        OUT     (31),A
        LD      A,(23761)   ;ошибку игнорировать ?
```

Ошибка!!! Надо LD A,(23831).

```
        CP      255
        RET     Z           ;если да, то возврат
16103   CALL    10027       ;установка TAPE LOADING ERROR
        LD      A,255       ;в следующий раз ошибку игнориро-
                            ;вать
        LD      (23831),A
        JP      10011       ;обработка ошибки NO DISC
```

```
16114   POP   BC          ;восстановление адреса регистра
                          данных
        IN    H,(C)       ;чтение первого байта
16117   IN    A,(255)     ;ожидание какого-нибудь сигнала
        AND   192
        JR    Z,16117
        EI                ;на случай возврата
```

Ошибка!!! Так можно потерять байт при чтении.

```
        RET   M           ;если поступил INTRQ, то возврат
        DI                ;запрет прерываний
        IN    A,(127)     ;чтение байта
        JR    16117       ;повтор
```

Адрес 16130. *В комментариях не нуждается.*

```
16130   LD    (23807),A
        RET
```

Адрес 16134. *Похоже на предыдущее.*

```
16134   LD    (23808),HL
        RET
```

Адрес 16138. *Запись сектора.*

⇒ Спозиционируйте головку и установите 23807 и 23808.

```
16138   LD    A,160       ;команда записи сектора
        JR    16144       ;переход к подпрограмме чтения / записи
                          сектора
```

Адрес 16142. *Загрузка сектора.*

⇒ Установите 23807 и 23808 и спозиционируйте головку. Используется также с адреса 16144 (загрузка / запись сектора; установите все для 16142 плюс в A поместите команду чтения или записи сектора).

```
16142   LD    A,128       ;команда чтения сектора
16144   LD    (23806),A   ;установка команды
16147   LD    D,10        ;10 попыток
16149   PUSH  DE          ;сохранение счетчика попыток
        DI                ;запрет прерываний
        LD    A,(23807)   ;установка сектора
        INC   A
        OUT   (127),A
        LD    HL,(23808)  ;берем адрес чтения / записи
        LD    C,127       ;берем адрес регистра данных
        LD    A,(23806)   ;отдаем команду
        OUT   (31),A
        CP    160         ;была команда записи сектора ?
        PUSH  AF
        CALL  Z,16314     ;если да, то запись данных
```

```
        POP     AF
        CALL    NZ,16341    ;иначе чтение данных
        POP     DE          ;восстановление счетчика попыток
        EI                  ;разрешение прерываний
        IN      A,(31)      ;берем состояние контроллера
        LD      B,A         ;сохраняем его
        AND     127         ;все в порядке ?
        RET     Z           ;если да, то возврат
        LD      HL,10712    ;адрес сообщения READ ONLY
        AND     64          ;запрещена запись ?
        JR      NZ,16203    ;если да, то переход на обработку
                            ошибки
        LD      A,B         ;восстановление состояния кон-
                            троллера
        AND     4           ;была потеря данных ?
        JR      Z,16288     ;если нет, то обработка дисковых
                            ошибок
        DEC     D           ;следующая попытка
        JR      NZ,16149    ;если попытки не кончились, то по-
                            втор
10200   LD      HL,10722    ;адрес сообщения DISC ERROR
16203   LD      A,208       ;команда прерывания
        OUT     (31),A
        LD      A,B         ;восстановление состояния кон-
                            троллера
        AND     1           ;контроллер был занят ?
        JP      NZ,16103    ;если да, то обработка ситуации
                            NO DISC
        IN      A,(63)      ;если ошибка произошла в секторе
                            10 дорожки
        OR      A           ;0, то возврат (игнорирование)
        JR      NZ,16223
        IN      A,(127)
        CP      10
        RET     Z
        PUSH    HL          ;сохранение адреса сообщения
        CALL    7575        ;очистка экрана
        POP     HL          ;восстановление адреса сообще-
                            ния
        RST     24          ;печать сообщения
        LD      HL,10771    ;печать TRK:
        RST     24
        IN      A,(63)      ;печать номера дорожки
```

```
        CALL    7587
        LD      HL,10776        ;печать SEC:
        RST     24
        IN      A,(95)          ;печать номера сектора
        CALL    7587
        LD      HL,10750        ;печать RETRY,ABORT,IGNORE?
        RST     24
16251   CALL    4178            ;ввод клавиши
        CP      "I"             ;нажато I ?
        RET     Z               ;если да, то возврат
        CP      "R"             ;нажато R ?
        JR      Z,16276         ;если да, то перепозиционирова-
                                ;ние и повтор
        CP      "A"             ;нажато A ?
        JR      NZ,16251        ;если нет, то повтор ввода клавиши
        CALL    10027           ;установка TAPE LOADING ERROR
        LD      A,7             ;дисковая ошибка
        LD      (23823),A
        JP      467             ;завершение
16276   LD      A,(23797)       ;перепозиционирование
        CALL    15971
        CALL    16032           ;задержка в 2168937 тактов
        JP      16149           ;повтор
```

Ошибка!!! Надо JP 16147 для того, чтобы обновить число попыток.

```
16288   DEC     D               ;следующая попытка
        JP      Z,16200         ;если попытки кончились, то обра-
                                ;ботка DISC ERROR
        PUSH    DE              ;сохранение счетчика попыток
        CALL    15880           ;берем время перемещения головки
                                ;дисковода
        AND     2               ;оно равно 10 ?
        JR      NZ,16301
        INC     (HL)            ;если нет, то увеличение его на 1
16301   CALL    15768           ;восстановление
        LD      A,(23797)       ;перепозиционирование
        CALL    15971
        POP     DE              ;восстановление счетчика попыток
        JP      16149           ;повтор
```

Адрес 16314. *Запись данных на диск.*

⇒ Установите в HL адрес блока данных, в DE - число, определяющее количество циклов, за которое должна начаться запись, в C - адрес регистра данных контроллера и отдайте команду записи. Использу-

ется также с адреса 16330 (то же, что и 16314, но без обработки NO DISC).

```
16314   LD      B,4         ;установка количества попыток
16316   IN      A,(255)     ;поступил какой-нибудь сигнал ?
        AND     192
        JR      NZ,16337    ;если да, то начинаем запись
        INC     DE          ;следующая попытка
        LD      A,E
        OR      D
        JR      NZ,16316
        DJNZ    16316
        RET                 ;если попытки кончились, то возврат
16330   IN      A,(255)     ;поступил какой-нибудь сигнал ?
        AND     192
        JR      Z,16330     ;если нет, то ожидание его
        RET     M           ;если поступил INTRQ, то возврат
16337   OUTI                ;вывод байта из вашей области в ре-
                            гистр данных
        JR      16330       ;повтор
```

Адрес 16341. *Чтение данных с диска.*

⇒ Все установки аналогично подпрограмме 16314, но отдать нужно команду чтения. Используется также с адреса 16357 (то же, что и 16341, но без обработки NO DISC).

```
16341   LD      B,4         ;установка количества попыток
16343   IN      A,(255)     ;поступило что-нибудь ?
        AND     192
        JR      NZ,16364    ;если да, то начинаем читать
        INC     DE          ;следующая попытка
        LD      A,E
        OR      D
        JR      NZ,16343
        DJNZ    16343
        RET                 ;если попытки кончились, то возврат
16357   IN      A,(255)     ;поступил какой-нибудь сигнал ?
        AND     192
        JR      Z,16357     ;если нет, то ждем его
        RET     M           ;если это INTRQ, то возврат
        INI                 ;чтение байта
        JR      16357       ;повтор
```

ГДЕ МОЖНО ПРИОБРЕСТИ НАШИ ИЗДАНИЯ.

1. Торговый центр фирм "ИНФОРКОМ" и "ФОРМАК"
   - г.Москва, Новый Арбат, д.2. 19-е отделение связи. Первый этаж операционного зала. Кроме воскресенья, с 10 до 14 и с 15 до 17 ч. Розничная и оптовая продажа по издательским ценам. Подписка на ZX РЕВЮ. Авторские и фирменные программы по каталогу.
   - **Почтовый адрес: 121019, Москва, а/я 16.**
2. г.Чебоксары, ул. Ярославская, д. 64, офис 24. Предпр. "Мой город"
3. г.Ростов-на-Дону, ул. Кручинина, Ростовский радиорынок. Магазин фирмы "СКОРПИОН" В будние дни с 9 до 15 ч., в вых. с 8 до 13 ч.
4. г.Йошкар-Ола, пл. Ленина, 3. Лавка радиодеталей "МОДУЛЬ"
5. г.Ангарск, Ворошилова, 10. "SPECTRUM"
6. г.Иркутск, ул. Урицкого, 2. "Все для ZX SPECTRUM"
7. г.Одесса, радиорынок, под вывеской "AcashiSoft" или по тел. 33-71-71.
8. г.Барнаул, ул. Э.Алексеевой, 2-а. тт.76-42-98, 76-42-25. Фирма КОМЭЛ.
9. О пунктах продажи в Екатеринбуге и области можно узнать, обратившись по адресу: 620133, Ек-г, до востребования, Лебедеву Дмитрию Михайловичу. Не забудьте вложить конверт с Вашим адресом.
10. г.Курск, Центральный рынок, Магазин "Хозтовары", СБ.,ВС. до 12.00.
11. г. Владивосток, ул. Светланская, 9. Магазин "Радиотовары"
12. г. Владивосток, ул. Гамарника, 18. Кинотеатр Маяк. Комп. клуб "SIGMA 7"

ВНИМАНИЮ ОПТОВЫХ ПОКУПАТЕЛЕЙ !!!

Адреса пунктов продажи изданий МКП «ИНФОРКОМ» и ТОО «ФОРМАК» публикуются БЕСПЛАТНО.

Для Вас разработана гибкая система скидок на полиграфическую продукцию (от 10% до 40%), осуществляется поставка по почте, информационная и рекламная поддержка.

ЛР № 063856 от 30.12.94 г. Подписано в печать 28.02.97 г.
Формат 60×90/16. Печать офсетная. Тираж 5000. Зак. 119.

Фирма “Формак”, 121019, Москва, а/я 16.

Отпечатано с готового оригинал-макета в типографии ИПО Профиздат, 109044, Москва, Крутицкий вал, 18. Плр № 050003 от 19.10.94 г.

ПАРАЛЛЕЛЬНОЕ ОБЕСПЕЧЕНИЕ
256К 256К
ОБМЕН ОПЫТОМ
ОБМЕН ОПЫТОМ
ИГРЫ
ИГРЫ
SPECTRUM
ZX