ВАШ ДОМАШНИЙ КОМПЬЮТЕР

всегда
попадает
в цель

РЕКЛАМА

размещенная в сборниках
научно-технической фирмы

Гранд-Информ

По вопросам размещения рекламы обращайтесь:

Адрес: 607200, Нижегородская обл., г.Арзамас-16,
п/о №9, а/я № 597 ТОО «Гранд-Информ»
Телефон: (831-30) 1-49-76
Реквизиты: р/с № 609238 в отделении Промстройбанка
г.Арзамас-16 Нижегородской области МФО 116886

ВАШ ДОМАШНИЙ КОМПЬЮТЕР

1992

УДК 681.3

Рецензент: С.В.Удалов - научный сотрудник (НИИ Экспериментальной Физики).

Ваш домашний компьютер. - 2-е изд., перераб. и доп. - Арзамас-16: «Гранд-Информ», 1992. - 86 стр.: ил.

Рассмотрено устройство, функционирование и практическое применение популярного домашнего компьютера **ZX Spectrum** (системы Sinclair) и его отечественных аналогов («Magic-05», «ALKOR-J», «Color», «Horus-48K», «Азбука», «Аса», «Байт», «Вега-64», «Дельта-С», «Дубна-48К», «Импульс», «Интер», «Искра-1085», «Кишинэу», «Квантор ZX1», «Квант-БК», «Кварц», «Корифей», «Компаньон», «Мастер», «Пифагор-128», «Престиж», «Радуга-001», «Сантака», «Сигма-С», «Синтез», «Сириус», «Спектр», «Хоббит», «Элин», «Эликон-020» и др.).

В доступной форме изложены вопросы применения и расширения возможностей ПЭВМ и программирования на ней. Даны практические советы и рекомендации по работе с ПЭВМ, программированию на языке BASIC и в машинных кодах. Приведено большое число примеров и упражнений, позволяющих получить необходимые навыки по самостоятельному программированию и обслуживанию ПЭВМ. Данная книга является руководством пользователя ПЭВМ ZX Spectrum.

Для широкого круга читателей - пользователей ПЭВМ ZX Spectrum. Книга может быть рекомендована для начального обучения основам информатики и программирования на ПЭВМ, совместимым с ZX Spectrum.

Содержание

Введение 5
История создания ПЭВМ «ZX Spectrum» 5
Некоторые соглашения 8
1.Знакомство с компьютером 9
1.1. Подключение компьютера 9
1.2. Клавиатура и дисплей 11
Клавиши CAPS SHIFT и SYMBOL SHIFT 12
Режимы «CAPS LOCK» и «GRAPHICS» 13
2. Основы программирования 14
2.1. Интерпретатор языка БЕЙСИК 14
2.2. Встроенный редактор программ 15
Что такое курсор редактирования? 16
Некоторые советы 17
2.3. Ввод русского шрифта 17
Использование графики пользователя 17
Изменение набора знаков 19
Генератор фонтов 20
Программный ввод знакогенератора 21
2.4. Защита Программ 23
Безостановочная работа программы 23
Отключение команды «LIST» 24
Нечитаемый текст программы 24
2.5. Операции с экраном 24
3. Элементы программирования в кодах 27
3.1. Использование программ в кодах 27
3.2. Программы на ассемблере Z80 28
Кто изобрел МИКРОПРОЦЕССОР? 30
Удаление фрагмента программы 31
4. Описание компьютера 32
4.1. Архитектура и принцип работы. 32
4.2. Операционная система 34
4.3. Карта памяти компьютера 35
4.4. Системные переменные 37
4.5. Синтезатор звуковых эффектов. 40
4.6. Встроенный таймер. 40
5. Программное обеспечение ПЭВМ 42

6. Компьютерные игры 43
6.1. Структура компьютерной игры 44
6.2. Как загрузить игру? 45
6.3. Как скопировать игру? 45
6.4. Как играть в игру? 45
6.5. Что делать, если все время убивают? 47
6.6. Проблема совместимости 48
7. Как загрузить программу? 49
8. Копирование программ 52
8.1. COPY 86/M 53
8.2. TF COPY 54
8.3. COPY-COPY 55
8.4. Дополнительные сведения 56

Приложение 1. Клавиатура компьютера. 58
Приложение 2. Таблица кодов. Мнемоника Ассемблера Z80 59
Приложение 3. Сообщения операционной системы 63
Приложение 4. Ключевые слова СПЕКТРУМа 66
Функции 66
Операторы 67
Приложение 5. Примеры программ на БЕЙСИКе 73
Приложение 6. Блок питания компьютера. Защита от помех 80
Приложение 7. Подключение телевизора 84
Приложение 8. Лучшие игры года 84
Приложение 9. Библиография 85
Литература 85
Периодические издания 85
Программное обеспечение 85

Введение

Добро пожаловать в мир домашнего компьютера «ZX Spectrum». В данной главе мы попробуем подготовить Вас к изучению этой замечательной «машины», завоёвывающей в нашей стране всё большую популярность. Что Вам для этого нужно? Во-первых, интерес, который, по-видимому, у Вас есть. Но, чтобы усилить его, мы кратко обрисуем некоторые привлекательные стороны данной ПЭВМ. Во-вторых, Вы нуждаетесь в руководстве пользователя для этой ПЭВМ — и этим руководством послужит Вам эта книга. Кроме того, Вам нужен сам компьютер. Это Вы должны обеспечить себе сами. Наконец Вам необходимо научиться программировать на нём, и мы дадим Вам несколько советов по этому поводу.

Чем же привлекателен Ваш домашний компьютер, совместимый с популярной английской ПЭВМ «ZX Spectrum»? Прежде всего очень хорошим соотношением производительность/стоимость, минимальными размерами и огромным банком системных и прикладных (игровых) программ (около 20000). Ваш компьютер — это незаменимое средство игротек и учебных классов, он прост в обращении, с ним могут работать как взрослые, так и дети. Ваш компьютер — это терпеливый домашний учитель, он поможет Вам освоить основы информатики и программирования.

По результатам социологического исследования журнала «Компьютер», наиболее массовым отечественным домашним компьютером является ПЭВМ типа «ZX Spectrum» («ZX Spectrum» — 40%, «БК 0010» — 12%, «Вектор 0,6Ц» — 12%, «РК-86» — 6%, «Atari» — 4%, «Commodore» — 2%, «Apple» — 2%, IBM-совместимые —2%, линии DEC — 1%).

Около 100 тысяч компьютеров системы «*Sinclair*» ежегодно выпускается государственными, совместными, малыми предприятиями и кооперативами (отечественные аналоги: «ALKOR-J», «Color», «Magic-05», «Horus DP/48K», «Азбука», «Аса», «Байт», «Вега-64», «Дельта-С», «Дубна-48К», «Импульс», «Интер», «Искра-1085», «Кишинэу», «Квантор ZX1», «Квант-БК», «Кварц», «Корифей», «Компаньон», «Композит», «Мастер», «Пифагор-128», «Престиж», «Радуга-001», «Сантака», «Сигма-С», «Синтез», «Сириус», «Спектр», «Хоббит», «Элин», "Эликон» и др.).

Несколько десятков тысяч установлены во дворцах молодежи и игротеках, в школах и других учебных заведениях. Более 500 тысяч компьютеров находится в личном пользовании.

История создания ПЭВМ «ZX Spectrum»

Мы начнем с того, что посвятим несколько слов истории создания «СПЕКТРУМа». Для тех наших читателей, кто недавно начал работу с этой замечательной машиной, будет, по-видимому, небезынтересно узнать о взлетах и падениях, пережитых ее создателем сэром Клайвом Синклером и, уж во всяком случае, здесь Вы найдете ответ на вопрос, почему «ZX-Spectrum» стал самым популярным домашним компьютером в мире и не сдает завоеванных позиций, хотя столько фирм выпускают гораздо более мощные машины.

Клайв Синклер родился 30 июля 1940 года. Еще школьником он начал печатать неплохие статьи в журнале «*Практическое Радио*». Окончив школу, он не стал поступать в университет, а был принят в этот журнал в качестве помощника заместителя редактора,

затем работал в издательстве, а в 1961 году зарегистрировал свою первую компанию «*Sinclair Radionics*». Первое изделие — микроусилитель, — он выпустил в 1963 году и во всех дальнейших разработках ставил перед собой две сверхзадачи — минимальные габариты и минимальная цена.

В 1972 году он изготовил карманный калькулятор, который назывался «Executive» и стоил «всего» 100 долларов. В 1977 году, опередив японцев, К.Синклер разработал и выпустил первый карманный телевизор — «Microvision», стоимостью 300 долларов. Не прошло и двух лет, и цену пользовавшегося спросом «телика» можно было понизить наполовину. Деньги, полученные от продажи своих изобретений, Синклер решил вложить в новую фирму — «*Sinclair Research Ltd*». Она была зарегистрирована в июне 1979 года. 10 месяцев спустя на рынке появился компьютер «ZX80». Благодаря весьма демократичной цене — всего 100 фунтов стерлингов — он стал пользоваться большим спросом, но не в Великобритании, а на континенте, куда направлялась большая часть продукции. Через полгода во Франции был продан стотысячный «ZX80».

Однако Синклер явно не хотел почивать на лаврах. Его ум, захваченный идеей массовой компьютеризации, продолжал работать. В марте 1981 года началось производство «ZX81» — первого подлинно домашнего компьютера. Благодаря цене (100 долларов) он был доступен всем желающим. Подключался к бытовому телевизору, накопителем служил обычный кассетный магнитофон, ОЗУ компьютера насчитывало всего 1 Кбайт.

За один год товарооборот фирмы вырос с 4,6 млн.ф.ст. до 30 млн.ф.ст., а Синклер уже готовил новую модель — «ZX Spectrum» (март 1982г.). Были разработаны две версии — 16K и 48K. Эта машина сильно отличалась от своих предшественников, и ее популярность превзошла все ожидания. Компьютеры «ZX Spectrum» продавались по 15 тыс.штук в неделю.

Задумывался этот компьютер как учебный для изучения программирования, но фирмы, выпускающие программное обеспечение, быстро поняли, что программирование на уровне команд процессора позволяет получить неплохую динамичную графику, и для этого компьютера стали выпускаться увлекательные видеоигры. Получилась своего рода положительная обратная связь. Чем больше «СПЕКТРУМов» покупалось населением, тем активнее выпускались для него программы, а чем больше на рынке высококачественных программ для компьютера, тем активнее он покупается.

Такой же процесс охватил и фирмы «третьего рынка», выпускающие периферийные устройства и аксессуары для компьютеров. К 1984 году, когда фирмы «*Atari*», «*Commodor*» и «*Amstrad*» выпустили компьютеры, превосходящие «ZX Spectrum 48», рынок уже был смещен в пользу Синклера, что продолжает чувствоваться и по сей день, а сам «ZX Spectrum» уже выпускался более чем в 30 странах мира.

За свои разработки и славу, которую он принес английской короне, Синклер был удостоен дворянского титула. С 1981 года перед его фамилией стали появляться три маленькие буквы — «сэр».

Ум гения не укладывался в привычные понятия. И в 1984 году К.Синклер разработал и выпустил на рынок совершенно новый компьютер — «QL», который до сих пор, по мнению многих специалистов, остается лучшим домашним компьютером для профессиональных применений. В начале 1985 года фирма выпустила новую модель «ZX Spectrum Plus», отличавшуюся усовершенствованной клавиатурой, а в конце 1985 года «ZX Spectrum 128» («ДЕРБИ»), имеющий 128K оперативной памяти и 32K ПЗУ. Кроме этого, новая модель имела звуковой процессор.

Однако эра домашних компьютеров уже подходила к концу. Беспрестанно посматривавшая на Америку Европа дождалась, наконец, разработки, моментально ставшей стандартом. Началась эра «*IBM*».

Хотя Клайв Синклер и продолжал выпускать новые версии Спектрумов, но на самом деле он был занят совершенно другой идеей —разработкой дешёвой автомашины. В 1985 году фирма «*Sinclair Research Ltd*» стала её выпускать под названием «*C5*». Но на этот раз удача покинула гения. Деньги, потраченные на разработку «*C5*», не принесли ожидаемого дохода. Через год права на все продукты «*Sinclair Research Ltd*»пришлось продать фирме «*Amstrad*». Это были «похороны» Спектрума. Алан Шугер — владелец «Amstrad» — выпустил только одну модель — «Спектрум 128» со встроенным 3-х дюймовым дисководом.

Сам Синклер продолжает свои разработки в новой фирме — «*Cambridge Computers*». В 1987 году появился «Z88» — портативный IBM-совместимый компьютер, в прошлом laptop «Psion MC400», в котором управление курсором осуществляется не мышью, а передвижением пальца по специальной сенсорной табличке. Продолжается разработка новой MS-DOS-совместимой машины с операционной системой, «прошитой» в ПЗУ.

В то же время Синклер не забывает о «грехах» своей молодости. В 1989 году он выпустил на рынок недорогой приемник спутникового телевидения. Достойна уважения и его настойчивость, поскольку, несмотря на неудачу, он продолжает разрабатывать в «*Sinclair Research Ltd*» новую модель недорогой городской электромашины «C15». Но на этот раз сэр Клайв Синклер решил подстраховаться и параллельно с «C15» готовит к выпуску «сверхпортативную», складную, почти карманную модель велосипеда. Как далеко «заедет» на нем в будущее Клайв Синклер — покажет время.

Если ретроспективно смотреть на историю развития компьютеров серии ZX, то можно выделить следующие этапы:

Июнь 1979 — основание фирмы «Sinclair Research Ltd»

Февраль 1980 — на рынке появился «ZX80»

Март 1981 —начало продажи« ZX81»

Июнь 1982 — появление «ZX Spectrum» с ОЗУ 48K

Апрель 1984 — продажа первых экземпляров «QL»

Январь 1985 — начало производства «ZX Spectrum Plus» и «Spectrum 128»

Апрель 1986 — право на выпуск всех компьютеров «Sinclair Research Ltd» продано фирме «Amstrad»

Август 1986 — начало производства «Spectrum +2»

Февраль 1987 — «Amstrad» начинает выпуск «Spectrum +3» со встроенным 3-х дюймовым дисководом

Март 1987 — новое поколение синклеровских компьютеров открывает «Z88»

Октябрь 1989 — начало производства «Psion MC400» (лучший laptop 1990 года в анкете журнала «CHIP»)

Август 1990 — фирма «*Amstrad*» принимает решение о прекращении производства «Spectrum +3»

Некоторые соглашения

Теперь мы уже почти готовы начать последовательное описание ПЭВМ ZX Spectrum и встроенного интерпретатора языка БЕЙСИК. Нам осталось только упомянуть о некоторых соглашениях, которых мы будем придерживаться.

Устройство ввода-вывода

Вообще говоря, существует много способов ведения диалога человека с ПЭВМ, но мы будем предполагать, что Вы вводите команды при помощи клавиатуры и читаете ответ на экране дисплея.

Изображение функциональных клавиш

Обычно Вы посылаете команду компьютера, нажимая на клавишу с надписью «ENTER» (Ввод). Название клавиши обозначаются прописными буквами:

Пример: <ENTER>.

Здесь угловые скобки означают, что Вы должны нажать на единственную клавишу, а не набирать всё слово по буквам.

Кроме того мы будем упоминать управляющие клавиши, называя их <CAPS SHIFT>, <SYMBOL SHIFT>, <SPACE BREAK> или сокращённо: <CS>, <SS>, <BREAK> или <SPACE> соответственно.

Такое обозначение как <CS>+<9> указывает на то, что необходимо нажать на клавишу <9>, держа одновременно нажатой клавишу <CAPS SHIFT>.

Изображение курсоров

Для обозначения курсора используются специальные полиграфические кавычки, а изображение самого курсора выполняется полужирным шрифтом.

Пример: «K»

Изображение текста программ

Для представления текста программ мы применяем специальный шрифт (Гельветика), похожий на тот, который Вы можете видеть на экране дисплея (телевизора) или на бумаге при выводе на печать.

```
PRINT AT 11,10;"Добро пожаловать"
```

СОВЕТ:

Вы должны изучать Ваш компьютер и программировать на нём активно, а не просто пассивно читать данную книгу. С этой целью мы включили в неё много примеров. Вы должны попытаться решить хотя бы некоторые из них на Вашем компьютере, чтобы получить лучшее представление о том, как они работают. Не бойтесь модифицировать программы, чтобы увидеть к чему это приведет. Если Вы будете активны, Вы изучите Ваш компьютер и язык *БЕЙСИК* быстро и узнаете его достаточно глубоко.

1.Знакомство с компьютером

Компьютер построен на высокоскоростном 8-разрядном процессоре «Z80» (Z80A) фирмы «ZILOG» с тактовой частотой 3,5 МГц, что позволило получить производительность на уровне 850000 операций в секунду. Это приблизительно в 4 раза лучше, чем у компьютеров «РАДИО РК-86» и «Микроша», близко к производительности компьютера «ДВК-3» и даже немного выше, чем у «ДВК-2».

В качестве монитора можно использовать обычный цветной или черно-белый телевизор, на экране которого можно получать сложные графические изображения, состоящие из 256 x192 точек при 16 цветах, а также текстовые изображения в форматах 32x24, 42x24, 51x24 и 64x24 символа. При этом операционная система использует самый крупный формат текста — 32x24, чтобы не утомлять Вас созерцанием мелких буквочек и облегчить этим работу. Переключать режимы работы контроллера дисплея (с графического на символьный или с одного символьного на другой), как, например, в компьютере «YAMAHA MSX», нет необходимости — все режимы можно использовать одновременно.

При использовании черно-белого телевизора 8 цветов представляются на нем как 16 градаций яркости.

Имеется возможность подключения накопителя на гибких магнитных дисках, светового пера, манипулятора «мышь» (все модели с разъемом «INTERFACE 2») и графического принтера (только модели с системным разъемом).

1.1. Подключение компьютера

Взглянем на Ваш компьютер(см рис. 1.1):

На задней стенке корпуса мы увидим четыре гнезда типа СГ-*5*:

- магнитофон
- питание
- видео (телевизор)
- джойстик

Соедините компьютер с внешними устройствами в соответствии с документацией к нему и еще раз внимательно посмотрите, не перепутали ли Вы местами разъемы.

Если все в порядке — включайте !

При включении компьютера немедленно активизируется тестовая система, проверяющая исправность компьютера.

Через 0,5—1,5 секунды после включения тестовая система завершит свою работу и на экране появится сообщение:

Рис.1.1.

Компьютер будет полностью готов к действию - операционная система будет активизирована и готова к исполнению команд (в том числе команд встроенного БЕЙСИКа). Пуск машины в работу не сложнее, чем включение обычного калькулятора!

Примечание: В последнее время кроме фирменной версии операционной системы СПЕКТРУМА 1982 года, получают распространение более поздние версии, например, 1990 года. При включении компьютера появляется сообщение:
© 1990 Sinclair Research Ltd

Эта операционная система сохраняет все функции старой версии (1982). Введены очень полезные дополнения и изменения:

- возможность работы с двойной плотностью записи на магнитную ленту, что повышает ёмкость кассеты и снижает время ввода программы в память компьютера;
- введены дополнительные функции для БЕЙСИКа;
- изменен знакогенератор (вместо квадратных и фигурных скобок, и некоторых других, редко используемых символов, вставлены буквы немецкого алфавита);
- возможность работы со встроенной программой МОНИТОР;
- изменена процедура сброса. При нажатии кнопки <СБРОС> происходит сброс процессора без очистки памяти, то есть информация в ОЗУ сохраняется. Для выполнения полного сброса необходимо одновременно нажать клавишу <CAPS SHIFT> и кнопку <СБРОС>.

Ниже приведены дополнительные команды БЕЙСИКа и МОНИТОРа (1990).

БЕЙСИК

<(> — при загрузке программы выдается информация о длине, стартовый адрес и т.п.;

<)> — снятие предыдущего режима;

<-> — переход на двойную плотность;

<+> —обычная плотность;

<> — индикация свободной области БЕЙСИК-системы;

<*> — переход в МОНИТОР.

МОНИТОР

<1> — выбор системы счисления;

<2> — переключатель символы/коды;

<3> — изменение ячеек памяти;

<5>-<8> — перемещение по памяти;

<9>,<0> —выбор регистровой пары;

<SS> + <8> — выход в БЕЙСИК без очистки ОЗУ и экрана;

<SS> + <0> — выход в БЕЙСИК без очистки ОЗУ с очисткой экрана;

<SS> + <7> — появляется GO SUB, вводишь адрес запуска и запускаешь программу;

<SS> + <9> — ?

<BREAK> — выбор ячейки памяти

На этом можете оставить осторожность и смело выполнять все примеры, приведенные в данном пособии. Не бойтесь отдать компьютеру не ту команду - этим его не испугаешь! Если все же в результате Ваших манипуляций компьютер «зависнет», не слушая клавиш, нажмите кнопку <СБРОС>, и все придет в нормальное состояние. Теперь смелее вперед!

1.2. Клавиатура и дисплей

Клавиатура Вашего компьютера выполнена по американскому стандарту «QWERTY» и насчитывает 40 клавиш, каждая из которых может выполнять по 6 и более функций.

Все клавиши клавиатуры снабжены автоповтором, и если вы будете удерживать клавишу нажатой дольше 1 *с*, Вы увидете, что компьютер начнет как бы повторять нажатия с частотой около 6 раз в секунду.

Аналогично, все клавиши, кроме переключателей, снабжены звуковым подтверждением нажатия - при их нажатии вы услышите щелчок.

Функции, выполняемые клавишами, зависят от типа курсора (мигающего прямоугольника с буквой внутри) на дисплее и от состояния клавиш переключателей (<CAPS SHIFT> и <SYMBOL SHIFT>).

На дисплее могут быть следующие курсоры:

K — курсор основных команд BASIC и ОС;

E — курсор дополнительных команд BASIC;

L — курсор маленьких и больших букв;

C — курсор больших букв;

G — курсор псевдографики;

? — курсор контроллера ошибок

После включения компьютера Вы не увидите никакого курсора, а только надпись внизу экрана :

© 1982 Sinclair Research Ltd.

Нажмите клавишу <ENTER>, и на экране в нижнем левом углу появится мигающий прямоугольник с буквой «K» внутри. Это и есть курсор основных команд. Если вы сейчас нажмете какую-либо буквенную клавишу, то обнаружите, что курсор на экране сдвинулся вправо, в нем вместо буквы «K» появилась буква «L», а на его прежнем месте появилась команда БЕЙСИКа или директива ОС. Например, если Вы нажали <P>, то на экране внизу вы увидите оператор БЕЙСИКа **PRINT**.

Следующие нажатия клавиш будут истолкованы компьютером как параметры для этой команды. Например, нажмите клавиши <1> и <2>. После этого Вы увидите на экране уже **PRINT 12**. Это совершенно правильная команда, требующая от компьютера напечатать на экране число «12». Как видите, интеллектуальный контроллер ошибок действует, не давая Вам возможности посылать компьютеру бессмысленные команды. Нажмите <ENTER> (ввод), и компьютер выполнит Вашу команду **PRINT 12**, напечатав на экране число «12».

Такой режим выполнения команд называется *непосредственным*, т.е. Вы непосредственно отдаете системе команду за командой, подтверждая их нажатием клавиши

<ENTER>. После нажатия <ENTER>, команда сразу исполняется. Можно задать для непосредственного исполнения несколько команд, отделяя их друг от друга символом «:». При этом все команды находятся в командной строке.

Что такое командная строка и где она располагается? Для понимания этого рассмотрим формат экрана, используемый операционной системой (см. рис. 1.2)

Интерпретатору БЕЙСИКа доступны (без использования специальных команд) только строки 0— 21, т.е. основной экран. Последние две строки экрана и есть командная строка. Она принадлежит операционной системе, хотя и используется иногда БЕЙСИКом, например для ввода и редактирования программы. Если команды или сообщения не помещаются в командной строке (т.е. более 64 символов), ОС увеличивает размер командной строки и может даже занять ею весь экран, потеснив на время БЕЙСИК.

Существует и другой режим выполнения команд — программный режим.

Аналогично вышеописанному, после включения машины нажмите <ENTER> для появления курсора «K». Но, в отличие от предыдущего примера, нажмите не буквенные, а цифровые клавиши, например, 1 и 0. Вы увидите, что курсор не изменился, а только сдвинулся, и слева от него появилось число «10». Теперь нажмите клавишу P, и сразу появится наш старый знакомый оператор **PRINT**.

Вот теперь уже курсор изменился, в нем появилась буква «L». Как и раньше нажмите клавиши <1> и <2>. В командной строке вы увидите следующую комбинацию:

```
10 PRINT 12
```

Это значит, что команда **PRINT 12** посылается нами в 10-ую строку программы. Нажмите <ENTER>. К нашему удивлению, на экране появилось не число «12», а следующая комбинация:

```
10 > PRINT 12
```

Одновременно командная строка очистилась. Теперь наш **PRINT** стал уже не просто командой, а программой, и чтобы его выполнить, надо подать команду **RUN**. Подайте ее, нажав на клавишу <R> и <ENTER> , и вы обнаружите наше старое доброе число «12» как результат выполнения программы. Как видите, и в этом режиме интеллектуальный контроллер ошибок не позволил Вам допустить неточности при вводе программы.

Рис.1.2.

Клавиши CAPS SHIFT и SYMBOL SHIFT

На клавиатуре Вашего компьютера есть еще две клавиши, при нажатии на которые на экране ничего не появляется:

<CAPS SHIFT> — большие буквы

<SYMBOL SHIFT> — специальные символы (. , + - и т.д.)

При нажатии и удержании этих клавиш (в курсоре «L») и одновременном нажатии алфавитно-цифровых клавиш на экране появятся соответственно

большие буквы или специальные символы.

При одновременном нажатии обеих клавиш происходит переключение курсора «K» в курсор «E», т.е. в курсор дополнительных команд БЕЙСИКа. Сброс этого курсора происходит автоматически при вводе дополнительной команды или при повторном нажатии обеих клавиш.

Нажатие и удержание одной из этих клавиш в курсоре «E» позволяет получить модификации дополнительных команд.

Режимы «CAPS LOCK» и «GRAPHICS»

Режим «CAPS LOCK» позволяет выводить большие буквы, не нажимая клавиши <CAPS SHIFT>. Для его включения нажмите одновременно клавиши <CAPS SHIFT> и <2>, при этом курсор «L» на экране заменится курсором «C». Для выключения этого режима еще раз нажмите клавиши <CAPS SHIFT> и <2>.

Режим «GRAPHICS» позволяет выводить псевдографику и символы, определяемые пользователем. Включается он клавишами <CAPS SHIFT> и <9>, при этом курсор «L» на экране заменяется курсором «G».

Как видите, любую команду можно ввести с клавиатуры нажатием одной-двух клавиш, причем написание команды всегда будет безусловно правильным. Переключение регистров происходит в основном автоматически, наиболее употребительные команды вводятся одной клавишей. Во вводимый текст программы интеллектуальный контроллер автоматически вставляет пробелы для улучшения читаемости программы (обратите внимание, вы вводили в нашем примере 10PRINT12 (слитно), а на экране увидели **10 PRINT 12**).

Рисунок (расположение и назначение клавиш) клавиатуры ПЭВМ «ZX Spectrum» приведен в Приложении 1.

2. Основы программирования

2.1. Интерпретатор языка БЕЙСИК

ПЭВМ ZX Spectrum использует язык высокого уровня, который называется БЕЙСИК (BASIC). Слово "BASIC" является аббревиатурой английского названия:

Beginners All-purpose Symbolic Instruction Code,

что в переводе означает "многоцелевой язык символических команд для начинающих". Этот язык был разработан в 1964 году в США, в Дармутском колледже в Нью-Гемпшире. Он очень широко используется в персональных компьютерах, но тем не менее каждый компьютер имеет свою версию БЕЙСИКа.

БЕЙСИК - язык интерпретирующий. Работа на БЕЙСИКе организуется с помощью специальной программы-интерпретатора, которая записана в ПЗУ или загружается в ОЗУ с внешнего накопителя - магнитной ленты или диска. Эта программа переводит каждый оператор бейсик-программы на машинно-ориентированный язык ПЭВМ.

БЕЙСИК ZX Spectrum не сильно отличается от (несуществующего) эталонного БЕЙСИКа, поэтому Вы не будете испытывать больших сложностей, если захотите использовать любую программу, написанную на БЕЙСИКе (например для ДВК или IBM PC) для Вашего Спектрума. В отличии от других версий БЕЙСИКа, Спектрум-БЕЙСИК не позволяет опускать оператор "LET" при вводе числовых значений переменных.

Встроенный в операционную систему Вашего компьютера интерпретатор языка БЕЙСИК ("прошит" в ПЗУ) распознает и исполняет около 177 различных команд, позволяющих:

- производить вычисления по программе результатов в форме с плавающей запятой с точностью 8 цифр числа +2 цифры порядка +знак. При этом результат автоматически представляется в удобном для восприятия виде, лишние нули отсекаются. Минимальное значение положительного числа 10 в степени -39 или 0.00000000000000000000000000000000000001, все числа, меньше данного, считаются машинным нулем. Максимальное значение положительного числа 10 в степени 39 минус единица и обрезанное до 8 значащих цифр - 99999990000000000000000000000000000000, числа, больше данного, не могут быть обработаны БЕЙСИКом, о чем выдается соответствующее сообщение **"Number too big"**;
- получить доступ к операционной системе, записывать, читать, проверять, компоновать файлы различного формата, как при использовании магнитофона, так и при подключении дисковода;
- получить доступ к машинным ресурсам - ячейкам памяти, портам ввода-вывода, системным переменным, программам в машинных кодах;
- управлять выводом информации на экран, изменять цветовую палитру и режимы вывода;

- получать на экране сложные многоцветные графические изображения и копировать их на графический принтер (ZX PRINTER).

По своей мощности этот интерпретатор БЕЙСИКа значительно превосходит интерпретаторы БЕЙСИКа машин ДВК-1, РК-86 и Микроша, удобнее и мощнее интерпретатора YAMAHA MSX и приближается по возможностям к GW BASIC компьютера IBM PC, отличаясь от него большим удобством работы, экономным использованием памяти (по этому параметру ему вообще нет равных) и некоторыми другими особенностями, страхующими невнимательного пользователя от неправильной работы программы.

Встроенный в интерпретатор интеллектуальный синтаксический контроллер значительно упрощает написание программы, просто не давая Вам сделать синтаксических и лексических ошибок и тут же указывая курсором на спорное, по его мнению место. При выполнении программы он тщательно контролирует соответствие данных и результатов, правильность формальной логики работы и при обнаружении ошибки выводит подробное сообщение о ней и возможной причине ее возникновения, с указанием точного места. Большинство ошибок не являются фатальными и при исправлении позволяют продолжить выполнение программы с места, в котором наступила ошибка (Приложение 3).

Подробнее с возможностями и работой интерпретатора БЕЙСИК Вы можете ознакомиться в специальном руководстве "Программирование на языке БЕЙСИК" Стивена Викерса (официальная инструкция по работе со "Спектрумом", широко распространенная на "рынках" страны).

2.2. Встроенный редактор программ

Ваш компьютер имеет встроенный в операционную систему редактор программ, специализированный на редактировании БЕЙСИК-программ. Это редактор строчного типа (с элементами экранного редактора), поскольку он редактирует программу по строкам.

Для нормальной работы редактора каждой строке программы должен быть присвоен индивидуальный номер. Это не значит, что номер нужно присваивать каждой строке, видимой на экране, ведь строка программы на БЕЙСИКе может содержать до 255 команд и символов текста (команды встроенного БЕЙСИКа для компьютера ничуть не больше по размерам, чем одна буква текста).

Допустимо использовать целые номера от 1 до 9999 с любым интервалом между ними. Обычно используют интервал 10.

При редактировании используются специальные клавиши - клавиши перемещения курсора и некоторые другие:

стрелка вверх (<CAPS> + <6>) - программный курсор на строку вверх

стрелка вниз (<CAPS> + <7>) - программный курсор на строку вниз

стрелка влево (<CAPS> + <5>) - курсор редактирования на символ влево

стрелка вправо (<CAPS> + <8>)- курсор редактирования на символ вправо

EDIT (<CAPS> + <1>) - вызвать строку, на которую указывает программный курсор на редактирование

DELETE (<CAPS> + <0>) - стереть символ слева от курсора редактирования

Все эти операции выполняются сразу при нажатии на клавишу(и).

Что такое курсор редактирования?

Вам хорошо известны курсоры «K», «L», «C», «E», «G». А вот чтобы понять, что такое программный курсор (курсор редактирования), Вам придется набрать на клавиатуре наш старый пример "**10 PRINT 12**" и нажать <ENTER>.

На экране Вы увидите:

```
10 > PRINT 12
```

Вот эта галочка ">" и есть программный курсор. Он указывает на ту строку программы, которая будет редактироваться.

Введите теперь комбинации 20 PRINT "------" (и не забудьте нажать клавишу <ENTER>) и 30 PRINT "+++++".

Теперь на экране Вы увидите:

```
10  PRINT 12
20  PRINT "——"
30 > PRINT "++++++"
```

Введите теперь директиву "EDIT" (<CAPS>+<1>). При этом внизу экрана, в командной строке, появится копия нашей 30-й строки программы, курсор редактирования в ней будет крайним справа (т.е. в конце строки).

Нажимая клавиши <стрелка влево> и <стрелка вправо>, можно подвести курсор редактирования к нужному месту. Проделайте это сами, обратив внимание на то, что при движении по строке курсор как бы "раздвигает" символы, вклиниваясь между ними. Но при этом, заметьте, что внутрь команды "PRINT" курсор не заходит - это опять дело рук интеллектуального контроллера ошибок, который не позволяет Вам исказить написание команды.

Добейтесь такого положения курсора:

```
10  PRINT 12
20  PRINT "——"
30 > PRINT "++++++"
```

```
30  PRINT "+[L]+++++"
```

здесь [L] - это курсор <<L>>

Введите директиву **DELETE** (<CAPS>+<0>). При этом Вы увидите, что знак "+", находившийся слева от курсора, будет стерт, и строка сдвинется, уменьшив свою длину. По этой директиве курсор выполняет роль стирательной резинки, стирая по одному символу слева от себя.

После этого нажмите клавиши <3>, <4>, <5>, и Вы увидите, как соответствующие цифры появятся на месте курсора, раздвигая строку. Такой режим редактирования называется редактированием с автораздвижкой.

Наконец, нажмите <ENTER> и посмотрите, что получилось:

```
10  PRINT 12
20  PRINT "——"
30 > PRINT "345+++++"
```

Если теперь подать команду **RUN** (клавиша <R>), то на экране появятся следующие результаты:

```
12
——
345++++
```

Полюбовавшись на творение своих рук, нажмите <ENTER>, чтобы вернуть на экран текст программы.

После этого клавишами <стрелка вверх> и <стрелка вниз> подвигайте программный курсор с одной строки на другую. Можете также понажимать клавиши <CAPS>+<1> (EDIT), вызывая тем самым редактирование то одной, то другой строки программы.

Некоторые советы:

Если Вам надо создать несколько одинаковых (или похожих) строк программы, не спешите их набирать с клавиатуры. Гораздо проще набрать одну строку, вызвать ее на редактирование, подправить, если нужно, и, стерев старый номер строки, подставить новый и нажать <ENTER>. При этом старая строка остается без изменения и появится новая строка с новым номером. Повторив это действие несколько раз, Вы получите нужное количество строк.

Аналогично производится перестановка или перенумерация строк программы. При этом старые строки стирают (см. ниже).

Если Вы хотите стереть строку программы целиком, не надо вызывать ее на редактирование - достаточно набрать ее номер и нажать <ENTER>.

Если Вы, редактируя строку, окончательно ее испортили, не волнуйтесь - пока Вы не нажали <ENTER>, все еще поправимо. Нажмите клавиши <CAPS>+<1> (EDIT), и строка станет как новенькая.

Если Ваша программа стала настолько большой, что Вам неудобно редактировать ее, поскольку слишком долго двигать программный курсор из одной части программы в другую, воспользуйтесь директивой **LIST**.

Например, если Вы хотите передвинуть программный курсор в строку 1200, наберите "LIST 1200", нажмите <ENTER>, а затем <BREAK> - и дело сделано.

2.3. Ввод русского шрифта

Обеспечение возможности работы компьютера с символами русского алфавита является, наверное, одной из первейших задач, которые возникают перед теми, кто хочет сам писать программы на "СПЕКТРУМе".

Существуют два принципиально различных подхода для решения этой задачи. Первый подход состоит в использовании символов графики пользователя, а второй - в создании нового набора символов, размещении его в оперативной памяти и назначении его в качестве действующего с помощью системной переменной "**CHARS**".

Примечание:	У некоторых серийно выпускаемых отечественных компьютеров имеется встроенный русский шрифт.

Использование графики пользователя

В графическом режиме Вы можете, во-первых, набирать графические символы блочной графики, расположенные на цифровых клавишах, а также можете воспользоваться символами графики пользователя. Их может быть до 21; они берутся нажатием на клавиши от <A>

до <U> в графическом режиме (курсор «G»). Изображения символов графики пользователя должны быть сформированы в специальной области памяти, на которую указывает системная переменная "UDG", расположенная в адресах **23675** и **23676**. Адрес, в котором хранится изображение символа "A" (адрес первого байта, с которого начинается графика пользователя "UDG"), может быть получен с помощью функции **USR**.

Например, если Вы дадите команду:

```
PRINT USR "A"
```

то получите число **65368**. Поскольку "A" - первый символ графики "UDG", то это число и есть адрес первого символа. Если Вы дадите команды:

```
PRINT PEEK 23675 : PRINT PEEK 23676
```

то получите 2 числа: **88** и **255** - это тоже информация об адресе первого символа "UDG". Просто таким образом в своей памяти компьютер хранит двухбайтные числа. Умножьте **255** на **256** и прибавьте **88**. Вы получите тот же адрес - **65368**.

Примечание: Если в системную переменную **UDG** поместить оператором **POKE** другие числа, то компьютер будет искать конструкцию символов **UDG** в других областях памяти. Таким образом, Вы можете иметь не 21 символ графики, а сколько пожелаете, необходимо только "сообщить" об этом Вашем желании компьютеру и "расплатиться" по 8 байтов за каждый новый символ.

Таким образом, заслав, начиная с адреса USR "A", 8 байт, формирующих изображение, скажем буквы "Я", мы будем иметь возможность нажатием клавиши <A> в графическом режиме печатать на экране букву "Я".

Учитывая, что многие буквы русского и латинского алфавита имеют сходное начертание (например : **O, A, X, H, P** и др.), можно задать прописные буквы, не имеющие аналогов и при этом уложится в 21 символ.

Изображение буквы "Я" формируется из точек экрана (пикселей) например так :

```
00000000      0
00XXXXX0 0*128+0*64+1*32+1*16+1*8+1*4+1*2+0*1 = 62
0X0000X0 0*128+1*64+0*32+0*16+0*8+0*4+1*2+0*1 = 66
0X0000X0 0*128+1*64+0*32+0*16+0*8+0*4+1*2+0*1 = 66
00XXXXX0 0*128+0*64+1*32+1*16+1*8+1*4+1*2+0*1 = 62
00X000X0 0*128+0*64+1*32+0*16+0*8+0*4+1*2+0*1 = 34
0X0000X0 0*128+1*64+0*32+0*16+0*8+0*4+1*2+0*1 = 66
00000000      0
```

Таким образом, в результате такого синтеза шаблона буквы "Я", мы можем представить ее последовательностью из 8 байт, разместив их в соответствующем месте

```
0, 62, 66, 66, 62, 34, 66, 0
```

Точно так же, можно изобразить и остальные желаемые символы. Для тех, кто не хочет глубоко вдаваться в суть процесса, мы предлагаем небольшую готовую программу, которую Вы можете набрать и выгрузить на ленту. Всякий раз, когда Вы хотите составить какую-либо программу, в которой необходимы русские буквы, загрузите эту программу и стартуйте ее (**RUN**). После остановки программы дайте **LIST** и продолжайте со строки **100** набирать свою программу.

```
10 FOR Z=1 TO 20
20 READ A$
30 FOR X=0 TO 7
40 READ Y
50 POKE USR A$ + X,Y
60 NEXT X
70 NEXT Z
80 DATA "B",0,126,64,124,66,66,124,0
81 DATA "G",0,126,64,64,64,64,64,0
82 DATA "D",0,28,36,36,36,36,126,66
83 DATA "J",0,65,73,62,73,73,65,0
84 DATA "I",0,66,70,74,82,98,66,0
85 DATA "L",0,30,34,34,34,34,98,0
86 DATA "P",0,126,66,66,66,66,66,0
87 DATA "O",0,66,66,36,24,16,96,0
88 DATA "C",0,68,68,68,68,68,126,2
89 DATA "H",0,66,66,66,126,2,2,0
90 DATA "N",0,65,73,73,73,73,127,0
91 DATA "M",0,65,73,73,73,73,127,1
92 DATA "E",0,60,66,30,2,66,60,0
93 DATA "U",0,76,82,114,82,82,76,0
94 DATA "A",0,62,66,66,62,34,66,0
95 DATA "S",24,66,70,74,82,98,66,0
96 DATA "R",0,64,64,124,66,66,124,0
97 DATA "T",0,192,64,124,66,66,124,0
98 DATA "F",0,127,73,73,73,127,8,0
99 DATA "Q",0,66,66,114,74,74,114,0
```

В результате работы этой программы устанавливается следующее соответствие между клавишами в графическом режиме и символами русского алфавита:

Клавиша	Q	E	R	T	U	I	O	P	A	S	D	F	G	H	J	L	C	B	N	M
Русская буква	Ы	Э	Ь	Ъ	Ю	И	У	П	Я	Й	Д	Ф	Г	Ч	Ж	Л	Ц	Б	Ш	Щ

Изменение набора знаков

Когда мы проводим русификацию полной заменой знакогенератора, то мы уже вторгаемся в поле действия международного стандарта **ASCII** и для начала маленькая историческая справка.

Набор символов компьютера содержит 256 символов, начиная с нулевого по 255-ый. Стандарт **ASCII** (*American Standart Codes for Information Interchange*) определяет коды с нулевого по 31-й как управляющие и с 32-го по 127-й как буквы английского алфавита, цифры и знаки препинания. Символы со 128-го по 255-й этот стандарт не определяет, разрешая размещать в них графические символы или национальные шрифты.

Здесь бы и разместить русский шрифт, как это делается, например в **IBM**-совместимых и других компьютерах, но для "**Спектрума**" из этого ничего не выйдет. Дело в том, что К.Синклер воспользовался предоставленной ему здесь свободой и уже разместил, начиная со 128-го по 143-й, символы блочной графики, со 144-го по 164-й отдал Вам в виде символов "**UDG**", а со 165-го по 255-й использовал для определения символов ключевых слов, что дает Вам возможность набирать такие слова как "*LOAD*" одним нажатием клавиши (код - 239). Полный набор символов приведен в Приложении 2.

Вывод один, необходимо производить переключение шрифтов с английского на русский и наоборот. Неважно каким способом это сделано - программно или аппаратно введением дополнительного регистра, но если и русский и английский шрифты занимают те же символы с 32-го по 127-й, то замену лучше всего делать так, чтобы используемый Вами принтер тоже мог легко переключаться одной командой.

Наиболее распространенными для работы с 8-разрядными компьютерами в стране являются принтеры **ROBOTRON** (ГДР) и **D-100** (Польша). И в том и в другом реализован стандарт **КОИ-7**.

Код	Англ	Рус
64	@	ю
65	A	а
66	B	б
67	C	ц
68	D	д
69	E	е
70	F	ф
71	G	г
72	H	х
73	I	и
74	J	й
75	K	к
76	L	л
77	M	м
78	N	н
79	O	о
80	P	п
81	Q	я
82	R	р
83	S	с
84	T	т
85	U	у
86	V	ж
87	W	в
88	X	ь
89	Y	ы
90	Z	з
91	[	ш
92	\	э
93	]	щ
94	↑	ч
95	_	ъ

Код	Англ	Рус
96	фунт	Ю
97	a	А
98	b	Б
99	c	Ц
100	d	Д
101	e	Е
102	f	Ф
103	g	Г
104	h	Х
105	i	И
106	j	Й
107	k	К
108	l	Л
109	m	М
110	n	Н
111	o	О
112	p	П
113	q	Я
114	r	Р
115	s	С
116	t	Т
117	u	У
118	v	Ж
119	w	В
120	x	Ь
121	y	Ы
122	z	З
123	{	Ш
124	|	Э
125	}	Щ
126	~	Ч
127	©	

Тогда соответствие кодов клавиш должно быть таким (мы начинаем с 64-го кода, поскольку цифры и знаки препинания совпадают), как указано в таблице.

(Код 94: "↑" -стрелка вверх)

Набор знаков компьютеров **"СПЕКТРУМ"** состоит из **96** символов (буквы английского алфавита, цифры и знаки препинания) и занимает **96*8=768** байт. Стандартный набор находится в ПЗУ по адресу **15616**. Этот адрес можно определить с помощью системной переменной **"CHARS"**, расположенной в адресах **23606** и **23607**.

```
PRINT PEEK 23606 + 256 *
PEEK 23607 + 256 <ENTER>
15616 O.K.
```

При этом по адресу **23606** находится **0**, а по адресу **23607 - 60**.

Изменение набора знаков состоит в том, чтобы сформировать с помощью какой-либо специализированной программы или вручную желаемый набор знаков, поместить его в любое свободное место оперативной памяти и изменить содержимое ячеек **23606** и **23607**, чтобы оно указывало на адрес места расположения нового набора. Рассмотрим эти процедуры на конкретном примере.

Генератор фонтов ("ART STUDIO")

Для формирования набора знаков очень удобно использовать графический редактор *"ART STUDIO."* Эта программа не только является идеальным графическим редактором, позволяющим легко выполнять графические изображения, но и позволяет конструировать шрифты. Выгрузим сформированный с помощью программы *"ART STUDIO"* шрифт на ленту.

Допустим, мы хотим разместить новый шрифт, начиная с адреса **30000**. Загрузим его с ленты

```
LOAD "" CODE 30000,768
```

Системная переменная CHARS должна указывать на адрес, находящийся на 256 байт ниже, т.е. на 29744 (30000-256=29744). Определим старший байт двухбайтной переменной **CHARS**.

```
INT (29744/256) = 116
```

Определим младший байт :

```
29744 - 116*256 = 48
```

Переключение на печать сформированным Вами шрифтом выполняется командой

```
POKE 23606,48: POKE 23607,116
```

Обратное переключение на латинский шрифт - командой

```
POKE 23606,0: POKE 23607,60
```

Программный ввод знакогенератора

Тем у кого нет графического редактора "*ARTSTUDIO*", мы предлагаем Вам программу ввода русского знакогенератора с помощью операторов **READ** и **DATA** (по материалам ж-ла "**Компьютер**" N2, 1990).

Коды знаков русских букв соответствуют стандарту **КОИ-7**. Выбранная форма знаков обеспечивает хорошую разборчивость текста и низкую утомляемость даже при работе на дисплеях, параметры которых оставляют желать лучшего.

Коды знаков занимают 768 байт и могут располагаться в свободной части оперативной памяти. Если "**RAMTOP**" установлен ниже начального адреса массива кодов алфавита, то команда **NEW**, стирающая BASIC-программы, не затронет массива кодов. "**RAMTOP**" устанавливается с помощью команды **CLEAR xxx**, где xxx - равно начальному адресу массива кодов минус единица.

В программу входят загрузчик (строки 5-60), преобразующий шестнадцатиричные коды в строках 9903-9998 в десятичные, и массив данных. Каждая строка массива состоит из данных одного знака и команды-комментария (**REM**), за которой следует буква латинского алфавита, превращающаяся после запуска программы в букву русского алфавита. По начертанию этой буквы можно судить о правильности набора строки.

Наберите программу и запустите ее командой **RUN**. С помощью **LIST** проверьте начертания букв. Если обнаружите ошибки, исправьте их и снова запустите программу. Сохранить массив кодов на магнитную ленту Вы можете с помощью команды

```
SAVE "R.ALF" CODE 64000,768
```

Считывание с ленты осуществляется командой **LOAD "R.ALF" CODE** либо **LOAD "R.ALF" CODE n**, где n - новый адрес, который должен быть кратен числу 256 (на самом деле не обязательно кратным 256. Здесь данное ограничение позволяет переключать шрифт с помощью одной команды **POKE**). Переключение с русского на латинский осуществляется командой **POKE 23606,60**, а с латинского на русский - **POKE 23607,adr**, где adr - адрес, разделенный на 256, минус 1.

Примечание: Пробелы в строках **DATA** даны для удобочитаемости. Вам их при наборе вводить не надо.

```
1 REM R.ALF
3 CLEAR 63999
5 LET A=10: LET B=11: LET C=12
10 LET D=13: LET E=14: LET F=15
12 LET x=64000: POKE 23607,249
15 READ a$
20 FOR n=1 TO 15 STEP 2
30 LET w=VAL a$(n)
40 LET w=w*16: LET w=w+VAL a$(n+1)
50 POKE x,w: LET x=x+1
60 NEXT n: GO TO 15
9903 DATA "00 00 00 00 00 00 00 00": REM " "
9904 DATA "18 18 18 18 18 00 18 00": REM !
9905 DATA "6C 6C 00 00 00 00 00 00": REM "
9906 DATA "6C 6C FE 6C FE 6C 6C 00": REM #
9907 DATA "18 7E E0 7C 0E FC 38 00": REM $
9908 DATA "00 C6 CC 18 30 66 C6 00": REM %
9909 DATA "38 6C 38 77 DE CE 7B 00": REM &
9910 DATA "18 18 30 00 00 00 00 00": REM '
9911 DATA "00 0C 18 18 18 18 0C 00": REM (
9912 DATA "00 30 18 18 18 18 30 00": REM )
9913 DATA "00 66 3C FF 3C 66 00 00": REM *
9914 DATA "00 18 18 7E 18 18 00 00": REM +
9915 DATA "00 00 00 00 18 18 30 00": REM ,
9916 DATA "00 00 00 7E 00 00 00 00": REM -
9917 DATA "00 00 00 00 00 18 18 00": REM .
9918 DATA "03 06 0C 18 30 60 C0 00": REM /
9919 DATA "7C C6 CE D6 E6 C6 7C 00": REM 0
9920 DATA "18 38 18 18 18 18 3C 00": REM 1
9921 DATA "3C 66 06 1C 30 66 7E 00": REM 2
9922 DATA "3C 66 06 1C 06 66 3C 00": REM 3
9923 DATA "0E 1E 36 66 7F 06 0F 00": REM 4
9924 DATA "7E 60 7C 06 06 66 3C 00": REM 5
9925 DATA "1C 30 60 7C 66 66 3C 00": REM 6
9926 DATA "7E 66 06 0C 18 18 18 00": REM 7
9927 DATA "3C 66 66 3C 66 66 3C 00": REM 8
9928 DATA "3C 66 66 3E 06 0C 38 00": REM 9
9929 DATA "00 18 18 00 18 18 00 00": REM :
9930 DATA "00 18 18 00 18 18 30 00": REM ;
9931 DATA "0C 18 30 60 30 18 0C 00": REM <
9932 DATA "00 00 7E 00 00 7E 00 00": REM =
9933 DATA "60 30 18 0C 18 30 60 00": REM >
9934 DATA "3C 66 06 0C 18 00 18 00": REM ?
9935 DATA "00 00 CE DB FB DB CE 00": REM @
9936 DATA "00 00 3C 06 3E 66 3F 00": REM A
9937 DATA "04 3C 60 7C 66 66 3C 00": REM B
9938 DATA "00 00 66 66 66 66 7F 03": REM C
9939 DATA "00 00 7E 66 66 66 FF C3": REM D
9940 DATA "00 00 3C 66 7E 60 3E 00": REM E
9941 DATA "00 18 7E DB DB DB 7E 18": REM F
9942 DATA "00 00 7C 60 60 60 60 00": REM G
9943 DATA "00 00 C6 6C 38 6C C6 00": REM H
9944 DATA "00 00 66 66 6E 76 66 00": REM I
9945 DATA "18 00 66 66 6E 76 66 00": REM J
9946 DATA "00 00 66 6c 78 66 66 00": REM K
9947 DATA "00 00 0E 1E 36 66 C6 00": REM L
9948 DATA "00 00 C6 EE D6 C6 C6 00": REM M
9949 DATA "00 00 66 66 7E 66 66 00": REM N
9950 DATA "00 00 3C 66 66 66 3C 00": REM O
9951 DATA "00 00 7E 66 66 66 66 00": REM P
9952 DATA "00 00 3E 66 3E 36 66 00": REM Q
9953 DATA "00 00 7C 66 7C 60 60 00": REM R
9954 DATA "00 00 3C 66 60 66 3C 00": REM S
9955 DATA "00 00 7E 18 18 18 18 00": REM T
9956 DATA "00 00 66 66 66 3E 06 3C": REM U
9957 DATA "00 00 DB 7E 18 7E DB 00": REM V
9958 DATA "00 00 7C 66 7C 63 7E 00": REM W
9959 DATA "00 00 60 60 7C 66 7C 00": REM X
9960 DATA "00 00 C6 C6 F6 DA F6 00": REM Y
9961 DATA "00 00 3C 66 0C 66 3C 00": REM Z
9962 DATA "00 00 D6 D6 D6 D6 FE 00": REM [
9963 DATA "00 00 7C 06 3E 06 7C 00": REM /
9964 DATA "00 00 D6 D6 D6 D6 FF 03": REM ]
9965 DATA "00 00 66 66 3E 06 06 00": REM ^
9966 DATA "00 00 00 00 00 00 00 FF": REM _
9967 DATA "CE DB DB FB DB DB CE 00": REM '
9968 DATA "0E 1E 36 66 FE C6 C6 00": REM a
9969 DATA "FC C0 C0 FC C6 C6 FC 00": REM b
9970 DATA "CC CC CC CC CC CC FC 06": REM c
9971 DATA "7E 66 66 66 66 66 FF C3": REM d
9972 DATA "FC C0 C0 F8 C0 C0 FE 00": REM e
9973 DATA "18 7E DB DB DB 7E 18 00": REM f
9974 DATA "7C 60 60 60 60 60 00": REM g
9975 DATA "C3 66 3C 18 3C 66 C3 00": REM h
9976 DATA "C6 C6 CE DE F6 E6 C6 00": REM i
```

```
9977 DATA "D6 C6 CE DE F6 E6 C6 00": REM j
9978 DATA "C6 CC D8 F8 CC C6 C6 00": REM k
9979 DATA "0E 1E 36 66 C6 C6 C6 00": REM l
9980 DATA "C6 EE FE D6 C6 C6 C6 00": REM m
9981 DATA "C6 C6 C6 FE C6 C6 C6 00": REM n
9982 DATA "7C C6 C6 C6 C6 C6 7C 00": REM o
9983 DATA "FE C6 C6 C6 C6 C6 C6 00": REM p
9984 DATA "7E C6 C6 7E 36 66 C6 00": REM q
9985 DATA "FC C6 C6 FC C0 C0 C0 00": REM r
9986 DATA "7C C6 C0 C0 C0 C6 7C 00": REM s
9987 DATA "FC 30 30 30 30 30 30 00": REM t
9988 DATA "C6 C6 C6 7E 06 C6 7C 00": REM u
9989 DATA "DB DB 7E 18 7E DB DB 00": REM v
9990 DATA "F8 CC CC FC C6 C6 FC 00": REM w
9991 DATA "C0 C0 C0 FC C6 C6 FC 00": REM x
9992 DATA "C6 C6 C6 F6 DA DA F6 00": REM y
9993 DATA "3C 66 06 1C 06 66 3C 00": REM z
9994 DATA "D6 D6 D6 D6 D6 D6 FE 00": REM {
9995 DATA "7C C6 06 1E 06 C6 7C 00": REM |
9996 DATA "D6 D6 D6 D6 D6 D6 FF 03": REM }
9997 DATA "66 66 66 3E 06 06 06 00": REM ~
9998 DATA "3C 42 99 A1 A1 99 42 3C": REM ©
9999 REM START 64000, END 64768
```

2.4. Защита Программ

Защита программ состоит в исключении возможности просмотра текста программы и внесения в него изменений лицом, не уполномоченным на это автором программы.

Методов защиты программ существует очень много и здесь мы рассмотрим только основные принципы и обозначим подходы к этой проблеме.

Можно выделить три основных направления:

- Исключить возможность остановки (прерывания работы программы)
- Исключить возможность подачи команды **LIST** для распечатывания текста программы
- Сделать листинг программы нечитаемым

Безостановочная работа программы

Самое первое, что Вы должны сделать - это выполнить программу автостартующей со строки номер N. Это выполняется при выгрузке программы командой

```
SAVE "имя" LINE N
```

Теперь программа после загрузки будет сама стартовать с указанной строки. Тем не менее, она может быть остановлена командой **BREAK** или по сообщению об ошибке.

Отключить клавишу <BREAK> можно, если в той строке, с которой происходит автостарт, поместить команду

```
POKE 23613, PEEK 23730 - 5
```

Если программа Вами хорошо отлажена, то остановку по ошибке можно исключить. Надо помнить, что чаще всего при взломе программы сознательно вносят ошибку во время исполнения оператора **INPUT**. Так, если программа просит от пользователя ввести какое-либо число, он сознательно нажимает на клавишу с буквой. Программа останавливается с сообщением:

```
VARIABLE NOT FOUND
```

Рекомендуем не использовать оператор **INPUT**, а организовывать опрос клавиатуры на основе функции **INKEY$**.

Если в строке автостарта поместить нижеприведенные команды, то при попытке сделать **BREAK** или при появлении ошибки, программа будет сбрасываться:

```
LET ERR=256*PEEK 23614 + PEEK 23613: POKE ERR,0: POKE ERR+1,0
```

Примерно тот же эффект дает помещение в стартовой строке команды:

```
POKE 23659,0
```

Одним из способов остановки автостартующей программы является использование команды **MERGE ""** вместо **LOAD ""** при загрузке.

Если Ваша программа имеет размер больше 7К, то защитить ее от этого приема можно, если перед выгрузкой подать следующие прямые команды:

```
LET X=256*PEEK 23636 + PEEK 23635: POKE X,60: POKE X+1,0
```

Другой способ защиты от **MERGE** состоит в том, чтобы перед выгрузкой Вашей программы на ленту, дать ложную информацию о том, что якобы первая строка Вашей программы имеет чрезмерно большую длину Это делается прямой командой :

```
POKE (PEEK 23635 + 256 * PEEK 23636 + 3),255
```

Отключение команды «LIST»

Введите первой строкой программы 1 **REM**. Перед выгрузкой программы на ленту дайте прямую команду:

```
POKE (PEEK 23635 + 256 * PEEK 23636),100
```

Теперь программу можно запускать (**RUN**), но читать нельзя.

Нечитаемый текст программы

Простейший прием, применяемый во многих программах состоит в том, что устанавливают одинаковым цвета символов (**INK**) и фона (**PAPER**).

```
Например: 10 INK 7: PAPER 7
```

В тех же местах, где программа должна сделать вывод на экран, в операторе **PRINT** вставляют в качестве временных правильные цвета:

```
20 PRINT INK 0; "ZX-Spectrum"
```

Другой способ состоит в искажении набора символов путем задания "фальшивого" значения системной переменной **CHARS**. Попробуйте, например :

```
POKE 23606,8: PRINT "ZX-Spectrum": POKE 23606,0
```

Если адресовать системную переменную **CHARS** в те области памяти, где вообще ничего нет, например:

```
POKE 23607,200
```

то все символы будут выглядеть как пробелы, и на экране вообще ничего не будет. Вам же нужно перед всяким оператором **PRINT** включать истинное значение **CHARS** и выключать его после **PRINT**.

Еще один регулярно используемый прием состоит в организации "нулевой" строки, которая реально исполняется программой, но не может быть удалена при редактировании. Чтобы первая строка стала нулевой, дайте команду

```
POKE 23755,0: POKE 23756,0
```

2.5. Операции с экраном

Типичной задачей для тех, кто начинает создавать собственные программы на "СПЕКТРУМе", является создание красочных изображений в графике высокого разрешения, хранение их в памяти компьютера и вывод на экран.

Мы не будем рассматривать графические операции с операторами **DRAW, CIRCLE, PLOT**, а также печать символов графики пользователя, а рассмотрим операции с экраном, созданным в графическом редакторе. Наиболее удобном в работе является графический редактор *ART STUDIO*, но можно использовать и какой-либо другой доступный.

Созданное в редакторе изображение Вы можете выгрузить на ленту в виде блока кодов, длина которого 6912 байт. Если теперь загрузить его командой

```
LOAD "" CODE 16384, 6912
```

то изображение сразу будет загружено на экран, но хранить его там, естественно, нельзя. Можно загрузить изображение в произвольное место и вызывать его оттуда на экран по мере необходимости. Примем адрес для хранения картинки 30000.

```
LOAD "" CODE 30000, 6912
```

Переброску на экран можно организовать в БЕЙСИКе.

```
500 FOR i=1 TO 6912
510 POKE 16383+i, PEEK 2999+i
520 NEXT i
530 RETURN
```

Всякий раз, когда Вам надо вызвать изображение на экран, Вы можете сделать это командой **GO SUB 500**.

К сожалению, этот метод очень медленно работает, что связано с ограниченными возможностями БЕЙСИКа. Построение экрана происходит более 20 сек.

Возможность быстрой переброски экрана (менее 1 сек.) предоставляет программирование в машинном коде. Процессор **Z-80** для этой цели имеет мощную команду перемещения блоков LDIR (ее код 237,176). Для ее работы необходимо, чтобы в регистре HL процессора находился адрес начала блока, подлежащего переброске (в нашем случае 30000), в регистре BC - длина блока (6912), а в регистре DE - адрес места назначения (16384). Тогда программа в машинном коде будет выглядеть так :

Машинный код	Ассемблер Z80	Комментарий
33 48 117	LD HL,48 117	Загрузить в регистр HL адрес 30000. обратите внимание на то, что 48+117*256=30000
1 0 27	LD BC,0 27	Загрузить в регистр BC число 6912
17 0 64	LD DE,0 64	Загрузить в регистр DE 16384
237 176	LDIR	Выполнить переброску блока в новое место
201	RET	Возврат в вызывающую программу

Таким образом, программа в машинном коде выглядит так:

```
33, 48, 117, 1, 0, 27, 17, 0, 64, 237, 176, 201
```

Поместим ее в произвольное место оперативной памяти, например начиная с адреса 65000 следующей программой, написанной на языке БЕЙСИК.

```
10 FOR i=1 TO 12
20 READ A
30 POKE 64999+i,A
40 NEXT i
50 DATA 33,48,117,1,0,27,17,0,64,237,176,201
```

Теперь всякий раз, когда Вам нужно поместить картинку на экран, Вам надо стартовать эту процедуру, что выполняется командой

```
RANDOMIZE USR 65000
```

Если же у Вас в памяти компьютера хранится не одно, а несколько графических изображений, то их можно также перебрасывать на экран этой процедурой, надо только изменить адрес, поступающий в регистр процессора HL, что можно сделать командой **POKE** по адресу 65001 и по адресу 65002.

Предположим, что у Вас есть вторая картинка, помещенная начиная с адреса 37000. Тогда INT (37000 / 256) = 144, а 37000 - 256 * 144 = 136. Выполним

```
POKE 65001,136: POKE 65002,144.
```

Команда **RANDOMIZE USR 65000** выполнит переброску второй картинки и т.д.

Очевидно, что поскольку каждый экран занимает 6912 байтов, то одновременно в памяти нельзя хранить более 6 изображений, что во многих случаях бывает недостаточно. Если учесть, что каждое изображение имеет длительные последовательности нулей, то имеется возможность компрессировать каждое изображение. Это позволяет во многих случаях вдвое сократить объем памяти, занимаемый графикой, а иногда и больше.

Тексты программ для компрессии/декомпрессии изображения с экрана и размещения его в заданном пользователем адресе приведены, например, в журнале **"Компьютер"** № 3, 1990 или **"ZX+ЕЩЁ"** № 1, 1991 или **ZX-РЕВЮ** № 1, 2 (1991).

3. Элементы программирования в кодах

3.1. Использование программ в кодах

Операционная система Вашего компьютера (как, впрочем, и БЕЙСИК) позволяет загружать, выгружать и запускать на выполнение программы в машинных кодах процессора Z80. При загрузке и выгрузке такие программы представлены файлами с типом BYTE, являющимися полной аналогией файлов типа .COM компьютера IBM PC. Эти программы полностью резидентны (т.е. при работе целиком находятся в ОЗУ компьютера), что повышает скорость их работы практически до скорости работы процессора (200-300 тысяч операций в секунду). Это самые быстрые программы из всех возможных.

Откуда же берутся такие программы?

Путей их возникновения несколько: результат компиляции программ на языках высокого уровня (*Паскаль, Си, Фортран, БЕЙСИК, специальные языки*); результат ассемблирования программ на языке ассемблера Z80; непосредственное написание программ в кодах с помощью специальных отладчиков или просто вручную, с помощью карандаша и бумаги.

Эти программы запускаются на исполнение командой **RANDOMIZE USR adr**, где adr — адрес (десятичный), с которого программа запускается. Такая же команда выполняется и в БЕЙСИКе.

Кроме того, для запуска таких программ в БЕЙСИКе имеется функция **USR**, возвращающая содержимое регистровой пары BC процессора Z80 в виде целого положительного числа от 0 до 65535.

Все программы такого типа должны заканчиваться кодом команды процессора RET (возврат) 201 (шестнадцатиричный C9). **Примеры:**

Таблица 3.1

RANDOMIZE USR 0	Запуск программы в кодах с адреса 0000, т.е. в ПЗУ, такая команда полностью аналогична нажатию кнопки «сброс».
RANDOMIZE USR 55000	Запуск программы в кодах с адреса 55000.
LET a=USR 55000	Запуск программы в кодах с адреса 55000 и присвоение переменной «a» значения, оказавшегося в регистровой паре BC процессора в момент возврата.
PRINT USR 55000	Запуск программы в кодах и печать значения регистровой пары BC процессора.

К сожалению, программы в кодах не имеют атрибута самозапуска, из-за чего они не могут сами запуститься после загрузки, если не применять для этого особых программных ухищрений. Поэтому для их загрузки и запуска применяют маленькие программки-загрузчики на БЕЙСИКе.

Более развернутый пример использования таких программ приведен в главе «Компьютерные игры».

Внимание: использовать программы в кодах надо с большой осторожностью, так как при их выполнении не производится контроль ошибок и может произойти зависание компьютера, устранимое только нажатием кнопки «СБРОС».

3.2. Программы на ассемблере Z80

Написание программ в кодах — занятие весьма кропотливое, ведь Вам приходиться помнить не только коды команд, но и их длину (1, 2, 3, 4 байта), внимательно следить за адресами, в которых размещается программа и данные. При возникновении необходимости доработать, улучшить, модифицировать программу — возникают значительные трудности. Кроме того, работать эта программа будет только в тех адресах, для которых была написана.

Использование ассемблера позволяет получить столь же эффективные и быстрые программы, как и при написании их в кодах, но с гораздо меньшими затратами сил и времени. Одновременно обеспечивается возможность работы программы в любых адресах (при перетрансляции) и легкость ее модификации.

Принцип программирования на ассемблере состоит в том, что вместо кодов машинных команд Вы пишите их условные обозначения — мнемоники, а вместо адресов - именные метки. Программа-ассемблер превращает мнемоники в коды команд, а метки — в конкретные адреса этих команд. Такая операция называется ассемблированием, и в результате ее получается программа в кодах, используемая так же, как и обычные программы в кодах.

Таблица 3.2

К580ВМ80	Z80	ДЕЙСТВИЕ
MOV B,C	LD B,C	B = C
LXI B,#4F	LD B,#4F	B = 4F (HEX)
LDAX B	LD A,BC	A = содержимое ячейки памяти BC
LHLD #4FBF	LD HL, #4FBF	HL = содержимое ячейки памяти #4F BF
SPHL	LD SP,HL	SP = HL
PUSH PSW	PUSH AF	Спасение аккум. и флагов
PUSH B	PUSH BC	Спасение рег. пары BC
ANI #4F	AND #4F	A=A & 4F (HEX)
ANA B	AND B	A=A & B
RPO	RET PO	Возврат, если не плюс
RNC	RET NC	Возврат, если не перенос
RET	RET	Возврат безусловный
XCHG	EX DE,HL	Обмен рег. пар DE и HL
XTHL	EX (SP),HL	Обмен рег. пар HL и стека

Ассемблеры очень сильно зависят от типа процессора машины. В нашем случае необходим ассемблер для процессора Z80. Если Вы уже программировали на ассемблере

для К580ВМ80 (*Intel 8080*), мнемоники Z80 могут показаться Вам странными и непривычными.

Система команд Z80 включает полную систему команд Intel 8080A. Хуже обстоит дело с физической совместимостью — цоколевки процессоров абсолютно различны. Вот, например, как выглядят одни и те же команды на этих ассемблерах (см. табл. 3.2).

Простое сравнение показывает, что мнемоники Z80 при меньшем их количестве более понятны и наглядны. Однако, если Вы привыкли программировать на ассемблере К580ВМ80, можете на нем и продолжать программировать, так как программы в кодах К580ВМ80 всегда будут работать на процессоре Z80 (но не наоборот!).

Внимание! Программы, написанные для К580ВМ80, всегда будут выполняться на Z80, разумеется при условии, что внешние устройства, к которым обращается (если обращается) программа одинаковы и подключены одинаковым образом.

Таблица 3.3

Регистры процессора Z80

НАБОР		НАЗВАНИЕ И НАЗНАЧЕНИЕ РЕГИСТРА
ОСНОВН.	АЛЬТЕРН.	
A	A	Аккумулятор (8 бит, A со вспомогат. 16 бит)
B	B	РОН B
C	C	РОН C
D	D	РОН D
E	E	РОН E
H	H	РОН H
L	L	РОН L
F	F	Регистр флагов (8 бит)
IX	IX	Регистр сегмента X (16 бит)
IY	IY	Регистр сегмента Y (16 бит)
IR	IR	Регистр прерываний и регенерации (16 бит)
SP	SP	Указатель стека (16 бит)
PC	PC	Указатель команды (16 бит)

Правда при этом Вы не используете всех возможностей Вашего процессора Z80 (например, не используете всех его регистров).

Итого имеется 21 программно-доступный регистр и еще несколько недоступных, которые процессор использует для хранения промежуточных результатов.

Как Вы, наверное, заметили, большинство регистров процессора или имеет размер 16 бит, или объединены в пары по 16 бит. Какова же разрядность Вашего процессора? 8 или 16 бит?

Ответить на этот вопрос очень сложно. Во всяком случае, у процессора имеется и набор команд 8-битовой арифметики, и набор команд 16-битовой арифметики. Наличие в составе процессора двойного АЛУ (арифметико-логического устройства) и 16-разрядной схемы сумматора-вычитателя, а также 16-разрядного инкрементора-декрементора

позволяет ему оперировать 16-битовыми данными столь же легко, как и 8-битовыми. Таким образом, можно сказать, что «внутри» Ваш процессор 16-разрядный, но «снаружи» выглядит как 8-разрядный (ведь внешняя шина данных имеет 8 разрядов).

Именно эта особенность Вашего процессора и обусловила наличие у него огромного количества команд (около 680 разновидностей) и регистров (21).

Исторická справка

Кто изобрел МИКРОПРОЦЕССОР?

Принято считать, что микропроцессор был изобретен и впервые реализован в начале семидесятых годов американской фирмой *Intel*. В 1973 году фирмой был реализован кристалл, ставший прототипом микропроцессоров для *IBM PC*-совместимых компьютеров (от 8086 до 486), на производство которых Intel имела патент.

Однако, американское бюро по регистрации изобретений и патентов объявило о своем решении: выдать патент № 4.942.516 «*Single Chip Integrated Circuit Architecture*» истинному отцу микропроцессора Gilbert Hyatt. Hyatt - инженер по электронике, работал в Калифорнии в фирме *Micro Computer Inc*, ещё в 1968 году сделал «computer on a chip». В 1970 году он заявил свою разработку к регистрации.

Микропроцессор Zilog Z80 разрабатывался как усовершенствованная модель по отношению к процессору Intel 8080A теми же разработчиками, уволившимися из фирмы *Intel* и образовавшими фирму *Zilog*.

Но этого создателям процессора Z80 показалось мало! В дополнение к 16- и 8-битовым операциям они ввели еще и поразрядные (1-битовые) операции, что на порядок упростило и, приблизительно, в 2 раза ускорило работу с внешними устройствами (например, с дисководом). Это позволяет в целом ряде случаев обойтись без соответствующих схем контроллеров внешних устройств. Все эти возможности имеет только процессор Z80, и реализует их только соответствующий ассемблер!

Одним из наиболее мощных ассемблеров является программа макроассемблер GENS-4/51 *из пакета* DEVPAC-4 *(кроме макроассемблера* GENS-4/51 *в пакет входит отладчик-дизассемблер* MONS-4). Приставка «макро» перед названием ассемблер означает, что кроме нормальных команд микропроцессора Z80 Вы можете сами создавать и использовать макрокоманды. Например, если Вам захочется иметь команду умножения MUL (ее нет в системе команд Z80), ее можно заменить программным эквивалентом и использовать так же свободно, как если бы она всегда была в системе команд.

Более подробно с возможностями и работой макроассемблера GENS-4 можно познакомиться в специальном описании «Ассемблер GENS-4. Отладчик MONS-4» [2].

Кроме ассемблера, превращающего программу на языке Ассемблер в программу в кодах процессора, имеется и программа дизассемблер, делающая обратное превращение. Две эти программы взаимно дополняют друг друга, поэтому их обычно включают в один пакет, а для облегчения отладки придают им отладчик программ в кодах. По такому принципу, например, построены все пакеты DEVPAC фирмы «*Hisoft*», в том числе и последний - DEVPAC-4.

Этот пакет состоит из двух программ в кодах, полностью перемещаемых и работающих в любых адресах. Загружают эти программы командой LOAD "" CODE adr , где adr — адрес, начиная с которого будет находиться программа. Затем программы запускают командой RANDOMIZE USR adr, где adr —тот же адрес, что и в команде LOAD.

А теперь небольшой, но полезный пример программы на Ассемблере Z80.

Удаление фрагмента программы

Для удаления любого числа записанных подряд строк программы на языке Бейсик необходимо делать следующее. К модифицируемой основной программе присоединить приведенную ниже программу и запустить ее командой **GO TO 9990**. Затем, выполняя команду **INPUT**, поочередно ввести в компьютер:

1) номер первой строки той части программы, которую нужно удалить (X);

2) номер строки, следующей за удаляемым фрагментом (Y).

После ввода этих чисел будет выполнена программа на внутреннем языке и появится сообщение **OK**. Теперь нужно удалить строку с номером X; вместе с ней будут автоматически удалены все строки с номерами от X до Y.

```
9990 INPUT "Delete from: ";x;" to: ";y
9991 LET xa=INT(x/256)
9992 LET xb=x-xa*256
9993 LET ya=INT(y/256)
9994 LET yb=y-ya*256
9995 RESTORE 9999
9996  FOR a=65400 TO 65452
9997  READ b: POKE a,b: NEXT a
9998 RANDOMIZE USR 65400
9999 DATA 1,xb,xa,42,83,92,205,163,
255,34,176,92,1,yb,ya,43,205,163,255,
68,77,42,176,92,237,161,237,161,121,
149,79,120,156,119,43,113,201,35,94,
35,86,25,35,126,184,35,32,245,126,
185,32,241,201
```

На Ассемблере эта программа будет выглядеть следующим образом:

```
10      ORG 65400
20      LD BC,0
30      LD HL,(23635)
40      CALL E1
50      LD (23728),HL
60      LD BC,0
70      DEC HL
80      CALL E1
90      LD B,H
100     LD C,L
110     LD HL,(23728)
120     CPI
130     CPI
140     LD A,C
150     SUB L
160     LD C,A
170     LD A,B
180     SBC A,H
190     LD (HL),A
200     DEC HL
210     LD (HL),C
220     RET
230 E2 INC HL
240     LD E,(HL)
250     INC HL
260     LD D,(HL)
270     ADD HL,DE
280     INC HL
290 E1 LD A,(HL)
300     CP B
310     INC HL
320     JR NZ,E2
330     LD A,(HL)
340     CP C
350     JR NZ,E2
360     RET
```

4. Описание компьютера

4.1. Архитектура и принцип работы

Основные блоки персонального компьютера «ZX Spectrum» показаны на Рис.4.1.

Центральный процессор (ЦП), содержащий устройство управления, выполняет арифметические и логические операции. Он выполнен на микропроцессоре Z-80A, либо его других фирменных аналогах, например UA 880D. Тактовая частота — 3,5 МГц.

В постоянном запоминающем устройстве (ПЗУ) содержатся набор команд или программ и постоянная информация, которая не изменяется в процессе выполнения программ. ПЗУ как правило выполнено на МС типа 2764 (отечественный аналог — К573РФ4) либо на МС большей емкости, например 27128, 27256, 27512.

Оперативное запоминающее устройство (ОЗУ) позволяет записывать и считывать информацию и используется для временного хранения данных. Узел динамического ОЗУ объемом 48К байт включает в себя собственно микросхемы памяти типа 565РУ5, мультиплексоры типа 555КП11 и буферный регистр типа 555ИР22. Здесь емкость микросхем 565РУ5 используется не полностью, так как старшие 16К байт адресного пространства процессора отведены постоянному запоминающему устройству. Фирменные компьютеры и некоторые самодельные имеют разделенные ОЗУ пользователя (32К) и служебное ОЗУ (16К).

Входные и выходные порты обеспечивают обмен информацией между ПК и внешней средой (клавиатура, НМЛ, НГМД, принтер и т.д.), а также формируют звуковой сигнал и цвета бордюра.

Генератор тактовых импульсов вырабатывает сигналы для синхронизации операций внутри ПК.

Все устройства в системе взаимодействуют друг с другом посредством системной шины, состоящей из адресной шины, шины данных и шины управления (см. рис. 4.1)

С точки зрения программиста компьютер имеет следующие логические устройства:

- все доступные регистры микропроцессора Z80;
- постоянная память ПЗУ (ROM), которая размещается в первых 16К адресного пространства и заключена между адресами 0000H и 3FFFH (0-16383);
- оперативная память ОЗУ (RAM), заключенная между адресами 4000H и FFFFH (16384-65535). Общий объём её составляет 48К;
- внешние устройства.

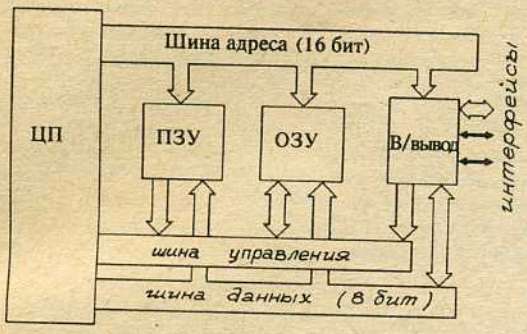


Рис.4.1.

Перейдём к описанию адресного пространства компьютера.

Постоянная память.

В постоянной памяти компьютера хранится резидентное программное обеспечение (ПО), которое поддерживает через порты ввода/вывода связь с внешними объектами, обеспечивает вывод на экран символьной и графической информации. В качестве системного программного обеспечения может быть использован БЕЙСИК для компьютера ZX SPECTRUM со встроенными функциями МОНИТОРА. Встроенные в БЕЙСИК функции позволяют осуществлять вывод на экран и ввод с клавиатуры, поддерживать связь с магнитофоном и другими внешними устройствами.

Пользователь по своему усмотрению может изменять содержимое и состав резидентного ПО. При разработке своего ПО необходимо помнить, что в случае, если полученное ПО занимает объём меньший чем 16К, то освободившееся адресное пространство не может быть использовано под ОЗУ в стандартном варианте.

Оперативная память.

Под оперативную память в стандартном варианте архитектуры компьютера отведена область 4000....FFFFH. В этом адресном пространстве можно выделить:

- область пикселей;
- область атрибутов;
- область пользователя.

Область пикселей и область атрибутов составляют область видео-ОЗУ (VIDEO-RAM).

Область пикселей.

Эта область, в которой собственно хранится изображение символов и графики. Используемый контроллер дисплея не имеет встроенного знакогенератора, поэтому в видео-ОЗУ символы хранятся в виде непосредственных изображений. Единичное значение соответствующего бита в байте вызывает засветку точки на экране, а нулевое значение, наоборот, гасит точку.

Область атрибутов.

Эта область, которая содержит специальные байты управления выводом пикселей на экран. В байте атрибута определяется: цвет точек, цвет фона, режим увеличения яркости, режим мигания. Особенностью формирования цветного изображения в данном компьютере является то, что один байт атрибута описывает область 8x8 точек экрана, т.е. по горизонтали 1 байт, по вертикали 8 строк. Это составляет одно знакоместо.

Цвет фона и цвет пикселя кодируются тремя знаками. Таким образом, на одном из восьми цветов фона можно выбрать один из восьми цветов пикселя, причем на одном цвете фона можно вывести один цвет точек в пределах блока 8x8 пикселей. Это минимальное поле, описываемое одним атрибутом, что создает некоторые трудности при программировании мелких деталей изображения. Но, как показала практика, умелое использование атрибутов позволяет свести к минимуму этот эффект. Достоинством такого способа формирования изображения является небольшой объем VIDEO-RAM, что существенно влияет на повышение быстродействия экрана дисплея. Такая архитектура дисплея позволяет использовать компьютер для отображения быстро изменяющихся процессов (например в качестве различных графических и знаковых индикаторов в системах отображения информации). Биты BRIGHT и FLASH управляют увеличением яркости и заданием режима мигания соответственно.

Область пользователя.

Область, не занятая VIDEO-RAM, остаётся в распоряжении пользователя. В этой области помещены также системные переменные. Эта область используется для загрузки программ с магнитной ленты или из другого внешнего устройства микросети. Адреса области пользователя 5B00...FFFF (23296-65535).

4.2. Операционная система

Операционная система (ОС) — это программная надстройка над аппаратурой (компьютером), облегчающая работу с ним. Операционная система Вашего компьютера относится к типу «твердотельных ОС», поскольку находится и работает в ПЗУ (постоянном запоминающем устройстве). Это защищает ее от случайного повреждения Вашей программой, повышая надежность работы системы, а также позволяет компьютеру быть готовым к работе сразу после его включения. Такой тип ОС появился сравнительно недавно, в 1980-х годах, но уже успел значительно потеснить «загружаемые ОС», поскольку работает быстрее и надежнее их, и к тому же не требует для работы системы наличия каких-либо внешних устройств типа дисководов и т.п.

ОС СПЕКТРУМА с находящимся внутри нее интерпретатором БЕЙСИКа занимает младшие 16 килобайт адресного пространства (из полных 64 Кбайта) Вашего компьютера. Это и есть 16К ПЗУ. Вся остальная память машины —48 К ОЗУ —доступна Вам для Вашего использования (небольшая — примерно 0,2 кбайта — часть ОЗУ занята системными переменными операционной системы, ею надо пользоваться очень осторожно). Рассмотрим структуру памяти наглядно:

ПЗУ ОС и БЕЙСИКа	ОЗУ экрана	Системные переменные	ОЗУ пользователя, программы, данные	Стеки и UDG
0 16384		23552	23734	65300 65535

- 16 Кбайт занимает ПЗУ.
- 6,5 килобайт занимает экран (это обычное ОЗУ, но его содержимое контроллер дисплея интерпретирует как графическое изображение и показывает на экране).

- 200 байт занимают системные переменные (подробнее о них см. раздел «Системные переменные»).
- 235 байт занимают стеки и UDG. UDG — это таблица знакогенератора пользователя, в которой Вы можете закодировать любые нужные Вам знаки (например, русские или греческие буквы, символы шахматных фигур и т.д.). Всего их может быть до 21, они вводятся нажатием на клавиши от A до U в графическом режиме (курсор «G»).
- Все остальное — ОЗУ пользователя. Его Вы можете использовать, как Вам будет угодно.

Основная функция ОС — операции над файлами, например, загрузка их в память, запуск, выгрузка, сравнение, компоновка.

Всего имеется 5 типов файлов:

- **BYTE** —программа в машинных кодах
- **PROGRAMM** — программа на БЕЙСИКе
- **SCREEN$** — копия экрана
- **DATA** — значения числовых массивов
- **DATA$** —значения символьных переменных или массивов

Кроме того, некоторые программы создают свои типы файлов. Однако эти типы файлов не являются стандартными и не обрабатываются операционной системой.

4.3. Карта памяти компьютера

Для возможности работы операционной системы компьютера в его памяти выделено несколько областей. Эти области отделены друг от друга некоторыми границами, которые могут иметь постоянный (фиксированный) характер, а могут изменяться в зависимости от конкретных требований.

Фиксированные границы обозначены числом, а переменные — именем системной переменной, в которой они хранятся.

Таблица 4.1

АДРЕС	НАЗНАЧЕНИЕ
0...16383	ПЗУ
16384...22527	Экранная область. Изображение хранится здесь в черно-белом виде.
22528...23295	Область атрибутов. Цвета, признаки яркости и мигания.

23296...23551	Область буфера ZX-принтера. При отсутствии такого принтера часто используется для хранения небольших процедур в машинных кодах. Здесь они не повреждаются БЕЙСИКом, а кроме того, размещение их в нижней половине памяти компьютера повышает их быстродействие. В компьютерах 128К в этой области хранятся дополнительные системные переменные для поддержания повышенных возможностей этих машин. В этих компьютерах категорически нельзя ничего засылать в эту область памяти.
23552...23733	Область системных переменных.
23734...CHANS-1	Карты памяти микродрайва. При работе с микродрайвом здесь хранится информация, например, об испорченных секторах на ленте и т.п. Если микродрайва нет, то эта область не организуется.
CHANS...PROG-2	Здесь размещается информация о каналах.
PROG...VARS-1	Текст программы, записанной на БЕЙСИКе. Отметим, что он хранится несколько в иной форме, чем то, что Вы видите на экране.
VARS...ELINE-2	Область хранения переменных, с которыми оперирует БЕЙСИК- программа.
ELINE...WORKSP-1	Буфер редактора. Сюда помещается строка программы, когда Вы выполняете ее ввод или редактирование.
WORKSP...STKBOT-1	Буфер инструкции INPUT.
STKBOT...STKEND-1	Стек калькулятора. К нему обращается интерпретатор БЕЙСИКа. Калькулятор служит для выполнения арифметических операций. Калькулятор может вызываться из машинного кода командой ассемблера RST 40. По окончании операции на вершину стека помещается ее результат.
STKEND...sp	Свободное пространство. Сюда развиваются навстречу друг другу стек калькулятора и машинный стек. sp - это не системная переменная. Это указатель машинного стека. Его можно вызвать только из машинного кода. Из БЕЙСИКа он недоступен.
sp...ERR SP	Машинный стек. Нужен процессору Z-80 для выполнения обработки данных, адресов и пр.
ERR SP+1...RAMTOP	Стек возвратов (GO SUB-стек). Здесь хранится информация о номере строки, к которой надо вернуться после выполнения подпрограммы RETURN.
RAMTOP...UDG	Свободная область. Может быть использована для хранения процедур в машинных кодах.
UDG...PRAMT	Область графики пользователя.
PRAMT	Физическая вершина памяти компьютера. Для СПЕКТРУМа 48 - PRAMT = 65535, а для СПЕКТРУМа 16 - это 32767.

4.4. Системные переменные

Компьютер выделяет место в оперативной памяти от адреса 23552 по адрес 23733 для размещения в нем некоторых переменных, которые нужны для нормального функционирования компьютера. Тем не менее, пользователь может использовать информацию, содержащуюся в них для повышения эффективности программирования. Первоначальные значения системных переменных устанавливаются после включения компьютера. Их устанавливают те процедуры, которые содержатся в ПЗУ и стартуют после включения. Вы можете в любое время посмотреть содержание однобайтной системной переменной с помощью команды:

PRINT PEEK ADDR, где ADDR - это ее адрес,

а для двухбайтной:

PRINT PEEK ADDR + 256 * PEEK (ADDR + 1)

Вы можете и сами послать какие-либо значения (от 0 до 255) в адрес системной переменной оператором POKE, но в некоторых случаях такое изменение будет иметь лишь кратковременный характер и предыдущее значение будет само восстановлено (такие системные переменные помечены буквой N), а в некоторых случаях изменение приведет к сбросу компьютера (такие переменные помечены буквой X).

Таблица системных переменных

Таблица 4.2

Длина	Адрес	Имя	Содержание
N8	23552	KSTATE	Используется при сканировании клавиатуры
N1	23560	LAST K	Запоминает последнюю нажатую клавишу
1	23561	REPDEL	Время, в течение которого надо задержать нажатую клавишу, чтобы произошел автоповтор. Измеряется в 50-х долях секунды. Исходная установка — 35, но можете заслать сюда свое значение
1	23562	REPPER	Интервал между повторениями нажатой и задержанной клавиши. Измеряется в 50-х долях секунды. Исходное значение — 5
N2	23563	DEFADD	При расчете функции, заданной пользователем, здесь хранится адрес ее аргументов. В противном случае — 0
N1	23565	K DATA	Второй байт кода управления цветом при вводе с клавиатуры
N2	23566	TV DATA	Здесь хранится информация о цвете и о параметрах AT и TAB для телевизора
X38	23568	STRMS	Адрес каналов, подключенных к потокам
2	23606	CHARS	Эта системная переменная указывает на адрес, который лежит на 256 байтов ниже, чем набор символов (знаков), начинающийся с символа "пробел" и кончающийся символом "копирайт". Обычно этот набор находится в ПЗУ, но Вы можете организовать свой набор в ОЗУ и изменить CHARS так, чтобы она указывала на него
1	23608	RASP	Продолжительность предупредительного звукового сигнала

Длина	Адрес	Имя	Содержание
1	23609	PIP	Продолжительность звукового щелчка при нажатии кнопок
1	23610	ERR NR	На единицу меньше, чем код сообщения об ошибке. В исходном состоянии — 255
X1	23611	FLAGS	Различные флаги операционной системы
X1	23612	TV FLAG	Флаги, связанные с работой телевизора
X2	23613	ERR SP	Адрес параметра в машинном стеке, который используется как адрес возврата после ошибки
N2	23615	LIST SP	Адрес возврата после автоматического листинга
N1	23617	MODE	Указывает на тип курсора «K»,«L»,«C »или «G»
2	23618	NEW PPC	Номер строки к которой следует переход
1	23620	NSPPC	Номер оператора в строке, к которому следует переход. Заслав нужное значение в NEWPPC, а затем в NSPPC, вы можете вызвать переход к нужному оператору в нужной строке
2	23621	PPC	Номер строки, в которой расположен исполняемый в данный момент оператор
1	23623	SUB PPC	Номер оператора в строке, который в данный момент исполняется
1	23624	DORDCR	Цвет бордюра, умноженный на 8. Здесь не содержатся атрибуты нижней части экрана
2	23625	E PPC	Номер текущей строки (в которой находится курсор)
X2	23627	VARS	Адрес программных переменных
N2	23629	DEST	Адрес переменной при исполнении оператора присвоения
X2	23631	CHANS	Адрес информации о каналах
X2	23633	CURCHL	Адрес информации, которая в настоящий момент вводится или выводится
X2	23635	PROG	Адрес, с которого начинается программа, написанная на БЕЙСИКе
X2	23637	NXTLIN	Адрес следующей строки программы
X2	23639	DATADD	Адрес разделителя после последнего введенного параметра из блока DATA
X2	23641	E LINE	Адрес вводимой команды
2	23643	K CUR	Адрес курсора.
X2	23645	CH ADD	Адрес следующего интерпретируемого символа
2	23647	X PRT	Адрес символа после маркера «?»
X2	23649	WORK SP	Адрес рабочего пространства
X2	23651	STK BOT	Адрес основания стека калькулятора
X2	23653	STK END	Адрес вершины стека калькулятора
N1	23655	BREG	Содержимое регистра «B» калькулятора
N2	23656	MEM	Адрес области, используемой в качестве области памяти калькулятора. Обычно совпадает с MEMBOT, но это не всегда
1	23658	FLAGS2	Флаги.
X1	23659	DF SZ	Количество строк в нижней части экрана (в системном окне)
2	23660	S TOP	Номер верхней программной строки при автоматическом листинге

Длина	Адрес	Имя	Содержание
2	23662	OLDPPC	Номер строки, к которой выполняется переход по CONTINUE
1	23664	OSPPC	Номер оператора в строке, к которому выполняется переход по CONTINUE
N1	23665	FLAGX	Вспомогательные флаги.
N2	23666	STR LEN	Длина строковой переменной (стринга) в операторе присваивания
N2	23668	T ADDR	Адрес следующего параметра в синтаксической таблице
2	23670	SEED	Здесь храниться величина для функции RND. Исходное значение можно установить оператором RANDOMIZE N
3	23672	FRAMES	Трехбайтная переменная. (Старший байт — первый). Получает приращение на единицу каждую пятидесятую долю секунды. Может использоваться при организации контроля времени. Так, команда PRINT (256*256*PEEK23672 + 256*PEEK23673 + PEEK23674)/50 покажет сколько времени (в секундах) прошло с того момента, как Вы включили компьютер
2	23675	UDG	Адрес 1-го символа графики пользователя. Вы можете его изменить
1	23677	CURDS	Координата X последней помещенной на экран точки
1	23678		Координата Y последней помещенной на экран точки
1	23679	P POSN	Номер колонки позиции, в которой находится принтер
1	23680	PR CC	Младший байт адреса следующей позиции печати по LPRINT (в буфере принтера)
1	23681		НЕ используется.
2	23682	ECHO E	Номер колонки и номер строки конца буфера клавиатуры
2	23684	DF CC	Адрес в экранной области памяти позиции печати
2	23686	DF CCL	То же, но для нижней части экрана
X1	23688	S POSN	Номер колонки позиции печати
X1	23689		Номер строки позиции печати
X2	23690	S POSNL	То же, что и S POSN, но для нижней части экрана
1	23692	SCR CT	На единицу больше, чем необходимое количество передвижек строк на экране до остановки с запросом SCROLL?. Если Вы организуете постоянную подачу после PRINT какого-либо числа большего,чем 1 в эту ячейку,то печать будет идти непрерывно, без запроса SCROLL?
1	23693	ATTR P	Текущие цветовые атрибуты (для всего экрана)
1	23694	MASK P	Маска локальных атрибутов. Каждый бит этой переменной, если он равен единице, показывает, что соответствующий цветовой атрибут надо брать не из ATTR P, а из того, что уже есть на экране в позиции, для которой выполняется печать
N1	23695	ATTR T	Временные атрибуты. Их установка справедлива только для выполнения текущего оператора PRINT, INPUT
N1	23696	MASK T	То же, что и MASK P, но для временных атрибутов
1	23697	P FLAG	Флаги атрибутов

Длина	Адрес	Имя	Содержание
N30	23698	MEMBOT	Область памяти калькулятора. Используется для хранения чисел, которые неудобно помещать в стек калькулятора
2	23728		Не используется.
2	23730	RAMTOP	Адрес вершины области, доступной для размещения программы на БЕЙСИКе
2	23732	P-RAMT	Адрес физической вершины памяти компьютера

4.5. Синтезатор звуковых эффектов.

Ваш компьютер обладает встроенной схемой синтезатора звуковых эффектов, позволяющей получать различные звуки.

Вы, кстати, уже сталкивались с работой этого синтезатора — вспомните звук щелчка при нажатии на клавиши клавиатуры. Это именно его рук дело!

В языке БЕЙСИК управление этим синтезатором представлено всего одной командой **BEEP**, позволяющей подавать звуковые сигналы и исполнять несложные мелодии. Подробнее эта команда описана в книге «**Программирование на языке БЕЙСИК**» Стивена Викерса.

Несколько большие возможности для формирования специальных эффектов представляет загружаемый пакет звуковых эффектов «DZWIEKI», которым можно пользоваться из БЕЙСИКа.

Если же Вы увлекаетесь компьютерной музыкой, для Вас создано большое количество различных музыкальных программ. Например, программа WHAM! MUSIC-BOX позволяет исполнять различные музыкальные произведения в разложении на два канала тонального сигнала и ритм-бокс, с изображением партитуры на нотном стане. Каждый из каналов имеет диапазон 4,5 октавы и свою нотную партитуру. Ритм-бокс имеет три перестраиваемых эффекта и автоподстраивающийся барабан. Партитура запоминается, редактируется, записывается на магнитофон и загружается с него. После отладки партитуру встроенным компилятором можно откомпилировать, превратив в программу в кодах, которую можно уже использовать как составную часть любой Вашей программы.

Если возможности ритм-бокса WHAM! Вам недостаточны, имеется более мощный специализированный ритм-бокс фирмы *EINSTEIN SOFTWARE*. Эта программа имеет уже 10 ударных инструментов, используемых в различных комбинациях, и запоминает до 10 ритмов произвольного размера и длины (вряд ли Вам хватит терпения стучать на нем так долго, чтобы переполнился буфер ритмов). Имеется 10 готовых ритмов. Ритмы можно переключать прямо в процессе исполнения.

Наконец, синтезатор звуковых эффектов позволяет синтезировать даже речь. Правда для этого Вам надо иметь специальные программы, например, LMOVA или TOKER, позволяющие компьютеру произносить слова и фразы на русском, польском или английском языках.

4.6. Встроенный таймер.

Ваш компьютер обладает весьма широкими возможностями по управлению различными процессами и объектами в реальном масштабе времени.

Скорость реакции Вашего компьютера весьма велика и позволяет, например, управлять небольшим промышленным роботом с электроприводом или же программно имитировать работу контроллеров таких устройств, как графический принтер или графопостроитель. В случае графического принтера компьютер одновременно управляет двумя электродвигателями постоянного тока, одним шаговым двигателем и игольчатой печатающей головкой - и имеет еще значительный запас по скорости, что позволяет ему одновременно с печатью еще и оптимизировать процесс печати, программно корректировать люфты печатающего механизма и регулировать контрастность печати.

Однако при решении задач управления в реальном времени очень часто необходим датчик времени — таймер, позволяющий компьютеру отсчитывать временные интервалы. В отличие от большинства компьютеров Ваш компьютер таким таймером располагает.

Технические характеристики таймера:

Дискретность отсчета времени.....0,02 с (50 Гц)

Режим работы .....маскируемое прерывание

Стабилизация частоты.....кварцевым генератором

Максимальный фиксируемый интервал.....328 000 секунд

Программная установка.....есть

Программная блокировка.....есть

Этот таймер используется операционной системой для обслуживания клавиатуры, что не мешает использовать его Вам — только помните, что блокировка таймера приводит к отключению клавиатуры от операционной системы. Однако сама клавиатура при этом остается работоспособной и может использоваться Вашими программами (при этом к ней надо обращаться как к портам ввода — см.описание **«Программирование на языке БЕЙСИК»** Стивена Викерса). При блокировке таймера его показания не сбрасываются, но фиксируются - фактически таймер продолжает работать, но подсчет временного интервала прекращается. При снятии блокировки это позволит Вам продолжить подсчет временного интервала с того места, на котором Вы его прервали.

Для блокировки таймера достаточно всего лишь запретить процессору прерывания - и таймер остановится. Разрешив прерывания, Вы разблокируете таймер.

5. Программное обеспечение ПЭВМ

С некоторыми программами Вы уже познакомились в разделах «Синтезатор звуковых эффектов» и «Программы на ассемблере Z80». На самом деле таких программ очень много, как и возможностей для их применения. Описывать их все невозможно, поэтому дадим лишь названия некоторых программ с пояснениями по областям их применения:

Языки программирования, редакторы, прикладные программы:

- БЕЙСИК-интерпретаторы: BETA-B.48 (BB48), BETA 3.1, LASER-BASIC, BETA BASIC V.1.8, BEYOND, MEGA BASIC, TOOLKIT
- БЕЙСИК-компиляторы: F-COMPILER, FP 48 K V1.7, TOBOS-FP, IS48K V1.7
- Паскаль: PASCAL HP-80, HP-4 TM 8, HP-4 TM 16
- Си: HISOFT C, CC, CCOMP.1
- Форт: FIG-FORTH, ED 50, EP 50, FP48K
- Лого: LOGO, LOGO 16
- Пролог: PROLOG
- Ассемблер: GENS-4, ZEUS, ASSEMBLER, EDITAS, DEVPAC, +80EDT/ASW
- Отладчики кодов: MONS-4, MONITOR 48, MONITOR 16, IR 48K, DILO, LMON, SPDE
- Картотеки: MASTERFILE V 09, INF1, KART, MF 09 LEER, MFC LEER, VU-FILE
- Электронные таблицы: OMNYCALK, OM 64000
- Текстовые экранные редакторы: TASWORD TWO, TAS.RUS, TLM, VU-FILE
- Графические редакторы: ART STUDIO, ARTIST, DYNAMIC-3, LORIGRAPH, M.DRAW, LPEN 48, LP MK 2.3

Внимание: Все графические редакторы, кроме последнего, работают с клавиатурой, джойстиком или манипулятором «мышь». Последний графический редактор работает со световым пером.

- Иммитаторы дисковой операционной системы: RAM DOS 2, RAM-DISC
- Тесты системы: TEST_PROG, TEST BAS, MEMORY TEST, TST 80.V3, MTR
- Трассировщики печатных плат: PLATA V.3, ТРАССА-3, PROJ/PLYT, TRACE V.1
- Спектроанализаторы и цифровые осциллографы: OSCILLATOR, TAPER, DIAG, TAPE HEAD, OSCILOSCOPE
- Математические программы: FUN F(X,Y), STATIST, FOURIER, PB-M2
- Редакторы знакогенератора пользователя: UDG DEFINER, PAINTBOX, UDG
- Синтез речи: FONGEN, LMOWA, SPEAKER, SPEAKEASY, PB-V1
- Пакеты для БЕЙСИКа: ZXED, COMPRESSOR, 64*32, RUS, RENUM, 64-COLUMNY

6. Компьютерные игры

Среди многих профессионалов-программистов бытует мнение о некоей «несерьезности» игр на компьютере, что это не та сфера, в которой следует тратить свои силы и время. Однако сравнение серьезных программных продуктов, созданных «серьезными» и «несерьезными» программистами, показывает преимущество последних. Их программы имеют более дружественный характер, написаны гораздо изобретательнее (не в смысле невидимых всему миру выкрутас в кодах, а в смысле подходов, методов, форм представления результатов и т.д.) и продуманнее.

Более того! Сравнение компьютеров по критерию компьютерных игр, на них реализованных, гораздо сильнее выявляет их возможности для «серьезного» использования, нежели само это использование.

И это не удивительно. Современные компьютерные игры являются сплавом новейших достижений в технике программирования, разработке интерфейса человек-машина, компьютерной графике, системах искусственного интеллекта, экспертных системах, синтезе речи и музыки, помноженным на мастерство и фантазию программистов. Здесь идет в ход все, даже то, что еще не решаются использовать в больших «серьезных» системах, что еще существует лишь в голове автора и не прошло никаких проверок - все равно ведь несерьезно!» А потом в результате серьезные люди смотрят на работу таких программ и говорят: «Этого не может быть, процессор не успеет (памяти не хватит, алгоритм не реализуем и т.д.)».

Однако программы от таких разговоров не начинают хуже работать, а лишь демонстрируют, насколько велики резервы компьютера при правильном их использовании.

Как же обстоит дело с компьютерными играми для Вашего компьютера?

По итогам 1988 года (данные журнала «BYTE») данный тип компьютеров занял второе место по количеству разработанных и проданных программ (более 11000 видов), львиную долю из которых составляют компьютерные игры, уступив только компьютеру «APPLE2». В настоящее время число программ для «Спектрума» превышает 20000 наименований.

По качеству игр Ваш компьютер оставил далеко позади такие машины, как ATARI XL/XE, COMMODORE 16/64/128, YAMAXA MSX и MSX2, не говоря уже об отечественных РК-86, Микроша, БК-0010, ДВК-3, ДВК-4 и даже ЕС1840/1841. С появлением модели SPECTRUM 3+, имеющей ОЗУ 128К и дисковод, качество игр данного семейства компьютеров еще более поднялось и вплотную приблизилось к таким системам, как IBM PC и COMMODORE AMIGA, несравненно более сложным и дорогостоящим, чем Ваша система.

Однако большинство игр так до сих пор и ориентируется на модель ZX SPECTRUM 48K, и следовательно, будет работать на Вашей машине.

6.1. Структура компьютерной игры

С точки зрения пользователя, компьютерная игра представляет собой несколько файлов, записанных на кассете для магнитофона. Типы, размеры и количество файлов варьируется от игры к игре, от фирмы к фирме, однако можно выделить три основных типа структур:

1) загрузчик на БЕЙСИКе — блок (блоки) кодов

2) загрузчик на БЕЙСИКе —загрузчик в кодах — блок(блоки) кодов

3) программа на БЕЙСИКе со встроенными кодами

Примером первой структуры может служить игра «**LUNA CRABS**», второй — «**SILENT SERVICE**», третьей — игра «**WHEELIE 2**».

Наиболее общим является второй тип структуры — его и рассмотрим подробнее. Начнем с первого файла типа «**Programm**». Если его загрузить не командой **LOAD ""** , а командой **MERGE ""**, можно увидеть примерно следующую программу - загрузчик на БЕЙСИКе:

```
0 ... : LOAD "..." SCREEN$ : LOAD "..." CODE RANDOMIZE USR ...
```

Здесь команда **LOAD "..." SCREEN$** загружает на экран картинку-заставку, рассматривая которую Вам будет легче ждать конца загрузки всего остального.

Следующая команда **LOAD "..." CODE** загружает загрузчик в кодах, и последняя команда **RANDOMIZE USR ...** его запускает.

Как видите, достаточно загрузить такую программку (она, как правило, имеет атрибут самозапуска и сама начинает работать), и все остальные файлы будут загружены и запущены безо всякого вмешательства, аналогично тому, как на компьютере IBM PC при запуске пакетного файла (.BAT-файлы) он сам начинает загружать и запускать нужные Вам программы в нужном порядке.

Обратите внимание на номер строки. Набрать такую строку с клавиатуры невозможно, как, впрочем, и отредактировать. «Нулевая» строка формируется специальным программным образом и служит для того, чтобы не дать Вам забраться в программу. Обычно кроме зануления строки применяются еще и специальные меры, чтобы Вы не увидели на экране текст программы. Например, дайте команды **INK 0 : PAPER 0 : CLS** — и Вы не увидите на экране ничего — он будет черен, как ночь. Бороться с этими командами очень просто — дайте команды **INK 9 : PAPER 9**, и видимость значительно улучшится.

Примечание: не всегда загрузчик на БЕЙСИКе выглядит точно так. Например, перед запуском загрузчика в кодах могут загружаться еще несколько файлов в кодах. Или, наоборот, загрузчик в кодах будет спрятан в картинке, и сразу после ее загрузки будет запущен. Или даже загрузчик в кодах может быть загружен самым первым, и уже он будет грузить заставку.

Загрузчики в кодах бывают трех видов:

- использующие стандартные подпрограммы загрузки из операционной системы;
- использующие собственные драйверы магнитофона, но сохраняющие стандартный или близкий к стандартному формат записи;
- использующие собственные драйверы магнитофона и собственный формат записи (или собственную структуру файла).

В последнем виде несколько особняком стоят так называемые компонующие загрузчики. Это наиболее изощренный вид загрузчиков, использующий специальную структуру файла - блочно-адресную структуру. Размер блока данных для такого

загрузчика может значительно превышать размер ОЗУ компьютера (48 кбайт), что создает большие трудности при копировании такого блока. Зато при работе такого загрузчика может формироваться более качественная программа, выводиться несколько последовательных заставок или одна заставка последовательно трансформируется в другую, что развлечет Вас, пока Вы ждете конца загрузки программы (загрузка может длиться от 2 до 5 минут).

6.2. Как загрузить игру?

После всего вышеописанного Вы, наверное, испуганы сложностью структуры игры и думаете, что загрузить ее сможет только мастер? Это вовсе не так. Как раз компьютерную игру загрузить проще всего — достаточно выполнить действия, описанные в разделе «Как загрузить и выгрузить программу?» для загрузки программы на БЕЙСИКе. При этом у Вас загрузится загрузчик на БЕЙСИКе, и он уже сам начнет загружать все, что нужно программе.

Внимание!	Если при загрузке компьютерной игры произошел сбой, стандартное сообщение «**TAPE LOADING ERROR**» может не появиться, так как загрузчики в кодах нарушают работу операционной системы. При этом сбой часто обнаруживается лишь в конце загрузки, когда программа не запускается или, поработав немного, зависает. В таком случае следует нажать кнопку <Сброс> и загрузить игру сначала.

Если после нескольких попыток загрузка все равно не удается, причиной может быть деформация ленты с записью программы. Обычно этот участок ленты повторно использовать невозможно. Не пытайтесь вырезать поврежденный участок ленты — в месте склейки все равно будет неустранимый сбой загрузки.

Если после нескольких попыток загрузка все-таки удалась, после игры перекопируйте такую программу на новую кассету, иначе со временем она совсем перестанет загружаться.

6.3. Как скопировать игру?

Компьютерные игры можно скопировать только программами-копировщиками. Процесс копирования подробно описан в разделе «**Копирование программ**».

6.4. Как играть в игру?

Обычно после загрузки игры на экране появляется меню управления, где перечислены различные устройства управления игровой ситуацией, например:

```
PRESS KEY TO SELECT MENU
1 KEYBOARD
2 KEMPSTON JOYSTICK
3 CURSOR JOYSTICK
4 SINCLAIR JOYSTICK
5 REDEFINE KEYS
0 START GAME
```

Здесь:

KEYBOARD — клавиатура
KEMPSTON JOYSTICK
CURSOR JOYSTICK } —различные типы джойстиков
SINCLAIR JOYSTICK
REDEFINE KEYS — переназначение клавиш

Джойстик —это специальная игровая ручка, в которой имеется пять контактов: четыре указывают направление, в котором наклонена ручка, а пятый находится под кнопкой «огонь», обычно установленной на ручке (на основании ручки может находится еще одна кнопка «огонь», подключенная параллельно основной кнопке). Все это устройство напоминает рукоятку управления самолета-истребителя (и не удивительно - ведь назначение их очень близко).

Существует несколько видов джойстиков:

SINCLAIR JOYSTICK
INTERFACE 2/2
AGF INTERFACE
AGF JOYSTICK
PROTEK JOYSTICK
KEYBOARD JOYSTICK

Одно из этих устройств обязательно будет использоваться в любой игре. Если же Вы собираетесь играть на клавиатуре, старайтесь использовать следующее назначение клавиш (оно стандартно для многих игр и наиболее удобно при QWERTY-клавиатуре Вашего компьютера):

UP.....Q
DOWN.....A
LEFT.....O
RIGNT.....P
FIRE.....M
другое.....X

При таком назначении клавиш указательный и средний пальцы Вашей правой руки нажимают клавиши <O> и <P>, управляя движением влево-вправо, а такие же пальцы левой руки нажимают клавиши <Q> и <A>, управляя движением вверх-вниз. Клавиши <M> и <X> нажимают большим пальцем правой и левой рук соответственно.

После нажатия клавиши, соответствующей «**START GAME**», начнется собственно игра. Многие игры больше не дадут Вам возможности выбрать другое устройство управления, поэтому отнеситесь к выбору управления внимательно.

Сложность игровой ситуации разных игр весьма различна. Есть игры, смысл которых ясен сразу —знай нажимай на клавиши. К таким играм относится, например, игра «**GREEN BERET**», в которой лихой десантник — «зеленый берет» проникает на ракетную базу и, вооруженный вначале лишь ножом, преодолевает сопротивление солдат и специальных охранных подразделений, прорывается в ядерный бункер, затем на ядерную подводную лодку, использует добытые в бою огнеметы и гранатометы — и одерживает победу, предотвращая ядерный удар противника. В игре Вам понадобится виртуозное владение клавиатурой, мгновенная реакция — ведь врагов придется убивать буквально голыми руками (и ногами) — а также максимальная концентрация внимания. Игра очень динамична.

Другие игры не требуют такой быстрой реакции, хотя и она временами пригодится, но отличаются значительной сложностью и запутанностью обстановки, требуя полного проявления Ваших аналитических способностей. Примером такой игры может служить игра **«MOVIE»**, действие которой происходит в большом доме в одном из латинских кварталов Лос-Анжелеса. Главному герою — невозмутимому агенту ФБР — противостоят мафиози, которыми набит дом, и разные странные штуки вроде стальных рыцарей—убийц с электронной начинкой или зарядов взрывчатки, установленных в самых неожиданных местах. Однако Вам могут и помочь— если не делом, так хоть информацией — другие обитатели дома. Общаться с ними можно на обычном английском языке, так что советуем его подучить. Игра отличается интересной объемной графикой, весьма сложна и требует кропотливого исследования.

В целый ряд игр можно играть, только имея специальное описание — сценарий. То есть играть-то можно и без описания, а вот выиграть без него нельзя — слишком сложна обстановка игры. Примером таких игр являются три игры фирмы *MICRO-GEN*: **«Everyone's Wally»**, **«Pyjamarama»** и **«Three Weeks in Paradise»**, объединенные одним героем - неунывающим толстяком Волли. Только краткое описание обстановки и действий героев каждой из этих игр занимает 4-5 машинописных страниц. Тем не менее, если Вы имеете описания этих игр, процесс игры не очень сложен и очень увлекателен.

6.5. Что делать, если все время убивают?

Такой вопрос не редкость. Целый ряд игр предъявляет слишком высокие требования к скорости реакции и внимательности играющего, так что даже виртуозы клавиатуры и джойстика не могут в них выиграть. Это происходит оттого, что авторы игр, зная досконально свои «детища», усложняют их до тех пор, пока сами не начнут выигрывать с трудом. Человек, не знакомый с особенностями такой программы, оказывается в таком случае бессилен.

Что же делать? От названия команды встроенного БЕЙСИКа POKE, записывающей в определенную ячейку ОЗУ компьютера определенное значение, произошел термин «запокование» программы. Именно эта команда позволяет так изменить работу программы, что у Вас появится «бесконечная жизнь», или «бесконечное время», или «бесконечное оружие», или все это вместе. Вся сложность состоит в том, что нужно знать параметры (адрес и байт данных) команды **POKE** и суметь подать ее (одну или несколько) по окончании загрузки программы.

Узнать необходимые параметры можно из специальных зарубежных изданий (SINCLAIR USER, BYTE, BYTEK и др.). Некоторые отечественные издания, которые помогут Вам в этом, приведены в Приложении 9 "Библиография", некоторые "POKES" приведены в Приложении 8 "Лучшие игры года". Также Вы можете обратиться к пользователям "Спектрум" - любителям вскрывать программы.

Подать необходимые команды по окончании загрузки можно, лишь взломав загрузчик игры и встроив эти команды в него. Правда, есть ещё вариант, можно копировать программу копировщиком, имеющим режим **POKE**, например, «COPY-COPY» или **«COPY NEW»**.

Если такой вариант возможен, он особо оговаривается.

Например, взломаем загрузчик игры **«Indiana Jones»**, для этого загружаем сам загрузчик с помощью команды **MERGE ""**:

```
1 CLEAR 24791 : LOAD "" CODE : LOAD "" CODE : RANDOMIZE USR
24830 : LOAD "" CODE 16464 : RANDOMIZE USR 24833
```

Так он выглядит в оригинале. Вставим в него **POKE 33948,0** (между командой **RANDOMIZE USR 24830** и командой **LOAD "" CODE 16464**) и запустим загрузчик командой **RUN**. Теперь можете включить магнитофон и загрузить программу **Indiana Jones** - и Вы увидите, что у Вас появилась бесконечная жизнь.

6.6. Проблема совместимости

В процессе работы с самодельным компьютером Вы наверняка столкнетесь с тем, что программы, нормально работающие у Ваших друзей, не желают работать на Вашем компьютере. Не спешите ругать свой магнитофон — иногда причина гораздо более серьезна. Эта причина — неполная аппаратная совместимость Вашего компьютера с фирменным компьютером ZX Spectrum. Возможные проявления: программа не стартует, зависает или происходит сброс системы, либо происходит замедление работы программы. Однако, программ, на которых проявляется несовместимость, довольно мало.

7. Как загрузить программу?

Ваш компьютер обладает способностью сохранять данные и программы на внешнем устройстве магнитной памяти — попросту на обычном магнитофоне. Для работы Вам потребуется магнитофон, все равно какой — катушечный или кассетный, моно или стерео, высокого или низкого класса, а также кассеты. Кассеты лучше применять импортные, но годятся и отечественные, по возможности неизношенные. Измятость и склейки ленты абсолютно недопустимы! Это же относится и к бобинам, за исключением того, что и отечественные ленты в бобинах работают не хуже импортных.

Подключите к разъему «магнитофон» компьютера шнур, соединенный с линейным выходом магнитофона. Вставьте кассету с программой в магнитофон. Перемотайте кассету к интересующей Вас программе.

Наберите на клавиатуре компьютера директиву **LOAD ""** (это делается очень просто: нажмите клавишу <J>, затем клавишу <SYMBOL SHIFT> и, удерживая ее, два раза клавишу <P>). Эта директива означает, что мы пытаемся загрузить программу на БЕЙСИКе. Нажмите клавишу <ENTER> и включите магнитофон на воспроизведение.

Внимание! Помните, что между кавычками не должно быть пробела.

Основную проблему на этом этапе представляет плохая загружаемость программ, о чем свидетельствует сообщение об ошибке «**TAPE LOADING ERROR**», зависание программы после загрузки или самопроизвольный сброс. Причина не обязательно связана с плохой загрузкой. Если лента получена Вами из стороннего источника, то это скорее всего по этой причине. Здесь мы приведем некоторые принципы, которыми надо руководствоваться, чтобы улучшить загружаемость программ.

1. По-настоящему надежную загрузку может обеспечить только высококачественная лента, если она записана на том же магнитофоне, на котором и воспроизводится. Поскольку стоимость программ обычно значительно превышает стоимость ленты, то не надо на ней экономить. А понравившиеся Вам программы надо откопировать на своем магнитофоне (см. ниже).

2. Для работы с компьютером годится любой бытовой кассетный магнитофон, причем чем он проще, тем легче добиться надежной работы. В то же время желательно, чтобы он имел регуляторы тембров (по возможности раздельные). Компьютер мало чувствителен к отклонениям скорости ленты, необходимо лишь следить за чистотой головок магнитофона. Монофонические магнитофоны во всех случаях предпочтительнее, чем стерео. Обязательным элементом является счетчик ленты, позволяющий быстро находить нужную программу.

3. Отечественные компьютеры рассчитаны на подключение к линейному выходу магнитофона. Если не удаётся получить надёжную загрузку, можно «снять» сигнал с выхода УНЧ магнитофона. В магнитофонах низкого класса есть выход для подключения «наушника». Тогда сигнал можно отрегулировать регуляторами громкости и тембра. Кстати, именно такое подключение предусмотрено в фирменных компьютерах.

Приступая к загрузке программы, установите уровень громкости на магнитофоне примерно на 70% —80% от максимального и, если загрузка не пошла, поэкспериментируйте с ним. Помните, что программы чаще страдают от избыточного уровня, чем от недостаточного. Поиск ведите в сторону уменьшения уровня. Иногда на фирменных компьютерах помогает следующий прием. Загрузочный шнур вставляют не в гнездо «EAR» компьютера, а в гнездо «MIC». На самодельных отечественных аппаратах этот метод не сработает, т.к. там эти цепи электрически разорваны.

4. Установка регуляторов тембра также имеет большое значение. Нормально «высокие» надо ставить на 80% от максимума, а «низкие» примерно на 20%. Если программа решительно не загружается, а регулировка громкости ничего не дает, то попробуйте установку 50%-50% и 20%-80%.

5. Загрузка программ сопровождается перемещением цветных полос по бордюрному полю телевизора. По нему можно также многое сказать о настройке магнитофона. Сначала идут в течение 3-х - 5-ти секунд широкие красные и голубые полосы. Это так называемый пилоттон. Он нужен для того, чтобы процессор подготовился к загрузке. Ширина красных и голубых полос должна быть одинаковой. Если голубые полосы шире, то значит уровень слишком большой. Если шире красные полосы — слишком низкий. Эти полосы должны медленно перемещаться по экрану. Слишком быстрое их перемещение говорит о несоответствии скоростей Вашего магнитофона и того, на котором была сделана запись. Неравномерное перемещение (с ускорением и рывками) свидетельствует о некачественности лентопротяжной системы магнитофона либо Вашего, либо того, на котором делалась запись.

Бывают случаи, когда широкие красные и голубые полосы «превращаются» в широкие желтые и синие. Это может указывать на то, что следует поменять местами установку регуляторов тембра. Однако, эти манипуляции с регуляторами могут понадобиться только для «вытягивания» плохо записанных программ. Обычно используют подключение к линейному выходу магнитофона.

6. Если у Вас стереомагнитофон и программа не загружается, попробуйте загрузить ее отдельно по левому и по правому каналам. Для этого на компьютере лучше сразу установить переключатель левого/правого каналов.

7. Если все пути регулировки исчерпаны, но программа не загружается и при этом точно известно, что копия работоспособна, по всей видимости причина в настройке головок у писавшего и воспроизводящего магнитофона. Как правило недорогие монофонические магнитофоны имеют в крышке отверстие для узкой отвертки, чтобы можно было при воспроизведении записи на слух, поворотом подпружиненного винта, регулировать положение головки. При этом следует добиваться наиболее четкого и резкого звучания сигналов. Обычно бывает достаточно поворота в ту или иную сторону на угол до 120 градусов. Если у Вас только один магнитофон, и Вы используете его также и для записи программ, то регулировать головку нежелательно, либо придется периодически сталкиваться с необходимостью юстировки. Самый лучший вариант — использовать для воспроизведения программ, полученных со стороны, отдельный магнитофон, выбрав его из самых простых и дешевых, а для записи программ — наиболее качественный, по возможности специализированный аппарат.

8. Обычно программы состоят из нескольких блоков, которые загружаются последовательно один за другим. При плохой загрузке бывает очень трудно загрузить программу, разные блоки которой требуют разной установки органов управления магнитофона. Для того, чтобы обойти эту проблему, надо проводить загрузку в копирующую программу. Если, предположим, при загрузке второго блока произойдет сбой, то первый блок уже загружен и остается, так что его повторять не надо. К тому же, по

окончании длительной трудоемкой загрузки можно сразу же выполнить качественную копию и больше не иметь с этой программой проблем.

9. Сделать процесс настройки магнитофона наглядным и значительно более простым можно с помощью специальных программ, например с помощью программы «TAPER».

Примеры директив загрузки:

LOAD ""—загрузить первую встреченную БЕЙСИК-программу

LOAD "DEMO" — загрузить БЕЙСИК-программу «DEMO»

LOAD "" CODE — загрузить первый встреченный блок кодов с адреса, указанного в заголовке блока

LOAD "RUNNER" CODE 30000 — загрузить блок кодов «RUNNER» с адреса 30000.

LOAD "" SCREEN $ —загрузить любое изображение в экран

LOAD "FULL" SCREEN $ — загрузить картинку «FULL»в экран

При успешном окончании загрузки в нижней строке экрана должно появиться сообщение "0,1 OK" (от слов O'KEY) или должна начать работать программа.

Примечание: файлы программ на БЕЙСИКе могут иметь атрибут самозапуска (AUTO-RUNNING), при этом загруженная программа сразу начинает выполняться, не дожидаясь Ваших команд. Внутри программы на БЕЙСИКе могут содержаться любые директивы загрузки (и выгрузки), поэтому программа сама может загружать любые файлы, в том числе и программы (на БЕЙСИКе и в кодах), а также запускать их на выполнение. Так что не удивляйтесь, если после загрузки БЕЙСИК-программы загрузка будет продолжаться, будут загружены картинки на экран, программы в кодах и т.д.

Механизм работы таких БЕЙСИК-программ похож на механизм работы пакетных файлов (BATCH-файлов) компьютера **IBM PC**, но имеет гораздо более широкие возможности.

Подробнее о таких программах см. раздел «**Компьютерные игры**».

8. Копирование программ

Если Вы сами написали какую-либо программу на БЕЙСИКЕ, то выполнить ее копию несложно. Для этого достаточно дать команду

SAVE "имя"

и нажать <ENTER>. Если Вы хотите, чтобы программа автостартовала после загрузки, начиная со строки M, то команда имеет вид:

SAVE "имя" LINE M.

Если в программе есть блоки, записанные в машинном коде, то дается команда

SAVE "имя" CODE M,N

где M — адрес, с которого начинается блок кодов; N — длина этого блока в байтах. При загрузке программ блоки, записанные на БЕЙСИКе, индицируются на экране как

PROGRAMM ... ,

а блоки машинных кодов как

BYTES...

Если Вы выгружаете свою разработку, то Вы, конечно, знаете те адреса, в которых располагаются Ваши блоки. Однако, если Вы имеете дело с чужой (фирменной) программой, то загрузив ее, Вам не так-то просто ее выгрузить. Во-первых, она автостартует и остановить ее сложно. Во-вторых, Вам неизвестны адреса блоков, из которых она состоит.

Упростить процесс копирования и автоматизировать его можно с помощью специальных копирующих программ. Эти программы имеют свою нестандартную загрузочную систему. Таких программ очень много. В предлагаемом списке основных программ-копировщиков в скобках указан автор программы, год её создания и название фирмы:

- BAUD COPY (*M.BAYER, 1984*)
- COPIER (*SINCLAIR*)
- COPIER FM-3, он же SINCLAIR COPY
- COPY 86/M
- COPY DE LUXE
- COPY PLUS (*R.D., 1983, DAKRASOFT*)
- COPY-COPY (*T.WILCEK, 1984-1985*)
- DUPLICATE (*J.STESSUN*)
- ENGINE COPY
- FILE COPY
- FRONT COPIER
- LADY COPY (*версия 1.05, LADINEK, 3/86*)
- LAST HOPE (*Д.Ю.Степаненко, 1990*)

- LERM-7 или TC-7
- MR COPY (*версия I/XI, P.RACZKOWSKI, 8/84*)
- MR COPY + (*P.RACZKOWSKI, ING.P.KREYBIG, 5/86*)
- PIRACY (*V.J.SOFT, 1984*)
- PIRATE 02
- QUICK COPY
- QUICK SAVE (*версия 1.0, INDIVIDUAL SOFTWARE, 1984*)
- SUPERCOPY (*SPENCER, 1983*)
- TAPE-TO-TAPE (*Ю.Никольский, 1986*)
- TF COPY (*ARNOST VECERKA, 86-88*)
- THE KEY (*V.7, KEYSOFT, 1983*)
- X-RAY (*версия 2.7, M.LAMB, 1983*)

Впрочем, такое количество копировщиков необходимо только профессионалам, имеющим дело с заводскими кассетами программ, для взлома блокировок и других программных вывертов. Вам, конечно, столько не потребуется. Здесь мы рассмотрим несколько наиболее широко распространенных копировщиков.

8.1. COPY 86/M

Это, по-видимому, наиболее удобный и наглядный копировщик общего применения. В исходном состоянии он имеет объем свободного пространства 45000 байтов, и способен компрессировать данные при загрузке. Компрессирование состоит в том, что например вместо последовательности из N нулевых байтов записывается один «нуль», а следующий байт указывает сколько их (N). Это позволяет загружать в копировщик значительно более 45 К. Особенно много места экономится при загрузке экранов, т.к. в графических изображениях часто встречаются длинные последовательности нулей.

Результат компрессии данных отправляется на хранение в область экрана, что изображается появлением на экране полос и точек, напоминающих телеграфный код.

Сводка команд программы COPY 86/M

BREAK — переход в исходное положение

L — загрузить файлы

C — выгрузить (скопировать файлы). Выгрузка производится с паузой 1,5 сек. между записями

M — то же, но с паузой 3 сек.

V (VERIFY) — проверка выгруженных записей

D (DELETE) — удаление ненужных записей

H — перевод в шестнадцатиричную систему

X — просмотр записей с отбоем ранее поданных команд

S — то же, но без отбоя команд

B — просмотр БЕЙСИК-программ. <Y> — построчно, <P> — на 4 строки вперед

A (ALL) — после этой команды очередная команда будет одновременно распространяться на все записи.

Все команды подтверждаются нажатием <ENTER>.

На экране файлы записываются в «окне», имеющем 4 строки. Если записей более чем 4, то одновременно их вывести на экран нельзя, поэтому и необходимы клавиши <S> и <X> для «протягивания» записей через «окно».

При использовании программы следует помнить, что она может не работать с некоторой периферией. Так, она не будет работать, если к компьютеру подключен кемпстон-джойстик.

К сожалению, эта программа не работает и со многими самодельными компьютерами. В таком случае как правило пользуются другой компрессирующей программой **TF COPY**.

8.2. TF COPY

После загрузки программы на экране появляется исходное меню. Нажмите <0>, программа стартует и перед Вами появится основное меню программы, которое выглядит так:

LOAD SAVE DELETE VERIFY MODE

Нажав первую букву нужной команды, Вы входите в нужный Вам режим.

LOAD — загрузка файлов

SAVE —выгрузка файлов

DELETE — удаление файлов

VERIFY — проверка выгруженных файлов

MODE — переключение режима

В программе имеются 3 режима работы:

Режим 1 — **41984** байта свободной памяти

Режим 2 — **44032** байта свободной памяти

Режим 3 — **44288** байтов свободной памяти.

Все команды должны завершаться нажатием <ENTER>.

Обратите внимание, что при переходе на другой режим происходит очистка памяти, т.е. загруженная информация будет уничтожена.

После выбора одного из вышеуказанных режимов программа выдает запрос. Например, нажатие <D> дает режим «DELETE» и появляется запрос

DELETE FROM ...TO ... (Удалить записи с ... по ...).

В ответ надо дать номера тех записей, которые Вам нужны. Например, если Вы хотите уничтожить записи с третьей по шестую, то соответственно надо ввести 3 и 6.

Внимание:	Серийно выпускаемый компьютер "ДУБНА" (его первая модель) не поддерживает работу с копировщиками COPY 86/M и TF COPY.

8.3. COPY-COPY

Эта программа находится в эксплуатации довольно давно и представляет классический образец копировщика. Она не выполняет компрессирование и имеет довольно ограниченный объем свободной памяти (42K), но в ней есть несколько оригинальных вспомогательных режимов, которые в ряде случаев делают ее незаменимой.

Сводка команд

Все команды являются ключевыми словами СПЕКТРУМа и потому не набираются по буквам. Они требуют завершения нажатием клавиши <ENTER>.

LOAD — выполнить загрузку очередной записи (очередного блока)

LOAD N TO M — выполнить загрузку записей, начиная с номера N до номера M.

При этом ранее существовавшие записи с номером N и выше будут уничтожены, т.е.

```
LOAD 1
```

уничтожит все ранее существовавшие записи и подготовит компьютер к загрузке новых.

Возможные вариации этой команды:

```
LOAD N TO
LOAD TO M
```

SAVE — выполнение выгрузки записи (записей) на ленту. Может иметь форму:

SAVE N TO M

SAVE TO M

SAVE N TO

SAVE N TO M STEP K

Параметр K после оператора STEP показывает величину паузы в секундах, которую компьютер выдержит между отдельными блоками.

VERIFY — проверка выгруженных записей. Имеет все те же формы, что и команда LOAD.

LOAD AT ADDR — загрузка блока, начиная с адреса ADDR.

Например:

LOAD AT 16384 — загрузка экрана.

LIST ADDR — выдача на экран содержимого памяти, начиная с указанного адреса. LIST = LIST 0.

POKE ADDR,N — поместить по указанному адресу байт N.

Этой командой часто пользуются совместно с командой LIST. Они позволяют вносить изменения в машинный код программы. В принципе с подобными задачами лучше справляются специализированные программы для отладки машинного кода (их часто называют МОНИТОРАМИ), но наличие такой функции в копировщике делает его весьма удобным.

COPY 16384 — команда, которая позволяет копировать блоки длиной 49K.

Этот режим делает данную программу незаменимой во многих случаях. Дело в том, что если программа имеет один значительный блок длиной порядка 49K и не поддается

компрессии, то откопировать его копировщиками типа «COPY 86/M» и «TF COPY» не удастся, т.к. он не помещается в памяти. (Например программы FIRELORD и URIDIUM фирмы «*HUSON CONSULTANTS*» и другие). В режиме "COPY 16384" программа «COPY-COPY» удаляет саму себя из памяти и оставляет только небольшой блок длиной несколько байтов, предназначенный для выгрузки загруженного блока. Выгрузка выполняется нажатием клавиши <CAPS SHIFT>. Возможна только однократная выгрузка, после этого копировщик необходимо снова загрузить. Разумеется магнитофон должен быть включен на запись до нажатия <CAPS SHIFT>.

8.4. Дополнительные сведения

Мы рассмотрели подробно только три наиболее распространенные копирующие программы. На самом деле их очень много, но принципы их работы отличаются не очень значительно. В то же время, надо помнить, что с помощью копировщиков можно только копировать программы, в которых не принято специальных мер для защиты от копирования. Многие фирмы такие меры применяют. Это например «спидлок» (ускоренный загрузчик), джеркитон (пилоттон, сопровождающий рывками), блоки избыточной длины (более 50K), фальшхэдеры (блоки длиной 17 байтов, которые воспринимаются копировщиком как заголовок очередного блока, в то время как они таковыми не являются), укороченный пилоттон, удлиненный пилоттон, слишком широкий или слишком узкий пиллотон, блоки с замеряемой паузой между ними и многие другие методы. *Принцип их действия состоит в том, что первый блок программы является* БЕЙСИК-загрузчиком, который подготавливает загрузку второго блока. Второй блок в машинных кодах подготавливает нестандартную загрузку прочих блоков, которые уже не могут быть загружены никуда, где предварительно не отработал второй блок, в том числе и в копировщик.Копирование таких программ довольно трудоемкий процесс, но со многими защитами справляется например специальная программа LERM-7 (TC-7).

Другой подход к таким программам состоит в том, что используются специальные периферийные устройства. Их назначение — остановить («заморозить») программу в каком-либо месте, а затем сделать полный дамп памяти (полную выгрузку всей памяти). Широко известны устройства типа MULTIFACE, позволяющие кроме этого производить выгрузку экрана в любом месте программы, вносить в программу изменения и продолжать исполнение программы с места прерывания. Аналогичными устройствами снабжаются иногда и дисковые системы, например, БЕТА-ДИСК ИНТЕРФЕЙС (TR DOS) фирмы *TECHNDOLOGY RESEARCH*. Он имеет специальную кнопку, называемую MAGIC BUTTON (волшебная кнопка) для прерывания работы и сброса ее на диск. Правда запустить откопированную таким образом программу через LOAD не удается. Для этого интерфейс реализует нестандартный подход и выполняет это по команде GO TO «имя».

Можно предложить и такой способ копирования с помощью двух магнитофонов. Один магнитофон включается на вход компьютера, а другой — на выход. Компьютер считывает информацию бит за битом и тут же транспортирует ее на выход, при этом он выполняет роль компаратора и выравнивает фронты сигналов.

Вот пример небольшой программы, предназначенной для копирования программ с одного магнитофона на другой через компьютер.

```
10 DIM a(12)
20 FOR i=1 TO 12
```

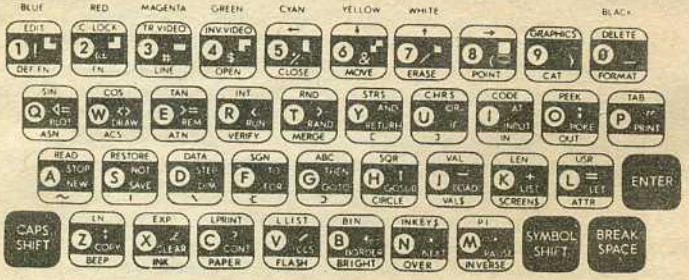
```
30 READ a(i)
40 POKE (29999+i), a(i)
50 NEXT i
60 DATA 243, 175, 219, 254, 31, 31, 31, 211, 254, 195, 50, 117
70 RANDOMIZE USR 30000
```

В принципе эта программа написана в машинном коде. БЕЙСИК служит только для того, чтобы организовать этот код по адресу **30000** и стартовать его.

Правда этот метод не очень надежно работает на фирменных СПЕКТРУМах, т.к. у них вход и выход имеют электрическую связь, но на компьютерах «ТАЙМЕКС» и на большинстве самодельных дает отличный результат.

Приложение 1

Клавиатура компьютера



Приложение 2

Таблица кодов. Мнемоника Ассемблера Z80

Дес. код	Символ	Шест. код	Ассемблер Z80
0	не использ.	00	NOP
1	не использ.	01	LD BC,NN
2	не использ.	02	LD (BC),A
3	не использ.	03	INC BC
4	не использ.	04	INC B
5	не использ.	05	DEC B
6	PRINT	06	LD B,N
7	EDIT	07	RLCA
8	курсор влево	08	EX AF,AF'
9	курсор вправо	09	ADD HL,BC
10	курсор вниз	0A	LD A,(BC)
11	курсор вверх	0B	DEC BC
12	DELETE	0C	INC C
13	ENTER	0D	DEC C
14	число	0E	LD C,N
15	не использ.	0F	RRCA
16	INK упр.	10	DJNZ DIS
17	PAPER упр.	11	LD DE,NN
18	FLASH упр.	12	LD (DE),A
19	BRIGHT упр.	13	INC DE
20	INVERSE упр.	14	INC D
21	OVER упр.	15	DEC D
22	AT упр.	16	LD D,N
23	TAB упр.	17	RLA
24	не использ.	18	JR DIS
25	не использ.	19	ADD HL,DE
26	не использ.	1A	LD A,(DE)
27	не использ.	1B	DEC DE
28	не использ.	1C	INC E
29	не использ.	1D	DEC E

Дес. код	Символ	Шест. код	Ассемблер Z80
30	не использ.	1E	LD E,N
31	не использ.	1F	RRA
32	пробел	20	JR NZ,DIS
33	!	21	LD HL,NN
34	"	22	LD (NN),HL
35	#	23	INC HL
36	$	24	INC H
37	%	25	DEC H
38	&	26	LD H,N
39	'	27	DAA
40	(	28	JR Z,DIS
41	)	29	ADD HL,HL
42	*	2A	LD HL,NN
43	+	2B	DEC HL
44	,	2C	INC L
45	-	2D	DEC L
46	.	2E	LD L,N
47	/	2F	CPL
48	0	30	JR NC,DIS
49	1	31	LD SP,NN
50	2	32	LD (NN),A
51	3	33	INC SP
52	4	34	INC (HL)
53	5	35	DEC (HL)
54	6	36	LD (HL),N
55	7	37	SCF
56	8	38	JR C,DIS
57	9	39	ADD HL,SP
58	:	3A	LD A,(NN)
59	;	3B	DEC SP
60	<	3C	INC A

Дес. код	Символ	Шест. код	Ассемблер Z80
61	=	3D	DEC A
62	>	3E	LD A,N
63	?	3F	CCF
64	@	40	LD B,B
65	A	41	LD B,C
66	B	42	LD B,D
67	C	43	LD B,E
68	D	44	LD B,H
69	E	45	LD B,L
70	F	46	LD B,(HL)
71	G	47	LD B,A
72	H	48	LD C,B
73	I	49	LD C,C
74	J	4A	LD C,D
75	K	4B	LD C,E
76	L	4C	LD C,H
77	M	4D	LD C,L
78	N	4E	LD C,(HL)
79	O	4F	LD C,A
80	P	50	LD D,B
81	Q	51	LD D,C
82	R	52	LD D,D
83	S	53	LD D,E
84	T	54	LD D,H
85	U	55	LD D,L
86	V	56	LD D,(HL)
87	W	57	LD D,A
88	X	58	LD E,B
89	Y	59	LD E,C
90	Z	5A	LD E,D
91	[	5B	LD E,E
92	/	5C	LD E,H
93	]	5D	LD E,L
94	↑	5E	LD E,(HL)
95	_	5F	LD E,A
96	фунт-стерл.	60	LD H,B
97	a	61	LD H,C
98	b	62	LD H,D
99	c	63	LD H,E
100	d	64	LD H,H

Дес. код	Символ	Шест. код	Ассемблер Z80
101	e	65	LD H,L
102	f	66	LD H,(HL)
103	g	67	LD H,A
104	h	68	LD L,B
105	i	69	LD L,C
106	j	6A	LD L,D
107	k	6B	LD L,E
108	l	6C	LD L,H
109	m	6D	LD L,L
110	n	6E	LD L,(HL)
111	o	6F	LD L,A
112	p	70	LD (HL),B
113	q	71	LD (HL),C
114	r	72	LD (HL),D
115	s	73	LD (HL),E
116	t	74	LD (HL),H
117	u	75	LD (HL),L
118	v	76	HALT
119	w	77	LD (HL),A
120	x	78	LD A,B
121	y	79	LD A,C
122	z	7A	LD A,D
123	{	7B	LD A,E
124	|	7C	LD A,H
125	}	7D	LD A,L
126	~	7E	LD A,(HL)
127	©	7F	LD A,A
128		80	ADD A,B
129		81	ADD A,C
130		82	ADD A,D
131		83	ADD A,E
132		84	ADD A,H
133		85	ADD A,L
134		86	ADD A,(HL)
135		87	ADD A,A

Дес. код	Символ	Шест. код	Ассемблер Z80
136		88	ADC A,B
137		89	ADC A,C
138		8A	ADC A,D
139		8B	ADC A,E
140		8C	ADC A,H
141		8D	ADC A,L
142		8E	ADC A,(HL)
143		8F	ADC A,A
144	(A) Графика	90	SUB B
145	(B)	91	SUB C
146	(C)	92	SUB D
147	(D)	93	SUB E
148	(E)	94	SUB H
149	(F)	95	SUB L
150	(G)	96	SUB (HL)
151	(H)	97	SUB A
152	(I)	98	SBC A,B
153	(J)	99	SBC A,C
154	(K)	9A	SBC A,D
155	(L)	9B	SBC A,E
156	(M)	9C	SBC A,H
157	(N)	9D	SBC A,L
158	(O)	9E	SBC A,(HL)
159	(P)	9F	SBC A,A
160	(Q)	A0	AND B
161	(R)	A1	AND C
162	(S)	A2	AND D
163	(T)	A3	AND E
164	(U)	A4	AND H
165	RND	A5	AND L
166	INKEY$	A6	AND (HL)
167	PI	A7	AND A
168	FN	A8	XOR B
169	POINT	A9	XOR C
170	SCREEN$	AA	XOR D

Дес. код	Символ	Шест. код	Ассемблер Z80
171	ATTR	AB	XOR E
172	AT	AC	XOR H
173	TAB	AD	XOR L
174	VAL$	AE	XOR (HL)
175	CODE	AF	XOR A
176	VAL	B0	OR B
177	LEN	B1	OR C
178	SIN	B2	OR D
179	COS	B3	OR E
180	TAN	B4	OR H
181	ASN	B5	OR L
182	ACS	B6	OR (HL)
183	ATN	B7	OR A
184	LN	B8	CP B
185	EXP	B9	CP C
186	INT	BA	CP D
187	SQR	BB	CP E
188	SGN	BC	CP H
189	ABS	BD	CP L
190	PEEK	BE	CP (HL)
191	IN	BF	CP A
192	USR	C0	RET NZ
193	STR$	C1	POP BC
194	CHR$	C2	JP NZ,NN
195	NOT	C3	JP NN
196	BIN	C4	CALL NZ,NN
197	OR	C5	PUSH BC
198	AND	C6	ADD A,N
199	<=	C7	RST 0
200	>=	C8	RET Z
201	<>	C9	RET
202	LINE	CA	JP Z,NN
203	THEN	CB,	
204	TO	CC	CALL Z,NN
205	STEP	CD	CALL NN
206	DEF FN	CE	ADC A,N
207	CAT	CF	RST 8
208	FORMAT	D0	RET NC
209	MOVE	D1	POP DE

Дес. код	Символ	Шест. код	Ассемблер Z80
210	ERASE	D2	JP NC,NN
211	OPEN#	D3	OUT (N),A
212	CLOSE#	D4	CALL NC,NN
213	MERGE	D5	PUSH DE
214	VERIFY	D6	SUB N
215	BEEP	D7	RST 16
216	CIRCLE	D8	RET C
217	INK	D9	EXX
218	PAPER	DA	JP C,NN
219	FLASH	DB	IN A,(N)
220	BRIGHT	DC	CALL C,NN
221	INVERSE	DD	Prefixes instructions using IX
222	OVER	DE	SBC A,N
223	OUT	DF	RST 24
224	LPRINT	E0	RET PO
225	LLIST	E1	POP HL
226	STOP	E2	JP PO,NN
227	READ	E3	EX (SP),HL
228	DATA	E4	CALL PO,NN
229	RESTORE	E5	PUSH HL
230	NEW	E6	AND N
231	BORDER	E7	RST 32
232	CONTINUE	E8	RET PE
233	DIM	E9	JP (HL)
234	REM	EA	JP PE,NN
235	FOR	EB	EX DE,HL
236	GO TO	EC	CALL PE,NN
237	GO SUB	ED,	
238	INPUT	EE	XOR N
239	LOAD	EF	RST 40
240	LIST	F0	RET P
241	LET	F1	POP AF
242	PAUSE	F2	JP P,NN
243	NEXT	F3	DI
244	POKE	F4	CALL P,NN

Дес. код	Символ	Шест. код	Ассемблер Z80
245	PRINT	F5	PUSH AF
246	PLOT	F6	OR N
247	RUN	F7	RST 48
248	SAVE	F8	RET M
249	RANDOMIZE	F9	LD SP,HL
250	IF	FA	JP M,NN
251	CLS	FB	EI
252	DRAW	FC	CALL M,NN
253	CLEAR	FD	Prefixes instructions using IY
254	RETURN	FE	CP N
255	COPY	FF	RST 56

Приложение 3

Сообщения операционной системы

Они появляются в нижней части экрана, если компьютер остановился при выполнении некоторого оператора Бейсика, и указывают причину, вызвавшую останов. Сообщение содержит кодовый номер или букву. Краткое сообщение помогает найти ошибочную строку и ошибочный оператор в этой строке. (Команда указывается как строка 0, оператор 1 располагается в строке первым, оператор 2 после первого или THEN и т.д.) От состояния CONTINUE зависит очень многое в сообщениях. Обычно продолжение начинается с оператора, специфицированного в предыдущем сообщении, но имеются исключения - сообщения 0,9 и D.

Код	Значение	Ситуация
0	OK (о'кей! порядок!). Успешное завершение или переход на строку с номером большим чем имеется всего. Это сообщение не меняет строки или оператора, определенного для CONTINUE	разное
1	NEXT WITHOUT FOR (NEXT без FOR). Управляющей переменной нет (не была определена в операторе FOR), но есть обычная переменная с тем же именем	NEXT
2	VARIABLE NOT FOUND (переменная не найдена). Для простой переменной выдается, если она используется без предварительного определения в операторах LET, READ или INPUT, или загружается с ленты или устанавливается в операторе FOR. Для индексируемой переменной сообщение выдается, если она не была предварительно определена в операторе DIM перед использованием или загрузкой с ленты	разное
3	SUBSCRIPT WRONG (ошибочный индекс) Индекс превышает размерность массива, либо ошибочное число задает индекс. Если индекс отрицательный или больше 65535, то выдается сообщение 8	в индексной переменной, подстроке
4	OUT OF MEMORY (вне памяти). В памяти недостаточно места для ваших действий. Вы можете освободить себе память удалив командные строки, используя DELETE, затем удалить одну или две строки программы (с целью возврата их в последствии), получить дополнительную память маневрируя оператором CLEAR	LET, INPUT, FOR, DIM, LOAD, MERGE
5	OUT OF SCREEN (вне экрана). Если INPUT оператор генерирует больше чем 23 строки в нижней полвине экрана. Также встречается с PRINT AT 22, ...	INPUT

Код	Значение	Ситуация
6	NUMBER TOO BIG (очень большое число). В результате вычислений получилось число больше 10**38	арифметические операции
7	RETURN WITHOUT GO SUB (RETURN без GO SUB). Встретилось больше операторов RETURN, чем было операторов GO SUB	RETURN
8	END OF FILE (конец файла)	операции с внешней памятью
9	STOP STATEMENT (оператор STOP). После этого сообщения CONTINUE не может повторить STOP, но может передать управление на следующий оператор	STOP
A	INVALID ARGUMENT (ошибочный аргумент).Аргумент функции не допустим в данной версии	SQR, LN, ASN, ACS, USR (со строковым аргументом)
B	INTEGER OUT OF RANGE (переполнение целого). Выдается, когда аргумент с плавающей точкой округляется к целому. Для случая массивов смотри также сообщение 3	RUN, POKE, RANDOMIZE, DIM, GO TO, GO SUB, LIST, LLIST, PAUSE, PLOT, CHR$, PEEK, USR (с числовым аргументом)
C	NONSENSE IN BASIC (выражение не БЕЙСИКА). Текст (строка) не распознается Бейсиком как допустимое выражение	VAL, VAL$
D	BREAK-CONT REPEATS. Клавиша BREAK нажата во время действия периферийной операции. Действия CONTINUE после этого оператора обычные, те что указаны в операторе. Сравните с сообщением L	LOAD, SAVE, MERGE, LLIST, LPRINT, VERIFY, COPY
E	OUT OF DATA (вне данных). Попытка выдать READ когда список данных в DATA кончился	READ
F	INVALID FILE NAME (неверное имя файла). Оператор SAVE с пустой строкой вместо имени или с именем длиннее 10 символов	SAVE
G	NO ROOM FOR LINE (нет места для строки). Недостаточно места в памяти для записи очередной строки программы	ввод строки в программу.
H	STOP IN INPUT. Некоторые введенные данные начинаются с оператора STOP, или была нажата INPUT LINE. Действия CONTINUE - обычное	INPUT
I	FOR WITHOUT NEXT (FOR без NEXT). Цикл FOR ни разу не выполнялся, не найден NEXT оператор	FOR
J	INVALID I/O DEVICE (неверное устройство ввода - вывода)	с внешними устройствами

Код	Значение	Ситуация
K	INVALID COLOUR (неверный цвет). Специфицированное число имеет неверное значение	FLASH, PAPER, BORDER, INK, BRIGHT,OVER, INVERSE, а также после передачи упр. символов.
L	BREAK INTO PROGRAM (BREAK во время выполнения программы). Нажата клавиша BREAK: это обнаруживается между двумя операторами. Строка и номер оператора в строке указывают на оператор, выполняемый перед нажатием BREAK, но CONTINUE переходит к следующему оператору	разное
M	RAMTOP NO GOOD (адрес RAMTOP не годен). Число, указанное для RAMTOP слишком велико или слишком мало	CLEAR, возможно RUN
N	STATEMENT LOST (оператор отсутствует). Переход к оператору, которого уже нет	RETURN, NEXT, CONTINUE
O	INVALID STREAM (ошибочный поток данных)	в операциях ввода-вывода
P	FN WITHOUT DEF (FN без DEF). Определяемая пользователем функция не определена в операторе DEF FN	FN
Q	PARAMETER ERROR (ошибка в параметре). Ошибочное число аргументов или один из них не того типа, какой был описан	FN
R	TAPE LOADING ERROR (ошибка загрузки с ленты). Файл на ленте найден, но не может быть считан	VERIFY, LOAD или MERGE

Приложение 4

Ключевые слова СПЕКТРУМа

Условные обозначения: аддр - адрес от 0 до 65535;
A - действительное число;
N, M, X, Y, R - целое число.

Функции

Таблица П.4.1.

Ключевое слово	Пример	Назначение
ABS	LET A = ABS - 3.5	Дает абсолютную величину числа
ACS	LET A = ACS 0.5	Арккосинус в радианах
AND		Логическое "И"
ASN	LET A = ASN 0.5	Арксинус в радианах
ATN	LET A = ATN 1.5	Арктангенс в радианах
ATTR	PRINT ATTR(10,5)	Дает число, показывающее состояние цветовых атрибутов в заданном знакоместе экрана. Если это число разложить на биты, то бит 7 старший) указывает на признак мигания. 1 - есть; 0 - нет. Бит 6 - признак яркости. Биты 3...5 - цвет фона (от 0 до 7). Биты 0...2 - цвет символа.
BIN	PRINT BIN 10011001	Перевод числа из двоичной формы в десятичную
CHR$	PRINT CHR$ N	Дает символ, код которого равен N
CODE	PRINT CODE ABCD$	Дает код первого символа строки (стринга)
COS	LET A = COS 0.5	Косинус числа
EXP	LET A = EXP 0.5	Экспонента числа
FN	PRINT FN W(X,Y...)	Вызов функции пользователя W, заданной DEF FN W. Аргумент должен быть в скобках. Если аргумента нет, скобки все равно должны быть
IN	PRINT IN X	Ввод данных с внешнего порта номер X
INKEY$	LET A$ = INKEY$	Читает клавиатуру. Выдает символ, равный тому, клавиша которого была нажата; если ничто не нажато, то выдает пустой стринг. INKEY$ не ждет нажатия. Это обходят, например, так : 10 LET A$ = INKEY$ 20 IF A$ = "" THEN GO TO 10 30 ..... продолжение работы после нажатия
INT	LET N = INT A	Выделение целой части числа (округление вниз)

Ключевое слово	Пример	Назначение
LEN	LET N = LEN ABSD$	Дает длину стринга в символах
LN	PRINT LN 2.345	Натуральный логарифм
NOT		Логическое отрицание "НЕ"
OR		Логическое "ИЛИ"
PEEK	LET N = PEEK адр	Дает содержимое ячейки памяти по ее адресу
PI	PRINT PI	Число "пи" - 3.14159265
POINT	PRINT POINT(X,Y)	Проверяет, включена или выключена точка экрана с координатами X,Y. Если включена - 1, если нет - 0. X=0...255, Y=0...175
RND	LET N = INT(5*RND)	Дает псевдослучайное число между 0 и 1
SCREEN	PRINT SCREEN$(X,Y)	Дает символ, равный тому, который находится на экране в знакоместе с координатами X,Y. Если символ не распознаваем, то выдается пустой стринг. X=0...23, Y=0...31
SGN	PRINT SGN A	Знак числа. 1 - положительное, -1 - отрицательное, 0 - если число равно нулю
SIN	LET A = SIN 2.345	Синус угла, заданного в радианах
SQR	PRINT SQR 25	Квадратный корень
STR$	PRINT STR$ 100	Перевод числа в строковую переменную (стринг)
TAN	LET A = TAN 2.345	Тангенс угла, заданного в радианах
USR адр	RANDOMIZE USR 30000	Вызывает подпрограмму в машинном коде и стартует ее с указанного адреса
USR A$	LET адр. = USR "A"	Дает адрес в области графики пользователя UDG, в котором расположено изображение символа графики пользователя, соответствующего "A"
VAL	LET A = VAL "100"	Функция, обратная STR$. Переводит стринг в соответствующее число
VAL$	PRINT VAL$ """100"""	Рассматривает свой стринговый параметр без ограничительных кавычек как новый стринг

Операторы

Таблица П.4.2.

Ключевое слово	Пример	Назначение
BEEP	BEEP 10.0,0.5	Воспроизводит в динамике компьютера звуковой сигнал продолжительностью 20 секунд и с высотой на 0.5 полутона выше, чем "ДО" первой октавы
BORDER	BORDER N	Установка цвета бордюра. N=0...7
BRIGHT	BRIGHT 1	Установка яркости. 0 - нормальная, 1 - повышенная
CAT		Распечатать каталог катриджа микродрайва

Ключевое слово	Пример	Назначение
CIRCLE	CIRCLE X,Y,R	Вычерчивание окружности с центром X,Y и радиусом R
CLEAR	CLEAR ADDR	Очищает все переменные, выполняет RESTORE и CLS, очищает стек GO SUB. Устанавливает новую верхнюю границу памяти по адресу "ADDR".
CLOSE#		Команда ИНТЕРФЕЙСА-1 (закрыть поток)
CLS		Очистка экрана
CONTINUE		Продолжить исполнение программы после прерывания
COPY		Команда ZX-принтера. Выполняет копирование экрана на бумагу
DATA		Данные для оператора READ
DEF FN	DEF FN W(X,Y,...)	Задает функцию пользователя. Имя функции W должно быть однобуквенным, а параметры - стоять в скобках. Если параметров нет - скобки все равно должны быть
DIM	DIM A(N1,N2...NK)	Задает k-мерный массив A. Устанавливает исходные значения элементов массива в нуль. Уничтожает все массивы с тем же именем
	DIM A$(N1,N2,...NK)	Задает символьный k-мерный массив. Исходное значение элементов массива устанавливается в "". Его можно рассматривать как k-1 мерный массив стрингов фиксированной длины, равной NK
DRAW	DRAW X,Y	Выполняется вычерчивание прямой линии от точки, в которой производилась печать в последний раз до точки, отстоящей от нее по горизонтали на X и по вертикали на Y
	DRAW X,Y,Z	То же самое, но между этими двумя точками вычерчивается не прямая, а дуга окружности, охватывающая угол Z, где угол измеряется в радианах
ERASE		Команда микродрайва
FLASH	FLASH 1	Признак мигания. 1 - включено, 0 - выключено
FOR	FOR i=x TO y	Составная часть оператора FOR...TO...NEXT
	FOR i=x TO y STEP z	Организует цикл с начальным значением параметра цикла x, конечным значением y и шагом z. Шаг может быть и отрицательным. В этом случае начальное значение больше, чем конечное
FORMAT		Команда микродрайва
GO SUB	GO SUB N	Переход к исполнению подпрограммы, расположенной начиная со строки N. Подпрограмма должна завершаться командой RETURN
GO TO	GO TO N	Переход к исполнению строки N

Ключевое слово	Пример	Назначение
IF X THEN Y	IF A>5 THEN GO TO N	Условный оператор. Если выражение X справедливо, то выполняется выражение после THEN. В противном случае выполняется переход к следующей строке. (Не к следующему оператору в этой строке, а именно к следующей строке! Типичная ошибка!)
INK	INK N	Устанавливает цвет печати. N=0...7
INPUT	INPUT A	Служит для ввода с клавиатуры числа во время исполнения программы. Может вводиться в символьный стринг. В этом случае оператор должен иметь форму INPUT LINE
INVERSE	INVERSE 0	Признак инверсии. 1 - включено (печать белым по черному), 0 - нормальная печать
LET	LET A=X	Оператор присваивания. Переменной A присваивается значение X
LIST	LIST	Выполняется печать листинга (текста) программы на экране. В форме LIST N листинг выдается, начиная со строки N. Если текст занимает более, чем один экран, то после заполнения экрана выдается запрос SCROLL? Если нажать клавиши BREAK или N, то распечатка прервется. Нажатие любой другой клавиши продолжит листинг.
LLIST	LLIST N	Команда ZX-принтера. То же, что и LIST, но распечатка выполняется на ZX-принтере
LOAD	LOAD "имя"	Загрузка Бейсик-программы с магнитофона. Если имя не указано (LOAD ""), то загрузится первая встреченная программа. Обратите внимание : между кавычками не должно быть пробела
	LOAD "имя"CODE M,N	Команда на загрузку блока в машинных кодах. Загружает N байтов, начиная с адреса M. Если эти параметры не указаны, то блок будет загружаться в те адреса, из которых он был когда-то выгружен
	LOAD "имя"DATA()	Загрузка блока данных
	LOAD "имя"DATA$()	Загрузка символьного массива
	LOAD "имя"SCREEN$	Загрузка экрана. То же самое, что и LOAD "имя" CODE 16384,6912. Экранная область памяти "СПЕКТРУМА" всегда начинается с 16384 и имеет 6912 байтов (6144 - графика; 768 - атрибуты)
LPRINT		Команда ZX-принтера. См. PRINT

Ключевое слово	Пример	Назначение
MERGE	MERGE "имя"	Аналогично LOAD, но при загрузке новой программы не уничтожает старую, а производит их слияние. При этом остается старая программа с добавлением таких строк из новой программы, каких в старой программе не было. Для блоков, записанных в машинных кодах, этот оператор неприменим
MOVE		Команда микродрайва
NEW		Уничтожает БЕЙСИК-программу, устанавливает в исходное положение системные переменные, очищает память до границы RAMTOP, установленной CLEAR
NEXT	см. FOR	Часть оператора FOR...NEXT. Заканчивает цикл
OPEN#		Команда Интерфейса-1 и микродрайва
OUT	OUT M,N	Выдает байт N на внешний порт M
OVER	OVER 1	Управляет условием наложения символов друг на друга при печати. 0 - новый символ полностью закрывает старый. 1 - изображение получается смешением символов
PAPER	PAPER N (N=0...7)	То же, что INK, но задает цвет фона
PAUSE	PAUSE N	Пауза в работе программы на время N/50 секунд
	PAUSE 0	Пауза до нажатия любой клавиши
PLOT	PLOT M,N	Выполняет печать точки на экране в координатах M,N. M=0...255, N=0...175.
POKE	POKE аддр,N	Засылает байт по указанному адресу
PRINT	PRINT N	Печать числа
	PRINT X	Печать результата расчета выражения X
	PRINT "ABCD.."	Печать текста, стоящего в кавычках
	PRINT A$	Печать стринга (строковой переменной). Одной командой PRINT может выполняться печать сразу нескольких переменных или чисел. При этом между ними должен стоять разделитель. Таких разделителей три. Это запятая (,), точка с запятой (;) или апостроф ('). Если стоит запятая, то печать выполняется на той же строке, но начиная с 16 колонки, т.е. в правой половине экрана. Точка с запятой означает печать в той же строке без зазора. Апостроф дает переход к новой строке без печати. Пример: PRINT A,B,C;D;E$'F$'N. Так PRINT: PRINT: PRINT эквивалентно PRINT '''.
	PRINT AT M,N	Печать в заданной позиции (строка, столбец)

Ключевое слово	Пример	Назначение
	PRINT TAB N	Печать с табуляцией. N показывает величину табуляции. Оператор PRINT может сопровождаться операторами задания цветовых атрибутов INK, PAPER, FLASH, BRIGHT, INVERSE, OVER. В этом случае они вводятся через точку с запятой. Пример: PRINT INK 7 ; PAPER 1; " VERY WELL ", FLASH 1;" "
RANDOMIZE	RANDOMIZE N	Устанавливает системную переменную SEED (см.раздел "Системные переменные". Эта переменная используется для работы генератора псевдослучайных чисел см.RND)
	RANDOMIZE	То же, что RANDOMIZE 0. N=0...65535
	RANDOMIZE USR адр.	Вызывает и стартует процедуру, записанную в машинных кодах, находящуюся по указанному адресу. RANDOMIZE USR 0 инициализирует компьтер, т.е. выполняет сброс, эквивалентный выключению питания. Сравните с NEW.
READ	READ V1,V2,V3...VM	Присваивает переменным значение, находящиеся в соответствующем операторе DATA
REM		Пустой оператор. Открывает строку комментария. Все, что записано за REM (в пределах строки) не исполняется, а только распечатывается по LIST. Это помогает впоследствии разобраться с программой
RESTORE	RESTORE N	Устанавливает указатель DATA, привязывающий DATA к соответсвующему READ, так, чтобы он указывал на DATA, расположенный в строке N или, если такой нет, к ближайшей старшей
	RESTORE	То же, что и RESTORE 0
RETURN	RETURN	Пустой оператор. Завершает подпрограмму, к которой обращались по GO SUB. Передает управление к строке, следующей за вызывающей
RUN	RUN N	Команда на исполнение программы. Все переменные, существовавшие до нее, уничтожаются и начинается исполнение со строки N. Эквивалентна CLEAR: GO TO N
	RUN	То же, что и RUN 0
SAVE	SAVE "имя"	Выполняет выгрузку БЕЙСИК-программы на магнитную ленту. Имя давать обязательно. Требует нажатия ENTER. После ENTER появляется указание "START TAPE, THEN PRESS ANY KEY". (Запустите ленту и нажмите любую клавишу)
	SAVE "имя" LINE N	Записывает программу на ленту так, что она после загрузки будет автостартовать со строки N
	SAVE "имя"CODE M,N	Запись на ленту блока машинных кодов
	SAVE "имя"DATA()	Запись числового массива

Ключевое слово	Пример	Назначение
	SAVE "имя"DATA$()	Запись на ленту символьного массива
	SAVE "имя"SCREEN$	Запись экрана. То же самое, что и SAVE "имя"CODE 16384,6912 Наличие всех параметров в команде SAVE - обязательно
STOP		Останавливает исполнение программы. Продолжить можно оператором CONTINUE
VERIFY	VERIFY ""	Аналогично LOAD, но в отличие от нее не производит загрузку в ОЗУ, а только сверяет то, что записано на ленте с тем, что находится в памяти компьютера и в случае несоответствия выдает сообщение об ошибке. Применяется для проверки качества записи. Имеет все те же формы записи, что и оператор LOAD

ПРИМЕЧАНИЕ:

Выше мы не рассматривали ряд операторов, указав, что они имеют отношение к Интерфейсу-1 или к микродрайву. Интерфейс-1 - это устройство, выполняющее одновременно три функции - контроллера локальной сети, порта RS232 и контроллера микродрайва. Выпускается той же фирмой "*Sinclair Research Ltd.*" Имеет собственное ПЗУ и служит для подключения компьютера к другим компьютерам (практически любым), к глобальной сети (через модем) и для обьединения класса "СПЕКТРУМА" в локальную сеть (до 64-х шт.). Команды Интерфейса-1, строго говоря, уже не являются операторами БЕЙСИКа как языка программирования, а командами операционной системы.

ZX-микродрайв - устройство, работающее с высокоскоростной закольцованной магнитной лентой, находящейся на сменных картриджах. Оставаясь ленточным носителем, он создает эффект работы с дисководом. Емкость картриджа порядка 100K. Одновременно можно подключать до восьми микродрайвов.

Приложение 5

Примеры программ на БЕЙСИКе

Пример 1.

Моделирование секундной стрелки часов.

Эти часы остановятся, проработав примерно 55,5 часов, что задается в операторе с номером 40. Рассматриваемый вариант часов обеспечивает 2%-ю точность или иными словами уход на полчаса в день. Возможен и более точный способ измерения времени, для этого нужно использовать системный таймер, который через каждые 20 мс увеличивает содержимое 3-х байтной переменной по адресу 23672.

```
10 REM Сначала рисуем циферблат часов
20 FOR n=1 TO 12
30 PRINT AT 10-10*COS(n/6*PI),16 +10*SIN(
n/6*PI);n
40 NEXT n
50 REM Теперь запускаем часы
60 FOR t=0 TO 200000: REM t - время в секундах
70 LET a=t/30*PI: REM a - угол поворота
сек.стрелки в радианах
80 LET sx=80*SIN a
90 LET sy=80*COS a
200 PLOT 128,88: DRAW OVER 1; sx,sy
210 PAUSE 42
220 PLOT 128,88: DRAW OVER 1; sx,sy
400 NEXT t
```

Выражение (256*256*PEEK 23674+ 256*PEEK 23673+PEEK 23672) /50 дает количество секунд, прошедших с тех пор, как компьютер был включён (вплоть до 3-х суток и 21-го часа). Модифицируем вышеприведенную программу моделирования часов:

```
10 REM Сначала рисуем циферблат часов
20 FOR n=1 TO 12
30 PRINT AT 10-10*COS(n/6*PI),16+
10*SIN(n/6*PI);n
40 NEXT n
50 DEF FN t( )= INT((256*256*
PEEK 23674+256*PEEK 23673+
PEEK 23672)/50)
100 REM Теперь запускаем часы
110 LET t1=FN t( )
120 LET a=t1/30*PI
130 LET sx=72*SIN a
140 LET sy=72*COS a
150 PLOT 131,91
160 DRAW OVER 1;sx,sy
200 LET t=FN t( )
210 IF t<=t1 THEN GO TO 200
220 PLOT 131,91
225 DRAW OVER 1;sx,sy
230 LET t1=t
240 GO TO 120
```

Эти часы обеспечивают точность 0,01% или уход на 10 секунд в день. Однако, это возможно, если Вы не использовали операторы BEEP, ввода/вывода на магнитофон или принтер.

Пример 2.

Расходящиеся спирали (SPIRALS).

Эта программа рисует расходящиеся спирали. Меняя константы в строках 10, 15 и 20, можно получать различные варианты спиралей. Константы, приведенные здесь, дают максимальный размер спирали на эк-

ране. Программа использует тригонометрические функции **SIN** и **COS**.

```
10 LET x=0.5
15 LET y=0.5
20 LET p=2.05
25 FOR z=1 TO 3
30 BORDER INT (RND*5): PAPER 6: INK INT
(RND*6): CLS
35 PRINT AT 1,2; INK INT (RND*4); FLASH 1;
"SPECTRUM"; AT 1,21; INK INT (RND*4);
FLASH 1; "SPIRALS"; FLASH 0
40 PLOT 128,88
45 FOR a=0 TO 75*PI STEP PI/p
50 DRAW x*a*SIN a, y*a*COS a
55 NEXT a
60 NEXT z
65 BEEP .5,1: BEEP .5,3: BEEP .5,7: BEEP .5,4:
BEEP .5,2; BEEP 1,1
70 GO TO 1
75 SAVE "spirals" LINE 1
```

Пример 3.

График функции Sin(x).

Программа использует тригонометрическую функцию **SIN** и оператор **PLOT**.

```
10 PRINT AT 0,0;"^"
15 PRINT AT 11,31;">"
20 PRINT AT 3,20;"F(x)=Sin(x)"
25 FOR n=0 TO 168
30 PLOT 3,n
35 NEXT n
40 FOR i=3 TO 253
45 PLOT i,83
50 PLOT i,83+80*SIN(i/128*PI)
55 NEXT i
60 STOP
```

Пример 4.

Игра "LIGHT-BULBS"

"*LIGHT-BULBS*" - это игра для двух человек. Цель игры - зажечь 6 лампочек на Вашей стороне экрана. Чтобы достичь этого необходимо поразить цель, которая появляется на верху экрана в случайном месте. Внизу экрана Вы видите пушку. Когда она перемещается слева направо, игрок слева ведет огонь, нажимая на клавишу <1>, когда же пушка перемещается с права налево, правый игрок ведет огонь с помощью клавиши <0>. Побеждает тот, кто первым зажжёт 6 лампочек.

При вводе программы будьте внимательны. <CS+9> - означает одновременное нажатие клавиш <CAPS SHIFT> и <9>. В строках 4020 - 4040 приведено описание правил игры на английском языке; в комментарии (**REM**) приведён перевод. Если Вы не сильны в английском, строки 4020, 4030 и 4040 можно не вводить.

В программе используется графика, определяемая пользователем (**UDG**).

```
2 BORDER 0: PAPER 0: CLS
6 GO SUB 5000
8 GO SUB 4000
10 BORDER 0: PAPER 0: CLS
20 FOR h=3 TO 21
30 PRINT INK 7; AT h,0; "<CS+9> H<CS+9>"; AT
h,31;"<CS+9>J<CS+9>"
40 NEXT h
50 PRINT AT 21,0; INK 0; PAPER 2;
"<CS+9>IIIIIIIIIIIIIIIIIIIIIIIIIIIIIIIII <CS+9>"
60 LET p=0: LET b=0: LET t=0
100 PRINT AT 0,0;" "
102 PRINT INK 4; FLASH 1; AT 0,14;a$
109 PRINT AT 2,t;" ": LET t= INT ( RND *24) + 4:
PRINT INK 4; AT 2,t; "<CS+9>C<CS+9>"
110 FOR f=3 TO 28: PRINT INK 4; AT 20,f-1; "
<CS+9>A<CS+9>"
120 IF INKEY$ = "1" THEN GO TO 150
130 BEEP .02,f
140 NEXT f: GO TO 1000
150 FOR g=20 TO 1 STEP -1
160 PRINT INK 4; FLASH 1; AT g,f;"<CS+9>
B<CS+9>"
170 BEEP .05,g
180 PRINT AT g,f;" "
190 NEXT g
200 IF f <> t THEN GO TO 1000
205 LET b=b+1
```

```
210 FOR a=0 TO 30

220 BEEP .01,a

230 NEXT a

240 IF b>0 THEN PRINT INK 1; AT 4,1;
"<CS+9>DE<CS+9>"; AT 5,1;"<CS+9>FG<CS+9>"

250 IF b>1 THEN PRINT INK 2; AT 7,1;
"<CS+9>DE<CS+9>"; AT 8,1;"<CS+9>FG<CS+9>"

260 IF b>2 THEN PRINT INK 3; AT 10,1;
"<CS+9> DE<CS+9>"; AT 11,1; "<CS+9> FG
<CS+9>"

270 IF b>3 THEN PRINT INK 4; AT 13,1;
"<CS+9>DE<CS+9>"; AT 14,1;"<CS+9>FG
<CS+9>"

280 IF b>4 THEN PRINT INK 5; AT 16,1;
"<CS+9>DE<CS+9>"; AT 17,1; "<CS+9>FG
<CS+9>"

290 IF b>5 THEN PRINT INK 6; AT 19,1;
"<CS+9>DE<CS+9>"; AT 20,1;"<CS+9>FG
<CS+9>"

295 IF b>5 THEN GO TO 2000

1000 PRINT AT 0,0;"                    "

1002 PRINT INK 5; FLASH 1; AT 0,14;b$

1009 PRINT AT 2,t;" "; LET t= INT ( RND *24) +
4: PRINT INK 5; AT 2,t;"<CS+9>C<CS+9>"

1010 FOR f=28 TO 3 STEP -1: PRINT INK 5; AT
20,f;"<CS+9>A<CS+9> "

1020 IF INKEY$ = "0" THEN GO TO 1060

1030 BEEP .02,f

1040 NEXT f: GO TO 100

1060 FOR g=20 TO 1 STEP -1

1070 PRINT INK 5; FLASH 1; AT g,f;
"<CS+9>B<CS+9>"

1080 BEEP .05,g

1090 PRINT AT g,f;" "

1100 NEXT g

1110 IF f <> t THEN GO TO 100

1115 LET p=p+1

1120 FOR u=0 TO 30

1130 BEEP .01,u

1140 NEXT u

1150 IF p>0 THEN PRINT INK 1; AT 4,29;
"<CS+9>DE<CS+9>"; AT 5,29;"<CS+9>FG
<CS+9>"

1160 IF p>1 THEN PRINT INK 2; AT 7,29;
"<CS+9>DE<CS+9>"; AT 8,29;"<CS+9>FG
<CS+9>"

1170 IF p>2 THEN PRINT INK 3; AT 10,29;
"<CS+9>DE<CS+9>"; AT 11,29;"<CS+9>FG
<CS+9>"

1180 IF p>3 THEN PRINT INK 4; AT 13,29;
"<CS+9>DE<CS+9>"; AT 14,29;"<CS+9>FG
<CS+9>"

1190 IF p>4 THEN PRINT INK 5; AT 16,29;
"<CS+9>DE<CS+9>"; AT 17,29;"<CS+9>FG
<CS+9>"

1200 IF p>5 THEN PRINT INK 6; AT 19,29;
"<CS+9>DE<CS+9>"; AT 20,29;"<CS+9>FG
<CS+9>"

1205 IF p>5 THEN GO TO 2000

1210 GO TO 100

2000 FOR  m=0 TO 30

2020 BEEP .050,m

2023 NEXT m

2031 BORDER 0: PAPER 0: CLS

2033 IF b>5 THEN PRINT INK 6; FLASH 1; AT
8,10;a$;"WINS."

2034 IF p>5 THEN PRINT INK 6; FLASH 1; AT
8,10;b$;"WINS."

2035 FOR b=0 TO 10

2036 FOR a=0 TO 7

2038 PRINT INK a; FLASH 1; AT 5,3;"GAME OVER"

2040 BORDER RND *7: BEEP .025,RND *55

2041 NEXT a

2042 BEEP .025,RND *55

2043 PRINT INK 6; FLASH 1; AT 10,8;
"CONGRATULATIONS"

2045 NEXT b

2060 GO TO 4050

4000 BORDER 0: PAPER 0

4005 PRINT INK 6; FLASH 1; AT 0,11;"GAME
LIGHT-BULBS"

4020 PRINT INK 4;'"This is a game for 2 people,
the object of the game is to light the 6 light
bulbs on your side of the screen. ": REM Эта иг-
ра для двух человек, цель игры - зажечь 6 лам-
почек на Вашей стороне экрана.

4030 PRINT INK 4;'"The way to do this is to hit
target wich appears at the top of the screen at a
random spot.      ": REM Чтобы достичь этого
необходимо поразить цель, которая появляет-
ся на верху экрана в случайном месте.

4040 PRINT INK 4;'"At the bottom of the screen
you will see a cannon, when this runs from left
```

```
to right, the person on the left will fire by
pressing (1), when it runs from right to left the
person on the right will fire by pressing (0).":
REM Внизу экрана Вы видите пушку. Когда
она перемещается слева направо, игрок слева
ведет огонь, нажимая на клавишу (1), когда же
пушка перемещается справа налево, правый
игрок ведет огонь с помощью клавиши (0).
4050 PRINT INK 3; FLASH 1; AT 21,1;"Press key
and INPUT PLAYERS"
4055 PRINT #1; INK 2; PAPER 6; FLASH 1;" (not
more than 7 letters) "
4056 PAUSE 0
4060 POKE 23658,8
4061 INPUT "ENTER name of RIGHT PLAYER
",a$: REM Введите имя ПРАВОГО игрока
4062 IF LEN a$>7 THEN GO TO 4061
4063 INPUT "ENTER name of LEFT PLAYER
",b$:REM Введите имя ЛЕВОГО игрока
4064 IF LEN b$>7 THEN GO TO 4063
4070 GO TO 10
5000 FOR a= USR "a" TO USR "j"+7
5010 READ user: POKE a,user
5020 NEXT a
5030 DATA 24,24,24,24,24,60,255,255
5040 DATA 24,24,24,24,24,60,60,36
5050 DATA 255,255,231,231,195,195,195,195
5060 DATA 3,15,63,127,127,255,255,255
5061 DATA 192,240,252,254,254,255,255,255
5062 DATA 255,255,255,127,127,63,15,3
5063 DATA 255,255,255,254,254,252,240,190
5065 DATA 248,0,184,0,248,0,232,0
5066 DATA 170,85,170,85,170,85,170,85
5067 DATA 31,0,29,0,31,0,23,0
5070 RETURN
```

Пример 5.

Игра "MISSION"

Клавиши <5> и <8> управляют движением пушки влево и вправо. Клавиша <0> - стрельба. При вводе программы будьте внимательны. <CS+3> - означает одновременное нажатие клавиш <CAPS SHIFT> и <3>. <CS+6> - означает одновременное нажатие клавиш <CAPS SHIFT> и <6>. <CS+8> - означает одновременное нажатие клавиш <CAPS SHIFT> и <8>. <CS+9> - означает одновременное нажатие клавиш <CAPS SHIFT> и <9> (переход в графический режим, курсор «G»).

```
10 LET s=0: LET n=1
15 LET a=1: LET b=1
20 LET c=10
25 GO TO 140
30 LET b=b+1
35 LET c=c+(INKEY$="8")-(INKEY$="5")
40 IF c=-1 THEN LET c=c+1
45 IF c=28 THEN LET c=c-1
50 IF b=30 THEN LET b=0
55 IF b=0 THEN LET a=a+2
60 PRINT AT a,b; "<CS+9><CS+6>
<CS+8><6><CS+9>"
65 IF a=15 AND b=29 THEN GO TO 140
70 PRINT AT a,b; "  "
75 PRINT AT 20,c;" <CS+9><CS+3>
<CS+8><CS+3><CS+9> "
80 IF INKEY$="0" THEN PRINT AT a,c+2;"*"
85 IF INKEY$="0" AND c=b THEN GO TO 95
90 GO TO 50
95 PRINT AT a,b+1;" * "
100 LET s=s+10
110 PRINT AT 0,0;"Score: ";s
120 PAUSE 50
130 GO TO 30
140 PRINT AT a,b; "  "
150 LET a=1: LET b=1
160 PRINT AT 0,0;"Score: ";s
170 PRINT AT 0,20; "Screen: ";n
180 IF n=10 THEN STOP
190 GO TO 30
```

Пример 6.

Игра "LANDER"

Необходиомо произвести посадку Вашего символического корабля на движущуюся в "пространстве" платформу. Управление кораблем происходит с помощью клавиш <8> и <5> (вправо-влево).

Каждая удачная посадка сигнализируется звуковым сигналом и увеличением очков. Вы по своему усмотрению можете наращивать и усложнять игру, улучшать её оформление, вводить дополнительные условия. Прервать программу можно с помощью клавиши **<BREAK>**.

```
10 LET s=0
15 CLS
20 PRINT AT 0,0;s
25 LET c=0
30 LET a=1
35 LET z=INT (RND*13)+7
40 LET b=10
45 LET a=a+1
50 LET b=b+(INKEY$="8")-(INKEY$="5")
55 LET c=c+1
60 PRINT AT c-1,b-1;" * "
65 PRINT AT
c,b;"<CS+9><CS+6><CS+8><6><CS+9>"
70 IF b=a AND c=z-1 THEN GO TO 90
75 IF c=20 THEN GO TO 95
80 PRINT AT z,a;" <CYAN><CS+9><CS+3>
<CS+3> <CS+3><CS+9><WHITE>"
85 GO TO 60
90 LET s=s+10
95 PAUSE 70
94 GO TO 15
95 PRINT AT c,b;" * "
97 PRINT AT z,a;"  "
99 GO TO 25
```

Пример 7.

Электронный карандаш.

Эта программка даёт Вам возможность рисовать не сложные графические объекты и посмотреть как работает оператор **INK**. Управление карандашом: <1> - установка цвета, <8> и <5> - вправо/влево, <7> и <6> - вверх/вниз.

```
10 LET m=0: LET n=0: LET i=0
20 LET m=m+(INKEY$="8")-(INKEY$="5")
30 LET n=n+(INKEY$="7")-(INKEY$="6")
40 LET i=i+(INKEY$="1")
50 IF i>7 THEN LET i=0
60 INK i
70 PLOT m,n
80 GO TO 20
```

Пример 8.

Программа "CONVERSIONS"

Эта полезная программа, написана Филиппом Бейлисом (*Philip Bayliss*) для **Spectrum 16K**. Она выполняет перевод 16 различных физических и метрических величин. Вы можете изменить её на свой вкус, переводить и пересчитывать "сажень" или "аршин" в метр или сантиметр и т.д.

Рассмотрено необычное применение операторов **READ** и **DATA**.

```
5 REM Screen set up
10 PRINT AT 9,0; FLASH 1; BRIGHT 1;"
CONVERSIONS  "
20 PAUSE 0: CLS
30 PRINT AT 2,3;"Enter choice (a-p)"'
40 PRINT "a) Inches -> M.M."' "b) M.M.  ->
Inches"' "c) Feet  -> Metres"' "d) Metres ->
Feet "' "e) Yards  -> Metres"' "f) Metres ->
Yards "' "g) Miles  -> K.M.  "' "h) K.M.  ->
Miles "' "i) Pints  -> Litres"' "j) Litres -> Pints
"' "k) Gals.  -> Litres"' "l) Litres -> Gals. "' "m)
Ounces -> Grams "' "n) Grams  -> Ounces"' "o)
Pound  -> K'gms "' "p) K'gms  -> Pound "'
50 INPUT c$
60 LET e$="Enter number of ": LET d$="Do you
want to repeat or calculate different conversions
(R/D)?"
100 FOR i=1 TO 16
110 READ x$,z$,w$,h
120 IF c$=x$ THEN CLS: PRINT AT 10,0;e$;z$:
INPUT y: LET a=y*h:  PRINT y;" ";z$;" = ";a;" "; w$
130 NEXT i
140 PAUSE 0: RUN 30
200 DATA "a","Inches","MM",25.4
210 DATA "b","MM","Inches",0.0393
220 DATA "c","Feet","Metres",0.305
220 DATA "c","Feet","Metres",0.305
230 DATA "d","Metres","Feet",3.281
```

```
240 DATA "e","Yards","Metres",0.914
250 DATA "f","Metres","Yards",1.094
260 DATA "g","Miles","KM",1.609
270 DATA "h","KM","Miles",0.621
280 DATA "i","Pints","Litres",0.568
290 DATA "j","Litres","Pints",1.76
300 DATA "k","Gals","Litres",4.55
310 DATA "l","Litres","Gals",0.22
320 DATA "m","Ounces","Grams",28.35
330 DATA "n","Grams","Ounces",0.035
340 DATA "o","Pounds","K'gms",0.454
350 DATA "p","K'gms","Pounds",2.205
```

Пример 9.

Цветовая гамма (72 COLOURS).

Эта программа написана Кристофером Смиттом (Christopher Smith). Запустив программу, Вы будете наблюдать на экране цвета, которые образуются из 8 цветов Спектрума и двух градаций яркости. Использован удобный и простой оконный интерфейс.

Графические символы в программе приведены в сокращенном виде (подобные сокращения, заключенные в скобки, применяются в частности в журнале "*SINCLAIR USER*"): (g5) - вводится в графическом режиме (курсор "G") "5"; (ig5) - вводится в графическом режиме (курсор "G") инверсия "5" (<CS>+<5>); (30sp) - вводится 30 пробелов в обычном режиме; (ig1:30ig3:ig2) - вводится в графическом режиме (курсор "G") инверсия "1" (<CS>+<1>), далее в инверсном режиме 30 раз "3" (<CS>+<3>) и наконец в инверсном режиме "2" (<CS>+<2>); (30gA) - вводится в графическом режиме (курсор "G") 30 раз символ "A".

```
1 REM SPECTRUM COLOUR
5 BORDER 7: INK 0: PAPER 7: BRIGHT 0: OVER 0: INVERSE 0: FLASH 0: CLS
10 LET a=BIN 01010101: LET b=BIN 10101010
20 FOR f=0 TO 15: READ x: POKE USR "a"+f,x: NEXT f
30 DATA a,b,a,b,a,b,a,b
40 DATA 255,129,129,129,129,129,129,255
45 FOR g=1 TO 20: PRINT AT g,0; "(g5)"; AT g,31; "(ig5)": NEXT g
47 FOR g=0 TO 31: PRINT AT 1,g; "(ig3)"; AT 20,g; "(g3)": NEXT g
50 FOR p=0 TO 7: FOR i=0 TO 7
60 LET a$="(30gA)"
70 PRINT AT 0,1; PAPER i; INK 0; "(gB)"; INK 0; PAPER 7; "("; p; ") = BRIGHT 0/1)..."
80 FOR a=2 TO 10: PRINT AT a,1; INK i; PAPER p; a$: NEXT a
90 FOR a=11 TO 19: PRINT AT a,1; INK i; PAPER p; BRIGHT 1; a$: NEXT a
100 PRINT AT 21,2; "Press any key to CONTINUE..."
110 FOR t=0 TO 500
120 IF INKEY$ <> "" THEN GO TO 130
125 NEXT t
130 BEEP .01,-5: PRINT AT 21,0; "(30sp)": NEXT i: NEXT p
133 FOR g=0 TO 30: NEXT g
135 FOR f=0 TO 21: PRINT AT f,0; PAPER 7; "(30sp)"; NEXT f
140 PRINT AT 10,0; ; "(ig4:30g3:g7)"; AT 11,0; "(ig5)"; AT 11,31; "(g5)"; AT 12,0; "(ig1:30ig3:ig2)"
150 PRINT AT 11,2; "Any key to RUN program again"
160 IF INKEY$= "" THEN BORDER 1: BORDER 2: BORDER 3: BORDER 4: BORDER 5: BORDER 7: PAUSE 1: GO TO 160
170 CLS: RUN
200 STOP
```

Пример 10.

Программы для начинающих

Клавиатура - музыкальный инструмент.

```
10 LET P=CODE INKEY$
20 IF P=0 THEN GO TO 10
30 BEEP .04,(P-30)/2
40 GO TO 10
```

Звук и цвет.

```
10 FOR f= TO 30
20 BEEP .1,RND*20 + 20
40 BORDER RND*7
50 NEXT f
```

Звуки.

```
10 FOR x=12 TO 36
20 BEEP .01,x
30 BEEP .01,24-x
40 NEXT x
```

Бегущая буква.

```
10 FOR i=0 TO 23
20 FOR f=0 TO 31
30 PRINT AT i,f;"G"
40 PAUSE 5
50 BEEP 0.02,20
60 PRINT AT i,f;" "
70 NEXT f
80 NEXT i
```

Увеличенный рисунок любого символа, введенного с клавиатуры (кроме "N"):

```
10 CLS
20 INPUT "Wwedite znak",A$
30 LET X=15616 + (CODE A$ -32)*8
40 LET Z=0
50 FOR I=X TO X+7
60 LET K=PEEK I
70 FOR J=7 TO 0 STEP -1
80 PRINT AT Z,J;  "■" AND (K-
2*INT(K/2))=1
90 LET K=INT(K/2)
100 NEXT J
110 LET Z=Z+1
120 NEXT I
130 PAUSE 0
140 GO TO 10
```

Геометрические узоры.

```
10 FOR a=127 TO 254 STEP 4
20 CLS
30 INK RND * 6
40 PLOT 60,40
50 DRAW 128,88,PI*(2*a-1)
60 NEXT a
```

УЗОР.

```
10 LET A$=" "
20 FOR X=1 TO 7
30 LET A$=A$+CHR$(RND*14+129)
40 NEXT X
50 INK RND*6
60 BORDER RND*7
70 FOR N=0 TO 87
80 PRINT A$
90 NEXT N
```

Приложение 6

Блок питания компьютера. Защита от помех.

Если в Вашем распоряжении компьютер промышленного изготовления, то, как правило, он снабжен блоком питания, параметры которого соответствуют требованиям, указанным в прилагаемой технической документации. Каких-то специальных мер по улучшению его характеристик принимать не следует. Единственное, что можно рекомендовать — это включить между сетью 220*В* и блоком питания помехоподавляющий фильтр типа **ФСП-20**, либо любой другой, выпускаемый промышленностью, рассчитанный на напряжение сети 220*В*. Если при работе Вашего компьютера все равно наблюдаются сбои, необходимо принять ряд мер для их устранения.

Бытовой компьютер работает в условиях сильных электромагнитных помех от бытовых электроприборов (холодильник, пылесос, электрокофемолка и т. д.). Компьютер наиболее чувствителен к помехам во время загрузки программы с магнитофона. Часто приходится в это время просто отключать холодильник.

Для защиты от помех можно принять различные меры. Это проверка и затяжка всех соединений как в квартире (контакты в счетчике, в розетках, пускателях, предохранителях), так и квартирного ввода с распредщита. Другими словами, необходимо проверить всю домашнюю электропроводку. Конечно, лучше это поручить специалисту.

Теперь несколько советов тем, кто пока не имеет блока питания и собирается изготавливать его в домашних условиях.

Самодельные компьютеры, как правило, имеют простейшие блоки питания, не защищающие от помех, распространяющихся по осветительной сети. Наиболее чувствительны к помехам линейные стабилизаторы на операционных усилителях из-за высокого петлевого усиления. Даже самая слабая помеха, наведенная на элементы такого стабилизатора, усиливается в десятки тысяч раз и не всегда конденсатор фильтра на его выходе может погасить ее.

Более помехоустойчив стабилизатор на микросхеме 142ЕН5А, обеспечивающей ток нагрузки до 3 *А*. Такой стабилизатор наиболее простой в сборке и наладке. Однако, для снижения рассеиваемой мощности и повышения к.п.д. стабилизатора разность напряжений между входом и выходом микросхемы выбирают близкой к минимально допустимой — 3 вольта. Из-за этого стабилизатор становится более чувствителен к помехам отрицательной полярности и понижению напряжения в сети, даже небольшому.

Наиболее устойчивы к импульсным помехам сети импульсные стабилизаторы, которые получают все большее распространение, благодаря высоким энергетическим показателям. Помеха способна разве что вызвать лишнее переключение регулирующего элемента, что при высокой рабочей частоте такого стабилизатора не окажет заметного влияния на выходное напряжение.

Однако, изготовление и наладка импульсного стабилизатора может вызвать затруднение у людей, недостаточно знакомых с электроникой. Для желающих все же собрать такой стабилизатор можно рекомендовать схему из журнала «*Радио*» *7, 1989* , статья

«Стабилизированный сетевой преобразователь напряжения». Следует только учесть приведенные в конце статьи рекомендации по увеличению мощности и изменению выходного напряжения стабилизатора.

Приведем схему блока питания, достаточно простую и имеющую хорошие характеристики (Рис.П.6.1).

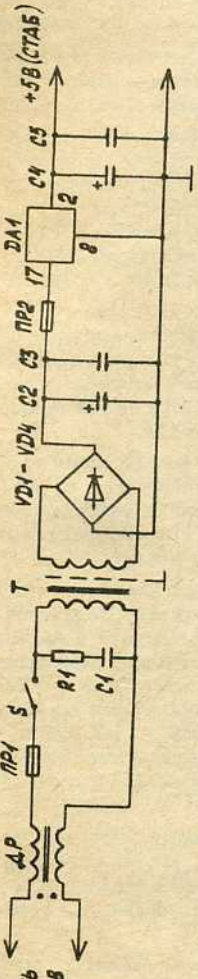


Рис.П.6.1. Схема блока питания

ПР1 – предохранитель 0,25 А
ПР2 – предохранитель 1 А
R1 – 100 Ом, 0,25 Вт
C1 – 0,1 мк, 630 В
C2 – 4700 мк, 16 В
C3.C5 – 0,15 мк
C4 – 200 мк, 6,3 В
DA1 – КР142ЕН5А
VD1-VD4 – КЦ405А-Е

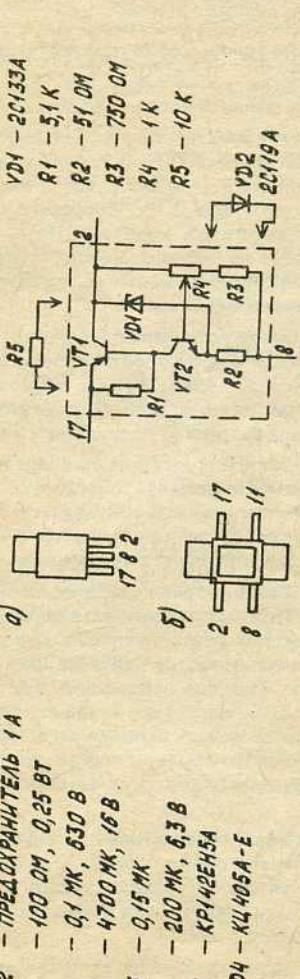


Рис.П.6.2. Выводы микросхем КР142ЕН5А (а), и К142ЕН5А (б)

Рис.П.6.3. Транзисторный стабилизатор напряжения

Схема блока питания.

Дроссель Др предназначен для защиты от высокочастотных сетевых помех и представляет собой восемь витков сложенных вместе сетевых проводов на кольце из феррита 2000НМ диаметром 16-24 *мм*. Цепь R1C1 тоже гасит сетевые помехи, а также смягчает переходные процессы при включении и выключении питания, благодаря чему увеличивается срок службы выключателя питания и ослабляются перенапряжения изоляции в трансформаторе. Трансформатор — любой, мощностью 12-15 *Вт*, с напряжением на вторичной обмотке 8-9 *В* при токе 1-1,5 *А*. Для самостоятельного изготовления трансформатора можно использовать следующие данные:

Тип сердечника — ШЛ или ШЛМ 16х20 Первичная обмотка — 2200 витков провода ПЭВ 0,15*мм* Вторичная обмотка — 85 витков провода ПЭВ 0,8*мм*

Очень полезен электростатический межобмоточный экран — один слой намотанного виток к витку провода ПЭВ 0,2, один конец которого соединен с общим проводом. Выпрямительный мост — КЦ402-405 с индексами А-Е, или четыре любых диода с допустимым током не менее 1А. Микросхему КР142ЕН5А можно заменить на 142ЕН5А (отличается только типом корпуса). Назначение выводов микросхем КР142ЕН5А и 142ЕН5А показано на рисунке. Можно также применить микросхемы КР142ЕН5В или 142ЕН5В, рассчитанные на меньший ток нагрузки — до 2 *А*. Этого, как правило, достаточно для питания компьютера.

После сборки необходимо подключить к выходу блока питания проволочный резистор мощностью 10 *Вт* и сопротивлением 5-8 *Ом* и проверить напряжения на входе (9-11*В*) и на выходе (4,9-5,1В). По возможности, следует с помощью осциллографа убедиться, что при минимальном напряжении, которое бывает в Вашей сети, на выходе стабилизатора под нагрузкой не появляется заметных пульсаций.

При отсутствии микросхемы стабилизатор можно собрать по схеме на рис.П.6.3. Такой стабилизатор имеет хотя и несколько худшие, чем микросхема, но вполне удовлетворительные параметры. К тому же минимальное падение напряжение на регулирующем транзисторе VT1 значительно ниже (менее 1*В* против 3*В* у микросхемы) и такой стабилизатор менее чувствителен к понижению напряжения в сети.

При использовании транзисторного стабилизатора сначала включите его без нагрузки и отрегулируйте подстроечным резисторов выходное напряжение. Подключите нагрузку. Если напряжение резко уменьшится, подберите транзистор VT1 с более высоким коэффициентом усиления (не менее 50 при токе 1*А*) или уменьшите номинал резистора R2 до 33-43 *Ом*. Падение напряжения при подключении нагрузки на 0,4-06*В* можно считать нормальным. Несколько улучшить параметры стабилизатора и уменьшить пульсации на его выходе можно, заменив резистор R3 на стабистор 2С119А (VD2). Если при включении питания транзисторы стабилизатора не открываются, нужно между коллектором и эмиттером транзистора VT1 включить резистор номиналом в пределах 5-15 *кОм* (R5).

При испытаниях транзисторного стабилизатора с активной нагрузкой резистором, необходимо подключать нагрузку после включения питания, так как такой стабилизатор плохо запускается под нагрузкой. Но если в качестве нагрузки подключен компьютер, стабилизатор обычно запускается нормально. Если это не так, подбирайте резистор R5 в сторону уменьшения.

В транзисторном стабилизаторе в качестве VT1 кроме указанных на схеме можно применить транзисторы КТ818, КТ837, в крайнем случае германиевые — ГТ806, П213-214 (уменьшив номинал R1 до 750 *Ом*), а в качестве VT2 — КТ608Б, КТ602БМ, КТ503. Стабилитрон VD1 можно заменить на 2С433А.

Микросхему (рис.П.6.1), или транзистор VT1 (рис.П.6.3) необходимо установить на теплоотвод площадью не менее 60 $см^3$. Если при изготовлении компьютера было произведено много замен микросхем серии 555 на серию 155 или 531, и потребляемый ток составил более 1*А*, необходимо пропорционально повысить мощность трансформатора, увеличить площадь теплоотвода и применить более мощные диоды в выпрямителе (например КД202), а также увеличить емкость конденсатора С2 до 6000 *мкФ* и поставить предохранитель ПР2 на 2 *А*. Для транзисторного стабилизатора потребуется также подобрать транзистор VT1 по коэффициенту усиления.

Подключать блок питания к компьютеру необходимо как можно более короткими проводами сечением не менее 1 $мм^2$, или даже разместить блок питания (или хотя бы только стабилизатор) в корпусе компьютера.

Приложение 7

Подключение телевизора

В литературе приводится много способов подключения телевизора к персональному компьютеру. Но одним из самых удачных является способ, описанный в журнале "Радио" N6, 1991 г. Достоинствами этого способа является то, что не требуется дополнительная согласующая плата и нет никаких дополнительных регулировок.

Необходимо лишь убедиться, что амплитуда сигналов "R", "G" и "B" составляет около 2 вольт в режиме нормальной яркости (BRIGHT 0) и около 2,5 вольт при повышенной яркости (BRIGHT 1), а также убавить контрастность в телевизоре.

Приложение 8

Лучшие игры года

Здесь мы приводим два небольших списка лучших игр по итогам прошлого года, один список из журнала "*SINCLAIR USER*" (в скобках приведено название фирмы-изготовителя), другой составлен по письмам читателей "*ZX-РЕВЮ*" (в этом списке в квадратных скобках также приведены "POKES" (адрес, данные) для облегчения прохождения некоторых игр).

TOP TEN «SINCLAIR USER»:

1. Rastan (*HIT SQUAD*)
2. Quattro Adventure (*CODEMASTERS*)
3. Fantazy World Dizzy (*CODEMASTERS*)
4. Pro Boxing Manager (*CODEMASTERS*)
5. Treasure Island Dizzy (*CODEMASTERS*)
6. Match Day 2 (*HIT SQUAD*)
7. Paperboy (*ENCORE*)
8. D.Thjmpson's Olympic Challehge (*HIT SQUAD*)
9. Pro Golf (*ATLANTIC*)
10. Indiana Jones and the Temple of Doom (*KIXX*)

«ZX-REVIEW» TOP TEN:

1. Elite (*FIREBIRD*) [46848,201]
2. Exolon (*NEWSON CONSULTANTS*) [38120,201; 31646,0; 35356,0; 42338,0; 36845,0; 38221,0]
3. Saboteur 2 (*DURELL*) [37121,0; 37122,0 - время, 61338,182; 61382,182 - жизнь]
4. Into the Eagle's Nest (*PANDORA*) [36640,0; 36641,0; 40512,0; 40513,0]
5. Saboteur 1 (*DURELL*)
6. Barbarian (*PALACE*) [51005,255]
7. Spy Hunter (*US GOLD*)
8. Down to Earth (*FIREBIRD*)
9. Army Moves (*IMAGINE*) [54597,0; 53772,0]
10. Pijamarama (*MIKRO-GEN*)

Приложение 9

Библиография

Литература

1. Ваш Спектрум. - Арзамас-16: "Гранд-Информ", 1990, 77 с.
2. Ассемблер GENS4. Отладчик MONS4. - Арзамас-16: "Гранд-Информ", 1990, 72 с.
3. Программа MONITOR. - Арзамас-16: "Гранд-Информ", 1990, 36 с.
4. Игровые программы. Выпуск 1. - Арзамас-16: "Гранд-Информ", 1990, 105 с.
5. Спектрум своими руками. - Арзамас-16: "Гранд-Информ", 1991, 67 с.
6. Руководство пользователя. /в 2-х книгах/. - Арзамас-16: "Гранд-Информ", 1991, 162 с.
7. Дисковый интерфейс TR DOS. - Арзамас-16: "Гранд-Информ", 1991, 60 с.
8. Графический редактор ART STUDIO. - Арзамас-16: "Гранд-Информ", 1991, 40 с.
9. Монитор 16К. Руководство пользователя. - Арзамас-16: "Гранд-Информ", 1992, 51 с.

Периодические издания

Журнал "ZX-РЕВЮ" (НТК "Инфорком", 107241, г.Москва, а/я 37)

Журнал "ZX+ЕЩЁ" (МП "Проспектор", 101000, г.Москва, Чистопрудный б-р, 12-А)

Информационный бюллютень "БИС" (НПК "СБИС", 167024, г.Сыктывкар, а/я 430)

Ежемесячная газета "Вокруг ZX Spectrum" (630105, г.Новосибирск-105, а/я 258)

Программное обеспечение

Нижегородское представительство фирмы **"Гранд-Информ"**.
Адрес: 603010, Нижний Новгород, а/я 78

Инфомационное сообщение

Уважаемый читатель, в 1992 году фирма "Гранд-Информ" планирует выпустить следующие сборники для пользователей Spectrum совместимых компьютеров.

- **Монитор 16К** (60 стр.) В сборнике содержится подробное описание ПЗУ-системы компьютера типа ZX SPECTRUM.
- **BASIC для ZX SPECTRUM** (250 стр., в 2-х частях) Часть 1: в основу положен перевод книги С.Викерса "Программирование на БЕЙСИКе". Часть 2: приводится описание диалектов стандартного БЕЙСИКа: ЛАЗЕР-БЕЙСИК, БЕТА-БЕЙСИК, МЕГА-БЕЙСИК и компилятора BLAST для компьютера ZX SPECTRUM.
- **Ваш домашний компьютер** (100 стр.) Рассмотрено устройство, функционирование и практическое применение популярного домашнего компьютера ZX SPECTRUM
- **Информационная система MASTERFILE** (50 стр.) Приводится подробное описание программы для создания электронных справочников MASTERFILE.
- **Справочник по программированию на ассемблере микропроцессора Z80** (40 стр.) Приводится описание команд микропроцессора Z80 (коды, мнемоника).
- **Игровые программы. Как их делают ?** (70 стр.) Приводится описание и листинги различных игровых программ.

Если Вы желаете приобрести эти книги, то просим выслать нам конверт с обратным адресом и вскоре Вы получите каталог и бланк заказа.

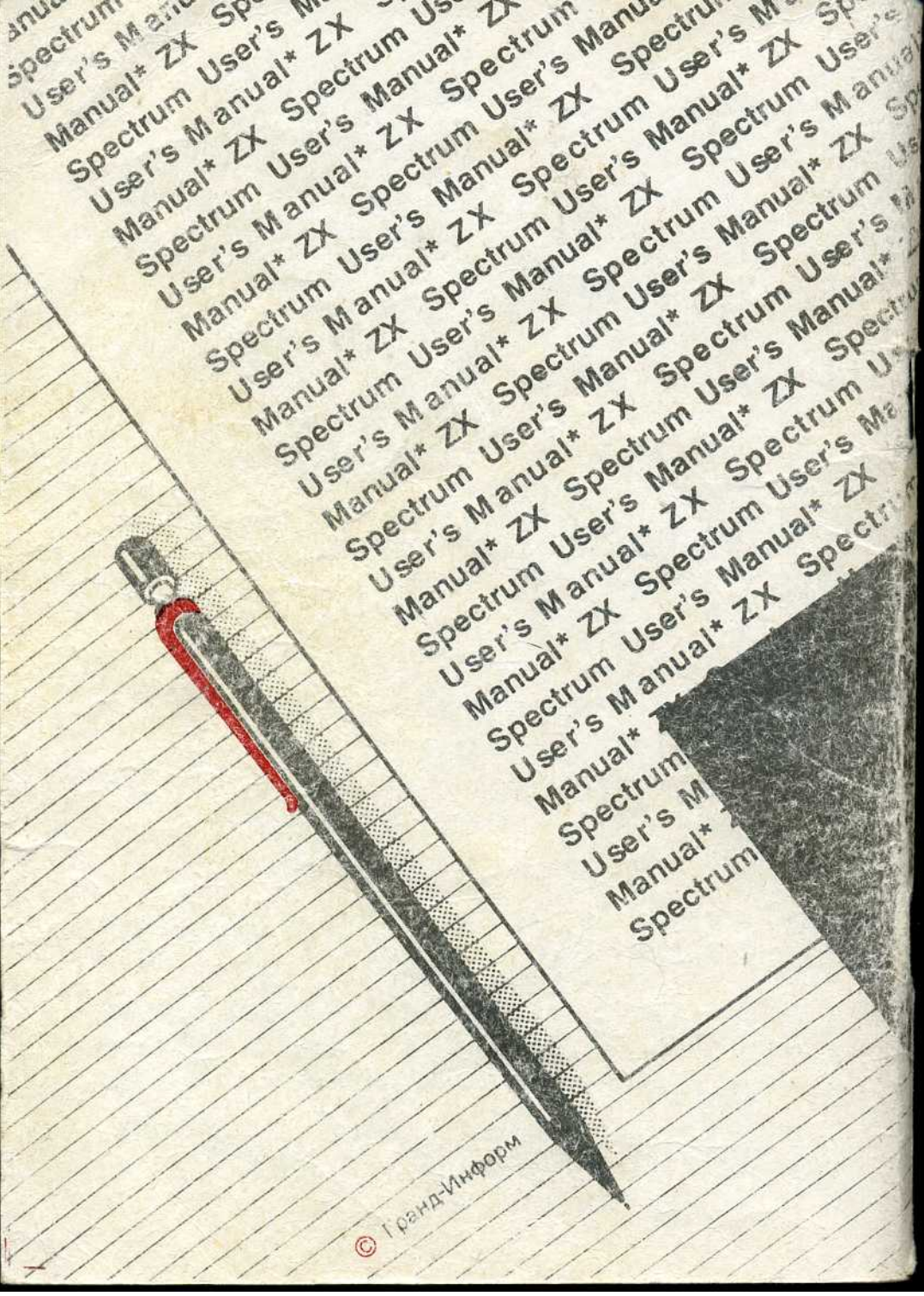
Внимание книготорговых организаций. Мы готовы рассмотреть Ваши предложения по распространению издаваемых нами книг.

Наш адрес: 607200, Нижегородская область, г.Арзамас-16, а/я 597

Программное обеспечение для Вашего Спектрума

Нижегородское представительство фирмы "Гранд-Информ" приступает к распространению программного обеспечения для Спектрум совместимых компьютеров. Широкий выбор игровых, системных, прикладных программ на дискетах и магнитофонных кассетах.

Банк программ пополняется еженедельно. По вопросам приобретения каталога обращаться по адресу: **603010, Нижний Новгород, а/я 78.**


Spectrum User's Manual* ZX
© Гранд-Информ