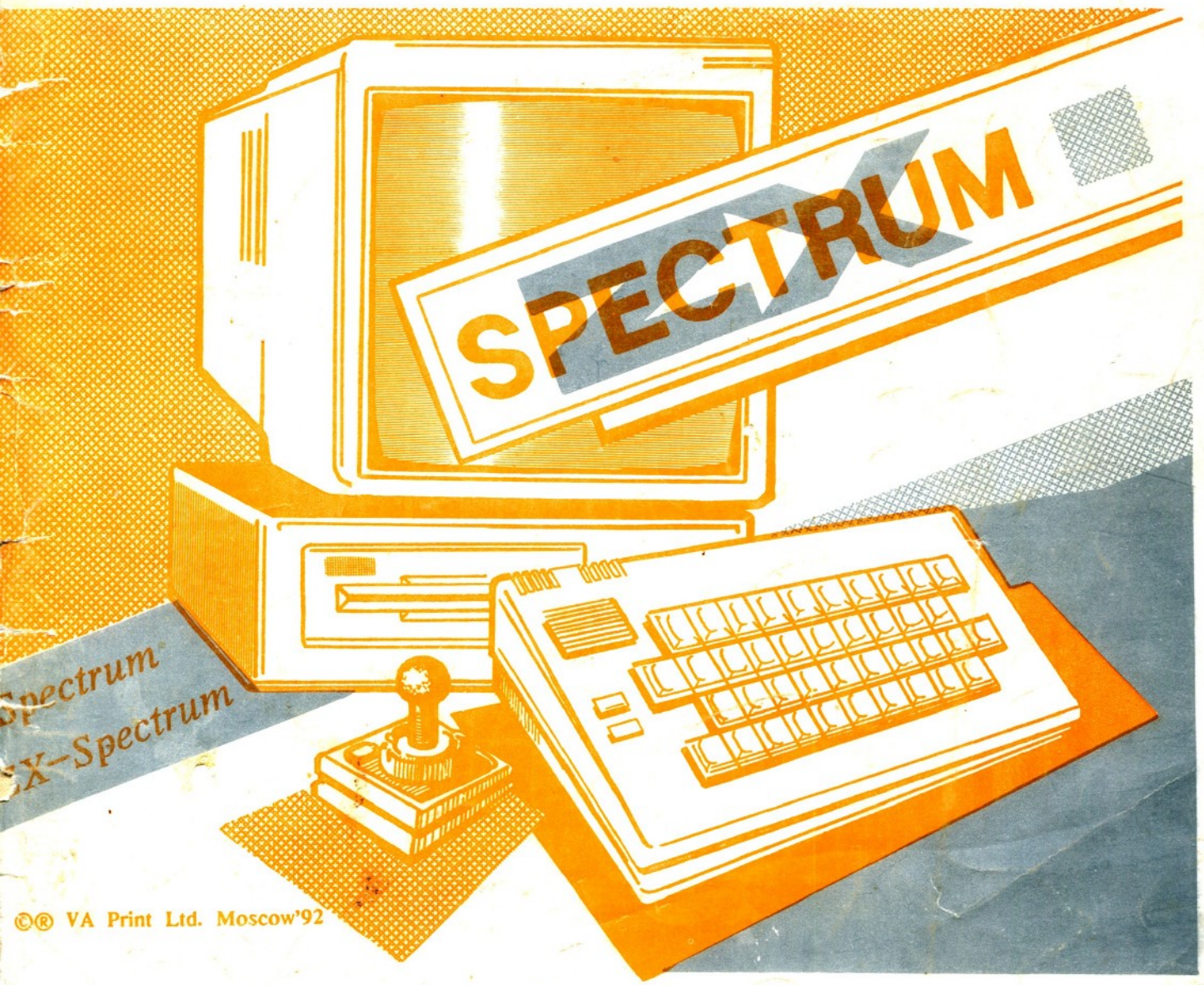
USER'S MANUAL

ZX–Spectrum

Sinclair

СПРАВОЧНИК

ПОЛЬЗОВАТЕЛЯМ



Spectrum

ZX–Spectrum

Уважаемые пользователи!

Настоящий сборник представляет собой справочное пособие для пользователей ZX SPECTRUM-совместимых персональных компьютеров и содержит в себе практически полное описание элементов программирования на языках Бейсик и Ассемблер.

Большое место занимает описание способов работы с информацией на гибких магнитных дисках, в том числе с большим набором сервисных программ (таких как копировщики, графический и текстовый редакторы, DISK DOCTOR и др.)

Приведены также принципы работы на компьютере SINCLAIR 128K.

Сборник будет крайне полезен как начинающему, так и опытному обладателю домашнего персонального компьютера.

ОГЛАВЛЕНИЕ

1.1. Немного истории

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, 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 Марлз Синклер родился 30 июля 1940 года. Еще школьником он начал печатать неплохие статьи в журнале "Практическое радио". Окончив школу, не стал поступать в университет, а был принят в этот журнал в качестве помощника зам. редактора, затем работал в издательстве, а в 1961 году зарегистрировал свою первую компанию "Синклер радионикс". Во всех разработках ставил перед собой 2 сверхзадачи: минимальные габариты и минимальная цена. Успех Синклера всегда основывался на том, что он со своим товаром был первым, причем часто ориентировался на рынок, который еще не существовал.

В 1979 г. Фирма "Коммодор" выпустила свой первый бытовой компьютер "ПЕТ" ценой 700 фунтов. Газета "Файнэншл таймс" тогда предсказывала, что цены на персональные компьютеры опустятся ниже 100 ф. ст. не ранее, чем через 5 лет, а Синклер уже через полгода выпустил ZX-80 ценой 99 фунтов.

Резкому снижению цены содействовала идея использования телевизора в качестве дисплея, а бытового магнитофона в качестве внешней памяти.

ZX-80 сломил расхожее мнение об ЭВМ, как о чем-то доступном лишь для избранных. В первые 8 месяцев было продано 20 тыс. компьютеров, и в марте 1981 г. была выпущена новая модель ZX-81 ценой 69 фунтов, а еще через несколько месяцев и принтер к нему. В эти дни американская фирма "Таймекс" купила право на производство всех разработок Синклера как сделанных, так и тех, которые будут выпущены впредь. Фирма "Митцуи" купила исключительное право на распространении ZX-81 в Японии. Решительным рывком вперед стал договор с британской книготорговой сетью о реализации компьютеров по их торговым каналам. За один год товарооборот фирмы вырос с 4,6 млн. ф. ст. до 30 млн. ф. ст., а Синклер уже готовил новую модель — "Спектрум" (март 1982 г.). Были разработаны две версии — 16K и 48K. Эта машина сильно отличалась от своих предшественников, и ее популярность превзошла все ожидания. "Спектрумы" 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 динамическую графику и для этого компьютера стали выпускать 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 "Атари", "Коммодор" и "Амстрад" выпустили компьютеры, превосходящие "Спектрум 48", рынок уже был смещен в пользу Синклера, что продолжает чувствоваться и по сей день, а сам "Спектрум" уже выпускался более чем в 30 странах мира.

В 1984 году Синклер выпустил модель "Спектрум+", отличавшуюся усовершенствованной клавиатурой, а в конце 1985 года "Спектрум+128" ("Дерби"), имеющий 128K оперативной памяти и 32K ПЗУ. Кроме этого, новая модель имела звуковой процессор.

В 1986 году компания SINCLAIR RESEARCH LIMITED была вынуждена под давлением финансовых и других трудностей продать все права на производство "Спектрум"-совместимых моделей французской фирме "Амстрад". Проблемы были связаны с не оправдавшей себя 32-разрядной модельюю "SINCLAIR-QL". Она задумывалась как дешевая альтернатива американским IBM PC, но в ее концепцию был заложен ряд просчетов (например, использование в качестве внешней памяти микродрайвов вместо дисководов). Получилось так, что бытовым компьютером эта машина не стала по цене, а профессиональным — по своему аппаратному обеспечению. К тому же фирмы, выпускающие программы, не поддержали эту в общем-то замечательную модель, опередившую идею "Амиги-500" и "Атари-520 СТ" как минимум на три года.

Положение компании усугублялось и недостаточной практической хваткой К. Синклера. Замечательный инженер, он так и не сумел стать бизнесменом. В самые напряженные дни штат его фирмы не превышал 12 человек (в том числе и представительство в Бостоне, США), а доход от продажи одного компьютера не превышал 1 ф. ст.

Продав все права на производство и реализацию своих изделий, Синклер оставил за собой исследовательскую лабораторию в Кембридже.

Последующие модели "ZX SPECTRUM+2" (1986) — со встроенным магнитофоном и "ZX SPECTRUM+3" (1987) — со встроенным дисководом, выпускались уже фирмой "Амстрад". Поэтому они так похожи по внешнему виду на компьютер "Амстрад-6128". Основным их преимуществом является полноценная клавиатура, в то время как встроенные магнитофон и дисковод воспринимаются скорее как "нагрузка", непропорционально увеличивающие цену. Особенно если принять во внимание нестандартный диаметр дискет 3,0 дюйма, малую их емкость (180K) и практическую сложность переноса имеющихся кассетных версий программ на диск, граничащую с нецелесообразностью.

С 1986 г. Фирма "Таймекс" на своих заводах в Португалии начала выпускать для Европы компьютер "Таймекс-2048", практически полностью совместимый со "Спектрумом", но имеющий ряд преимуществ: улучшенную клавиатуру, встроенный порт манипулятора "джойстик", светоиндикатор и выключатель питания, две экранных области памяти, режим расширенной цветной графики.

В заключении этого вступления упомянем только о перспективной разработанной модели, планировавшейся к выпуску в 1987 г. В основу "Суперспектрума" ("Локки") был положен процессор Z-80H, который может работать с частотой 7 МГц. При такой скорости удается организовать и обслужить два банка памяти по 64K и экран емкостью более 51K. Он имел разрешающую способность 192x256 с возможностью одновременного воспроизведения 64-х цветов (для каждой точки). Эта машина была программно совместима со "Спектрумом", стоила менее 200 фунтов и была бы серьезным конкурентом для "Амиги". Но фирма "Амстрад", пользуясь своими правами, опасаясь конкуренции для своих машин, отказала в разрешении на его производство.

1.2. Что это за система?

Итак, у вас появился персональный компьютер SINCLAIR ZX SPECTRUM, скорее всего сделанный вашими собственными руками, но при этом совместимый с оригинальным компьютером производства английской фирмы SINCLAIR RESEARCH LTD.

Компьютер построен на высокоскоростном 8-разрядном процессоре Z80 фирмы ZILOG с тактовой частотой 3,5 МГц, что позволило получить производительность на уровне 950 000 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.

Система команд процессора Z80 содержит все команды процессора, применяемого в машинах радио РК-86 и Микроша, и плюс к ним еще в два раза больше своих.

Контроллер дисплея позволяет получать на экране обычного цветного телевизора сложные графические изображения, состоящие из 256*192 точек при 16 цветах, а также текстовые изображения в форматах 32*24, 42*24, 51*24 и 64*24 символа. При этом операционная система использует самый крупный формат текста — 32*24, 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 MSX, нет необходимости — все режимы можно использовать одновременно.

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

Имеется возможность подключения светового пера, манипулятора "мышь" (все модели с разъемом INTERFACE 2) и графического принтера (только модели с системным разъемом).

При включении компьютера немедленно активируется тестовая система, проверяющая исправность компьютера и пытающаяся при обнаружении неисправностей все же обеспечить нормальное его функционирование. Она настолько мощна, что позволяет нормально работать при на 3/4 не работающем ОЗУ компьютера и других серьезных неисправностях, что значительно повышает надежность системы.

Через 1,5 секунды после включения тестовая система завершит свою работу и компьютер будет полностью готов к действию — операционная система будет активирована и готова к исполнению команд (в том числе команд встроенного бейсика). При этом не требуется загрузка каких-либо программ откуда-либо извне. Пуск

машины в работу не сложнее, чем включение обычного калькулятора!

Технические данные:

Объем ОЗУ	48 кбайт (49151 байт)
Объем ПЗУ	16 кбайт (16384 байт)
Количество внешних портов	48000 (на системной магистрали)
Скорость записи на магнитофон	1500 бод (около 180 байт/с)
Емкость кассеты магнитофона	1 Мбайт (МК-90)
Скорость вычислений	950 тыс. оп./с
Формат экрана	256*192 точки
Количество цветов	16
Количество градаций яркости	16
Режим мерцания	есть, 8*8 точек
Цветов бордюра (рамки)	8
Таймер	кварцованный, дискретность 1/50 с

1.3. Встроенный бейсик

Встроенный в операционную систему вашего компьютера интерпретатор языка бейсик (BASIC) 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. Числа, меньше данного, считаются машинным нулем. Максимальное значение положительного числа — 10 в степени 39 минус единица и обрезанное до 8 значащих цифр. Числа, больше данного, не могут быть обработаны бейсиком, о чем выдается соответствующее сообщение.

— 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.

В данном сборнике приведены операторы языка, сообщения интерпретатора и системные переменные.

1.4. 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 увидите, 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.

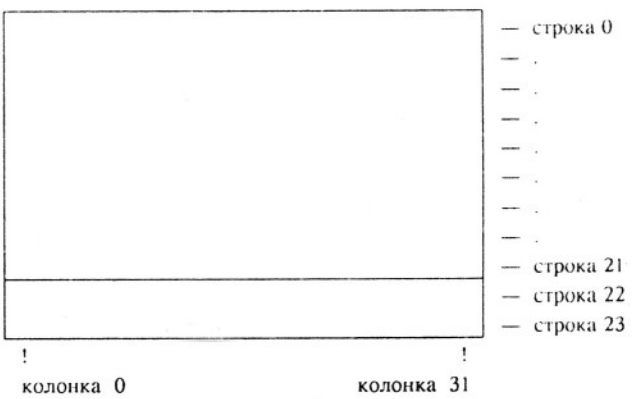
После включения компьютера вы не увидите никакого курсора, а только надпись внизу экрана

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 команду 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>. При этом все команды находятся в командной строке. Что такое эта командная строка и где она располагается? Для понимания этого рассмотрим формат экрана, используемый операционной системой:



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 вышеописанного, 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 нашу любимую клавишу P, и сразу появится наш старый знакомый 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 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, 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.

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 клвиш на экране появятся соответственно большие буквы или специальные символы.

При одновременном нажатии обоих 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 обоих 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. В вводимый текст программы интеллектуальный контроллер автоматически вставляет пробелы для улучшения читаемости программы (обратите внимание, вы вводили в нашем примере 10PRINT12, а на экране увидели 10 PRINT 12).

1.5. 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 ''SINCLAIR RESEARCH'' с находящимся внутри нее интерпретатором бейсика занимает младшие 16 килобайт адресного пространства (из полных 64 килобайта) вашего компьютера. Это и есть 16K ПЗУ. Вся остальная память машины — 48K 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.

235 байт занимают стеки и UDG. UDG — это таблица знакогенератора пользователя, в которой вы можете закодировать любые нужные вам знаки (например, греческие буквы, символы шахматных фигур и т.д.), всего до 20 разных знаков.

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.

2.1. Язык программирования бейсик

Все числа в системе могут иметь точность 9 или 10 знаков. Наибольшее число 10**38, а наименьшее положительное число 4*10**(-39). Числа имеют внутреннее представление как числа с плавающей (двоичной) точкой, с выделением одного байта на показатель степени 'E' (экспоненты) в интервале от 1 до 255, и четырех байтов на мантиссу 'M' в интервале от 0.5 до 1 (M^=1). Это представляется числом M*2**E(-128).

Поскольку 1/2<=M<1, старший бит мантиссы всегда 1. Следовательно, мы можем заменить его на бит, обозначающий знак: 0 — для положительного числа и 1 — для отрицательного.

Наименьшее целое имеет специальное представление, в котором первый байт 0, второй байт знака (0 и FFH), а третий и четвертый — само число в дополнительном коде (младшие значащие цифры в первом байте).

Числовые переменные имеют имя произвольной длины, начинающееся с буквы и продолжающееся буквами или цифрами. Пробелы и символы управления цветом игнорируются и все буквы преобразуются к минимально упакованному виду.

Управляющие переменные для FOR-NEXT циклов имеют имена длиной в одну букву. Числовые массивы имеют имена длиной в одну букву, которая может быть такой же, как имя скалярной переменной. Эти массивы могут иметь произвольное количество измерений и произвольный размер. Начальный индекс всегда 1. Строки символов более гибкие в своей длине. Имя строковой переменной в отличие от простой переменной заканчивается символом доллара ($).

Строковые массивы также могут иметь произвольное количество измерений и размер. Их имена представляют собой одну букву и следующий за ней символ $, но не могут совпадать с именем простой строки символов.

Все строки в массивах имеют фиксированную длину, которая определяется числом, задающим последнюю размерность в операторе DIM. Начальный индекс 1.

Подстрока от строки может быть получена как сечение. Сечение может быть:

А) пустым;

Б) числовым выражением;

В) некоторым 'числовым выражением' 'то' другим 'числовым выражением' и использоваться в:

*) строковых выражениях (сечениях);

**) строковых массивах переменных

(индекс 1, индекс 2,..., индекс N, сечение)

или, что тоже самое

(индекс 1, индекс 2,..., индекс N) (сечение).

В случае *), строка выражения имеет значение S$. Если сечение массива пусто, то S$ считается подстрокой от самой себя. Если сечение представляется числовым выражением со значением 'M', то результатом будет M-ый символ от S$ (подстрока, длиной 1)

Если сечение представлено в форме В) и первое числовое выражение имеет значение 'M' (умалчиваемое значение 1), а второе 'N' (умалчиваемое значение S$), и если 1<=M<=N<= чем длина S$, то результатом будет подстрока от S$ с M-ым начальным символом и N-ым конечным.

Если 0<=N<M, то результатом будет пустая строка. В любом другом случае выдается сообщение об ошибке '3'.

Сечение выполняется перед функцией или операцией, которая осуществляется, если скобками не предписано сделать иначе. Подстрока может назначаться (смотри оператор LET). Если часть строки записывается в строковой литерал, она должна удваиваться.

2.1.1. Функции бейсика

имя функции	тип аргумента	действие (возвращаемое значение)
ABS	число	абсолютное значение
ACS	число	арккосинус в радианах. Выдает сообщение об ошибке A, если H не лежит в интервале от -1 до 1.
AND	логическая операция. правый операнд всегда число. Слева может быть: — число, тогда—> A AND B = Г A, если B<>0; L B, если B=0 — строка, тогда—> A$ AND B = Г A$, если B<>0; L "", если B=0	
ASN	число	арксинус в радианах. Выдает сообщение A, если X не лежит в интервале от -1 до 1.
ATN	число	арктангенс в радианах.
ATTR	два числовых аргумента X и Y, заключаемые в скобки	число, двоичный код которого, представляет собой атрибуты Y-ой позиции X-ой строки экрана. Бит 7 (старший) равен 1 для мерцающего поля, и 0 для немерцающего. Биты с 5 по 3-цвет фона биты с 2 по 1 — цвет закрашивания. Выдает сообщение B, если 0<=X<=23 и 0<=X<=31.
BIN		Это необычная функция. За BIN записывается последовательность нулей и единиц, представляющая собой двоичное представление числа, которое записывается в память.
CHR$	число	символ, чей код представим числом X, округленным к ближайшему целому.
CODE	строка символов	код первого символа в строке X (или 0, если X — пустая строка).
COS	число в радианах	косинус X
EXP	число	E в степени X
FN		FN с последующим именем, определенной пользователем функции (см. DEF). Аргументы должны заключаться в скобки. даже, если нет аргументов, скобки все равно должны записываться.
IN	число	осуществляется ввод на уровне микропроцессора из порта X (0<=X<=FFFFH). Загружается пара регистров BC и выполняется команда ассемблера IN A(C).
INKEY$	нет	чтение с клавиатуры. Возвращает символ введенный с клавиатуры (в режиме [L] или [C], если было действительное нажатие клавиши, или пустую строку в противном случае.
INT	число	округление к ближайшему меньшему целому.
LEN	строка символов	длина строки
LN	число	натуральный логарифм. Выдает сообщение A, если X<=0.

имя функции	тип аргумента	действие (возвращаемое значение)
NOT	число	0, если X<>0, 1, если X=0. Операция имеет четвертый приоритет.
OR	логическая операция. оба операнда числа.	A OR B = Г 1, если B<>0 / L A, если B=0 операция имеет второй приоритет.
PEEK	число	значение байта в памяти по адресу X, округленному к ближайшему целому.
PI	нет	число пи (3.14159265...)
POINT	два числовых аргумента X и Y, заключенных в скобки	1, если точка экрана с координатами (X,,Y) закрашена. 0, если эта точка имеет цвет фона. Выдает сообщение B, если не выполняются условия 0<=X<=255 и 0<=Y<=175.
RND	нет	очередное псевдослучайное число из последовательности, получаемой возведением в 75 степень модуля числа 65537, вычитанием 1 и делением на 65536. Число лежит в интервале 0<=Y<=1.
SCREEN$	два числовых аргумента X и Y, заключенных в скобки	символ (обычный или инверсный), который появляется на экране в строке X, позиции Y. Дает пустую строку, если символ не опознан.
SGN	число	-1, если X<0; 0, если X=0; 1, если X>0
SIN	число в радианах	синус X
SQR	число	корень квадратный. Выдает сообщение A, если X<0.
STR$	число	строка символов, которая должна быть отображена, если X выводится.
USR	число	вызывает подпрограмму в машинных кодах, начальный адрес которой X. При возврате результатом будет содержимое регистровой пары BC.
USR	строка символов	адрес группы байтов, задающих определенный пользователем символ для закрепления его за X.
VAL	строка символов	вычисление X как числового выражения. выдает сообщение C, если X содержит синтаксические ошибки или дает строковое (нечисловое) значение. Возможны и другие ошибки.
VAL$	строка символов	вычисляет X как строковое выражение. выдает сообщение C, если X содержит синтаксическую ошибку или дает нестроковое (числовое) значение.

2.1.2. Операции бейсика

префиксные:

- число => отрицательное значение

инфиксные (двухоперандовые):

+	сложение для чисел, конкатенация для строк	
-	вычитание	
*	умножение	
/	деление	
**	возведение в степень (стрелка вверх). Сообщение B, если левый операнд отрицательный.	
=	равенство	Г
>	больше	: оба операнда должны быть
		: одного
<	меньше	: типа. Результат равен 1,
		: если
>=	больше или равно	: сравнение истинно и равно
<=	меньше или равно	: 0, если нет.
<>	не равно	L

Функции и операции имеют следующий приоритет:

индексация и сечения	— 12
все функции за исключением:	
NOT и префиксного минуса	— 11
возведение в степень	— 10
префиксный минус	— 9
*,/	— 8
+,- (вычитание)	— 6
=,>,<,<=,>=,<>	— 5
NOT	— 4
AND	— 3
OR	— 2

2.1.3. Операторы бейсика

Принятые обозначения:

A — одна буква;

V — переменная;

X, Y, Z — числовые выражения;

M, N — числовые выражения, которые округляются к ближайшему целому;

E — некоторое выражение;

F — выражение, имеющее строковое значение;

S — последовательность операторов, разделенных двоеточием ':'

C — последовательность символов управления цветом.

Каждый заканчивается ',' или ';'. Цветовой символ имеет форму операндов:

PAPER, INK, FLASH, BRIGHT, INVERSE или OVER.

Текст произвольного выражения может располагаться в любом месте строки (за исключением номера строки, который должен размещаться в начале строки).

Все операторы, кроме INPUT, DEF и DATA могут использоваться и как команды и в программах.

Команда или строка программы может содержать несколько операторов, разделенных двоеточием ':'.

Нет ограничений на положение оператора в строке, хотя есть некоторые ограничения в IF и REM.

Все операторы языка сведены в следующую таблицу:

оператор	действие оператора
BEEP X,Y	воспроизводит звук длительностью X сек. и высотой Y полутонов вверх от основного тона ДО (или вниз, если Y отрицательное).
BORDER M	устанавливает цвет рамки (бордюра) экрана. Выдает сообщение об ошибке K, если 0> M >Y.
BRIGHT M	устанавливает яркость выводимого символа: 0 — для обычной яркости; 1 — для повышенной яркости; 8 — сохраняет существующую яркость.
CAT	без MICRODRIVE не работает.
CIRCLE X,Y,Z	изображает дугу или окружность с центром в точке с координатами (X,Y) и радиусом Z.
CLEAR	уничтожает все переменные и очищает занимаемую ими память. Выполняет RESTORE и CLS, устанавливает PLOT позицию в нижнюю левую точку экрана и очищает GO SUB стек.
CLEAR N	подобно CLEAR, но дополнительно изменяет системную переменную RAMTOP на 'N' и задает новый GO SUB стек.
CLOSE#	без MICRODRIVE не работает.

оператор	действие оператора
CLS	(CLEAR SCREEN) очищает файл экрана.
CONTINUE	продолжает выполнение программы, начатой ранее и остановленной с сообщением, отличным от 0. Если было сообщение 9 или L, то выполнение продолжается со следующего оператора, в других случаях с того оператора, где случилась ошибка. Если сообщение возникло в командной строке, то CONTINUE вызовет попытку повторить командную строку и перейдет в цикл, если было сообщение 0:1, дает сообщение 0, если было 0:2, или дает сообщение N, если было 0:3 или более. В качестве CONTINUE используется ключевое слово CONT на клавиатуре.
COPY	пересылает копию 22 строк экрана на принтер, если он подключен. Помните, что по COPY нельзя распечатать находящийся на экране автоматический листинг. Выдает сообщение D, если нажать клавишу BREAK.
DATA E1,E2,E3,...	часть списка данных. Должна располагаться в программе.
DEF FN A(A1,A2,.. ..,AK) = E	определяемая пользователем функция. Должна располагаться в программе. A,A1,A2 и т.д. Единственные буквы или буквы и $ для строковых аргументов, значений. Используется форма DEF FNA(), если нет аргументов.
DELETE F	без MICRODRIVE не работает.
DIM A (N1,N2,...,NK)	уничтожает массив с именем 'A' и устанавливает числовой массив 'A' с 'K' измерениями и присваивает всем его элементам значение 0.
DIM A$ (N1,N2,...,NK)	уничтожает массив или строку с именем 'A$' и устанавливает символьный: массив с 'K' измерениями и присваивает всем его элементам значение '' ''. Массив может быть представлен как массив строк фиксированной длины NK, с K-1 размерностью. Сообщение 4 выдается, если недостаточно места для размещения массива. Массив не определен до его описания в операторе DIM.
DRAW X,Y	то же самое, что и DRAW X,Y,0. Чертит прямую линию.
DRAW X,Y,Z	изображает линию от текущей графической позиции в точку с приращениями X,Y по дуге в Z радиан. Выдает сообщение B при выходе за пределы экрана.
ERAZE	без MICRODRIVE не работает.
FLASH N	определяет: будет ли символ мерцающим или с постоянным свечением. N=0 для постоянного свечения, N=1 — для мерцания, N=8 — для сохранения предыдущего состояния.
FOR A=X TO Y FOR A=X TO Y STEP 1 FOR A=X TO Y STEP Z	уничтожает скалярную переменную 'A' и устанавливает управляющую переменную 'X', предел 'Y', шаг приращения 'Z', зацикливает адрес, указанный в утверждении после FOR оператора. Проверяет, если начальное значение больше (если STEP>=0) или меньше (если STEP<0), чем предел, то происходит переход к утверждению NEXT A или выдача сообщения 1, если нет (см. NEXT). Сообщение 4 выдается, если недостаточно места для размещения управляющей переменной.
FORMAT F	без MICRODRIVE не работает.
GO SUB N	проталкивает строку с оператором GO SUB в стек для использования затем как GO TO N. Выдается сообщение: 4, если не все подпрограммы завершились с RETURN.
GO TO N	продолжает выполнение программы со строки 'N'. Если 'N' опущено, то с первой строки после этой.
IF X THEN S	если 'X' истинно (не равно 0), то выполняется 'S'. 'S' включает все операторы до конца строки. Форма 'IF X THEN номер строки' не допустима.
INK N	устанавливает цвет закрашивания (т.е. цвет, которым будут изображаться символы на цвете фона). 'N' в интервале от 0 до 7 указывает цвет. N=8 — оставить цвет без изменений, N=9 — увеличение контраста. Выдает сообщение K, если 'N' не лежит в интервале от 0 до 9.
INPUT...	где '...' есть последовательность вводимых символов, разделяемых как в операторе PRINT запятыми, точками с запятой или апострофами. Вводимыми символами могут быть: а) некоторый PRINT-символ, начинающийся не с буквы; б) имя переменной; в) строка имен переменных строкового типа. PRINT-символы в случае а) представляются также, как и в операторе PRINT, за исключением того, что они все выводятся в нижнюю часть экрана в случае б) компьютер останавливается и ждет ввода некоторого выражения с клавиатуры, значение которого будет присвоено переменной. Ввод осуществляется обычным образом, а синтаксические ошибки выдаются мерцающим знаком вопроса [?]. Для строкового выражения вводной буфер устанавливается для размещения двух таких строк (который при необходимости может быть увеличен). Если первый вводимый символ STOP, то программа останавливается с сообщением H. Случай в) подобен случаю б) с той лишь разницей, что вводимая информация представляет собой строковый литерал неограниченной длины, и STOP в этом случае не сработает. Для останова вы должны нажать клавишу 'курсор вниз'.
INVERSE N	символ управления инверсией выводимого символа. Если N=0, символ выводится в обычном виде с прорисовкой цвета закрашивания (INK) на цвете фона (PAPER). Если N=1, то цветовое решение изображения символа меняется на обратное. Смотри приложение B. Выдает сообщение K, если 'N' не 0 или 1.
LET V=E	присваивает значение 'E' переменной 'V'. Ключевое слово LET не может быть опущено. Скалярная переменная не определена, пока не встретится в операторах LET, READ или INPUT. Если 'V' индексируемая строковая переменная или сечение строкового массива (подстрока), то присваивание осуществляется с усечением справа или дополнением пробелами до фиксированной длины.
LIST	то же, что и LIST 0.
LIST N	записывает текст программы в верхнюю часть экрана, начиная с первой строки, меньшей, чем 'N', и делает 'N' текущей строкой.
LLIST	то же, что и LLIST 0
LLIST N	подобно LIST, но вывод осуществляется на принтер.
LOAD F	загружает программу и переменные.
LOAD F DATA ()	загружает числовой массив.
LOAD F CODE M,N	загружает старшие 'N' байтов, начиная с адреса 'M'.
LOAD F CODE M	загружает байты, начиная с адреса 'M'.
LOAD F CODE	загружает байты по тому же адресу, с которого они были разгружены.
LOAD F SCREEN$	аналогично LOAD F CODE 16384,6912. Очищает файл экрана и загружает его с кассетного магнитофона. Смотри главу 20.
LPRINT	подобно PRINT, но использует принтер.

оператор	действие оператора
MERGE F	подобно LOAD F, но не затирает всю старую программу в памяти, а заменяет только те строки и переменные, у которых совпадают номера или имена с такими же на ленте.
MOVE F1,F2	без MICRODRIVE не работает
NEW	запускает по новой систему программирования бейсик, уничтожая старую программу, переменные и используемую память, включая и байт адреса в системной переменной RAMBOT, но сохраняет системные переменные UDG, P RAMT,RASP и PIP.
NEXT A	А) находит управляющую переменную 'A'; Б) прибавляет к ней значение STEP; В) если STEP>=0, а значение 'A' стало больше значения 'предел', или STEP<0, а значение 'A' меньше, чем значение 'предел', то происходит переход к оператору цикла.
OPEN#	без MICRODRIVE не работает.
OUT M,N	выводит байт 'N' в порт 'M'. Операция выполняется на уровне микропроцессора (загружает в регистровую пару BC адрес 'M', а регистр A-'N' и выполняет команду ассемблера OUT (C)<A). 0 <= M <= 65535, -255 <= N <= 255, иначе выдается сообщение B.
OVER N	управляющий символ надпечатывания по выведенной строке. Если N=0, то выводимый символ затирает существующий в данной позиции. Если N=1, то новый символ соединяется со старым, образуя закрашивающий цвет, при условии, что старый символ имел указание цвета, отличное от старого, или цвет фона, если оба указывают на один и тот же цвет (либо фона, либо закрашивания, сложение по модулю 2). Смотри приложение B.
PAPER N	подобен INK, но управляет цветом фона.
PAUSE N	останавливает выполнение программы и задерживает изображение на экране на 'N' кадров (50 кадров в сек. — частота кадровой развертки) или до нажатия любой клавиши. 0 <= N <= 65535, иначе выдается сообщение B. При N=0 время задержки не учитывается и продолжается до первого нажатия клавиши.
PLOT C;M,N	выводит точку закрашивающего цвета (обработанную OVER и INVERSE) с координатами (ABS(M), ABS(N)) смещает графическую (PLOTPOSITION) позицию. Если цветной символ 'C' не специфицирован иначе, то цвет закрашивания в позиции, где расположена эта точка, изменяется на текущий сплошной закрашивающий цвет, и другие указания (цвет фона, мерцание, яркость) остаются без изменения. 0 <= ABS(M)<= 65535, 0 <= ABS(N)<= 175, иначе — сообщение B.
POKE M,N	записывает значение 'N' в байт памяти по адресу 'M'. 0 <= M <= 65535,, -255 <= N <= 255, иначе сообщение B.
PRINT...	где '...' последовательность PRINT-символов, разделенных запятыми, точками с запятой или апострофами, которые выводятся в экранный файл для отображения на экране телевизора. Точка с запятой сама действия не вызывает, а используется для разграничения символов. Запятая порождает управляющий символ 'запятая', и апостроф порождает символ ENTER. В конце оператора PRINT, если он не заканчивается точкой с запятой, запятой или апострофом, автоматически выводится символ ENTER. PRINT-символом может быть: А) пустая строка; Б) числовое выражение. Если значение выражения отрицательное, то выводится знак минус. Если X <= 10**-5 или X >= 10**13, вывод осуществляется в показательной форме. Мантисса представляется 8 цифрами (с нормализацией) и десятичной точкой (отсутствует только тогда, когда в мантиссе одна цифра) после первой цифры. Показатель степени записывается после буквы 'E' с последующим знаком и двумя цифрами порядка. Иначе X выводится как обычное десятичное число с 8-ю значащими цифрами. В) строковое выражение. В строке возможны пробелы до и после символов. Управляющие символы вызывают определяемое ими действие. Не отражаемые на экране символы выводятся как '?'. Г) AT M,N — вывод в строку 'M', позицию 'N'. Д) TAB N — вывод управляющего символа TAB с последующими 2-мя байтами 'N' (первый байт — старший). Вызывает TAB-останов. Е) цветовой символ в форме PAPER, INK,FLASH,BRIGHT,INVERSE или OVER-оператора.
RANDOMIZE	то же, что и RANDOMIZE 0
RANDOMIZE N	устанавливает системную переменную SEED, используемую для вычисления очередного значения функции RND. Если N <> 0, то SEED принимает значение 'N'. Если N=0, то SEED принимает значение другой системной переменной FRAMES, подсчитывающей кадры, отображаемые на экране, что обеспечивает вполне случайное число. Оператор запускает сокращение RAND (см. клавишу). Выдает сообщение B, если 'N' не лежит интервале от 0 до 65535.
READ V1,V2,...,VK	присваивает переменным одна за другой значения, последовательно представленные в списке DATA.
REM...	не выполняется. '...' может быть последовательностью символов (исключая ENTER). Может включать двоеточие (':') для указания отсутствия операторов в строке с REM.
RESTORE N	перезаписывает указатель данных в первый оператор DATA в строке меньшей, чем 'N'. Следующий оператор READ начнет считывание отсюда.
RETURN	ссылается на оператор GO SUB в стеке и передает управление на строку после него. Выдает сообщение 7, если нет указываемого оператора в стеке. Характерная ошибка, когда операторы GO SUB не сбалансированы операторами RETURN.
RUN	то же самое, что и RUN 0.
RUN N	CLEAR, а затем GO TO N.
SAVE F	записывает на ленту программы и переменные.
SAVE F LINE M	записывает на ленту программу и переменные таким образом, что при загрузке программа автоматически выполняется со строки 'M'.
SAVE F DATA ()	запись на ленту числового массива.
SAVE F DATA $()	запись на ленту строкового массива $.
SAVE F CODE M,N	записывает на ленту 'N' байтов, начиная с адреса 'M'.
SAVE F SCREEN$	аналогично SAVE F CODE 16384,6912. выдает сообщение F, если 'F' пустая строка или имеет длину более 10. смотри главу 20.

оператор	действие оператора
STOP	останавливает выполнение программы с выдачей сообщения 9. CONTINUE (продолжение) будет осуществляться со следующего оператора.
VERIFY	то же, что и LOAD, за исключением того, что данные загружаются в ОЗУ, но сравниваются с находящимися там. Выдает сообщение B, если обнаружен хотя бы один не совпадающий байт.
дополнение	
LOAD F DAT$()	загружает строковый массив.
RESTORE	то же самое, что и RESTORE 0

2.2. Сообщения

Они появляются в нижней части экрана, если компьютер остановился при выполнении некоторого оператора бейсика, и указывает 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 следует после первого или THEN и т.д.).

От состояния CONTINUE зависит очень многое в сообщениях. Обычно сообщение начинается с оператора, специфицированного в предыдущем сообщении, но имеется исключения — сообщение 0,9,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 адрес) индекс превышает размерность массива, либо ошибочное число задает индекс, если индекс отрицательный или больше 65535, то выдается сообщение B.	в индексной переменной или 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 1 или 2 строки программы (с целью возврата их впоследствии), получить дополнительную память, маневрируя оператором CLEAR.	LET, INPUT, FOR, DIM, GO SUB, LOAD, MERGE.
5	OUT OF SCREEN (вне экрана) если оператор INPUT генерирует больше, чем 23 строки в нижней половине экрана. Также встречается с PRINT AT 22,...	PRINT, PRINT AT.
6	NUMBER TOO BIG (число больше макс. допуст.) в результате вычислений получилось число больше 10**38.	арифметич. 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 следуюший 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 см. также сообщение 3.	RUN, RANDOMIZE, POKE, 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, VERIFY, MERGE, LPRINT, LLIST, COPY (только когда компьютер запросил свертку, а вы ответили N, SPACE или STOP.
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. действие 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)	в операциях с внешними устройствами

код	значение	ситуация
K	INVALID COLOUR (неверный цвет) специфицированное число имеет неверное значение.	INK, PAPER, BORDER, FLASH, BRIGHT, INVERSE, OVER, а также после одной из передач 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 указывает 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, который был описан.	FN
R	TAPE LOADING ERROR (ошибка загрузки с ленты) файл на ленте найден, но не может быть считан.	VERIFY, LOAD, MERGE.

2.3. Системные переменные

Байты памяти с 23552 до 23733 предназначены для специального использования. В них размещаются т.н. системные переменные. не надо путать их имена с именами переменных в программе. Компьютер не распознает ссылки к этим переменным из бейсик-программы по их именам. Имена используются только для мнемонического обозначения этих переменных в этом описании.

Информация, записанная в первом столбце таблицы, имеет следующее значение:

X — переменная не должна изменяться, т. к. Это может нарушить работу системы.
N — изменение переменной не приводит к длительному эффекту.
Цифра — число байтов переменной (для 2-байтовых переменных младший байт -1-й).

зн.	адрес	имя	содержание
N8	23552	KSTATE	используется при чтении с клавиатуры
N1	23560	LAST K	запоминается вновь нажатая клавиша
1	23561	REPDEL	время в 50-х долях секунды, в течение которого клавиша должна быть зафиксирована в нажатом состоянии. Начальное значение 35, но может быть изменено
1	23562	REPPER	задержка, в 50-х долях секунды, между последовательными опросами клавиш. начальное значение 5.
N2	23563	DEFADD	адрес аргументов функций пользователя если они используются, иначе 0.
N1	23565	к DATA	второй байт управления цветом с клавиатуры.
N2	23566	TVDATA	байты цвета, AT, TAB управления телевиз.
X38	23568	STRMS	адреса подключенных каналов
2	23606	CHARS	адрес символьного набора-256. Обычно этот набор находится в ПЗУ, но может быть размещаем и в озу с указанием в CHARS адреса размещения.
1	23608	RASP	продолжительность звукового сигнала.
1	23609	PIP	длительность задержки, устраняющей дребезг клавиатуры.
1	23610	ERR NR	код сообщения -1. Начальное значение 255 (для "-1"), т.е. PEEK 23610 = 255.
X1	23611	FLAGS	управляющие флажки бейсика.
X1	23612	TV FLAG	флажок телевизора.
X2	23613	ERR SP	адрес в аппаратном стеке, используемый как адрес возврата при ошибке.
N2	23615	LIST SP	адрес возврата из автоматического листинга.
N1	23617	MODE	режим. Спецификация [K], [L], [C], [E] или [G] курсора.
2	23618	NEW PPC	номер строки, на которую должен быть сделан переход.
2	23621	PPC	номер строки, оператор в которой выполняется.
1	23623	SUB PPS	порядковый номер выполняющегося оператора в строке.
1	23624	DORDCR	цвет рамки экрана, содержит атрибуты.
2	23625	E PPC	количество текущих строк (с курсором).
X2	23627	VARS	адреса переменных.
N2	23629	DEST	адрес переменной в заданнии.
X2	23631	CHANS	адрес канала данных.
X2	23633	CURCHL	адрес данных для ввода-вывода.
X2	23635	PROG	адрес бейсик-программы.
X2	23637	NXTLIN	адрес следующей строки в программе.
X2	23639	DATADD	адрес терминатора последнего символа в DATA.
X2	23641	E LINE	адрес выведенной команды.
2	23643	K CUR	адрес курсора.
X2	23645	CH ADD	адрес следующего интерпретируемого символа: символ аргумента в PEEK, NEWLINE или в POKE операторах.

зн.	адрес	имя	содержание
2	23647	X PRT	адрес символа, следующего за маркером [?].
X2	23649	WORK SP	адрес временной рабочей области.
X2	23651	STK BOT	адрес "дна" программируемого стека.
X2	23653	STK END	адрес начала резервной области памяти.
N1	23655	BREG	B-регистр калькулятора.
N2	23656	MEM	адрес области, используемой как память калькулятора (обычно MEMBOT, но не всегда)
1	23658	FLAG52	старшие флажки.
X1	23659	DF SZ	число строк (включая и одну чистую) в нижней части экрана.
2	23660	S TOP	количество верхних строк программы в автоматическом листинге.
2	23662	OLDPPC	номер строки, на которую указывает CONTINUE.
1	23664	OSPPC	номер оператора в строке, на которую указывает CONTINUE.
N1	23665	FLAGX	переменные флажки.
N2	23666	STR LEN	размер расстояний между строками.
N2	23668	T ADDR	адрес следующего символа в синтаксической таблице.
2	23670	SEED	начальное значение для RND, изменяется функцией RANDOMIZE.
3	23672	FRAMES	счетчик кадров — приращение через каждые 20 MS (см. Главу 18).
2	23675	UDG	адрес первого определяемого пользователем символа.
1 1	23677 23678	COORDS	X-координата точки графопостроителя Y-координата точки графопостроителя
1	23679	P POSN	33-позиционное число для позиционирования принтера
1	23680	PR CC	младший байт адреса позиции для LPRINT для печати.
1	23681		не используется.
2	23682	ECHO E	33-позиционное и 24-строковое числа (в нижней половине) конца входного буфера.
2	23684	DF CC	адрес PRINT-позиции в области экрана.
2	23686	DF CCL	подобно DF CC в нижней части экрана.
X1 X1	23688 23689	S POSN	33-позиционное число для PRINT позиции 23-строковое число для PRINT позиции.
X2	23690	S POSNL	подобно S POSN для нижней части.
1	23692	SCR CT	счетчик сверток:всегда на 1 больше числа сверток, которые должны быть проведены перед ост-вом со сверткой. Если вы установите это число больше, чем на 1 (скажем 255), то экран будет сворачиваться без запроса.
1	23693	ATTR P	сплошные цвета.
1	23694	MASK P	используется для высвечивания цветов бит, установленный в 1, показывает, биты атрибутов берутся не из ATTR P, а из того, что указано на экране.
N1	23695	ATTR T	временный указатель цветов
N1	23696	MASK T	временный MASK P
1	23697	P FLAG	старшие флажки.
N30	23698	MEMBOT	область памяти для калькулятора. используется для записи чисел
2	23728		не используется
2	23730	RAMTOP	адрес последнего байта области бейсик-системы.
2	23732	P-RAMT	адрес последнего байта физического ОЗУ.

3.1. BETA BASIC

BETA BASIC версии 1.8 добавляет 30 новых команд и 21 новую функцию к уже имеющимся в ZX SPECTRUM. Вызов некоторых из имевшихся команд улучшен, сделан ряд небольших изменений с целью облегчения работы с компьютером (например, мигающий курсор в текущей строке, полная реализация действия клавиши BREAK, а также возможность перемещения курсора в любом направлении во время редактирования строки программы).

Новые команды вызываются путем перехода в режим GRAPHICS и последующего нажатия соответствующей клавиши, после чего сразу появляется новое ключевое слово из состава команд BETA BASIC.

Новые функции появляются при вводе нормального ключевого слова FN и последующей соответствующей буквы и знака $ (см. примечание) и ''('. После этого появляется название новой функции, после чего надо ввести ее аргументы.

Загрузка системы BETA BASIC производится как обычно, т.е. по команде LOAD '' '' или LOAD ''BETA BASIC''. Программа состоит из двух частей, первая из которых располагается в строках с номерами 0, 1 и 2, а вторая — это исполнительная программа в машинных кодах. При загрузке строки 0, 1 и 2, содержащие определения функций BETA BASIC вводятся автоматически и всегда находятся в программе, хотя и скрыты от пользователя.

После этого система готова к работе.

Часть 1. Описание команд BETA BASIC

Ниже даны описания следующих в алфавитном порядке ключевых слов со списком аргументов. Аргументы, показанные в угловых скобках могут быть опущены и употребляются по необходимости. В квадратных скобках указаны клавиши под которыми ''скрыты'' данные ключевые слова.

ALTER <список атрибутов> TO список атрибутов [A]

Быстрая смена атрибутов экрана (INK, FLASH, BRIGHT, PAPER) без необходимости очистки экрана.

Пример:

ALTER TO PAPER 1, INK 6

Включение автоматической нумерации строк начинается с указанной строки. Если строка не указана, то нумерация начнется с текущей строки с указанным шагом. Если шаг не указан, то по умолчанию принимается шаг=10. Выключение нумерации — при нажатии клавиши BREAK дольше 1 секунды.

BREAK [SHIFT SPACE]

Останов выполнения программы. BETA BASIC использует прерывания в режиме INTERRUPT 2 микропроцессора Z80. Команда не является ключевым словом и не вызывается в режиме GRAPHICS.

CLOCK строка [C]

Задание параметров в часах-будильнике.

Примеры:

CLOCK ''09:29:05'' — установка точного времени в часах,

CLOCK ''A06:20'' — установка времени срабатывания будильника,

CLOCK ''A9000'' — установка номера строки — адреса перехода при срабатывании будильника.

CLOCK аргумент [C]

Управление часами с помощью поданного аргумента.

	Переход на строку	звуковой сигнал	световой
CLOCK 0	откл	откл	откл
CLOCK 1	откл	откл	вкл
CLOCK 2	откл	вкл	откл
CLOCK 3	откл	вкл	вкл
CLOCK 4	вкл	откл	откл
CLOCK 5	вкл	откл	вкл
CLOCK 6	вкл	вкл	откл
CLOCK 7	вкл	вкл	вкл

CHR$ цифра

Знаки, вставляемые в строку, печатаемую оператором PRINT, для изменения позиции печати. Можно применять для вертикальных надписей и в псевдографике.

CHR$ 8	— курсор влево
CHR$ 9	— курсор вправо
CHR$ 10	— курсор вниз
CHR$ 11	— курсор вверх

DEF KEY имя клавиши:строка SHIFT1
DEF KEY имя клавиши:инструкция:инструкция:...

После подачи этой команды нажатие указанной клавиши будет вызывать появление заданной строки или цепочки инструкций. Имя клавиши — это строка, состоящая из одной буквы.

DEF PROG имя процедуры [1]

Ключевое слово, обозначающее начало процедуры, вызываемой по имени.

DELETE <номер строки> TO <номер строки> [7]

Стирание строк в заданном диапазоне. Строка с курсором подразумевается по умолчанию.

Пример:

DELETE 10 TO 50

DO [D]
DO WHILE условие входа [D]
DO UNTIL условие входа [D]

Ключевое слово, обозначающее начало цикла.

Цикл DO начинается всегда и без условий.

Цикл DO WHILE начинается, если заданное условие входа истинно.

Цикл DO UNTIL начинается, если заданное условие входа ложно. Цикл должен закончиться командой LOOP.

DPOKE адрес, число [P]

Загрузка двухбайтового числа в диапазоне 0...65535 по данному адресу.

EDIT <номер строки> [0]

Позволяет нормальное редактирование строки. Появляется при нажатии клавиши [0], если до этого была нажата клавиша ENTER.

ELSE инструкция [E]

Третья (альтернативная) часть структуры IF-THEN.

Пример:

IF условие THEN инструкция ELSE инструкция

END PROG [3]

Слово, обозначающее конец процедуры, вызываемой по имени.

EXIT IF условие [1]

Выход по заданному условию из цикла DO — LOOP.

FILL X,Y [F]
FILL <INK цвет>;X,Y [F]
FILL <PAPER цвет>;X,Y [F]

Закрашивание знака данным цветом (если нажато FILL или FILL INK) и фона знака (если нажато FILL PAPER).

GET числовая переменная или переменная-строка [G]

Присвоение переменной номера клавиши (счет клавиш идет с 11 для A, 12 для B, 13 для C и т.д.) или самой буквы на нажатой клавише.

JOIN <номер строки> [SHIFT 6]

Объединение строки с заданным номером (или текущей строки, если номер не указан) со строкой, находящейся в нижней части экрана. Новый номер равен указанному номеру или номеру текущей строки.

KEYIN строка [SHIFT 4]

Ввод в программу поданной строки (например, строки программы). Допускается применение только как инструкции в программе.

KEYWORDS 0 [8]
KEYWORDS 1 [9]

Переключатель ключевых слов BETA BASIC на знаки псевдографики, вызываемые в режиме GRAFICS. Это ключевое слово не высвечивается.

LIST номер строки TO номер строки
LLIST номер строки TO номер строки

Печать строк программы в заданном диапазоне. Расширение синтаксиса ключевого слова из набора SPECTRUM BASIC. Вызывается нормально.

LOOP [L]
LOOP UNTIL условие [L]
LOOP WHILE условие [L]

Ключевое слово, обозначающее конец цикла. При LOOP цикл кончается всегда и без условий. При LOOP UNTIL цикл кончается, если заданное условие входа истинно. При LOOP WHILE цикл кончается, если заданное условие входа ложно.
Цикл должен начинаться командой DO.

GO TO ON переменная;номер строки, номер строки,...[O]
GO SUB ON переменная;номер строки, номер строки,...[O]

Переход на строку в зависимости от значения переменной.

ON ERROR номер строки [N]

Включение части программы обслуживания ошибок. При ошибке управление передается на заданную строку. Переменная ERROR получает значение кода ошибки. Не работает при кодах 0 (OK) и 9 (STOP). Выключение обслуживания ошибок происходит после выполнения перехода или по оператору ON ERROR 0.

PLOT X,Y; строка

Вывод строки знаков в любое место экрана. Координаты задают позицию левого верхнего угла знака строки. Допустимы знаки управления курсором.

POKE адрес, строка

Расширение синтаксиса нормального ключевого слова. Вводит в память знаки заданной строки, начиная с данного адреса.

POP <числовая переменная> [Q]

Возвращает адрес, откуда был сделан вызов GO SUB, DO — LOOP, PROC. Адрес в виде номера строки представляется как значение переменной, если ее имя было указано.

PROC имя [2]

Выполнение процедуры с данным именем.

RENUM <начало TO конец><LINE новое начало>
<STEP шаг> [9]

Перенумерование строк из заданного диапазона на строки, начинающиеся с заданной строки с данным шагом. По умолчанию первая строка — 10, шаг — 10.

ROLL направление <, сдвиг><;X,Y;ширина, высота> [R]

"Перемотка" заданного окна экрана. Изображение, исчезающее с одной стороны окна, появляется с противоположной. Ширина и высота окна задается в знаках, а не в точках экрана. Координаты задают левый верхний угол окна. Сдвиг задается как количество точек. Если задано только направление, то "перемотка" выполняется для всего экрана. Можно двигать изображение с атрибутами, только изображение или только атрибуты. При сдвигах атрибутов число сдвигов должно быть равно 8. Направление и вид сдвига выбираются по таблице.

код	направление	объект передвижения
1	влево	атрибуты
2	вниз	атрибуты
3	вверх	атрибуты
4	вправо	атрибуты
5	влево	изображение
6	вниз	изображение
7	вверх	изображение
8	вправо	изображение
9	влево	изображение и атрибуты
10	вниз	изображение и атрибуты
11	вверх	изображение и атрибуты
12	вправо	изображение и атрибуты

SCROLL <направление><, сдвиг><;X,Y;ширина, высота>[S]

Сдвиг содержимого заданного окна экрана на одну линию. Синтаксис такой же, как в команде ROLL. При отсутствии параметров изображение на всем экране сдвигается на одну линию вверх.

SORT таблица или строка [M]
SORT INVERSE таблица или строка [M]

Упорядочение таблицы знаков или строки в прямом либо обратном порядке, а также таблицы чисел в последовательность от максимального числа по убыванию либо наоборот. Числовая таблица может быть одномерной или двумерной.

SPLIT

Разбиение редактируемой строки на две части. Не является нормальным ключевым словом, вместо него вводится знак "<>" (SYMBOL SHIFT W). В месте нахождения курсора строка разделяется на две части: первая часть пересылается в программу с тем же номером, а часть, находящаяся после курсора, остается в зоне редактирования, где ей можно присвоить другой номер.

TRACE <номер строки> [T]

Пуск пошагового выполнения программы, начиная с заданной строки. Выключение режима командами RUN, CLEAR и TRACE 0.

UNTIL условие [K]

Часть операторов цикла DO — LOOP, используемая в виде DO UNTIL или LOOP UNTIL, означающая выход из цикла, если условие выполняется.

USING строка-образец;число [P]

Использование в команде PRINT USING позволяет задавать формат распечатки чисел. Знаки "+" в строке-абзаце означают пробелы перед числами.

WHILE условие [J]

Часть операторов цикла DO — LOOP, используемая в виде DO WHILE или LOOP WHILE, означающая выход из цикла, если условие не выполняется.

Часть 2. Описание функций BETA BASIC

Ниже приводится краткое описание функций, добавляемых системой BETA BASIC с описаниями аргументов. Способы ввода функций в программу указаны в квадратных скобках.

AND (число, число) [FN A( ]

Двоичная операция "логическое И", выполненная над заданными числами.

BIN$ (число) [FN B&( ]

Двоичная распечатка данного десятичного числа.

CHAR$ (число) [FN C$( ]

Преобразование целого числа без знака в диапазоне 0 — 65535 в эквивалентную строку из двух знаков.

COSE (число) [FN C( ]

Косинус числа, вычисляемый с 4 значащими цифрами. Более быстрое вычисление, чем в оригинальной системе SPECTRUM BASIC.

DEC (строка) [FN D( ]

Преобразование строки с записью шестнадцатиричного числа в десятичное число.

DPEEK (адрес) [FN P( ]

Возвращает двухбайтовое число, находящееся в памяти по данному адресу.

FILLED () [FN F( ]

Количество элементов изображения, заполненных последней командой FILL.

HEX$ (число) [FN H( ]

Преобразование десятичного числа в строку с записью шестнадцатиричного числа.

INSTRING (начало, строка 1, строка 2) [FN I( ]

Возвращает позицию первого знака строки 2, находящейся внутри строки 1 при просмотре строки 1, начиная с заданной стартовой позиции. Если внутри строки 1 нет строки 2, то возвращается 0.

MEM () [FN M( ]

Возвращает количество свободных байтов памяти.

MEMORY$ () [FN M$( ]

Возвращает значение всей памяти от адреса 0 до 65535 интерпретированное как одна строка.

MOD (число 1, число 2) [FN V( ]

Остаток деления числа 1 на число 2 (деление по модулю).

NUMBER (строка) [FN N( ]

Преобразование строки из двух знаков в двухбайтовое число, где каждый байт содержит число, отвечающее заданному коду ASCII.

OR (число 1, число 2) [FN O( ]

Двоичная операция ''логическое ИЛИ'', выполненная над заданными числами.

RNDM (число) [FN R( ]

Возвращает псевдослучайное число из диапазона от 0 до заданного числа.

SCRN$ (ряд, столбец) [FN K$( ]

Возвращает знак, находящийся на экране в заданной позиции.

SINE (число) [FN S( ]

Синус числа, вычисляемый с 4 значащими цифрами. Более быстрое вычисление, чем в оригинальной системе SPECTRUM BASIC.

STRING$ (количество, строка) [FN S$( ]

Повторение строки заданное количество раз.

TIME$ () [FN T( ]

Текущее время, измеренное по часам CLOCK.

USING$ (строка-образец, число) [FN U$( ]

Знаковая запись числа в заданном формате (как USING).

XOR (число 1, число 2) [FN X( ]

Двоичная операция ''логическое ИСКЛЮЧАЮЩЕЕ ИЛИ'', выполненная над заданными числами.

Часть 3. Специальные переменные

В системе BETA BASIC имеются встроенные переменные, доступные по имени.

XOS, YOS — относительное начало координат, первоначально установленное на 0, 0. Может быть изменено в диапазонах 0—255 для XOS и 0—175 для YOS. Обнуляются по командам CLEAR и RUN.

XRG, YRG — максимальные границы изображения, первоначально установленные на 256 для XRG и на 176 для YRG.

Во время выполнения команд ON ERROR и TRACE обновляются следующие специальные переменные:

ERROR — код последней обнаруженной ошибки.

LINE — номер последней выполненной строки программы (при TRACE), номер строки с обнаруженной ошибкой (при ON ERROR).

STAT — номер последнего выполненного оператора программы (при TRACE), номер оператора с обнаруженной ошибкой (при ON ERROR).

Имена специальных переменных можно вводить заглавными и строчными буквами.

3.2. MEGA BASIC

1. Клавиатура

Как только программа будет загружена, вы увидите короткое сообщение и инверсный квадрат в левом нижнем углу экрана, это новый курсор который будет показывать, где будет появляться входная информация. При использовании ''MEGA BASIC'' весь экран используется для входной информации в отличии от обычного бейсика, где весь экран разбивается на 2 части.

Попытайтесь записать несколько символов с клавиатуры и вы заметите, что при нажатии клавиш на экране не появляются привычные для вас слова, соответствующие нажатой клавиши, а появляются отдельные символы. Теперь вы будете записывать весь оператор полностью, как в обычных компьютерах, не используя отдельные слова вашего ZX SPECTRUM. Хотя переход от системы клавишных слов имеет много преимуществ, но имеются и недостатки. Определенные операторы, такие как PRINT, могли быть записаны в одно нажатие клавиши ''P'', тогда как теперь вам необходимо будет писать ''P'' ''R'' ''I'' ''N'' ''T''. Однако ''MEGA BASIC'' позволяет многие операторы записывать в виде аббревиатур. Заметьте, что клавишные слова, не попавшие в этот список, не могут быть записаны в виде аббревиатуры и должны записываться полностью. Также заметьте что аббревиатура должна заканчиваться точкой. Например аббревиатура для CONTINUE есть CON.

A.TTR	ER.ASE	ME.RGE	RES.TORE
BE.EP	E.XP	M.OVE	RET.URN
B.IN	FL.ASH	NE.XT	R.ND
BO.RDER	F.ORMAT	N.OT	SA.VE
BR.IGHT	GOS.UB	OP.EN#	S.CREEN$
CH.R$	G.O TO	OV.ER	ST.R$
CI.RCLE	I.NKEY$	PA.PER	T.AB
CLE.AR	INP.UT	PAU.SE	TH.EN
CL.OSE#	INV.ERSE	PE.EK	U.SR
C.ODE	L.EN	PL.OT	V.AL$
CON.TINUE	LI.NE	P.OINT	VE.RIFY
DA.TA	LL.IST	PR.INT,	
D.EFFN	LP.RINT	RA.NDOMIZE,	
DR.AW	LO.AD	RE.AD,	

Помните, что если вы записываете такие команды, как GO TO и OPEN# полностью, то не забывайте оставлять необходимые пробелы: компьютер поймет запись GO TO, а при записи GOTO на экране появится сообщение SYNTAX ERROR.

Нижняя строка на экране используется для индикации режима курсора, ниже в таблице показаны обычные режимы SPECTRUM'а а рядом сообщение на экране которое вы получаете в нижней строке.

Таблица 1

режим курсора	нижняя строка экрана
L	CAPS OFF
G	CAPS OFF GRAPHICS
C	CAPS ON
E	CAPS OFF EXTENDED

Режим курсора изменяется обычным путем — нажатием клавиш CAPS LOCK, GRAPHICS, CAPS SHIFT, SIMBOL SHIFT.

2. Редактор

Возможности у ''MEGA BASIC'' намного больше, чем у обычного бейсика (по редактированию строк).

Таблица 2

команда	содержание
EDIT	копирование очередной программной строки во входную строку, готовую для редакции.
TRUE VIDEO	удаление всей входной строки
INVERSE VIDEO	удаление символа правее курсора
CURSOR LEFT	перемещение курсора влево на один символ
CURSOR RIGHT	перемещение курсора вправо на один символ

команда	содержание
CURSOR DOWN	перемещение курсора вниз на одну строку
CURSOR UP	перемещение курсора вверх на одну строку
DELETE	удаление символа слева от курсора
<=	перемещение курсора на начало входной строки
<>	удаление всех символов от начала курсора до конца входной строки
>=	перемещение курсора в конец входной строки
SCREEN$	автоматический листинг, (выход на окно 1) верхней строкой листинга является текущая строка
OR	перемещение очередной строки вниз на одну и листинг
AND	перемещение очередной строки вверх на одну и листинг
STOP	перемещение COPY курсора влево на один символ
STEP	перемещение COPY курсора вверх на одну строку.
NOT	перемещение COPY курсора вниз на одну строку
TO	перемещение COPY курсора вправо на один символ
AT	копировка символа от COPY курсора к входному курсору. Этот оператор используется для копировки символов, имеющих стандартный размер и не может применяться для символов при 64 рабочих столбцах экрана, символов с двойной шириной и высотой
OVER	перемещение COPY курсора на следующее окно
INVERSE	удаление COPY курсора в верхний угол очередного окна.

Заметим, что экран разделен на 3 различные секции или окна и каждое используется для специфических задач.

Таблица 3

секция	содержание
окно 0	для индикации поступающей от пользователя информации и сообщений об ошибках
окно 1	индикация листинга программы начиная с редактируемой строки
окно 2	выходной информация по командам бейсика
окно 3	используется как фронт-панель

Когда "MEGA BASIC" будет загружен, вы можете подумать, что на экране только одно окно, но вы ошибаетесь — фактически здесь все 4 окна, но они перекрывают друг друга.

Заметим также, что 2-ой курсор может быть использован, чтобы скопировать текст с другой части экрана к входному курсору. Этот курсор появляется как мигающий квадрат на экране и может перемещаться внутри очередного окна с помощью запрограммированных управляющих клавиш. COPY курсор функционирует только в окнах 0, 1, 2.

Также как использование CAPS SHIFT 1 для редакции очередной строки, возможна редакция любой строки в программе, используя клавишу EDIT. За этой командой следует числовое выражение, указывающее, какая строка будет редактироваться. Если требуемая строка не существует, то происходит переход к следующей строке для ее редакции, если же нет и следующей строки, то на экране появляется сообщение:

LINE NOT FOUND
(строка не найдена)

3. Клавиши, определяемые пользователем (UDK)

Имеется возможность запрограммировать верхний ряд клавиш и создать тем самым более 255 символов. Для программирования используйте команду KEY_, за которой должно следовать числовое и символьное выражение, разделенные запятой. Числовое выражение определяет, какая из клавиш программируется, символьное выражение — содержание этой клавиши. Записывая ENTER символ CHR$13 в конце определяемого вами символьного выражения, вы получите автоматическое его выполнение после нажатия UDK. "MEGA BASIC" уже включает в себя следующие UDK.

Таблица 4

UDK	содержание
VERIFY	в символьном выражении записано RUN+CHR$13, при нажатии на эту клавишу выполняется очередная программа в памяти.
VAL$	в символьном выражении записано LOAD "", RUN+CHR13 при нажатии этой клавиши загружается, а затем выполняется очередная программа на кассете.

4. Управляющие клавиши

В течение времени выполнения программы SPACE-клавиша может быть использована как SHIFT-клавиша. Поэтому, чтобы получить пропуски между символами, обе эти клавиши должны быть нажаты одновременно. SPACE-клавиша, в сочетании с другими клавишами, теперь будет представлять собой управляющую клавишу. Здесь следуют возможные клавиши управления в течение времени выполнения программы (RUN-TIME).

Таблица 5

клавиша	содержание
CONTROL F	вызов фронт-панели
CONTROL E	останов выполнения программы. Печать сообщения ESCAPE и пе- реход к строке редактора.
CONTROL E	останов выполнения программы, установка начальных значений PAPER BORDER и INK. Переход к строке редактора. Эта функция не уничтожает программы на бейсике в памяти компьютера.

5. Команды экрана

Размер и форма символов, а также способ, по которому они могут быть выведены — факторы, облегчающие работу с "MEGA BASIC". Теперь, к примеру, возможно ограничить площадь экрана для выходной информации. Эта площадь может быть любого размера, а может занимать и весь экран. Отдельные участки площади экрана назовем "окнами", каждое окно имеет свой номер. Вы можете получить 10 окон, пронумерованные от 0 до 9, и все они могут быть использованы в вашей программе.

Когда вы запустите вашу программу, все PRINT операторы будут использовать окно 0, тем не менее имеется возможность перейти на другое окно, используя команду:

CURRENT-

За командой должно следовать числовое выражение, определяющее номер окна для использования. Если это выражение больше 9, то на экране появится сообщение об ошибке ILLEGICAL WINDOW. Окна могут иметь любой размер и занимать произвольное положение на экране. Для определения размера и положения окна на экране используется команда

WINDOW-Y,X,D,W

где:

Y — строковая позиция верхнего левого угла (может изменяться от 0 до 23)

X — столбцовая позиция верхнего левого угла (может изменяться от 0 до 63)

D — глубина окна (в строках)

W — ширина окна (в столбцах)

"MEGA BASIC" разбивает экран на 24 строки и 64 столбца. Функции ATTR и SCREEN$ используют старую координатную систему, а оператор PRINT AT — новую систему. Также координаты для PRINT AT задаются относительно верхнего левого угла очередного экрана, в то время как ATTR SCREEN$ используют абсолютные координаты экрана. Если Y+D больше 24 или X+W больше 64, то вы получите сообщение об ошибке WINDOW TOO LARGE (окно слишком большое). Если же значение меньше 0, то вы увидите WINDOW TOO SMALL (окно слишком маленькое) После выполнения команды WINDOW позиция устанавливается (для печати) в верхнем левом углу окна.

CLS и CLW-

Обычную команду CLS, используемую для очистки всего экрана нельзя использовать, если вы желаете очистить только одно окно. Тем не менее, новая команда CLW позволяет решить такую задачу. За командой должны следовать одно или 2 числа. Если два числа, то первое определяет номер окна для очистки, а второе означает, какую запись требуется очистить. Если число только одно, то оно означает номер очищаемого окна. Существует 4 различных типа команды CLW (N — номер окна):

Таблица 6

CLW	содержание
CLW-N,0	затушевывается окно, используя цвет PAPER
CLW-N,1	затушевывается окно, используется цвет INK
CLW-N,2	инвертируется содержимое экрана, цвет PAPER меняется на цвет INK
CLW-N,3	очистка только от атрибутов. Эта опция позволяет пользователю изменить цвет окна, не разрушая его содержимое

Во всех случаях позиция для печати устанавливается в верхнем левом углу окна. Команда CLW всегда использует текущие атрибуты. Так, например, если текущее окно 3, то после записи команды CLW 0,0 и ее выполнения окно 0 будет затушевываться, используя атрибуты окна 3.

PAN и SCROOL

Имеется возможность пиксель за пикселем перемещать окно вверх, вниз, влево, используя команды PAN и SCROOL. PAN — позволяет перемещать окно в стороны, а SCROOL по вертикали. После обеих команд должны следовать 2 числа, первое показывает цвет кромки экрана. Второе число показывает, в каком направлении и на сколько пикселей окно должно быть перемещено.

Для PAN-: если второе число положительное, то перемещение вверх, если отрицательное — вниз.

Возможно окна также повернуть, для этого используйте команды:

PANW- и SCROOLW-

После обеих команд должно быть число, указывающее в каком направлении и на сколько пикселей переместить. Все эти команды действуют только на дисплейный файл. Выбор окна осуществляется по команде:

FX-

После этой команды следуют 2 числа: первое показывает, для каких целей окно, а второе — номер окна.

Таблица 7

FX-	содержание
FX-0,N	окно N назначается для входной информации и сообщений об ошибках
FX-1,N	окно N назначается для автоматического листинга.
FX-2,N	окно N назначается для выходной информации
FX-3,N	окно N назначается для фронт-панели.

К примеру, запись FX-0,5 означает, что окно 5 выбрано для входной информации и сообщений об ошибках. При желании наберите и запустите программу.

```
10 BORDER 0
20 CURRENT-0;WINDOW-0,0,15,32
30 PAPER 2: INK 7:CLW-0
40 CURRENT-1: WINDOW-0,32,22,32
50 PAPER 1:INK 7: CLW-0
60 CURRENT-2: WINDOW-15,0,7,32
70 PAPER 7: INK 1: CLW-0
```

Вы увидите окна, перемещающиеся по экрану. Для остановки перемещения нажмите клавишу M.

Используя команду:

MODE-

можно получить 4 различных размера символов, выводимых на экран.

Таблица 8

MODE	содержание
MODE-N,1	используется 64 столбца и каждый символ имеет размер 4*6 пикселей
MODE-N,2	обычный размер 8*8 пикселей
MODE-N,3	символы двойной высоты 8*16
MODE-N,4	символы размером 16*16

За каждой командой MODE следует 1 или 2 числа. Если два числа, то первое указывает номер окна, а второе — размер символов. Для каждого окна могут быть заданы свои размеры символов. При использовании команды MODE-N,4 возможно получение символов с оттенками, поскольку в этом случае каждый пиксель занимает 4 пикселя экрана, т.е. установив один образ для 4-х точек, можно получать теневые эффекты. Для записи образа служит команда:

STRIPPLE3A

Командой следует числовое выражение, которому соответствует определенный образ. Число может изменяться от 0 до 15 и задает интенсивность тени.

FONT

"MEGA BASIC" позволяет формировать 3 типа набора символов. Выбор набора символов производится по команде FONT-, после которой следует числовое выражение, определяющее выбор. Вот что вы будете получать, применяя эту команду:

Таблица 9

FONT	содержание
FONT-0	обычные символы SPECTRUM'a
FONT-1	символы подобные применяемым в BBC MICRO AKORN
FONT-2	символы подобные применяемым в AMSTRAD CPC 464

Новые символьные наборы хранятся в ОЗУ и могут быть использованы с помощью описанных команд. Каждый символ занимает 8 байт и имеет определенную форму. В псевдо BBC MICRO символы находятся в памяти, начиная с адреса 48000, а псевдо AMSTRAD — с адреса 45000.

Если A есть определенный символ, то адрес, по которому он находится определяется по формуле:

S+8*(CODE A-32)

где S — начальный адрес.

SHR$

Символьный набор SPECTRUM'a содержит печатаемые символы и управляющие символы. Несколько новых управляющих символов, входит в "MEGA BASIC".

Таблица 10

SHR$	содержание
SHR$1-4	выбор режима очередного окна
SHR$7	перестановка символа к курсору
SHR 24-31	выбор окна для выходной информации. SHR$ 24 соответствует окну 0, SHR$ 31 — окну 7

Команда VDU- эквивалентна PRINT CHR$. К примеру: VDU-7 обеспечивает одинаковый размер всех последующих символов, а VDU 65,66 печатает "AB".

Строки символов могут быть выведены вниз экрана командой

DOWN-Y,X,A$

где Y, X — соответственно начальная строка и столбец, на которые выводится информация. A$ — выводимая информация.

Если информация занимает больше 1 строки то продолжение появится уже наверху.

Используя команду:

SPRINT X, Y, A, B, A$

вы можете вывести на экран символы любого размера.

Здесь

X,Y — позиция первого курсора на экране.

A,B — коэффициент увеличения символа в направлении соответственно.

A$ — строка выводимых символов. Если символы достигли правой кромки экрана, то запись появляется с левой стороны экрана.

Команда PRINTER

Обеспечивает вывод информации на периферийные устройства, такие, например, как принтер. Вслед за командой следует числовое выражение. Если результат его равен 0, то информация будет выводится только на экран. Если результат отличен от 0, то при каждом появлении символа на экране будет вызываться подпрограмма вывода в машинных кодах. Адрес ее хранения находится в ячейках 59934 и 59935. Заметим, что следующие команды: CLEAR# OPEN#2, CLOSE#2 не выполнимы в "MEGA BASIC".

6. Графика

CHANGE и SWAP

В "MEGA BASIC" выполнены некоторые команды, которые позволяют непосредственно манипулировать файлом атрибутов. CHANGE — новая команда, позволяющая изменить определенную часть каждого атрибута байта. После этой команды следуют 2 числа, представляющие маску и данные. Маска указывает, какой из битов каждого атрибутного байта должен быть изменен. Данные показывают, в чем должно состоять это изменение.

Таблица 11

CHANGE	содержание
CHANGE-1	логическое NOT, все 1 превращаются в 0, а все 0 в 1.
CHANGE-2	логическое AND для байта файла атрибутов и отрицательной маски
CHANGE=#	каждый атрибутный байт через логическое OR соединяется с данным байтом.

Команда SWAP позволяет поменять одни атрибуты на другие. За командой следуют 2 числовых выражения: первое — новые атрибуты, второе — старые. По этой команде в файле атрибутов отыскиваются атрибуты, требующие замены, и заменяются новыми.

FADE

Эта команда производит некоторые специальные эффекты. Введите следующую короткую программу:

```
10 FOR A=0 TO 703
20 POKE 22528+A, PEEK A
30 NEXT A
40 FADE-0
```

Первые 3 строки заполняют файл атрибутов случайными символами, по команде FADE файл атрибутов исследуется. Каждый байт, который равен численному выражению после команды FADE, остается неизменным, другие уменьшаются. Этот процесс продолжается до тех пор, пока каждый байт в файле атрибутов не будет равен числовому выражению стоящему после команды FADE.

INVERT

Это просто команда, по которой инвертируется весь экран. Цвет INK меняется на цвет PAPER, а цвет PAPER на цвет INK.

DEFG

Способ, по которому создаются новые графические элементы пользователя в обычном бейсике довольно громоздок. В "MEGA BASIC" это решается с помощью одной команды DEFG, за которой следует любой символ от A до U и 8 чисел, разделенных запятыми. Символ определяет местонахождение нового графического элемента, а числа — его форму (первое число определяет верхний ряд, а последнее — нижний ряд).

GET и PUT

Возможно сохранять содержимое экрана в памяти, а затем перенести его из памяти на экран в любую позицию. Для этого служат 2 команды: GET и PUT. По команде GET содержимое экрана передается в память, а по команде PUT осуществляется обратная операция.

Формат команды GET:

GET-0,A,Y,X,D,W

где:

A — адрес начиная с которого хранится информация экрана

Y,X — соответственно номер строки и столбца верхнего угла записанной площади экрана

D,W — соответственно глубина и ширина этой площади

PUT и GET используют ту же координатную систему, что и ATTR и SCREEN$. Их координаты абсолютны и не зависят от положения верхнего левого угла окна на экране.

Команда GET обеспечивает сохранение дисплейного файла в памяти с информацией и о атрибутах. Использование команды CLEAR позволяет резервировать участок памяти для сохранения информации экрана. Вы можете определить число байтов, занимаемых информацией экрана, с помощью уравнения Y, W, D. Команда PUT прямо противоположна GET и имеет следующий формат:

PUT-F,A,Y,X,W,D

где: F — показывает способ восстановления экрана из памяти (F=0.....6)

остальные переменные такие же как и у команды GET

SPUT-A,X,Y,B,C,W,D

Эта команда аналогична PUT, но позволяет увеличить выводимое изображение. B и C — коэффициенты увеличения.

7. Управление выполнением программ

"MEGA BASIC" обеспечивает работу с процедурами, которые могут быть вызваны по их имени на выполнение, как будто это новые команды. И точно также, как и после команды, после имени процедуры, если это необходимо, следует серия выражений. Эти выражения могут быть назначены как переменные, готовые для манипуляции внутри программы. Недостатком процедур "MEGA BASIC" является то, что они не могут манипулировать с локальными переменными, а их переменные являются общими для всей программы. Используя процедуры, можно разделить программу на ряд задач. Для каждой частной задачи составляется своя процедура. Каждая процедура может быть проверена отдельно и затем они все объединяются вместе для окончательной программы. Выбирайте внимательно имена для ваших процедур и тогда будет намного легче находить их в программном потоке. Начало процедуры определяется символом @, а следом имя процедуры. Оператор, определяющий процедуру, должен быть первым в строке. После имени процедуры должны следовать ее параметры.

Конец процедуры определяется оператором

ENDPROG

Как только встретится этот оператор, компьютер перейдет к оператору, следующему за оператором вызова процедуры. В конце оператора можно записать имя процедуры. Посмотрите на пример:

```
9000 @DISPLAY A,A$
9010 PAPER A: INK 9
9020 MODE -4 : STRIPPLE-6
9030 PRINT A$
9040 ENDPROG-DISPLAY
```

Строка 9000 определяет процедуру. DISPLAY A,A$, показывает, что для него необходимы числовое выражение (A) и символьное выражение (A$). Строка 9010 устанавливает цвет, а строка 9020 — правильный размер символов. Строка 9030 выводит на печать символьное выражение A$ и, наконец, строка 9040 определяет конец процедуры. Для активизации этой подпрограммы вы будете использовать в своих программах строку подобную этой.

DISPLAY-2, "MEGA SPECTRUM"

Так как переменные являются локальными, вы должны быть уверены, что они используются только один раз в одной процедуре и не используются в других частях программы. Процедуры могут быть вызваны только изнутри программы и не могут вызываться прямыми командами.

REPEAT-UNTIL

Так же хорошо, как и с процедурами, "MEGA BASIC" обеспечивает работу с циклами типа REPEAT-UNTIL. REPEAT команда начинает цикл, после команды UNTIL следует числовое выражение. Если этому числовому выражению присваивается нулевое значение, то выполнение программы возвращается назад, к оператору записанного после последнего REPEAT. Если выражение является положительным числом, то выполняется следующий оператор программы. Допускается до 10 циклов вложения типа REPEAT-UNTIL.

Для сохранения номеров строки операторов процедур и циклов REPEAT-UNTIL используется стек. Когда вызывается процедура, номер строки, следующей за оператором вызова, размещается в стеке — это позволяет программе знать, куда возвращаться после выполнения оператора процедуры. Когда будет выполнена команда REPEAT, номер строки с оператором после команды REPEAT заталкивается в стек. Это необходимо, чтобы после выполнения команды UNTIL было известно, где последняя команда REPEAT. После выполнения команды ENDPROG и перехода к строке, номер которой хранится в стеке, последний убирается оттуда. Также и после выполнения команды UNTIL верхнее значение выталкивается из стека.

Если стек пустой и будет попытка вытолкнуть из него, то на экране появится сообщение об ошибке:

"PROG STACK UNDERFLOW"

Любая попытка записать в стек более 10 значений заканчивается сообщением об ошибке, и на экране появится:

"PROG STACK OVERFLOW"

POP и PUSH

Оператор POP служит для выталкивания значений из стека, а PUSH — для записи значений в стек. За командой PUSH следуют 2 числовых выражения — номер оператора и следом номер строки.

Для очистки стека необходимо использовать команду:

PCLEAR

BRANCH

По этой команде компьютер выполняет подпрограмму после окончания каждой программной строки. После этой команды следует числовое выражение, указывающее на начало подпрограммы. Если это значение равно 0, то "MEGA BASIC" действует как обычно в конце строки. Конец подпрограммы определяется оператором ENDPROC.

MTASK

Эта команда позволяет выполнять программу с 2 различных мест. Считается, что программа разделена как бы на 2 части, 1-я начинается после оператора MTASK, а 2-я со строки, указанной в числовом выражении, следующем сразу же за командой MTASK. "MEGA BASIC" выполняет поочередно строки из каждой программы. Если числовое выражение равно 0, то такой многопрограммный режим невозможен.

8. Отладка и редактирование программ

BRANCH, TRON, TROFF, SPEED.

Команда BRANCH очень полезна при отладке программ. Если, к примеру, вас интересует какая-нибудь переменная, вы можете с помощью команды вызвать подпрограмму печати после каждой программной строки для распечатки значений этой переменной.

Другой способ проверить программу — печатать на экране номер очередной выполняемой строки, что возможно по команде TRON. При этом номер выполняемой строки экрана будет в нижнем углу экрана, команда TROFF отменяет такой режим.

Команда SPEED, после которой следует численное выражение, определяющее скорость выполнения программы. Если численное значение равно 0, то это обычная скорость выполнения программы. Но чем больше будет число, тем меньше будет скорость выполнения.

Максимально возможное число = 255. При этом компьютер будет останавливаться после выполнения каждого оператора и ждать, пока пользователь не нажмет клавишу.

AUTO и DELETE

При вводе больших программ весьма полезен бывает ввод номера очередной строки при нажатии клавиши ENTER. Такой режим работы возможен при применении команды AUTO, после которой следуют 2 числовых выражения, разделенных запятой. Первое представляет номер строки программы (первой), второе — шаг нарастания номеров. Остановить режим автоматической нумерации строк можно, используя EXTEND SYMBOL SHIFT 'L'.

Часто случается, что необходимо удалить большой блок программных строк. Эту задачу можно решить, используя команду DELETE. После DELETE следуют 2 числовых выражения — номер 1-ой строки удаляемого блока и номер последней строки этого блока

BRON и BROFF

Желающие защитить свои программы от любопытных глаз найдут полезным возможность делать непригодной клавишу BREAK. Это может быть сделано при применении команды BROFF. Возвращение клавиши BREAK и ее возможностей осуществляется по команде BRON.

Многие версии языка BASIC имеют команду ON ERROR, GO TO, позволяющую не прерывать выполнение программы при возникновении ошибки, а осуществить обход строки с ошибкой.

Аналогичная команда имеется и в "MEGA BASIC". Это команда

RESTART

После команды следует числовое выражение, указывающее номер строки, к которой осуществляется переход в случае возникновения ошибки.

По команде

RESTART OFF

такой режим работы отменяется. Команда RESTART не действует с INTERFASE 1 и по отношению к ошибкам "MEGA BASIC". После обработки ошибки с помощью команды RESTART некоторая полезная информация сохраняется в ячейках памяти:

Таблица 12

адрес	содержание
69873,,69874	номер строки в которой обнаружена ошибка.
69875	оператор внутри строки в котором обнаружена ошибка
59862	код обнаруженной ошибки

9. Звук

"MEGA BASIC" открывает для вас 2 новых пути создания звука на вашем SPECTRUM'е. Это применение оператора PLAY и использование генератора звуковых прерываний (ISG).

PLAY — N,L,S,D,F

где N=0 — звук
N=1 — шум
L — длительность шага
D — число шагов
F — изменение частоты после каждого шага

ISG

Команда PLAY является расширением команды BEEP, однако при использовании "MEGA BASIC" имеется другая возможность производить звук даже при выполнении другой программы. Эта возможность имеется благодаря использованию ISG. При выполнении обычных программ SPECTRUM через каждую 1/50 секунды останавливает выполнение программ и вызывается подпрограмма в машинных кодах, по которой происходит сканирование клавиатуры. При использовании "MEGA BASIC" также производится сканирование клавиатуры, но при этом производить звук. Имеются следующие команды управляющие ISG:

Таблица 13

команда	содержание
SOFF	отключение ISG
SON	включение ISG
SPEP N	если N=0, то данные, хранимые в звуковом буфере, будут воспроизводиться только 1 раз. Если N=1, то каждый раз
SOUND	это новая команда управления звуковой буфером. Формат команды SOUND приведен ниже.

SOUND — N,A,B,C,D

где N=0 — очистка звукового буфера
N=1 — добавление в звуковой буфер новых звуков
A=0 — звук
A=1 — шум
B — шаг звуковой частоты
C — число шагов
D — число повторений одной последовательности

Когда выполняется команда PLAY, ISG автоматически отключается. Для включения применяйте команду SON. Более сложные звуки можно получать, замедлив выполнение программы. Вот короткий пример, показывающий работу ISG:

```
10 SOUND-0,0,1,20,255
20 SREP-1
30 SON
40 MODE-4: STIPPLE-6: FONT-2
50 VDU-/128+RND*15/
60 PAPER RND 7: INK 9
70 GO TO 50
```

10. Машинные коды и "MEGA BASIC"

DOKE и CALL

В "MEGA BASIC" возможно выполнение подпрограмм в машинных кодах, однако прежде чем перейти к этому, отметим 2 момента:

Во-первых, все подпрограммы в машинных кодах должны размещаться в участке памяти до адреса 45 000. Начиная с этого адреса и выше начинается сама программа "MEGA BASIC".

Во-вторых, "MEGA BASIC" позволяет сразу же записывать в память 2-Х байтовое число с помощью команды DOKE. После этой команды следует 2 числа. Первое число указывает адрес, а второе — записываемую в память информацию.

Вызов подпрограммы в машинных кодах осуществляется с помощью команды CALL, за которой следует число, указывающее адрес этой подпрограммы. За адресом может следовать число, являющееся параметром для вызываемой подпрограммы, которое записывается в стек.

11. Фронт-панель

Фронт-панель обеспечивает возможность выполнять изменения памяти и в регистрах Z-80. Фронт-панель активизируется либо по команде MON, либо по команде CONTROLF при выполнении программы. Фронт-панель использует окно 3, а это всегда почти по крайней мере 40 столбцов и 20 строк.

Как только фронт-панель активизируется, вы сразу же увидите столбцы 16-ричных чисел. Слева список регистров с аккумулятором вверху. Звездочка указывает текущий регистр. Справа — текущий участок памяти, на инверсной полосе которого находится выполняемая строка. Манипуляции с данными фронт-панели можно выполнять с помощью ряда однобуквенных команд, после которых может идти 3 16-ричных числа. Числа должны быть записаны полностью, т.е. даже если отдельные разряды равны 0, то это надо указать. Отметим, что N представляет собой 8-ми битовое число, а NN — 16-ти битовое число.

Таблица 14

команда	содержание
K NN	запись в текущий регистр 16-ричного числа.
P	смещение указателя регистра
L NN,NN,NN	перемещение блока памяти, 1 число указывает адрес откуда, 2 число — адрес куда, 3 — длительность блока
M NN	задание адреса текущей ячейки памяти.
S	смещение указателя текущей ячейки
K	продолжение выполнения программы после указателя
I NN,NN,NN	заполнение блока памяти. 1-е число — начальный адрес блока, 2-е — длительность блока, 3-е — значение числа записываемого в блок
J NN	вызов подпрограммы в машинных кодах записанной по адресу NN
ENTER	смещение указателя текущей ячейке
"-"	возврат к предыдущей ячейке

3.3. LASER BASIC

LASER BASIC (LB) — средство расширения стандартного бейсика (в ROM ZX). В него входят графика, мультипликация, движение графических объектов.

LB не создает независимой программы (интерпретатор должен быть в памяти ZX SPECTRUM в качестве резидента). Существует компилятор LB, который значительно увеличивает скорость выполнения операций и не требует присутствия интерпретатора.

Терминология

Спрайт — это управляемый графический объект. Лазер-бейсик позволяет задать до 255 спрайтов, каждый имеет размеры, определяемые пользователем. Размеры спрайтов зависят от объема свободной памяти.

Прилагается программа "генератор спрайтов". После создания спрайтов вы можете записать их на ленту или в кэртридж в одном из двух режимов.

1) спрайты могут использоваться только с генератором спрайтов.

2) спрайты могут использоваться только с лазер-бейсиком.

Два набора готовых спрайтов (в режиме 2) имеются в приложении — это SPRITE 2A и SPRITE 2B.

Окно экрана — это часть его, заданная четырьмя переменными:

COL — от 0 до 31; ROW — от 0 до 23;
LEN — от 1 до 32; HGT — от 1 до 24;

COL и ROW указывают на номер столбца и номер строки относительно левого верхнего угла экрана.

Например:

```
.ROW=5:.COL=6:.HGT=4:
.LEN=3:.INVV
```

Окно спрайтов — это часть спрайта заданная переменными: SPN, SCL, SRW, HGT, LEN.

SPN — указывает на номер спрайта.
SCL и SRW — определяет столбец и строку в спрайте.
HGT и LEN — размер окна в спрайте.

Если заданное таким образом окно выходит за пределы спрайта, то команда не проходит, хотя сообщение об ошибке не выдается.

Область спрайта — это область памяти, в которой хранятся все заданные спрайты. Ее верхняя граница — 56575, нижняя граница — плавающая. Узнать ее можно:

```
PRINT PEEK(62464)+256*PEEK(62465) или
LET X=PEEK(62464):PRINT X
```

RAMTOP

Область спрайта распространяется вниз и не должна пересекать границу RAMTOP, которая определяется:

```
PRINT PEEK(23730)+256*PEEK(23731) или
LET X=PEEK(23730):PRINT X
```

При каждом задании спрайта расходуется 9*HGT*LEN+5 байтов.

Обычно не стоит волноваться о выходе за пределы RAMTOP, кроме тех случаев, когда спрайты создаются во время работы программы через.ISPR или.SPRT. Проверку можно выполнить приведенными выше вычислениями.

Мы рекомендуем сначала создавать спрайты в генераторе спрайтов (ГС), а затем загружать в отведенную для них область одним из следующих способов:

```
CLEAR (N-1)
LOAD "FN" CODE (N)
POKE 62464,N
```

где "N" — адрес начала области спрайтов, а "FN" — имя файла, данное файлу спрайтов при выгрузке его из ГС. Стартовый адрес области спрайтов — это низший байт из файла спрайтов, он также задан ГС.

Пиксели

Элементарный блок графики размером в один стандартный символ содержит 8*8 точек или пикселей, каждый из которых представлен битом в памяти.

Пиксель — это точка. Если он включен, он имеет цвет INK, а если выключен — PAPER.

Атрибуты

Атрибуты цвета INK или PAPER блока, а также параметры BRIGHT и FLASH управляются специальными байтами и называются атрибутами.

Операции с экраном

Есть операции, которые выполняются с частью экрана. Область экрана при этом называется окном. Эти операции включают в себя: скроллинг, инверсию, построение симметричных изображений и т.п.

Все команды этой группы имеют на конце букву V:

.SRIV,.INVV,.MIRV

При этом, если окно выходит за пределы экрана, оно автоматически подгоняется под экран.

Операции между экраном и спрайтом

Размеры спрайта используются в качестве размеров окна; COL и ROW используются для задания координат верхнего угла окна. Таким образом, эти функции выполняются с переменными SPN, COL, ROW.

Команды этой группы начинаются с "PT" или "GT".

Например: .GTBL,.PTXR,.PTND и т.п.

Операции со спрайтами

Они аналогичны операциям с окнами. Используемая переменная — SPN. Команды имеют окончание "M".

Операции между окном экрана и окном спрайта

Как и ранее,.ROW,.COL,.HGT,.LEN определяют окно экрана, но в этом случае.CSL и.SRW используются для определения положения окна в спрайте. Измеряются.SRW и.SCL в символах.

.SCL — слева,.SRW — вверху. Если.SRW+.HGT больше, чем высота спрайта или.SCL+.LEN больше, чем его ширина или .LEN-

COL больше 32 или.HGT+.ROW больше 24, то команда не исполняется. Команды этой группы начинаются с ''PW'' или с ''GW''.

Операции между спрайтом и окном экрана

Номера спрайтов хранятся в SP1 (для первого спрайта) и в SP2 (для второго спрайта с окном). Размеры окна — это размеры спрайта без окна, а положение окна в спрайте (SP2) задается .SCL и.SCW.

Если спрайт SP1 пересекает границу спрайта SP2, то команда не выполняется. Команды этой группы начинаются с ''PM'' или с ''GM''.

Операции спрайт-спрайт

Это команды, по которым спрайт трансформируется и результат заносится во второй спрайт. В этой группе только две команды:

.SPNM и .DSPM

Пустой спрайт: он не содержит данных для экрана. Его можно использовать например для хранения подпрограммы в машинных кодах, массива; он может быть использован для процедуры определения столкновения спрайтов.

Редактирование и запуск программ

Редактор ZX расширен настолько, что дополнительные команды LB проходят проверку на синтаксис при вводе программы. Все команды LB начинаются с точки, за которой идут четыре символа, изображенные заглавными буквами. Присваивания, такие, как.COL или.ROW, не должны содержать пробелов между ''='' и именем переменной. Дополнительные функции записываются через ''?'' с последующими тремя символами.

Например: COL, ROW и т.п. Они могут использоваться только для передачи значения переменной и не могут использоваться как часть выражений или с оператором PRINT.

Например:

```
LET X=?COL X:LET Y=?KBF D:LET Z=?GET (правильно)
LET X=?COL*3+Y:PRINT ?COL (неправильно)
```

Начало работы с программой

Сначала загрузите LB. Остановите ленту по указанию. Программа готова к работе после появления на экране меню. Теперь загрузите файл ''SPRITE 2A''. Он загружается набором опции ''3'' в меню и нажатием ''PLAY'' на магнитофоне. После загрузки спрайтов вы можете работать с LB (опция 1). Вы сможете пользоваться своими собственными спрайтами, выполненными в программе ''SPRITE GENERATOR''.

Все ключевые слова стандартного бейсика вводятся как обычно.

Наберите:

```
10 REM THIS IS YOUR LASER PROGRAM
20 .COL=1
30 .ROW=1
40 .HGT=20
50 .LEN=30
60 .INVV
RUN (или GOTO 10)
```

Сервисные команды .RNUM,.REMK,.TRON,.TROF

.RNUM служит для перенумерации строк: RNUM 102,100,5

Строки до 102 не изменяются, строке 102 присвоится номер 100, а последующие строки пронумеруются с шагом 5. Шаг по умолчанию равен 10. Новый номер строки по умолчанию принимается равным старому, первый номер по умолчанию принимается равным 0.

.REMK применяется для удаления всех строк REM с целью экономии памяти.

.TRON,.TROF — это отладочные операторы для пошагового выполнения программы. После встречи.TRON интерпретатор останавливается на каждой строке программы, печатает ее и ждет нажатия клавиши. После.TROF этот режим отключается.

Графические переменные

Способ, которым параметры передаются графическим процедурам, предназначен для повышения скорости расчетов. Каждая команда с графика может использовать набор графических переменных (из десяти). Всего допускается до 16 наборов. Из них можно делать индивидуальную выборку по форме:.SET=<EXP>, где <EXP> может быть выражением из бейсика от 0 до 15. Эти 10 переменных:

ROW — вертикальная координата (0-23) верхний ряд имеет ROW=0

COL — горизонтальная координата (0-31) левый столбец имеет COL=0

HGT — высота текущего окна (1-24)

LEN — длина текущего окна (1-32)

SRW — вертикальная координата внутри спрайта (0-(HGT-1))

SCL — горизонтальная координата внутри спрайта (0-(LEN-1))

NPX — выражает направление и величину вертикального скроллинга. Положительное значение — вверх, отрицательное — вниз. Единица измерения — пиксель (не символ). Диапазон от -128 до +127.

SPN — номер спрайта (1-255). Применяется для тех команд, которые работают с одним спрайтом.

SP1 — для операций между спрайтом и окном другого спрайта. Здесь содержится номер первого спрайта.

SP2 — для тех же операций. Содержит номер спрайта, имеющего окно.

Пример:

```
10 .SET=0:.SPN=4:.ROW=0:.COL=-4
20 BORDER 1:BRIGHT 1:INK 6:PAPER 1:CLS:.ATOF
30 FOR I=-4 TO 32
40 .PTXR:.COL=I+1:.PTXR
50 PAUSE 40
60 NEXT I
70 STOP
```

10 Выбирается графический набор 0. Спрайт N4.

20 Устанавливаются атрибуты экрана, он очищается, флаг атрибутов отключается.

30 Организуется цикл для движения спрайтов N4 по горизонтали от координаты -4 до +32.

40 Удаляется старый спрайт, помещается новый.

50 Пауза для плавности движения.

Можно расширить эту программу для движения двух спрайтов в противоположных направлениях. Второй спрайт будем получать зеркальным отображением первого. Здесь надо использовать два набора переменных.

```
20  DEF FNA#(X,Y):.SET=X:.SPN=4:.ROW=Y
30  .RETN
40  INK 6:PAPER 0:BORDER 0:BRIGHT 1:CLS:.ATOF
50  .PROC FNA#(0,0):.PROC FNA#(1,1)
55  .COL=32:.SET=X:.COL=-4
60  FOR I=1 TO 32
70  .SET=1:.MIRM:.PTXR:.COL=28-I:.PTXR:.MIRM
80  PAUSE 2
90  NEXT I
100 GOTO 40
```

20 и 30 задают процедуру, которая устанавливает переменные для соответствующих наборов. При первом проходе SET=0 и ROW=0. При втором проходе SET=1 и ROW=1. Это значит, что оба набора идентичны, за исключением ROW.

40 — дважды рассчитывается процедура A#

50 — устанавливает столбцы

60 — начало цикла

70 — перемещает спрайт слева направо, используя при этом переменные из набора 0; строит зеркальное отображение спрайта и перемещает его слева направо.

Изменение значений переменной

Кроме необходимости назначения графических переменных, приходится менять их текущие значения. Для этого имеются 11 функций: ?COL ?ROW ?LEN ?HGT ?SP1 ?PSP ?NPX ?SCL ?SRW ?SET

Этими функциями текущим значениям переменных присваиваются значения синклеровских системных переменных.

Например: LET X=?COL: LET ROW=?ROW: LET SET=?SET

Все эти функции не могут быть частью выражений или параметрами оператора.

Процедуры обработки спрайтов

.SPRT — нужен для задания новых спрайтов. Область спрайтов поднимается на величину: 9*HGT*LEN+5 байт. Если номер этого спрайта был ранее задан, то сначала стирается старый спрайт (при этом нижняя граница неизменна, а верхняя — понижается), а затем размещается новый.

Если спрайт задается впервые, то данные в него должны быть внесены оператором.

Например:.GTBL

параметры команды	назначение
.SPN	номер помещенного спрайта (1-255)
.LEN	длина спрайта в символах (1-255)
.HGT	высота спрайта в символах (1-255)

Поскольку эта команда расширяет область спрайтов вверх, она имеет ограниченную применимость. Чаще используются командой

.ISPR. Неаккуратным использованием можно испортить программу.

.ISPR — то же самое, что и.SPRT, но здесь область спрайтов развивается вниз. Если спрайт с данным номером уже есть, то появляется сообщение об ошибке. Если при использовании этой команды начало области спрайтов опустится ниже RAMTOP, то программа может испортиться.

.WSPR — служит для стирания существующего спрайта. Верхняя граница области понижается. Обычно для этого применяют.DSPR.

Пример:.SPN=1:.WSPR

параметр	назначение
.SPN	номер спрайта, подлежащего стиранию (1-255)

.DSPR — аналогична предыдущей. При стирании спрайтов нижняя граница области повышается.

Горизонтальный скроллинг экрана выполняется на 1, 4 или 8 пикселей влево или вправо с возвратом или без него.

Команды

.WL1V — скроллинг влево на 1 пиксель с возвратом
.WR1V — скроллинг вправо на 1 пиксель с возвратом
.SL1V — скроллинг влево на 1 пиксель без возврата
.SR1V — скроллинг вправо на 1 пиксель без возврата
.WL4V,.WR4V,.SL4V,.SR4V — то же, на 4 пикселя
.WL8V,.WR8V,.SL8V,.SR8V — то же, на 8 пикселей

параметры команды	назначение
.COL	номер левого крайнего столбца (0-31)
.ROW	номер строки (0-23)
.LEN	длина окна (1-23)
.HGT	высота окна (1-24)

Описанные выше команды работают с экраном, выполняя скроллинг окна, размеры которого —.LEN и.HGT, а координаты —.CO', и.ROW. Скроллинг на произвольное количество пикселей выполняется последовательной подачей ряда описанных команд.

Вертикальный скроллинг экрана

Работа этих команд похожа на работу горизонтального скроллинга, но в добавление к параметрам окна добавляется переменная.NPX, которая задает размер и направление скроллинга в пикселях.

.NPX=-1 — на один пиксель вниз
.NPX=1 — на один пиксель вверх

Команды:

.WCRV — вертикальный скроллинг с возвратом
.SCRV — вертикальный скроллинг без возвратом

Вертикальный скроллинг пикселей или атрибутов экрана или спрайтов требует пространства в буфере. Необходимое пространство определяют перемножением NPX и LEN. Его объем не должен превышать 256 байт, поскольку в качестве временного хранилища используется память буфера принтера.

Скроллинг атрибутов экрана похож на скроллинг пикселей, но выполняется на ширину символов и всегда с возвратом.

Команды:

.ATLV — скроллинг атрибутов влево
.ATRV — скроллинг атрибутов вправо
.ATVV — скроллинг атрибутов вверх
.ATDV — скроллинг атрибутов вниз

Параметры:

.COL (0-31).ROW (0-23)
.HGT (1-32).LEN (1-24)

Горизонтальный скроллинг спрайтов

Горизонтальный скроллинг спрайтов выполняется на 1, 4, 8 пикселей влево, вправо, с возвратом и без.

Команды:

.WL1M — скроллинг влево на 1 пиксель с возвратом
.WR1M — скроллинг вправо на 1 пиксель с возвратом
.SL1M — скроллинг влево на 1 пиксель без возврата
.SR1M — скроллинг вправо на 1 пиксель без возврата
.WL4M,.WR4M,.SL4M,.SR4M — то же, на 4 пикселя
.WL8M,.WR8M,.SL8M,.SR8M — то же, на 8 пикселей
Параметр .SPN — номер спрайта.

Вертикальный скроллинг спрайта:

.WCRM — вертикальный скроллинг с возвратом
.SCRM — вертикальный скроллинг без возвратом

Параметры:

.SPN — номер спрайта.
.NPX — направление и количество пикселей скроллинга (128-127)

Скроллинг атрибутов спрайта:

4 команды для скроллинга в четырех направлениях на 2 символа с возвратом:

.ATLM — влево
.ATRM — вправо
.ATUM — вверх
.ATDM — вниз
параметр .SPN — номер спрайта.

Операторы GET и PUT, группа 1

PUT — помещают спрайт на экран или в другой спрайт.

GET — наоборот, берут данные с экрана или спрайта и помещают их в спрайт.

Есть 3 группы операторов GET и PUT:

Первая группа более быстродействующая, выполняет операции между целыми спрайтом и предварительно заданным окном. Всем этим командам предшествуют "GT" и "PT". Эта первая группа не имеет специальных команд для перемещения пиксельных данных и атрибутики, а вместо этого имеет переключатель атрибутов (см..ATIN,.ATOF) для перемещения пикселей без атрибутов или с ними.

.GTBL — поместить блок из окна экрана в спрайт
.GTOR — наложить (OR операция) блок окна экрана на спрайт
.GTXR — наложить (XOR операция) блок окна экрана на спрайт
.GTND — наложить (AND операция) блок окна экрана на спрайт
.PTBL — поместить спрайт в окно экрана
.PTOR — наложить (OR операция) спрайт на окно экрана
.PTXR — наложить (XOR операция) спрайт на окно экрана
.PTND — наложить (AND операция) спрайт на окно экрана

Параметры:

.COL — номер левой колонки требуемой позиции экрана (0-31)
.ROW — номер верхнего ряда (0-23)
.SPN — номер спрайта

Пример:.SPN=39:.ROW=1:.COL=1:.PTBL

Поместите спрайт номер 39 на экран в позицию 1.1. Наберите.ATOF (отключение переноса атрибутики), теперь:

.ROW=S:.PTBL, спрайт номер 39 появится на экране в другой позиции, но без атрибутов.

Наберите .ATOF:.PTBC и спрайт появится с атрибутами.

Оператор GET берет данные из окна в заданной позиции, размеры которого равны размеру спрайта и помещает их в спрайт.

Операторы GET и PUT, группа 2

Эти команды выполняют операции между окнами экрана и окном спрайтов.

.ATON и.ATOF имеют то же значение, как и ранее.

Здесь нужны 4 новых параметра для задания столбца и строки, определяющих положение окна в спрайте, а также высоты и длины этого окна. Этим командам предшествует "GW" или "PW".

.GWBL — перенос блока с экрана в окно спрайта
.GWOR — наложение по принципу OR
.GWXR — наложение по принципу XOR
.GWND — наложение по принципу AND
.PWBL — перенос блока из окна спрайта на экран
.PWOR — наложение по принципу OR
.PWXR — наложение по принципу XOR
.PWND — наложение по принципу AND
.GWAT — перенос атрибутов блока с экрана в окно спрайта
.PWAT — перенос атрибутов блока из окна спрайта на экран

Параметры:

.COL — номер столбца позиции на экране (0-31)
.ROW — номер строки позиции на экране (0-23)
.SCL — координата левой границы окна спрайта 0 — длина спрайта -1
.SRW — координата верхней строки окна спрайта 0 — до высоты спрайта -1
.HGT — высота окна (1-24)
.LEN — длина окна (1-32)
.SPN — номер спрайта (1-225)

Операторы GET и PUT, группа 3

Группа содержит команды для операций между спрайтом и окном в другом спрайте.

Эти команды начинаются с "PM" или с "GM".

.GMBL — перенос спрайта в окно другого спрайта
.GMOR — наложение OR
.GMXR — наложение XOR

.GMND — наложение AND
.PMBL — перенос окна спрайта в другой спрайт
.PMOR — наложение OR
.PMXR — наложение XOR
.PMND — наложение AND
.GMAT — перенос атрибутики спрайта в окно другого спрайта
.PMAT — перенос атрибутики окна спрайта в другой спрайт
Параметры:
.SP1 — номер первого спрайта (1-255)
SP2 — номер второго спрайта (1-255)
.SCL — координата левой границы окна спрайта (1 — длина спрайта).
.SRW — координата верхней строки окна спрайта (1 — высота спрайта).

Оператор .MOVE

Используется для эффекта мультипликационного перемещения спрайтов. Операция выполняется только по XOR или.XOR. Поэтому, прежде чем перемещать спрайт по экрану, его надо сначала туда поместить через PTXR.

Параметры:
.COL — номер столбца спрайта, подлежащего перемещению (0-31)
.ROW — номер его строки (0-23)
.HGT — величина перемещения по вертикали ((-24)-(+24))
.LEN — длина перемещения по горизонтали ((-32)-(+32))
.SP1 — номер спрайта, подлежащего перемещению (1-225)
.SP2 — его номер после перемещения (...225)

По команде MOVE берется через PUT спрайт SP1 с позиции экрана, заданной ROW и COL и помещается спрайт SP2 в позицию COL+LEN, HGT+R

Операторы .ATON и .ATOF

Эти две команды определяют, надо ли переносить атрибуты спрайтов при выполнении команд PUT и GET.
.ATON — включение переноса атрибутов
.ATOF — выключение переноса атрибутов

Изменение изображения

.INVV — инвертирование окна экрана.

Инвертирование — включение/выключение пикселей. Цвета INK и PAPER меняются. Параметры для окна:.COL, .ROW,.HGT,.LEN.

.INVM — такая же операция, но над спрайтом, номер которого задан SPN. Параметры:.SPN.

.MIRV — содержание окна, зеркально отображенного относительно вертикальной оси, проходящей через его центр.

.MIRM — содержание спрайта, номер которого задан SPN зеркально отображается относительно оси, проходящей через его центр.

.MARV — зеркальное отображение атрибутов окна относительно вертикальной оси. Параметры:.COL,.ROW,.HGT,.LEN

.MARM — тоже самое для спрайта, параметры:.SPN.

.SPNM — поворот спрайта SP2 на 90 градусов по часовой стрелке. После поворота присваивается номер SP1.

Пример: если первый имел размер 3*8, то второй — 8*3. Спрайт SP1 должен быть предварительно очищен командой.CLSM.

.DSPM — увеличение размеров спрайта SP1. После увеличения присваивается номер SP1. SP1 должен иметь точно вдвое большие размеры по сравнению с SP2.

Другие команды

.SETV — установка постоянных атрибутов INK и PAPER в заданном окне.

.SRTM то же, но для спрайта, заданного в SP2. Параметры: SPN

.CLSV — очистка заданного окна экрана (атрибуты не задействованы).

.CLSM — то же, но для спрайта. Параметры: SPN

.ADJM — эта команда служит для настройки переменных: COL, ROW, HGT, SCL, SRW, SPN, таким образом, чтобы отдельный спрайт мог быть взят командами PUT или GET второй группы. Значения ROW, HGT, LEN, SCL, SRW, могут меняться при исполнении команды. До исполнения SCL и SRW должны быть равны 0.

Параметры: .SPN — номер спрайта, который надо взять;
.COL — номер столбца;
.ROW — номер строки;
.SCL — установить в 0 до исполнения команды;
.SRW — то же;
.ADJV — то же, но не для спрайта, а для окна. Параметры: .COL,.ROW,.HGT,.LEN.

Присваивание

Их всего 11. Эти команды присваивают графическим переменным значение выражений бейсика. Использовать команды режима "E" нельзя.

.COL=	.ROW=	.HGT=	.LEN=
.SP1=	.SP2=	.SPN=	.SRW=
.SCL=	.NPX=	.SET=	

Дополнительные функции
Их всего 16:

?COL	?ROW	?NGT	?LEN
?SPN	?SP1	?SP2	?SET
?SCL	?SRW	?NPX	?KBF
?SCV	?SCM	?TST	?PEK

Их форма: LET VAR=?FUN, где VAR — переменная бейсика, ?FUN — одна из 16-ти функций.

Эти функции не должны быть частью выражений бейсика.

?COL — выдает значение переменной COL и т.п.

?KBF — эта функция предназначена для определения нажатия клавиши. Она проверяет заданную клавишу и выдает ненулевой результат при нажатии.

Переменные ROW и COL служат для задания полуряда и столбца на клавиатуре.

ряд	клавиши
1	от CS до V
2	от A до G
3	от Q до T
4	от ! до 5
5	от 0 до 6
6	от P до Y
7	от ENTER до H
8	от SPACE до B

Отсчет COL ведется по рядам снаружи внутрь.

?SCV — по этой команде элементарный блок экрана (8x8) положение которого задано переменными COL и ROW, проверяется на предмет включения пикселей. Если есть включение пикселя, то выдается результат, отличный от 0, в противном случае 0. Эта функция применяется часто, например, для определения момента столкновения спрайтов. Для этого проверяется состояние пикселей элементарного блока (8x8), находящегося впереди движущегося спрайта.

?SCM — по этой команде проверяются пиксели спрайта, номер которого содержится в SPN. Если есть включенные пиксели, выдается ненулевой результат и наоборот.

?TST — отыскивается спрайт, номер которого хранится в SPN. Если его нет, то выдается сообщение об ошибке. Если он найден, выдается его адрес, а переменные HGT и LEN принимают значения, равные его размерам.

Спрайты хранятся в памяти в следующем формате:
байт 1 — номер спрайта
байт 2 — младший байт адреса следующего спрайта
байт 3 — старший байт адреса следующего спрайта
байт 4 — длина спрайта
байт 5 — высота спрайта
8*HGT*LEN — данные о состоянии пикселей
HGT*LEN — атрибуты
Итого: 9*HGT*LEN+5 байтов

?PEK — это 16-ти битная версия оператора PEEK. В отличие от прочих функций, она имеет при себе выражение в скобках, в котором содержится адрес.

Пример:

```
LET X=?PEK(64264) эквивалентно
LET X=PEEK(64264)+256*PEEK(64265)
```

Дополнительные команды

POKE X,Y — это 16-ти битная версия POKE из бейсика ZX. Она помещает младший байт числа Y в X, а старший байт Y в X+1.

Диапазон X — (0-65534), Y — (0-65535).

Процедуры

Одной из сильных сторон языков Паскаль. BASIC BBC, позволяющих выполнять структурное программирование, является возможность использования процедур. Часто бывает надо передать переменную в программу так, чтобы не были нарушены значения переменных из других частей программы. Для этого нужны локальные переменные. В бейсике этого понятия нет.

Локальные и глобальные переменные

При задании процедуры ее описание содержит список параметров, которые могут быть строками, числами, но не массивами.

Пример: 10 DEF FN A$# (X,Y,Z,A$,B$)

Надо помнить, что ключевое слово DEF FN набирается не по буквам. Здесь параметры X, Y, Z, A$, B$ — локальные переменные, т.е. значения, полученные ими внутри процедуры A, не повлияют на их значения в других частях программы. Все другие переменные, задействованные в процедуре, будут глобальными.

Вызов процедур

Для выполнения приведенной выше процедуры используется оператор: PROC FN A$#(3,2,K,"HELLO",B$)

Здесь: FN — ключевое слово на клавиатуре. Тогда локальные переменные примут значения: X=3; Y=2; Z=K; A$="HELLO"; B$=B$

Процедура должна всегда заканчиваться командой.RETN, после чего управление передается оператору, следующему за вызовом процедуры. Исполнение команд идет тем быстрее, чем ближе к началу программы расположено их определение. Разрешается применять до 52-х процедур (26 букв латинского алфавита +26 букв со знаком $). Чтобы отличать процедуры от функций пользователя, после процедуры должен стоять знак #

Пример: DEF FN C$#=0: DEF FN C$#=0...

Выполнение процедуры

Определение процедуры может включать в себя вызов другой процедуры, но при этом помните, что значения локальных переменных для внутреннего вызова распространяются и на первый (внешний) вызов.

Пример: процедура для печати на экране слова "HELLO":

```
10 DFN FN A$#()
20 PRINT AT X,Y;"HELLO"
30 .RETN
```

Скобки пустые, поскольку никакие параметры не нужны. Вызывается эта процедура командой. PROC FN A$#().

Для задания координат надписи (X,Y):

```
10 DEF FN A$#(X,Y)
20 PRINT AT X,Y;"HELLO"
30 .RETN
```

Чтобы напечатать "GOOD", теперь вызов:

```
.PROC FN A$#(5,8,GOOD).
```

Включение процедур в LB затрагивает систему защиты собственного стека ZX, поэтому возвраты из подпрограмм RETURN после GOSUB и.RETN после PROC должны выполняться в правильном порядке.

Техника программирования

Движение спрайтов

Скроллинг экрана под управлением от клавиатуры используется, когда спрайт движется по окну экрана, не содержащему других спрайтов или данных.

Выполним скроллинг окна на 1 пиксель, используя функцию ?KBF.

```
10 INK G: PAPER 0: BRIGHT 1: BORDER 0: CLS
20 .COL=14 :.ROW=10 :.SPN=2 :.PTBL
30 .SET=5 :.COL=5 :.ROW=4
40 .SET=6 :.COL=3 :.ROW=5
50 .SET=7 :.COL=0 :.ROW=10 :.LEN=32 :.HGT=2
60 .SET=5 : LET KB=?KBF : IF KB=0 THEN.SET-7
:.WL1V
70 .SET=6 : LET KB=?KBF : IF KB=0 THEN.SET-7
:.WL1V
80 GOTO 60
```

Использование PUT

2-ой метод состоит в помещении спрайта на экран оператором ROW, меняя при этом значения ROW и COL. Спрайт, используемый для этого метода, должен иметь вокруг себя бордюр шириной в один символ.

Возьмем спрайт #30 размером 3*3.

Сначала очистим данные в спрайте:.SPN=30 :.CLSM

Теперь поместим в его центр спрайт номер SP1 (разм. 1*1) .SP1=50 :.SP2=30 :.SCL=1 :.GMBL

Программа для его перемещения с помощью курсорных клавиш (нажатие их проверяется с помощью ?KBF):

```
10 .ATOF : INK 5 :BRIGHT 1 : PAPER : BORDER 0 :
CLS
20 .SET=1 :.ROW=4 :.COL=5
30 .SET=2 :.ROW=5 :.COL=5
40 .SET=3 :.ROW=5 :.COL=4
50 .SET=4 :.ROW=5 :.COL=3
60 .SET=7 :.ROW=10 :.COL=13 :.SPN=30
70 LET X=13 : LET Y=10
80 .SET=1 : LET KB=?KBF : IF KB<>0 THEN LET X=X-1
90 .SET=2 : LET KB=?KBF : IF KB<>0 THEN LET Y=Y-1
100 .SET=3 : LET KB=?KBF : IF KB<>0 THEN LET
X=X+1
110 .SET=1 : LET KB=?KBF : IF KB<>0 THEN LET
Y=Y+1
120 IF X>30 THEN LET X=30
130 IF X<-1 THEN LET X=22
140 IF Y>22 THEN LET Y=22
150 IF Y<-1 THEN LET Y=-1
160 .SET=7 :.COL=X :.ROW=Y :.PTBL
170 GOTO 80
```

Минус этого метода в том, что данные, имеющиеся на экране, будут стираться в результате прохождения спрайта по ним. Для устранения этого применяют логические операции. В LB их 3: OR, XOR, AND.

Если два спрайта накладываются по принципу OR, то в результате будут включены те пиксели, которые были включены в одном из этих спрайтов или в обоих вместе.

По принципу AND результат будет содержать включенными только те пиксели, которые были включены в первом и во втором спрайтах одновременно.

По принципу XOR — включены те, которые были включены в одном из двух, но те, которые были включены в обоих вместе будут включены.

Использовать OR или XOR при движении спрайта можно через оператор.PTOR или.PTXR. Так, через.PTOR данные экрана накладываются после перемещения спрайта в новую позицию, что исключает стирание. Через.PTXR спрайт снимается с экрана и снова помещается рядом, что также дает движение спрайта без стирания основного изображения.

Использование команды .MOVE

После задания значения COL и ROW задается их приращение на каждом шаге HGT и LEN. Каждое исполнение команды.MOVE приводит к увеличению COL и ROW.

```
10 INK 0 : PAPER 6 : BRIGHT 1 : BORDER 7 : CLS
:.ATOF
20 FOR N=1 TO 100
30 LET X=INT(RND*32) : LET Y=INT (RND*22)
40 PRINT AT X,Y ; "*"
50 NEXT N
60 LET DR=1 :.HGT=DR : LET DC=1 :.LEN=DC
70 .SP1=50 :.SP2=50 : LET R=10 :.ROW=R : LET C=13 :
.COL=C : .SPN=50 :.PTXR
80 .MOVE
90 LET C=?COL : LET R=?ROW
100 IF C=29 OR C=0 THEN LET DC=DC-1 :.LEN=DC :
BEEP 0.0,5
110 IF R=19 OR R=0 THEN LET DR=DR-1 :.HGT=DR :
BEEP 0.0,5
120 GOTO 80
```

В LB нет команд для установки спрайтов на экран с точностью до пикселя для перемещения спрайтов по экрану с более высокой разрешающей способностью.

Если надо переместить спрайт слева направо с шагом 2 пикселя, а скроллинг экрана при этом делать нельзя, то нужно сначала с помощью программы ГС создать 4 спрайта, каждый из которых сдвинут относительно предыдущего вправо на 2 пикселя. Если их пронумеровать от одного до четырех, то программа будет выглядеть так:

```
10 FOR C=0 TO 31
20 .COL=C:.SPN=1:.PTBL:.SPN=2:.PTBL
30 .SPN=3:.PTBL:.SPN=4:.PTBL
40 NEXT C
```

Определение столкновения спрайтов

Для этого служат команды ?SCV и ?SCM.

?SCV служит для проверки позиции символа на экране. Если в позиции, заданной COL и ROL, нет ничего — выдается 0, в противном случае — число. Иногда пользуются более мощной, но более медленной командой ?SCM. Она проверяет спрайт, номер которого находится в SPN и выдает 0, если там нет включенных пикселей, или число в противном случае. Эта команда обычно используется для выполнения следующих функций:

— для проверки столкновения символов;
— для определения, что находится в заданном квадрате;
— для отыскания заданного объекта на экране.

Проверка на "столкновение" наиболее часто проводится для определения наложения движущегося спрайта на какой-либо графический символ, находящийся на экране.

В основе лежит следующий метод. Примем часть экрана в том месте, где предполагается размещение спрайта, за квази-спрайт. Затем вызвать.PTND для спрайта и квази-спрайта, а затем проверить ?SCM. Если значение не 0, то столкновение (коллизия) не произошло. Но здесь есть недостаток в том, что если новый спрайт перекрывает старый, то старый спрайт будет удален до того, как начнется вышеописанная процедура, и, соответственно, до помещения нового спрайта на экран. Наилучшее решение — работать через XOR, т.е. через.PTXR. Надо взять содержимое окна, наложить по XOR его на содержимое старого спрайта в памяти (чтобы удалить все данные из спрайта), затем сделать проверку на коллизию с немедленным перемещением спрайта на экран.

Если коллизия произошла, часто бывает полезным узнать, с чем же спрайт столкнулся. Для начала предположим, что окно экрана, которое мы обследуем, содержит известный набор символов и никаких прочих данных там нет. Метод состоит в том, чтобы загрузить квази-спрайт с рабочим объектом и затем сравнить его со спрайтами из набора.

Для этого нужно только наложить проверяемый спрайт на квази-спрайт по XOR, затем сделать ?SCM. Если в результате 0, то они идентичны, если нет — то сделать XOR для другого спрайта и т.д.

И, наконец, рассмотрим случай, когда исследуемый объект содержит кроме одного из возможных спрайтов еще целый ряд данных. В этом случае проверяемые объекты сначала накладываются по AND для удаления всех данных, затем XOR, затем ?SCM.

Скроллинг пейзажа

Эта операция настолько часто встречается в видео-играх, что ее надо рассмотреть отдельно.

Первое — это двигать ровно столько, сколько нужно и не больше. Например, если у вас есть горный пейзаж, причем горы занимают в нем три верхних ряда, то надо эти три ряда и двигать.

Самый простой и эффективный метод выполнения плавного перемещения — это пожертвовать одним столбцом экрана для организации обмена со спрайтом, подлежащим скроллингу.

Предположим, что скроллинг идет на пиксель и горизонтального перемещения атрибутов нет. Фактически все равно, каким столбцом атрибутов мы пожертвуем: самым правым (31) или самым левым (0). Давайте используем столбец 0. Нам надо задать окно в левом столбце шириной в один символ и высотой в 3 символа, задав в нем одинаковые INK и PAPER, т.е. пиксели в нем видны не будут. Это задается командой.SETV.

Теперь 31 столбец спрайта помещаются на активную часть экрана по команде.RWBL. Если скроллинг идет вправо, то в это вспомогательное окно помещается 32-ой столбец спрайта, затем выполняется скроллинг на 1 пиксель 8 раз, в дополнительный столбец помещается 33-й столбец спрайта и т.д.

Если скроллинг идет влево, то вспомогательное окно организуется на правом краю экрана.

Перезадание набора символов и символов графики пользователя

21 символ UDG (графика определяемая пользователем), имеющиеся в стандартном бейсике, здесь не допустимы. Существует системная переменная CHARS, указывающая на адрес в памяти, который на 256 байт меньше, чем начало набора символов в ROW (от 32-го символа ASCII до 128-го). Пользователь может заслать новое значение в CHARS, чтобы эта переменная указывала на другую область памяти, где будет размещен новый набор символов. Очевидным местом для хранения набора является область спрайтов.

Наборы переменных

Применение 16 наборов переменных значительно повышает скорость программных вычислений. Если надо перемещать 4 окна на экране, можно задать параметры для каждого окна в виде набора переменных.

Например:

```
.SET=1 : .HGT=5 : .LEN=5 : .ROW=0 : .COL=0
.SET=2 : .HGT=4 : .LEN=6 : .ROW=0 : .COL=5
.SET=3 : .HGT=6 : .LEN=4 : .ROW=0 : .COL=12
.SET=4 : .HGT=7 : .LEN=3 : .ROW=0 : .COL=18
```

Теперь, чтобы выполнить скроллинг для каждого окна, достаточно знать:

```
.SET=1 : .WRIV : .SET=2 : .WL2V : .SET=3 : .WL8V :
.SET=4 : .WRIV
```

Загрузка и запись программ в LB

Программы могут быть загружены или выгружены как подачей прямой соответствующей команды, так и из программы.

Запись и загрузка прямой командой

Для ленты для неавтостартующей программы напрямую: SAVE "FILENAME"

Для микродрайва: SAVE *"M";N;"FILENAME".

Если хотите, можете помещать несколько SAVE и LOAD в одну инструкцию.

Например: SAVE "FILENAME":LOAD "FILENAME" CODE

Если вы хотите подать прямую команду стандартного бейсика, то действуйте обычным порядком, а если команда из расширенного интерпретатора, то между ней и LOAD или SAVE должно стоять RANDOMIZE USR 58841.

Например:

```
SAVE *"M";N;"FILENAME":LOAD
*"M";N;"FILENAME" CODE:
RANDOMIZE USR 58841:.REMK:.RNUM
```

или

```
SAVE *"M";N;"FILENAME", LOAD
*"M";N;"FILENAME" CODE
```

с последующей подачей

```
.REMK:.RNUM
```

Запись автостартующей программы прямой командой

Запись производится следующим образом:

```
SAVE "FILENAME" LINE N
SAVE *"M";N;"FILENAME" LINE N
```

Записанная таким образом программа должна выполнять одну из нижеописанных до того, как встретится какая-либо команда из расширенного интерпретатора :RUN, GOTO, GOSUB — для ленты и микродрайва с последующим RANDOMIZE USR 58841.

Запись программы из программы

Программы, записанные из программы, могут быть загружены тоже только из программы. Программы, записанные прямой командой, загружаются также прямой командой.

Прежде чем пользоваться одной из приведенных схем загрузки/выгрузки, надо загрузить и запустить программу LB (она должна быть резидентной).

Программа LASER BASIC COMPILER

Автор: Крис Меллинг

Компилятор языка LASER BASIC предназначен для создания автономной программы, независимой от интерпретатора лазер-бейсика.

Он может выполнять и обработку программ, написанных на стандартном бейсике, но реального выигрыша во времени работы откомпилированных программ не представляет (в два раза или менее). Компилированный код занимает меньший объем памяти, чем исходная программа, но при запуске он должен сопровождаться пакетом рабочих процедур. Любая программа, созданная в расширенном интерпретаторе LASER-BASIC'а поместится в компиляторе.

1. Расположение файлов на ленте

```
A: COMPCODE   LOAD "" CODE
   LOADER     LOAD ""
   RT CODE    LOAD "RT CODE" CODE
B: DEMO       LOAD ""
```

2. Компиляция

При компиляции программы в памяти должны размещаться только текст исходной программы и компилятор, следовательно после ввода программы с помощью лазер-бейсика и ее запуска, программу следует записать на ленту, сбросить компьютер и загрузить в компьютер только свою программу без интерпретатора.

Программа, написанная на языке лазер-бейсик, не может редактироваться средствами синклер-бейсика.

Перед загрузкой следует выполнить:

```
CLEAR 59799
```

Загружается компилятор:

```
LOAD "" CODE или LOAD "COMPCODE" CODE
```

Исходная программа может загружаться как до, так и после компилятора.

Начало компиляции:

```
RANDOMIZE USR 59800
```

При компиляции экран отображает случайную информацию, не волнуйтесь, это не сбой — все в порядке, после окончания компиляции экран очищается, появляется сообщение "O.K.", а также номер последней откомпилированной строки.

3. Структура откомпилированной программы

Скомпилированный код скрывает исходный текст, тем не менее компьютер воспринимает его как обычную бейсик-программу. Процедуры загрузки в память и записи на ленту откомпилированного кода ничем не отличаются от соответствующих операций для обычных бейсик-программ.

Откомпилированный текст дальнейшим изменениям не подлежит, нельзя также выполнять "MERGE", если надо скомпилировать программу. Сначала надо создать один исходный модуль (MERGE), а затем его откомпилировать.

После команды "LIST" на экране появляется несколько строк текста. Первые две устанавливаются компилятором, независимо от содержания программы.

Первая строка (не выполняется) : 0 PRINT 0, она содержит недоступную пользователю информацию для пакета рабочих процедур.

Вторая строка — всегда : RANDOMIZE 59800.

Это вызов откомпилированной программы. Если в исходной программе имеются функции, определяемые пользователем, то их список (подряд) помещается в третьей строке с номером 0. Это не относится к назначениям процедур лазер-бейсика, которые компилируются отдельно.

Если в исходной программе есть операторы "DATA", то они все собираются в один оператор "DATA" и размещаются в четвертой строке, которая также нумеруется 0. Несмотря на такое объединение, оператор "RESTORE" в откомпилированной программе будет выполняться правильно.

4. Выполнение откомпилированной программы

Для выполнения откомпилированной программы необходимо снабдить ее дополнительно пакетом рабочих процедур. RAMTOP следует установить в 59799 (не забывайте об этом при загрузке откомпилированных модулей). Пакет рабочих процедур загружается командой: $PRITES.

Если программа имеет файл SPRITES, его следует загрузить точно также, как и в программе, выполненной на языке лазер-бейсик.

Загрузка откомпилированной программы выполняется как и для обычной программы на бейсике.

Инициализация выполнения программ:

RANDOMIZE USR 59800 или GOTO 1

Примечание: нажатие BREAK не прерывает выполнение программы, кроме режимов загрузки и выгрузки.

5. Компоновка автономной программы.

Программу, написанную на лазер-бейсике, необходимо соответственно настроить, чтобы для работы было достаточно набрать LOAD " ". Для этого программу необходимо скомпоновать и снабдить загрузчиком. Он входит в набор, имеющийся на исходной ленте. Вот его текст:

```
10 CLEAR 59799
20 LET SPST=
30 LOAD "RTCODE" CODE
40 LOAD "SPRITES" CODE SPST
50 LET T=INT(SPST/256): POKE 62463, SRST-256*T:
POKE 62465,T
60 POKE 62466,255: POKE 62467,220
70 POKE 56575,0
80 LOAD " "
```

Здесь :

строка 10: выделение пространства для пакета рабочих процедур

строка 20: установка начала областей размещения файла SPRITES (см. описание генератора спрайтов).

строка 30: загрузка пакета рабочих процедур.

строка 40: загрузка файла спрайтов.

строка 50: внесение в программу адреса начала области размещения спрайтов.

строка 60: внесение в программу адреса конца области размещения спрайтов.

строка 70: внесение конца области размещения знакогенератора спрайтов.

строка 80: загрузка текста скомпилированной программы.

Примечание: если спрайты в программе не используются, строки 20-70 можно исключить.

Вам остается внести требуемое значение SPST и подготовить набор файлов на чистой ленте. Помните, что делая "SAVE" для загрузчика и для откомпилированной программы, необходимо указывать строку автостарта.

6. Ограничения для стандартного бейсика

Нижеперечисленные ограничения были заложены в компилятор уже на этапе его проектирования.

1. Нельзя использовать вычисляемые выражения в операторах RUN, GOTO, GOSUB, RESTORE, так как на этапе компиляции все переменные будут установлены в 0, как и длина всех стринговых переменных.

2. В операторе DIM допустимы вычисляемые границы, но только такие, которые в процессе компиляции будут установлены в 0, и, следовательно, нельзя объявлять переменные вида DIM (N+12) — здесь значение в скобках после компиляции будет равно 12, а в ходе выполнения программы перевычисляться не будет.

3. Ограничения на число переопределений массива компилятор не накладывает. Учтите, что первое определение выполняется в ходе компиляции и оно переносится в откомпилированную программу. Следовательно, если в программе есть переход к началу программы в точку, расположенную выше первого оператора DIM, то при втором проходе этой же части программы объявление массива может оказаться недействительным. Эту сложность можно обойти, поместив все описания массивов в самом начале программы до той точки, где начинаются ветвления.

4. Использование оператора CLEAR не допускается.

5. Команда RUN как и в бейсике. Но сброса переменных нет.

6. Недопустимы операторы: MERGE, CONTINUE, LIST, LLIST.

7. Командой CONTINUE нельзя запускать сбившуюся программу или программу с оператором STOP.

8. Команды OPEN и CLOSE работают только с потоками S и P. Подкачка и сброс файлов в режиме работы с микродрайвом не поддерживаются.

9. Команды LOAD, SAVE, VERIFY являются недопустимыми. Если загружается бейсик-программа, рабочие процедуры рассматривают ее как откомпилированную программу на лазер-бейсике и попытаются ее выполнить, передав управление на первый оператор.

7. Сообщения об ошибках

1. ILLEGAL STATEMENT FOUND

— встретился один из недопустимых операторов: CLEAR, MERGE, LIST, LLIST, CONTINUE.

2. PROCEDURE DEFENITION NESTING ERROR

— найдено два описания процедуры без промежуточного оператора RETN. В описаниях процедур вложенность не допускается.

3. RETN WITHOUT DEF IN ERROR

— найден оператор RETN без соответствующего описания процедуры.

4. PROCEDURE NOT FOUND

— обращение к неописанной процедуре. Описание процедуры обязательно должно предшествовать ее использованию.

5. PROGRAMM NOT COMPILED

— это сообщение пакета рабочих процедур. При старте программы ее начало не было идентифицировано как начало откомпилируемой программы.

В ходе выполнения программы могут также выполняться стандартные сообщения об ошибках. Однако пакет рабочих процедур не в состоянии указать точку возникновения ошибки. Кроме того, следует игнорировать цифровые коды, сопровождающие сообщения об ошибках стандартного бейсика. Словом для того, чтобы быть полностью уверенным в работе откомпилированной программы, ее предварительно следует тщательно отладить и отредактировать перед компиляцией.

При возникновении ошибок в ходе выполнения, программа или ее части должна быть перекомпилирована, для чего необходимо загрузить компилятор и исходную программу (предварительно исправленную).

8. Карта памяти

Компилятор и пакет управляющих процедур загружается, начиная с адреса 59800. Пакет рабочих процедур занимает всю верхнюю часть памяти конец компилятора — по адресу 62586, то есть его длина 3057 байт. Следовательно перед загрузкой этих модулей необходимо зарезервировать для них области памяти, соответствующие их начальным и конечным адресам.

Откомпилированная программа размещается в памяти следующим образом:

программа	переменные	спрайты	стек Z80	RT CODE
PROG	VARS	ELINE	56575	59800 - 65535

9. Рекомендации по программированию в кодах

Между областью спрайтов и пакетов рабочих процедур имеется область памяти более 3K. Обычно для стека процессора требуется менее 256 байт, поэтому здесь можно размещать процедуры в машинных кодах, установив соответствующие значения в операторе CLEAR на конец файла спрайтов.

Кроме того, можно, в принципе, отводить под процедуры пользователя всю область, начиная с ELINE и до адреса 59800.

Примечание: значок $ следует читать как символ доллара.

Генератор спрайтов

Автор: Пол Ньюхем

Вступление

Генератор спрайтов предназначен для создания и редактирования спрайтов с последующих использованием их в ваших программах, написанных в LB. Спрайты создаются на экране, затем переносятся в память компьютера и потом выдаются на ленту.

Загрузка

Программа загружается по команде LOAD ''SPTGEN'' или LOAD '' ''.

Если вы хотите записать ее на ленту, сделайте BREAK, затем GOTO 9999. Обе части программы будут выгружены на ленту. Правда теперь чтобы запустить программу, ее надо перезагрузить.

Чтобы записать программу на микродрайв, надо изменить строку 2: вместо LOAD '' '' CODE поставьте: LOAD* ''M''; I; ''G'' CODE. Теперь дайте GOTO 9998.

Начало работы

После загрузки программы вы получите запрос: какой запуск вам нужно — ''холодный'' (COLD) или ''теплый'' (WARM). Поскольку сейчас мы запускаемся впервые, наберите ''C'' (COLD), а затем дайте ''G''. Появится рабочий экран.

Терминология

''Холодный пуск'' — при таком пуске программы все спрайты, находящиеся в ней, вычищаются, а все системные переменные устанавливаются в исходное состояние. В первый раз в программу всегда надо входить через ''холодный'' пуск.

''Теплый пуск'' — при таком входе в программу все спрайты сохраняются и системные переменные не изменяются. Это используется в первую очередь после случайного BREAK или какой-либо ошибки. Если вы случайно вызвали прерывание, наберите GOTO 3, затем входите через ''теплый'' пуск. Но информация на экране при этом будет утрачена. Дайте GOTO 100, что возвратит вас в исходный режим.

CHR#SQR — эта аббревиатура используется для обозначения элементарного квадрата экрана и относится к сетке 8*8, расположенной в левой части экрана. Здесь создаются и редактируются спрайты.

Экран спрайта — это область экрана размером 15*15 символов, обычно работа со спрайтами идет здесь.

Курсор CHR#SQR — это мигающий курсор, который используется для конструирования и редактирования изображения, находящегося в CHR#SQR.

Курсор экрана спрайта — это 2 мигающих курсора. Они расположены ниже и правее экрана спрайтов. Применяются для задания позиции левого верхнего угла окна, с которым идет работа. Этот угол находится на пересечении вертикали, проходящей через нижний курсор, и горизонтали, проходящей через правый курсор.

Окно экрана спрайтов — часть экрана, с которой непосредственно идет работа. Координаты окна задаются XPOS, YPOS, которые соответствуют положениям курсоров экрана спрайтов. Размеры этого окна задаются через SPRITE HEIGHT (высота спрайтов) и SPRITE LENGTH (длина спрайтов). Чтобы узнать окно, с которым вы в данный момент работаете, нажмите F и оно высветится.

Спрайты — прямоугольные элементы изображения. После того, как вы закончите создание спрайтов, вы можете записать их на ленту или микродрайв, чтобы загрузить далее в LB и использовать их в ваших программах. Для записи спрайтов на ленту вы имеете различные опции:

Опция 1. Этот режим предполагает дальнейшее редактирование спрайтов, поэтому они записываются в форме, допускающей их последующую перезарядку в генератор. Их нельзя использовать в программах на LB.

Опция 2. Этот режим выгружает спрайты, готовые к использованию в SPST — лазер-бейсике. Вам следует запомнить выданные при этом значения SPND. Спрайты, выданные в этом режиме, не могут быть загружены в генератор спрайтов.

Для тех, кто не чувствует в себе художественных способностей, мы подготовили 2 набора спрайтов на прилагаемой кассете:

SPRITE 1A — это набор из 50 спрайтов различных аркадных персонажей;

SPRITE 1B — это файл спрайтов, входящих в демонстрационную программу.

Информационный прямоугольник

Здесь отображается следующая информация, например:

MEMORY LEFT	7488	XPOS1	YPOS1
SPRITE	60218	SPRITE	HEIGHT — 2
SPST	60218	SPRITE	LENGTH — 3
SPND	65218	SPRITE	NUMBER — 15

MEMORY LEFT — указывает, какой объем памяти остался для размещения спрайтов;

XPOS, YPOS — текущие положения курсоров экрана x и Y относительно левого верхнего угла экрана спрайтов;

SPRITE — указывает адрес в памяти, где находится заданный вами спрайт;

SPST — указывает на начало области спрайтов в памяти. До задания каких-либо спрайтов имеет значение 65218;

SPND — указывает на конец области спрайтов;

SPRITE HEIGHT — обозначает высоту заданного спрайта;

SPRITE LENGTH — обозначает длину заданного спрайта;

SPRITE NUMBER — номер текущего спрайта.

В текстовой строке показывается выполняемая в данный момент команда, а также возможные опции.

Назначение клавиш генератора спрайтов

A. Задействует переключатель атрибутов (аналогично ATOF, ATON). 1 — включение, 0 — выключение.

B. Задействует переменную BRIGHT. 1 — включено, 0 — выключено.

C. Задействует переменную PAPER. Нажмите любую клавишу от 0 до 7, чтобы выбрать цвет.

SIMBOL SHIFT + C — создание больших спрайтов. Позволяет создавать спрайт (номер которого содержится в соответствующей переменной) с размером, заданным пользователем в диапазоне от 1 до 255 символов. Спрайт может быть создан ''пустым'', если никакие данные не будут введены в него.

D. Задействует прямой ввод данных. Принимает 8 байтов данных, по одному байту за 1 раз с последующим нажатием ENTER. Ввод идет с клавиатуры. Данные помещаются на экран спрайтов в позицию, отмеченную курсорами. Вводимые данные должны быть в диапазоне от 0 до 255 (десятичное) или от H00 до HFF (шестнадцатеричное). В этом случае H должен предшествовать числу.

Примечание: если переключатель атрибутов установлен на 1, то все текущие атрибуты помещаются также в эту позицию.

E. Задействует функции экрана. Имеет 3 опции: 1, 2, 3.

1. Инверсия (аналогично). По этой опции все выключенные пиксели включаются, а включенные — выключаются. Действует на все окно, размеры которого заданы SPRITE HEIGHT и SPRITE LENGTH.

2. Зеркальное отображение (аналогично MIRV). Эта опция ''переворачивает'' относительно вертикальной оси окно, заданное теми же переменными, что и опции 1.

F. Задействует режим FLASH (мигание) окна экрана, заданного переменными SPRITE HEIGHT и SPRITE LENGTH.

Этот режим является вспомогательным для проверки правильности установки курсоров экранов спрайтов, а также чтобы визуально увидеть размеры текущего окна.

SYMBOL SHIFT + F — отформатировать кэртридж микродрайва. Кэртридж подготавливается к записи спрайтов. Устанавливается 5 ''пустых'' спрайтов с номерами от 1 до 5.

G. Занесение спрайта с экрана в память. По этой команде берется спрайт, размеры которого хранятся в SPRITE HEIGHT и SPRITE LENGTH, а номер — в SPRITE NUMBER, снимается с окна, позиции которого заданы курсором спрайтов и отправляется в память компьютера.

Примечание: если при этом переключатель атрибутов включен, то они также снимаются с окна и записываются в память, в противном случае они игнорируются.

H. Ввод высоты спрайта SPRITE HEIGHT (аналогично HGT). Допускается высота от 1 до 15 символов.

I. Выдача атрибутов. По этой команде окно заданных размеров заполняется установленными атрибутами.

SYMBOL SHIFT + I — скроллинг окна вправо на один пиксель.

J — переносит CHR#SQR в экран спрайтов. Помещает рисунок из CHR#SQR в элементарный квадрат, отмеченный курсорами. Атрибуты переносятся, если переключатель установлен в 1.

SYMBOL SHIFT + J — загрузка спрайтов с ленты и микродрайва.

Примечание: спрайты, находящиеся в памяти, по этой команде могут быть разрушены.

K. Перенос элементарного квадрата, помеченного курсорами, из экрана спрайтов в CHR#SQR. Атрибуты переносятся, если переключатель установлен в 1.

L. Ввод длины спрайта SPRITE LENGTH (аналогична LEN).

M. Команда аналогична "E". Здесь также предоставляется 3 опции. Разница состоит в том, что здесь эти функции над спрайтами выполняются только в памяти компьютера, а на экране изменений не происходит.

N. Этой командой дается ответ "нет".

O. Задействует логические функции. По этой команде вам предоставляется 3 опции. Каждая опция снимает область с экрана спрайтов, левый верхний угол которого установлен двумя курсорами, а размеры соответствуют определенному спрайту и помещают эту область в спрайт, номер которого определен переменной SPRITE NUMBER. При отключенных атрибутах они игнорируются.

Опция 1. Накладывает данные с экрана на предварительно определенный спрайт по принципу "или" ("OR") и результат оставляет в спрайте (экран не меняется).

Опция 2. Накладывает данные с экрана по принципу "исключающее или" ("XOR").

Опция 3. Накладывает данные с экрана по принципу "и" ("AND").

P. Помещает спрайт, номер которого содержится в SPRITE NUMBER на экран спрайтов в позицию, отмеченную курсорами.

Q. Выполняет очистку блока CHR#SQR.

SYMBOL SHIFT + Q. Очистка экрана спрайтов.

R. Переворот спрайта в памяти на 90 градусов. Исходный спрайт не изменяется. Новому спрайту после поворота присваивается новый номер. Атрибуты поворачиваются автоматически.

S. Управляет переменной SPRITE NUMBER (аналогично SPN). Позволяет задавать спрайты. Если запрашивает номер уже существующего, то будет выдано предупреждение.

SYMBOL SHIFT + S. Выполняет команду "SAVE" для спрайтов. После этого выполняется "VERIFY". Если в результате VERIFY произойдет сбой, то дайте GOTO 100, тогда ваши данные в памяти не будут утрачены.

T. Выполняет тестирование спрайтов (аналогична TST). Проверяет спрайт, номер которого содержится в переменной SPRITE NUMBER. Выполняет:

1. Помещает высоту спрайта в переменную SPRITE HEIGHT;

2. Помещает длину спрайта в переменную SPRITE LENGTH;

3. Помещает адрес памяти, в котором начинается данный спрайт, в переменную SPRITE;

4. Помещает адрес начала области спрайтов в переменную SPST;

5. Помещает адрес конца области спрайтов в переменную SPND;

6. Рассчитывает объем свободной памяти, доступной для хранения спрайтов и помещает его в переменную MEMORY LEFT.

Примечание: изображение этих двух переменных на экране в случае необходимости будет переработано.

SYMBOL SHIFT + T. Скроллинг окна влево на один пиксель.

U. Берет параметры атрибутов символа из экрана спрайтов с позиции, отмеченной курсорами и помещает их в 4 переменных атрибута.

SYMBOL SHIFT + U. Скроллинг окна вверх.

V. Включение переменной FLASH (это одна из 4-х переменных атрибутов).

1 — включение;

2 — выключение.

W. Включение функции стирания спрайтов "WIPE SPRITE" (аналог DSPR). Полностью стирает из памяти спрайт, на который указывает переменная SPRITE NUMBER. Все спрайты, находящиеся в памяти ниже этого спрайта поднимаются вверх, выбирая свободное пространство.

X. Задает переменную INK. После нее надо нажать клавишу от 0 до 7, выбирая нужный цвет.

Y. Дает положительный ответ "да" при запросе "Y/N".

SYMBOL SHIFT + Y. Скроллинг окна на 1 пиксель.

SPACE — дает возможность переместить спрайт небольших размеров внутрь второго, большего, спрайта в позицию, заданную переменными ROW и COL второго спрайта.

Здесь существует 4 опции:

1. GETBIS: меньший спрайт помещается непосредственно в больший.

2. GETORS: накладывает по "OR";

3. GETXRS: накладывает по "XOR";

4. GETND: накладывает по "AND".

5, 6, 7, 8 — перемещение курсора в CHR#SQR в соответствующих направлениях.

9 — включает CHR#SQR в выбранной позиции.

0 — выключает CHR#SQR в выбранной позиции.

Глава 4. Программирование на ассемблере

4.1. BEYOND BASIC (введение в ассемблер)

Обучающая программа 'BEYOND BASIC' предназначена для того, чтобы глубже познакомить вас с компьютером ZX-SPECTRUM. Она объясняет, что происходит внутри компьютера, когда вы прогоняете программу, а также, чтобы обучить вас некоторым аспектам программирования в машинных кодах.

Основной отличительной особенностью программы 'BEYOND BASIC' является ее наглядность. Здесь вы можете написать свои собственные программы на языке ассемблера Z80 и проследить на экране, каким образом они воздействуют на память и регистры во время 'прогона'.

Сама программа не является истинным ассемблером Z80, также в ней не ставится цель обучения всем командам ассемблера Z80. Ее задача — дать основные навыки программирования на ассемблере, что позволило бы вам двигаться дальше — возможно, с помощью учебников, которые вам кажутся сейчас слишком заумными.

Загрузка программы

Когда все готово, для загрузки программы нажмите клавишу 'LOAD', запустите магнитофон и нажмите клавишу 'ENTER'. Загрузка программы происходит в два этапа. Это занимает около 4-х минут. Когда на экране покажется 'приглашение', выключите магнитофон. Программа самозапускается.

Режимы работы программы

После того, как вы загрузили программу и прочли четыре кадра введения, вам прежде всего нужно выбрать один из следующих режимов:

1. Просмотр памяти и регистров.
2. Обучение командам ассемблера.
3. Создание и прогон программы пользователя.
4. Возврат к бейсику.

Выберите нужный режим, просто нажав клавишу цифры, высвечиваемой на экране телевизора и затем — 'ENTER'. Всегда можно, переходить из режима в режим в любой последовательности, но первоначально их нужно пройти все последовательно.

Просмотр памяти и регистров

В этом режиме объясняются основные положения архитектуры и работы компьютера. Сюда входят и описания:

— RAM (памяти с произвольным доступом);
— ПЗУ;
— регистров;
— простых машинных команд;
— регистра программного счетчика.

Описываемые машинные команды не являются исходными операциями Z80. Вначале они вводятся как простые команды, типа команд языка бейсик, которые постепенно становятся все более похожими на истинные коды Z80, по мере того, как объясняются новые концепции.

Обучение командам ассемблера

В этом режиме объясняется широкий набор истинных команд ассемблера Z80. Каждая команда сопровождается текстовыми пояснениями и 'живым' примером. Пример иллюстрирует как выполнение данной команды влияет на память и регистры.

Вы можете пройти через все команды по порядку или выбирать их по желанию. Это будет особенно полезно впоследствии, когда вы будете достаточно подготовлены для написания программ на ассемблере и вам понадобиться вспомнить, какие функции выполняет та или иная команда.

Создание и прогон программ пользователя

Этот режим позволит вам закрепить все изученное ранее, написав свои собственные программы на языке ассемблера Z80. Затем вы сможете проследить, что происходит при выполнении программы по изменению содержимого памяти и регистров, индицируемого на экране телевизора. Этот режим представляет возможность редактирования кодов ассемблера или содержимого памяти. Можно легко перемещаться вверх или вниз по программе для повторного прогона какого-либо ее участка или можно вернуться в предыдущий режим для проверки форматов команд.

По желанию вы можете сохранить ('SAVE') и перезагрузить ('RELOAD') свои программы, написанные на ассемблере.

Возврат к бейсику

Этот режим позволяет вам вернуться к программированию на языке бейсик.

Наши соображения

Мы намеренно упростили концепции ассемблера, описываемые в программе 'BEYOND BASIC'. Основной целью программы является — дать новичку достаточно информации, чтобы облегчить обучение. Чрезмерный объем информации мог бы вызвать замешательство. По этой причине мы не давали подробного описания того как хранятся адреса в машинных кодах Z80. Мы 'предназначили' для вас ряд областей памяти и условились, что вам необходимо обращаться к адресу ячейки, а не к ее содержимому. Это в полной мере описывается в самой программе. На экране выводится упрощенный вариант.

Мы надеемся, что это введение в ассемблер побудит вас к дальнейшему чтению литературы по ассемблеру Z80.

4.2. MONS–4

Раздел 1. Запуск

MONS4 поставляется в перемещаемой форме. Вы просто загружаете его с желаемого адреса, и затем вызываете с этого же адреса. Если вы хотите запустить MONS4 вновь (вернувшись из MONS4 в BASIC), то вы должны ввести адрес, с которого первоначально загрузили отладчик.

Пользователи 'PLUS 3' должны прочитать дополнительный листок с особенностями внешнего отладчика, который расположен на адресуемом диске и требует только 100 байтов в обычном 48-Кбайтовом ОЗУ. Пример:

Скажем, вы хотите загрузить MONS4 по адресу #C000 (49152 десятичный). Для этого выполните следующие операции:

```
LOAD '' '' CODE 49152 <ENTER>
RANDOMIZE USR 49152 <ENTER>
```

Для повторного входа используйте:

```
RANDOMIZE USR 49152 <ENTER>
```

Будучи однажды помещенным в память, MONS4 занимает приблизительно 6K в длину, но вы должны предоставить непрерывную область 7K на загрузку MONS4, т.к. после основных кодов надо поместить таблицу перемещаемых адресов. MONS4 содержит собственный внешний стек, так что он является самообслуживающейся программой.

MONS4 по умолчанию загружается с адреса 55000, хотя вы можете загрузить его с любого разумного адреса. Обычно удобно загружать MONS4 в верхние адреса памяти.

Получение копии

Однажды загрузив MONS4 в память вашего SPECTRUM'а, вы можете сделать копию, проделав следующие операции:

```
SAVE 'MONS4'CODE XXXXX, 6656 <ENTER> — на кассету
SAVE *'M',1,'MONS4'CODE XXXXX,6656 <ENTER> — на дискету
SAVE *'M',1,'MONS4'CODE XXXXX,6780 <ENTER> — на диск OPUS
```

где XXXXX — адрес, с которого вы загрузили MONS4.

Когда вы войдете в MONS4, вам представится изображение так называемой фронтальной панели (в дальнейшем — Ф.П.). (см. примечание с примером). Оно состоит из регистра Z Z80, флага OB вместе с их содержимым, плюс 24-х байтовая секция памяти, сосредоточенная вокруг указателя памяти, первоначально указывающего на нулевой адрес. В верхней части экрана находится дизассемблированная инструкция, на которую указывает указатель памяти. (в дальнейшем — MP).

На входе MONS все адреса, отображаемые внутри Ф.П. представлены в шестнадцатиричном формате. Вы можете изменить это с помощью команды:

```
<SYMBOL SHIFT> 3
```

и адреса будут представлены в десятичном формате (см. следующий раздел). Заметим, однако, что адреса всегда должны вводиться в шестнадцатиричном формате. Команды вводятся с

клавиатуры в соответствии с подсказкой '>' как с верхнего, так и с нижнего регистров.

Некоторые команды, эффект от которых может быть гибельным в случае их ошибочного использования, требуют нажатия клавиши <SYMBOL SHIFT>. Это ручное нажатие клавиши <SIMBOL SHIFT> будем представлять знаком ' ', т.е. запись ' Z' означает нажатие <SIMBOL SHIFT> и 'Z' вместе.

Команды выполняются немедленно и не требуют после себя ввода <ENTER>. Неверные команды просто игнорируются.

Многие команды требуют ввода 16-ричного числа. При вводе такого числа вы можете ввести любое количество 16-ричных цифр (0...9 и A...F) и закончить ввод любым отличным от 16-ричного числа символом. Если ограничителем является значащая команда, то эта команда выполняется после того, как будет выполнена предыдущая. Если ограничителем является знак '-', то введенное отрицательное число будет возвращено в двойном дополнительном коде: '1800-' дает 'E800'. Если при вводе 16-ричного числа вы ввели более 4-х цифр, то только 4 последние введенные цифры сохраняются и отображаются на экране.

Для возврата из MONS4 в BASIC просто нажмите <STOP> (т.е. 'A').

Пользователи 'SPECTRUM PLUS 3' должны ознакомиться с дополнительными инструкциями, поставляемыми с этим руководством, прежде чем идти дальше.

Раздел 2. Команды MONS-4

В MONS4 используются нижеприведенные команды. В этом разделе <ENTER> употребляется в качестве ограничителя 16-ричного числа. Фактически это может быть любой не 16-ричный символ (см. Раздел 1). Для обозначения пробела используется символ '-'.

1. <SYMBOL SHIFT> 3 или #

Переключает систему счисления, в которой отображаются адреса, с 16-ричной на десятичную и обратно. На входе MONS4 все адреса показываются в 16-ричном виде, для десятичного отображения используйте ' 3' и вновь ' 3' для возврата в 16-ричную форму. Это влияет на все адреса, отображаемые MONS4, включая адреса, получаемые дизассемблером, но это не меняет отображения содержимого памяти, которое всегда выдается в 16-ричном виде.

2. <SYMBOL SHIFT> 4 или $

Отображает дизассемблированную страницу, начиная с адреса, указанного MP. Полезно посмотреть и следующие инструкции. Нажмите '4' вновь или <EDIT> для возврата к отображению Ф.П. или другую клавишу для получения следующей дизассемблированной страницы.

3. <ENTER>

Наращивает счетчик адресов на 1, так что 24-х байтовая область памяти сдвигается на 1 байт вперед.

4. <стрелка вверх>

Уменьшает MP (указатель памяти) на 1.

5. <стрелка влево>

Уменьшает значение MP на 8. Используется для быстрого просмотра.

6. <стрелка вправо>

Увеличивает значение MP на 8. Используется для быстрого просмотра.

7. <,> (запятая)

Заменяет MP адресом, находящимся в стеке (SP). Это полезно, когда вы хотите посмотреть адрес возврата вызванной подпрограммы.

8. <G>

Поиск по образцу введенной строки. Сначала высвечивается подсказка на ввод первого байта, который необходимо найти. За байтом следует <ENTER>. Затем вводите следующий байт (и <ENTER>) в ответ на ':' до тех пор пока вы не определите всю строку образа. После этого введите только <ENTER>, это ограничит строку образа. Будет произведен поиск образа, начиная с текущего адреса до первого появления введенной строки. Когда эта строка будет найдена, обновится Ф.П. Так, что Ф.П. будет указывать на первый символ строки.

Например:

Скажем, вы хотите исследовать память, начиная с адреса #80000 на появление образа #3E #FF (2 байта). Последовательность действий должна быть следующей:

M:80000 <ENTER> — устанавливает MP

G:3E <ENTER> — определяет 1-й байт строки

FF <ENTER> — определяет 2-й байт строки

<ENTER> — ограничивает строку

После последнего <ENTER> (или любого не 16-ричного символа) G начинает поиск с адреса #80000 первого появления #3 #FF. Когда будет найдена строка, то обновится изображение на Ф.П. Для поиска дальнейших появлений строки используйте команду N.

9. <H>

Вам выводится подсказка ':' на ввод десятичного числа, ограниченного не цифрой (любым символом, кроме 0... 9). Как только вы введете число и ограничитель, в той же строке высветится знак '=' и 16-ричный эквивалент введенного десятичного числа. Теперь нажмите любую клавишу для возврата в командный режим.

Например:

H:441472-=A200 — пробел использован как ограничитель.

10. <I>

Используется для копирования блока памяти из одного места в другое, причем блок памяти может быть скопирован в ячейки с перекрытием своего первоначального местоположения. <I> выдает подсказки на ввод стартового и конечного адресов блока, который должен быть скопирован (<FIRST:>, <LAST:>), затем для адресов, по которым он должен быть помещен (<TO:>). Введите в ответ на каждую подсказку 16-ричные числа. Если стартовый адрес больше конечного, то команда прерывается, в противном случае блок перемещается как указано.

11. <J>

Выполняет задачу с определенного адреса. Эта команда выдает подсказку ':' для 16-ричного числа, после введения которого сбрасывается внешний стек, экран очищается и выполнение перемещается в область определенного адреса. Если вы желаете вернуться к Ф.П. после выполнения, то установите точку останова (см. команду W) в месте, где вы желаете вернуться к отображению. Например:

J:B0000 <ENTER> — выполняет программу с адреса #B0000

Вы можете прервать программу перед концом ввода адреса, используя <EDIT>. Отметим, что выполнение команды <J> портит регистры Z80, поэтому программа в машинных кодах не должна делать присвоений значений, содержащихся в регистрах. Если вы желаете выполнять программу с регистрами, то нужно использовать 'K'

12. <SYMBOL SHIFT> K

Продолжает выполнение с адреса, находящегося в счетчике инструкций. Эта команда, вероятнее всего, будет использоваться совместно с командой W — пример пояснит это.

Скажем, вы в пошаговом режиме (используя ' Z') проходите через коды, приведенные ниже, и вы дошли до адреса #90000, но вы хотите увидеть, как выставляются флаги после вызова подпрограммы в #8800.

```
891E  3EFF          LD    A,-1
8920  CD0090          CALL  39000
8923  2A0080        LD    HL,(#8000)
8926  7E            LD    A,(HL)
8927  111488        LD    DE,#8814
892A  CD0088        CALL  #8800
892D  2003          JR    NZ,LABEL
892F  320280        LD    (#8002),A
8932  211488  LABEL LD    HL,#8814
```

Действуйте в следующем порядке:

Установите точку останова, используя W, по адресу #892D (напомним, что сначала нужно использовать ' M' для установки MP) и затем вводите команду ' K'. Выполнение будет продолжаться до тех пор, пока не дойдем до адреса, указанного в точке останова, затем будет обновлен экран и вы сможете проверить состояние флагов и т.п. после вызова процедуры с адреса 8800. Затем можно возобновить пошаговый режим. ' K' полезна для выполнения кода без первоначального сброса стека или без разрушения содержимого регистров, что делает <J>.

13. <L>

Распечатывает блок памяти, начиная с адреса, находящегося в данный момент в MP.

<L> очищает экран и отображает 16-ричное представление и ASCII-эквиваленты 80 байтов памяти, начиная с текущего адреса, на который указывает MP. Адреса будут указаны в 16-ричном или десятичном виде, в зависимости от текущего состояния Ф.П. (см. ' 3'). Отображение состоит из 20 рядов по 4 байта в каждом, ASCII-эквиваленты показаны в конце каждого ряда. Применительно к ASCII от отображения любого значения выше 127 отнимается код 128 и любые значения между 0... 31 включительно показаны как '.'.

В конце страницы листинга вы можете вернуться к основному отображению Ф.П. нажатием <EDIT> или продолжать со следующей страницы из 80 байтов нажатием любой другой клавиши.

14. <M>

Заносит в MP определенный адрес. Выдается подсказка ':' на ввод 16-ричного адреса (см. раздел 1). MP обновляется введенным адресом и, соответственно, на экране меняется отображение области памяти.

15. <N>

Находит следующее появление строки, определенной в прошлый раз командой <G>. <G> позволяет вам определить строку и затем осуществить поиск ее первого появления. Если вы хотите

найти следующее появление строки, используйте <N>. <N> начинает поиск с адреса, указанного в MP и обновляет отображение памяти, когда искомая строка будет вновь найдена.

16. <O>

Команда берет байт, адресуемый в данный момент MP, и представляет его как относительное смещение, соответственно обновляя отображение. Например:

Допустим, MP установлен в #6800 и содержимое ячеек #67FF и #6800 равны #20 и #16 соответственно (это может быть представлено как JR NS, $+24). Чтобы выяснить, куда будет переход по ненулевому условию, просто нажмите <O>, тогда MP адресуется на назначенный байт #16, экран будет обновлен и отобразится область памяти с #6817, требуемого значения перехода.

Помните, что относительные перемещения выше #7FF (127) трактуются процессором как отрицательные, <O> принимает это в расчет.

См. также команду <U> в связи с <O>.

17. <P>

Заполняет память между определенными границами нужным байтом. <P> выдает подсказки <FIRST:>, <LAST:>, <WITH:>. Введите 16-ричные числа в соответствии с этими подсказками (соответственно: начальный и конечный адреса (включительно) блока, который вы желаете заполнить, и байт заполнитель). Например:

```
<P>
FIRST: 7000   <ENTER>
LAST:  77FF   <ENTER>
WITH:  55     <ENTER>
```

Ячейки с #7000 по #77FF будут заполнены байтом #55. Если стартовый адрес больше конечного, то <P> будет оборвана.

18. <Q>

Переключение наборов регистров. На входе Ф.П. отображается стандартный набор регистров (AF, HL, DE, BC). При использовании <Q> будет отображаться альтернативный набор регистров (AF', HL', DE', BC'), который отличается от стандартного штрихом <'> после имени регистра. Если <Q> используется, когда отображается альтернативный набор регистров, то на экран будет выдан стандартный набор.

19. <SYMBOL SHIFT> T

Устанавливает точку останова после текущей инструкции и продолжает выполнение. Например:

```
9000   B7       OR     A
9001   C20098   CALL   NZ,#9800
9004   010000   LD     C,0
9800   21FFFF   LD     HL,-1
```

Вы в пошаговом режиме продвигаетесь по кодам и достигли адреса #9001 с ненулевым значением в регистре A, так что флаг 'ZERO' будет в состоянии NZ после инструкции OR A.

Если вы сейчас используете ' Z' для продолжения пошагового режима, то выполнение продолжится с адреса #9800 подпрограммы. Если вы не желаете шагать по подпрограмме, то используйте команду ' T', тогда инструкции с #9800 и вызов будут пройдены автоматически и выполнение остановится на адресе #9004 для дальнейшего продолжения в пошаговом режиме. Помните, что ' T' устанавливает точку останова после текущей инструкции и затем использует команду ' K' (см. команду ' Z' с расширенным примером пошагового прохода).

20. <T>

Дизассемблирование. Дизассемблирует блок кодов на принтер и/или дискету (по ключу). Сначала выдается подсказка на ввод начального (<FIRST:>) и конечного (<LAST:>) адресов кодов, которые вы желаете дизассемблировать. Введите их в 16-ричном виде (как описано в разделе 1).

Если начальный адрес больше конечного, то команда аннулируется. После ввода адресов выдается подсказка (<PRINTER?>). Для направления результатов дизассемблирования на принтер отвечайте 'Y'. При наборе любого другого символа результат будет выведен на терминал. Затем выдается подсказка <TEXT:> на ввод 16-ричного начального адреса текстового файла, который будет являться результатом дизассемблирования. Если вы не хотите, чтобы был сгенерирован текстовый файл, то просто нажмите <ENTER> после этой подсказки. Если вы определили адрес, то дизассемблированный файл будет получен, начиная с этого адреса, в форме, пригодной для использования GENS4. Если вы хотите загрузить этот файл с помощью GENS4, то вы должны заметить его конечный и начальный адреса, вернуться в BASIC и сохранить текст как файл типа 'CODE'. Затем вы можете загрузить код в редактор GENS4 прямо с помощью команды <G>.

Вместо ввода адреса в ответ на <TEXT:> вы можете ввести имя файла на дискете и на ней текст будет сохранен, как только пройдет дизассемблирование. Например:

```
TEXT: 2: DCODE <ENTER>
```

Результат дизассемблирования будет сохранен на 2-м дисководе под именем DCODE. Если вы дизассемблируете на дисковод, то листинг на экран выдаваться не будет. Файл, полученный на дисководе, является самозагружающимся с помощью команды <G>. Если на некоторой стадии генерации текстового файла текст будет перекрывать MONS4, то дизассемблирование будет прекращено — нажмите любую клавишу для возврата к Ф.П.

Если вы определили адрес текстового файла, то вас попросят определить 'WORKSPACE:' — начальный адрес свободной области памяти, которая используется как примитивная таблица символов для любых меток, которые производит ассемблер. Требуется по два байта на каждую метку. По умолчанию (когда вы просто нажимаете <ENTER>) резервируется 4K пространства ниже MONS4.

После этого повторно появляются запросы на начальный и конечный адреса любых областей данных, существующих внутри блока, который вы желаете дизассемблировать. Области данных — это области, скажем, текста, который вы не желаете представлять как инструкции Z80 — вместо этих областей дизассемблером должны быть сгенерированы директивы DEFB.

Если значение байта данных лежит между 32 и 127 (#20 #7F) включительно, то дается интерпретация байта в коде ASCII, т.е. #41 заменяется на A после DEFB. Когда вы закончили определение областей данных, или вы не желаете их определять, то просто нажмите <ENTER> в ответ на обе подсказки.

Команда <T> определяет область в конце MONS4 для хранения адресов областей данных и, таким образом, вы можете определять столько областей данных, сколько позволяет память. Каждая область данных требует 4 байта пространства. Отметим, что использование <T> уничтожает любые точки останова, которые первоначально были установлены (см. команду W).

Байт после выполнения инструкции RST 8 дизассемблируется как DEF N, т.к. этот байт собирается ПЗУ SPECTRUM'а и никогда не выполняется.

Экран будет очищен. Если будет запрос на создание текстового файла, то произойдет короткая задержка (в зависимости от размеров секции памяти, которую вы хотите дизассемблировать) пока создается таблица символов во время первого просмотра. Когда это будет сделано, на экране (или принтере) появится листинг дизассемблирования, если только вы не дизассемблируете на дискету — в этом случае листинг не производится. Вы можете остановить листинг в конце строки нажатием <ENTER> или <SPACE>, последующее нажатие <EDIT> возвращает к отображению Ф.П., любой другой ключ продолжает дизассемблирование.

Если встречается неверный код, то он дизассемблируется как <NOP> и отмечается звездочкой после кода в листинге. В конце дизассемблирования изображение будет приостановлено и, если был запрос на создание текстового файла, то возникнет сообщение:

```
END OF THE TEXT XXXXX
```

Метки генерируются там, где это уместно (т.е. в C30078) в форме XXXXX, где XXXXX — абсолютный 16-ричный адрес метки, но только если их адреса попадают внутрь границ дизассемблирования. В противном случае метки не генерируются, а просто даются 16-ричные или десятичные адреса.

Например, если вы проассемблировали область между #7000 и #8000, то затем инструкция C30078 может быть дизассемблирована как JP L7800; с другой стороны, если вы дизассемблируете с #9000 по #9800, то эта же инструкция может быть дизассемблирована как JP #7800 или JP #30720, если используется десятичное отображение.

Если особый адрес отнесен к инструкции внутри дизассемблирования, то его метка появится в поле метки (перед мнемоникой) дизассемблированной конструкции этого адреса, но только если листинг направляется в файл. Например:

```
T
FIRST:      8B  <ENTER>
LAST:       9E  <ENTER>
PRINTER:    Y   <ENTER>
TEXT:           <ENTER>
FIRST:      95  <ENTER>
LAST:       9E  <ENTER>
FIRST:          <ENTER>
LAST:           <ENTER>
```

может быть произведено нечто подобное:

```
008B   FE16     CP     #16
008D   3801     JR     C,L0090
008F   23       INC    HL
0090   37       SCF
0091   225D5C   LD     (5C5D),HL
0094   C9       RET
0095   BF524E   DEFB   #BF,'R','N'
0098   C4494E   DEFB   #C4,'I','N'
```

```
009B   4B4559     DEFB  'K','E','Y'
009E   A4         DEFB  #A4
```

21. <U>

Используется совместно с командой <O>. Помните, что она обновляет отображение памяти в соответствии с относительным смещением, т.е. показывает эффект инструкций JR или DJNZ. <U> используется для возврата к отображению, которое было при последнем использовании <O>. Например:

```
7200   A7     71F3     77
7201   20     71F4     C9
>7202  F2<    >71F5    F5<
7203   06     71F6     C5
картинка 1    картинка 2
```

На вашем терминале — картинка 1 и вы хотите узнать, когда будет относительный переход 20 F2. Нажмите клавишу <O> и появится картинка 2 — обновленное отображение памяти.

Итак, вы нажимаете <O>, и появляется картинка 2. Теперь вы исследуете код по адресу #71F5 и затем хотите вернуться к кодам, соответствующим первоначальному относительному смещению для того, чтобы увидеть, что происходит, если установлен флаг нуля. Нажмите <U>, и отображение памяти вернется к картинке 1. Отметим, что вы можете использовать <U> только для возврата к последнему проявлению команды <O>, все предыдущие проявления <O> утеряны.

22. <V>

Используется совместно с командой <X> и подобна команде <U> по эффекту, с той разницей, что она обновляет отображение памяти на картинку, которая была перед последней командой <X>. Например:

```
8702   AF     842D     18
8703   CD     842E     A2
>8704  2F<    >842F    E5<
8705   44     8430     21
картинка 1 картинка 2
```

На вашем экране — картинка 1 и вы хотите посмотреть подпрограмму с адреса #842F. Для этого вы нажимаете <X>. При данной картинке отображение памяти обновляется на картинку 2. Вы просматриваете эту подпрограмму, и затем хотите вернуться к кодам первоначального вызова подпрограммы. Для этого нажмите <V> и вновь появится картинка 1.

Как и в случае с <U>, используя эту команду, вы можете достичь только адресов, при которых была введена последняя команда <X>. Все предыдущие адреса, в которых использовалась команда <X>, будут утеряны.

23 <W>

Установка точки останова. Точка останова для MONS4 — это просто инструкция вызова внутри MONS4 подпрограммы, которая отображает Ф.П. Так что программисту дается возможность остановить выполнение программы и проверить регистры Z80, флаги и любые значащие ячейки памяти.

Таким образом, если вы хотите остановить выполнение программы в #9876, то используйте команду <W> для установки MP на #9876 и затем используйте <W> для установки точки останова по этому адресу. 3 байта кода, которые были первоначально по адресу #9876, заменяются инструкцией CALL, которая останавливает выполнение, когда на нее выходят. Когда достигается эта инструкция CALL, то она восстанавливает три первоначальные байта по адресу #9876 и высвечивает Ф.П. со всеми регистрами и флагами в состоянии, в котором они находились перед выполнением точки останова. Теперь вы можете использовать любые возможности MONS4 обычным способом.

Замечания по использованию точек останова:

Когда встречается точка останова, MONS4 будет выдавать звуковой сигнал и ожидать вашего нажатия клавиши перед возвратом к Ф.П. MONS4 использует область в конце самого себя, которая первоначально содержит перемещаемые адреса для сохранения информации о точках останова. Это означает, что вы можете установить столько точек останова, сколько позволит память. Каждая точка останова требует 5 байтов памяти. Когда останов произошел, MONS4 автоматически восстанавливает значения памяти, которые существовали до этой точки останова.

Отметим, что команда <T> использует эту область, поэтому при работе команды <T> все точки останова потеряются. Точки останова могут быть установлены только в ОЗУ. Т.к. точка останова состоит из трехбайтовой инструкции, то в определенных случаях необходимо проявить предусмотрительность, например:

Предполагаемый код:

```
8000   3E     8008     00
8001   01     8009     00
8002   18     800A     06
8003   06     800B     02
>8004  AF<    >800C    18
8005   0E     800D     F7
8006   FF     800E     06
8007   01     800F     44
```

Если вы установили точку останова по #8004 и затем начали выполнение с ячейки #8000, то регистр A будет загружен значением 1, выполнение переместится на #800A, регистр B загрузится значением 2 и выполнение переместится на #8005. Но в #8005 помещается младший байт вызова точки останова и, таким образом, мы испортим код и появятся непредсказуемые результаты. Эта ситуация нетипична, но вы должны оградить себя от нее — в этом случае единственный выход — использовать пошаговый проход (см. команду ' Z' ниже с подробным описанием пошагового прохода).

24. <X>

Используется для обновления MP по назначению абсолютного вызова подпрограммы или по инструкции JP. <X> берет 16-разрядный адрес, определяемый байтом, на который указывает MP, и байтом MP+1, и затем обновляет отображение памяти на экране таким образом, что отображаются адреса, близлежащие к данному адресу. Помните, что младшая половина адреса определяется первым байтом, а старшая вторым (формат INTEL). Например:

Вы хотите посмотреть подпрограмму, которая вызывается кодом CD0563. Установите MP (используя <M>) так, что он адресует к коду 05 внутри инструкции вызова и нажмите <X>. Отображение памяти будет обновлено и отобразятся ячейки с адресами, близкими к #6305 (см. также команду <V>).

25. <Y>

Ввод ASCII. <Y> дает вам новую строку, которую вы можете ввести символами ASCII прямо с клавиатуры. Эти символы попадают на эхо-печать и их 16-ричные эквиваленты вводятся в память, начиная с текущего значения MP. Строка символов должна заканчиваться <EDIT>. <DELETE> (<CAPS SHIFT> 0) может быть использована для удаления символов из строки.

Когда вы закончите ввод символов ASCII (и введете <EDIT>), то экран обновится и MP спозиционируется на конец строки, введенной в память.

26. <SYMBOL SHIFT> Z

Перед использованием ' Z' (или ' T') PC должен быть установлен на адрес инструкции, которую вы хотите выполнить.

'Z' просто выполняет текущую инструкцию и обновляет Ф.П., чтобы отобразить изменения, вызванные выполнением данной инструкции.

Отметим, что пошаговый режим возможен в любом месте карты памяти (и в ОЗУ, и в ПЗУ), но невозможен во внешнем ПЗУ. Далее следует расширенный пример, который пояснит действие многих команд отладчика MONS4. Вы должны их внимательно изучить и постараться выполнить самостоятельно.

Рабочий пример:

Давайте предположим, что у нас в машине есть 3 секции кода, приведенного ниже. Первая секция — программа, которая загружает HL и DE числами и затем вызывает подпрограмму для их перемножения (вторая секция) с результатом в HL и затем дважды вызывает подпрограмму для вывода результатов умножения на экран (3-я секция). Текст секций:

```
7080   2A0072          LD    HL,(#7200)
7083   ED5B0272        LD    DE,(#7202)
7087   CD0071          CALL  MULT
708A   7C              LD    A,H
708B   CD1D71          CALL  AOUT
708E   7D              LD    A,L
708F   CD1D71          CALL  AOUT
7092   210000          LD    HL,0
7100   AF      MULT    XOR   A
7101   ED52            SWC   HL,DE
7103   19              ADD   HL,DE
7104   3001            JR    NC,MU1
7106   EB              EX    DE,HL
7107   B2      MU1     OR    D
7108   37              SCF
7109   C0              RET   NZ
710A   B3              OR    E
710B   5A              LD    E,ED
710C   2007            JR    NZ,MU4
710E   EB              EX    DE,HL
710F   C9              RET
7110   EB      MU2     EX    DE,HL
7111   19              ADD   HL,DE
7112   EB              EX    DE,HL
7113   29      MU3     ADD   HL,HL
7114   DB              RET   C
7115   1F      MU4     RRA
```

```
116    30FB          JR     NC,MU3
7118   B7            OR     A
7119   20F5          JR     NZ,MU2
711B   19            ADD    HL,DE
711C   C9            RET
711D   F5      AOUT  PUSH   AF
711E   0F            RRCA
711F   0F            RRCA
7120   0F            RRCA
7121   0F            RRCA
7122   CD671         CALL   NIBBLE
7125   F1            POP    AF
7126   E60F    NIBBLE AND    %1111
7128   C690          ADD    A,#90
712A   27            DAA
712B   CE40          ADC    A,#40
712D   27            DAA
712E   FD213A5C      LD     IY,#5C3A
7132   D7            RST    #10
7133   C9            RET
7200   1B2A          DEFW   10779
7202   033A          DEFW   3
```

Теперь мы хотим исследовать вышеприведенный код. Сделаем это со следующим набором команд. Отметим, что пошаговый режим — это просто один из способов, он недостаточно эффективен, но весьма иллюстративен:

```
M: 7080 <ENTER> установить MP на #7080 7080 установить PC на
#7080
Z простой шаг
Z простой шаг
Z простой шаг
следует вызов CALL
M:7115 <ENTER> пропуск предподготовки чисел
W установка точки останова
K продолжить с #7100 до останова
Z простой шаг
Z простой шаг
Z простой шаг
Z простой шаг
Z простой шаг
Z простой шаг
Z возврат из подпрограммы умножения
Z простой шаг
Z следует вызов CALL
M:7128 <ENTER> установка MP на нужный байт
W установка точки останова
K продолжить с #711D до останова
Z простой шаг
Z простой шаг
Z простой шаг
Z простой шаг
, просмотр адреса возврата
W установка там точки останова
K продолжение
Z
,
W
K
Z
T
```

Пожалуйста, проработайте этот пример. Сначала введите код подпрограммы (см. Модификация памяти) — можно использовать GENS4, и затем выполните команды, приведенные выше. Вы высоко оцените пример в качестве иллюстрации того, как пройти по программе.

27. <SYMBOL SHIFT> P

Эта команда аналогична LIST, но вывод вместо экрана идет на принтер. Помните, что в конце страницы вы нажимаете <EDIT> для возврата к Ф.П. или любую другую клавишу для получения другой страницы.

28. Модификация памяти

Значения ячеек по адресам, которые дает MP, могут быть модифицированы вводом 16-ричного числа, за которым следует ограничитель (см. раздел 1). Две последние 16-ричные цифры (если одна, то она дополняется нулем) вводятся в ячейку, на которую указывает MP, и затем команда (если она есть), определенная ограничителем, выполняется. Если ограничивается незначащая команда, то она игнорируется. Например:

F2 <ENTER> #F2 вводится и MP:=MP+1

123 <CAPS SHIFT> 8 #23 вводится и MP:=MP+1

EM:E00- #E0 вводится в текущую ячейку и затем MP:=#E0.

Пробел ограничивает команду 'M'

8C0 #8C вводится и MP обновляется (т.к. использована команда 'O' для назначения относительного смещения #8C)

2A5D- #5D вводится и MP не меняется, т.к. ограничитель пробел, а не команда.

Если 16-ричное число вводится в ответ на подсказку и ограничивается точкой, то введенное число будет занесено в текущий регистр Z80, адресуемый стрелкой.

На входе MONS4 стрелка '>' указывает на PC, и использование '.', как ограничителя 16-ричного числа, будет модифицировать PC. Если вы хотите изменить другие регистры, то используйте точку саму, а не как ограничитель числа. Указатель '>' будет циклически указывать регистры от PC до AF. Отметим, что невозможно адресовать и изменять SP или IY регистры. Например:

Предположим, что '>' первоначально указывает на PC:

```
.      Указывает на IY
.      Указывает на IX
0.     Устанавливает IX в 0
.      Указывает на HL
123.   HL:=#123
.      Указывает на DE
.      Указывает на BC
E2A7.  BC:=#E2A7
.      Указывает на AF
FF00.  AF:=#FF и сбрасывает все флаги
.      Указывает на PC
8000.  PC:=#8000
```

Отметим, что точка также может использоваться для модификации адреса альтернативного набора регистров, если он отображается. Используйте команду 'Q' для переключения набора регистров.

Пример отображения фронтальной панели

```
710C   2007    JR NZ,#7115
>PC    710C    20 07 EB C9 EB 19 EB
SP     D0AF    8A 70 06 03 0A 03 0D
IY     CF6A    0D 11 0C 0F 09 18 8
IX     D09F    04 03 04 00 00 00 1B
HL     2A1B    DF FE 29 28 02 CF 02
DE     0000    F3 AF 11 FF FF C3 CB
BC     0004    FF C3 CB 11 2A 5D 5C
AF     0304    >
IR     3F7C ON

7100   AF 7108 37 7110 EB
7101   ED      7109 C0    7111 19
7102   52      710A B3    7112 EB
7103   19      710B 5A    7113 29
7104   30     >710C 20<   7114 D8
7105   01      710D 07    7115 1F
7106   EB      710F C9    7117 FB
7107   B2      710F C9    7117 FB
>
```

Выше показано типичное изображение Ф.П. (одно из изображений, полученных при пошаговом проходе по п/п MULT в примере использования команды Z).

Первые 9 строк изображения содержат 9 регистров, сначала — имя регистра (от PC до IR), затем (для PC...BC) — значения содержимого регистров, и затем — содержимое 7 ячеек памяти, начиная с адреса, находящегося в регистре.

Регистр флагов декодируется, чтобы показать текущие значения битов флагов, установленных так, как они используются внутри регистра. Если регистр флагов установлен в #FF, то за отображением имени AF будет следовать:

00FF CZ H UNS

т.е. знак, ноль, полуноситель, паритет/переполнение, сложить/вычесть и будут установлены все флаги носителей. Справа от регистров IR находятся слова ON или OFF, которые показывают текущее состояние внешних прерываний. Указатель регистров '>' указывает на регистр, который адресуется в данный момент (см. раздел 2).

24 байта памяти отображаются ниже отображения регистров. Отображение состоит из адреса (2 байта, 4 символа), за которым следует значение (1 байт, 2 символа) памяти по этому адресу. Отображение размещено вокруг текущего значения MP, указываемого '>'.

Команды вводятся вверху экрана в ответ на подсказку '>'.

Отображение обновляется после каждой отработанной команды.

4.3. GENS–4

Раздел 1. Запуск

1.1. Введение и инструкции загрузки

ENS4 — это мощный и легкий в использовании ассемблер для Z80, который очень близок к стандартному ассемблеру ZILOG по определению. В отличии от многих других ассемблеров, пригодных для компьютеров 'SPECTRUM', GENS4 является обширной профессиональной частью программного обеспечения и побуждает вас очень тщательно изучать следующие разделы вместе с примером приложения 3 перед попытками использования ассемблера.

Если вы еще новичок, то вначале проработайте примечание 3 или проконсультируйтесь в одной из отличных книг, приведенных в библиографии.

У нас есть версии систем DEVPAC 4 на диске, для DISCIPLE $ DISCOVERY дисковой системы и для компьютеров SPECTRUM PLUS 3. Эти версии работают точно так же, как описано в этом документе, просто заменяйте слово 'микропривод', там где оно встречается, на слова 'диск OPUS', 'диск PLUS 3' или 'DISCIPLE диск' в соответствии с вашей системой. Существует несколько внешних особенностей для версии PLUS 3, которые описаны на дополнительном листке.

GENS4 занимает приблизительно 10K в памяти и использует свой внешний стек, так что это — замкнутая часть программного обеспечения. Она содержит собственный строчный редактор, который помещает текстовой файл сразу за кодами GENS4, в то время как таблица символов ассемблера создается за текстовым файлом, поэтому вы должны предоставить достаточное пространство, чтобы поместить сам ассемблер, таблицу символов максимального размера и текстовый файл, который вы, вероятно, будете использовать в текущем сеансе. Поэтому загружать GENS4 часто удобнее в нижние адреса памяти.

Существует 2 версии ассемблера на кассете, обе на стороне 1. Сначала помещена версия с 51 символом в строке, за ней следует рассматриваемая версия с 32 символами. Их имена на ленте являются соответственно GENS4-51 и GENS4 и вы должны использовать ту версию, которая больше вам подходит. Версия GENS4-51 на 400 байт длиннее, чем GENS4.

Для загрузки GENS4 поместите магнитную ленту с ним в кассетный магнитофон и введите:

LOAD '' '' CODE XXXXX ENTER и нажмите клавишу 'воспроизведение' магнитофона. Или:

LOAD 'GENS4' CODE XXXXX ENTER или:
LOAD 'GENS4-51' CODE XXXXX ENTER,

где: XXXXX — десятичный адрес, с которого вы хотите загрузить GENS4.

Только после того, как вы загрузите коды GENS4 в компьютер, вы можете войти в ассемблер, набрав:

RANDOMIZE USR XXXXX, где

XXXXX — адрес, с которого загружены коды ассемблера.

Если в любой последующий момент вы вновь пожелаете войти в ассемблер, то вы должны просто адресоваться к ячейке XXXXX, которая хранит предварительно созданный файл информации.

Например, вы хотите загрузить ассемблер с десятичного адреса 26000. Для этого наберите:

LOAD '' '' CODE 2600 ENTER
RANDOMIZE USR 2600 ENTER

Для перезапуска ассемблера используйте:

RANDOMIZE USR 2600 в пределах бейсика.

Только после ввода GENS4 на экране возникнет подсказка в виде символа '>' (подсказка команд редактора).

Раздел 3 содержит информацию о том, как нужно вводить и редактировать текст, а раздел 2 — о том, что нужно вводить.

1.2. Получение копии системы

После загрузки GENS4 в память вашего SPECTRUM'а вы можете получить копию ассемблера следующим образом:

SAVE 'GENS4-51' CODE XXXXX,11392 ENTER или
SAVE 'GENS4'2 CODE XXXXX,11010 ENTER для кассеты
SAVE * 'M';1;'GENS4-51' CODE XXXXX,11392 ENTER или
SAVE * 'M';1;'GENS4' CODE XXXXX,11392 ENTER для микродрайва

где: XXXXX — адрес, с которого вы загрузили GENS4.

Вы должны сделать эту копию перед входом в GENS4 для того, чтобы сохранить перемещаемую информацию внутри программы.

Инструкции получения копий для DEVPAC серии PLUS смотрите на дополнительном листке.

Раздел 2. Подробности GENS–4

2.0. Принцип работы GENS4, ключи ассемблера, формат листинга

GENS4 является быстрым двухпроходным ассемблером для Z80, который ассемблирует всю стандартную мнемонику для Z80 и содержит следующие особенности: макросы, условное ассемблирование, множество команд ассемблера и таблицу символов в виде двоичных троек.

Когда вы вызываете ассемблирование, то вы используете команду 'A':

A4,2000,1:TEST ENTER

Первый код (4 — см. выше) после A определяет условия трансляции, которые вы хотите иметь в данный момент. Эти условия трансляции и их ключи будут перечислены ниже. Если вы не хотите использовать ключи, то номер набирать не надо, поставьте только запятую.

Второй код (2000 — см. выше) — это десятичный размер символов в байтах. По умолчанию (при простом использовании запятой без кодов), GENS4 будет выбирать размер таблицы символов, подходящий к размеру исходного файла — обычно это бывает вполне приемлемо. Однако, при использовании ключа 'включить' вы можете задать размер таблицы символов больше обычного, ассемблер в этом случае не будет определять размер таблицы символов, которая будет построена.

После указания размера таблицы символов вы можете ввести имя файла, например, 1:TEST, как указано выше. Если вы это сделаете, то результирующий объектный код, полученный в результате ассемблирования, будет сохранен автоматически, независимо от размера генерируемого кода. Если вы не хотите использовать такую возможность, то имя файла не вводите. Не вводите вторую запятую, иначе GENS4 будет считать, что вы вводите пустое имя файла! Более подробно с командой 'A' можно познакомиться в приложении 3.

Ключи ассемблирования:

1 — выдает листинг таблицы символов в конце второго прохода ассемблирования;

2 — запрет на генерацию объектного кода;

4 — производит листинг ассемблирования (отметим, что в ранних версиях ассемблера такой возможности не было);

8 — направить листинг ассемблирования на печать;

16 — просто помещает объектный код, если он, конечно, получен, после таблицы символов. Счетчик адресов обновляется таким образом, что объектный код может быть помещен в одну область памяти, а работать — в какой-либо другой;

32 — исключение проверки того, куда помещается объектный код (это бывает полезно для ускорения ассемблирования);

Для того, чтобы скомбинировать ключи, просто складывайте их один с другим. Так, код 'A33' производит быстрое ассемблирование — не выдается листинг, не производится проверка. Чтобы увидеть, куда в настоящий момент помещают объектный код, при этом в конце выдается листинг таблицы символов.

Отметим, что при использовании ключа 'A16' директива 'ENT' не будет иметь эффекта. Если использован ключ 'A16', то вы можете определить, где находится объектный код, используя команду редактора 'Y'. По этой команде можно определить конец текста (его называют второй номер) и затем добавить к нему назначенный размер таблицы +2.

Ассемблирование происходит за два прохода: в течение 1-го прохода GENS4 ищет ошибки и собирает таблицу символов, второй проход генерирует объектный код (если не указан ключ 2). В течение первого прохода на экране и принтере ничего не отображается, пока не встретится ошибка. В случае ошибки будет выдано сообщение об ошибке с номером ошибки за ним (см. приложение 1). Ассемблирование приостанавливается — нажмите клавишу 'E' для возврата в редактор или любую другую клавишу для продолжения ассемблирования следующей строки.

В конце первого прохода будет выдано сообщение:

PASS 1 ERRORS: NN

Если хоть одна ошибка будет обнаружена, то ассемблирование затем будет остановлено и перехода ко второму просмотру не произойдет. Если какие-нибудь метки были обнаружены в поле операнда, но не объявлены в поле метки, то последует сообщение:

WARNING LABEL ABSENT

В течение второго прохода генерируется объектный код (если генерация не запрещена ключом 2 (см. выше)).

Во время второго прохода листинг ассемблера не производится, кроме случая, когда указан ключ 4 или команда ассемблера L+.

В 32-х символьной версии листинг ассемблера состоит обычно из двух строк в следующей форме:

```
C000 210100 25 LABEL
LD HL,1
1 6 11     21... 26
```

тогда как в 51-символьной версии листинг печатается в одну строку. Если листинг не помещается в одну строку, то он завершается на следующей.

На первом месте в строке стоит значение счетчика адресов, который соответствует данной строке. Если в строке встречаются инструкции ORG, EQU или ENT (см. 2.7), то на первом месте будет стоять значение поля операнда инструкции. Эта запись обычно отображается в шестнадцатиричном виде, хотя используя команду '*D+' ее можно отобразить и в неудобном десятичном виде (см. 2.9).

Следующая запись с позиции 6 занимает до 8 символов в строке (представляет до 4-х байтов) и является объектным кодом текущей инструкции (см. ниже команду *C).

Затем следует номер строки — целое число в диапазоне 1... 32767.

Позиции 21... 26 первой строки содержат 6-символьную метку, определенную в этой строке.

После метки следует мнемоника инструкции, которая занимает поз. 21... 24. (в 32-символьной версии это будет новая строка, если только не использовать команду *C).

Затем, с позиции 2, следует поле операндов и завершают строку комментарии, которые при необходимости можно продолжить на другой строке.

Команда *C ассемблера может быть использована для получения короткой строки листинга ассемблирования — ее вводят, чтобы не включать 9 символов, представляющих объектный код инструкции. Это дает возможность отобразить большинство строк ассемблера в виде, удобном для экрана со строкой в 32 позиции (см. ниже разд. 2.8.).

Изменение формата листинга. (только для 32-символьной версии)

С помощью команды 'POKE' вы можете модифицировать форму, по которой каждая строка листинга разбивается внутри 32-символьной версии GENS4 на 3 части. Указания, как это сделать, приведены ниже. Мы отличаем понятие 'строка ассемблера', которая является текущей строкой листинга ассемблера и хранится во внешнем буфере, от понятия 'экранная строка', которая является строкой, действительно возникающей на дисплее. Строка ассемблера обычно порождает более одной экранной строки.

1. 51-й (#33) байт от начала GENS4 содержит значение, определяющее позицию, до которой будет отображаться строка листинга на экране. Циклически меняя этот байт от 0 до любого другого значения (256) можно заканчивать первую экранную строку в необходимом месте (это особенно полезно при использовании микроформатного принтера).

2. 52-й (#34) байт от начала GENS4 дает номер позиции, с которой должна начинаться каждая последующая экранная строка.

3. 53-й (#35) байт от начала GENS4 указывает количество символов остатка строки ассемблера, который будет выведен во второй экранной строке после первой.

Например, вы хотите, чтобы каждая первая экранная строка содержала 20 символов (т.е. не включала бы поле метки) и затем каждая последующая экранная строка начиналась с позиции 1 и занимала бы всю строку. Также предположим, что вы загрузили GENS4 с десятичного адреса 26000. Чтобы получить вышеуказанные изменения, надо из бейсика ввести следующие инструкции:

```
POKE 26051,20
POKE 26052,1
POKE 26053,31
```

Вышеприведенные модификации возможны только в том случае, когда не была использована команда *C.

Листинг может быть приостановлен в конце строки нажатием CAPS SHIFT и SPACE вместе. В последующем нажимайте клавишу 'E' для возврата в редактор или любую другую клавишу для продолжения листинга.

Единственными ошибками, которые могут появиться во втором проходе, являются ошибки типа '*ERROR*10' (см. Прил. 1) 'BAD ORG!' (которая появляется, когда объектный файл накладывается на GENS4, исходный файл или на таблицу символов — это может быть предотвращено использованием другого формата) *ERROR* 10 — это не фатальная ошибка, поэтому вы можете продолжать ассемблирование, как и на первом проходе, тогда как ошибка 'BAD ORG!' является фатальной и сразу же возвращает управление редактору.

В конце второго прохода на дисплее будет отображено сообщение, следуя за предыдущим об отсутствующих метках:

```
PASS 2 ERRORS:NN
```

Затем появится следующее сообщение:

```
TABLE USED: XXXXX FROM YYYYY
```

Это сообщение информирует о том, какая часть таблицы символов была заполнена. В этом случае, если была правильно использована директива 'ENT', выдается сообщение:

```
EXECUTES: NNNNN IS DISPLAYED
```

Это показывает стартовый адрес задачи. Вы можете выполнить задачу, используя команду редактора 'R'. Будьте осторожны при использовании команды 'R' в том случае, когда вы не завершили успешно ассемблирование и не увидели сообщения 'EXESUTES:NNNNN'.

Итак, если указан ключ '1', то используется список меток в алфавитном порядке и будут напечатаны соответствующие им значения. Количество информации, отражаемой в одной строке, может быть изменено директивой 'POKE' с адреса 'начало GENS4 + 50' помещением в эту ячейку уместного значения, по умолчанию заносится 2.

Контролируйте возврат в редактор.

2.1. Формат оператора ассемблера

Каждая строка текста, которая будет обрабатываться GENS4, должна иметь следующий формат, где определение поля необязательно:

```
метка   код команды      операнды комментарий
BEG     LD               HL,LABEL ;получение метки
```

Пробелы и табуляция обычно игнорируется. Строка обрабатывается следующим образом:

Первый символ строки проверяют и последующие действия зависят от его сущности:

; — целую строку трактуют как комментарий, т.е. просто игнорируют;

* — предполагается, что следующий символ (символы) образуют команду ассемблера (см. пункт 2.8). Все символы после команды трактуются как комментарии;

<CR> — символ конца строки. Линия просто игнорируется;

— — пробел или табуляция. Если первым символом строки является пробел или табуляция, то GENS4 предполагает, что следующий отличный от пробела или табуляции символ будет началом кода команды Z80.

Если первый символ строки отличается от вышеприведенных, то в строке должна быть метка — см. п. 2.2.

После обработки присутствующей метки (или если первый символ в строке — пробел или табуляция) ассемблер исследует следующий значащий символ и предполагает, что он будет или символом конца строки, или началом команды Z80 (см. Прил. 2), состоящий из 4-х символов и ограниченный пробелом (табуляцией или <CR>).

Иногда присутствует код команды и требуется один или более операндов, тогда поле операндов следует после набора пробелов (табуляции).

Оператор может состоять только из одной, это бывает полезно для улучшения читаемости листинга.

Комментарии могут следовать в любом месте после поля операндов или поля кода команды (если у нее нет аргументов).

2.2. Метки

Метка — это символьное имя, представляющее 16 бит информации. Метка может быть использована для выделения адреса особенной инструкции или для выделения области данных, или может быть использована как константа для директивы 'EQU' (см. п.2.7).

Если метку представить значением более 8 битов и она затем используется в контексте применительно к 8-битовой константе, то ассемблер будет выдавать сообщение об ошибке:

```
LABEL EQ    #1234
      LD    A,LABEL
```

Будет выдано *ERROR*10, когда во время второго прохода идет обработка второго операнда. Метка может содержать любое количество символов (см. ниже), хотя только первые 6 символов — значащие. Эти первые 6 символов должны быть уникальными, иначе метка будет повторно определена (*ERROR*). Метка не должна образовывать зарезервированное слово, хотя зарезервированное слово можно внедрить как часть метки.

В метках можно использовать символы 0... 9, A... Z и $. Можно использовать как большие, так и малые буквы, символы .\,, ,#,и -. Метка должна начинаться с буквы. Некоторые примеры возможных меток представлены ниже:

```
LOOP
A-LONG-LABEL
L1
L2
```

A-
LDIR —это незарезервированная инструкция
TWO S

2.3. Таблица символов

Когда метка встречается в первый раз, ее помещают в таблицу вместе с двумя признаками, которые позже показывают, как эту метку соотносят по алфавиту с другими, находящимися внутри таблицы. Если в первый раз метка появляется в поле меток, то ее значение (данное счетчиком адресов или значением выражения 'EQU') заносят в таблицу символов. В другом случае значения заносятся, когда бы ни встретилось имя метки в последующем в поле метки.

Данный тип таблицы символов называется таблицей символов типа двоичного дерева и ее структура дает возможность заносить имена и извлекать их за очень короткое время — это существенно для больших программ.

Запись в таблице занимает от 8 до 13 бит, в зависимости от длины имени.

Если за время первого прохода имя встречается более одного раза, то будет выдано сообщение об ошибке (*ERROR*4), так как ассемблер не знает, какое значение должно соответствовать имени метки.

Если значение для имени не найдено, то в конце ассемблирования будет выдано сообщение:

WARNING SYMBOL ABSENT Отсутствие определения имени не мешает продолжению ассемблирования.

Отметим, что только первые 6 символов метки вводятся в таблицу символов, что определяется ее размером. В конце ассемблирования вам могут быть выданы сообщения статистики, о том, как много памяти было использовано таблицей символов в течение трансляции — вы можете изменить максимальный размер памяти, отведенный под таблицу символов.

2.4. Выражения

Выражения — это запись операндов, образованная или простым термом или комбинацией термов, разделенных операндом. Ниже определены понятия терма и операнда:

Терм:

десятичная константа, например:	1024
шестнадцатиричная константа, например:	#4a5
двоичная константа, например:	%100101
символьная константа, например:	'A'
метка, например:	L1029

Может также использоваться для обозначения текущего значения счетчика адресов.

Оператор:

+	— сложение
-	— вычитание
$	— логическое 'И' (AND)
	— логическое 'ИЛИ' (OR)
!	— логическое 'XOR'
*	— алгебраическое умножение
/	— алгебраическое деление
?	— MOD-функция (A?B=A-(A/B)*B

Замечание:

— используют для того, чтобы отметить начало 16-ричного числа.

% — используют для того, чтобы отметить начало двоичного числа.

' — символьная константа

При считывании числа (десятичного, шестнадцатиричного или двоичного) GENS4 выбирает последние значения 16 бит числа (т.е. MOD 65536), например, 70016 станет 4480 и #5A2C4 становится #A2C4.

Обеспечивается широкий набор операндов, но их приоритет не соблюдается: выражения вычисляются строго слева направо. Операторы *,/ и ? приведены для версий с дополнительными возможностями и не являются частью данного набора выражений, который мог бы увеличить размер GENS4. Если выражение заключено в круглые скобки, то его представляют как содержимое адреса памяти. Так, в инструкции LD HL,(LOC+5) в регистровую пару HL загружается 16-ти битовое значение, содержащееся в ячейке памяти с адресом LOC+5.

Некоторые инструкции Z80 (JR и DJNZ) предполагают операнды, которые предполагают 8-ми битовые значения, а не 16-ти битовые — это называется относительной адресацией. Когда имеет место относительная адресация, GENS4 автоматически выдает значение счетчика адресов следующей инструкции из значения, представленного в поле операндов текущей инструкции для того, чтобы получить относительный адрес для текущей инструкции. Область допустимых значений относительного адреса простирается от -128+значение счетчика адресов следующей инструкции до значения счетчика адресов следующей инструкции +127.

Если же вы желаете определить относительный переход от значения счетчика адресов текущей инструкции, то вы должны использовать символ $ (резервную инструкцию), за которым следует требуемое смещение. Относительно значения счетчика адресов текущей инструкции смещение должно находиться в диапазоне от -126 до +129 включительно.

Примеры выражений:

5000 — LABEL
%1001101!%1011 — дает 1000110
#3456?#1000 — дает 456
4+5*3-8 — дает 19
$-LABEL+8
2345/7-1 — дает 334
'Y'-';'+7
(5*LABEL-#1000$%1111)
17$%1000 — дает 25

Отметим, что пробелы могут быть помещены между термами и операторами и наоборот, но не внутри термов.

Если операция умножения получит ответ с абсолютным значением, большим чем 32767, то появится ошибка *ERROR*15, тогда как при делении на 0 — ошибка *ERROR*14, в противном случае переполнение игнорируется.

Вся арифметика использует вторую дополнительную форму, где любое число, большее чем 32767, представляется как отрицательное, например:

60000=-5536 (60000-65536)

2.5. Макроопределения

Макроопределения позволяют вам писать более плотные и более значимые ассемблерные программы, но они должны использоваться осторожно и не должны конфликтовать с подпрограммами.

Макроопределение состоит из набора инструкций ассемблера вместе с именем макроопределения. Когда это имя в последующем используется в поле кода операции, то оно будет заменено на все инструкции ассемблера, составляющее это макроопределение.

Так, макроопределение NSUB может быть определено следующим образом:

```
NSUB  MAC
      OR A
      SBC HL,DE
      ADD HL,DE
      ENDM
```

Когда бы мы в дальнейшем ни применили NSUB как код операции, оно будет генерировать три инструкции ассемблера: OR A. SBC HL,DE и ADD HL,DE.

Это спасет вас от лишнего печатания и сделает вашу программу легче для понимания, но вы должны помнить, что при каждом появлении NSUB генерируется объектный код, и, может быть, эффективнее использовать CALL для вызова вместо нее подпрограммы. Ниже мы приводим формат макроопределения и его вызов вместе с некоторыми примерами. Изучайте их, пожалуйста, внимательно.

Макроопределение имеет следующую форму:

```
имя   MAC
      ...
      ...
      тело макроопределения
      ...
      ...
      ENDM
```

По имени макроопределения будет вызываться его текст, когда это имя в последующем встретится в поле кода операции. 'MAC' показывает начало макроопределения, а 'ENDM' указывает на его конец.

Параметры макроопределения могут упоминаться внутри макроопределения путем использования знака равенства, за которым следует номер параметра (0...31) включительно. Таким образом, макроопределение

```
MOVE  MAC
      LD HL,=0
      LD DE,=1
      LD BC,=2
      LDIR
      ENDM
```

имеет три параметра: адрес источника, адрес приемника и длину, загружает нужные значения в HL, DE, и BC и затем выполняет инструкцию LDIR. Для того, чтобы вызвать макроопределение из тела вашей программы, просто используйте его имя в поле ко-

манд. За именем макроопределения должны следовать необходимые вам три параметра:

```
MOVE 16384,16385,4096
```

В этом примере мы используем особенные адреса, но в действительности вы можете работать с любыми существующими выражениями для определения значения параметра макроопределения:

```
MOVE START,START+1,LENGTH
```

Подумайте, являются ли вышеприведенные примеры хорошим использованием макроопределения? Не могли ли вы использовать подпрограмму?

Внутри макроопределения параметры могут появляться в любом возможном выражении, например:

```
HMC   MAC
      LD HL,-0*3600
      LD DE,-1*60
      ADD HL,DE
      LD DE,-2
      ADD HL,DE
      ENDM
```

Это макроопределение имеет три параметра: часы, минуты, секунды. Макроопределение вычисляет общее количество секунд, определяемое параметрами, и помещает его в регистр HL. Вы можете использовать его следующим образом:

```
HOURS     EQU 2
MINUTES   EQU 30
SECONDS   EQU 12
START     EQU 0
          HMC     HOUR,MINUTE,SECOND
          LD      DE,START
          ADD     HL,DE ;HL — окончательное
                        ;время
```

Макроопределения не могут быть вложенными, так что вы не сможете ни определить макроопределение, ни вызвать макроопределение из макроопределения.

Во время ассемблирования, как только имя макроопределения встречается в поле команды, ассемблируется его текст. Обычно этот текст тела макроопределения не попадает в ассемблерный листинг и печатается только для макроопределения. Однако вы можете усилить листинг расширением макроопределения, используя команду ассемблера *M+, а, используя *M-, запретить распечатывать текст макроопределения.

Если вы вышли за пределы пространства, отведенного под буфер макро, то будет выдано сообщение и прервется ассемблирование. Используйте команду редактора 'C' для резервирования буфера больших размеров.

2.6. Директивы ассемблера

GENS4 распознает определенные псевдокоманды — так называемые директивы ассемблера. Они не влияют на процессор Z80 при прогоне, т.е. не декодируются, а просто заставляют ассемблер выполнять определенные действия во время трансляции. Эти действия определенным образом изменяют код, генерируемый GENS4.

Псевдокоманды ассемблера точно такие же, как и выполняемые инструкции: они могут помещаться за меткой (необходимо для EQU) и за ними может следовать комментарий.

Возможны директивы:

ORG <выражение> — устанавливает счетчик адресов равным значению выражения. Если ключи 2 и 16 оба не указаны и результат ORG попадает в область GENS4, текстового файла или таблицы символов, то появляется выражение 'BAD ORG!' и прекращается трансляция (см. п.2.0.). Для более детального ознакомления с тем, как ключи 2 и 16 влияют на использование ORG см. команду 'A' в разделе 3 для некоторых предосторожностей в использовании ORG, когда автоматически сохраняется объектный код.

EQE <выражение> — этой директиве должна предшествовать метка. Значению метки присваивается значение выражения. Выражение не должно содержать имен, которым еще не присвоено значение. В противном случае ассемблер выдает вам сообщение об ошибке — *ERROR*13.

DEFB <выражение>, <выражение>... — каждое выражение должно оценивать 8 бит; байт, адрес которого в настоящий момент содержится в счетчике адресов, принимает значение выражения и счетчик адресов увеличивается на 1. Все вышеуказанное повторяется для каждого выражения.

DEFW <выражение>, <выражение>... — дополняет слово (2 байта), адрес которого указан счетчиком адресов, значением выражения и увеличивает счетчик адресов на 2. Младший байт помещается сначала, за ним следует старший байт. Повторяется для каждого выражения.

DEFS <выражение> — увеличивает счетчик адресов на значение выражения. Эквивалент резервирования области памяти размером, равным значению выражения.

DEFM 'S' — определяет, что последовательность N байтов памяти будет будет равна представлению строки 'S' в коде ASCII, где N — длина строки. Теоретически, длина строки может быть от 1 до 255 включительно, но практически она ограничена длиной строки, которую можно ввести редактором. Первый символ в поле операндов (', как показано выше) является ограничителем и строка 'S' определяется только символами, лежащими между двумя ограничителями; символ конца строки также действует как ограничитель строки.

ENT <выражение> —эта директива никак не влияет на генерируемый объектный код, она просто используется для определения адреса, на который осуществляется переход по команде редактора 'R'. Выражение ENT устанавливает этот адрес равным значению выражения. Используется совместно с командой редактора 'R' (см. разд. 3). Для выполнимого адреса нет умолчания.

2.7. Команды условной трансляции

Команды условной трансляции обеспечивают программисту возможность включения или не включения определенных секций исходного текста в процесс ассемблирования. Это возможно при использовании команд IF, ELSE, END.

IF <выражение> — эта команда оценивает выражение. Если результат равен нулю, то прекращается ассемблирование следующих линий до тех пор, пока не встретится ELSE или END. Если значение выражений отлично от нуля, то ассемблирование происходит нормально.

ELSE — эта команда просто переключает ассемблирование. Если перед появлением ELSE было ассемблирование, то в последующем оно выключается и наоборот.

END — просто включает ассемблирование.

Отметим, что команды условной трансляции не могут быть вложенными, на вложенность 'IF' не сделано никаких проверок и любая попытка вложения может привести к непредсказуемым результатам.

2.8. Команды ассемблера

Команды ассемблера подобно директивам ассемблера не влияют на работу ассемблера Z80 т.к. они не декодируются в объектные коды. Однако, в отличие от директив ассемблера, они не влияют на объектный код, производимый ассемблером — команды ассемблера просто модифицируют формат листинга. Команда ассемблера — это строка исходного текста, которая начинается с символа *.

Надпись после звездочки определяет тип команды и должна быть выполнена заглавными буквами. Ограничением строки может быть любой текст. Исключение составляют команды L и D, предполагающие знаки '+' или '-' после команды. В распоряжении программиста имеются следующие команды:

*E — выдает 3 пустые строки на экран или принтер — это полезно для разделения модулей.

*HS — выдает строку 'S' — заголовок, который будет печататься после каждого вывода '*E'. *H автоматически выполняет *E.

*S — указывает, что с данной строки листинг должен быть остановлен. Листинг может быть продолжен нажатием любой клавиши. Полезно для чтения адресов в середине листинга.

Примечание: если *S появляется после *L-, то листинг не останавливается.

*L+ — прекращает листинг и печать, начиная с данной строки.

*L- — начинает листинг и печать, начиная с данной строки.

*D+ — указывает, что значение счетчика адресов в начале каждой строки должно выдаваться в десятичном виде вместо обычного 16-ричного.

*D- — возвращает к использованию шестнадцатиричного вида значения счетчика адресов в начале каждой строки.

*C- — укорачивает листинг ассемблера, начиная со следующей строки. Листинг характеризуется тем, что он не содержит объектного кода, генерируемого текущей строкой — это укорачивает строку листинга на 9 символов и делает строку ассемблера более удобной для дисплея с 32-мя символами в строке, что улучшает читаемость.

*C+ — возвращает к полному листингу, как описано в разделе 2.0.

*M+ — включает печать макроопределения.

*M- — выключает печать макроопределения.

*F <имя файла> — это очень мощная команда, позволяющая вам ассемблировать текст с ленты или дисковода — этот текст читается с ленты или дисковода в буфер, блокируется там на время и затем ассемблируется из этого буфера; это позволяет создавать объектный код большого размера, только если исходный ассемблируемый текст не занимает много места в памяти. Имя текстового файла (до 10 символов), который вы желаете проассемблировать таким образом не обязательно должно быть определено и ему должен предшествовать пробел. Если файл находится на кассете, то вы указываете это, набивая перед именем файла номер привода с двоеточием:

*F 2:TEST — для файла на микроприводе N2;

*F TEST — для файла на ленте.

Если не указано имя файла, то с ленты считывается первый найденный файл, но при считывании с дисковода вышеуказанное нельзя принимать в расчет. Если вы работаете с дисководов, то файл должен быть предварительно сохранен с помощью команды редактора 'P' обычным способом.

Если файл находится на ленте, то вы должны предварительно загрузить его с ленты при помощи команды редактора 'T', а не 'P' — это необходимо, так как текстовый файл, содержащийся на ленте, должен быть преобразован в блоки с достаточным размером межблочных интервалов, что позволяет ассемблировать текущий блок перед тем, как следующий начнет грузится с ленты. Размер блока, используемого данной командой (и командой редактора 'T'), устанавливается с помощью команды редактора 'C' (см. следующий раздел). Возможность выбора размера этого буфера позволяет вам оптимизировать отношение размер/скорость для любых выводов текстов с ленты. Например, если вы не собираетесь использовать команду 'F' во время ассемблирования, то полезно будет определить размер буфера равным 1 для минимизации пространства, занимаемого GENS4 и минимизации рабочего пространства.

Где бы ни встретил ассемблер команду 'F', он определит, откуда нужно считывать файлы, с дискеты или с ленты. Это происходит на обоих просмотрах, так как на каждом просмотре необходимо сканировать текст. Если считывается лента, то на ней ищется файл с требуемым именем или первый файл.

Если имя файла во время его поиска на ленте не соответствует заданному, то выдается сообщение: 'FOUND FILE' и поиск продолжается, в другом случае выдается сообщение: 'USING FILE', файл загружается блок за блоком и обрабатывается. Подробнее ознакомиться с 'F' вам поможет пример приложения 3.

Остальные команды ассемблера обрабатываются только на втором просмотре.

Если ассемблирование было прервано одной из команд условной трансляции, то и работа любой команды ассемблера также прерывается.

Замечание: описанные возможности не годятся для систем DISCIPLE, OPUS и PLUS 3 DEVPAC 4.

Быстрее и легче вместо ленты использовать диск.

Следствием этого является то, что в этих версиях команда 'C' не позволяет вам определить размер буфера загрузки и команда 'T' не существует.

Раздел 3. Строчный редактор

3.1. Введение в редактор

Редактор, поставляемый со всеми версиями GENS4, является простым строчным редактором, разработанным для обслуживания всех операционных систем, сделанных для Z80. Редактор прост в использовании и дает возможность редактировать программы просто и эффективно.

Для уменьшения размера текстового файла редактор выполняет определенное сжатие пробелов. Это происходит по следующей схеме: когда вводится строка с клавиатуры, она символ за символом заносится во внешний относительно ассемблера буфер и затем, в конце строки (когда вы нажимаете клавишу <ENTER>), строка перемещается из буфера в текстовый файл. Во время этого перемещения происходит определенное сжатие пробелов: если первый символ строки — пробел, то в текстовый файл вводится символ табуляции, и все последующие пробелы пропускаются. Если первый символ в строке не пробел, то символы заносятся из буфера в текстовый файл, пока не встретится пробел, после чего обработка ведется также, как если бы следующий символ был бы первым символом в строке. Это продолжается и в дальнейшем. В результате символы табуляции включаются в начале строки или между меткой и кодом операции, а также между кодом операции, операндами и комментариями. Конечно, если код возврата каретки (ENTER) встречается в любое другое время, то преобразование завершается и управление передается редактору.

Этот процесс сжатия понятен, и вы можете просто использовать клавишу '—>' для того, чтобы получить кратко протабулированный файл, который также экономичней для хранения. Заметим, что пробелы не сжимаются внутри комментариев и что пробелы не должны присутствовать в полях меток, кодов операций и операндах.

Режим редактора включается автоматически при запуске GENS4 и за вспомогательным текстом следует подсказка редактора '>'.

В ответ на подсказку вы можете ввести командную строку следующего формата:

C N1, N2, S1, S2 <ENTER>, где:

'C' — мнемоника команды, которую необходимо выполнить;

N1 — число в пределах от 1 до 32767 включительно;

N2 — число в пределах от 1 до 32767 включительно;

S1, S2 — строка не более чем из 20 символов.

Запятая используется для разделения различных аргументов, хотя это можно изменить — см. команду 'S'. Пробелы игнорируются во всех случаях кроме тех, когда они находятся внутри строк. Никакой из аргументов не является обязательным, хотя некоторые команды, например, команда 'DELETE', не будут работать, если опущены аргументы N1 и N2.

Редактор помнит ранее введенное число и строки и использует сформированное значение для применения, если вы не определили другие значения внутри командной строки. Первоначально значения N1, N2 устанавливаются равными 10, а строки пустыми.

Если вы вывели неправильную командную строку, например, F - 1,100,HELLO, то эта команда будет проигнорирована и появится сообщение 'PARDON?'. При этом вы должны ввести командную строку правильно, например, F1,100,HELLO.

Это же сообщение появится, если длина строки S2 превышает 20 символов. Если более 20 символов содержит строка S1, то лишние символы игнорируются.

Команды могут вводится как на верхнем, так и на нижнем регистрах.

При вводе команды определенные комбинации ключей могут использоваться для ее редактирования. Так, клавиша '<—' используется для стирания символов в направлении начала строки, '—>' — для продвижения курсора в следующую позицию табуляции, 'CAPS SHIFT 0' или 'DELETE' — для уничтожения предыдущего символа.

Следующий раздел представляет команды, используемые редактором. Заметим, что если аргумент заключен в квадратные скобки, то такой аргумент обязателен для данной команды.

3.2. Команды редактора

3.2.1. Вставка текста

Текст может быть введен в текстовый файл или указанием номера строки, пробела и требуемого текста, или посредством команды 'I'. Заметим, что если сразу за номером строки вы введете <ENTER>, то эта строка будет удалена из текста, если, конечно он существует. Где бы не вводился текст, можно применять клавиши '<—', '—>' и <EDIT> (возврат к метке команды).

Клавиша <DELETE> будет уничтожать предыдущий символ (но не далее начала строки текста).

Текст вводится во внешний буфер внутри GENS4 и вы должны оградить его от переполнения использованием клавиш <DELETE> или '<—' для освобождения свободного пространства. Если во время ввода текста редактор определит, что конец текста перекрывает вершину адресуемой памяти, то выдается сообщение <BAD MEMORY!>. Это показывает, что далее текст вводить нельзя и что текстовый файл или последняя его часть должна быть сохранена на кассете для дальнейшего восстановления.

Команда ' I N,M '

Использование этой команды переводит ввод в автоматический режим: вам подсказываются номера строк, начиная с N, приращением M на каждом шагу. После высвечивающегося номера вы вводите нужный текст, по желанию используя нужные клавиши, и завершаете строку текста вводом <ENTER>. Для выхода из этого режима используется <EDIT>.

Если вы вводите строку с уже существующим номером, то строка текста с этим номером удаляется и заменяется на вновь введенную после нажатия <ENTER>.

Если автоматическое увеличение номера строки дает значение, большее 32767, то происходит автоматический выход из режима ввода.

Если при вводе текста вы добрались до конца строки на экране, но еще не ввели 64 символа (размер буфера), то экран сдвинется вверх на одну строку и вы можете продолжать ввод со следующей строки — номер строки автоматически будет соответствовать введенному тексту.

3.2.2. Распечатка текста

Текст может быть проверен с помощью команды 'L'; номер строки, постоянно отображаемый при выполнении данной команды, устанавливается заранее, но он может быть изменен с помощью команды 'K'.

Команда ' L N,M '

Эта команда выводит листинг со строки N до строки M включительно на терминал. По умолчанию N присваивается значение 1, M - 32767, т.е. значениями по умолчанию не являются ранее введенные аргументы.

Для листинга целого файла просто используйте команду 'L' без аргументов. Строки на экране форматируются по левой границе экрана, так что номер строки отображается ясно. Строка протабулирована автоматически, в результате чего получаем четкое разделение полей в строке. Номер отображаемой на экране строки может быть проконтролирован с помощью команды 'K' — после листинга определенного количества строк листинг будет приостановлен (даже если это еще не

строка M). Для возврата в точку входа в редактор нажмите клавишу <EDIT> или любую другую клавишу для продолжения.

Команда ' K N '

'K' устанавливает количество экранных линий, которые должны отображаться на терминале перед паузой, как это было описано выше (см. команду 'L'). Значение N (не более 256) хранится в памяти. Например, если вы хотите при последующем использовании 'L' выдавать на экран по 5 строк, то введите команду 'K5'.

3.2.3. Редактирование текста

Однажды созданный текст неизбежно будет нуждаться в редактировании некоторых строк. В GENS4 имеются команды, позволяющие исправлять, стирать, перемещать и перенумеровывать строки.

Команда ' D N,M '

Все линии от N до M включительно удаляются из текстового файла. Если M<N или определено менее двух аргументов, то команда игнорируется. Это сделано для предотвращения ошибок из-за небрежности. Одиночная строка может быть уничтожена указанием N=M. Этого же можно достичь простым введением номера строки, за которым следует <ENTER>.

Команда ' M N,M '

Эта команда помещает текст строки с номером N в строку M, уничтожая текст, уже существующий там. Т.е. эта команда позволяет вам перемещать строку текста внутри текстового файла. Если строки с номером N не существует, то команда игнорируется.

Команда ' N N,M '

Команда 'N' перенумеровывает M строк текстового файла, начиная со строки N. N и M должны быть реальными и если номер линии превышает 32767, то остается первоначальная нумерация.

Команда 'F N,M,F,S '

Текст со строки N до строки M исследуется на появление строки F. Если такая строка найдена, то она отображается на терминале и включается режим 'EDIT' (см. ниже).

те ввести командную строку следующего формата:

Команда ' E N '

Редактирует строку с номером N. Если строки N не существует, то команда игнорируется; в противном случае строка копируется в буфер и отображается на терминале вместе с номером строки. Номер строки вновь отображается под строкой и включается режим редактирования. Все последующее редактирование происходит внутри буфера, а не внутри самого текста, так что первоначальная строка может быть получена в любой момент.

В этом режиме курсор отображается движущимся по строке, (начиная с первого символа) поддерживая различные подкоманды, позволяющие редактировать строку. В вашем распоряжении имеются следующие подкоманды:

<SPACE> — перемещает курсор на одну позицию к следующему символу в строке. Вы не можете шагнуть на конец строки.

<DELETE> — возвращает курсор на предыдущий символ в строке. Невозможно шагнуть левее первого символа строки.

< —> > — перемещает курсор на следующую позицию табуляции в каждой экранной строке.

<ENTER> — конец редактирования данной строки. Сохраняет все сделанные изменения.

<Q> — выход из режима редактирования данной строки, т.е. покидание редактируемой строки с игнорированием всех сделанных изменений. Строка остается такой же, как она и была до редактирования.

<R> — перезагружает буфер редактирования текстом, т.е. забывает все сделанные в строке изменения и восстанавливает строку в первоначальном виде.

<L> — распечатывает остаток строки, который должен быть отредактирован, т.е. остаток строки за текущей позицией курсора. Вы остаетесь в режиме редактирования с указателем, перепозиционированным в начало строки.

<K> — уничтожает символ в указываемой курсором позиции.

<Z> — уничтожает все символы, начиная с указанного курсором и до конца строки.

<F> — ищет следующее появление строки 'F', ранее определенной в командной строке (см. команду 'F' выше).

Эта подкоманда будет автоматически выводить систему из режима редактирования текущей строки (сохраняя все изменения) даже если цепь символов 'F' в текущей строке не встретилась. Если цепочка 'F' встретилась в последующих строках текста внутри первоначально определенного диапазона, то будет включен режим редактирования для строки, в которой найдена заданная последовательность символов 'F'. Отметим, что курсор всегда устанавливается в начало найденной строки.

<S> — замещает ранее определенной строкой 'S' текущее появление цепи символов 'F' и затем выполняет подкоманду 'F', т.е. осуществляет поиск следующей строки 'F'. Так, вместе с вышеупомянутой командой 'F' она используется для пошаговой замены строк символов 'F' строками 'S'. (см. раздел 3.3 для примера.)

<I> — вводит символ в указанную курсором позицию. Вы будете оставаться в этом режиме до тех пор, пока не нажмете <ENTER>. По этой клавише вы возвращаетесь в основной режим редактирования с указывающим на последний введенный символ. Используя <DELETE> внутри этого режима, вы уничтожите символ в буфере слева от курсора, тогда как, используя < —> >, переместите курсор в следующую позицию табуляции, включая пробелы внутри буфера.

<X> — перемещает курсор в конец строки и включает описанный подрежим.

<C> — изменяет подрежим. Это позволяет перезаписать символ в текущей позиции, затем передвигать курсор через одну позицию. Вы останетесь в измененном подрежиме до тех пор, пока не нажмете клавишу <ENTER>, нажатие которой возвратит вас в основной режим редактирования с курсором, указывающим на последний измененный символ.

<DELETE> просто сдвигает через одну позицию курсор влево, а < —> > не имеет эффекта.

3.2.4. Команды дисковода и магнитной ленты

Текст может быть сохранен на магнитной ленте/дисководе или загружен с них с помощью команд P, Q и T. Объектный код может быть сохранен на магнитной ленте с использованием команды O. Имена файлов не должны содержать более 10 символов.

Команда ' P N,M,S '

Строки с номерами от N до M сохраняются на ленте/диске в файле с именем, заданным строкой S. Текст будет записан на диск, если перед именем файла через двоеточие стоит номер дисковода. Помните, что эти аргументы могут быть установлены предыдущей командой. Примеры:

P 10,200,EXAMPLE — записать строки 10... 200 на магнитофон в файл с именем 'EXAMPLE'.

P 500,900,1:TEST — записать строки 500... 900 на дисковод 1.

Перед введением этой команды убедитесь, что ваш магнитофон включен в режим записи. Не используйте эту команду, если вы желаете на последующей стадии ввести текст с ленты. Вместо этого используйте команду T. Если вы намерены вводить текст с дисковода, то используйте команду P.

Если вы заносите на диск файл с уже имеющимся на диске именем, то вас спросят:

FILE EXIST DELETE (Y/N)? (существующий файл удалять?)

Отвечайте 'Y' для удаления файла и продолжения записи или нажмите любую другую клавишу для возврата в редактор без записи файла.

Команда ' G,,S '

На ленте или дискете производится поиск файла с именем 'S'. Когда файл найден, он загружается в конец текущего текста. Если строка 'S' пустая, то загружается первый файл с кассеты. Для дисковода обязательно надо указать имя файла и номер дисковода.

При работе с кассетой, после того как вы ввели команду 'G', появится сообщение:

START TAPE

Вы должны нажать клавишу 'воспроизведение' своего магнитофона. Ведется поиск файла с указанным именем или первого файла по умолчанию. Когда нужный файл найден, появится сообщение:

USING FILENAME в противном случае высветится сообщение:

FOUND FILENAME и поиск на ленте продолжается. При использовании дисковода и в том случае, если не найден искомый файл, появится сообщение

ABSENT.

Заметим, что если в памяти уже находится текстовый файл, то файл, загружаемый с ленты, будет добавлен к существующему и строки всего совокупного файла будут перенумерованы, начиная с первого с шагом 1.

Команда T N,M,C

Выводит блок текста между строками N и M включительно на ленту в формате, подходящем для дальнейшей работы под управлением директивы ассемблера *F — см. раздел 2.9. Обрабатывается файл с именем 'S'. Вывод начинается сразу после нажатия клавиши <ENTER>, так что вы должны убедиться, что ваш магнитофон готов к записи перед вводом этой командной линии. Если вы намерены осуществлять ввод с дисковода, то используйте лучше команду 'P', а не 'T'. Отметим, что эта команда должна использоваться только в том случае, если вы хотите позднее ассемблировать текст с ленты. Это неприменимо для диска — только в версиях DEVPAC 4.

Команда ' O,,S '

Выводит ваш объектный код на кассету или дисковод. Имя файла может иметь длину более 8 символов и должно начинаться с номера привода (1... 8) и двоеточия, если вы хотите сохранить объектный код на дискете.

Только последний блок кода, произведенный ассемблером, может быть сохранен этим способом, т.е. если у вас более одной директивы 'ORG' в исходной программе, то сохранится только код, произведенный после последней директивы 'ORG'.

Код должен быть получен в памяти перед тем, как он может быть сохранен с помощью команды 'O'. Часто удобнее использовать команду 'A,,FILENAME' для автоматического получения файла. Эта команда работает быстрее 'O' — см. ниже.

3.2.5. Ассемблирование и запуск из редактора

Команда ' A O,S,F '

Эта команда ассемблирует текст с первой строки текстового файла. Команда 'O' позволяет вам задать ключи, которые должны использоваться во время этой трансляции. Обычно достаточно использовать значения ключей по умолчанию, просто употребляя запятые.

S — дает возможность изменить размер таблицы символов для этой трансляции. Размер таблицы по умолчанию обычно бывает достаточным, за исключением режима ввода (INCLUDING).

F — имя файла, имеющегося на дискете. Имя файла должно начинаться с <D:>, где D — номер дисковода, на котором размещен файл. Имя файла не должно содержать более 10 символов. Присутствие имени файла здесь побуждает ассемблер производить трансляцию иным способом.

Вместо простого ассемблирования объектного кода в память и остановки по достижении вершины памяти, ассемблер не будет транслировать в память до тех пор, пока не достигнет ее вершины (верхнюю границу памяти вы можете установить командой 'U') и затем полученный объектный код будет сохранен на диске в указанном вами файле. Затем ассемблирование будет вновь продолжено с нижних адресов памяти и этот процесс будет продолжаться до тех пор, пока вся программа не будет проассемблирована и сохранена на дискете.

Не существует ограничений на размер программы, которую вы хотите проассемблировать (кроме размера существующего пространства на вашей дискете (кассете)).

Пара важных замечаний относительно использования директивы 'ORG' для данных средств:

1 — директива 'ORG' будет помещать объектный код по адресам, указанным в данной директиве первоначально, и каждый раз после помещения объектного кода в объектный файл (если не был указан ключ 16, чтобы поместить объектный код сразу после таблицы символов).

Следовательно, ключ 16 более разумно использовать будет в том случае, когда ассемблирование происходит прямо на дискету, т.к. это дает максимальный размер для вашего буфера объектного кода; рабочие адреса вашего кода не будут поражены.

2 — вы должны избегать использования более одной директивы 'ORG' в вашей программе, если только вы не введете нули в память между двумя этими директивами с помощью 'DEF'.

Например:

```
ORG 50000
; некоторые коды
RET
ORG 60000
; еще коды
```

Этот фрагмент не будет сохранен на ленте правильно, т.к вторая директива 'ORG' переопределяет начало буфера объектного кода. Но:

```
ORG 50000
; некоторые коды
RET
; добиваем нулями до 60000
DEFS 60000-$
; еще коды
```

Фрагмент будет сохранен правильно, т.к. DEF- произведет достаточное количество нулей, чтобы последующий код начинался с адреса 60000.

Очевидно, это неэффективно применительно к качеству кода, сохраняемого на дискете, но простота этого действия сохраняет ассемблер небольшим и быстрым.

Примеры использования команды 'A':

A20,,1:TEST <ENTER>

Ассемблирует с листингом, помещает объектный код сразу после таблицы символов (это увеличивает размер буфера объектного кода в памяти), использует размер таблицы символов по умолчанию и сохраняет объектный код на дисководе 1 под именем 'TEST'.

A,3000 <ENTER>

Ассемблирует программу, используя ключи по умолчанию, с размером таблицы символов 3000 байт (см. раздел 2 для дальнейшей детализации того, что может случиться при ассемблировании).

Команда 'R'

Если исходная программа была проассемблирована без ошибок и рабочие адреса были определены с помощью директив 'ENT', то может быть использована команда R для выполнения объектной программы. Объектная программа может использовать инструкцию RET (C9) для возврата в редактор сколько угодно раз, если стек в конце выполнения программы будет таким же, каким он был в ее начале.

Заметим, что директива 'ENT' не будет действовать, если для ассемблирования использовался ключ 16.

Прерывания перед вводом кода разрешены и регистр IY загружен значениями 5C3A, важным для подпрограммы обслуживания прерываний SPECTRUM ROM.

3.2.6. Другие команды

Команда 'B'

Просто возвращает управление операционной системе. Для перезапуска ассемблера используйте

RANDOMIZE USR XXXXX,

где: XXXXX — адрес, с которого был загружен GENS4.

Команда 'C'

Позволяет вам изменить размер входного буфера и буфера макроопределений (только для версий DEVPAC на магнитной ленте).

Входной буфер — это буфер, в котором содержится текст, когда происходит трансляция непосредственно с кассеты или дискеты — чем больше этот буфер, тем больший текст может быть считан с кассеты или дискеты и, следовательно, тем быстрее будет происходить трансляция. Но, с другой стороны, используется большая память. Однако возможен компромисс между скоростью трансляции и используемой памятью, команда 'C' позволит вам управлять этим процессом, предоставляя вам возможность установки размера входного буфера.

Буфер макроопределений используется для хранения текста макроопределений, которые вы могли использовать.

Команда 'C' печатает подсказку на ввод размера входного буфера, а затем — на ввод размера буфера макроопределений. В обоих случаях вводить надо просто число десятичное байтов, которое вы желаете зарезервировать, и <ENTER>. Если вы нажали <ENTER> без ввода числа, то операция игнорируется. Определяемый вами размер входного буфера не должен быть менее 256 байт. Вы можете прервать команду с помощью <CAPS SHIFT> 1.

Отметим, что для дисковых версий DEVPAC 4, вы можете менять только размер буфера макроопределений.

Команда 'C' не уничтожает ваш текст, она просто сдвигает его вверх и вниз в памяти в зависимости от размера буфера. Лучше зарезервировать буферы такого размера, который будет нужен в начале сеанса.

Команда ' S,,D '

Эта команда позволяет вам изменить символ-разделитель аргументов в командной строке. На входе редактора таким разделителем является запятая, это может быть изменено с помощью команды 'S' на первый символ определяющей строки 'D'.

Помните, что однажды определив новый разделитель, вы должны его использовать до тех пор, пока не определите новый — даже внутри команды 'S'. Заметьте, что разделителем не может быть пробел.

Команда ' U N '

Позволяет установить верхнюю границу памяти равной N. Если N не указать (т.е. ввести только U и <ENTER>), то отображается текущая верхняя граница памяти.

GENS4 не позволяет вашему текстовому файлу или объектному коду распространяться выше верхней границы памяти и будет выдавать сообщение об ошибке при приближении к этой границе.

По умолчанию верхняя граница памяти принимается равной вершине стека памяти системы SPECTRUM.

Команда ' V '

Выдает на дисплей полезную информацию: значение параметров команды N1 и N2 по умолчанию; символ-разделитель команды по умолчанию; десятичное значение начала и конца текста и значение первой командной строки S1.

Команда ' W N,M '

Выводит строки текста с N по M включительно на принтер. Если ни N, ни M не указаны, то будет напечатан весь файл. Печать будет приостановлена после вывода некоторого числа линий, определенного командой 'K' — нажмите любую клавишу для продолжения печати.

Команда ' X N '

Выдает каталог диска. В версии с 51-м символом перед выдачей каталога происходит очистка экрана. Каталог всегда распечатывается в строку из 32-х позиций.

Команда ' Z '

Эффективно уничтожает весь ваш текст, но перед этим она спрашивает, уверены ли вы в необходимости этого. Отвечайте 'Y' для уничтожения текста.

Кроме быстрого уничтожения командой D1,32767 команда 'Z' позволяет вам очистить ваш текстовый файл, если он каким-либо образом был испорчен.

Команда 'Z' устраняет необходимость для стартовой точки входа GENS4.

Команда ' H '

Выдает подсказку на экран: список возможных команд в виде столбца с заглавной буквой, обозначающий команду и отображающий текущее значение определенных важных параметров.

3.3. Пример использования редактора

Предположим, что вы ввели следующую программу (используя I10,10):

```
10   *H 16 BIT RANDOM NUMBERS
20
30   ;INPUT: HL CONTAINS PREVIOUS RANDOM
NUMBER
40   ;OUTPUT: HL CONTAINS NEW RANDON OR
SEED NUMBER
50
60   RANDOM PUSH AF ; SAVE REGISTERS
70   PUSH BC
80   PUSH HL
90   ADD HL,HL ; *2
100  ADD HL,HL ; *4
.....
.....
.....
140  ADD HL,HL ; *64
150  PIP BC ; OLD RANDOM NUMBER
160  ADD HL,DE
170  LD DE,L1
180  ADD HL,DE
190  POP BC ; RESTORE REGISTERS
200  POP AF
210  REY
```

Эта программа содержит несколько ошибок:

строка 40: вместо 'RANDOM' набрано 'RANDON';

строка 70: PUSH BC начинается с поля метки;

строка 150: PIP вместо POP;

строка 160: требуется комментарий (это не ошибка, а недостаток);

строка 210: вместо REY должно быть RET.

Также должны быть добавлены две команды ADD HL,HL между строками 140 и 150. Кроме того, все ссылки на пару регистров DE в строках 160... 180 должны быть ссылками на BC.

Для внесения этих исправлений мы должны выполнить следующее:

```
F40,40, RANDON, RANDOM <ENTER>
```

затем подкоманду 'S'

E70 <ENTER> затем I (режим вставки), один пробел и

```
<ENTER><ENTER>
I142, 2<ENTER> 142 ADD HL,HL ; *128
144 ADD HL,HL ; *256
<EDIT>
F150, 150,PIP,POP <ENTER>
```

затем подкоманду 'S'

```
E160 <ENTER> затем X, 2 пробела;*257 +L1
<ENTER><ENTER>
E160,180,DE,BC <ENTER>
```

затем повторное использование подкоманды 'S' E210 <ENTER> затем 2 пробела, C, T <ENTER><ENTER>

```
N10,10 <ENTER>,
```

чтобы перенумеровать текст.

Рекомендуем вам хорошенько проработать вышеприведенный пример, используя редактор.

Приложение 1

Коды ошибок и их значение.

*ERROR*1 — ошибка в константе этой строки;

*ERROR*2 — мнемоника не распознана;

*ERROR*3 — утверждение плохо сформировано;

*ERROR*4 — символ определен более 1 раза;

*ERROR*5 — строка содержит неверный символ (т.е. ничего не значащий в данном случае);

*ERROR*6 — один из операндов в строке — незаконный;

*ERROR*7 — символ в строке является резервной инструкцией;

*ERROR*8 — ошибочная пара регистров;

*ERROR*9 — слишком много регистров в строке;

*ERROR*10 — выражение, которое должно занимать не более 8 бит, занимает больше;

*ERROR*11 — неверные инструкции JP(IX+N) и JP(IY+N);

*ERROR*12 — ошибка в директиве ассемблера;

*ERROR*13 — незаконная ссылка вперед, т.е. EQU ссылается на символ, который еще не был определен;

*ERROR*14 — деление на ноль;

*ERROR*15 — переполнение в операции умножения;

*ERROR*16 — вложенное макроопределение;

*ERROR*17 — этот идентификатор — не макро;

*ERROR*18 — вложенный макровызов;

*ERROR*19 — вложенный условный оператор;

BAD ORG! — директива ORG работает с адресом, который может испортить GENS4, текстовый файл или таблицу символов. Управление возвращается в редактор.

OUT OF TABLE SPACE — появляется во время первого прохода, если на таблицу символов выделено недостаточно памяти. Управление возвращается редактору.

BAD MEMORY! — нет места для ввода текста, т.е. конец текста близок к вершине ОЗУ памяти. Вы должны спасти текущий текстовый файл или его часть.

Приложение 2

Резервирование слов, мнемоника и т.д.

Список резервных инструкций внутри GENS4. Эти символы не могут использоваться как метки, хотя они могут быть частью меток. Все резервные инструкции состоят из заглавных букв:

A	B	C	D	E	H	L	I	R	S
AF	AF'	BC	DE	HL	IX				
IY	SP	NC	Z	NZ	M				
P	PE	PO							

Ниже приводится список мнемоники Z80, директив ассемблера и его команд. Они также должны состоять из заглавных букв.

ADC	ADD	AND	BIT	CALL
CCF	CP	CPD	CPDR	SPI
CPIR	CPL	DAA	DEC	DI
DJNZ	EI	EX	EXX	HALT
IM	IN	INC	IND	INDR
INI	INIR	JP	JR	LD
LDD	LDDR	LDI	LDIR	NEG
NOP	OR	OUTDR	OTIR	OUT
OUTD	OUTI	POP	PUSH	RES
RET	RETI	RETN	PL	RRA
RLC	RLCA	RLD	RR	RLA
RRC	RRCA	RRD	RST	SBC
SCF	SET	SLA	SRA	SRL
DEFB	DEFM	DEFS	DEFW	ELSE
END	ENT	EQU	IF	ORG
MAC	ENDM			
*D	*E	*H	*L	*S
*C	*F	*M		

Приложение 3

Рабочий пример

Ниже приводится пример типичного сеанса работы с использованием GENS4. Если вы новичок по части использования ассемблерных программ или вы немного не уверены в том, как использовать ассемблер/редактор GENS4, то мы настоятельно советуем вам тщательно проработать этот пример. Лишний раз заметим, что <ENTER> означает, что вы должны нажать клавишу 'ENTER' на клавиатуре.

Цель примера:

Написать и проверить программу целочисленного умножения, текст которой должен быть сохранен на магнитной ленте, используя команду редактора 'T' так, чтобы легко было включать ее в состав будущих программ с магнитной ленты.

Последовательность работы:

1 — напишите программу умножения как подпрограмму и запишите ее на ленту, используя команду редактора 'P', чтобы ее можно было легко восстановить во время сеанса при наличии ошибок.

2 — отладьте подпрограмму умножения, редактируя ее, если это необходимо.

3 — запишите отлаженную программу на ленту, используя команду 'T' так, чтобы эта подпрограмма могла быть помещена с ленты в другие программы. Перед стартом мы должны загрузить GENS4 в компьютер — делайте так:

```
LOAD '' '' CODE 26000 <ENTER>
```

чтобы загрузить ассемблер с адреса 26000. Затем введите:

```
RANDOMIZE USR 26000 <ENTER>
```

Сейчас вы находитесь в режиме редактирования, т.е. компьютер готов для создания программ на ассемблере. 1-я стадия: запись программы целочисленного умножения.

Мы используем команду редактора I, чтобы ввести текст, используя < —> > (символ табуляции) для получения табулированного листинга.

Ниже мы не указываем, где используется табуляция. Но вы можете предположить, что табуляция используется между мнемоникой команды и между мнемоникой и операндом. Заметим, что адреса, показанные в ассемблерном листинге примера, который следует, могут не соответствовать полученным в вашем компьютере, они служат только для иллюстрации:

```
>I10,10 <ENTER>
10  ;A FAST INTEGER MULTIPLY <ENTER>
20  ;ROUTINE>MULTIPLIES HL <ENTER>
30  ;BY DE.RETURN THE RESULT <ENTER>
40  ;IN HL.C FLAG SET ON AN <ENTER>
50  ;OVERFLOW. <ENTER>
60  ;<ENTER>
70              ORG #7F00 <ENTER>
80  <ENTER>
90      MULT    OR A <ENTER>
100             SBC HL,DE;HL>DE? <ENTER>
110             ADD HL,DE <ENTER>
120             JR NC,MU1;YES <ENTER>
130             EX DE,HL <ENTER>
140     MU1     OR D <ENTER>
150             SCF;OVERFLOW IT <ENTER>
160             RET NZ;DE>255 <ENTER>
170             OR E;TIMES 0? <ENTER>
180             LD E,D <ENTER>
190             JR NZ,MV4;NO <ENTER>
200             EX DE,HL;0 <ENTER>
210             RET <ENTER>
220 <ENTER>
230 ;MAIN ROUTINE <ENTER>
240 <ENTER>
250     MU2     EX DE,HL <ENTER>
260             ADD HL,DE <ENTER>
270             EX DE,HL <ENTER>
280     MU3     ADD HL,HL <ENTER>
290             RET C;OVERFLOW <ENTER>
300     MU4     RRA <ENTER>
310             JR NC,MU3 <ENTER>
320             OR A <ENTER>
330             JR NZ,MU2 <ENTER>
340             ADD HL,DE <ENTER>
350             RET <ENTER>
360             <EDIT>
```

Вышеприведенные команды создадут текст программы. Сохраним ее на ленте, используя:

```
>P10,350,MULT <ENTER>
```

Помните, что вы должны включить свой магнитофон в режим записи перед использованием команды 'P'. 2-я стадия: отладка программы.

Сначала убедитесь в правильном ассемблировании исходного текста. Мы будем использовать ключ 2, чтобы не производить листинг и объектный код.

```
>A2 <ENTER>
*HISOFT GENS4 ASSEMBLER*
COPYRIGHT HISOFT 1983, 84, 87
ALL RIGHT RESERVED
PASS1 ERROR:00
PASS2 ERROR:00
*VWARNING* MV4 ABSENT TABLE USED:74 FROM
161
>
```

Мы видим из этой трансляции, что сделали ошибку в строке 190, набрав MV4 вместо MU4, которая является меткой, на которую мы желаем перейти. Редактируем строку 190:

```
>F190,190,MV4,MU4 <ENTER>
190 JR NZ, (сейчас используйте подкоманду 'S')
```

Теперь ассемблируйте текст вновь. Он должен ассемблироваться без ошибок. Мы должны написать некоторый текст для проверки подпрограммы:

```
>N300,10 <ENTER> (перенумеруем для того, чтобы мы
могли написать больше текста)
>I10,10 <ENTER>
10 ;SOME CODE TO TEST <ENTER>
20 ;THE MULT ROUTINE <ENTER>
30 <ENTER>
40              LD HL,50 <ENTER>
50              LD DE,20 <NETER>
60              CALL MULT;MULTIPLY <ENTER>
70              LD A,H;O/P RESULT <ENTER>
80              CALL AOUT <ENTER>
90              LD A,L <ENTER>
100             CALL AOUT <ENTER>
110             RET; RETURN TO EDITOR <ENTER>
120 <ENTER>
130 ;ROUTINE TO O/P A IN HEX <ENTER>
140 <ENTER>
150     AOUT    PUSH AF <ENTER>
160             RRCA <ENTER>
170             RRCA <ENTER>
180             RRCA <ENTER>
190             RECA <ENTER>
200             CALL NIBBLE <ENTER>
210             POP AF <ENTER>
220     NIBBLEAND %1111 <ENTER>
230             ADD A,#90 <ENTER>
240             DAA <ENTER>
250             ADC A,#40 <ENTER>
260             DAA <ENTER>
270             LD IY,5C3A;FOR ROM <ENTER>
280             RST #10;ROM CALL <ENTER>
290             RET <ENTER>
300             <ENTER>
>
```

Теперь ассемблируйте текстовую программу совместно программой MULT:

```
>A2 <ENTER> *HISOFT GENS4 ASSEMBLER*
COPYRING HISOFT 1983,84,87
ALL RIGHT RESERVED
7EAC 190 RECA
*ERROR* 02 (нажмите любую клавишу для продолжения)
PASS 1 ERROR: 01
TABLE USED: 88 FROM 210
```

В нашей программе есть ошибка: вместо RECA в строке 190 должно быть RRCA. Делаем:

```
>E190
190 RECA
190 —>, один пробел, C,R <ENTER>>, <ENTER>
>
```

Теперь проассемблируйте вновь, используя ключи по умолчанию (только A <ENTER>), и текст должен проассемблироваться правильно.

Мы уже в состоянии проверить работу нашей программы MULT, следовательно, нам нужно сообщить редактору, откуда он может выполнить код. Мы делаем это директивой 'ENT': >35 ENT S <ENTER>

Теперь вновь проассемблируем текст, и трансляция должна завершиться правильно с сообщениями:

```
TABLE USED: 88 FROM 211
EXECUTES: 32416
>
```

или подобными этим. Теперь можно запустить наш код на выполнение, используя команду редактора 'R'. Помножив 50 на 20, мы должны получить 1000, что равно 3E8 в шестнадцатиричном коде:

```
>R <ENTER>
0032>
```

Не работает! Почему не работает? Распечатайте строки 380 по 500 (L380, 500). Вы увидите, что в строке 430 находится инструкция, OR D, за которой следует RET NZ. Все, что она делает — это логическую операцию 'или' между регистром D и аккумулятором A и возвращает установленный флаг ошибки (C-флаг), если результат не равен 0. Целью этого действия является возможность убедиться,

что DE<256 и, следовательно, что операция умножения не вызвала переполнения — это делается проверкой того, что в регистре D находится 0. Но 'или' будет работать правильно только в том случае, когда в аккумуляторе изначально находится 0, а у нас нет гарантий, что это будет так.

Мы должны убедиться, что A=0 перед выполнением OR D, в противном случае мы получим переполнение со старшими разрядами в качестве результата. Для проверки кода OR A в строке 380 можно изменить на XOR A, которая установит флаг для SBC HL,DE инструкции и установит A=0. Следовательно:

```
>E380 <ENTER>
380 MULT OR A <ENTER>
380 —> I (вход в режим вставки) X <ENTER><ENTER>
>
```

Теперь вновь проассемблируйте и запустите код, используя R. Теперь ответ должен быть правильным: 3E8. Мы можем проверять программу дальше, отредактировав строки 40 и 50, чтобы перемножать различные числа. Ассемблируя и запуская, вы увидите, что программа работает. Теперь, закончив программу, мы можем сохранить ее на магнитной ленте в формате 'INCLUD':

```
>T300,999,MULT <ENTER>
```

Не забудьте включить магнитофон в режим записи перед нажатием <ENTER>.

Если вы хотите сохранить программу на дискете, то не нужно использовать программу 'T'. Программа, сохраненная обычной командой 'P', может быть запущена с дисковода.

Однажды запущенная подобным образом подпрограмма может быть вставлена в программу как показано ниже:

```
500 RET
510
520 ;INCLUD THE MULT ROUTINE HERE
(включаем здесь MULT)
530
540 *F MULT
550
560 ;THE NEXT ROUTINE (следующая подпрограмма)
```

Когда вышеприведенный текст будет транслироваться, ассемблер спросит вас 'START TAPE... ' когда он доберется до строки 540 во время как первого, так и второго проходов. Следовательно, вы должны направить MULT на вывод с ленты в обоих случаях. Для этого обычно нужно будет перемотать ленту после первого прохода. Вы можете записать два дубля программы MULT на ленту один за другим, и один использовать во время первого прохода, а второй — во время второго.

Когда происходит ввод с дисковода, никаких сообщений не появляется, все работает автоматически.

Пожалуйста, изучите тщательно вышеприведенный пример и постарайтесь выполнить его самостоятельно.

Глава 5. Работа с магнитофоном

5.1. COPY-COPY — копировщик программ

Руководство пользователя

CAT — клавиша 'C' — просмотр содержимого магнитной ленты.

LOAD — клавиша 'J' — загрузка файлов в память:

LOAD X — загрузить файл в память на место X-го файла. Если X=1, то загруженные перед этим файлы теряются и загрузка производится в начало рабочей области (с адреса 23296);

LOAD X TO XX — загружаются файлы с номерами от X до XX;

LOAD AT XX — загружать файлы с адреса XX. По умолчанию файл с номером 1 загружается по адресу 23296. Можно задать XX=23040, в этом случае величина рабочей области для загрузки файлов увеличивается до 42496 байт. По умолчанию величина этой области равна 42240 байт. (очевидно, XX < 23040 задавать нельзя);

LOAD (XX — считывание первых XX байтов файла. Пример:

LOAD (6912 — считывание только экранной области.

SAVE — клавиша 'S' — сохранение загруженных файлов на м/д.

SAVE — сохранение всех загруженных файлов без пауз;

SAVE X — сохранить файлы, начиная с номера X;

SAVE X TO XX — сохранить файлы с номерами от X до XX;

SAVE TO XX — сохранить файлы с номерами от 1 до XX

SAVE STEP X — сохранение всех загруженных файлов. Между файлами делать паузы X секунд. Если X=9, то после каждого файла выдавать запрос 'PRESS ANY KEY';

SAVE X TO XX STEP XXX — сохранение файлов с номерами от X до XX с паузами между файлами XXX секунд.

VERIFY — клавиша 'V' — проверка сохраненных файлов:

VERIFY — аналогично SAVE;

VERIFY X TO XX — аналогично SAVE X TO XX;

VERIFY X — аналогично LOAD X.

LET — клавиша 'I' — изменение полей заголовка файла. Например:

LET 2=AAA,, 1 — файл с номером 2 будет иметь имя AAA, стартовый адрес 1;

LET 3=500, 1, 5— файл с номером 3 будет иметь длину 500 байт, стартовый адрес 1, длина программы 5 байт.

LIST — клавиша 'K' — распечатка памяти:

LIST XX — XX задает адрес памяти. Если адрес не задан, то он равен 0. По этой команде выводится 15 байт памяти, для которых показывается: адрес памяти, десятичное значение байта, десятичное значение двух смежных байт памяти и символьное значение байта. Для вывода следующих 15 байт нажмите 'ENTER'.

POKE — клавиша 'O' — изменение десятичного значения байта:

POKE X, AA — X — адрес, AA — десятичное значение байта. Если AA лежит в диапазоне 256 — 65535, то считается, что задано значение двух смежных байт.

USR — клавиша 'U' — вызвать программу пользователя:

USR X — вызвать программу пользователя по адресу X. можно использовать эту функцию для полного сброса системы, задав X=0.

RETURN — клавиша 'Y' — возврат в BASIC, инициализируются системные переменные и таблица каналов, полный сброс не выполняется.

COPY — клавиша 'Z' — копирование больших программ:

COPY — перевод программы в специальный режим копирования файла без заголовка длиной до 49096 байт. После выдачи команды программа загружает файл в память, а затем, по нажатию клавиши 'CAPS SHIFT', выгружает ее необходимое число раз. Повторная загрузка возможна только если остается не менее 200 байт незанятой памяти. После записи необходимого количества копий компьютер необходимо сбрасывать отключением от сети;

COPY NN — копирование файлов длиной до 49152 байт. Копирование выполняется только один раз.

Условные обозначения типов файлов:

P — программа;

B — машинные коды (BYTES);

A — числовой массив;

$ — символьный массив;

— числовой массив.

'ENTER' — повторяет предыдущую команду

'CAPS SHIFT + 0' — уничтожает набранную строку.

5.2. TFCOPY — копировщик программ

Программа TFCOPY (TAPE FILE COPY) предназначена для копирования информации на магнитную ленту. В процессе загрузки программы (по команде LOAD ''TFCOPY'') на экране появляется промежуточное сообщение:

```
TAPE FILE COPY — 11/86
OPTIONS:    LOAD
(функций)   SAVE
            VERIFY
            DELETE
            CLOCK
            MODE
            RENAME
            CURSOR KEYS
PARAMS:     NUMBER
(параметры) ALL
            BEGIN
            END
            PROGRAM (BASIC +...)
            FILE (HEADER + DATA)

— COMPRESSED FORM IN RAM —
(сжатое хранение данных в памяти)

© ARNOST VECERKA — OLOMOUC.F
```

После окончательной загрузки программы появляется стартовое меню (звездочкой отмечены мерцающие цифры — режим по умолчанию)

```
    0 — START                 — старт
    1 — CLOCK — SET           — вкл. Часов
*   2 — MODE — SELECT         — выбор режима
    3 — RENAME                — переименование
    4 — 41984 BYTES — 14 LINES
*   5 — 44032 BYTES — 6 LINES
    6 — 44288 BYTES — ATTR USED
```

До нажатия на клавишу 0 (START) можно изменить режим работы программы нажатием соответствующих клавиш (1, 3, 4 или 6). Клавиши 4, 5 и 6 выбирают режим вывода на экран каталога загруженных в память программ (6 или 14 строк), при этом изменяется объем памяти, предназначенный для загрузки программ (при выводе большего числа строк меньше объем памяти). По умолчанию выводится 6 строк каталога при 44032 байт свободной памяти.

После нажатия на клавишу 0 (START) в верхней части экрана высвечивается объем свободной памяти и команды, которые может выполнять программа:

LOAD (клавиша L) — загрузить файл(ы) в память. После нажатия клавиши L необходимо пустить магнитофон. На экране отображается номер загружаемого файла, количество свободной памяти после загрузки очередного файла и каталог загруженных в память программ (имя файла, тип информации, объем и адрес загрузки). Процесс загрузки можно прервать в любой момент одновременным нажатием клавиш CAPS SHIFT и BREAK SPACE. Если в память загружено больше файлов, чем может отображаться на экране, то для просмотра всего каталога следует пользоваться клавишами сдвига курсора вниз и вверх.

SAVE (клавиша S) — выгрузить файл(ы). После нажатия клавиши S необходимо уточнить с какого (FROM) по какой (TO) файлы необходимо выгрузить. Номер надо задавать 2-х значный или оканчивать ввод нажатием <ENTER>. Перед окончательным ответом следует включить магнитофон на запись.

DELETE (клавиша D) — удалить файл из каталога. После нажатия клавиши D необходимо уточнить номера удаляемых файлов.

VERIFY (клавиша V) — сравнить файл, находящийся в памяти и записанный на магнитную ленту. Перемотать ленту на начало сравниваемого файла. После нажатия клавиши V и уточнения номера пустить магнитофон на воспроизведение.

MODE (клавиша M) — модификация режима вывода программы с очисткой памяти. После нажатия на клавишу M появляется надпись:

CLEAR 1-2-3

Последующее нажатие клавиши 1 переводит программу в режим вывода 14 строк каталога (41984 байт памяти свободно). Клавиша 2 — 6 строк (44032 байт памяти свободно). Клавиша 3 — выводится 6 строк каталога (44288 байт памяти свободно), при этом

в нижней части экрана отображается состояние экранной памяти, занимаемой загружаемой программой.

RENAME (клавиша R) — переименование программ. После нажатия клавиши необходимо ввести номер файла, который требуется переименовать. Выведется старое имя, после этого следует набрать новое имя (до 10 символов).

CLOCK (клавиша C) — установка параметров ввода информации с магнитофона. После нажатия на клавишу C выведется сообщение:

WAIT:456 SAMPLE:2400 CURSN

Клавишами управления курсором необходимо установить нужные характеристики.

ALL (клавиша A) — выдает команду LOAD, SAVE, DELETE, VERIFY на все файлы, введенные в ОЗУ.

Сообщения:

При ошибках ввода с ленты появляется сообщение:

ACCEPT PARITY BYT = XXXXX

Необходимо повторить ввод.

При прекращении ввода нажатием CAPS SHIFT / BREAK SPACE появляется сообщение:

ACCEPT BREAK BYT = XXXXX

5.3. COPY 86M

LOAD (клавиша 'L') — загрузка файлов в память с очередным номером. Исходной свободной памяти — 45 000 байт.

Внимание: в случае ошибки считывания с магнитной ленты в левом углу появляется вместо LOAD надпись TAPE ERROR и знак вопроса после файла, введенного с ошибкой. Нажать клавишу 'BREAK', после чего клавишу 'L' и продолжать ввод файлов дальше.

Пример:

TAPE ERROR	FREE: 38541	TIME 1:18		
P	DAN DARE	20	164	164
C		65235	168	168
H		28723	534	534
H		NOT FOUND	41000	41000

Где: P — программа (BASIC)

C — машинные коды (BYTES)

H — неименованный файл

Внимание: в случае появления надписи 'OUT OF MEMORY', необходимо файл, который не умещается в ОЗУ, загрузить вновь (перед этим записать предыдущие файлы на ленту и удалить их).

COPY (клавиша 'C') — перевод программы в специальный режим копирования. Против каждого копируемого файла появляется буква 'C'. После нажатия 'ENTER' появляется надпись 'START TAPE...', после второго нажатия 'ENTER' — надпись 'COPY' и программа переходит в режим копирования.

COPY/PAUSE (клавиша 'M') — режим копирования файлов с паузой в 6 секунд между файлами. В конце каждого копируемого файла появляется надпись 'CP'. Используется в том случае, если необходимо сделать паузу между программами. Например:

COPY	FREE: 991	TIME 7:35			
P	GREEN BERET	2	175	175	C
C		30000	6912	6912	C
C			41000	41000	C
P	HARD GUY	2	191	191	CP
C		28000	6912	6912	C

VERIFY (клавиша 'V') — проверка скопированных файлов. После каждого проверенного файла появляется буква 'V'.

Клавиша 'X' — отмена ранее заданной операции с файлами.

Клавиша 'S' — просмотр загруженных файлов.

ALL (клавиша 'A') — применяется при операциях со всеми загруженными файлами. Например:

ALL ('A') — COPY ('C') — копирование всех файлов.

DELETE (клавиша 'D') — удаление ранее загруженных файлов. Напротив файла, подлежащего удалению, ставится буква 'D', а поверх букв — пунктирная линия.

Адресация (клавиша 'H') — триггерная команда. При нажатии вместо десятичного отображения свободной памяти появляется шестнадцатиричное, по второму нажатию — наоборот. При первом нажатии все загруженные файлы будут иметь указанные адреса, символьное значение, десятичное значение, обозначенное в шестнадцатиричном коде.

PRINT (клавиша 'B') — распечатка загруженных файлов (BASIC)

'ENTER' — перевод в рабочий режим команд копировщика.

'BREAK' — переход из одного режима в другой.

5.4. MR. COPY

После загрузки программы на экране появляется мистер Копи, который приветствует вас: 'Хелло! Я мистер Копи'. На доске написано: 'Мистер Копи версия 1. 1983. Автор Ван-Сондерен. Дублирующие программы могут быть нарушением закона об авторском праве'.

Чтобы начать работу с программой нажмите клавишу '0'. Мистер Копи говорит: 'Пожалуйста выберите из меню'. На доске предоставлено 'Главное меню':

1 — загрузка оригинального файла;

2 — запись копии на кассету;

3 — верификация записанной копии;

4 — чтение заголовков файлов;

5 — загрузка и запись копии файлов без заголовков;

6 — сброс программы и выход в бейсик.

Выбор из меню производится нажатием соответствующей клавиши. Итак, по порядку:

1. М-р Копи: 'Сколько блоков данных будете вводить? (1-10)'.

На доске вверху появляется надпись 'LOADING' — загрузка. Для ввода 10 блоков нажмите '0'.

М-р Копи: 'Нажмите PLEY' (в программе допущена ошибка, вместо 'PLEY' м-р Копи советует нажать 'REG')

Блоки: (количество оставшихся незагруженных блоков).

На доске появляется заголовок файла: тип файла (PROGRAM, DATA ARRAY, BYTE, CHARAKTER ARRAY — программа в бейсике, числовой массив, машинные коды, текстовый массив).

Название файла (не более 10 символов).

Начальный адрес в программе или памяти.

Длина файла в байтах.

Водится в память программы и ожидается ввод следующего блока данных.

2. М-р Копи: 'Нажмите REG и PLEY, затем '0''.

На доске вверху появляется надпись 'SAVING' — запись. По нажатию '0' происходит запись всех загруженных блоков последовательно, в том порядке в котором они были загружены с выдержкой стандартных пауз между блоками. По окончании записи дается короткий звуковой сигнал и программа возвращается к главному меню.

4. М-р Копи: 'Нажмите PLEY блоки '...''.

На доске вверху 'READING' — чтение. Программа ждет ввода заголовков файлов. На доске появляются заголовки файлов.

5. М-р Копи: 'нажмите PLEY'.

На доске вверху 'HEADERLESS' — без заголовка; программа ждет ввода файла без заголовка. Как только файл будет загружен в память программы м-р Копи говорит: 'Нажмите REG и PLEY, затем '0'. По нажатию '0' происходит запись файла без заголовка на ленту. После записи программа возвращается к главному меню. Файл без заголовка можно проверить в режиме 3.

8. М-р Копи: 'Вы уверены? тогда нажмите '6'. По нажатию '6' происходит сброс программы, аналогично нажатию кнопки RESET.

При нажатии BREAK в любом режиме м-р Копи: 'Нажата клавиша BREAK — нажмите '0''. По нажатию '0' программа возвращается к главному меню.

5.5. TEST.PROG — программа проверки работоспособности

Программа написана Яном Логаном — одним из разработчиков компьютера 'ZX SPECTRUM'. Загрузка теста производится командой:

LOAD 'TEST$PROG'. По окончании загрузки на экран выводится следующее меню:

KEYBOARD SPECTRUM 16/48K	1
KEYBOARD SPECTRUM+	2
STOP KEY (SPECTRUM+)	3
COLOUR/FLASH	4
SOUND	5
ULA	6
RAM/ROM	7
CASSETE	8
ALL TESTS SPECTRUM 16/48K	9
ALL TESTS SPECTRUM+	0

Для всех тестов сообщение: 'TEST PASSED', выводимое после его завершения, говорит о правильном выполнении теста, Сообщение: 'TEST FAILED' — о том, что обнаружена неисправность проверяемого устройства. Для повтора теста нажмите клавиши SYMBOL SHIFT/BREAK.

Проверка клавиатуры.

Нажмите клавишу 1 в основном меню. На экране появится изображение клавиатуры. Последовательно нажимайте клавиши в порядке их следования на экране. Если ввод был воспринят правильно, символ введенной клавиши заменится красным прямоугольником.

Примечание: клавиши 'ENTER', 'CAPS SHIFT', 'SYMBOL SHIFT', 'пробел' обозначаются как малые буквы 'e', 'c', 's' и 'b' соответственно.

Проверка формирования цвета.

Нажмите клавишу '4'. на экран будут выведены 8 цветных полос в следующем порядке:

— черный (BLACK)
— голубой (BLUE)
— красный (RED)
— фиолетовый (MAGNETE)
— зеленый (GREEN)
— синий (CYAN)
— желтый (YELLOW)
— белый (WHITE)

Левая часть каждой полосы — нормальной яркости, правая — повышенной. Через некоторое время появится вопрос: 'IS THIS CORRECT <Y?N>' ('Все правильно?'), введите 'Y', если цвета соответствуют их названиям, иначе 'N'.

Проверка работы звукового сопровождения.

Клавиша '5'. На экран выводится заголовок теста ('SOUND TEST') и вы должны услышать звук одного тона и длительностью примерно 10 сек, после чего задается вопрос: 'DID YOU HEAR THE TONE <Y/N>?' ('Вы слышали звук <да/нет?')

Проверка видеопроцессора.

Клавиша '6'. На экран выводится надпись 'ULA TEST', по краям экрана — чередование черных полос, сопровождаемое звуком. По окончании проверки выводится сообщение о ее результате.

Проверка памяти (ПЗУ и ОЗУ).

Клавиша '7'. Экран заполняется меняющимся цветным узором. Через некоторое время появляется сообщение о результате: 'THIS SPECTRUM HAS 48 K OF MEMORY IN WORKING ORDER, IS THIS CORRECT <Y/N>?' ('этот компьютер имеет 48 К ОЗУ, правильно?').

Проверка работы с магнитофоном.

Клавиша '8'. Тест состоит из двух этапов: проверка вывода и проверка ввода. Для проверки вывода включите магнитофон на запись, и нажмите клавишу 'SYMBOL SHIFT' в ответ на запрос: 'PRESS THE KEY TO MAKE A TONE', после чего на магнитофон будет произведена тестовая запись (тон). Отметьте, что при правильной работе компьютера операции с магнитофоном сопровождаются появлением перемещающихся по краям экрана красных полос и звуком. По окончании выдачи тона, остановите магнитофон, перемотайте назад ленту и, после появления на экране надписи 'PRESS PLAY NOW', нажмите воспроизведение. Если тест закончится правильно, будет выведено сообщение: 'LOADING O'K'.

Цепочка тестов.

Клавиша '9'. Выполнение всех вышеописанных тестов один за другим (без промежуточного выхода в главное меню).

6.1. TRDOS

6.1.1. Вступление

Эта дисковая система представляет собой профессиональный метод хранения программ и файлов данных в домашних и персональных компьютерных системах.

Дисковые системы имеют большие преимущества перед системами на лентах, такими как системы на кассетных лентах и микронакопители. на дисках 5,25'', 3,5'', 3''. Их называют гибкими дисками, дискетами, мини-дисками или микро-дисками. Мы будем называть их просто диски. В настоящее время наиболее используемый вид дисков — 3,5''.

Процесс форматирования дает возможность TRDOS и дисководу электронно разделять диск на 40 или 80 дорожек (в соответствии с дисководом), а каждую дорожку — на секторы. Количество секторов на дорожку и количество байтов на сектор полностью зависят от операционной системы на дисках (DOS).

TRDOS обеспечивает 16 секторов на дорожку и 256 байтов на сектор. Такое большое количество небольших секторов имеет несколько преимуществ.

Во-первых, если хранению подлежит только небольшое количество данных, использованию подлежит не слишком большая часть диска. Это ведет к экономии большого количества файлов.

Во-вторых, при использовании файлов с произвольным доступом обеспечивается большая гибкость программы и увеличивается скорость работы.

Чтобы знать, что где расположено, TRDOS использует дорожку 0. В основном для своих собственных целей поиска.

При подключении к сети или после сброса (предполагая, что системный переключатель в обычном положении) вы автоматически переходите в режим TRDOS, и система всегда выбирает дисковод A. TRDOS делает попытку загрузить программу BASIC, называемую ''BOOT'' (самозагрузка). Если в дисководе ''A'' диск отсутствует или крышка дисковода открыта, или диск не содержит программы ''самозагрузки'' BASIC, то на экран выводится обычная подсказка.

Если диск находится в дисководе с закрытой крышкой, при подключении электроэнергии диск может быть испорчен, особенно если дисководы имеют постоянно загруженные (опущенные) головки.

TRDOS подгоняется так, чтобы соответствовать скорости шага дисковода. Это значит, что более быстрая скорость шага новейших дисководов находит полное применение. TRDOS также осуществляет проверку на соответствие технических условий дисковода на 40 или 80 дорожек, одно или двухстороннего.

6.1.2. Краткий перечень команд TRDOS

Команда	Функция
*''A:''	установка по умолчанию дисковода A
*''B:''	-''- дисковода B
*''C:''	-''- дисковода C
*''D:''	-''- дисковода D
40	информирует TRDOS, что дисковод 40-дорожечный
80	-''-, что дисковод 80-дорожечный
CAT	вывод на экран справочника диска
CAT#	распечатка справочника диска
CLOSE#	закрыть файл с последовательным произвольным доступом
COPY	копирование файлов с диска на диск
COPY S	копирование файла в системе с одним диском
COPY B	дублирование дисков в системе с одним диском.
ERASE	удалить файл с диска.
FORMAT	форматирование диска.
LIST	вывод содержания диска.
LIST#	распечатка содержания диска.
LOAD	загрузка программы с диска
INPUT#	чтение файла с последовательным произвольным доступом
MERGE	объединить программу BASIC с диска и программу BASIC в ОЗУ
MOVE	сортировать и упаковать вместе файлы на диск
NEW	изменить имя файла
OPEN#	открыть файл
PEEK	считать сектор с диска в ОЗУ
POKE	записать файл из ОЗУ сектор диска
PRINT#	распечатка файла
RANDOMIZE USR 15615	переход к TRDOS из SOS
RANDOMIZE USR 15619	вызов TRDOS-команд из SOS
RETURN	переход к SOS из TRDOS
RUN	загрузить и запустить программы с диска
SAVE	записать программу на диск
VERIFY	сравнить программу в ОЗУ с диском

TRDOS является гибкой системой, и вы можете осуществлять доступ к системе на дисках следующим образом: непосредственно с TRDOS; прямым доступом из SOS; из BASIC; из программ в машинных кодах.

Когда вы работаете с TRDOS, вы видите обозначение дисководов плюс стрелку. Это будет называется подсказка TRDOS, например:

```
A>
B>
```

В соответствии с только что законченной операцией, за подсказкой может следовать дальнейшая часть команды TRDOS, например:

```
A>RUN ''BOOT''
```

Команды TRDOS могут быть введены непосредственно после подсказки. Если вслед за подсказкой идет предыдущая команда, вы должны отменить старую команду с помощью клавиши DELETE.

6.1.3. Переход от TRDOS к SOS и от SOS к TRDOS

Для перехода на SOS используется команда RETURN (возврат).

В нижеследующем примере слова в [] представляют собой объяснение, а не часть синтаксиса.

```
A>                      [подсказка TRDOS на экране]
A> RETURN               [V нажато — теперь на экране]
(C) SINCLAIR COPYRIGHT
                        [нажато ENTER — теперь на экране]
```

Переход от SOS к TRDOS

Для перехода от SOS к TRDOS, при указателе ''K'', необходима следующая запись:

```
RANDOMIZE USR 15615
```

Если только один дисковод подогнан, это возвратит вас к дисководу ''A''. Однако, если подогнано более одного дисковода, то команда возвратит вас к последнему выбранному дисководу

6.1.4. Выбор дисковода

Дисковод по умолчанию — это тот дисковод, доступ к которому может осуществляется с помощью любой команды, в которой не указывается подлежащий использованию дисковод.

При подключении к сети или после сброса системы дисковод по умолчанию всегда ''A''.

TRDOS поддерживает до 4 дисководов, обозначенных как A,B,C или D. Для изменения дисковода по умолчанию формат команды следующий:

```
*''DRIVE:''
```

Выбор временного дисковода

В некоторых случаях желательно оставаться в состоянии умолчания на одном дисководе, но стремится осуществить доступ к другому дисководу. Синтаксисом для этого является суффикс() к команде, указывающей требуемый дисковод, суффикс выглядит как изменение умолчания, но без звездочки, например:

```
''A:'' или ''B:'' или ''C:'' или ''D:''
```

Ниже приводится пример полного оператора из SOS:

```
RANDOMIZE USR 15619:REM:LOAD''B:PROGRAM''
```

Это обеспечивает загрузку ''программы'' из дисковода ''B'' независимо от того, какой из дисководов является текущим дисководом по умолчанию, включая ''B''.

Из TRDOS с диском "A" по умолчанию команда будет следующей:

LOAD "B:PROGRAM"

Таким образом осуществляется загрузка "PROGRAM" из дисковода "B", но текущим остается дисковод "A" для дальнейших операций.

6.1.5. Форматирование диска

Диск должен быть отформатирован до использования его компьютером. Это означает, что секторы на каждой дорожке должны проверяться, идентифицироваться и получать метку электронным методом с помощью TRDOS. Начиная с этого момента TRDOS будет следить за тем что и где хранится на диске.

В TRDOS содержится стандартная программа форматирования диска, и нет необходимости в загрузке дополнительного матобеспечения. Форматирование может выполнятся в любое время, даже если программа находится в памяти. Формат команды следующий:

FORMAT "DISCONE", где DISCONE — имя диска.

Нажмите <ENTER> для завершения команды и ждите. Время, затрачиваемое TRDOS для разметки секторов, колеблется между одно и двухсторонними 40 и 80-дорожечными дисководами. Если дисководы двухсторонние, то обе стороны форматируются автоматически. по завершении на экране появляется:

DISCONE
624/624 или 1264/1264 или 2544/2544
A>

На этом дисплее представлено название диска, за которым следует количество секторов для данного формата диска. Если первый номер меньше второго, ваш диск дефектный.

6.1.6. Команды CAT и LIST

Для представления на экране содержания диска существуют две команды. Первая и наиболее часто используемая команда CAT, вторая команда LIST.

Команда CAT представляет на экране название, тип и размер сектора файлов и подходит для осуществления почти всех целей.

Команда LIST представляет на экране каталог с расширенной информацией и идеально подходит для анализа программы.

Если количество файлов, подлежащих представлению на экране более 30, появляется "SCROLLER". Нажатие обычной клавиши SPECTRUM обеспечит продолжение представления на дисплее. Перечень завершается символами "N" или "BREAK".

6.1.7. Копирование файлов

Существуют три команды для копирования:

COPY — для копирования обычного файла.

COPY S — для копирования файла системы с одним дисководом.

COPY B — для дублирования системы с одним дисководом.

Основной элемент синтаксиса — COPY.

Это ключевое слово SPECTRUM на клавише "Z". Основной синтаксис следующий:

COPY "NEW FILE", "OLD FILE"TYPE

Требуемый синтаксис следует практике SOS таким образом, что названия нового, так и старого файла помещаются в кавычках, а тип файла обозначается с помощью ключевых слов SOS.

Следует помнить, что в тех случаях, когда следует заключить в кавычки 2 наименования, например для команд COPY или NEW, первым заключается в кавычки новое наименование. К тому же тип не следует за новым названием, т.к. он идентичен старому.

Копирование на один и тот же диск

Следует сообщить TRDOS название файла, подлежащего копированию и название, под которым следует записать эту копию. Например:

COPY "VAT69", "WHISKY"CODE

Когда подсказка TRDOS снова появится, явного изменения не будет. При выполнении новой команды CAT мы обнаружим, что на дисплее представлен новый файл VAT69<C>.

Вы не можете записать два файла на один и тот же диск с идентичными названиями.

Копирование и дублирование на одном диске

Если имеется только один дисковод, нельзя использовать обычную команду COPY. Обе команды "COPY S" и "COPY B" предназначены для системы с одним дисководом.

Первая "COPY S" используется для копирования одного файла с одного диска на другой, используя один и тот же дисковод. Вторая команда "COPY B" используется для дублирования диска на другой диск, то есть для копирования всех файлов. Вот примеры команды "COPY S".

COPY S "WINES" или
COPY S "BEER" CODE

Где "WINES" и "BEER"CODE — существующие файлы. Заметьте разницу, прежде мы вносили новое наименование в кавычках тотчас же после подсказки. Теперь мы сообщаем TRDOS, какую программу копировать, и таким образом существующее наименование идет в кавычках после "COPY S".

Чтобы удостовериться, что у вас в дисководе правильный диск, вам дается подсказка вставить диск и нажать "Y". После прочтения диска программа дает вам подсказку заменить его вторым диском и ввести новое наименование, под которым будет копироваться новая программа.

Команда COPYB действует как расширение команды "COPY S". Стандартная программа прогоняется с помощью ввода команды "COPY B". Начиная с этого момента на экране появится подсказка относительно смены диска и какие клавиши нажимать.

Копирование и дублирование двойного дисковода

Копирование на другой дисковод автоматически означает, что вы будете копировать на другой. Хотя взаимный обмен может происходить между любыми комбинациями двух дисководов, мы используем дисководы "A:" и "B:". Из синтаксиса очевидно, как будет осуществляться ввод других комбинаций.

COPY "A:WHISKY", "B:WHISKY"

Хотя дисковод с умолчанием — это "A:", мы производим копирование с диска "B:" на "A:" и используем то же самое наименование, которое не используется на диске "A:". Если бы диски поменять местами в двух дисководах, у нас было бы:

COPY' 'B:WHISKY", "A:WHISKY"

Таким образом, копируется файл с дисковода "A:" на дисковод "B:".

Дублирование производится гораздо проще при использовании системы с двойным дисководом. Диск, подлежащий копированию, помещается в одном дисководе, а приемный диск (пустой и отформатированный или частично использованный) вставляется в другой дисковод. Могут использоваться любые дисководы. Мы используем дисковод "A:" в качестве источника и дисковод "B:" в качестве приемника. Синтаксис команды подобен копии одиночного файла с "*" (т.е. вместо имени файла — *) заменяя специфическое название файла, например:

COPY "B:*", "A:*"

Остальное будет выполнено вводом команды TRDOS. Если названия файлов не дублированы, а на принимающем диске имеется достаточно места, работа заканчивается возвратом подсказки A>.

Преимущество "*"COPY — это более быстрое копирование по сравнению с копированием каждого файла отдельно.

6.1.8. Переименование и стирание файла

В отличие от других команд TRDOS этот диск должен находиться в дисководе с умолчанием, и этим дисководом должен быть дисковод "A:".

Сделайте сначала CAT диска с тем, чтобы на экране был представлен подлежащий изменению файл. Теперь вводите ключевое слово NEW, а затем — новый заголовок (как всегда, в кавычках), а затем существующий заголовок, они разделяются запятой.

NEW "BOOT", "PROG"

Если файл на диске устарел и больше не требуется, его можно стереть. Командное ключевое слово — ERASE, оно получается с помощью перехода в режим E, нажатия и удержания SYMBOL SHIFT и нажатия клавиши 7, например:

ERASE "OLDPROG"

Теперь мы вводим <ENTER> команду. Когда через пару секунд вновь появляется подсказка, CAT подтверждает, что этот файл исчез, и факт зарегистрирован в заголовке CAT под номером устраненного файла.

6.1.9. Уплотнение пространства на диске — команда MOVE

Когда файл стерт, секторы, которые он занимал, должны быть освобождены для других файлов. Чтобы обнаружить такие "потерянные" секторы, используем команду MOVE. Это команда,

которая с наибольшей вероятностью должна использоваться из TRDOS, а не из SOS.

Войдя в TRDOS, мы выполним CAT с целью исследования диска, возможно, произвели некоторые стирания, и теперь заканчиваем с помощью MOVE с целью запроса пространства для использования.

Сама команда MOVE делает всю работу по реорганизации диска и его указателя. По завершении снова появляется подсказка TRDOS. Если теперь выводится CAT, на распечатке будет показано 0 устраненных файлов и увеличенное количество свободных секторов.

6.1.10. Запись, проверка, загрузка, прогон и слияние

Команды SAVE и VERIFY

С помощью команды SAVE программа в SPECTRUM записывается на диск. Следует указать название программы и взять его в кавычки.

Для программы BASIC не требуется типа файла. Однако, номер строки может быть указан после ключевого слова LINE для автопрогона. Если номер строки не указан, то программа будет прогоняться с первой строки. Например:

```
SAVE "HOMEACC" LINE 100
SAVE "GRAPH" LINE
SAVE "A:DESIGN"
```

Для программы в машинных кодах следует указать тип файла CODE, за которым следует стартовый адрес и количество байт, подлежащих сохранению, например:

```
SAVE "DISCOUNT"CODE 47800,955
SAVE "B:COLLEC"CODE 32768,4000
```

Команда VERIFY проверяет, является ли файл, записанный на диске, тем же самым, что и файл в памяти. Команда VERIFY может быть использована для проверки BASIC-программ, программ в машинных кодах и файлов массивов данных.

Если файлы различаются появится сообщение — VERIFY ERROR.

Команды LOAD и RUN

Если программа BASIC предназначена для автопрогона с номера строки, например:

```
SAVE "INTEREST"LINE 25
```

то автопрогон будет осуществляться, какая бы команда (LOAD или RUN) не использовалась.

Если программа BASIC не сохранена для автопрогона, то LOAD будет осуществлять загрузку и распечатку, а RUN будет осуществлять загрузку и прогон неавтоматической программы прогона. Например:

```
LOAD"INTEREST"
RUN"DESIGN"
```

При прогоне программ в машинных кодах адрес автопрогона должен быть тем же самым, что и стартовый адрес программы. например:

```
RUN "DISCOUNT"CODE 47800
```

При условии, что этот код предназначается для автопрогона, начиная от 47800, загрузка пройдет хорошо.

Команда MERGE (слияние)

Слияние TRDOS — то же, что и слияние SOS. Она использует то же ключевое слово (E MODE SYMBOL SHIFT) и служит той же цели слияния в памяти SPECTRUM программы BASIC с диска и этой же программы, уже имеющейся в памяти. Например:

```
MERGE "SUBROUT"
```

6.1.11. Волшебная кнопка

Волшебная кнопка предназначена для записи (сохранения) программ, расположенных на кассете и прогона их на диске без какого-либо преобразования.

Существует много программ, таким образом, что их очень трудно или невозможно преобразовать и прогнать на диске. Это могут быть программы со сложной защитой или программы, осуществляющие хранение и загрузку файлов данных для использования главной программой. Т.к. замена команд TRDOS невозможна, требуется другой метод. В этом случае программа загружается и создаются файлы данных. Вместо использования инструкций программы SAVE волшебная кнопка используется для вывода всей программы и файлов на диск. Последующая перезагрузка означает загрузку всей упаковки вместо файла данных.

Файлу присваивается имя "@".

6.1.12. Сообщения об ошибках и коды ошибок

Неавтономные сообщения

Когда вы вводите команду в TRDOS, команда будет выполнена при условии, что она достоверна. Если эта команда не относится к числу вызванных TRDOS (LOAD, RUN, FORMAT и т.д.), TRDOS проигнорирует ее. Если имеется синтаксическая ошибка или ошибка возникает при выполнении команды, на дисплее появится сообщение об ошибке. Ниже приведены сообщения об ошибках и причины их вызывающие.

1. NO DISK (диск отсутствует)

Диск отсутствует, или в дисководе неформатированный диск, или открыта заслонка диска. На экране возникает команда после подсказки A>, вы можете вставить диск и/или закрыть заслонку дисковода и нажать ENTER для выполнения той же самой команды.

2. NO FILE(S) (нет файла(ов))

TRDOS не может найти файла на диске. Эта ошибка появляется также в случае использования любой команды, которая неправильно обозначает файл.

3. *ERROR*

Это сообщение появляется, если в команде, которую вы напечатали, есть синтаксическая ошибка.

4. OUT OF MEMORY (вне памяти)

Это сообщение появится при загрузке программы с диска и при отсутствии достаточного места в памяти для нее и при использовании команды MOVE, если в наличии нет 4 кбайт необходимого рабочего пространства. Эта проблема обычно решается с помощью сброса компьютера.

5. FILE EXISTS (файл существует)

Файл того же названия и типа, который вы пытались записать, уже существует на диске.

6. OVERWRITE EXISTING FILE ? (Y/N)

Перезапись на существующий файл? (Д/Н). Это сообщение появляется при копировании всех файлов с одного диска на другой. Уже существует файл с таким названием и такого же типа на диске назначения. Вы можете ввести Y с целью перезаписи на существующий файл или N с целью игнорирования этого файла.

7. DISK ERROR (ошибка на диске)

На диске имеется неисправность на дорожке XX, в секторе YY, как указано в сообщении. У вас есть три выбора:вы можете напечатать "R", чтобы сделать повторную попытку и, в большинстве случаев успешно завершить повторную операцию; "A" — чтобы прервать операцию и вернутся в TRDOS; "I" — чтобы проигнорировать этот сектор и продолжить работу с остальными операциями.

8. WRITE PROTECT (защита от записи)

Диск защищен от записи. Как и вышеприведенном варианте существует три выбора. Однако нет смысла делать повторную попытку если вы не изменили диск или не сделали его незащищенным.

9. VERIFY ERROR (ошибка проверки)

Это сообщение может появится, когда используется команда VERIFY (проверка). Сообщение информирует, что файл на диске — не тот же, что и файл в памяти.

Коды ошибок

Все вышеназванные сообщения об ошибках появляются только тогда, когда ввод производится с TRDOS. Если команда была выдана с SOS или с машинного кода, либо в качестве прямого кода, либо из программы, на экране не появится никакого сообщения. Однако они записываются в виде CODE в регистровой паре Z80. Для кода используются следующие значения:

0 — нет ошибок.
1 — нет файла.
2 — файл существует.
3 — нет пространства.
4 — указатель переполнен.
5 — переполнение номера записи.
6 — нет диска.
7 — ошибка на диске.
8 — ошибка синтаксиса.
10 — поток уже открыт.
11 — диск не форматирован.
12 — поток не открыт.

Чтобы получить код ошибки, устанавливается переменная, равная команде TRDOS. Эта переменная примет величину кода ошибки по завершению команды TRDOS.

6.1.13. Информация о TRDOS

TRDOS занимает 112 байтов памяти с произвольным доступом.

Без подсоединенной TRDOS RAM пользователя начинается с:

(А) — адрес 23755 без подсоединенного интерфейса 1.

(Б) — адрес 23812 с подсоединенным интерфейсом 1.

При подсоединенной TRDOS RAM пользователя начинается с:

(А) — адрес 23867 без подсоединенного интерфейса 1.

(Б) — адрес 23925 с подсоединенным интерфейсом 1.

TRDOS использует сектора диска для хранения. Если количество байтов превышает 256, используется другой сектор. Это продолжается до тех пор, пока не происходит запись всего файла. Для записи 522 байтов потребуется 3 сектора. 3-ий сектор содержит только 10 байтов. Только эти 10 байтов будут загружены с диска.

В дополнение к 112 байтам RAM TRDOS также используется буфер в 256 байтов, при осуществлении доступа к диску. Этот буфер имеет динамическое распределение. При выполнение большого количества команд TRDOS сначала сдвигает программу BASIC (если таковая существует) вверх с целью создания буфера. После завершения команды программа BASIC передвигается назад к своей исходной позиции. Эта операция происходит незаметно.

Команда MOVE требует 4K (минимум) из SPECTRUM RAM в качестве рабочего пространства. Если MOVE выполняется, а программа еще в памяти, необходимо перевести компьютер в исходное состояние и затем выполнить команду MOVE. Перевод компьютера в исходное состояние может быть осуществлен с помощью возврата к SOS и ввода RANDOMIZE USR 0 или с помощью положения ''сброс''.

На одном диске может хранится до 128 файлов.

6.1.14. Системные переменные TRDOS

# адр.	адрес	длина	комментарий
5CB6	23734	1 X	используется, если есть INTERFACE 1 (если значение равно #F4, то область переменных не переносится, если равно #00, то проверяется 23832
5CC2	23746	1 X	содержит #C9. используется системой TR-DOS для вызова подпрограмм из SOS
5CC8	23752	1	код, определяющий режим работы дисковода А: бит 7 = 0 — дисковод 40-дорожечный = 1 — дисковод 80-дорожечный бит 1 = 1 — дисковод двухсторонний бит 0 = 0 — использовать 80-дорожечный, как 40-дорожечный
5CC9	23753	1	то же для дисковода В
5CCA	23754	1	то же для дисковода С
5CCB	23755	1	то же для дисковода D
5CCC	23756	1 X	текущий сектор при чтении каталога
5CCD	23757	1 X	#80 — готовность дисковода
5CCE	23758	1 X	#00 — чтение сектора, #FF — запись
5CD6	23766	1 X	при #FF — команда не выполнена
5CD7	23767	2 X	промежуточный старт (тип В и С) после проверки типа дисковода содержит количество дорожек
5CD9	23769	2 X	внутренний аналог CH ADD промежуточная длина (тип В и С)
5CDB	23771	2 X	промежуточная длина программы
5CDD	23773	8	имя файла в ASCII
5CE5	23781	1	тип файла
5CE6	23782	2	при типе С — стартовый адрес при типе В — длина BASIC-программы
5CE8	23784	2	длина файла
5CEA	23786	1	обьем файла в секторах
5CEB	23787	1	номер первого сектора файла (0-15)
5CEC	23788	1	номер первого трека файла
5CEF	23791	1 X	1, если есть INTERFACE 1
5CF4	23796	1 X	текущий номер сектора
5CF5	23797	1 X	текущий номер трека
5CF6	23798	1	дисковод для временной операции (0-3)
5CF7	23799	2	при возврате из 15616 обнуляется
5CF8	23800	1	дисковод при операции с двумя файлами #FF, если канал открыт
5CF9	23801	1	дисковод при операции с двумя файлами признак операции READ/VERIFY номер дисковода при команде #07
5CFA	23802	1	время перемещения головки дисковода А
5CFB	23803	1	время перемещения головки дисковода В
5CFC	23804	1	время перемещения головки дисковода С
5CFD	23805	1	время перемещения головки дисковода D
5CFE	23806	1 X	код команды для 1818ВГ93
5CFF	23807	1 X	номер сектора для подпрограммы чтение, запись сектора
5D00	23808	2 X	текущий адрес буфера (#05/#06)
5D02	23810	2 X	сохраняет HL для внутренних нужд
5D04	23812	2 X	сохраняет DE для внутренних нужд
5D06	23814	1	количество знаков при поиске имени файла (см. ком. #0A). Начальное значение #09
5D07	23815	1 X	счетчик удаленных файлов (ком. #12)
5D08	23816	1 X	первый символ имени файла (ком. #12)
5D0C	23820	1 X	флаг состояния рабочего буфера TRDOS (257 байт с адреса 23846) #FF — буфер отсутствует #00 — буфер существует
5D0E	23822	1 X	флаг принадлежности команды (#FF — работает BASIC, иначе — TR-DOS)
5D0F	23823	1 X	код ошибки TR-DOS внутри TR-DOS, при неравенстве 0 вводит команду RETURN, иначе пустую строку (подпрограмма #20EF)
5D10	23824	1 X	старший байт ошибки, при вызове 15616 обнуляется, при вызове 15635 необходимо обнулять принудительно (во избежание ошибочных ситуаций)
5D11	23825	2 X	адрес строки команды для TR-DOS при вызове 15616 повт. E.LINE(23641) при вызове 15619 — CH ADD (23645)
5D13	23827	2 X	копия ERR SP: при равенстве старшего байта #AA автоматически выполняется команда RUN ''BOOT'', а в 23833 код #FE
5D15	23829	1 X	при равенстве #00 печатает сообщения TR-DOS, иначе — не печатает

# адр.	адрес	длина	комментарий
5D16	23830	1	копия системного регистра (555ТМ9)
5D17	23831	1	при неравенстве #AA, при вызове 15616 рисуется заставка, при равенстве #FF не попадает на ошибку при чтении неверного адресного маркера
5D18	23832	1 X	используется при подключенном INTERFACE 1 (если значение равно #FF, то меняются местами блоки в памяти по адресам 23747-23859 объемом 45 байт, при вызове TR-DOS заносится #FF)
5D19	23833	1	дисковод по умолчанию (0-3)
5D1A	23834	2 X	внутренний адрес процедуры завершения (#0201)
5D1C	23836	2 X	сохраняет SP
5D1E	23838	1	номер файла, если он найден (ком. #0A)
5D20	23840	3	первые три символа введенной строки

Символом "X" отмечены внутренние переменные TR-DOS, которые не рекомендуется изменять в процессе работы.

6.1.15. Подпрограммы TRDOS

#00 восстановление (сброс контроллера). Ожидание появления сигнала INTRQ, выходит из ожидания при нажатии BREAK.

#01 выбор дисковода. Номер выбираемого дисковода указывается в регистре "A". При значении #FF по адресу 23802+ "номер выбираемого дисковода" (см. системные переменные) происходит инициализация дисковода (определяется количество дорожек дисковода, константа позиционирования и заносится в соответствующие системные переменные). В ячейку 23798 заносится номер выбранного дисковода. Во избежание ошибочных ситуаций рекомендуется его дублировать в ячейках 23800 и 23801.

#02 позиционирование. Головка устанавливается на трек, указанный регистром "A". В случае двухстороннего дисковода номерам 0 и 1 соответствует первый физический трек (0 и 1 сторона соответственно), номерам 2 и 3 второй и т.д.

#03 помещает содержимое регистра "a" по адресу 23807 (номер сектора).

#04 помещает содержимое регистровой пары "HL" по адресу 23808 (адрес буфера).

#05 чтение группы секторов. Перед вызовом подпрограммы в регистр "B" помещается количество читаемых подряд секторов (при значении 0 с диска считывается только область заголовка, считывание в память не происходит). В регистр "D" помещается номер трека, а в регистр "E" номер сектора. Регистр "HL" должен содержать адрес буфера в памяти, в которой будет производиться чтение.

#06 запись группы секторов. Параметры аналогичны #05.

#07 вывод каталога диска. В регистре "A" должен быть указан номер канала (для вывода на экран аккумулятор должен содержать значение 2, на принтер — 3, и т.д.). Выполнение подпрограммы аналогично выполнению команды TR-DOS "CAT". Перед выводом каталога выполняется команда #18.

#08 чтение информации о файле. В аккумуляторе должен быть номер интересующего файла (0-127). Из директории диска 16 байт информации будут помещены с адреса 23773 (проверка на наличие данного файла не происходит). В номера файлов входят и удаленные.

#09 запись в каталог информации о файле. С адреса 23773 16 байт переписываются в каталог диска на место информации о файле, номер которого задается регистром "A".

#0A поиск файла. Поиск файла в директории ведется по имени и типу (с адреса 23773). Количество байт, по которым ведется поиск задано по адресу 23814.

#09. Если файл найден, то, по возвращению из подпрограммы, регистр "C" будет указывать его порядковый номер (то же по адресам 23828, 23823). В противном случае, старший бит регистра устанавливается в 1 (23828 не изменяется, 23823 будет содержать #FF)

#0B запись файла. С адреса 23773 — имя и тип файла, в регистровой паре "HL" — адрес начала в памяти, в паре "DE" — длина файла.

#0C запись BASIC-программы. С адреса 23773 — имя и тип файла (при типе отличном от <B>, файл записывается под именем "BOOT <B>").

#0D нет команды

#0E чтение/проверка файла. Имя и тип файла должны быть помещены с адреса 23773. Адрес загрузки файла берется из директории (при аккумуляторе равном #00) или из регистровой пары "DE" (при аккумуляторе равном #03). Значение по адресу 23801: #00 — LOAD, #FF — VERIFY.

#0F нет команды

#10 нет команды

#11 нет команды

#12 удаление файлов. Имя и тип файла — с адреса 23773. Удаляются все файлы с такими данными

#13 переписываются 16 байт информации из памяти, адресуемые регистровой парой "HL", по адресу 23773.

#14 переписываются 16 байт информации из 23773 в память по адресу "HL".

#15 проверка дорожки. Регистр "D" должен содержать номер проверяемого физического трека.

#16 загрузка системного регистра. Код — в аккумуляторе. Предварительно к нему прибавляется #3C.

#17 выбрать нижнюю сторону.

#18 настройка на диск. Проверяет тип диска (8-ой сектор директории).

6.1.16. Формат каталога диска

Каталог занимает на диске нулевую дорожку. Сектора с 0 по 7 используются для хранения информации о файлах (по 16 байт на файл):

BYTE 0...7 — имя файла в ASCII
BYTE 8 — тип файла
BYTE 9, 10 — параметр "START"
BYTE 11, 12 — длина файла
BYTE 13 — объем файла в секторах
BYTE 14 — номер первого сектора файла
BYTE 15 — номер первого трека файла

Если первый байт заголовка содержит значение #01, то файл считается удаленным. По значению #00 определяется конец каталога.

8-ой сектор нулевой дорожки содержит информацию о диске в целом:

адрес от начала сектора	комментарий
#E1	первый свободный сектор
#E2	первый свободный трек
#E3	тип диска: #16 — 80 дорожечный, двухсторонний #17 — 40 дорожечный, двухсторонний #18 — 80 дорожечный, односторонний #19 — 40 дорожечный, односторонний
#E4	общее количество файлов
#E5,#E6	количество свободных секторов на диске
#E7	код #10, определяющий принадлежность диска к системе TR-DOS
#F4	количество удаленных файлов
#F5 — #FC	имя диска в ASCII

6.2. MOA SERVICE

Возможности программы MOA SERVICE V01.05:

— позволяет в короткое время (нажатием двух клавиш) получить полную информацию о дискете в удобной форме;

— быстрая смена дисковода;

— наглядный поиск интересующего файла в каталоге;

— запуск из себя программы написанной на бейсике, сброшенной "волшебной" кнопкой", а также программы написанной в кодах;

— групповые операции с файлами: поиск, удаление, переименовывание и копирование;

— копирование файлов в системах с любым числом дисководов (в том числе и с одним);

— просмотр и изменение (восстановление) удаленных файлов;

— "дампинг" файлов;

— выполнение из себя любой команды TRDOS;

— память последних набранных команд и возможность их редактирования.

Начало работы

Программа состоит из двух модулей, один из которых является загрузчиком, написанным на бейсике. Рекомендую присвоить ему имя ''BOOT'' — при этом для запуска вам нужно будет только вставить дискету в дисковод и нажать на кнопку 'RESET'.

После загрузки перед вами появится заставка, указывающая на тот дисковод, с которым вы будете работать (после загрузки это будет дисковод A.

Вставьте интересующий вас диск в дисковод A и нажмите [ENTER]. Если вы хотите сменить диск нажмите букву B, C или D и затем [ENTER].

В случае, если вы не забыли закрыть дверцу дисковода, перед вами появится картинка.

Каждое окно предназначено для определенных целей, рассмотрим их.

Поле, расположенное слева, предназначено для выбора интересующих вас файлов. Текущим считается тот файл, который отмечен курсором (линия повышенной яркости). Курсор можно перемещать по экрану клавишами [UP] и [DOWN], либо использовать комбинацию [CAPS SHIFT/6] и [CAPS SHIFT/7].

Если каталог диска содержит много файлов, удобно использовать возможность быстрого перемещения курсора. Для этого перед клавишами [UP] или [DOWN] нажмите на [EXTEND MODE], курсор переместится сразу на 16 файлов.

Справа вверху выводится сводная информация о диске. Сразу под ней находится окно, предназначенное для информационных сообщений, подсказки и сообщений об ошибках.

Третье окно справа показывает параметры текущего файла. В низу экрана находится поле, в котором работает строчный текстовый редактор. Это окно предназначено для ввода команд TRDOS, поэтому будем называть его командным.

Смена дисковода (отмена команды)

В любой момент вы можете отменить непонравившуюся вам команду, нажав только на клавишу [EDIT] (либо на [CAPS SHIFT/1]).

Нажмите клавишу [ENTER], если хотите перечитать каталог или еще раз на клавишу [EDIT], если вы решили отменить операцию.

Запуск программ

Если окно команд чисто, то для запуска программы необходимо сделать интересующий вас файл текущим (его параметры должны отображаться в третьем сверху окне справа) и нажать клавишу [ENTER]. Командное окно можно очистить, для этого нажмите на [SYMBOL SHIFT/U].

Файлы с типом B или сброшенные по MAGIC начнут загружаться сразу же. Загрузка и выполнение файлов, содержащих коды (имеющих тип C), в общем аналогична команде TRDOS 'RUN ''NAME'' CODE ADDRESS', однако имеется то преимущество, о котором может мечтать искушенный пользователь: программа MOA SERVICE после загрузки файла передает ему управление согласно соглашениям BASIC, чего не делает TRDOS.

Строчный редактор

Если вы видите на экране символ, значит вы работаете в строчном редакторе 2MOA SERVICE0. Можно использовать стрелки [CAPS SHIFT/5] и [CAPS SHIFT/8] для перемещения курсора (если предварительно нажать на [EXTEND MODE], то курсор переместится к началу или концу строки); клавишу [DELETE] для удаления символа перед маркером. Очистка поля редактора — [SYMBOL SHIFT/U]. Фиксация строчных символов — [CAPS LOOK] ([CAPS SHIFT/2]). Для ввода специальных символов (<, [и др.) поступайте так же, как в BASICe — [EXTEND MODE] + [SYMBOL SHIFT/KEY]. (Напомню, что [EXTEND MODE] это [CAPS SHIFT] + [SYMBOL SHIFT]).

Вызов команд TRDOS

Находясь в MOA SERVICE вы можете выполнить любую команду TRDOS, для этого ее нужно набрать в командном экране. Здесь вы можете использовать клавиши [UP], [DOWN], [DELETE]. Выполнение команды начинается с момента нажатия на клавишу [ENTER].

Вы можете вызвать на экран предыдущую команду нажав на [SYMBOL SHIFT/I] (в памяти хранится информация примерно о трех ранее набранных командах), очистить экран можно нажав на [SYMBOL SHIFT/U].

Выделение файлов

Для групповых операций над файлами их нужно выделить, достигается это нажатием на клавишу [GRAF] ([CAPS SHIFT/9]). При этом текущий файл окажется выделенным в прямом смысле этого слова — строка содержащая информацию о нем изменит цвет. Отменить выделение можно повторно нажав на [GRAF].

Снять выделение со всех файлов можно дважды нажав на клавишу [EDIT].

Общий объем выделенных файлов высвечивается в специальном окне на экране.

Вызов каталога

Иногда бывает необходимо просмотреть каталог диска, не пользуясь средствами программы MOA SERVICE (например, чтобы увидеть файлы подавленные фильтром, см. ниже). Для этого нажмите на клавиатуре оператор 'CAT' ([EXTEND MODE] + [SYMBOL SHIFT/0]). Возврат в программу — любая клавиша.

Выбор меню команд

Программа MOA SERVICE предоставляет возможность выполнить ряд групповых команд с файлами. Для этого перейдите в меню команд нажав клавишу [TRUE VIDEO] ([CAPS SHIFT/3]).

Клавишами [UP] и [DOWN] выберите интересующий вас пункт и нажмите на [ENTER].

Функция QUIT позволит вам выйти из командного меню не отменяя при этом выделенных файлов.

Функция INSERT отыщет и выделит в каталоге указанные вами файлы по маске. В появившемся экране отредактируйте маску по которой будет произведен поиск, символ [?] в маске означает любой символ ASCII в имени файла.

Функция DELETE удалит из каталога выделенные вами файлы.

Функция RENAME позволяет переименовывать файлы. подробнее чуть ниже.

Функция COPY может быть использована для копирования файлов. Подробнее см. в пункте посвященном копированию.

Функция HEX DUMP выведет запрошенный вами файл в шестнадцатиричном виде. Подробнее о использовании этой функции написано в пункте посвященном просмотру файлов.

Функция FILTR позволит вам работать только с теми файлами, которые соответствуют выбранному фильтру. Задайте имя фильтра, так же, как и для 1INSERT0. В поле каталога будут выведены файлы, для которых имя и тип совпадают с фильтром. Помните, что фильтр действует на все последующие операции с любыми дисками и дисководами.

Функция ONE DEL FILE приведет к выводу каталога в обычном виде.

Функция +DEL FILE включит в каталог удаленные файлы, можно восстановить их переименовыванием. Этой операцией нужно пользоваться с большой осторожностью.

Функция EXIT производит выход из MOA SERVICE в BASIC.

Переименование файлов

При некоторых операциях с TRDOS на вашем диске могут оказаться файлы с одинаковыми именами и типами (скажем при использовании несколько раз кнопки MAGIC). В таких случаях необходимо иметь возможность переименовать находящиеся на диске файлы. Такая возможность предоставляется программой MOA SERVICE посредством выбора функции RENAME. Перед тем, как ее использовать выделите в каталоге файлы, которые вы хотите переименовать.

Введите новое имя файла в появившемся окне, при этом вместо используемого вами символа [%] в имена выделенных файлов будет подставлен символ ASCII, монотонно увеличивающийся от файла к файла, начиная со значения ''0''. Как можно догадаться это облегчает переименовывание файлов, сброшенных MAGIC в ZX SPECTRUM +2. Вы можете, как и для функции INSERT, использовать символ [?] для указания на любой символ ASCII в данном месте имени файла (кроме типа файла)

В случае переименовывания удаленного файла, находившегося за границей каталога, может возникнуть ошибочная ситуация, связанная с нумерацией положения файла на диске и его длины. При этом программа MOA SERVICE предупредит вас об возможной ошибке.

Каталог будет перезаписан только при нажатии на клавишу [Y].

Копирование файлов

Копирование файлов одна из самых мощных возможностей программы MOA SERVICE. Этой командой можно скопировать один или несколько файлов на одном или на разных дисководах значительно удобнее, чем используя прямые команды TR-DOS.

Для копирования выделите нужные вам файлы и выберите функцию COPY. В появившемся окне отредактируйте имя дисковода, куда вы собираетесь копировать, закончите редактирование клавишей [ENTER].

Сразу после этого будет произведена проверка на дублирование файлов и размер свободной области на выходном диске. При ответе [N] файл будет удален из операции копирования, любой другой ответ перезапишет файл. При превышении размера выходного диска будет выдано соответствующее сообщение, после которого копирование будет продолжено, пока хватит места.

Если вы выбрали для копирования то же устройство, что и исходное, то программа попросит вас несколько раз сменить диски, Если же устройства выбраны разные, то все произойдет без вашего участия. Помните, что 'OUTPUT DISK' — это выходной диск, т.е. тот на который вы собрались копировать, а 'INPUT DISK' — входной диск, тот с которого вы копируете. Не перепутайте!

Возможно, что при копировании вы передумали или оно вас утомило. В этом случае вы можете прервать операцию нажав, как всегда, на клавишу [EDIT]. Перед вами появится странное сообщение, суть которого сводится к тому, что вы должны решить сохранять или нет на выходном диске те файлы, которые к этому моменту уже скопированы. Дело в том, что программа, копируя файлы, не вносит их названия в каталог, а делает это в конце операции за один раз — оптом, поэтому, если вы не запишите каталог, то это будет означать, что ни один файл не записался на выходной диск — ведь каталог не изменится.

При чтении файла с диска его параметры выводятся на экран, так что вы всегда знаете текущий копируемый файл. Если вы прервали копирование или оно окончилось неудачей, именно этот файл окажется пограничным, т.е. скопированы будут все файлы до него. К сожалению, из-за недостатка TRDOS, при ошибке диска ни один файл не будет скопирован. Что бы не попадать в такое положение используйте сначала копирование в ''нулевое'' устройство, речь о котором пойдет ниже.

Если вы работаете в системе с одним дисководом, то несомненную пользу доставит вам возможность программы MOA SERVICE производить копирование в ''нулевое'' устройство, т.е. такое устройство, которого физически нет. Объясню на примере. Вы долго и терпеливо переставляете дискеты, вы даже не перепутали их, но на самом последнем файле произошла ошибка чтения, конечно, все, что вы переписали до сих пор, сохранилось на выходной дискете, однако, как уже говорилось выше, в каталог ничего не записалось.

Согласитесь, что лучше копировать хорошие файлы, но как узнать какие из файлов испорченные, а какие нет? Для этого отметьте те файлы, которые вам нужны, выберите копирование, а вместо имени дисковода укажите ''0:'' (двоеточие обязательно) — и файлы будут последовательно считаны, но никуда не записаны, причем, если все прошло успешно, то вам нужно выбрать опять копирование, но указать уже существующий дисковод (при фиктивном копировании отметка с файлов не снимается).

В заключение хочу заметить, что даже если вы копируете один файл (в системе с одним дисководом), то программа попросит минимум дважды сменить диски, так что подумайте прежде, чем использовать этот режим, а не проще ли набрать команду SAVES''NAME''? (обратите внимание, что SAVES набрано без пробела).

Сказанное выше ни коим образом не означает, что если вы копируете два файла, то программа попросит четырежды сменить диски — нет, нет и нет! Просто минимально необходимое число смен дисков для обеспечения сервиса и компактности не получается меньше, чем два. Так что извините...

Просмотр файлов

Иногда необходимо внести некоторые, небольшие изменения в программу (файл) не прибегая к услугам программ—трансляторов, такое необходимо на стадии отладки. Может возникнуть ситуация, когда нужно немного изменить испортившийся каталог. В этих случаях удобно использовать программу ''DISK DOCTOR''.

Однако, чтобы лишний раз не обращаться к диску, вы можете просмотреть файл (или область каталога диска), используя встроенную функцию программы MOA SERVICE — HEX DUMP. Выберите ее в командном меню. Затем сделайте выбор: если вы решили посмотреть каталог, то нажмите на клавишу 'C', любой другой ответ вызовет для просмотра файл, на который указывает курсор. При выводе указывается логический номер трека и сектора, считая от начала диска, т.о. можно узнать действительное местоположение файла на диске.

Каждая выводимая на экран строка, вначале содержит номер сектора и, отделенный от него точкой, относительный адрес первого выводимого в строке байта. Далее следуют восемь шестнадцатиричных значений, а за ними те же восемь значений, только в форме ASCII.

Выход из программы

Для выхода из MOA SERVICE можно воспользоваться функцией меню команд EXIT, при этом не происходит инициализации области BASIC-переменных, так что можно, например, выполнить команду TRDOS MERGE''NAME'' и при выходе программа ''NAME'' останется в памяти.

Проще, однако, набрать на клавиатуре оператор STOP, нажав [SYMBOL SHIFT/A], при этом все, что было в памяти уничтожается.

Возврат в программу возможен командой BASICa RANDOMIZE USR 57000, однако, помните, что для успешного рестарта программы вы должны ничего не менять в памяти выше этого адреса.

Помните, что при входе в программу MOA SERVICE BASIC-переменные всегда инициализируются, т.е. все что было в памяти пропадает.

При запуске программ BASICA переменная RAMTOP принимает значение 56999, для программ в кодах — 24300. Не сохраняется знакогенератор пользователя — переменная UDG будет указывать на знакогенератор расположенный в ПЗУ.

6.3. DISK DOCTOR V4.3

Экранный редактор программы DISK DOCTOR позволяет редактировать содержимое диска вводом шестнадцатеричных или символьных значений в зависимости от поля, в котором находится курсор.

Передвижения курсора осуществляются курсорными клавишами или KEMPSTON джойстиком. Клавиша EDIT позволяет выйти из программы в TRDOS. По возврату из операционной системы управление будет передано программе DISK DOCTOR. Переход в командный режим осуществляется клавишей EXTEND MODE, после нажатия которой курсор становится красным. Отказ от командного режима производится теми же клавишами, что и переход в него.

Клавиши TRU VIDEO и INVERSE VIDEO соответственно сбрасывают или устанавливают старший бит по указателю. Работают только в том случае, если курсор находится на символьном поле и в режиме ASCII-редактора.

Командный режим позволяет выполнить следующие операции (вызов производится нажатием соответствующей клавиши в командном режиме):

A — ASCII EDITOR

Переход в экранный ASCII-редактор. EDIT — возврат в основной режим; S. S+Q — страница назад; S. S+E — страница вперед.

B — PAGE BACKWARD — возврат на одну страницу.

F — PAGE FORWARD — следующая страница.

H — HELP PAGE — вывод на экран списка возможных команд.

I — INFORMATION — вывод на экран информации о диске, положении курсора и файле, на котором находится курсор.

N — FIND NEXT STRING — поиск следующей последовательности байтов, заданной командой FIND.

O — OPEN FILE — открыть файл. Курсор устанавливается на первый байт заданного файла.

На подсказку сначала вводится название файла, а затем первая буква спецификации (BASIC, CODE и т.п.). Если файл с таким именем не найден, будет выдано соответствующее сообщение.

P — PUT SECTOR — запись текущего сектора на диск.

R — RELOAD TRACK — перегрузка текущего буфера (дорожки). может использоваться при смене диска или внесении ошибочных изменений.

S — SAVE CHANGES — запись на диск произведенных изменений.

T — SELECT TRACK/SECTOR — выбор сектора для редактирования. На подсказку сначала вводится номер дорожки, затем номер сектора. Если вместо номера дорожки введена пустая строка, выбирается текущая дорожка.

X — FIND STRING — поиск на диске последовательности байтов. На подсказку вводятся интервал (номера дорожек с которой и до которой будет производится поиск), затем последовательность байтов, которая может включать как шестнадцатеричные значения, так и символьные строки в произвольном сочетании. Символьные строки необходимо заключать в кавычки. В случае ввода пустой строки будет производится поиск ранее заданной строки.

Если в текущем буфере были произведены изменения, то при выполнении всех операций, связанных с перегрузкой буфера, автоматически будет выбираться режим SAVE CHANGE.

В новой версии (4.3) DISK DOCTOR учтены и доработаны некоторые недостатки. В том числе:

— добавлена возможность восстановления первой цифры ошибочно введенного шестнадцатеричного числа (во всех режимах);

— разрешен ввод специальных символов (<, > и др.);

— при выходе из ASCII-редактора положение курсора не изменяется;

— введена дополнительная проверка формата записи на диске, что позволило предохранить программу от возникновения ошибочных ситуаций;

— предоставлен выбор рабочего дисковода.

6.4. DCU

Как правило, владельцы синклер-совместимых компьютеров, работающих в среде TRDOS, обращают внимание на слабые сервисные возможности операционной системы. Значительно расширить их позволяет программа Н. Родионова 'DRIVE CONTROL UTILITY' (сокращенно DCU).

Существует несколько версий программы. Подробно в данном описании дана лишь первая версия программы DCU 2.01. Последующие версии программы отличаются от первой лишь некоторыми дополнительными возможностями о которых будет сказано ниже.

Программа DCU предназначена для: форматирования дисков в формате TRDOS; проверку дисков на наличие сбойных участков; восстановление испорченных дисков.

После загрузки программы на экране появится главное меню:

SELECT OPTIONS

DIAGNOSTICS
FORMAT DISK
CHECK DISK
RESTORE DISK
CHANGE DRIVE
ABOUT ME
QUIT PROGRAMM

В поле главного меню находится стрелочный курсор, который управляется либо одним из джойстиков, либо с клавиатуры клавишами O, P, X, S, SP. Выбрав курсором необходимую команду нажмите 'огонь', после чего программа выполнит команду или запросит дополнительные данные.

Итак, что же позволяют опции главного меню:

1. DIAGNOSTICS

При выборе этой команды программа выдаст сообщение о текущем состоянии вашей микрокомпьютерной системы:

USING TR_DOS V5.XX	— среда TRDOS версия 5.XX
CURRENT DRIVE: A	— текущий дисковод: A
DISK TITLE: XXX	— имя диска: XXX
HARD DISK NOT ATTACHED	— жесткий диск не подключен

2. FORMAT DISK

Эта опция главного меню служит для форматирования дисков, при ее выборе появится дополнительное меню:

FORMAT DISK

SINGLE SIDED		— односторонний
DOUBLE SIDED	*	— двухсторонний
40 TRACKS		— 40 дорожек
80 TRACKS	*	— 80 дорожек
MAXIMUM TRACKS	*	— максимальное количество дорожек
FAST DISK	*	— быстрый диск
ORDINARI DISK		— обычный диск
GO		— выполнить форматирование

С помощью стрелочного курсора вы можете перенастроить параметры форматирования вашего диска.

Первые четыре параметра общеизвестны. Функция MAXIMUM TRAKS позволит отформатировать на вашем диске максимальное количество дорожек (как правило больше 80). Функция FAST DISK позволит разметить сектора на дорожках вашего диска не подряд, а таким образом, что чтение файла с этого диска будет осуществляться значительно быстрее, чем с обычного.

После настройки параметров форматирования дайте команду GO, и программа запросит у вас имя диска. Набрав с клавиатуры имя диска нажмите ENTER. После этого начнется форматирование диска, причем DCU выведет на экран окно в котором будет отображаться состояние каждого сектора. Если диск форматируется нормально, то сектора на экране будут закрашиваться одним цветом. Если сектор не форматируется с первого раза, он будет окрашен в другой цвет, Если один из секторов на экране вообще исчезнет, то форматируемый диск лучше не использовать.

3. CHECK DISK

Эта команда DCU позволит вам проверить диск на наличие плохих (сбойных или вовсе не читаемых) секторов.

После ввода команды последует запрос: обычный диск или быстрый. Укажите как был отформатирован ваш диск, если ошибетесь, программа будет проверять диск гораздо медленнее.

Информация о состоянии диска будет отображаться практически так же как и при форматировании.

4. RESTORE DISK

Очень полезная опция, которая поможет вам восстановить сбойный диск (т.е. который читается с нескольких повторов).

При запуске RESTORE DISK, программа, как и в случае CHECK DISK запросит как отформатирован диск: FAST или ORDINARI. Получив ваш ответ DCU начнет реставрировать диск: считав информацию с дорожки отформатирует ее, затем снова запишет считанную информацию, проверит дорожку на наличие сбоев и если они есть снова повторит эти операции. После этого перейдет к восстановлению следующей дорожки.

Информация о состоянии дорожек и секторов выводится на экран как и в CHECK DISK.

5. CHANGE DRIVE

Эта команда позволит вам выбрать другой накопитель (в системе с двумя дисководами). Если текущим был дисковод A, то выберется B и наоборот.

6. ABOUT ME

При вводе этой директивы на экране появиться информация автора программы о способе ее приобретения, и санкционированного распространения.

7. QUIT PROGRAMM

Выход из программы. При выборе QUIT PROGRAMM программа даст дополнительный запрос, если ответить OK, то ваш компьютер перезагрузится и выйдет в TRDOS.

8. Отличия DCU V2.02

В этой версии программы автор добавил две новых возможности: встроенные часы и отключение контроля ошибок при форматировании.

Часы устанавливаются опцией SET TIME главного меню

Если вы уверены в качестве вашей дискеты и дисковода то можете отключить контроль ошибок при форматировании командой CHECK MEDIA (так чтобы напротив нее установился крестик) и форматирование будет выполнено значительно быстрее.

9. Отличие DCU V2.12

В этой версии DCU появилась возможность получения обобщенной информации о состоянии диска. Эта информация появляется после выполнения команды CHECK DISK. Если на вашем диске найдены сбойные участки их адрес появиться на экране, а также DCU сообщит какие файлы расположены на сбойных местах.

7.1. ARTIST

Введение

Программа ARTIST является наиболее мощной и гибкой среди существующих в настоящее время графических программ для компьютеров класса «SPECTRUM». Программа создавалась в расчете на любых пользователей — как начинающих, так и художников, занимающихся профессиональным созданием графических рисунков.

Обзор

После загрузки программы в нижней части рабочего экрана высвечивается меню, содержащее следующие опции:

```
  1 2  3
BRUSH BRUSH PATTERN      TEXT
(кисть)   (узоры кистью) (текст)

 4 5  6
VICK       MOVE  CIS
(обзор) (перемещение)      (стирание)

  7     8
STORAGE  CHR
```

Перед тем, как будут подробно рассмотрены вышеперечисленные опции, рекомендуется «поупражняться» с данной системой. Для того, чтобы это осуществить, необходимо знать, с помощью каких клавиш перемещается по экрану курсор, что нужно для того, чтобы курсор «закрасил» пиксели (наименьшие точки на экране), каким образом стирать пиксели с экрана и т.п.

Q— движение курсора вверх
S — движение курсора вниз
R — движение курсора влево
T — движение курсора вправо.

Нажимая на вышеуказанные клавиши, можно убедиться, что «точечный» курсор действительно перемещается по экрану. Нажатие на клавиши (комбинации клавиш) и удерживание их в нажатом состоянии приведет к перемещению курсора вдоль диагоналей (пока не нужно обращать внимания на второй курсор, имеющий вид крестика). Можно увидеть, что курсор (его движение) обладает некоторой «интеллектуальностью», т.е. ускоряет свое перемещение по экрану при длительном нажатии клавиши.

Для того, чтобы нарисовать фигуры, линии и т.п. на экране монитора, необходимо установить («включить») пиксели, находящиеся под перемещающимся по ним курсором. По желанию можно удалять из рисунка некоторые точки и закрашивать на экране различные фигуры.

C— установка (включение) пикселя (пикселей),
X— удаление пикселя(лей),
Z— установка атрибутов (при установке цветов можно также пользоваться клавишей SYWBOL/SHIFT).

При экспериментировании с выше указанными клавишами, видно, что можно легко получить круговые движения курсора.

Виды меню

В программу заложены три основных и два дополнительных меню, но сначала мы рассмотрим другие вопросы.

В каждом случае в состав высвечивающихся на экране меню входят цифры, стоящие над определениями функций и их возможностей. Эти цифры соответствуют номерам клавиш верхнего ряда клавиатуры, отвечающих данным функциям. Так, нажатием на клавишу «1», мы выбираем опцию, определенную словом »BRUSH» (кисть).

«BRUSH».

Выбор этой опции приведет к тому, что на экране пропадает меню и на его месте появляются цифры от «1» до «0». Под ними будут указаны различные величины так называемой «ширины кисти». Цифрами от 1 до 8 выбирается толщина линии и автоматически осуществляется переход к основному рабочему экрану. Существуют еще и другие виды кистей, обозначенные цифрами «9» и «0». Цифра «9» нажимается в том случае, когда нужно писать с изменяющейся толщиной кисти (как в случае письма пером). Цифра »0" приведет к выполнению функции «AIR BRUSH» (распылитель), с помощью которого можно «опрыскивать» экран, а не рисовать на нем четко очерченные линии или знаки, что, в сочетании с выбранным узором кисти, позволит производить растушевку экрана.

«BRUSH PATTERN» (узор кистью).

После выбора этой опции (нажатием клавиши «2») основного меню на рабочем экране, на экране вновь появятся цифры от «0» до «9», а под ними — прямоугольники, составленные из различных узоров. Как можно догадаться, выбор требуемого узора производится нажатием соответствующей клавиши.

«TEXT» (текст).

Точно так же, нажатие на клавишу «3» при основном рабочем экране, приведет к переходу в режим »TEXT WRITTING MODE" (написание текста), после чего на экране высветится новое меню:

```
WRITE MODE

CAP+3-4 SYM+G SYM+S SYM+D
INVERT OVER SMALL NORMAL
(инверсия) CHR CHR
(малые знаки) (нормальная высота)
CAP+2 CAP+SYM
CAPSLOCK EXTENTED
(большие буквы)
```

CAPS SHIFT + 5, 6, 7, 8 — курсор.

В этом режиме клавиши соответствуют указанным на них стрелкам, однако в этом случае невозможно выполнение режима GRAPHICS.

CAPS SHIFT и «4».

Одновременное нажатие на указанные клавиши приведет к включению режима «INVERT». Выключение производится одновременным нажатием CAPS SHIFT и «3».

CAPS SHIFT и «S»

Использование заложенного в программу комплекта малых знаков (64 символа в строке). Возвращение к знакам нормального размера делается нажатием на клавиши SYMBOL SHIFT и «D».

CAPS SHIFT и «2»

Переход в режим заглавных букв.

SYMBOL SHIFT и CAPS SHIFT — переход в режим EXTENTED (расширение).

Другие возможности текстового режима:

SYMBOL SHIFT и «F»

Изменение комплекта знаков (существует 8 различных комплектов).

SYMBOL SHIFT и «I»

Вывод на экран выбранного комплекта знаков; для того, чтобы просмотреть все комплекты, нужно трижды одновременно нажать клавиши SYMBOL SHIFT и «I», помня, что между очередными нажатиями необходимо отпускать клавиши. Возвращение к главному меню — нажатие клавиши ENTER.

«VIEW» (обзор)

Нажатие клавиши «4» в то время, когда высвечивается основной рабочий экран, позволяет увидеть всю область рисунка, т.к. исчезают надписи основного меню. Вернуться к нему можно, нажав любую клавишу.

Конечно, использование режима VIEW необходимо в случае проектирования рисунков, полностью заполняющих область экрана.

«MOVE» (движение)

Это — альтернативный режим по отношению к VIEW, т.е. способ рассмотрения результатов своей графической деятельности на экране. Нажатие на клавишу «5» приведет к перемещению экрана вверх с потерей трех верхних текстовых линий и открытием линий, закрываемых меню. Этот режим необходим в процессе работы над картиной.

«CIS» (стирание)

Нажатие на клавишу «6» приведет к стиранию экрана, после чего можно вновь начинать работу. Во избежание случайного стирания, после нажатия на клавишу »6", на экране высвечивается надпись:

PRESS /Y/ TO CLEAR SCREEN «STORAGE»

Клавиша «7» позволяет перейти к следующему меню, связанному с различными методами записи данных и рисунков. Сюда входят следующие режимы:

C— COPY (копирование). С помощью этого режима можно перенести картинку с экрана на принтер;

O— SEARCH (поиск). Позволяет просматривать магнитную ленту с записями без ввода их в компьютер. Нажатие на SPACE прекращает поиск и возвращает к STORAGE;

Y— LOAD CHR$ SETC (ввод комплекта знаков). Данный режим используется при вводе комплекта знаков, записанных ранее на ленте. Данный режим загрузится, на экране появляется вопрос — какой комплект должен быть введен (начиная с какого).

T— SAVE CHR$ SETC (запись комплекта знаков на носитель памяти) — так же, как и предыдущий режим, но для записи;

S— SAVE SCREEN (запись графики на носитель памяти). Этот режим позволяет перенести картинку с экрана на выбранный вид носителя памяти;

L— LOAD SCREEN (ввод графики). Ввод экрана в память компьютера;

V— SAVE USR GRAPHICS (запись знаков, определяемых пользователем, на носитель памяти). С помощью этого режима вы можете записать знаки UDG. В действительности — это последние 21 знак комплекта знаков 7 программы ARTIST;

I— LOAD USR GRAPHICS (ввод знаков UDG) на место, занимаемое последними 21 знаками комплекта 7.

Режимы второго основного меню

Как уже упоминалось выше, для того, чтобы выбрать одно из трех рассматриваемых основных меню программы, достаточно просто нажимать на клавишу «SYMBOL SHIFT». После того как это будет сделано, на экране появится новое меню, а именно:

1 2 3 4
OVER INVERT OVERLAY PATTERN
5 6 7 8
ENLARGE LINE CIRCLE BOX
9 0
ARC FILL

Вместе с опцией выбора величины и узора кисти из первого меню, настоящее меню является ключом для создания очень сложных графических картин с помощью программы ARTIST.

«OVER» (инверсия). Клавиша «1».

Этот приказ касается только четырех рассматриваемых далее графических инструкций (LINE, CIRCLE, BOX, ARC). По этому приказу происходит инверсия пикселей, рисуемых с помощью четырех вышеуказанных инструкций.

«INVERT». «2».

Этот приказ похож на предыдущий и касается только графических инструкций. Вместо инверсии состояния пикселей эта команда приводит к их удалению.

«OVERLAY». «3».

Это мощная и сложная команда, поэтому она будет рассмотрена в дальнейшей части.

«PATTERN» (сетка). «4».

Эта опция позволяет наложить на экран клетчатый узор, состоящий из светлых и нормальных квадратов, соответствующих размерам знаков. Эффект шахматной доски идеально подходит для определения границ, разделяющих отдельные поля знаков на экране, что особенно важно при нало.кении цветов.

«ERANGLE» (увеличение). «5».

Этот режим позволяет увеличивать фрагмент экрана вокруг позиции основного курсора. Для того, чтобы убедиться в необходимости рассматриваемого режима, нужно нарисовать несколько произвольных линий и форм на экране, а потом перейти к режиму ERANGLE, нажимая клавишу «5». Область вокруг курсора увеличивается. Теперь можно приступать к работе с увеличенным фрагментом рисунка. Все графические инструкции, такие как LINE, ARC и т.п., доступны при работе с этим режимом; кроме того, легко заметить, что можно перемещаться по рисунку, без необходимости входить и выходить в режим ERANGLE. В данном режиме возможны также установка и устранение атрибутов (при помощи клавиш «Z» или «SIMBOL/SHIFT», как указывалось раньше).

«LINE» (линия). «6».

Судя по названию, этот режим позволяет рисовать линии на экране, хотя и здесь также встречаются несколько видов черчения линий на экране.

Чтобы подробнее рассмотреть работу отдельных режимов, необходимо сначала вернуться к первому основному меню (двухкратным нажатием на клавишу SYMBOL/SHIFT, а также очистив экран с помощью клавиши «6»). Теперь нужно перейти ко второму основному меню. Если нажать на клавишу »6" (LINE), можно увидеть линию, рисующуюся на экране между точками, определенными двумя типами курсоров. Если сейчас нажать на клавишу «6», то появится следующая линия, нарисованная от места, занимаемого вспомогательным курсором, до точки, указанной основным курсором.

Сейчас мы находимся в режиме, называемом в программе ARTIST «PLOTPOINT» (черчение-точка). Это название высвечивается в правом конце меню под цифрами 7, 8 и 9. Нажимая на клавишу «м», можно изменить «PLOT-POINT» на «PLOT-MOVE» (черчение-движение), и теперь, если начать рисовать линии при помощи клавиши «6», перемещая курсор между линиями, увидим движущийся одновременно вспомогательный курсор. Так можно легко нарисовать параллельные линии, необходимые при создании одинаковых прямоугольных фигур, а также окружностей.

Вторичное нажатие на клавишу «M» переводит нас в режим «PLOT-TRACE» (черчение линий), в котором позиция вспомогательного курсора автоматически актуализируется при выполнении команды «LINE». Таким образом мы можем с легкостью обводить любые фигуры.

Установка вспомогательного курсора

Итак, вспомогательный курсор необходим при создании рисунков с использованием линий. Для того, чтобы определить его позицию на экране, нужно просто установить основной курсор на то место, где необходимо наличие вспомогательного курсора, и нажать на клавишу «SPACE». Однако не следует пытаться изменить позицию курсора, когда программа находится в режиме «PLOT-MOVE», это не получится, т.е. в данном режиме оба курсора перемещаются по экрану параллельно.

«CIRCLE» (окружность). «7».

Эта команда, как и следовало ожидать, позволяет рисовать окружности. При этом используется основной курсор, а также вспомогательный для определения радиуса окружности. Положение центра окружности детерминируется позицией, занимаемой вспомогательным курсором, основной же курсор находится в это время на периметре будущей окружности. Для того, чтобы проверить скорость и четкость выполнения этой команды, необходимо поместить вспомогательный курсор близко к центру экрана и одновременно нажать на клавиши «T» (перемещение вправо) и «7» (черчение окружности).

«BOX» (прямоугольник). «8».

В этом режиме оба курсора опять определяют форму и величину рисуемого прямоугольника. Для этого нужно установить курсоры в противоположных углах будущего прямоугольника и нажать клавишу «8», после чего на экране моментально будет изображен требуемый прямоугольник. Скорость процедуры может быть увеличена посредством одновременного нажатия на клавишу, определяющую движение курсора, и клавишу »8".

«ARC» (дуга). «9».

Эта команда позволяет рисовать дуги между двумя курсорами, для чего нужно нажать на клавишу «P» (для отказа от режима ARC, нужно нажать на клавишу «ENTER»). Для того, чтобы дуга рисовалась в одном направлении, необходимо использовать клавиши «U», «I» или «O»; противоположное направление определяется клавишами «L», «K» или «J». Клавиши «U» и «J» позволяют получить наиболее быстрый эффект.

«FILL» (заполнение). «0».

Это — один из наиболее привлекательных режимов программы ARTIST. Для того, чтобы получить соответствующее представление о возможностях этого режима, необходимо вернуться к первому меню, чтобы очистить экран, а затем опять перейти к режиму FILL; установить курсоры таким образом, чтобы в центре экрана они образовывали довольно большую окружность, поместить главный кур-

сор где-либо внутри изображенной окружности и нажать на клавишу «0», чтобы режим FILL начал работать. Мы увидим, что почти мгновенно нарисованная окружность закрасится черным цветом. В месте, занимаемом текстом меню, появится ассортимент узоров — тот самый комплект, что и при выборе узора кисти. Нажимая на любую клавишу в верхнем ряду (''1''—''0''), мы будем иметь возможность заполнить компонируемые фигуры соответствующим узором. Выходить из этого режима нужно с помощью остальных клавиш. Не следует забывать о возможности выбрать другой узор, созданный вами раньше, для чего нажать на клавишу «S» (так же, как и в случае выбора узора кисти) перед нажатием на цифровые клавиши в режиме выбора узора. Клавишу «S» можно нажимать многократно до выявления всех возможных ассортиментов узоров. Необходимо оговорить еще одно свойство режима FILL, позволяющее закрашивать очень сложные фигуры.

Ошибка! Команда UNDO

Что произойдет, если будет допущена ошибка, такая, например, как другой цвет заполнения фигуры, неровная линия и т.п.? В таких случаях используется программная команда UNDO (уничтожить сделанное). Ее сокращение (U=UNDO), находится с левой стороны меню. Другое, до сих пор не рассмотренное сокращение O, означает O'KEY (O'K).

Нажатие на клавишу «U» аннулирует все, что выполняется в настоящий момент. Аннулирование касается места предыдущей команды или же происходит до того момента, когда в последний раз была нажата клавиша «O», подтверждающая правильность выполнения операции. Из этого следует, что команда OK фактически показывает программе, до какого момента должны аннулироваться действия после команды UNDO.

Конечно, команда UNDO используется не только для простого удаления возможных ошибок, она просто бесценна во время экспериментирования с подбором цветов оттенков, форм и т.п.

Необходимо заметить, что после выхода из режима OVERLAY (см. ниже), аннулировать ничего нельзя, но зато можно убрать команду UNDO самой командой UNDO.

Нужно обратить внимание на автоматическое включение команды OK, после выполнения операции FILL, после выхода из режима TEXT, а также режима OVERLAY.

А что касается цвета? Меню палитры цветов

До сих пор мы рассматривали вопросы, касающиеся черно-белых графических рисунков. А как выглядит раскрашивание рисунка?

Следующее главное меню отведено так называемым атрибутам — то есть цветам, рассветлениям и пульсациям фрагментов экрана.

Вход в меню происходит после нажатия клавиши SYMBOL/SHIFT. Как обычно, атрибуты ограничиваются определением одного цвета «чернил» (INK), одного цвета «бумаги» (PAPER), включением или выключением пульсации (FLASH) и рассветлением поля знака.

Кроме того, программа ARTIST позволяет легко контролировать и управлять атрибутами экрана. Когда появится рассматриваемое меню, то мы увидим окошко с курсорами определяющими его форму и размер. Окно окружает поля знаков не совсем точно в указываемых курсором местах. Каждая попытка изменить атрибуты трактуется программой, как обращение к области окна. Для того, чтобы увидеть, каким образом это происходит, необходимо начертить довольно сложную фигуру на экране (заполнение окружности каким-либо узором — самый простой способ), а затем образовать окно, заходящее на область графики на экране. Нажимая на клавиши «1» и «2», можно изменять цвета — PAPER и INK соответственно. После выбора цветов появляется возможность проследить их влияние на содержание «окна» после нажатия на клавиши «6» или »7".

Клавиша «6» изменяет цвет бумаги-окна в избранном цвете; клавиша»7" изменяет цвет чернил-окна в выбранном цвете.

Возможен также выбор интенсивности цветов (BRIGHT или NORMAL), что происходит после нажатия на клавишу «3» — очередные изменения: нормальный цвет — более светлый.

Точно также происходит включение и выключение атрибута FLASH (пульсация) — посредством клавиши «4».

Для того, чтобы проследить влияние вышеуказанных операций на содержимое окна, можно использовать команды WBRI или WFLA (соответственно клавиши «8» и «9»).

С помощью клавиши «5» можно повлиять на цвет BORDER'а; нажатие на эту клавишу приведет к установке цвета рамки, соответствующего следующему из палитры цветов компьютера.

Нужно отметить, что рамка не является частью создаваемого пользователем графического изображения, и ее цвет может быть установлен на более позднем этапе программы.

В конце нужно обратить внимание на то, что установленные атрибуты в рассматриваемом режиме являются теми же самыми установками в основном меню, которые появляются после нажатия на клавиши «Z» или «SYMBOL SHIFT», контролирующие атрибуты.

Работа режима OVERLAY

Этот режим, без сомнения, представляет наиболее уникальные и всесторонние свойства программы ARTIST.

Необходимо вернуться ко второму основному меню, где при помощи клавиши «3» выбирается режим OVERLAY. В упрощенном виде работу этого режима можно представить как наложение на создаваемый рисунок своего рода »матрицы" — пластиковой накладки, с помощью которой можно отделять друг от друга, а затем «склеивать» различные фрагменты создаваемого на экране рисунка. Это делается следующим образом: сначала выбирается фрагмент рисунка, который нужно «вырезать» из экрана и который впоследствии может быть перенесен в другую область экрана и, если вы решите, что это — лучшее для него место, закреплен в этом новом месте. Но это еще не все, чего можно достичь с помощью режима «OVERLAY».

Для того, чтобы получить лучшее представление о возможностях этого режима, нужно изобразить на экране окружность и заполнить ее произвольным узором (перед этим необходимо очистить экран) и перейти к режиму OVERLAY, нажимая на клавишу «3».

Сразу же будет заметна инверсия цветов слова OVERLAY, после чего мы увидим, что на рисунок на экране как бы наложена некая пленка.

Перемещение курсора остается прежним, но, кроме того, его возможно передвигать так же, как и в режиме PLOT-TRACE и рисовать петлю вокруг части заполненной узорами окружности (или в каком-либо другом месте экрана). Необходимо убедиться в полном замыкании петли или, что лучше, дать команду CIRCLE. Затем нужно посмотреть, находится ли основной курсор внутри фигуры, наложенной на имеющийся на экране рисунок, после чего нажать на клавишу «0» (FILL — заполнение). Сразу же будет видно, как нарисованная окружность или фигура в режиме OVERLAY будет заполнена черным цветом, не оказывающим влияния на оригинальный рисунок на основном рабочем экране (если не считать эффекта подцвечивания, общего для вышеуказанных фигур).

После повторного нажатия на клавишу «3» (OVERLAY) увидим, что »зацепляющая" секция сотрется с экрана.

Сейчас вы можете: вырезать фигуру (CUT), скопировать ее (COPY) или же стереть (ABORT).

Выбор режима CUT приведет к удалению выбранной части рисунка из оригинала.

Режим COPY позволяет сделать копию выбранного фрагмента рисунка. На экране высветится новое меню:

1 2 3 4
INVERT MIRROR MIRROR-2 VIEW
5 6 7 8 9
USCAX DSCAX USCAY DSCAY PATT

INVERT.

Этот режим позволяет менять контрастность «вырезанного» из оригинального рисунка фрагмента. При использовании этого режима можно попробовать наложить вырезанную часть рисунка на основной экран.

MIRROR (зеркало).

Выполнение этой команды приведет к зеркальному варианту (лево-право) режима OVERLAY.

MIRROR-2.

Приведет к выполнению зеркального варианта (по верх-низ) режима OVERLAY.

VIEW (обзор).

С ее помощью вы можете рассмотреть эффект наложения вырезанного ранее фрагмента на графику основного экрана (пока без фактического присоединения). (для лиц с техническим складом ума это означает сложение функции XOR графики типа OVERLAY с графикой основного экрана и повторное проведение операции XOR, что означает возможность рассмотреть результаты наложения без фактического нарушения содержания экрана).

USCAX, DSCAX, USCAY, DSCAY.

Эти режимы позволяют увеличивать или уменьшать поверхность, вырезанную из основного рисунка, в плоскостях X или Y.

К примеру, нужно выбрать режим USCAX (клавиша «3»). Вы будете спрошены о степени изменения масштаба (0 — минимальный, 9 — максимальный). После нажатия, например »5", увидим,

что выбранная поверхность расширилась. Возвращение к начальной форме происходит после выбора режима DSCAY и повторного нажатия на клавишу «5». Перед тем, как приступать к изменению масштаба вверх, рекомендуется расположить объект или графику в верхнем левом углу экрана.

Нужно заметить, что уменьшение, произведенное непосредственно перед увеличением, может дать труднопредсказуемый результат.

PATT (сетка).

Эта команда влечет за собой наложение на экран сетки типа шахматной доски, примерно так же, как в случае режима основного экрана. При помощи этой команды можно также использовать режим OVERLAY для минимального, в одну или другую стороны, перемещения фрагментов рисунка, что способствует повышению эффективности раскрашивания.

После выбора месторасположения выбранной части рисунка, а также после выбора его размера, нажатием на клавишу ENTER вызываются следующие режимы:

1=XOR 2=OR 3=EXCLUSIV

Если выбрать режим, связанный с клавишей «1», то область накладываемой части будет »смешана" с рисунком основного экрана (он будет суммирован MODULO 2 с содержимым основного экрана). После нажатия клавиши «2», поверхности, заходящие друг на друга в режиме OVERLAY, будут расположены на экране, причем все то, что находилось внизу, сотрется. нажатие на клавишу »3" приведет к появлению на рабочем экране (оригинальном рисунке) внутренности вырезанного фрагмента — окружности, окружения или общей Поверхности.

Существует также возможность выбрать режим, создающий копии накладываемой графики (OVERLAY).

Создание знаков и знаков, определяемых пользователем (UDG).

Вернемся опять к основному меню, в котором цифра «8» обозначена сокращением »CHR". После нажатия на указанную клавишу, мы войдем в тот режим программы ARTIST, который связан с созданием знаков. В этом режиме существует возможность создавать новые комплекты знаков, их может быть даже 7 видов (комплект 0 содержится в ROM).

Сразу же после включения этого режима мы увидим на экране сетку, состоящую из 9 больших квадратов, каждый из которых разделен на 64 маленьких квадрата.

Каждый большой квадрат соответствует одному знаку, следовательно, мы можем одновременно работать над 9-ю знаками.

Рядом с нижним левым углом экрана мы увидим меню функций (FUNCTIONS), над которым находится комплект 9 полей знаков, представляющих собой натуральную величину увеличенных 9 знаков, расположенных в правой стороне экрана. Над этими квадратами, обозначенными «USR», находятся следующие 4 таких же квадрата, закрашенных поочередно зеленым и голубым цветами.

Это — цветные квадраты, каждый из которых содержит знак из выбранного комплекта (до момента ввода программы это будет 7-й комплект, содержащий знаки UDG. В действительности квадраты содержат 36 центральных знаков выбранного комплекта).

Чтобы рассмотреть комплект полностью, нужно нажать на клавишу «L» и тогда на экране высветится этот самый комплект знаков. Выбранный номер (здесь «7») указывается в верхнем левом углу экрана, а квадрат знака »USR" вновь появится с левой стороны. Знаки комплекта расположены в пронумерованных строках и колонках с целью более легкого к ним доступа.

Сейчас мы убедимся в том, что знаки, образующие эти 4 прямоугольника и состоящие из 9 знаков каждый на втором экране, здесь также обозначены зеленым и голубым цветами. Они занимают место от 3 ряда, 2 колонки до 6 ряда 7 колонки. Каждый из знаков, занимающих выше указанные позиции, будет высвечиваться на втором экране — полное описание этих функций приводится в дальнейшей части текста.

Вернемся ко второму экрану посредством нажатия клавиши «ENTER» и рассмотрим функции:

1-4 PRINT BLOCK.

Эта функция связана с клавишами 1-4 и в результате ее действия происходит перемещение избранного блока, состоящего из девяти знаков, из группы 4 таких блоков, находящихся в левой верхней части экрана, как в поле с сеткой (для редактирования и исправления), так и в поле USR. Нужно потренироваться клавишами 1-4 пока убедимся в результатах, к которым они приводят. Нужно заметить, что блок «1» является блоком, находящимся выше всех с левой стороны, блок »2" находится под блоком «1», блок «3» находится справа от блока «1», а блок «4» — под блоком «3».

SYMBOL SHIFT и (1-4) LOAD BL.

Чтобы осуществить эту функцию необходимо одновременно нажать клавиши SYMBOL SHIFT и одну из 1-4. Это приводит к обратному эффекту чем функция, рассматриваемая выше, то есть приводит к установке в блок 3*3 находящегося в верхней левой части экрана содержимого сеточного экрана. Пока не нужно экспериментировать с этой опцией — эти графики будут очень полезны для представления способа действия другой функции (см. ниже).

P MIRROR.

Эта команда дает зеркальное отражение каждого одинарного знакового поля.

G MIRROR SIX.

Эта команда приводит к образованию зеркального изображения только шести самых левых на сетке знаковых полей.

H MIRROR NINE.

Как легко догадаться, эта функция дает зеркальное изображение всех знаковых полей.

K TURN.

Используя эту функцию, можно вращать каждое знаковое поле вокруг его оси на 90 градусов.

CAPS SHIFT + 4 TURN FOUR.

Нажатие клавиши «CAPS», а затем клавиши «4», позволяет вращать только 4 верхние левые знаковые поля на сетке.

CAPS SHIFT + 9 TURN NINE.

При нажатии этих клавиш имеем возможность вращать все содержимое девяти знаковых полей 3*3.

I INVERT.

Как и можно было ожидать, эта функция приводит к инверсии содержания сетки, а также квадрата USR.

L CHARACTER SET.

Как уже упоминалось, эта функция способствует переходу к высвечиванию комплекта знаков, и кроме того, позволяет записывать знаки в комплекты или списывать их оттуда (см. ниже).

U UNDO.

Так же как и в основной работе программы, эта команда позволяет эффективно стирать последние проведенные пользователем операции. Она действует на пикселях, которые нанесены на сетке (при необходимости их можно стереть).

O OK

Приводит к установке содержания сетки в квадрат USR. Нужно отметить, что не все изменения, проводимые на экране автоматически, передаются в квадрат USR. Использование режима OK важно потому, что содержание квадратов USR является соединением эффектов вашей работы, проводимой на сетке, с содержанием различных комплектов знаков.

P CLS.

Этот режим стирает содержимое сетки. Если стирание произошло случайно, нужно не забывать об использовании команды UNDO, которая поможет вам вернуться к данному рисунку.

7 MOVE.

Эта чрезвычайно мощная команда, приводящая к переходу в другой режим. Нажимая на клавишу 7, мы удаляем пульсирующий курсор, взамен чего получаем возможность перенести содержание сетки по всем направлениям с шагом в 1 пиксель. Нажимая на клавишу ENTER выключаем этот режим.

8 ANIMATE FOUR.

Посредством нажатия на клавишу 8 мы переписываем поочередно 4 блока знаков, находящихся в левой верхней части экрана, в квадрат USR (пользователя). Если нажать на клавишу 8 сейчас, то она приведет в действие силуэт человека, стреляющего из пистолета. Для остановки мультипликации нужно нажать клавишу ENTER, не удивляясь тому, что остановка происходит не сразу — программа ждет соответствующего момента.

9 ANIMATE SIX.

Приводит к мультипликации шести групп блоков размерами 3*3: четырех самых левых в верхней части экрана и двух следующих блоков девяти знаков выбранного комплекта. При работе с комплектом знаков N 7 пользоваться этим режимом не нужно.

Образование знаков

Работу нужно начать с очистки экрана (клавиша P). Управление движением курсора на фоне сетки осуществляется так же, как и управление основным курсором в остальных режимах программы, т.е. клавиши Q, S, R и T перемещают курсор соответственно вверх, вниз, налево и направо.

Клавиши X и C используются как обычно. Кроме того, существует еще возможность использования режима, связанного с клавишей Z, который приводит к стиранию восьми пикселей сразу, а также к стиранию всего ряда пикселей внутри данного знакового поля.

Запись и ввод знаков в (из) комплект знаков.

Существует возможность переопределить семь различных комплектов знаков во время работы с данной программой. То, что подлежит занесению в комплект знаков, находится в поле пользователя (USR).

Чтобы увидеть результаты записи созданного пользователем знака, нужно сначала нарисовать один из них — простая форма знака будет хорошим примером; затем выбрать фронт работы — «комплекты знаков» с помощью клавиши «L». Вы увидите, что созданный знак появился и на этом экране в поле пользователя (USR).

Чтобы приступить к записи этого знака (знаков), нужно выбрать режим, связанный с клавишей S, в результате чего вам будут заданы следующие вопросы:

STARTING AT WHITCH LINE? (0-9)
(с которой линии начинать?)
STARTING AT WHTCH COLOMN?(0-9)
(с которой колонки начинать?)
HOW MANY CHARACTERS?(0-9)
(сколько знаков?)

На последний вопрос можно ответить, например, нажав на клавишу «9», чтобы получить запись содержания всего поля пользователя (USR). После введения выше указанных параметров все содержание поля пользователя будет расположено в первых девяти знаковых полях комплекта знаков. Кроме того, поочередно, ряд за рядом, с поля пользователя (USR) происходит трансформация знаков.

Ввод знаков в поле пользователя (USR). Эта операция так же проста, как и операция записи.

Мы решили ввести силуэт стреляющего человечка в поле пользователя. Для этого при помощи клавиши «L» необходимо перейти в режим LOAD, в результате чего вы будете спрошены о параметрах (так же, как во время записи знаков — SAVE).

На первый вопрос ответим «3» (для 3-го ряда), а на последний вопрос нажмем на клавишу »9" (для пересылки всех девяти знаковых полей в поле пользователя, начиная с позиции 3, 2). После введения выше указанных параметров мы увидим фигуру стреляющего человечка в поле пользователя (учитывая, что до настоящего времени форма рассматриваемой графики не изменялась).

Изменение комплекта знаков

Происходит просто. В результате нажатия на клавишу «C» вы будете спрошены о номере комплекта знаков, который должен заменить настоящий комплект знаков. Нужно отметить, что выбор нового комплекта знаков не изменяет содержания поля USR.

Единственным комплектом, содержание которого не может быть изменено — это комплект № 0.

Существует, однако, возможность использовать знаки этого комплекта (это — стандартный комплект, расположенный в ROM).

Можно также заметить, что во время высвечивания каждого комплекта стандартный комплект показывается с целью указания его места внутри комплекта каждого из знаков.

Программа ARTIST позволяет также вводить (LOAD) графику в поле пользователя (USR) из области основного экрана, образованного там в режиме DRAWING-PAINTING (рисование/раскрашивание). Это происходит после нажатия на клавишу «K», после чего сразу же высвечивается основной рабочий экран программы вместе с зеленым прямоугольным полем, размеры которого соответствуют полю пользователя. Существует возможность перемещать этот прямоугольник по рабочему экрану с помощью обычных курсорных клавиш. Как только зеленый прямоугольник окажется над рисунком, его движение нужно перехватить с помощью клавиши «X». Звуковой сигнал подтвердит удачную трансферизацию рисунка. Если сейчас нажать на клавишу ENTER, то вы вернетесь к экрану, связанному со знаковым режимом, что позволяет проверить, попал ли выбранный фрагмент в поле пользователя (USR).

Теперь с этой графикой можно поступать так же, как с любым произвольным комплектом 9 знаков, подлежащим модификации (посредством возвращения к режиму делительной сетки), а также записать в произвольный комплект знаков.

При помощи данной команды можно также располагать знаки в области основного экрана. Если произойдет ввод нескольких знаков в поле пользователя (при использовании функции «L» LOAD), то после повторного нажатия на клавишу «K» вы вернетесь в область основного экрана. Здесь, после перемещения зеленого поля на выбранную позицию, нужно нажать на клавишу «C», что приведет к «вписыванию» этих знаков на нужное место.

Замечания

Нужно отметить, что «узоры» для заполнения (FILL) из комплекта знаков № 5, а также малые знаки из комплекта N 6 используются самой программой ARTIST. Если этого не запомнить, то могут возникнуть недоразумения в случае их переопределения.

Создание собственных узоров

«Заполнение» рисунков можно осуществить после работы в режиме определения знаков и последующего ввода их в комплект знаков N 5. Можно также увидеть, что в данном комплекте «запоминаются» узоры кисти толщиной от позиции 2,3 до 3,1.

«Заполняющие» узоры занимают место от позиции 3, 3 вверх. Кроме этого, существует возможность определять формы курсоров, которые также сохраняются в комплекте знаков № 5.

Компрессор экрана

После программы ARTIST на кассете находится программа, позволяющая произвести компрессию созданного на экране рисунка. В основном компрессия производится от 1/3 до 1/2 нормальной величины экрана (что для полностью заполненного экрана означает занятость свыше 6000 байтов памяти). Существует возможность производить компрессию нескольких экранов и даже компрессию отдельных третей экранов. Кроме того, программа запоминает эти экраны (или их 1/3 или 1/2), давая каждой части порядковый номер, по которому их можно «вызвать».

Программа имеет свои собственные меню, позволяющее вводить экран, рассматривать рисунок, удалять последний введенный экран, а также сохранять готовый блок экранов.

Если вы хотите только произвести компрессию 1/3 или 2/3 экрана, нужно нажать на клавишу «C», на экране появятся вопросы:

FROM THIRDT? (от которого места?)

И затем:

TO? (до которого места?)

Вам нужно сообщить, которая именно часть экрана вас интересует. Если первая треть — нажмите два раза «1». Точно также нужно нажать на клавишу »2", если компрессия должна касаться только центральной части экрана. Чтобы ввести верхние 2/3 экрана, нужно сначала нажать «1», а потом »2" и т.д.

Необходимо заметить, что компрессия производится автоматически после загрузки рисунка с магнитофонной ленты, поэтому спецификацию компрессии нужно произвести перед загрузкой. Программа сообщит о количестве оставшейся свободной памяти.

Для использования в работе подвергшегося компрессии блока рисунков, необходимо сначала записать его на носитель, с которого потом можно производить загрузку. Если вы решили загрузить какой-либо рисунок или его часть, необходимо вписать номер экрана в ячейку памяти с адресом на 2 позиции большим, чем адрес, под которым был введен рисунок. Например, экран после компрессии был введен в память, начиная с адреса 50000; тогда нижеприведенная программа позволит его повторно вызвать:

```
10 CLEAR 49999
20 LOAD "" CODE 50000
30 INPUT "PICTURE:";X
40 POKE 50002;X
50 RANDOMIZE USR 50000
60 PAUSE 0:GOTO 30
```

Наличие последней строки в программе обусловлено вводом комплекта рисунков или же вводом одного рисунка. Команда PAUSE служит для высвечивания полного экрана без выдачи сообщения «OK» до момента нажатия на клавишу.

Использование собственных комплектов знаков

В описании программы вы видели, что она позволяет легко создавать собственные комплекты знаков или, иначе, создавать знаки, определяемые пользователем (как указывается в описании бейсика).

Но что нужно делать для их использования?

Для использования комплекта знаков необходимо переопределить переменную, называемую «CHARS» и расположенную в памяти системы в области, отведенной на так называемые системные переменные. Это легко сделать, используя программу:

```
10 LET X=64000
20 CLEAR X-1
30 LOAD "" CODE X
40 POKE 23606,X-256*INT(X/256)
50 POKE 23607,INT (X/256)-1
```

Если вы хотите расположить комплект знаков не с адреса 64000, а в другом месте памяти, то нужно только изменить значение переменной «X» в 10 строке программы.

Указания, касающиеся использования программы ARTIST

Без сомнения, нужно согласиться с тем, что программа «ARTIST» является программой с действительно большими возможностями. Как можно себе представить, часто существует много путей для достижения одного и того же результата, а также большое количество трюков и способов, которые можно получить при создании различных эффектов.

Эллипсы.

Есть возможность чертить эллипсы с помощью программы ARTIST. Для этого необходимо сначала изобразить окружность на рабочем экране нормальной величины, потом перейти в режим OVERLAY и, не изменяя положения обоих курсоров, опять нарисовать эту же окружность.

7.2. ARTSTUDIO

Как пользоваться этим руководством

Это руководство разработано таким образом, чтобы охватить и программу Артстудия и программу расширенная Артстудия (EXTEND ARTSTUDIO). Первая разработана для работы с магнитофоном, а вторая — для работы с микродрайвом или дисковой системой и имеет расширенный набор команд, который приведен в приложении 1.

Прежде, чем пользоваться программой, надо выполнить персональную копию, настроив ее так, чтобы она соответствовала вашему аппаратному обеспечению, см. приложение 4.

Затем необходимо изучить введение. Почувствуйте программу, накопите навык, не бойтесь экспериментировать. После этого дочитайте руководство до конца, освоив более подробно узкие вопросы.

Предполагается, что начать работу с программой можно до окончания изучения данного руководства.

Введение

В основу концепции программы положен базовый принцип, согласно которому программа должна быть простой для человека, впервые приступившего к работе с ней. Для этого вся необходимая информация должна быть перед пользователем и не должно быть необходимости ввода сложных последовательных команд. Вместо выдачи команды, надо просто указать на требуемую опцию в меню.

После исходного запуска А.С., вы увидите чистый экран с бело-голубым меню вверху и черной стрелкой в центре (это курсор). Курсор — ключ к работе программы. Он управляется клавиатурой, джойстиком или мышью, в зависимости от того, какой выбор вы сделали при создании персональной копии. При длительном нажатии клавиши движение курсора ускоряется.

Меню содержит набор команд. Выбор команды из набора осуществляется перемещением курсора с последующим нажатием клавиши: для джойстика это «огонь», а для клавиатуры — специально назначенная, при выполнении персональной копии клавиша — назовем ее «выбор». После этого вытягивается очередное меню. Оно может содержать новый набор опций, которые вызывают очередное меню или выполняют определенные команды. Если опция является командой то «вытянутое» меню исчезает, и вместо курсора появляется символ, который, во-первых, отражает режим работы программы и, во-вторых, является курсором. Действие клавиши «выбор» здесь зависит от вида символа. Так, например, символ «кисть» означает рисование кистью и кнопка «выбор» запрашивает пиксели на экране. Если ввести курсор в меню, то он превратится в стрелку. Для большинства команд можно сделать «скроллинг» экрана, если ввести курсор в прямоугольник с изображенной в нем вертикальной стрелкой.

Итак, если при движении курсора по меню, тот или иной прямоугольник высвечивается, то в этот момент он доступен для выбора. В некоторых меню есть опции, которые высвечиваются только при выполнении определенных дополнительных условий. Рассмотрим, например, опцию «окна» (WINDOWS). Естественно, очистить окно нельзя, если оно не задано, то есть не определено (DEFINE WINDOW). Поэтому, при движении курсора по меню «окна» в первый раз, прямоугольник DEFINE WINDOW может загореться, CLEAR WINDOW (очистить окно) — нет. После загорания окна, если вы снова обратитесь к меню WINDOW во второй раз, то выбор CLEAR WINDOW будет возможен.

Некоторые меню содержат опции, которые не являются командами, но в то же время и не «вытягивают» очередные меню. Это «флаги» или переключатели, они могут быть в двух состояниях — включено (ON) или выключено (OFF). Они обычно изменяют состояние других опций меню. Состояние «вкл» обозначается галочкой, а «выкл» — крестиком.

Часто в меню бывает тройной выбор. Например, есть три размера символов при работе с текстовым меню (TEXT). эти три размера — NORMAL HEIGHT (нормальный), DOUBLE HEIGHT (двойной высоты) или TREBLE HEIGHT (тройной высоты).

Полный экран Спектрума содержит 24 строки, но меню уже занимает три из них. Можно подумать, что осталась 21 строка, но если ввести курсор в прямоугольник с вертикальной стрелкой, то можно выполнить «скроллинг» и увеличить доступное поле экрана. «Щелкнув» три раза, вы получите полный размер. Маленький белый прямоугольник слева от прямоугольника со стрелкой показывает, какая часть полного экрана сейчас используется. Чтобы убрать меню, вызванное по ошибке, выведите курсор за пределы меню и нажмите «выбор».

Остальная часть руководства разбита на разделы, при этом каждый раздел посвящен одной из опций главного меню. Разделы следуют в том же порядке, что и окна в меню — слева направо, сверху вниз.

Некоторые термины

Пиксель — это точка на экране, мельчайший элемент графики. Пиксель может быть либо включен, либо выключен. Включенный пиксель может быть цветным, при этом он имеет цвет INK, назначенный для своего блока. Надо помнить, что для одного блока разрешены только два цвета.

Печать (PRINT)

Это меню позволяет вам сделать копию законченной картины на точечном принтере. Возможны пять способов получения распечатки, они записаны в меню как 1x1, 2x2, 3x3, 4x4, 5x5. Эти числа показывают, сколько точек печати соответствуют одному пикселю на экране.

Не все из этих режимов возможны с любым принтером. Это зависит от полного количества точек, которое может вывести принтер по ширине страницы. Заметьте, что информация о цвете на распечатку не выводится, а выводится информация о том, включен данный пиксель или нет.

Надо иметь в виду, что возможно некоторое несоответствие между тем, что есть на экране и тем, что будет на принтере. Например, если на экране цвета INK и PAPER — белые, то ничего не видно. В то же время, на распечатке включенные пиксели будут видны.

Другая опция — GREY SCALE (шкала серого). Здесь используются совокупности точек разной плотности, так можно представить на принтере цвета Спектрума. Размеры печати в этом режиме эквивалентны 3x3.

Выдачу на принтер можно прервать в любое время нажатием CAPS SHIFT + SPACE. Имейте в виду, что ваш принтер может после этого нуждаться в «сбросе»: выключите и включите его снова.

Большинство точечных принтеров могут работать в двух режимах по плотности печати: обычная — SINGLE DENSITY и двойная — DOUBLE DENSITY. Разрешающая способность в последнем случае выше. Некоторые из режимов печати, невозможны в двойной плотности, т.к. количество точек в строке здесь больше. Выбор того или иного режима по плотности осуществляется через меню печати. Обратите внимание на то, что в режиме двойной плотности возможно некоторое растягивание изображения по вертикали.

Можно также применять выгрузку на принтер изображения боком. Так как экран имеет ширину большую, чем высоту, то это может позволить применить недоступные ранее режимы печати.

Опции LEFT JUSTIFY (выравнять по левому полю), CENTRE (установить по центру) и RIGHT JUSTIFY (выравнять по правому полю) — определяют, в каком месте страницы располагается текст.

Если ваш принтер автоматически переводит строку после возврата каретки, то переключатель LINE FEED (перевод строки) надо отключить.

Опция ZX/ALPHACOM назначается, если печать выполняется на ZX-принтере или на принтере ALPHACOM, которые имеют 32 колонки.

Файл (FILE)

«Артстудия» может загружать (LOAD), выгружать (SAVE), проверять (VERIFY), и сливать (MERGE) файлы записанные на ленту.

Чтобы записать файл на ленту, надо вызвать меню FILE и выбрать в нем опцию SAVE FILE. Появится запрос имени файла. Наберите имя (до 10 символов) и нажмите ENTER. Если дать ENTER без имени, то это отбой команды SAVE. После записи A.C. возвратиться к главному меню. Рекомендуем после SAVE давать VERIFY. Если все в порядке, то после VERIFY программа автоматически переходит к главному меню.

Заметьте, что A.C. загружает байты, записанные как CODE или SCREEN$, т.е. загружает файлы размером 6.75K. Запись или загрузка могут быть прерваны: CAPS SHIFT + SPACE.

Файл, записанный на ленту может быть слит с содержимым экрана. Если переключатель OVER выключен (см. меню атрибутов), то итог получится по принципу OR (или). Если же переключатель OVER выключен, то слияние идет по принципу XOR (исключающее или). Цвета слиянию не подвергаются, а берутся с исходного экрана.

Обратите внимание на то, что буферный блок памяти во время загрузки и слияния занят, поэтому операция UNDO (отбой) недоступна.

Атрибуты

Это меню управляет включением цвета и других атрибутов. Эти атрибуты применимы ко всем рисункам, чертежам и т.д.

Например, установим цвет красителя INK зеленым. Введите курсор в опцию меню, обозначенную SET INK, затем нажмите «выбор». «Вытягивается» новое меню, содержащее 8 цветных прямоугольников и один, обозначенный буквой «Т».

Введите кусор в зеленый прямоугольник и нажмите «ввод». Теперь все, что нарисовано на экране будет иметь зеленый цвет INK. Эта операция имеет тот же результат, что и операция INK 4 в Бейсике.

Квадрат с буквой «Т» означает TRANSPERET (прозрачный). Атрибут, отмеченный «Т» при рисовании не устанавливается, а остается тем же, каким он и был на этом месте до вашей операции.

Цвет бордюра можно изменять из меню атрибутов путем введения курсора в опцию SET BORDER и нажатием «ввод». Для бордюра нет понятия «прозрачности» («Т»).

Из этого же меню можно управлять яркостью BRIGHT и миганием FLASH. Они могут быть включены (ON) или выключены (OFF) или установлены в «Т».

Переключатели OVER и INVERSE играют ту же роль, что и их бейсиковские аналоги.

Опция STANDART — это возвращение всех атрибутов в исходное первоначальное состояние.

Рисование (PAINT)

Это меню предоставляет вам три основных инструмента: перо, распылитель и кисть. Все они рисуют установленными INK и PAPER, при этом учитываются текущие атрибуты BRIGHT и FLASH. Выполняется также INVERSE, а OVER 0 нет.

Чтобы выбрать перо, надо ввести курсор в опцию PEN и нажать «ввод». Появится набор из 16 перьев. Отметьте курсором желаемое перо и нажмите «ввод». Вместо стрелочного курсора появится символ пера. Его можно перемещать по экрану, включая при этом пиксели клавиший «ввод». Если « ввод» нажат непрерывно, то при этом вычерчивается сплошная линия.

Если при этом включен режим INVERSE, то состояние пикселей, по которым проходит перо, меняется на противоположное. Так можно эффективно выполнить стирание.

Чтобы выбрать распылитель, введите курсор в прямоугольник SPRAY. Вы можете выбрать распылитель из 8 различного диаметра. Каждый из них распыляет случайный набор точек на экране, при этом распылитель можно перемещать.

Введя курсор в опцию BRUSH (кисть), вы можете выбрать одну из 16 возможныхкистей.

Каждая кисть имеет связанную с ней маску, которая имеет те же размеры, что и кисть. Когда выполняется рисование кистью на экране, то сначала переключаются пиксели, соответствующие маске, а затем включаются пиксели, соответствующие самой кисти или наоборот, если включен режим INVERSE. Преимущество использования маски состоит в том, что с ее помощью могут быть получены более сложные рисунки, но для рисования простых рисунков кистью об этом можно пока не думать.

Первая кисть в наборе — это нулевая кисть, т.к. ею можно водить по экрану, не меняя цветовую гамму информации без включения или отключения пикселей. Может быть, вы сочтете удобным для себя такой подход, когда сначала изображается рисунок на экране черно-белым и только в конце закрашивается.

Каждую из 16 кистей можно переработать (EDIT) в соответствии с конкретными потребностями. Для этого выберите курсором окно в меню EDIT BRUSH, нажмите «ввод». Тем самым вы «вытянете» новое меню, содержащее увеличенное изображение кисти и маски, а также копию кисти в нормальном размере. Теперь в изображении кисти и маски вы можете переключать отдельные пиксели. Чтобы начать рисовать полученной таким образом кистью, введите курсор в нормальное изображение кисти и нажмите «ввод».

Режим EDIT BRUSH действует только на текущую или на последнюю использовавшуюся кисть.

Чтобы переделать произвольную кисть, вам надо сделать ее текущей, выбрав из меню SELECT BRUSH, а затем вернуться в меню PAINT и выбрать опцию EDIT BRUSH.

Другие возможности (MISCELLANEOUS)

Первая опция в этом меню устраняет меню с экрана и дает вам возможность рассмотреть все 24 строки одновременно. Возврат в меню осуществляется клавишей «ввод». При этом режим UNDO (отбой), см. ниже, не работает. Если же он вам нужен, то тогда пользуйтесь для просмотра всего экрана «скроллингом» путем введения курсора в прямоугольник с вертикальной стрелкой и нажатием «ввод».

Вторая опция этого меню очищает весь экран и устанавливает все атрибуты в исходное состояние. Режим UNDO возможен, если вы стерли изображение по ошибке.

Три последующие опции связаны с возможностью применения вспомогательных сеток из светлых и темных блоков. Они полезны, когда вы размещаете элементы рисунка на экране так, чтобы избежать «наползания» одних цветов на другие.

Режим BRIGHT GRID 1 устанавливает сетку на экране.

Режим BRIGHT GRID 2 устанавливает на экране сетку двойного размера.

Режим BRIGHT GRID удаляет сетку.

Следующая опция этого меню — CHANGE COLOR (изменить цвет). Она работает только с окнами, которые были заранее заданы (см. раздел окна). Функция ее — превратить один цвет в другой на всем экране или на его части. Эти цвета задаются через меню атрибутов. Исходный цвет задается как PAPER, а требуемый — как INK. Данная функция изменяет в пределах заданного окна все исходные цвета на требуемые.

Последняя опция меню MISCELLANEOUS изображает сообщение с номером той версии программы A.C., которой вы пользуетесь.

Отбой команды (UNDO)

Это один из наиболее употребляемых режимов программы. Когда задается одна из команд, например, FILL, содержание экрана копируется в специальный буферный блок памяти, а затем на экране выполняется поданная команда. Если результат не тот, который нужен, то команда UNDO возвращает содержимое буфера на главный экран. Эта команда отбивает действие только последней команды.

Окно (WINDOW)

Окно — это прямоугольник на экране, очерченный линией. Все, что находится в нем, воспринимается как одно целое.

Прежде, чем какие-либо действия будут выполнены над содержимым окна, оно должно быть задано. Чтобы это сделать, надо вызвать опцию WINDOW и выбрать во вновь полученном меню опцию DEFINE WINDOW. Вместо стрелки в качестве курсора появится маленький квадрат. Установите его на экране и нажмите «ввод» в двух точках — противоположных углах окна.

Окно размером более чем 21 строка задается так:

— задайте один из верхних углов;

— выполните «скроллинг», (см. выше);

— задайте противоположный нижний угол окна.

Опция WHOLE WINDOW определяет весь экран в качестве окна

Опция LAST WINDOW задает последнее ранее использовавшееся окно.

Опция CLEAR WINDOW (очистить окно) выключает все пиксели в пределах окна и устанавливает атрибуты, равные текущим значениям INK, PAPER, FLASH и BRIGHT.

Окно можно скопировать опцией CUT & PASTE (вырезать и наклеить). Появляется новое окно, которое можно установить в нужном месте, после чего нажатием клавиши «ввод» выполняется копирование содержимого старого окна в новое.

Опция CUT, CLEAR & PASTE (вырезать, очистить и наклеить) похожа на предыдущую. Добавляется очистка старого окна после копирования в новое.

Можно получить многочисленные копии окон. Для этого надо включить переключатель MULTIPLE перед тем, как вызвать команды CUT & PASTE и CUT, CLEAR & PASTE.

Обычно новое окно после «наклеивания» забивает содержимое основного экрана, однако есть возможность слить содержимое окна с содержимым экрана. Для этого должен быть включен переключатель MERGE. Если в это время в меню атрибутов переключатель OVER выключен (OFF), то слияние происходит по принципу OR (или), а если OVER включен (ON), то по принципу XOR (исключающее или). Атрибуты не сливаются, а берутся те, которые были на основном экране.

Выбор опции INVERT WINDOW изменяет состояние всех пикселей на противоположное, но не изменяет атрибуты.

Окно можно увеличить, уменьшить, сжать, растянуть, в общем, изменить его размер (RESCALE). Это выполняется через RESCALE WINDOW. После этого надо задать новое окно. Содержимое исходного окна проецируется на новое, причем так, чтобы его размеры соответствовали размерам нового окна.

Процесс изменения размера выполняется в две стадии. Сначала изменяется размер по вертикали, при этом содержимое копируется в буферную память, а затем изменяется размер по горизонтали и окно изображается на экране. Первая стадия выполняется с некоторым запозданием.

Имейте в виду, что может быть потеря мелких деталей при уменьшении размеров окна. Возможно также искажение цветов.

Опция CLEAR & RESCALE очищает исходное окно при изменении его размеров. Режим MERGE и MULTIPLE применяется здесь так же, как и с опцией CUT & PASTE.

Можно получить зеркальные изображения окна относительно горизонтальной и вертикальной осей — FLIP HORIZONTAL или FLIP VERTICAL.

И, наконец, окна можно вращать на 90, 180, 270 градусов по часовой стрелке, например — ROTATE 90.

Заполнение (FILL)

Заполнение — это метод быстрого включения пикселей внутри графического обьекта на экране. Заполняемый обьект должен быть замкнутым по контуру. Он можен быть заполнен гладким цветным фоном или рисунком (текстурой). Вам предлагается обширный выбор готовых текстур, а также возможность создания собственной текстуры.

Чтобы заполнить об'ект одним цветом, поместите курсор в бокс SOLID FILL и нажмите «ввод». Стрелочный курсор заменится изображением красящего валика. Введите его внутрь заполняемаого об'екта и нажмите «ввод».

Чтобы заполнить об'ект текстурой, вызовите бокс TEXTURE FILL (меню FILL), тогда получите новое меню, содержащее 32 готовых текстуры. Вы можете наложить текстуру на уже готовый гладкий цветовой фон внутри об'екта.

Все заполнения выполняются текущими значениями INK и PAPER с учетом установленных значений FLASH и BRIGHT. Режим INVERSE не работает при заполнении гладким цветом, но работает при заполнении текстурой. Режим OVER не работает вообще.

Имейте в виду, что команда FILL распространяется на весь экран, в том числе и на те строки внизу, которых вы не видите. Прервать заполнение можно одновременным нажатием CAPS SHIFT и SPASE. Если заполнитель просачивается через контур рисунка или результат не устраивает вас по какой-либо причине, вы можете ликвидировать его по команде UNDO.

Текстура в конце первого ряда — это нуль текстура. Она полезна для установки атрибутов в какой-либо области экрана без изменения состояния пикселей.

Алгоритм заполнения основывается на использовании внутреннего стека, который хранит строчные сегменты заполняемого об'екта. Этот стек имеет примерно 6К памяти, но тем не менее возможен случай, когда очень сложная форма об'екта может привести к переполнению памяти. Если это произойдет, А. С. не выйдет из строя, а оставит на экране незаполненные участки. Однако такой случай маловероятен.

Опция WASH TEXTURE (намывка текстуры) позволяет вам рисовать непосредственно одной из выбранных текстур. В этом режиме доступны те же 32 текстуры, что и раньше.

Каждую из 32 текстур можно изменить — подредактировать (EDIT). Для этого в меню FILL вызовите бокс EDIT TEXTURE и нажмите «ввод». Вы получите увеличенное изображение избранной текстуры и рядом образец в нормальном размере. После этого на увеличенном изображении вы можете переключить отдельные пиксели. Чтобы использовать полученную таким образом текстуру, введите курсор в образец нормаль ного размера и нажмите «ввод».

Режим EDIT TEXTURE работает только с текущей или последней использовавшейся текстурой. Чтобы переработать произвольную текстуру, вам надо сначала сделать ее текущей.

Увеличение (MAGNIFY)

А. С. позволяет увеличить в размере отдельные зоны экрана для детального их рассмотрения. Существуют 3 степени увеличения.

Чтобы увеличить часть экрана, вызовите меню MAGNIFY, а в нем выберите один из режимов (MAG x 2), (MAG x 4), (MAG x 8).

При этом курсор превратится в изображение увеличенного стекла. Переведите его в ту часть экрана, которую хотите увеличить. Затем вновь нажмите «ввод». Основной экран заменится увеличенным изображением части вашего рисунка.

При увеличении в 8 раз каждый отдельный пиксель в увеличенном изображении может быть включен, выключен или переключен. Делается это так:

— курсор устанавливается на изображение пикселя;

— нажатием «ввод» над ним выполняется операция.

Характер этой операции задается вызовом соответствующего бокса:

SET — включение;

RESET — выключение;

TOGGLE — переключение.

Пиксели включаются и выключаются с учетом текущих значений INK и PAPER и установок FLASH и BRIGHT. Атрибуты INVERSE и OVER здесь не применимы.

На увеличенном изображении можно выполнять «скроллинг». Размер увеличения можно поменять в любое время. Возврат в главное меню осуществляется через бокс MENU.

Режим UNDO не работает на увеличенном изображении, т.к. буферная память занята экраном.

В режиме MAG X 8 возможно получение решетки. Один квадрат решетки соответствует одному пикселю основного экрана.

Текст (TEXT)

Этим меню управляется изображение символов: букв, цифр и знаков препинания. Текст можно вводить в двух направлениях слева на право и сверху вниз. Отдельные символы могут изображаться в одном из 9 доступных размеров. Все символы печатаются в соответствии с заданными цветами INK, PAPER, с учетом текущих значний BRIGHT, FLUSH, INVERSE, OVER.

Чтобы ввести текст, поместите курсор в бокс LEFT TO BRIGHT (слева направо) или в бокс DOWNWORDS (сверху вниз) и нажмите «ввод». Курсор опять изменится. Вводите текст с клавиатуры. Если допустили ошибку, устраните ее (CAPS SHIFT + 0). Нажатие ENTER означает конец ввода текста. Знаки препинания вводятся соответствующими клавишами с нажатой клавишей SYMBOL SHIFT.

Возможны три размера букв по высоте и три по ширине:

NORMAL WIDTH — обычная ширина;

DOUBLE WIDTH — двойная ширина;

TREBLE WIDTH — тройная ширина;

NORMAL HEIGHT — обычная высота;

DOUBLE HEIGHT — двойная высота;

TREBLE HEIGHT — тройная высота.

Возможны 9 комбинированных режимов. Размер каждого из них на экране индицируется размером курсора «I».

Символы могут изображаться как общепринято, так и боком, например при выполнении надписей на диаграммах. Для этого в меню TEXT существует бокс SIDEWAYS. Выбор этот надо сделать до выбора LEFT TO RIGHT или DOWNWORDS.

Текст можно выделить напечатанием каждого символа дважды, причем со смещением на один пиксель. Для этого служит бокс BOLD.

Последняя опция в меню TEXT — это редактирование шрифта.

Редактирование шрифта (FONT EDITOR)

Это меню позволяет редактировать набор символов. Всего в наборе 96 знаков (коды ASCII от 32 до 127). С помощью этого меню можно как создавать новые шрифты, так и изменять существующие. Наборы можно хранить на ленте и загружать по мере необходимости.

96 символов из набора изображаются в нижней части экрана. Каждый имеет размер 8x8 пикселей. Один из этих символов текущий.

Выше на экране изображены в увеличенном масштабе три символа: текущий, а также находящиеся справа и слева от него. В этих трех изображениях можно с помощью курсора переустанавливать отдельные пиксели.

Любой символ может быть сделан текущим путем указания на него курсором и нажатием «ввод». Текущий символ можно поменять на любой из набора, если ввести курсор в бокс со стрелкой и нажать «ввод».

Над текущим символом можно выполнить ряд операций, если вызвать меню из бокса CHARACTER. Эти операции:

— CLEAR CHARACTER — стереть символ;

— INVERT CHARACTER — переключить пиксели на противоположные;

— FLIP VERTICAL и FLIP HORIZONTAL — построение зеркального отражения относительно горизонтальной и вертикальной осей;

— ROTATE 1/4 — повернуть на 90 градусов.

Эти же самые операции можно выполнять над всем набором в целом, если вызвать меню из бокса FONT. Опция CLEAR (очистить) в этом случае требует подтверждения.

Набор символов, содержащихся в ПЗУ «Спектрума», может быть введен с использованием опции COPY ROM в меню MISCELLANEOUS

В этом же меню есть опция CAPTURE FONT. Ее задача — копировать блоки из заданного окна в набор символов. Она работает только если окно задано. Копирование начинается с текущего символа и продолжается слева направо и сверху вниз по всему окну до тех пор, пока не используются всепиксели.

Набор символов может быть выгружен на ленту и загружен с нее. Режим этот очень похож на работу с файлами, но функция слияния (MERGE) файлов здесь невозможна. Набор содержит 768 байтов, т.е. 96x8, поэтому загрузить файл большого размера в набор нельзя, хотя меньше — можно. Выбор SAVE или LOAD выполняется после вызова бокса FILE.

По окончании редактирования возврат в главное меню выполняется через блок MENU.

Набор символов, который вы создадите, используя этот режим, вы можете использовать в собственных программах после следующей процедуры:

```
10 LET X=64000 (это значение выберете сами)
20 CLEAR X-1 (защита набора символов)
30 LOAD"FILENAME" CODE X
40 POKE 23607,INT(X/256) (установка системных
50 POKE 23606,X-256*INT(X/256) переменных)
```

Для тех, кто работает с микродрайвом, строка 30 будет выглядеть так:

```
30 LOAD * "M";1;"FILENAME" CODE X
```

Для тех, кто работает с дисковой системой:

```
30 PRINT 4, LOAD"FILENAME" CODE X
```

Графические элементы (SHAPES)

Это меню служит для изображения на экране различных графических элементов. Это: точки, линии, прямоугольники, треугольники, окружности, лучи. Эти геометрические фигуры изображаются путем перемещения курсора по экрану с нажатием клавиши «ввод». Все фигуры изображаются в текущих атрибутах INK и PAPER, с учетом текущих значений FLASH и BRIGHT. INVERSE и OVER также справедливы.

Например, для изображения треугольника вызовите меню SHAPES, затем вызовите опцию TRIANGLE. Установите курсор в вершинах будущего треугольника и трижды нажмите «ввод».

При изображении окружности задаются две точки. Одна центр и другая любая точка на окружности. Фигуры изображаются на экране, включая три невидимые строки. Чтобы нарисовать эллипс, надо изобразить окружность и затем растянуть ее, используя для этого WINDOWS.

Лучи — это часть линий, выходящие из одной общей точки. Большинство фигур могут изображаться без необходимости вызова меню SHAPES. A.C. обычно находится в этом меню до тех пор, пока не будет вызвана другаяопция.

Фигуры могут быть выполнены пластично (ELASTICALLY). Такие фигуры перемещаются по экрану как одно целое вместе с движением курсора. Нажатие «ENTER» фиксирует ее положение. Для получения таких фигур нужно установить переключатель ELASTIC до того, как делать выбор какой-либо опций в меню SHARES.

Опорные точки фигур можно привязать к элементарным блокам экрана 8x8. Привязка возможна независимо по вертикали и по горизонтали. Выбор SNAP и по вертикали и по горизонтали одновременно поместит точку в угол элементарного блока.

Приложение 1
Расширенная (EXTENDED) Артстудия

Дополнительные команды

Все возможности A.C. справедливы и для р. A.C. При этом добавляются команды для управления микродрайвером и кемпстоновской дисковой системой.

Р. А. С. — это слишком большая программа, чтобы полностью поместиться в память «Спектрума». Некоторые из менее часто встречающихся функций храняться отдельно в картридже или диске и загружаются в память только по требованию.

Так, например, меню управления печатью — это внешний модуль, загрузка которого идет 10 сек с картриджа и 2 сек с диска. Для такой работы картридж или диск с р. А. С. Должны находиться на устройстве №1.

Файл

Р. А. С. полностью поддерживает кассеты, картриджи и диски при работе с файлами. Каждое из внешних устройств управляется отдельным меню, которое вызывается через субменю после вызова бокса FILE в главном меню. Режим UNDO при этом не работает.

Работа с микродрайвом

Р. А. С. может иметь RAM-каталог. Это каталог, который хранится в памяти RAM (озу). Его можно изобразить на экране и пользоваться без необходимости всякий раз читать картридж. Чтобы вызвать этот каталог, введите курсор в бокс CATALOGUE CARTRIDGE и нажмите «ENTER». Через 10 сек в левой части экрана вы увидете большой белый прямоугольник с информацией. Вверху имя картриджа, а ниже об'ем свободного пространства в к. Это та же информация, которую дает бейсиковский оператор CAT. В оставшейся части прямоугольника запишутся имена файлов, имеющихся на этом картридже. Эти файлы можно вызвать, указав на них курсором и нажав «ENTER», после чего с ним можно выполнять SAVE, LOAD, MERGE в зависимости от того, в какой бокс вы ввели курсор. Это намного более изящный путь, по сравнению со вводом с клавиатуры.

Полная емкость RAM-каталога — 50 позиций, хотя только 8 могут изображаться одновременно. Чтобы сделать «скроллинг», надо обратиться к боксу со стрелкой. Бокс с символом в виде двух сцепленных квадратов возвращает вас к началу каталога. Полоса, расположенная между двумя боксами со стрелками показывает, какая часть полного каталога изображается на дисплее.

Если вы записываете файл на картридж в первый раз, его имя пока отсутствует в каталоге. Для команды SAVE вы должны набрать имя с клавиатуры. Для этого служит бокс ENTER FILENAME (ввод имени файла). Наберите имя и нажмите ENTER. Нажатие ENTER без имени прекратит операцию.

Р. А. С. автоматически проверяет файл после загрузки (VERIFY). Если проверка не проходит, то испорченная запись автоматически стирается и все повторяется снова. И так до 5 раз, или пока копия не будет сделана. Режимы SAVE и LOAD могут быть прерваны нажатием CAPS SHIFT и SPACE одновременно.

Режим MERGE работает так же, как то было описано для работы с магнитофоном.

Заметьте, что с начала набираем SAVE или LOAD и только потом имя файла, а не наоборот. Это нужно для того, чтобы ошибка в выборе SAVE или LOAD не испортила вам картридж.

Р. А. С. поддерживает до 4 микродрайвов. Конкретный микродрайв может быть выбран через конкретный бокс в верхней части меню.

Работа с кемптоновской дисковой системой

Р. А. С. здесь также может иметь RAM-КАТАЛОГ. Чтобы выполнить каталог диска, введите курсор в бокс CATALOGUE DISC и нажмите «ENTER». Как и в начале работы с микродрайвом,

вы увидите ту же информацию. Она равнозначна информации, получаемой по команде бейсика PRINT#4 CAT.

Вы может быть, заметили, что только определенные файлы изображаются в RAM-КАТАЛОГЕ. Это файлы, записанные в кодах и имеющие размер 6К или 6 3/4к, т.е. это экраны.

Работа с файлами: команды SAVE, LOAD, MERGE аналогичны командам для микродрайва.

Полная емкость RAM-КАТАЛОГА — 49 позиций, хотя только 8 из них могут изображаться одновременно.

Можно загружать файлы, которые не распечатались в RAM-КАТАЛОГЕ, т.к. размер их не точно 6к или 6 3/4к. В этом случае надо набирать имя файла вручную.

Редактор набора символов

Набор символов может быть загружен с кассеты, микродрайва или диска после вызова меню из бокса FILE в меню FONT EDITOR.

Здесь опции похожи по своему действию на опции главного меню р. А. С. Исключение составляет команда MERGE, которая здесь невозможна. При работе с микродрайвом или дисковой системой должно быть указано имя файла, номер же физического устройства выбирается из главного меню.

Файл с набором символов должен содержать 768 байтов (96x8) и р. А. С. откажется его загрузить, если его размер больше.

На прилагаемой кассете имеется ряд готовых наборов символов. Их можно использовать при разработке собственных картин и рисунков.

Графические элементы

Р. А. С. Имеет дополнительные фигуры — дуги, которые являются частями окружности (до полуокружности). Первые две точки определяют концы дуги, третья — прогиб. Дуга соединяет две концевые точки, но не обязательно должна проходить через третью.

Хотя и есть возможность изображать пластичные (ELASTIC) дуги, процесс это медленный и пользоваться им не рекомендуется.

Приложение 2
Компрессор экрана

Кассета с Р. А. С. Включает в себя дополнительную программу — компрессор для сжатия ваших картин. Степень возможного уменьшения размера зависит от сложности картины, но обычно размер можно уменьшить на треть или наполовину.

Одновременно можно сжимать и хранить в памяти несколько картин. Компрессор -это отдельная программа, которую необходимо загружать отдельно от р. А. С. Для этого сделайте RESET, а затем LOAD «COMPRESSOR».

Программа самозапускается после загрузки и предоставляет вам меню опций. Первые три опции управляют загрузкой рисунка с ленты, микродрайва, дисков. Компрессор запросит имя файла. Для загрузки с ленты указывать его не обязательно, тогда загружается первый встретившийся файл. После загрузки изображение автоматически будет сжато и на экране отобразится новый размер в байтах. Возврат к исходному меню выполняется нажатием любой клавиши.

Другие рисунки также можно загружать, сжимать и хранить в памяти «Спектрума». Количество их лимитируется ее об'емом. «Компрессор» сообщает вам, какой объем памяти уже использован, сколько еще осталось и сколько экранов уже хранится в памяти. Отдельные экраны могут быть изображены или стерты нажатием клавиш «4» или »5" соответственно.

Весь набор хранимых картин может быть выгружен на ленту, микродрайв или диск одной из трех последних опций. Компрессированые экраны вы можете вставлять в собственные программы, используя для этого следующую г. ~цедуру:

```
10 LET X=32768 (это число выберите сами)
20 CKTFR X-1 (защита картины)
30 LOAD "FILENAME" CODE X
40 INPUT "WHICH SCREEN?",N (какой экран, номер?)
50 POKE 23681,N
60 RANDOMIZE USR X
70 PAUSE 0
```

80 GO TO 40 При работе с микродрайвом строка 30 записывается так:

30 LOAD * "M";1;"FILENAME" CODE X При работе с дисковой системой:

30 PRINT 4: LOAD "FILENAME" CODE X

"Компрессор» можно перенести с ленты на картридж или диск. Для этого загрузите его как обычно, затем прервите (CAPS SHIFT+«6»),

Затем введите пару следующих команд:

— для микродрайва:

```
SAVE * "M";1;"COMPRESSOR" LINE 9010
SAVE * "M";1;"COMP.MC" CODE 28672,768
```

— для дисковой системы:

```
PRINT # 4:SAVE"COMPRESSOR" LINE 9050
PRINT # 4:SAVE"COMP.MC" CODE 28672,768
```

Приложение 3
Настройка программы Артстудия

Загрузите кассету и остановите ленту по указанию. На первом этапе задается устройство ввода:

1. Пять клавиш по вашему выбору.
2. Синклер-джойстик (левый или правый).
3. Кемпстон-джойстик.
4. Курсор-джойстик или курсорные клавиши (любая другая клавиша работает как ENTER).
5. Мышь.

Если вы выберете опцию 1, то от вас запросят ввод пяти клавиш в следующем порядке: «влево», «вправо», «вверх», «вниз», «ENTER». Для каждой нажмите соответствующую клавишу и ENTER.

Если вы выберете опцию 5, то от вас запросят ввод масштабного фактора «мыши». Он означает чувствительность. Чем выше число, тем меньше чувствительность.

Остальная часть настройки относится к используемому вами интерфейсу и принтеру.

По указанию продолжите загрузку с ленты.

Глава 8. Текстовые редакторы

8.1. TLW, TLW 2, TLW 2+

Общее описание

TLW — это текстовый редактор для компьютеров «Спектрум» и «Спектрум плюс». Он обеспечивает пользователю широкий круг возможностей по обработке текстов и позволяет получить результаты, удовлетворяющие лучшим профессиональным стандартам. Программа составлена в машинных кодах, кроме доступной пользователю области бейсика и содержит генератор символов фирмы MIRMIDON SOFTWARE, позволяющий производить разбивку экрана по строке в стандартах 80, 60, 48, и 40 знаков.

Программа содержит команды для обработки текстов и большое количество команд для управления печатью принтера. Она позволяет не только глубоко использовать технические возможности принтера, но и позволяет легко обращаться к «внешнему» текстовому файлу и в нижележащую бейсик-программу во время работы принтера, чтобы извлекать оттуда почти любые типы текстов или данных. «Внешний файл» может быть записан на ленте, на микродрайве, может быть введен с клавиатуры и т.п. Надлежащим образом обработанный, этот текст может быть включен в текущий файл; в этом состоит основополагающий принцип слияния повторяющихся данных при исполнении конкретного документа.

Текст программы разделен на две основные части. Это небольшой бейсиковский раздел, предназначенный для снижения верхней границы операционной системы компьютера с тем, чтобы основной блок, записанный в машинных кодах, мог бы быть загружен в верхнюю часть памяти, а также 14.5K основной блок в кодах Z-80, который непосредственно выполняет необходимые расчеты. Текст или данные, расположенные в «собственной» бейсик-области компьютера, могут вызываться из редактора разными способами. Пользователь может записать в предназначенной для пользователя бейсик-области такие строки, чтобы часть программы в машинных кодах обращалась к бейсику при исполнении печати по мере необходимости. Это выполняется введением специальных операторов в текстовый файл. Обе основные части редактора взаимодействуют для того, чтобы дать возможность пользователю со слабым знанием бейсика достичь широкого применения программ в разных ее режимах.

Перед началом работы с программой необходимо тщательно изучить эту разработку. В основе своей программа чрезвычайно проста и при некоторой практике она поможет в исполнении широкого диапазона документов.

Экран редактора

Экран содержит 256 точек в ширину и 192 в высоту. В редакторе все строки имеют по высоте 8 точек, так что экран имеет 24 строки. Однако генератор символов позволяет печатать символы шириной 6, 5, 4, или 3 точки, что дает 42, 51, 64 или 85 символов в строке.

Операционная система редактора выделяет верхние три строки экрана для заголовка («хэдера»), где отображается 18 единиц информации о текущем состоянии программы, текстового файла и положении курсора. Нижние 20 строк предназначены для изображения обрабатываемой страницы текстового файла. Запись начинается слегка отступя от левой части страницы, что позволяет «оконтурить» текст, если потребуется. Правое поле страницы используется для изображения некоторых специальных пометок, дающих информацию о статусе каждой строки текста, реальной длине строки, на которой текст может быть набран (40, 48, 60 или 80 символов), в зависимости от того, какой режим выбран. Рассматривайте эту страницу просто как «необработанное» изображение куска текстового файла, которое само по себе мало полезно, поскольку это просто список символов и цифр и не соответствует реально видимому написанию страницы. Разбивка экрана, которую вы выбрали, абсолютно не влияет на содержимое файла, это просто способ отображения, не связанный с тем, как буквально сформирован текст, как он хранится в памяти, и как он печатается на бумаге.

Конец строки

Когда строка текста длиннее, чем строка на экране, текст будет продолжаться на другой экранной строке без перерыва и так далее до тех пор, пока не будет достигнут управляющий маркер «конец строки» или конец файла. Например 84-х символьная строка занимает одну экранную строку (в режиме 80) и еще 4 позиции следующей видео строки или 1 строку + 24 позиции (в режиме 60) и т.п. Экранная строка, на которой должна кончаться текстовая строка отмечается маркером на правом поле страницы, если этот маркер не отключен.

Курсор

Место, в котором вы набираете текст называется позицией курсора. На нем базируется практически все в программе. Курсор всегда присутствует на экране или на самой странице или в командной позиции, тогда он показывает, куда войдет вводимый символ. Обычно курсор — это мигающий прямоугольник, но иногда он превращается в четыре горизонтальные полосы (в тех случаях, когда текст дошел до правого поля и ожидается очередной символ, который должен определить, что нужно сделать с введенной строкой — выровнять ее по правому полю, сделать перенос и т.д.). Для перемещения курсора имеется 19 команд — от простого смещения на одну позицию до перехода к избранной (номер вводится) строке.

Изображение команд

Некоторые команды требуют серии последовательных вводов и, возможно, наличия некоторых вспомогательных данных, которые надо иметь перед глазами. В этом случае редактор «вытягивает» кусок чистой страницы, а текст сдвигает вниз. Команда набирается на этой странице и вводится с нее, а текст возвращается на прежнее место скроллингом 20 строк вверх, если он остался без изменений, или полным его переписыванием, если изменения внутри произошли. Таким образом, здесь нет различных меню, все выполняется на одной странице.

«Белым по черному»

Вы можете менять цвет бордюра или «хэдера», присваивая им номера одного из спектрумовских цветов от 1 до 7, но можете также присвоить бордюру 0, что даст белый цвет на черном фоне. Телевизионный экран — это не самый удобный дисплей для работы с 80 знаками в строке и такой «обратный» подход может вам понравиться. Качество настройки телевизора также очень сильно влияет на разборчивость текста.

Текстовый файл

Редактор обслуживает текстовый файл, который первоначально имеет нулевой размер. Его увеличение происходит только за счет тех символов, которые вы в него вводите, плюс за счет некоторых управляющих кодов, отмечающих конец строки и конец абзаца.

Формат текстового файла

Весь текст, который вы вводите в файл, и то, как он формируется, жестко определяется правым и левым полями, а также включением и выключением режимов W.WRAP/W.SPLIT и JUSTIFY/RAGGED.

Управляющий маркер «конец строки»

Каждая строка, которую вы набираете, по достижении конца приобретет специальный управляющий код, который информирует редактор о том, что надо начинать новую строку при печати на экране или на принтере. При оформлении текста работает флаг. Он может быть двух типов.

1. Код, изображаемый стрелкой, обращенной влево и вниз. Она расположена на правом поле и указывает на то, что вы достигли края и перешли на другую строку. Код в системе ASCII — 14. Этот маркер вводится в редактор автоматически, когда при наборе нового текста он переходит границу предварительно установленного правого поля. Вы сами впечатать его не можете.

2. Код, изображаемый обращенной буквой «С» на полях действует как конец параграфа (абзаца). Его код — 13 (ASCII), что соответствует возврату каретки. Введен он может быть только оператором ENTER при создании нового текста. Строка может быть пустой и содержать только этот маркер, тогда номер колонки будет равен нулю, а вы впоследствии сможете расширить эту нулевую строку вставкой с помощью оператора INSERT.

Разрешенные движения курсора

Представьте себе, что строки текста — это полоски бумаги, наклеенные на чистую доску, другими словами — на видеостраницу. Вы можете свободно перемещать курсор влево и вправо, вверх и вниз по этим полоскам, но пространство экрана вне этих полосок «не существует» и вы никогда не сможете ввести курсор в эту запрещенную область. Если вы запросите такое невозможное перемещение,

то курсор перейдет к ближайшей колонке экрана, которая разрешена.

Положение курсора в файле

Отсчет номера строки и колонки в левом прямоугольнике «хэдера» относится к номеру строки и колонки в текстовом файле, а не к номеру строки видимой видеостраницы, кроме случаев, когда они случайно совпадают.

Размеры текстового файла

Величина возможного пространства текстового файла на один байт ниже адреса нижней границы машинно-кодовой части, то есть 50999. Нижняя часть файла находится в изначально установленном адресе в памяти «Спектрума» — 30000 (таким образом текстовый файл имеет размер 20,5K). Но нижняя граница может быть переустановлена пользователем с помощью оператора CLEAR. Так, CLEAR 49975 дает размер файла 1K, то есть 1024 байт, а область бейсика при этом составит примерно 25,5K. Максимальный размер пространства, которое можно занять под текстовый файл в 48K «Спектруме» при предельном сжатии бейсика составляет 25K.

«Хэдер» изображает текущее свободное пространство текстового файла в целых единицах килобайт. Когда пространство меньше 1K, то выводится в восьмых долях килобайта, так 3 означает, что свободны 3*128 байт.

Изменение размера файла

Когда вы выходите из редактора, в программе остается копия выходного адреса из текстового файла. При последующем возвращении текущий адрес RAMTOP сравнивается с последним выходным адресом и, если они отличаются на единицу, как раньше, никакие действия не выполняются. Если, однако, происходило изменение по команде CLEAR, текстовый файл будет вычищен и возврат в программу произойдет в пустом состоянии. К такому маневру надо подходить с осторожностью, так как он может быть очень разрушительным.

Такая подвижность нижней границы области текстового файла дает возможность легкой установки размера пространства, занимаемого текстовым файлом для того, чтобы адаптироваться к различным размерам бейсиковских программ или блоков переменных, которые надо ввести и использовать совместно с самим текстовым файлом.

Начало работы

Загрузка редактора

Загрузите программу с магнитофона. Сначала загрузится бейсиковский раздел, а затем раздел в машинных кодах. Когда загрузка закончится, мигающая строка LOADING CODE заменится вопросом INITIALIZE PRINTER? (Y). Это исходное положение редактора, записанное в строке 2030 применительно к ZX-LPRINT III-интерфейсу. Если вы используете другой, то эту строку вам надо переделать, так как все они требуют разных подходов. Далее компьютер переходит к операционной странице редактора и изображает данные по версии, которой вы пользуетесь. Дальнейшее нажатие клавиши очистит экран, и вы готовы к работе.

Разбивка текста

Первоначально программа установлена на 40 знаков в строке. Правое поле тоже установлено на этот размер. Может быть, это не то, что вы хотите, но преимущество редактора в легкости переключения на другие размеры. Попробуйте набрать какую-либо строку текста и посмотрите, как она изображается на экране. Когда вы дойдете до правого поля, продолжайте набор, и слова будут автоматически выровнены по правому полю. Так будет на каждой строке. Теперь попробуйте изменить разбивку экрана.

— нажмите обе клавиши SHIFT, в хэдере изменится режим WRITE на режим COMMAND;

— нажав CAPS SHIFT, нажмите «V» и в хэдере появится слово «VIDEO»;

— теперь вы можете набрать 48, 60 или 80 для установки количества символов в строке вместо первоначальных 40;

— в конце нажмите ENTER и ваш текст будет переписан в соответствии с выбранным размером строки.

Вы видите, что чем мельче шрифт, тем важнее качество настройки телевизора. 60 знаков в строке читаются без особых проблем, а на хорошем экране можно работать и с 80-ю знаками. Однако в этом режиме буквы набегают друг на друга, поэтому может быть вам лучше работать на одном из более низких режимов, а в режим 80 переходить только для окончательного взгляда на законченную работу, перед выдачей текста на принтер. Переход от одного размера к другому никак не влияет на содержание файла, а только на то, как эта информация представлена на экране.

Клавиша ENTER

Хотя большинство клавиш при нажатии вызывают вполне очевидные результаты, ENTER имеет специальное значение и пользоваться ею надо аккуратно. Так, когда вы набираете новый текст, курсор идет впереди набираемой строки, нажатие клавиши ENTER в этот момент приводит к вводу кода возврата каретки, т.е. конец абзаца, а курсор перемещается к левому полю следующей строки. Такой маркер очень важен для последующей работы программы с файлом и не может быть устранен обычным путем с помощью DELETE. Он будет там до тех пор, пока часть файла, содержащая его, не будет стерта более мощной командой.

Если вы находитесь внутри файла, то есть текст уже написан, то нажатие ENTER приведет к простому перемещению курсора на одну строку вниз, но не изменит никакого текста, который там уже есть.

Функция «TAB» (табуляция)

Заметьте, что команда TAB (CAPS SHIFT + I) передвигает курсор сначала к левому полю, если оно меньше, чем величина TAB, а затем вправо на величину TAB. При вводе нового текста TAB образует пробелы, а в уже существующем тексте TAB просто перемещает курсор на заданное число позиций вправо.

Левое и правое поля

Вы уже заметили, что то, что вы вводите, формируется по 40 знаков в строке от колонки 1 до колонки 40. Это результат работы полей. Величина их изначальной установки приведена в хэдере. Чтобы ее изменить, перейдите в командный режим, как вы это делали при изменении ширины видеострок и нажмите «M» для левого поля или «CAPS SHIFT+ M» для правого. Правое поле можно увеличить до 132 или уменьшить до величины, на единицу большей, чем размер левого поля. Левое поле — от единицы до величины правого поля минус один. На экране ничего не изменится, кроме надписи в хэдере, но весь текст будет подчиняться этим новым полям.

Маркеры конца строки и абзаца

Вы видите, что правое поле содержит специальные отметки в тех местах, где кончается строка. Это или наклонная стрелка или обращенная буква «C». Это визуальное отображение того, как редактор делит ваш текст, чтобы вписать его в заданные поля. Они только служат для удобства компоновки и при печати на бумаге, конечно, не воспроизводится.

Переоформление абзаца

С помощью курсора вверх (CAPS SHIFT + 7) перейдите к верхней строке вашей страницы, войдите еще раз в командный режим и нажмите «CAPS SHIFT + R». Это мощная команда REFORM. Она очистит экран от строки, в которой стоит курсор, до ближайшего конца абзаца и перепишет текст в соответствии с новыми введенными значениями полей. Измените поля еще раз, добавьте несколько пробелов, и еще раз выполните команду REFORM. Вы увидите, как аккуратно запись будет переписана для новых условий.

Буфер клавиатуры

Если вы умеете печатать достаточно быстро, то скоро увидите, что если программа и поспевает за вами в пределах одной строки, то после перехода правого поля она отстает т.к. ей нужно время для организации строки в соответствии с введенными заданиями. В этот момент необходимы специальные меры, чтобы сохранить то, что вы вводите в строку. Все, что вы вводите после каждого нажатия клавиши идет в буфер в памяти компьютера, откуда потом извлекается в удобное время.

В тех случаях, когда в буфере находится более чем один символ или команда, номер колонки в левом окне хэдера становится ярче. Буфер может принимать до 21 символа, но только одну команду. Когда вы превышаете 21 знак, программа ничего не примет, пока содержимое не опустится ниже.

Небольшой пример

Для вас подготовлено письмо. Оно записано на кассете под именем «LETTER». После того, как вы прочитаете очередную главу, посвященную записи и загрузке, вы сможете очистить текстовый файл (командный режим CAPS SHIFT + «Z»), загрузить файл «LETTER» и попробовать разные режимы работы с представленным на экране видеотекстом. Поэкспериментируйте с тем, что вы уже знаете. Чем больше вы играете, тем большему вы научитесь.

Первые итоги

В первую очередь поля определяют форму того текста, с которым вы работаете. Вторым по значению является задание «JUSTIFY»

или «RAGGED», которое выполняется: командный режим «J». В зависимости от того, что вы примете, текст по правому полю будет выравниваться (JUSTIFY), как на той странице, на которой вы сейчас читаете эти строки, или останется необработанным, как на обычной пишущей машинке (RAGGED). Кроме этого, вам еще предстоит все освоить и только одно средство для этого имеется — поставьте перед собой несложную задачу и приступайте. Не заботьтесь пока о том, как она будет распечатана, просто поработайте с клавиатурой, добейтесь уверенности в работе. В свое время поэкспериментируйте с другими командами, чтобы посмотреть, как они влияют на результат.

Персонализация программы

Когда вы вполне убедитесь в том, что редактор ведет себя на экране в соответствии с вашими ожиданиями, потратьте некоторое время на изучение управляющих операторов и указаний по подключению принтеров в приложении 3, аккуратно настройте программу, чтобы она работала так, как вам это нужно. Затем, следуя указаниям приложения 3, исполните для себя личную копию программы.

Запись и считывание текста

Если вы уже что-то набрали и дошли до той стадии, когда вам нужна постоянная копия или у вас есть в хранении какой-то текст и вы хотите ввести его в компьютер, вам необходим магнитофон или микродрайв для ввода или вывода. Для этой цели в редакторе имеются две очень простые процедуры, каждая из которых применима к обеим типам устройств.

Запись текста

Процедура вызывается: режим E «S». Теперь перед вами три выбора: задать номера первой и последней строк (включительно) того куска, который вы хотите записать; выбрать тип устройства — магнитофон или микродрайв; выбрать имя файла, под которым эта запись будет храниться.

1. Вы можете записать любую группу строк из текстового файла или весь файл целиком. Если вы сделаете частичную запись, то вам надо знать номера начальной и конечной строк до того, как вы дадите команду на запись, т.к. эти данные вводятся раньше. Если номер последней строки целого файла вам неизвестен, то подойдет любой, заведомо больший, программа сама поймет, что речь идет о последнем байте и сработает соответственно. После того, как вы введете данные о первой и последней строке, редактор сообщит вам количество килобайт, подлежащих выводу для того, чтобы вы могли спланировать свое место на внешнем носителе.

2. Выбор между микродрайвом и магнитофоном достаточно прост. Если вы имеете один или несколько микродрайвов, то есть простой и удобный путь. Программа запросит ввод от 0 до 8. Если 0, то это означает запись на магнитофон, а от 1 до 8 соответствует номеру задействованного микродрайва.

3. Выбор имени файла — ваше личное дело, любая комбинация длиной до 10 букв подходит, но первый символ — обязательно буква, а не пробел. Наберите буквы как обычно, если необходимо, пользуйтесь DELETE для устранения ошибок. Если вы введете больше, чем 10 букв, текст сдвинется и первая буква пропадает.

Когда вы с ним покончили, нажмите ENTER для того, чтобы привести в действие процедуру записи. Если вы выбрали магнитофон, вам предложат включить магнитофон на запись и нажать любую клавишу, после чего на экране появятся обычные красные и голубые полосы. Редактор не делает паузы между хэдером и главной частью, а ведет запись в обычном формате в кодах, которые могут быть впоследствии загружены либо в редактор, либо в любую другую программу, воспринимающую коды.

Если был выбран микродрайв, то выбранное устройство включится и начнет поиск, является ли для него этот файл новым или такое имя уже есть. В последнем случае компьютер проинформирует вас сообщением FOUND! NEW? (Y) и даст вам возможность выбора делать выгрузку или отменить команду. «Y» — загрузка; ENTER — отменяет команду и возвращает вас к текстовому файлу. Выбор «стереть» исполняется так: режим E, затем клавиша E. Результат тот же самый, но процесс длиннее, т.к. надо нажимать больше клавиш.

Запись текста на ленту ни в коем случае не уничтожает файл, а просто копирует избранный кусок на указанное устройство.

Считывание текста

Вызывается: режим E, затем «L». Процесс считывания очень похож на процедуру записи, за исключением, конечно, того, что не надо указывать количество материала. Необходимо только указать на магнитофон или микродрайв и имя файла.

То, что вы вводите в текстовый файл, добавляется к тому, что там уже есть, т.е. новый материал записывается в память, начиная с адреса на единицу большего, чем верхний адрес содержимого. При этом системная переменная редактора «конец файла» соответствующим образом изменяется. Это означает, что вы можете смешивать любые записанные на внешнем носителе файлы последовательной их загрузкой. Вы можете также вставить новый блок в середину текстового файла. Это делается оператором INSERT, при этом верхняя часть файла сохраняется компьютером, затем выгружается середина, а затем возвращается верхний блок.

На ленте байты хранятся как коды, а на картридже микродрайва — в формате данных.

1. Для выбора ленты вводится 0, а для микродрайва — номер, соответствующий номеру устройства от 1 до 8.

2. Вам необходимо знать точно имя файла, который вы собираетесь ввести, и набирать его точно так, как оно было записано. Любые отклонения не дадут результата. В конце нажмите ENTER, при этом для магнитофона появится надпись TAPE-ON/KEY, а микродрайв включится для поиска указанного имени.

Для ленты все названия, проходящие мимо головки, будут изображаться как SEEN: FILENAME..., а то, которое нужно, будет обработано. Для микродрайва файл будет загружен сразу после того, как будет найден. Если его нет в картридже, появится надпись ABSENT! Для ленты отсутствие имени ничего не покажет.

Вполне возможно загружать в редактор текстовые файлы или файлы данных природы иной, чем TLW. Но они должны иметь ту же организацию маркеров конца строки и конца абзаца, что и редактор. В противном случае они будут выглядеть как бесконечный стринг, не разделяемый на строки как для экрана, так и для принтера. Для переработки таких файлов на кассете имеется служебная программа TLW64BASIC — сведения по ее применению даны в приложении 4.

Команды

Для ввода команд в редактор существует два главных приема. Для первой группы наиболее широко используемых команд применяется одно нажатие шифта для того, чтобы не прерывать процесс печатания. Для второй группы команд характерна необходимость более глубокого осмысления желаемого результата, здесь обычно нужно добавление каких-либо деталей к первоначально введенной команде и для этой группы необходимо перевести хэдер в командный режим.

В целях единообразия ниже все команды даются как для 48K «Спектрума», а не для «Спектрума плюс», хотя, конечно, и на нем можно работать с программой.

В большинстве случаев длительное нажатие командной клавиши вызывает многократное повторение команды до тех пор, пока клавиша не будет отжата.

Оба способа ввода команд будем называть далее прямым и непрямым вводом.

Прямой ввод команды: CAPS SHIFT + клавиша.

Непрямой (требует перехода в командный режим — оба SHIFT вместе) — затем: — просто клавишу или CAPS SHIFT + клавиша.

Команды разделены на пять основных групп:

1. Команды курсору.
2. — « « — обработки текста.
3. — « « — структурные и утилиты, они устанавливают, каким образом построить текстовый файл и экран.
4. — « « — принтеру.
5. — « « — ввода/вывода.

Перемещение курсора

— вправо на один символ — CAPS SHIFT «8»
— влево на один символ — CAPS SHIFT «5»
— вверх на одну строку — CAPS SHIFT «7»
— вниз на одну стоку — CAPS SHIFT «6»
— табуляция вправо — CAPS SHIFT «1»

Курсор смещается на величину TAB или до левого поля, если оно ближе.

— вправо на одно слово — CAPS SHIFT «4»
— влево на одно слово — CAPS SHIFT «3»
— вправо на одно предложение — CAPS SHIFT «W».

Курсор сдвигается до ближайшей точки, а затем устанавливается около следующей позиции.

— вправо до конца строки — режим E CAPS SHIFT «8»
— влево к началу строки — режим E CAPS SHIFT «5»
— к началу первой строки — режим E CAPS SHIFT «7»
— к началу последней строки — режим E CAPS SHIFT «6»
— вниз к началу первой строки следующей страницы — режим E CAPS SHIFT «D»

Экран перепечатывается так, что последняя строка данной страницы устанавливается первой строкой сле-

дующей.
— вверх к началу предыдущей страницы — режим E «U»
— к началу текстового файла — режим E «A»
— к концу текстового файла — режим E «Z»
— к заданной строке — режим E «N».
Запрашивается и вводится номер требуемой строки.
— экран вверх на одну строку — режим E CAPS SHIFT «3»
— экран вниз на одну строку — режим E CAPS SHIFT «4»

Обработка текста

— стирание символа — CAPS SHIFT «0».
Стирается символ или оператор управления печатью.
— стереть текст до конца строки — CAPS SHIFT «9».
Эта команда требует «Y» для подтверждения или ENTER для отбоя.
— стереть текст до конца файла — режим E CAPS SHIFT «9».
— стереть весь текст — режим E CAPS SHIFT «Z».
— переключатель INSERT/OWERWRITE — (вставить/переписать) — режим E «I».

В режиме OWERWRITE, когда вводится новый текст, курсор находится на верхней границе файла и файл постепенно расширяется. Если курсор находится внутри файла, идет его «переписывание».

Если курсор находится внутри файла и включен режим INSERT, то весь текст, находящийся между курсором и концом файла удаляется с экрана и сохраняется в самой верхней части свободного пространства памяти.

Курсор устанавливается в начале освободившегося пространства и можно начать ввод нового текста или вернуть курсор назад в оставшуюся часть файла для работы с ним.

Файл ведет себя так, как будто оставшийся кусок является цельным и полным, т.е. все вводимые команды распространяются только на него и не касаются сохраненной части. Команды CLEAR и LOAD его не затрагивают. Можно сохранить весь текст и работать с другим, введенным с клавиатуры, ленты и т.п.

При повторном вызове этой команды включается режим OWERWRITE и сохраненный текст возвращается и добавляется к имеющемуся на экране. Курсор может быть где угодно и полученный текст можно, конечно, перерабатывать. Можно выполнить команду REFORM для изменения нестандартных строк, полученных в результате вставки.

— сдвинуть весь текст влево — CAPS SHIFT «Q». Текст между началом строки и курсором сдвигается влево за счет уменьшения ширины пробелов.

— сдвинуть текст вправо — CAPS SHIFT «E». Текст между курсором и концом строки смещается вправо за счет уменьшения пробелов. Эта команда не срабатывает, если промежутки между словами имеют минимальный размер, равный единичному пробелу или, если будет достигнут конец файла до того, как обнаружится пробел.

— централизовать текст в строке — режим E «H»

— выровнять (JUSTIFY) строку — режим E «J». Если справа нет маркера «конец строки», т.е. файл закончен, то эта команда не срабатывает.

— ограничить пробелы в строке — режим E CAPS SHIFT «L». Пробелы уменьшаются до единичных. Если справа нет маркера «конец строки», эта команда не срабатывает, как и выше.

— переоформить (REFORM) текст от курсора до «конца абзаца» — режим E CAPS SHIFT «R». Эта мощная команда обычно применяется для корректировки блоков текста после его набора. Текст переоформляется, сжимается или наоборот, в соответствии с тем, что установлено в правом окне хэдера. Переоформление выполняется в пределах текущего значения левого и правого поля. Если команда не проходит, т. е. встретилось слово такой длины, что не вмещается в рассматриваемую строку полностью, работа команды прекращается, а курсор устанавливается на этом слове.

— вставить оператор управления печатью — CAPS SHIFT «I». По этой команде у вас запрашивается ввод для определения требуемого оператора. В результате байт, представляющий этот оператор, вставляется в текст слева от позиции курсора, а метка, определяющая его, отображается в хэдере. Этот ввод может быть:

1. Число от 1 до 24, связанное с соответствующим кодом принтера, предназначенным для управления принтером во время печати.

2. Число от 90 до 99, указывающее на строку в бейсике, к которой надо сделать переход (строка определяется как 100, умноженное на это число, т.е. от 9000 до 9900).

3. Буква от A до Z со знаком $, когда требуется напечатать избранный бейсиковский стринг. При этом переключатель PCT должен быть включен и, конечно, операторы управления принтером на печать не выводятся.

— ввод символа «копирайт» — режим E CAPS SHIFT «K». Синклеровский знак «копирайт», CHR$127 вводится по этой команде в файл. Имейте в виду, что этот символ не стандартизован и на принтере его может не быть.

— повтор куска текста — режим E «R». Запрашивается ввод первой и последней строки потребного куска.

— удаление куска текста — режим E «K». Вводятся номера первой и последней строк удаляемого текста.

— поиск или изменение некоторого блока текста — режим E «X». Вводятся номера первой и последней строк, внутри которых производится поиск. Затем вводится стринг, подлежащий замене, а затем либо заменяющий стринг, либо ENTER, если замена не нужна. Программа ищет и изображает номера строк и колонок для найденных позиций. Каждый раз вы можете произвести замену клавиш «S», а ENTER — оставить без замены. Продолжить поиск — «F». Прекратить поиск — «C».

Данная процедура может быть полезна, если вам надо найти строку, номер которой вы не знаете, но знаете что-либо, что в ней находится.

Структурные команды и утилиты

— переключатель «JUSTIFY/RAGGED» — режим E «J». Команда переключает два режима. В первом концы строк по правому полю автоматически выравниваются по окончанию текущей строки, когда появляется «полосатый курсор», а также, конечно, после команды REFORM. При установке на «RAGGED» текст уплотняется, правый край остается неровным.

— переключатель W.WRAP/W.SPLIT — (округление слов/перенос слов) — режим E «W». Действие этой команды похоже на действие предыдущей. Работает при создании текста или при выполнении команды REFORM. Когда включен режим WRAP, слова не делятся и не переносятся в конце строки. В режиме SPLIT текст режется на куски, определяемые заданными значениями полей. Этот процесс необратим. Когда включен этот режим, в окне JUSTIFY/RAGGED загорается стрелка, указывающая на то, что ни один из этих режимов не может быть задействован, но они будут работать нормально после команды REFORM.

— установка регистра букв на одну строку — режим E «2». Вся строка будет печататься заглавными буквами или строчными.

— включение в текст бейсиковского стринга — режим E CAPS SHIFT «4». Запрашивается ввод значения стринга из области бейсика, т.е. имя от A$ до Z$. Изображаются первые 8 символов стринга с последующим знаком $ и указанием длины стринга.

Стринги должны быть «простыми» и должны браться целиком. В программу можно вводить все коды между 22 и 151, т.е. весь спектр ASCII плюс 24 оператора управления принтером, но не стринги, вызываемые процедурой ввода операторов управления принтеров (PCT) от A$ до Z$ и не строки от 9000 до 9900. Результат можно включить в файл — «U», сбросить и повторить — «N», отменить команду и выйти из нее — «Q».

— включение/выключение контурной линии — режим E «O»

— включение/выключение видеомаркеров — режим E «V». Когда режим включен, то в хэдере рядом со значением ширины страницы «VIDEO» изображается маркер «конец абзаца».

— переключатель регистров вр/нр — CAPS SHIFT «2».

— установка величины левого поля — режим E «M». Запрашивается ввод числа, которое может быть от 1 до величины правого поля минус единица.

— установка величины правого поля — режим E CAPS SHIFT «M». Запрашивается ввод числа, которое может быть от величины левого поля плюс 1 до числа 132.

— установка величины разбивки экрана — режим E CAPS SHIFT «V». Запрашивается ввод числа: 40, 48, 60, 80. Экран перестраивается.

— установка величины табуляции — режим E CAPS SHIFT «T». Величина TAB может изменяться от 1 до величины правого поля минус 1, но не может быть более 80.

— изменение цвета бордюра — режим E CAPS SHIFT «B». Цвет бордюра может быть одним из основных спектрумовских цветов от 1 до 7, независимо от цвета страницы и хэдера. Если принять 0, то будут белые символы на черном фоне.

— изменение цвета хэдера — режим E CAPS SHIFT «H».

— счетчик слов — режим E CAPS SHIFT «W». По этой команде пересчитываются все группы символов, разделенные пробелами, независимо от того, являются ли они словами. Результат изображается вместе с «видимой» длиной файла в байтах.

— работа калькулятора — режим E CAPS SHIFT «C». Страница текста убирается вниз, чтобы вы могли записать выражения дли-

ной до 40 символов, после чего самый левый символ исчезает. Выражения должны быть законченными математически, хотя знак «=» не требуется, так выражение 3*128 является законченным вводом. Пробелы не требуются. Стринговые выражения могут быть «урезанными», т.е. если A$="12345678", то выражение VALA$(2T05) будет расценено как 2345 и т.д.

После нажатия ENTER автоматически будет выполнен расчет до получения числового ответа, а если это невозможно, то загорится вопрос WHAT? и неправильное выражение будет на экране для определения ошибки до тех пор, пока не будет нажата какая-либо клавиша, после чего экран будет очищен.

Далее вы сможете сохранить этот результат в одном из регистров памяти, нажав M0, M1,...M9. Он запоминается как стандартная бейсиковская переменная и доступен для калькулятора, когда это необходимо.

Нажатие любой другой клавиши приведет вас к следующей стадии. Здесь вы можете использовать полученный результат — «U» (т.е. вставить его в текстовый файл), отбросить его и повторить команду — «N» или вернуться в режим ввода текста — «Q».

При написании математических выражений клавиатура модифицируется таким образом, чтобы иметь возможность использовать большинство синклеровских логических и математических функций.

функция	клавиша	SHIFT
+	K	SYMB
-	J	SYMB
*	B	SYMB
/	V	SYMB
^	N	SYMB
SIN	Q	CAPS
COS	W	CAPS
TAN	E	CAPS
ASN	S	CAPS
ACS	C	CAPS
ATN	T	CAPS
EXP	X	CAPS
SQR	H	CAPS
INT	R	CAPS
LN	Z	CAPS
VAL	J	CAPS
SGN	F	CAPS
ABS	G	CAPS
PI	M	CAPS
<	R	SYMB
=	L	SYMB
<>	W	SYMB
>	T	SYMB
>=	E	SYMB
OR	U	CAPS
AND	Y	CAPS
NOT	S	CAPS

— установка времени — режим E CAPS SHIFT «X». В редакторе имеются часы со встроенным таймером (будильником). Таймер может быть установлен на время от 0 до 250 минут (изначально — 30 мин.). Если на 0, то таймер выключен. Таймер предназначен для регулярности сброса введенного текста на ленту. Отключается звуковой сигнал — «T». Возврат к вводу текста — «Q».

— отбой введенной команды — CAPS SHIFT «Q».

— возврат к бейсику — режим E CAPS SHIFT «B».

Печать

— управление печатью — режим E «G». Это не столько команда, сколько программа для управления процессом печати. Загорается меню, изображающее состояние канала принтера. Если требуется пересмотреть какие-либо исходные данные — нажмите «R». В каждой позиции нажатие ENTER приводит к перемещению курсора без внесения изменений.

1. LINE PRINTER — переключается на ZX-принтер при нажатии «R». Когда стоит LINE PRINTER, то запрашивается ввод адреса возврата. Введите 0, если используется интерфейс без обратного обращения после инициализации или обычный адрес для интерфейса, который имеет блок памяти и которому нужно также обычно из бейсика обращение RANDOMIZE USR для задействования.

ZX-принтер всегда использует резидентный генератор шрифта из редактора для печати 40, 48, 60, 80 символов в строке, в соответствии с тем, какой режим избран.

2. Байт, который использует принтер для возврата каретки, обычно — 13, но надо уточнить по руководству к принтеру.

3. Байт, необходимый для перевода строки. Он может быть необходим отдельно от B.K. Если не нужен — введите 0.

4. Все 4 байта каждого оператора управления принтером. Они вводятся автоматически вводом специального числа — префикса. Он сообщает интерфейсу принтера, что за ним должна пойти группа управляющих кодов.

Отсюда можно перейти прямо к меню печати — «P» или к пересмотру — «R» или к тексту — «Q».

— печать текста — режим E «P». Загорается меню, на котором изображены:

— номера первой и последней строк файла;

— установленное количество строк файла;

— состояние операторов управления печатью (PCT).

Для изменения чего-либо — «R», для выполнения печати — «P», для возврата к тексту нажмите — «Q».

За вводом «P» каждая копия печатается по команде ENTER. Хотя нажатие «Q» позволяет выйти из этого меню и перейти к тексту, нажатие «Q» во время работы принтера прерывает печать и переводит к началу новой копии.

Оператор BREAK переводит в бейсик.

— файл PCT (файл операторов управления принтером) — режим E «T». Все 24 оператора управления принтером полностью дисассемблированы вместе со своими метками и копией стандартного префикса. Вы можете ввести любое число от 1 до 24 для движения курсора по меткам и кодам или вернуться к текстовому файлу — «Q». Все метки или коды можно переопределить. Все коды могут быть от 0 до 254. Ввод 255 при печати игнорируется и обозначает пробел.

Хранение файлов и вызов

— записать текст на внешний носитель — режим E «S». Запрашиваются номера первой и последней строк выводимого текста, а также число: 0 — для ленты или от 1 до 8 для микродрайва. Ввод завышенного номера последней строки всегда приводит к установлению действительного значения конца файла. Так, ввод 1 и 9999 приводит к записи большинства файлов полностью. Если необходимо точно знать номер последней строки файла, то можно воспользоваться первой частью команды вывода на печать.

Если файл данного имени уже есть на картридже микродрайва, то появится сообщение FOUND! NEW?(Y). Ответ (Y) приведет к стиранию старого файла и записи нового. Просто ENTER — возврат к текстовому файлу.

Запись на микродрайв идет в формате данных, а на ленту — в кодах.

— загрузить текст — режим E «L». Текст, загружаемый в редактор, не переписывает то, что уже есть там, а пристегивается к концу файла. В пустой файл, соответственно, загрузка идет в самую нижнюю часть памяти.

Если используется команда вставки INSERT и имеется некоторый «спрятанный» текст, то весь материал, вводимый в файл, добавляется к видимой части файла, что позволяет вам вставлять любой материал внутрь уже существующего файла. Это позволяет выполнять большинство требуемых «сшиваний» текстов. Команда включения бейсиковского стринга выполняет аналогичную функцию.

После команды LOAD вводится номер носителя 0 — для ленты и от 1 до 8 для микродрайва и имя, которое нужно отыскать. При работе с магнитофоном, все имена читаются и записываются на экране, пока не найдется искомое, тогда начинается загрузка.

При работе с микродрайвом, если искомое имя отсутствует, то выдается сообщение ABSENT! Если файл испорчен, то появится обычное бейсиковское сообщение FILE NOT FOUND, но при этом все «хорошие» 512-байтные секторы до испорченного загрузятся в редактор. Если файл на картридже, который вы хотите загрузить, слишком велик, загрузится сколько возможно, а потом появится сообщение SPACE, то есть от дальнейшего ввода программа отказывается. С ленты байты воспринимаются в формате кодов, а с микродрайва — в формате данных. Если ваш текст записан на микродрайве в кодах, то самый лучший способ — переписать его на ленту обычным порядком и загружать его оттуда.

— раскаталогизировать картридж — режим E «C». Вводится номер микродрайва. В окне «TITLE» появится имя картриджа и на экране распечатываются имена файлов и затем размер свободного пространства в килобайтах.

— стереть файл на картридже — режим E «E». Запрашивается номер микродрайва и имя файла. Если этого файла нет, то загорается ABSENT!

— выполнить форматирование картриджа — режим E «F». Вводится номер микродрайва и новое имя картриджа.

Печать текста

Все рассмотренные программы служат для создания текста, но основной, заключительной частью является печать. Вы выбираете что печатать и как печатать, управляя принтером по программе.

Рассматривайте процедуру печати как управляющую программу. Ее материал — это текстовый файл, включающий в себя операторы PCT — операторы управления печатью.

Процедура вызывается — режим E «P» и затем следуют ответы на следующие запросы:

1. Номера первой и последней строк выводимого текста. При этом программа сама обеспечивает вас следующими данными: номера первой и последней строк существующего файла, интервал между строками, количество копий, установка PCT. Если эти данные вас устраивают, можно непосредственно начинать печать без изменений. Если нет, то их можно изменить — «R». Как и в операции SAVE, первый номер строки должен быть точным, а вместо последнего (для выведения файла целиком) годится любое заведомо большее число.

2. Интервал между строками — может быть любое число от 1 до 254.

3. Количество копий — от 1 до 254.

4. Статус ACTIVE/NON-ACTIVE шестидесяти операторов управления печатью. Он может быть включен/выключен вводом «R», когда курсор находится на PCT.

Более специфическое управление процессом печати осуществляется посредством вставленных в текст PCT операторов, обращающихся к принтеру для изменения формата печати, к области переменных бейсика для отыскания стринга или к области бейсик-программы для отыскания и выполнения некоторых строк перед возвратом в машинно-кодовый блок. Если произойдет сбой (встретится невыполнимый оператор PCT и т.п.), печать прерывается и вы выбираете — переходить к другой копии (если она есть) или возвратить программу к текстовому файлу.

Имеет смысл хранить копию текста на ленте или микродрайве перед выводом ее на печать. В случае непредвиденных сбоев, которые бывают, вам это поможет.

Количество переменных, которые «дрейфуют» между принтером и текстовым файлом может быть немалым, при этом вовлекаются многочисленные PCT операторы. Связь между всеми элементами превращает редактор в довольно мощное средство, но искусство работы зависит от вас.

Операторы управления принтером (PCT)

Коды, которые компьютер выдает принтеру для печати, находятся в диапазоне от 32 до 127. Кроме того, принтеры используют специальные коды для исполнения специальных функций при печати, для чего редактор имеет банк из 60 настраиваемых операторов управления печатью, которые могут быть вставлены заранее где надо в текстовый файл для исполнения какой-либо процедуры при исполнении самой печати.

Операторы PCT разбиваются на три группы:

— непосредственное управление принтером (24 оператора);

— для отыскания и печати именованных стрингов из бейсик области (26 операторов);

— для передачи управления из редактора в бейсик (10 операторов).

Именно последние две группы обеспечивают пользователю возможность создания комплексных документов, где стандартные блоки текста искусно включают в себя блоки данных из внешнего источника, т.е. из бейсик-программы или от микродрайва и т.п.

Все операторы PCT вставляются в текст с использованием CAPS SHIFT «I». За этой командой требуется ввод числа от 1 до 24 для операторов непосредственного управления принтером, ввод буквы от A до Z с последующим знаком доллара для одного из 26 стринговых операторов PCT или ввод числа от 90 до 99 для обращения к строкам бейсика от 9000 9900. Оператор PCT всегда вставляется в текст слева от видимого на экране символа. Эти операторы не занимают экранного пространства, строки остаются правильной длины, отсчет колонок не нарушается.

Вы можете вставить оператор PCT где угодно в тексте и он ничего не затрет.

Для того, чтобы указать, где в тексте спрятан оператор PCT, следующий за ним символ отображается в инверсном виде. Если это невозможно, т.е. оператор стоит в самом конце строки, то в этом случае маркер конца строки отображается подчеркнутым. Инвертирование символов может быть включено или выключено. Для этого применяется команда включения видео маркера — режим E «V», которая работает как переключатель. Индикатором на экране при этом является наличие или отсутствие маркера «конец абзаца» в хэдере рядом с величиной разбивки экрана. Генерация маркеров «конец строки» также включается и выключается по этой команде.

Если курсор ввести в середину текста и поместить его на PCT операторе, то метка оператора появится в хэдере. Это произойдет независимо от того, включено или выключено изображение видеомаркеров.

Влияние всех операторов PCT может быть включено или выключено командой «R» при инициализации меню печати. При этом 24 оператора управления печатью на печать не выводятся, но остальные два типа — печать стрингов или обращение к строкам бейсика — выводятся, если переключатель PCT включен.

Операторы непосредственного управления принтером

Редактор содержит файл из 24 управляющих операторов, цель которых — обеспечить выдачу на принтер избранных групп по 4 кодовых байта для сообщения принтеру о необходимости выполнения специальных действий и, тем самым, для изменения формата печати.

Эти операторы, вставленные в файл, — одиночные байты из диапазона от 128 до 151, пронумерованные от 1 до 24. Команда режим E «T» отображает на экране текущий формат операторов PCT, все метки и соответствующие им значения кодов в диапазоне от 0 до 254. Значение 255 изображается как «-». Все метки и байты могут быть пересмотрены.

Используя команду управления печатью, можно ввести стандартный префиксный байт, который будет выдаваться перед группой байтов PCT и выполнять роль инициатора для интерфейса или принтера.

Файл операторов управления принтером выглядит так:

N	метка	управляющ. коды			
1	FRM.F	1	12	—	—
2	ELIT+	2	27	77	—
3	PICA+	2	27	80	—
4	ENLG+	3	27	87	49
5	ENLG-	3	27	87	48
6	COND+	1	15	—	—
7	COND-	1	18	—	—
8	UNDL+	3	27	45	49
9	UNDL-	3	27	45	48
10	EMPH+	2	27	69	—
11	EMPH-	2	27	70	—
12	D.ST.+	2	27	71	—
13	D.ST.-	2	27	72	—
14	SUB+	2	27	83	48
15	SUB-	3	27	83	49
16	SP/B-	2	27	84	—
17	ITAL+	2	27	52	—
18	ITAL-	2	27	53	—
19	BELL!	1	7	—	—
20	PPR-	2	27	56	—
21	PPR+	2	27	57	—
22	INITL	2	27	64	—
23	A4/70	3	27	67	70
24	RRF	3	27	78	9

стандартный префикс — 1

Как уже было сказано выше, метки и значения, приведенные здесь, относятся к интерфейсу ZX LPRINT III и к принтерам системы EPSON-RX80. Все эти метки и коды могут быть изменены. Файл операторов управления принтером может быть вызван также и из бейсика, см. приложение 1.

Если вам необходимо иметь копию таблицы операторов управления принтером на микродрайве или на ленте, это можно сделать следующим образом:

На ленту, на микродрайв 1 и т.д.

```
SAVEPCT-TABLECODE5103, 217 SAVE*M;1;PCT-
TABLECODE5103, 217
LOAD PCT-TABLECODE LOAD*M;1;PCT-TABLECODE
```

Печать именованного стринга

Если после PCT — CAPS SHIFT «I», набрать какую-либо букву и знак $, а затем ENTER, то можно вставить управляющий оператор, который скомандует редактору найти и вставить в это место при печати бейсиковский стринг, имя которого совпадает с данными оператором управления печати. Другими словами, оператор A$ вызовет печать бейсиковского стринга, который в этот момент имеет это имя. Стринг будет напечатан точно так, как будто он является частью текста, хотя сам текстовый файл останется без изменений. Если этот стринг не может быть найден, т.е. он не был присвоен бейсиковской переменной или он многомерный, что недо-

пустимо, то в этом случае печать остановится и редактор выдаст сообщение: CHECK A$ (проверьте A$). Программа перейдет к исполнению следующей копии, если она нужна, можно выйти отсюда — «Q» или вернуться в текстовый файл, если необходимости в копиях нет.

Переход к бейсик-строке

Командой CAPS SHIFT «I» и вводом числа от 90 до 99 осуществляется ввод оператора управления печатью, который в процессе печати остановит принтер и обратится к бейсик-строке, номер которой в сто раз больше введенного числа. Так, ввод 96 отправит компьютер к строке 9600.

Цель этого маневра — дать пользователю возможность записать некоторую бейсик-программу для переделки одного или нескольких стрингов, находящихся в памяти бейсика, чтобы иметь преимущества при дальнейшей печати.

Принтер можно, естественно, задействовать напрямую режимом LPRINT, что может быть удобным для выдачи некоторых простых вещей — комментариев, подчеркиваний слов и т.п. Вы можете делать PRINT и на экран этим способом, хотя необходимо учитывать, что печать должна идти от 4-й строки до 21-й, т.к. если страница и будет переписана после возврата из бейсика, то хэдер — нет. Если вы запортите верхнюю часть экрана, то команда режим E «A» исправит экран, но вы окажетесь в начале текстового файла.

Когда программа в бейсике отработала полностью, заключительная команда должна быть GO TO 9999, чтобы совершить прыжок назад в редактор. Переход к машинным кодам осуществляется оператором RANDOMIZE USR 65200, расположенным в этой строке, хотя это можно сделать и из любого другого места программы.

Если возврат в редактор невозможен по этому адресу, т.е. при вызове принтером бейсика не был «оставлен» адрес возврата, программа начнется с начала текстового файла, как будто по команде режим E «A», а принтер не будет работать до следующего обращения к нему.

Составление и слияние комплексных документов

Существует два способа, с помощью которых вы можете включить «внешний» текст в редактор там, где вам это надо. Первый выполняется при наборе, когда вам надо вставить заранее подготовленную последовательность символов, например, данных или часто используемую повторяющуюся последовательность символов или букв. Второй способ применяется при печати на принтере и позволяет вставлять практически неограниченное количество материала в виде бейсиковских стрингов, взятых из области переменных бейсика. Вставляются они в те места, где были помещены оператором PCT. Можно также выполнять вычисления в бейсик-области, поместив в выводимый файл оператор PCT для перехода в бейсик-область.

Включение стринга при наборе текста

Если простой бейсиковский стринг существует в области переменных бейсика, то командой INCLUDE — CAPS SHIFT «I» вы можете набрать имя этого стринга при вводе текста, тогда на экране появятся первые 8 символов со знаком $ и указанием его длины. Далее вы можете использовать его, т.е. вставить в текст — «U», перейти к другому — «N», или отбросить совсем и вернуться к набору текста — «Q». Если вы его использовали, то стринг впишется в ваш текстовый файл так, как будто вы его впечатали и, конечно, все заданные режимы, например, выравнивание правого поля, перенос слов и т.п. будут соблюдены.

Включение стринга при печати текста на принтере

Опять, если предположить, что такой стринг существует, вы можете использовать один из 26 операторов управления печатью, предназначенных для впечатывания стрингов в качестве единицы печатного материала. Это должен быть простой стринг. Операторы PCT находятся в пределах от A$ до Z$. Пользователь должен обратить внимание, что мы нигде не говорим о длине стрингов, поэтому внимательно смотрите, чтобы их длина соответствовала размерам принятой вами страницы.

Если у вас есть текст или данные, содержащиеся в файлах, полученных другими программами (например, программой кэмпбелл мастерфайл), то наилучший способ ввода этих файлов в редактор осуществляется через бейсик, до входа в TLW. Для этого текст, подлежащий вводу, присваивается (LET) бейсиковским стрингам, которые затем вызываются и вставляются или из TLW или непосредственно во время печати операторами PCT, предназначенными для вставки стрингов в текст.

Материал, который вам нужен, легче всего получить из мастерфайла через бейсик-строки 4900, 6000, 7000. В этих строках записи доступны в виде стринга C$ (), имеющего размер 130. Первый байт — это метка данных (от A до Z), а остальные 129 байтов — это сами данные. Их можно записать на микродрайв, например, так:

```
4900 LET D$=
4910 LET N=0:REM указатель номера файла
4920 LET N=0: указатель номера файла
6000 LET D$ L D$+C$(R т0) + CHR$0: REM прием
данных, установка маркеров.
6010 GO TO USR R: REM возврат в мастерфайл за
другой порцией данных.
7000 LET N=N+1: REM подготовка к выгрузке файла
7010 OPEN #;M;1;MFL DATA+STR$N
7020 PRINT #8;D$:CLOSE #S:REM выгрузка записей
7030 GO TO USR R: REM возврат в мастерфайл.
```

Здесь данные D$ записываются на микродрайв 1 с именами файлов MFL DATA1, MFL DATA2, и т.д. Последующий вызов в редакторе может быть выполнен, например, такими строками:

```
9000 LET N=1:REM для инициализации используйте
оператор
9001 REM управления принтером 90.
9010 GO TO 9999: REM возврат в TLW...
9500 OPEN #5;M;1;MFL DATA +STR$N:REM здесь ис-
пользуйте
9501 REM оператор PCT 95.
9510 INPUT #5;H$:CLOSE #5:LET N=N+1:REM за-
пись вводится
9511 REM как H$ и т.д.
9520 GO TO 9999:REM возврат в TLW
9600 FOR A=0 TO 4: REM для пяти записей и т.д.
9610 LET T$=H$((A*130+1)TO(A*130+129))
9620 GO TO 9999:REM возврат в TLW
9700 NEXT A: REM чтобы вернуться к 9600 четыре раза,
9701 REM используйте PCT 97
9710 GO TO 9999:REM окончательный возврат в TLW
```

Обратите внимание на то, что поскольку информация из мастерфайла записывается на микродрайве в формате данных, то теоретически есть возможность их загрузить в TLW и просмотреть, но при этом, конечно, с ними нельзя оперировать без наличия программы мастерфайл.

Если у вас накопился большой объем информации, которую нужно переработать таким образом, то определенно имеет смысл освоить работу с файлами при помощи микродрайва и интерфейса-1.

Вызов бейсика в процессе печати

С помощью команды вставки кода управления принтера и вводом числа от 90 до 99 можно отправить компьютер в бейсик-область в процессе печати текста. Цель этого маневра — выполнение каких-либо вычислений, в частности для того, чтобы пересмотреть некоторые стринги и создать новые, которые можно использовать соответствующей командой управления печатью для включения стрингов. Пользователь для этого должен в бейсик-области написать соответствующую программу. В то же время, из области бейсика пользователь может управлять печатью в обычном режиме LPRINT, а для каких-либо действий может вызвать и экран редактора. Если используется микродрайв для обеспечения каких-либо стринговых данных через команду INPUT, то две нижние строки экрана будут очищены, но это не имеет значения, т.к. страница будет переписана по окончании процесса текущей печати. Если в процессе работы бейсика происходит ошибка, компьютер останавливается обычным образом, это не влияет на продолжение процесса печати при условии, что строка бейсика 9999 или оператор RANDOMIZE USR 65200 по прежнему используется для возврата в редактор, что обычно и бывает в случае появления ошибок.

Итак, когда компьютер отправляется в бейсик-область, его цель — строки 9000, 9010,.... 9900. На каждой из них можно организовать выполнение какой-либо специфической задачи при условии, что предусмотрен возврат через строку 9999 или через оператор RANDOMIZE USR 65200. Фактически любая ошибка, возникающая при работе в бейсик-области не влияет на редактор, так что вы можете выполнять любые действия и рассчитывать на возврат в редактор через адрес 65200.

Калькулятор

В любое время, когда программа находится в режиме ввода текста, вы можете ее использовать и как калькулятор для вычислений, если вы запишите выражение, соответствующее вашим требованиям. Результат вычислений может быть присвоен одной из бейсиковских переменных от M0 до M9, как бы калькулятор имеет десять регистров памяти.

Когда калькулятор вызывается -- режим E CAPS SHIFT «C» — страница опускается на две строки и высвечивается слово ASSES: в высвечиваемой зоне вы можете записать какие-либо выражения, результат которых тоже число, например, 4*3+8*5, а может быть набор чисел, функций, логических операций, имен переменных, операций над стрингами. Если, например, вы запишете выражение в виде стринга:

LET C$="M1+125.5#(INT(M2/M3)*(1+VAT),

то вы можете выполнить и VAL C$, если все входящие переменные M1, M2, M3 VAT и т.п. существуют и определены. Если результат не может быть получен по какой-либо причине, т.е. вычисления прервутся, вы получите вопрос (WHAT?) и выражение, которое вы записали, будет на экране до нажатия другой какой-либо клавиши.

Если результат будет выдан, перед вами есть выбор запомнить его в бейсиковской переменной от M0 до M9, для этого просто нажмите цифру от 0 до 9. Любая другая клавиша приведет вас к другому тройному выбору — «U», чтобы использовать результат, т.е. ввести его в текст, как будто он набран с клавиатуры; «N» — опустить результат и повторить команду; «Q» — отбой команды. Ваш текст не изменится, если вы не нажимали клавишу «R».

Имейте в виду, что не надо пытаться использовать результат, если курсор стоит или приближается к строке нулевой длины. Напрямую вставить текст туда невозможно, для этого надо использовать оператор INSERT.

Если в соответствии с какой-либо специфической ситуацией вам необходима особенная последовательность выражений, вы можете иметь ее на кассете и впоследствии перезагрузить в редактор, когда это необходимо. Однако предусмотрите, чтобы при этом не выполнялись CLEAR и RUN, не то те переменные, которые у вас существуют, могут быть потеряны.

Приложение 1. Технические данные

Эти данные приводятся для возможности определения из бейсика некоторых системных переменных редактора.

Таблица адресов некоторых системных переменных

блок в машинных кодах	от 51000 до 65535
адрес вызова	51000
адрес возврата при печати	65200
режим работы процессора с прерываниями, 2 рода	
размер текстового файла	0...25K, в зависим. от RAMTOP
файл операторов управления печатью	от 51003 до 51218
формат: 24 различных оператора, каждый состоит из:	
байты от 1 до 5	— байты операторов ASCII
байты от 6 до 9	— байты кодов, их значения от 0 до 254 (255 — пробел)
стандартный префикс — от 0 до 254	— 51219, один байт
блок кодов интерфейса принтера	от 65176 до 65199 вкл
память калькулятора — 10 бейсик-переменных от M0 до M9	
переменные калькулятора — все бейсик-переменные	
длина включаемых стрингов — в соответствии с бейсиком	
имена включаемых стрингов переменных — от A$ до Z$	

Некоторые системные переменные:

установка таймера	65203, 1 байт
вершина пространства текст. файла	65207, 2 байта
начало текст. файла	65209, 2 байта (RAMTOP+1)
байт «возврат каретки»	65255, 1 байт
байт «перевод строки»	65256, 1 байт
величина разбивки экрана	65270, 1 байт
адрес обращения к строчн. принтеру	65271, 2 байта
число байт в буфере клавиатуры	65277, 1 байт
вершина текстового файла	65285, 2 байта
адрес курсора в файле	65286, 2 байта
колонка курсора в тексте	65288, 1 байт
строка курсора в тексте	65290, 2 байта
левое поле	65295, 1 байт
правое поле	65296, 1 байт
интервал между строк для печати	65299, 1 байт
количество печатаемых копий	65300, 1 байт
значение табуляции	65303, 1 байт
буфер клавиатуры	65514, 2 байта

Все вышеперечисленные данные могут быть вычислены через бейсик:

PRINT PEEK ADRESS — для однобайтных переменных

PRINT PEEK ADRESS +256*PEEK(ADRESS+1) — для двухбайтных.

Копирование файла операторов управления принтером

На кассете с редактором записана программа, которая может дать распечатку операторов управления принтером. Ее имя — «TLW-TO KENS». Она может оказаться полезной для пользователя. Сводка этих операторов не включена в кодовый блок редактора из соображений экономии памяти, но в случае потребности в них, прилагаемая программа вполне достаточна.

Приложение 2. Сводка команд

Перемещение курсора

вправо на позицию	C.S.	«8»
вверх влево на этой странице	p.E S.S.	«7»
влево на позицию	C.S.	«5»
вверх на строку	C.S.	«7»
влево вниз на этой странице	p.E S.S.	«8»
вправо на величину табуляции	C.S	«I»
влево вверх на след. страницу	p.E	«D»
вправо на слово	C.S.	«4»
влево на слово	C.S.	«3»
влево вверх на след. страницу	p.E	«U»
вправо на предложение	S.S.	«W»
к началу файла	p.E	«A»
вправо до конца строки	p.E S.S.	«8»
к заданной строке	p.E	«N»
влево к началу строки	p.E S.S.	«5»
страницу вверх на строку	p.E S.S.	«3»
страницу вниз на строку	p.E S.S.	«4»

Обработка текста

стереть символ	C.S.	«0»
стереть строку до конца	C.S.	«9»
стереть до конца файла	p.E S.S.	«9»
стереть весь текст	p.E S.S.	«Z»
переключатель INSERT/OVERWRITE (вставить/переписать)	p.E	«I»
сдвинуть текст влево	S.S.	«Q»
сдвинуть текст вправо	S.S.	«E»
отцентрировать текст в строке	p.E	«H»
выравнять данную строку	p.E S.S.	«J»
сжать строку	p.E S.S.	«L»
переоформить текст до конца абзаца	p.E S.S.	«R»
вставить оператор управления печатью	S.S.	«I»
вставить символ «копирайт»	p.E S.S.	«K»
повторить блок текста	p.E	«R»
уничтожить блок текста	p.E	«K»
найти/заменить текст	p.E	«X»

Утилиты и структурные команды

вкл/выкл. выравнивание строк	p.E	«J»
вкл/выкл. переноса слов	p.E	«W»
перекл. регистра печати букв на одну строку	p.E S.S.	«2»
включить бейсик-стринг	p.E S.S.	«I»
вкл/выкл. контурную линию	p.E	«O»
вкл/выкл. видеомаркеры	p.E	«V»
верхний/нижний регистр	C.S.	«2»
изменение левого поля	p.E	«M»
изменения правого поля	p.E S.S.	«M»
изменить разбивку экрана	p.E S.S.	«V»
изменить величину табуляции	p.E S.S.	«T»
изменение цвета бордюра	p.E S.S.	«B»
изменение цвета хэдера	p.E S.S.	«H»
подсчет слов/байт	p.E S.S.	«W»
калькулятор	p.E S.S.	«C»
установка таймера	p.E S.S.	«X»
отбой команды	S.S.	«Q»
возврат в бейсик	p.E	«B»

Команды печати

вызов меню печати	p.E	«G»
вызов файла операторов управления принтером	p.E	«T»
напечатать текст	p.E	«P»

Команды ввода/вывода

загрузить текст	р.Е	«L»
записать текст	р.Е	«S»
отформатировать картридж	р.Е	«F»
раскаталогизировать картридж	р.Е	«C»
стереть картридж	р.Е	«E»

р.Е — режим Е; C.S. — CAPS SHIFT; S.S. — SYMBOL SHIFT.

Приложение 3. Подключение принтера

Существует много типов интерфейсов принтеров, приемлемых для «Спектрумов». Большинство из них по-своему преобразуют сигналы, которые поступают через разъем компьютера, в форму, необходимую для работы принтера. Редактор хорошо оснащен для того, чтобы работать со многими из этих интерфейсов. Эта глава посвящена настройке программы для работы с разнообразными интерфейсами.

В редакторе применен общепринятый подход, согласно которому вывод на экран производится по 2 каналу, а вывод на принтер — по третьему каналу. В то же время, ZX-принтер работает через генератор символов по каналу 2 и сильно отличается от обычного принтера. Интерфейс, который вы используете, может быть вполне самообеспечен, т.е. необходимости в специальных кодах для его запуска нет, но может для своей работы потребовать дополнительные коды либо внутри редактора, либо где-нибудь в ПЗУ. Тогда эта последовательность кодов будет служить связующим звеном между компьютером и принтером при всяком обращении к принтеру. Она может, например, просто перезадать адрес вызова 3 канала так, чтобы задействовалась стандартная инструкция «RESTART 10», которую редактор использует для выдачи символа, содержащегося в регистре А микропроцессора, на канал принтера, или можно просто заменить байт «RESTART 10» какой-то другой процедурой, удовлетворяющей требованиям принтера другим способом. Информация, приведенная ниже, позволяет вам решить, что сделать в конкретной ситуации или как модифицировать программу, чтобы достичь желаемого результата.

После того, как вы сделаете необходимые изменения в программе и убедитесь, что принтер работает нормально, вам надо вернуться в бейсик и записать модифицированную программу на внешний носитель для дальнейшего использования без переналадки.

После того, как вы «персонализировали» свою программу своими изменениями, вам надо задать «базовое» время таймера, а затем записать программу, используя строку 220 для ленты или 320 для микродрайва.

Интерфейс ZX LPRINT III. Дополнительных кодов не надо

Для работы этого интерфейса необходимо в бейсике до первого обращения к принтеру выполнить определенную последовательность выражений, в результате которой любое обращение к 3 каналу пойдет на строчный принтер. Это может быть выполнено в любое время, желательно при первой загрузке программы. Вам, может быть, также потребуется ввести адрес с длинной строки, которую вы хотите печатать (132). В программе управления печатью введите адрес вызова 0 и пересмотрите, если необходимо, коды «возврат каретки» и «перевод строки», а затем введите 1 в качестве стандартного префикса к операторам управления принтером.

Интерфейс типа TASPRINT. Необходим собственный кодовый блок

Этот тип интерфейса работает от собственного блока кодов, который должен быть установлен в буфере ZX-принтера до начала работы интерфейса. Этот блок залает адрес вызова канала 3 при обращении по адресу 23296 и, хотя ему это необходимо только один раз, проще и надежнее обращаться к нему всякий раз при обращении к принтеру. Этот кодовый блок называется «TASBUFF» и загружается через бейсик до редактора. Адрес вызова 23296 вводится в программу управления печатью. Введите также в соответствии с требованиями принтера байты «ВК» и «ПС» и установите стандартный префикс операторов управления печатью — 27.

Интерфейс типа KEMPSTON

Коды для этого интерфейса уже записаны в редакторе, но необходимо выполнить небольшие изменения для того, чтобы они работали вместо байта «RESTART 10», предназначенного для вызова принтера по установленному каналу. Фактически, два первых байта кемпстоновских кодов забиты байтами «RESTART 10», поэтому все, что вам надо сделать — это вернуться в бейсик после загрузки программы и выполнить два прямых ввода:

```
POKE 65176,245 : POKE 65177,1
```

Теперь интерфейс будет работать правильно. Введите 0 в программу управления печатью и стандартные байты «ВК» и «ПС» для вашего принтера. Вам также надо определить значение префикса, чтобы наиболее полно использовать операторы управления печатью.

Интерфейс типа MOREX

Коды для этого интерфейса не содержатся в редакторе, но те 18 байт, которые вам потребуются, легко вводятся в пространство, занимаемое кемпстоновскими кодами, см. выше. Вам надо написать простую бейсик-программу и запустить ее после загрузки редактора. Например такую:

```
10 DATA 245,219,251,230,1,32,250,241
20 DATA 211,251,62,1,211,127,175,211,127,201
30 FOR N=0 TO 17: READ A: POKE 65176+N,A
40 NEXT N: STOP
```

Теперь вам надо установить адрес вызова 0 в программе управления печатью и задать коды «ВК» и «ПС» в соответствии с требованиями принтера. Можно также для удобства ввести префикс операторов управления печатью.

Другие интерфейсы. Общий обзор

Если у вас есть интерфейс, который не вошел в вышеприведенное описание, то вам необходимо выполнить некоторые приготовления для его работы в соответствии с литературой, прилагающейся к нему. Вы можете применять два основных способа. Можно использовать 256 байтное пространство буфера ZX-принтера для размещения некоторых кодов, которые будут работать как посредники между компьютером и интерфейсом (это пространство — от 23296 до 23551 включительно) или вы можете для этой цели использовать пространство, занимаемое кемпстоновскими кодами. Важным при этом является принять байт «RESTART 10» в 65176 или его отвергнуть. Этот байт является первым в последовательности из 27 байт. При этом помните, что всякий байт, подлежащий передаче на принтер, будет храниться в регистре А процессора Z-80 и обращение по этому адресу используют этот байт и вернется в TLW.

Приложение 4. Использование текстов, полученных в других редакторах

Любой текстовый файл из любой программы может быть загружен в данный редактор, если он содержится в формате кодов на ленте или в формате данных на микродрайве, но при этом не будет работать изображение на экране и выдача его на принтер до тех пор, пока не будут введены управляющие байты, маскирующие концы строк текста и концы абзаца. Это байты 14 — «ПС» и байт 13 — «конец абзаца». Если программа, которую вы предполагаете использовать, предварительно не размечена таким образом, то ее надо отмаркировать. Для этой цели прилагается программа TLW64BASIC. Лучше всего ее применять до загрузки редактора, так как она при выполнении маркировки очищает память компьютера. Затем производится запись переработанного текста на ленту и последующая его загрузка в редактор.

Программа TLW64BASIC используется следующим образом:

1. Очистите память (RANDOMIZE USR 0) и загрузите TLW64BASIC. Она загрузит 75 байт TLW64CODE и опустит RAMTOP компьютера до 29999, что текстовый файл начнется с 30000.

2. Теперь вы можете присвоить файлу имя и загружать ваш текст в компьютер с магнитофона или микродрайва.

3. Программа запросит от вас размер этого файла. Если вы не знаете, то программа может сама вам сообщить его, а также «стандартную» длину строки данного текста (обычно 32 или 64)

4. Сейчас вы должны решить, будете ли вы во всех строках ставить маркер «конец строки» или «конец абзаца». Сама программа слишком мала, чтобы решить этот вопрос в каждой строке конкретно.

Обычно применяют «конец строки», если заранее не известно, что текст состоит из отдельных, не связанных предложений и ему не потребуется делать команду «REFORM». Введите «Y» для «конца абзаца» или просто ENTER для «конца строки».

5. Когда все выполнено правильно, программа сообщит вам новую длину файла и предложит записать текст на ленту для последующей загрузки в TLW.

Приложение 5. Полезные советы

Если вам надо, чтобы текст начинался с 1-ой колонки, тем не менее установите поля пусть небольшим, но положительным числом, например 5. Это позволит вам потом по команде сдвига текста влево расширить интервалы между словами и вставить пропущенный символ или два, так как у вас всегда будет запас в 5 пробелов. Если необходимо большее пространство, то установите величину правого поля достаточно большую, затем выравняйте текст по правому полю, в результате чего создадутся пробелы, затем сдвигом

текста влево/вправо подготовьте место для вставки, вставьте то, что надо, восстановите поля и выполните команду REFORM для ликвидации беспорядка. Конечно вставку можно сделать и с помощью команды INSERT, но если вы ее уже используете, то данный подход окажется очень полезным.

Если при разбивке экрана 60 или 80 знаков в строке включены видеомаркеры, то иногда бывает трудно разобрать на экране символы, если они идут за операторами управления печатью, вставленными в текст. Временно отключите маркеры, затем после прохождения курсора по этому участку текста вновь включите.

Если вы хотите приостановить принтер для смены бумаги и т.п., вы можете сделать это где угодно установкой оператора управления печатью с обращением к бейсиковской строке, например:

```
9200 INPUT «PAUSING! ENTER CONTINUES»; Z$;
GO TO 9999
```

Если вы имеете текстовый файл очень ценного материала, а при попытке записать его на микродрайв прошло сообщение SPACE!, то это значит, что место для создания соответствующего 525-байтного канала недостаточно так как редактор защищает себя от потенциального затирания. Чтобы освободить место, спуститесь в бейсик и очистите все переменные, например: RUN 30000 и попытайтесь еще раз. Другой способ — переместить курсор в начало текстового файла и выполнить INSERT, чтобы спрятать его целиком, пересмотреть вверх ненамного значение RAMTOP подачей более высокого значения CLEAR, чем было до этого, затем вернитесь в редактор и наберите первый символ «спрятанного» текстового файла, а затем верните файл на прежнее место. Первый способ предпочтительнее, но с помощью второго можно выйти из более серьезных неприятностей.

Дополнение 1. 2-я версия программы TLW

Вторая версия редактора включает в себя ряд дополнений, расширяющих возможности его применения. Так, в частности, реализована возможность эксплуатации программы в дисковых системах опус, кемпстон, гордон, а также в картриджной системе WAFA. Подробности приведены ниже. Изучать их следует совместно с основным описанием.

1. Страница сводки команд

Все 60 команд могут быть вызваны на экран набором режим E «H», там же изображаются комбинации клавиш, необходимых для выполнения этих команд. Эту страницу можно выдать на принтер, используя для этой цели программу TLW2.DATA, которая входит в прилагаемую кассету.

2. Цвета экрана

Ранее существовавшие команды для изменения цвета бордюра и хэдера заменены одной, вызываемой режим E CAPS SHIFT «B», с помощью которой обеспечивается доступ к установке цвета страницы, символов (для увеличения их яркости прибавьте 64), бордюра и хэдера.

3. Красная строка

Команда TAB была несколько расширена и теперь за ней следует оператор INSET и число от 0 до 8. Набранный текст хранится в файле обычным путем, но если проходит команда REFORM то первая строка каждого абзаца сдвигается вправо на величину INSET.

4. Трансляция символов, выдаваемых на экран или принтер

Сделано дополнение, позволяющее отдельно осуществлять трансляцию восьми символов между экраном и текстом, а также между принтером и текстом.

Обычно при нажатии данной клавиши на клавиатуре всегда один и тот же символ появляется на экране, но до восьми символов могут быть пересмотрены так, что, например, при нажатии «#» на экране изобразится «&» и т.п. Но это не влияет на результат, который пойдет на принтер.

До восьми значений байтов символов могут быть пересмотрены для выдачи на печать.

Обе эти трансляции выполняются с помощью TLW2.DATA, которая не входит в основную программу из соображений экономии памяти, но легко сливается с ней оператором MERGE. Проведенная трансляция записывается на ленту с кодовой частью программы и загружается всякий раз, когда необходимо.

5. Выбор паузы между печатью копий на принтере

Теперь вы можете печатать до 254 копий без необходимости нажимать ENTER после каждой копии. Это выполняется путем переключения паузы на OFF в хэдере. (в подпрограмме подготовки печати введите «R»).

6. Надстраничные и подстраничные записи, нумерация страниц во время печати

Для управления принтером во время печати используются специальные операторы. В их число входят операторы «BK» и «ПС». Обычно первый из них установлен как 13, а второй — 10. В новой версии редактора вы можете вводить их с помощью операторов управления печатью «вставить стринг» и «GO TO 9000» и т.п. Так, например, если A$ или какой-либо другой стринг, которым вы пользуетесь, задан как CHR$13, а строка 9000 (или другая) — LPRINT CHR$10 с последующим возвратом в редактор через строку 9999, то результат будет тот же самый. Но зато теперь вы можете прерывать выдачу текста на принтер в конце каждой строки, организовав один или два счетных цикла для организации печати из бейсика заранее заготовленных данных, а параметр цикла использовать, например, при подсчете номера страницы.

7. Объединение файлов во время печати

Системные переменные редактора, обозначающие начальный и конечный адреса файла могут пересматриваться из бейсика. Так вы можете объединять большое количество файлов и последовательно выдавать их на печать целиком или частично.

Первый файл, подлежащий печати, должен содержать оператор «GO TO 9900». Этот файл будет печататься нормально, пока не встретится оператор, после чего управление будет передано избранной строке бейсика.

В этой строке должна быть, например, такая запись:

```
9900 LET A=USR 64978: LOAD *M;1;NEWFILECODE
A: RANDOMIZE USR
65010: GO TO 9999
```

Первый оператор полностью очищает пространство текстового файла, второй осуществляет загрузку второго файла в программу, начиная с адреса, установленного A. Третий устанавливает системные переменные редактора, отвечающие за начало и конец печати. И, наконец, возврат в кодовую часть редактора. Теперь новый файл будет напечатан так, как это необходимо. Он тоже может содержать определенный оператор управления печатью для вызова последующего файла и т.д.

8. Другие системы хранения файлов

Новая версия редактора легко перенастраивается на работу в дисковых системах. Прилагаемая кассета содержит отдельно кодовые блоки дисковых систем опус, кемпстон, гордон, бета и для картриджной системы WAFA.

После того, как вы введете редактор в ваш компьютер, войдите в бейсик и загрузите оттуда соответствующий блок для вашей системы. Заметьте, что кроме этого в каждом случае вы должны добавить или переделать несколько строк в бейсике. В частности, четырехтысячные строки должны быть точно такими, как приведенные ниже:

Кемпстон KDOS — «KDOS2CODE»

```
40   CLEAR 26500:GO SUB 100: PRINT #4: LOAD
TLW2CODE: GO TO 1000
70   PRINT #4: SAVE AUTO LINE 40
80   PRINT #4: SAVE TLW2 CODE 50000,15535
4000 REM LDOS LINES
4010 PRINT #4: SAVE 1234567890 CODE 0,0: PRINT
1: PRINT USR 8
4020 PRINT #4: CAT: PRINT USR 8
```

Замечание: PEEK 50067 В 50113 должны быть по 72, а не по 69. Если не верно, то введите правильное значение посредством POKE.

Бета — «BETA2CODE»

```
40   CLEAR 26500: GO SUB 100: RANDOMIZE USR
15363: REM:LOAD TLW2 CODE
45   GO TO 1000
70   RANDOMIZE USR 15363: REM: SAVE BOOT
LINE 40
80   RANDOMIZE USR 15363: REM: SAVE TLW2
CODE 50000,15535
4000 REM BETA LINES
4010 RANDOMIZE USR 15363: REM:
SAVEA;12345678 CODE 12345,12345
4020 PRINT USR 8
```

Опус — «OPUS2CODE»

```
2020 IF I$=Y OR I$=Y THEN OPEN #3;B: LPRINT
(для порта принтера), а затем уберите (DELETE) строки
2025 ,2030,2040,2050
4000 REM OPUS LINES
4010 SAVE *M;1;1234567890 CODE 0,0: PRINT USR 8
4020 CAT 1: PRINT USR 8
```

Гордон — «GRDN2CODE»

```
40    CLEAR 26500: GO SUB 100: LOAD
D1TLW2CODE: GOTO 1000
70    SAVE D1TLW2B LINE 40
80    SAVE D1TLW2CODE 5C000,15535
4000  REM GORDON LINES
4010  D11234567890 CODE 0,0: PRINT USR 8
4020  CAT1: PRINT USR 8
```

WAFA — «WAFA2CODE»

Сначала, без подключения WAFA-драйва, загрузите редактор с кассеты и измените адрес CLEAR в строке 30 с 26500 на 28500. Перепишите программу на ленту. Отключите питание. Подключите WAFA. Загрузите новую программу, затем:

```
40    CLEAR 28500: GO SUB 100:
LOAD#TLW2,50000: GOTO 1000
70    SAVE #TLW2B LINE 40
80    SAVE #TLW2,50000,15535
2020  IF I$=Y OR I$-Y THEN
FORMAT#R;2400;OPEN#B$,R:
LPRINT (для RS232)
2020  IF I$=Y OR I$=Y THEN OPEN #*C: LPRINT
(для центроникса)
в обоих случаях уберите (DELETE) строки 2025, 2030,
2040, 2050
4000  REM WAFA LINES
4010  SAVE #A:1234567890,0,0: PRINT USR 8
4020  CLS: CAT *A: PRINT USR 8
```

Замечание: когда вы работаете с программой TLW64BASIC, вам надо убрать все бейсиковские команды, связанные с лентой/диском, чтобы задействовать дополнительные 2К системных переменных WAFA.

Интерфейс кемпстон «S»

После полной загрузки редактора наберите следующие прямые команды:

```
POKE 64954,245: POKE 64955,1
```

затем замените строку 2000 на:

```
2000 PAUSE 50
```

и уберите все остальные двухтысячные строки. Затем установите префикс PCT, «ВК», «ПС» так, как это необходимо для вашего принтера.

Интерфейс-1, порт RS232

После полной загрузки редактора наберите следующие прямые команды:

```
POKE 64954,207: POKE 64955,30: POKE 64956,201
```

измените строку 2000:

```
2000 PAUSE 50: FORMAT B;19200: LPRINT
```

остальные двухтысячные строки уберите.

После того, как вы выполнили все предписания и смогли задействовать вашу дисковую или иную систему, запишите программу на ленту вместе с внесенными в нее изменениями, для этого используйте команду RUN 70.

Прикладная программа «TLW2.DATA»

Эта программа включает в себя раздел для печати сводки команд — RUN 7000, а также сводку операторов управления печатью PCT — RUN 7100 и имеет подпрограмму, позволяющую вам выполнить трансляцию символов для экрана и для печати — RUN 7400. Эта программа может быть слита с программой с помощью команды MERGE. Предварительно надо выполнить CLEAR 35000.

Устранение маркера конца строки и пр.

Функция DELETE была несколько расширена. Теперь, когда курсор находится на строке нулевой длины, т.е. состоящей только из маркера «конец строки», то действие команды DELETE приводит к полному изъятию строки.

Длина строки в новой версии была увеличена до 148 символов. Изменение можно внести изменением POKE в строке 2050. Заметьте, что такая длина строки не для всех интерфейсов доступна и может быть ими понижена, если вы зададите значение выше 128.

Префикс операторов управления принтером выдается на сводке в качестве оператора 0.

В отличие от первой версии программы сводка операторов PCT переделана так, чтобы соответствовать интерфейсу кемпстон-Е вместо ZX LPRINT III. Если у вас интерфейс ZX LPRINT III, то вам надо вернуть назад значения первых байтов в соответствии с данным руководством. Если вы хотите отдельно записать файл операторов PCT, то обратите внимание на то, что начальный адрес был изменен, см. список системных переменных.

Внимание!

Во время работы принтера, когда по оператору управления печатью производится обращение к бейсик-строке, может быть ситуация, в которой оператор INPUT работает неправильно. Для предотвращения этого, войдите в бейсик и оттуда поменяйте значение в ячейке 58282 на 0. Больше ни на что в программе это не повлияет.

Измененная сводка системных переменных

блок в машинных кодах	от 50000 до 65535
адрес вызова	52410
адрес возврата при печати	65154
режим работы процессора	с прерываниями 2-го типа
размер текстового файла	от 0 до 24К, начиная от RAMTOP+2
файл операторов PCT	от 52193 до 52408
формат: 24 оператора, каждый из них включает в себя:	
байты от 1 до 5 — коды ASCII	
байты от 6 до 9 — коды от 0 до 254	
стандартный префикс —	от 0 до 254 52409,1 байт
операторы управления печатью 1...24	значения от 128 до 151
операторы управления печатью 90 — 99	значения от 152 до 161
операторы управления печатью А — Z	значения от 230 до 255
блок кодов интерфейса принтера (под интерфейс кемпстон-Е)	от 64954 до 64977
память калькулятора	10 переменных от М0 до М9
переменные калькулятора	как в бейсике
длина включаемых стрингов	до 4К
имена включаемых стрингов	от А$ до Z$

Некоторые системные переменные

адрес очистки текстового файла	64978
установка таймера	65189
вершина пространства текстового файла	65193, 2 байта
начало текстового файла	65195, 2 байта
список транслируемых символов экрана	от 65157 до 65165
список транслируемых символов принтера	от 65173 до 65181
код «ВК»	65241
код «ПС»	65242
строка бейсика для выхода	65244, 2 байта
величина разбивки экрана	65254
адрес вызова принтера	65257
последний байт текстового файла	65268
адрес курсора в файле	65270
колонка курсора в тексте	65272
строка курсора в тексте	65272, 2 байта
левое поле	65281
правое поле	65282
интервал между строк	65285
количество печатных копий	65286
установка величины табуляции	65289

Дополнение 2. Введение русского шрифта в редактор

Возможность введения русского шрифта в текстовый редактор является острой необходимостью для тех, кто хочет широко использовать его в повседневной работе. Надо сразу отметить, что печатающее устройство (принтер), подключенный к компьютеру и программно увязанный с редактором может печатать любым шрифтом из заложенных в его ПЗУ. Для этого достаточно из редактора в нужных местах подать коды управления принтером для переключения шрифта на принтере. Тем не менее хотелось бы, чтобы и на экране текст распечатывался русскими буквами.

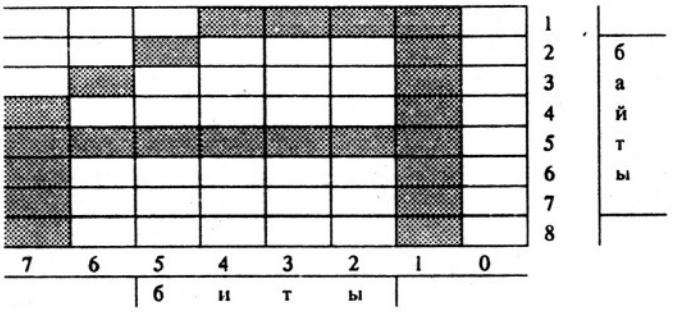
К сожалению, фирма, разрабатывавшая редактор TLW, ничего не сделала в программе для того, чтобы упростить переделку шрифта на национальный и даже более того, максимально затруднила возможность такой переделки, оставив за собой возможность поставки программы со встроенным по заказу шрифтом за особую плату.

Мы разработали методику встраивания в редактор генератора русского шрифта, позволяющий одновременно сохранить и имеющийся в редакторе латинский шрифт. В результате такой переделки редактор становится чрезвычайно мощным средством для подготовки и редактирования комплексных документов. Все свои разработки, в том числе и эту, Инфорком готовит с помощью такой русифицированной версии редактора.

Наибольшую трудность при введении дополнительного шрифта представляет нестандартный генератор символов, позволяющий получать вместо стандартных 32-х символов в строке 40, 48, 60, 80 символов.

Работа генератора символов при печати 40...80 знаков

Стандартное изображение символа представляет из себя шаблон размером 8x8 пикселей и задается с помощью 8 байт. Рассмотрим, например, как изображается буква А.



Каждой строке этого шаблона соответствует один байт, состоящий из 8 бит. Так, для первой строки этого шаблона (первого байта) включенными являются биты 4, 3, 2, 1, а для второй строки биты 5 и 1.

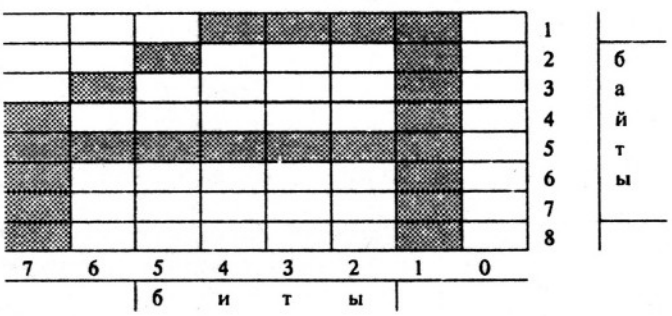
В этом случае первый байт равен:

2^4+2^3+2^2+2^1 = 16 + 8 + 4 + 2 = 30, а второй:

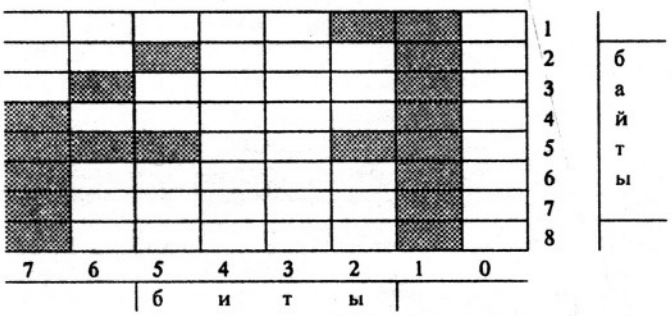
2^5+2^1 = 32 + 2 =34

В редакторе текстов TLW в режиме 40 знаков для каждого символа выделяется шаблон шириной 6 пикселей, в режиме 48 — 5 пикселей, в режиме 60 — 4 пикселя и в режиме 80 — всего 3 пикселя на каждый символ. Изображение на экране символа, имеющего нестандартную ширину, выполняется в три приема. Рассмотрим их на примере печати той же буквы А в режиме 48 знаков в строке.

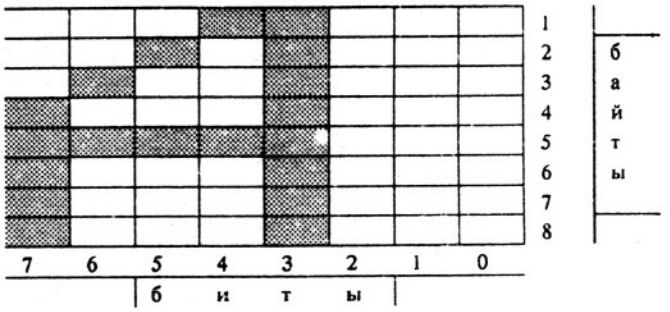
1. На первом этапе выполняется нормальная печать буквы А в отведенном для нее знакоместе.



2. Затем в изображении буквы «вырезаются» три лишние столбца, чтобы осталось необходимых 5 пикселей по ширине символа. Пусть это будут столбцы 4, 3, 0. В программе «вырезание» столбцов выполняется просто выключением всех пикселей, находящихся в этих столбцах. Тогда изображение буквы примет следующий вид:



3. На третьем этапе изображение сдвигается влево до упора:



В итоге вы получаете изображение буквы А, имеющее в ширину 5 пикселей. Следующая буква будет начинаться в следующем знакоместе экрана размером 8x8, но после операций «вырезания» столбцов и сдвига влево перейдет частично в знакоместо буквы А и т.д.

Таким образом, для режима 40 знаков из стандартного изображения необходимо вырезать 2 столбца, для режима 60 — 4 столбца и для режима 80 — 5 столбцов.

Чтобы программа знала, какие столбцы можно вырезать в каждой букве, она имеет две таблицы. Длина каждой — по 96 байт (т.к. шрифт содержит 96 символов). Первая таблица служит для режимов 40 и 48 и находится в адресах с 63100 по 63195, а вторая — для режимов 60 и 80 и находится в адресах с 63198 по 63293. Поскольку эти таблицы являются одномерными, то правильнее их назвать не таблицами, а векторами.

Т.к. изображение цифр, знаков препинания и символа «пробел» при переходе с латинского шрифта на русский не меняются, вам придется вносить изменения только в ячейки:

С 63133 по 63195 для 1-го вектора и

С 63231 по 63293 для 2-го вектора.

Номера вырезаемых столбцов для каждой буквы задаются в этих векторах однобайтным числом. Так, например, если для какой-то буквы в режиме 48 вы хотите, чтобы вырезались столбцы 4, 5, 0, то в соответствующей ячейке 1-го вектора надо поместить число 49 т.к. 2^5+2^4+2^0 = 32 + 16 + 1 = 49.

Если же вам надо, чтобы в режиме 80 для какой-то буквы вырезались столбцы 7, 5, 4, 3, 0, то в соответствующей ячейке 2-го вектора должно храниться число 185, т.к.

2^7+2^5+2^4+2^3+2^0 = 128 + 32 + 16 + 8 + 1 = 185

Введение русского шрифта

Обычно в программе можно изменить шрифт, если разместить шрифт, нужный вам, в известных вам адресах и изменить двухбайтовую системную переменную CHARS, находящуюся в адресах 23606 и 23607, так, чтобы она указывала на адрес, находящийся на 256 байт ниже, чем начало нужного вам шрифта.

В редакторе TLW этот метод не срабатывает, т.к. редактор сам, из своего машинного кода обращается к шрифту, не прибегая для этой цели к услугам системной переменной CHARS. Изменить это обращение можно внесением изменения в машинный код по адресам 63661 и 63662 командой POKE.

Для латинского шрифта в этих ячейках содержится значение 0 и 61, соответственно. Для русского шрифта надо подставить свои числа. Например, если вы хотите, чтобы русский шрифт начинался с адреса 30000, то вам надо заслать числа:

POKE 63661,48: POKE 63662,117

Обратите внимание что 117*256+48 = 30000.

При возврате же на латинский шрифт надо, естественно, опять восстановить исходные значения.

Соответствие между буквами лат. и рус. алфавитов

Решая проблемы, связанные с введением русского шрифта в редактор, вам придется и решить для себя вопрос о том, какой русской букве какая клавиша компьютера будет соответствовать. В принципе, это дело произвольное и выбор может быть за вами, но все же лучше это назначение сделать обдуманно, если вы предполагаете подключать к компьютеру принтер и работать с редактором и принтером. Дело в том, что в наборе шрифтов принтера за каждым знаком (буквой) уже закреплен какой-то код и вам надо сделать так, чтобы в редакторе этому же знаку соответствовал такой же код. Тогда принтер сможет распечатывать точно то, что написано на экране.

В компьютерах «Спектрум» принят американский стандарт кодов ASCII, согласно которому за каждым кодом от 32 до 127 закреплен определенный символ. Коды от 0 до 31 — служебные и управляющие, коды от 128 до 255 этот стандарт не устанавливает, здесь каждая фирма может разместить то, что хочет. Синклер разместил коды графических символов (от 128 до 143), графику пользователя (от 144 до 164) и так называемые токены ключевых слов

(от 165 до 255). Так, например, код 239 — это токен LOAD, а код 248 — SAVE. Кстати, именно благодаря этому при нажатии одной клавиши в программу вводится сразу все ключевое слово и его не надо набирать по буквам.

В СССР коду ASCII соответствует код КОИ-7/Н0. Он содержит знаки латинского алфавита. Имеется также код КОИ-7/Н1, содержащий знаки русского алфавита. Многие принтеры кроме латинского имеют и русский шрифт, работают в этих стандартах и могут гибко, из программы, с помощью кодов управления принтеров переключаться с латинского шрифта на русский и наоборот. Поэтому наиболее удобно при адаптации редактора сразу установить такое же соответствие между латинской клавиатурой и русскими буквами, какое существует между стандартами КОИ-7/Н0 и КОИ-7/Н1.

В принципе таблицы соответствия имеются в инструкциях к принтерам, но для тех, кто их не имеет, мы здесь приведем справочную таблицу.

Соответствие между латинскими и русскими символами

код	ASCII (КОИ-7/НО)	КОИ-7/Н1
32		пробел
33	!	!
34	"	"
35	#	#
36	$	$
37	%	%
38	&	&
39	'	'
40	(	(
41	)	)
42	*	*
43	+	+
44	,	,
45	-	-
46	.	.
47	/	/
48	0	0
49	1	1
50	2	2
51	3	3
52	4	4
53	5	5
54	6	6
55	7	7
56	8	8
57	9	9
58	:	:
59	;	;
60	<	<
61	=	=
62	>	>
63	?	?
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
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	_	'
96	ФУНТ СТЕРЛ.	Ю
97	a	а
98	b	и
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	м
122	z	з
123	<	ш
124	I	э
125	>	щ
126	-	ч
127	,	

В соответствии с этой таблицей ваша задача при создании русского шрифта обеспечить, чтобы, например, 47-м шаблоном вместо шаблона буквы «Q» стоял шаблон буквы «Я» и т.д.

Конструкция букв русского алфавита

Поскольку в результате «вырезания» нескольких столбцов буквы приобретают не вполне приемлемый вид, целесообразно загружаемый шрифт готовить специально с учетом того, что столбцы будут вырезаться. Так, например, буквы не должны иметь никаких декоративных элементов, желательно избегать закруглений. Наиболее удобно пользоваться для подготовки шрифта специализированными программами, например, графическим редактором «ARTSTUDIO», но для тех, кто таких программ не имеет, остается загрузить шрифт вручную с клавиатуры. Можете, например, загрузить наш шрифт. Для этого наберите и запустите (RUN) нижеприведенную программу. Когда она отработает, выгрузите сформированный шрифт на ленту. Этот блок кодов расположен, начиная с адреса 30000 (запомните этот адрес) и имеет длину 768 байт (96 символов по 8 байт).

```
10  FOR I = 1 TO 768
20  READ A
```

```
30  POKE (29999+I),A
40  NEXT I
50  SAVE RUS CODE 30000,768
100 DATA 0,0,0,0,0,0,0,0
101 DATA 0,16,16,16,16,0,16,0
102 DATA 0,36,36,0,0,0,0,0
103 DATA 0,36,126,36,36,126,36,0
104 DATA 0,8,62,40,62,10,62,8
105 DATA 0,98,100,8,16,38,70,0
106 DATA 0,16,40,16,42,68,58,0
107 DATA 0,8,16,0,0,0,0,0
108 DATA 0,4,8,8,8,8,4,0
109 DATA 0,32,16,16,16,16,32,0
110 DATA 0,0,20,8,62,8,20,0
111 DATA 0,0,8,8,62,8,8,0
112 DATA 0,0,0,0,0,8,8,16
113 DATA 0,0,0,0,62,0,0,0
114 DATA 0,0,0,0,0,24,24,0
115 DATA 0,0,2,4,8,16,32,0
116 DATA 0,60,70,74,82,98,60,0
117 DATA 0,24,40,8,8,8,62,0
118 DATA 0,60,66,2,60,64,126,0
119 DATA 0,60,66,12,2,66,60,0
120 DATA 0,8,24,40,72,126,8,0
121 DATA 0,126,64,124,2,66,60,0
122 DATA 0,60,64,124,66,66,60,0
123 DATA 0,126,2,4,8,16,16,0
124 DATA 0,60,66,60,66,66,60,0
125 DATA 0,60,66,66,62,2,60,0
126 DATA 0,0,0,16,0,0,16,0
127 DATA 0,0,16,0,0,16,16,32
128 DATA 0,0,4,8,16,8,4,0
129 DATA 0,0,0,62,0,62,0,0
130 DATA 0,0,16,8,4,8,16,0
131 DATA 0,60,66,4,8,0,8,0
132 DATA 0,0,72,84,116,84,72,0
133 DATA 0,0,56,72,72,120,72,0
134 DATA 0,48,64,112,72,72,120,0
135 DATA 0,0,72,72,72,72,120,8
136 DATA 0,48,8,120,72,72,120,0
137 DATA 0,0,48,72,120,64,56,0
138 DATA 0,0,56,84,84,56,16,0
139 DATA 0,0,120,64,64,64,64,0

140 DATA 0,0,72,48,48,48,72,0
141 DATA 0,0,72,88,104,72,72,0
142 DATA 0,16,72,88,104,72,72,0
143 DATA 0,0,72,80,96,80,72,0
144 DATA 0,0,56,72,72,72,72,0
145 DATA 0,0,68,108,84,84,68,0
146 DATA 0,0,72,72,120,72,72,0
147 DATA 0,0,48,72,72,72,48,0
148 DATA 0,0,120,72,72,72,72,0
149 DATA 0,0,56,72,72,56,72,0
150 DATA 0,0,112,72,72,112,64,0
151 DATA 0,0,48,72,64,72,48,0
152 DATA 0,0,124,16,16,16,16,0
153 DATA 0,0,72,72,120,8,120,0
154 DATA 0,0,84,56,16,56,84,0
155 DATA 0,0,120,72,120,72,120,0
156 DATA 0,0,64,112,72,72,112,0
157 DATA 0,0,68,116,76,76,116,0
158 DATA 0,0,120,8,56,8,120,0
159 DATA 0,0,84,84,84,84,124,0
160 DATA 0,0,48,72,24,72,48,0
161 DATA 0,0,84,84,84,84,124,4
162 DATA 0,0,72,72,56,8,8,0
163 DATA 0,0,96,56,40,40,56,0
164 DATA 0,92,84,116,84,84,92,0
165 DATA 0,56,72,72,120,72,72,0
166 DATA 0,120,64,120,72,72,120,0
167 DATA 0,72,72,72,72,72,120,8
168 DATA 0,56,72,72,72,72,120,72
169 DATA 0,120,64,120,64,64,120,0

170 DATA 16,124,84,84,84,124,16,0
171 DATA 0,120,64,64,64,64,64,0
172 DATA 0,72,48,48,48,48,72,0
173 DATA 0,72,72,88,104,72,72,0
174 DATA 16,72,72,88,104,72,72,0
175 DATA 0,36,40,48,48,40,36,0
176 DATA 0,56,72,72,72,72,72,0
177 DATA 0,68,108,84,68,68,68,0
178 DATA 0,72,72,120,72,72,72,0
179 DATA 0,48,72,72,72,72,48,0
180 DATA 0,120,72,72,72,72,72,0
181 DATA 0,56,72,72,120,40,72,0
182 DATA 0,120,72,72,120,64,64,0
183 DATA 0,120,64,64,64,64,120,0
184 DATA 0,124,16,16,16,16,16,0
185 DATA 0,72,72,72,120,8,56,0
186 DATA 0,84,84,84,56,84,84,0
187 DATA 0,120,72,120,72,72,120,0
188 DATA 0,64,64,120,72,72,120,0
189 DATA 0,68,68,116,76,76,116,0
190 DATA 0,120,8,8,56,8,120,0
191 DATA 0,84,84,84,84,84,124,0
192 DATA 0,48,72,24,8,72,48,0
193 DATA 0,84,84,84,84,84,124,4
194 DATA 0,72,72,72,120,8,8,0
195 DATA 60,66,153,161,161,153,66,60
```

Дополнительные сведения

Создатели программы THE LAST WORD по-видимому сочли неудовлетворительной конфигурацию некоторых букв шрифта, находящегося в ПЗУ компьютера и встроили специальные процедуры для изображения букв «A» — код 97, «D» — код 68, «Y» — код 89. При переходе на русский шрифт эти процедуры надо отключить:

POKE 63669,250: POKE 63676,250: POKE 63683,250

а при возврате к латинскому шрифту — подключить снова:

POKE 63669,97: POKE 63676,68: POKE 63683,89

Качество отображения символов в режиме 80 знаков в строке далеко не удовлетворительное для латинского шрифта, а для русского — еще хуже. Изобразить буквы ЖБ, ШБ, ЩБ, ЫБ, Ю и т.п. шириной в 3 пикселя конечно невозможно. Поэтому этот режим может применяться только для прикидочного взгляда на сформированную страницу текста, а основная работа производится в режимах 40 и 48 знаков.

Порядок переделки программы на русский шрифт

1. Загрузите фирменную программу TLW2.

2. На вопрос о том, подключен ли принтер, ответьте «N». После этого на экране появится рабочее окно редактора.

3. Перейдите в режим 40 знаков в строке (режим E + «V»). На экране появится запрос VIDEO: дайте в ответ число 40. Выйдите в бейсик (режим E + «B»). Теперь можно вносить изменения в программу.

4. Чтобы обеспечить место для изменений, дайте прямую команду CLEAR 31000. Если вы этого не сделаете, то начиная с какого-то момента программа откажется принимать те изменения, которые вы в нее вводите. Теперь в строке 30 поднимите адрес RAMTOP, задаваемый оператором CLEAR до 31000. Это нужно для того, чтобы, во-первых, разместить русский шрифт, начиная с адреса 30000, а во-вторых потому, что размер бейсиковской части программы существенно увеличится после внесения предлагаемых дополнений, а для этого надо освободить место.

Далее в строке 30 поместите команду на загрузку русского шрифта, сформированного ранее.

Строка 30 примет вид (изменения подчеркнуты):

```
30 CLEAR VAL 31000:GO SUB VAL 100:PRINT
"":LOAD TLW2 CODE:
LOAD RUSCODE 30000,768:GO TO VAL 1000
```

5. Строки 40, 70, 80 предназначенные для работы с микродрайвом, можете удалить, если у вас его нет.

6. Строка 60 предназначена для выгрузки настроенной программы на ленту. Внесите в нее изменения с тем, чтобы выгружался и встроенный русский шрифт.

```
60 SAVE TLW2CODE VAL 50000, VAL15535:SAVE
RUS CODE 30000,
768: GO TO VAL90
```

7. В строке 3000, которая инициализирует программу, введите два новых параметра L и R. Это номера строк, в которых начинается переделка шрифта:

L = 7000 — русский шрифт.
R = 8000 — латинский шрифт.

```
3000 LET R=7000:LET L=8000:RANDOMIZE USR VAL
52410
```

8. Введите новые семитысячные строки, обеспечивающие печать русскими буквами:

```
7000 REM
7010 REM *** RUSSIAN LETTERS ***
7020 REM
```

ввод 1-го вектора для вырезания столбцов.

```
7030 RESTORE 7500
7040 FOR I=63133 TO 63195
7050 READ A
7060 POKE I,A
7070 NEXT I
```

ввод 2-го вектора для вырезания строк

```
7080 RESTORE 7600
7090 FOR I=63231 TO 63293
7100 READ A
7110 POKE I,A
7120 NEXT I
```

подключение русского шрифта

```
7130 POKE 63661,48:POKE 63662,117
```

отключение процедур изображения символов «A»,"Y", «D»

```
7140 POKE 63669,250:POKE 63670,250:POKE
63683,250
```

первый вектор

```
7500 DATA 131,7,7,7,7,7
7510 DATA 131,7,7,7,7,7,7
7520 DATA 131,7,7,7,7,7,7
7530 DATA 131,7,131,7,7,131,7
7540 DATA 131,7,131,7,7,131
7550 DATA 7,7,7,7,7,131
7560 DATA 7,7,7,7,131,7,131
7570 DATA 7,7,7,131,7,7,131
7580 DATA 7,131,7,7,131,7,131
7590 DATA 7,131,7
```

второй вектор

```
7600 DATA 171,167,167,167,167,167,171
7610 DATA 167,167,167,167,167,167,171
7620 DATA 167,167,167,167,167,167,199
7630 DATA 167,171,167,167,179,167,171
7640 DATA 167,171,167,151,155,167,167
7650 DATA 167,167,167,171,167,167,151
7660 DATA 151,211,167,171,167,167,167
7670 DATA 167,167,167,199,167,171,167
7680 DATA 167,169,167,171,167,171,167
7999 GO TO 3000
```

9. Введите новые восьмитысячные строки, обеспечивающие печать латинскими буквами.

```
8000 REM
8010 REM *** ENGLISH LETTERS ***
8020 REM
```

ввод 1-го вектора

```
8030 RESTORE 8500
8040 FOR I=63133 TO 63195
8050 READ A
8060 POKE I,A
8070 NEXT I
```

ввод 2-го вектора

```
8080 RESTORE 8600
8090 FOR I=63231 TO 63293
8100 READ A
8110 POKE I,A
8120 NEXT I
```

подключение латинского шрифта

```
8130 POKE 63661,0:POKE 63662,61
```

подключение процедур печати символов «A»,"D","Y"

```
8140 POKE 63669,97:POKE 63676,68:POKE 63683,89
```

первый вектор

```
8500 DATA 137,13,13,13,25,13,13
8510 DATA 21,13,82,13,49,7,137
8520 DATA 13,13,13,37,25,13,35
8530 DATA 13,137,137,137,137,67,112
8540 DATA 131,7,131,7,73,25,69
8550 DATA 67,11,11,35,11,11,67
8560 DATA 67,67,35,131,11,11,11
8570 DATA 25,67,11,67,11,131,131
8580 DATA 131,11,67,82,49,70
```

второй вектор

```
8600 DATA 185,185,185,185,185,185,185
8610 DATA 185,185,229,185,179,185,185
8620 DATA 157,185,185,173,185,185,199
8630 DATA 185,181,181,181,173,211,241
8640 DATA 199,143,199,248,211,157,217
8650 DATA 211,179,179,227,179,179,199
8660 DATA 211,203,227,227,179,179,179
8670 DATA 179,203,179,199,179,171,171
8680 DATA 179,179,199,229,227,203,227
```

10. Загрузите заранее подготовленный 768-байтный блок кодов, содержащий русский шрифт.

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

11. Теперь вы можете выгрузить созданную копию редактора на ленту командой GO TO 50.

12. Для проверки войдите в редактор командой GO TO 3000.

13. Далее выполняйте выход из редактора в бейсик через (режим E + «B»), а вход:

GO TO R — в русский шрифт;
GO TO L — в латинский шрифт;

Заключение

1. Уважаемые товарищи! Предлагаемый порядок введения дополнительного шрифта был нами оттестирован после того, как была напечатана предыдущая глава. Если вы с полным пониманием выполните все указания, программа должна работать нормально. В то же время, к нам поступают сведения о том, что на некоторых самодельных моделях компьютеров после переделки программы после нормального старта и набора 2...3 букв компьютер зависает. Мы используем только фирменные компьютеры, поэтому проверить эти сведения не имеем возможности. Причина может быть не в программе, а в компьютере.

2. После перехода на русский шрифт все надписи и сообщения редактора будут выводиться русскими буквами. На данном этапе с этим надо смириться. Более глубокие переделки редактора выходят за рамки данного пособия. Со своей стороны мы дали здесь максимум информации для тех, кто имеет возможность и желание экспериментировать сам.

Дополнение 3. Краткое дополнение по TLW 2+

Основным отличием этой версии программы от базовой является полная русификация команд, сообщений и хэдера. Программа может работать с несколькими шрифтами (как с латинскими, так и с русскими буквами). Переход на другой шрифт осуществляется с помощью 'SYMBOL SHIFT + I', после чего программа выдает запрос номера шрифта. Необходимо выбрать один из 10 шрифтов.

8.2. TASWORD II

TASWORD II обеспечивает два режима печати — 64 символа в строке и 32 символа в строке. Режим печати заглавных букв устанавливается путем нажатия CS+буква. Символы ! @ # $ % & * ( ) + ? - < >,. ; : > < \ — выводятся через SS либо CS.

1. Управляющие клавиши

При нажатии одного из SHIFT и соответствующей клавиши выполняются следующие действия:

EDIT	— вывод HELP PAGE;
CAPS LOCK	— печать заглавных букв;
TRUE VIDEO	— курсор на слово влево;
INV. VIDEO	— курсор на слово вправо;

и т.д.

Если вы работаете с HELP PAGE, то при нажатии SS+CS вы получите возможность изучить функции управляющих клавиш расширенного режима.

2. Расширенный режим

Расширенный режим устанавливается нажатием SS+CS. При этом нижняя часть экрана начнет мигать.

Управляющие клавиши в расширенном режиме.

прокрутка (на 22 строки)	форматирование
F — быстрая вниз;	E — выравнивание вправо (вкл\откл);
G — быстрая вверх;	W — перенос слова (вкл\откл);
	J — выравнивание строки;
упр. принтером	K — антивыравнивание;
P — печать файла,вых Q;	
L — маркер включения режима печати удвоенных букв;	границы
K — маркер отмены печати удвоенных букв;	A — установка левой границы по курсору;
разное	S — установка нормальных границ;

D — установка правой границы по курсору;

C — изменение окна;
X — очистка текстового файла;
R — замена\поиск текста;
блоковые команды
I — вставка вкл\выкл;
B — метка начала блока, < ;
M — пересылка блока в место курсора, > ;
(1)+(CS) — HELP PAGE;
(5-8)+(CS) — управление курсором;

Перенос слова:

Если строка не умещается в установленные границы, TASWORD II переносит последнее слово на новую строку.

Выравнивание текста.

При выравнивании текста строка дополняется пробелами так, чтобы последняя буква последнего слова строки находилась на правой границе.

Высокий курсор.

После того, как вы напечатали последний символ в строке, TASWORD II переносит курсор в начало следующей строки. При этом курсор увеличивается. При таком виде курсора TASWORD II воспринимает введенный символ как часть последнего слова в предыдущей строке и, следовательно, переносит это слово (см. «выравнивание»). Т.е. если слово в последней строке закончено, при высоком курсоре вы должны напечатать пробел.

Полезные советы

Всегда печатайте пробел в конце предложения после любого знака препинания.

Всегда печатайте новый абзац с «красной» строки или вставляйте между абзацами пустые строки (можно делать то и другое).

Копирование TASWORD II

Копирование и передача TASWORD II третьим лицам запрещены. Для копирования TASWORD II необходимо нажать управляющий символ STOP во время работы TASWORD II, при этом появится меню. Далее необходимо нажать «T» для записи TASWORD II на магнитофон.

3. Запись/чтение текстовых файлов

Для записи файла необходимо нажать STOP (SS+A) в нормальном режиме работы. Далее выбрать в меню «SAVE TEXT FILE» нажатием «S». На запрос имени файла вы должны набрать имя файла (не более 10 символов) и нажать «ENTER». После записи вашего файла TASWORD II предоставляет возможность его верификации. При появлении сообщения «TAPE LOADING ERROR» необходимо нажать «R» для «RUN» и «ENTER».

При загрузке нового файла с ленты, файл, находящийся в данный момент в TASWORD II стирается. Для загрузки файла необходимо войти в меню (по «STOP») выбрать «LOAD TEXT FILE» нажатием «J» Далее набрать имя файла и «ENTER».

В TASWORD II существует возможность догружать текст с ленты, который располагается после уже существующего текста в TASWORD II. Для этого в меню необходимо выбрать «M». При этом необходимо помнить, что максимальная длина текста в TASWORD II — 320 строк. Если это условие не соблюдено, вы окажетесь в BASIC'e. Для повторного запуска TASWORD II наберите «TASWORD II» + «RUN» + «ENTER».

4. TASWORD II TUTOR

TASWORD II TUTOR является текстовым файлом, предназначенным для обучения работе с TASWORD II.

5. Управляющие клавиши

Ниже описаны управляющие клавиши, работающие в нормальном режиме т.е. тогда, когда командные строки не мигают.

EDIT	(CS+1)	— HELP PAGE. В этом режиме можно вывести информацию о режиме расширения нажатием «SS»+«CS».
ENTER		— выход из HELP PAGE.
CAPS LOCK	(CS+2)	— режим больших букв вкл/выкл.
TRUE VIDEO	(CS+3)	— курсор в конец предыдущего слова.
INV. VIDEO	(CS+4)	— курсор в начало следующего слова.
DELETE	(CS+0)	— удаление символа над курсором.
GRAPHICS	(CS+9)	— режим печати графических символов.
<=	(SS+Q)	— сдвиг текста под курсором влево.
>=	(SS+E)	— сдвиг текста под курсором вправо.
<>	(SS+W)	— центровка строки.
AND	(SS+Y)	— вставка строки,символа

Для вставки пустой строки необходимо установить курсор в начало строки, перед которой вставляется строка, и нажать «AND».

OR	(SS+U)	— курсор в конец текста.
AT	(SS+I)	— курсор в начало текста
STOP	(SS+A)	— выход в меню. При выходе в BASIC, в редактор можно вернуться, нажав «RUN» и «ENTER».
NOT	(SS+S)	— удаление строки.
STEP	(SS+D)	— выравнивание текста от строки, содержащей курсор, до конца абзаца. Конец абзаца определяется по пробелам или пустой строке. Текст выравнивается только при включенном RIGHT JUSTFY и не выравнивается, когда R.JUSTFY выключено.
TO	(SS+F)	— сдвиг текста вниз на одну строку.
THEN	(SS+G)	— сдвиг текста вверх на одну строку.
ENTER		— переход на новую строку либо вставка пустой строки в режиме «вставки».

6. Расширенный режим

Здесь дано более подробное описание отдельных функций, описанных в пункте 2.

R — замена/поиск текста от курсора до конца текста (для поиска/замены от начала текста следует использовать AT перед входом в расширенный режим). В режиме «R» TASWORD II запрашивает слово для поиска или замены (пробелы при наборе слова не допускаются). После этого нажимают «ENTER» и TASWORD II находит в тексте найденное вами слово. Можно продолжить поиск нажав еще раз «ENTER». Для замены найденного слова необходимо набрать требуемый текст (пробелы допускаются) и нажать «ENTER». TASWORD II будет переформатировать текст в соответствии с режимом «R.JUSTIFY».

L — маркер включения печати букв с удвоенной высотой. Маркер представляет собой специальную строку, содержащую сообщение: «PRINT AT DOUBL HEIGHT ON», которая печатается перед требуемой строкой. Для удаления маркера можно использовать «NOT» в нормальном режиме. Перед включением этого режима курсор должен быть в начале строки.

K — вставка сообщения «PRINT AT DOUBLE HEIGHT OFF». Курсор должен быть в начале строки.

C — изменение «окна». Служит для открытия/закрытия 32-х символьного «окна» в текстовом файле. При закрытом «окне» длина текстовой строки равна 64 символам. Когда «окно» открыто, изменяется цвет BORDER и на экране появляется часть текстового файла шириной 32 символа в строке. «Окно» можно двигать с помощью «ARROWS». Боковой сдвиг происходит автоматически при вводе текста.

B — маркер начала блока. Блок текста может быть переслан или скопирован в любую часть текста. Для удаления маркера можно использовать «NOT» в нормальном режиме. Маркером начала блока является <.

V — маркер конца блока >.

M — пересылка блока. Блок вставляется перед строкой, содержащей курсор.

7. Печать текста

Перед печатью необходимо передать в принтер управляющие коды. Для этого необходимо выйти в меню по «STOP» в нормальном режиме и выбрать «DEFINE GRAPHICS\PRINTER» нажатием «G».

8. Графические символы

Графические символы интерпретируются TASWORD II как последовательность управляющих кодов для принтера. TASWORD II поставляется с набором графических символов, указанных в «HELP».

После входа в «DEFINE GRAPHICS\PRINTER» на экране появится список кодов графических символов ZX-SPECTRUM и список назначенных им управляющих кодов принтера. Для переназначения необходимо исправить соответствующий графический код и нажать <ENTER>.

Если вы нажали <ENTER> без набора кода, TASWORD II запросит вас:

1. Управляющий код, который использует ваш интерфейс. Введите соответствующее число (27 — для HILDERBAY, 5 — для EUROELEKTRONICS).

2. Управляющий код, который использует ваш принтер для протяжки листа и возврата каретки. Если принтер имеет одинаковые коды и для того, и для другого, введите «0» вместо второго кода.

3. Левая граница.

Ответив на эти вопросы, вы получите собственную версию TASWORD II, которую можно сохранить на ленте.

Изменение знакогенератора.

Таблица 64-х символьного знакогенератора находится в ОЗУ по адресам с 61184 по 62079, а базовым адресом является 60928. Каждый символ определяется 8-ю байтами обычным способом. Четыре старших бита каждого байта должны быть равны нулю.

За исключением графических символов, 32-х символьный знакогенератор не может быть изменен. Шестнадцать символов 32-х символьного знакогенератора хранятся по адресам 60928-61055 с базовым адресом 59904.

Глава 9. Работа на SPECTRUM-128

9.1. Общие сведения

SPECTRUM 128 может использоваться с программным обеспечением ранних моделей ZX SPECTRUM. Это означает, что вы можете использовать тысячи существующих программных пакетов: игр, научных, образовательных и др.

Нажмите клавишу RESET, тестовый сигнал исчезнет с экрана монитора и появится входное меню. Предлагается выбор:

TAPE LOADER — загрузчик с ленты.

128 BASIC — используется для программирования на BASIC 128.

CALCULATOR — калькулятор для SPECTRUM 128.

48 BASIC — используется с программным обеспечением SPECTRUM 48.

Нужная функция выбирается при помощи клавиш перемещения курсора с последующим нажатием на клавишу ENTER.

9.2. Загрузка программ и BASIC

Загрузка программы

SPECTRUM 128.

Для загрузки программы выполните следующие операции:

1. Вызовите входное меню.
2. Выберите ''TAPE LOADER''.
3. Вставьте кассету в магнитофон и перемотайте ленту в начало.
4. Нажмите клавишу ''воспроизведение'' магнитофона. На экране появится индикация начала загрузки. Если вы хотите отменить загрузку, нажмите клавишу BREAK.

После загрузки вверху экрана появится наименование программы. Программа готова к работе. В конце работы нажмите RESET, имея в виду, что при этом память очищается, поэтому в конце работы вы должны быть совершенно уверены, что программа вам больше не нужна.

SPECTRUM 48

Для загрузки программы выполните следующие операции:

1. Вызовите входное меню.
2. Выберите ''BASIC 48''.
3. Входное меню исчезнет и появится стандартное сообщение, затем наберите LOAD '' ''. Нажмите ENTER.
4. Включите магнитофон в режиме воспроизведения. На экране появится индикация начала загрузки. Если вы хотите отменить загрузку, нажмите клавишу BREAK.

После загрузки вверху экрана появится наименование программы. Программа готова к работе. В конце работы нажмите RESET, имея в виду, что при этом память очищается, поэтому в конце работы вы должны быть совершенно уверены, что программа вам больше не нужна.

Загрузка BASIC

В SPECTRUM 128 используется BASIC, язык более чем распространенный в компьютерах. SPECTRUM BASIC разработан с учетом требования простоты использования и изучения и отличается от других версий рядом аспектов. Если вы привыкли к SPECTRUM 48, вам многое покажется знакомым.

SPECTRUM 128 обладает усовершенствованным редактором для создания и модификации программ 128 BASIC. Для запуска редактора во входном меню выберите ''128 BASIC''.

В верхнем углу появится курсор, а внизу полоса с сообщением об используемом программном обеспечении, в данном случае 128 BASIC.

Нижняя от полосы часть экрана используется для сообщений системы.

Теперь нажмите клавишу EDIT. Курсор исчезнет и появится новое меню.

Выбор в меню редактора производится так же как и в предыдущем — курсором и клавишей ENTER.

128 BASIC — меню исчезает и появляется курсор.

RENUMBER — происходит перенумерация строк программы со строки 10 с шагом 10. Если перенумерация невозможна, то появляется номер строки превышающий число 9999, раздается низкий звук и происходит возврат в меню.

Для того, чтобы перенумеровать программу с другими значениями строк или если количество строк превышает число 1000, следует использовать следующую программу:

```
10 INPUT ''START LINE'', START
20 INPUT ''STEP SIZE'', STEPSIZE
30 LET HISTART = INT(START/256)
40 LET HISTEP = INT(STEPSIZE/256)
50 POKE 23444,START-256*HISTART
60 POKE 23445,HISTART
70 POKE 23446,STEPSIZE-256*HISTEP
80 POKE 23447,HISTEP
90 PRINT ''PRESS <EDIT> THEN SELECT
RENUMBER OPTION''
```

SCREEN — переводит курсор в нижнюю часть экрана и позволяет вводить и редактировать программу в нижних двух строках. Это часто полезно при работе с графикой. Возврат к верхнему экрану происходит при вторичном выборе функции SCREEN.

PRINT — если принтер подсоединен, то программа находящаяся в памяти распечатывается. По окончании распечатки меню исчезает и появляется курсор. Если распечатка по каким-либо причинам не произошла, нажмите дважды на BREAK, что приведет к возврату в редактор.

EXIT — возвращает ко входному меню. Программа в памяти сохраняется. К ней можно вернуться выбрав из меню '128 BASIC'.

Если вы выберите из входного меню '48 BASIC' или выключите компьютер, то программа утрачивается. Выбор 'CALCULATOR' не приводит к потере программы.

SPECTRUM 128 может работать в режиме SPECTRUM 48. Для этого следует выбрать 48 BASIC из входного меню. В этом режиме нельзя использовать ряд возможностей SPECTRUM 128: дополнительную память, многоканальный звук, RS232/MIDI, дополнительную клавиатуру.

Другой способ выбора 48 BASIC — набор команды:

SPECTRUM <ENTER>

Программа при этом сохраняется. Однако при этом нет другой возможности вернуться в 128 BASIC, кроме как через <RESET> или выключение питания.

В режиме 48 BASIC команды и функции можно вводить нажатием на одну из клавиш с клавишей регистра (CAPS SHIFT, SYMBOL SHIFT) — режим ключевых слов индицируется на экране курсором [K].

Режим [K] включается автоматически, сменяя режим [L] в начале строки, после ':' или после слова 'THEN'.

Режим [L] (LETTERS) работает в остальных случаях.

Режим [C] (CAPITALS) вариант режима [L], использующий только заглавные символы. Для смены режимов [L]/[C] используется <CAPS LOCK>.

Режим [E] (EXTEND), позволяющий использовать дополнительные символы, включается одновременным нажатием <CAPS SHIFT> и <SIMBOL SHIFT>, и действует только на один символ.

Длительное нажатие на клавишу (2-3 сек) приводит к повторению символов.

Редактирование текста программы производится при помощи клавиш управления курсором и клавиши <DELETE>. После нажатия <ENTER> строка вписывается в программу. В случае синтаксической ошибки появляется '?'.

Следующая за введенной строка является текущей строкой и в ней возможно редактирование. При помощи клавиши <EDIT> любую строку программы можно сделать текущей.

9.3. Руководство по программированию

Экран

На экране отображаются одновременно 24 строки по 32 символа в каждой. Верхняя часть экрана, 22 строки показывает текст программы и чаще всего используется для редактирования. При достижении последней строки экрана происходит сдвиг на одну строку вверх (SCROLL). Если в верхней строке размещался текст, то на экране возникает вопрос:

SCROLL ?

При нажатии любой клавиши, кроме N, <BREAK> или пробела происходит сдвиг экрана, если же нажимаются указанные выше клавиши, то появляется сообщение:

D BREAK - CONT REPEATS

Нижняя часть экрана используется для редактирования небольших программ, ввода команд непосредственного действия и сообщений системы.

Ввод программ

При вводе может быть показана любая часть программы командой

LIST XXX

Где 'XXX' номер строки.

При выполнении команды или программы ее результат показывается в верхней части экрана. После выполнения программы результаты сохраняются на экране до нажатия клавиши.

Если для редактирования используется нижняя часть экрана, то результаты демонстрируются в верхней части и остаются до тех пор, пока не будут перекрыты другой записью, стерты командой CLS или сдвигом экрана.

Если во время выполнения программы нажать на <BREAK>, то выполнение прерывается с сообщением:

D.. OR.. L

В этом случае программа может быть отредактирована.

Ввод программы возможен с произвольным порядком строк, т.е. сначала можно ввести строку 20, а затем строку 10. На экране они расположатся в порядке возрастания номеров.

При редактировании можно переписать строку заново, но проще подвести курсор к нужному месту, внести исправления и нажать <ENTER>.

Пуск программы происходит по команде RUN.

Для уничтожения строки достаточно набрать ее номер. При наборе номера не существующей строки курсор перемещается в то место где она должна быть в тексте программы.

Для демонстрации текста программы наберите:

LIST<ENTER>

Перенумерация строк возможна при нажатии <EDIT> и выборе из меню 'RENUMBER'.

Команда NEW уничтожает программу и все переменные.

При ошибке в работе программы, например при ошибке ввода данных появляется сообщение STOP, а при нажатии <ENTER>

H STOP IN LINE 40:1

Что означает причину остановки, номер строки и номер инструкции в строке где произошла остановка. Если вы хотите продолжить работу наберите

CONTINUE

Если вы использовали в программе инструкцию перехода к несуществующей строке (GO TO 31), то выполнится переход к следующей за ней существующей строке (напр. 40), то же касается команды RUN.

В команде PRINT используются различные символы следующие после параметра:

Так PRINT F, — запятая указывает, что вывод переменной на экран производится либо с крайней левой позиции, либо с середины экрана;

PRINT F; — точка с запятой означает, что вывод будет произведен сразу после предыдущего и на той же строке экрана.

PRINT F' или PRINT F — вывод производится с новой строки экрана.

Двоеточие применяется для разделения команд в одной строке:

PRINT F: GO TO 40

Возможна инструкция RUN 100, в отличии от GO TO 100 происходит не только переход к строке программы 100, но и очистка экрана и всех переменных. RUN можно давать без номера строки, что будет воспринято как RUN 0, а GO TO всегда требует указания номера строки.

CLS, IF, STOP

Рассмотрим программу:

```
10 INPUT "ENTER A SECRET NUMBER", A:CLS
20 INPUT "GUESS THE NUMBER", B
30 IF B=A THEN PRINT "THAT IS CORRECT":STOP
40 IF B<A THEN PRINT "THAT IS TOO SMALL, TRY
AGAIN"
50 IF B>A THEN PRINT "THAT IS TOO BIG, TRY
AGAIN"
60 GO TO 20
```

CLS в строке 10 стирает экран, а оператор IF имеет форму:

IF условие THEN XXX

Если условие истинно, то выполняются все команды разделенные ':', следующие за THEN. В противном случае команды игнорируются.

Следующие операторы используются для сравнения числовых и строчных переменных. Строчные переменные сравниваются по отношению к алфавитному порядку.

= — равно
< — меньше
> — больше
<= — меньше или равно
>= — больше или равно
<> — не равно

По команде STOP внизу экрана возникает сообщение:

9 STOP STATEMENT, 30:3

в некоторых версиях BASIC возможен формат:

IF условие THEN номер строки

Но чаще возможно только:

IF условие THEN GO TO номер строки.

Циклы. FOR, NEXT, TO, STEP

Для организации циклов существует специальная команда

FOR...NEXT

Наиболее общая форма команды:

FOR контрольная переменная = начальное значение TO

конечное значение STEP значение шага

FOR C = 1 TO 5 STEP 3/2

Во время выполнения цикла контрольная переменная будет увеличиваться на 3/2 с каждым шагом, шаг может быть так же задан переменной — STEP S, и будет с каждым шагом принимать значения этой переменной. Значения шага могут быть и отрицательные.

Циклы могут быть вложены друг в друга, но они не должны пересекаться.

Цикл может быть записан в строку и в непосредственном режиме:

FOR M=0 TO 10:PRINT M:NEXT M

Можно организовать бесконечный цикл:

FOR M=0 TO 1 STEP 0:INPUT A:PRINT A:NEXT M

Подпрограммы. GO SUB, RETURN

Если какая-то часть программы должна выполняться более одного раза, то нет смысла ее переписывать дважды, а следует написать ее один раз и обращаться к ней в нужный момент.

Для этого используется GO SUB XXX и RETURN, где XXX номер первой строки программы.

Когда команда RETURN считывается в конце подпрограммы, то происходит возврат в место вызова, и выполнение продолжается со следующей за GO SUB команды.

Одна подпрограмма может успешно вызывать другую или саму себя, в последнем случае она называется рекурсивной.

Данные в программе. READ, DATA, RESTORE

Вводить данные в программу можно не только при помощи команды INPUT, но также при помощи команд READ, DATA, RESTORE. Например:

```
10 READ A, B, C
20 PRINT A, B, C
30 DATA A, B, C
```

За READ может следовать список переменных, разделенных запятыми, выполнение такое же как при INPUT с той разницей, что значения берутся из строки, начинающейся с DATA.

DATA представляет собой список числовых или строковых значений, разделенных запятыми, которые могут располагаться в любом месте программы. READ считывает последовательно эти значения, если данных не хватает генерируется сообщение об ошибке. Последовательный порядок считывания данных может быть изменен командой

RESTORE XXX

где XXX номер строки с которой начинается считывание. Если команда RESTORE дается без указания номера строки, то считывание начинается с первой строки DATA.

Строки

Существует возможность выделения из строк их частей, в общей форме:

Строковое выражение (начало TO конец)

Например:

"ABCDEF" (2 TO 5)

Результат: "BCDE"

Можно опускать один или оба параметра

```
"ABCDEF"( TO 5)        -> "ABCDE"
"ABCDEF"( TO )         -> "ABCDEF"
"ABCDEF"()             -> "ABCDEF"
"ABCDEF"(3)            -> "ABCDEF"(3 TO 3)
                       -> "c"
```

Если первый параметр превышает длину строки, то результатом является пустая строка и появляется сообщение об ошибке:

3 SUBSCRIPT WRONG.

Для извлечения части строки из выражений нужны скобки:

("ABC"+"DEF")(1 TO 2) -> "AB"

Функции

Функции действуют следующим образом:
Передача аргумента -> функция -> выдача результата
Как правило функция вызывается по имени, за которым передаются параметры, например функция LEN принимает значение равное длине строки.

PRINT LEN "SPECTRUM 128"

LEN "FRED"+LEN "BLOGGS" то же, что и
LEN ("FRED"+"BLOGGS")

и результат равен 10.
Функция STR$ превращает число в строку, число передается в качестве аргумента:

LET A$=STR$ 1E2 то же, что и
LET A$="100"

VAL обратная функция, она превращает строку в число

VAL "3.5"

Принимает значение равное числу 3.5.
Параметром функции VAL может быть любое выражение:

VAL ("2*3")

Значение равно 6.
Функция SGN определяет знак выражения:
Если выражение > 0, функция равна +1;
Если выражение < 0, функция равна -1;
Если выражение = 0, функция равна 0.
Функция ABS принимает абсолютное значение выражения:

ABS -3.2 равно 3.2

Функция INT принимает значение целой части выражения:

INT 3.2 равно 3, но
INT -3.2 равно 4

Функция SQR вычисляет квадратный корень. Если аргумент отрицательное число то появляется сообщение об ошибке:

A INVALID ARGUMENT.

Вы можете определять и использовать собственные функции:

10 DEF FN S(X) = X*X

где S имя функции, а X передаваемый параметр.
Вызов функции происходит по команде FN S(...)

PRINT FN S(3+4)

SPECTRUM BASIC не имеет функций некоторых других версий LEFT$, RIGHT$, MID$, TL$, но их легко сделать при помощи DEF FN.

DEF FN 1$(A$, N)=A$(TO N): REM LEFT$

Случайные числа. RND, RANDOMIZE

RND напоминает функцию, но не требует параметров.
RND выдает псевдослучайную последовательность чисел в интервале от 0 до 1. 5*RND дает числа от 0 до 5.
RANDOMIZE используется для указания начального числа псевдослучайной последовательности. RANDOMIZE 1 означает, что псевдослучайная последовательность начинается с числа 0.0227335596. Параметры могут быть в пределах от 1 до 65535.

Массивы

Обратите внимание, что управление строковыми массивами в SPECTRUM 128 не совсем стандартное.
Перед использованием массивы требуют объявления:

DIM M(10)

где M — имя массива, а 10 его размер. Команда DIM, инициализирующая массив, присваивает всем его элементам значение 0.
Можно организовать более чем одноразмерные массивы:

DIM M(3, 6)

Строковые массивы требуют указания не только количества элементов, но и длины строк:

DIM A$(3, 6)

где первый параметр — количество строк, а второй длина каждой из них.
PRINT A$(2, 7) печатает на экран 7-ой символ 3-й строки.
DIM A$(10) образует строку длиной 10 символов.

Условия

В командах IF условие THEN можно использовать также логические операторы AND, OR, NOT.

IF A$="YES" AND X>0 THEN PRINT X

AND означает, что выражение истинно, если обе части его истинны.
Выражение, связанное оператором OR истинно, если по крайней мере один из элементов или оба вместе истинны.
NOT реверсирует отношение, т.е. истинно если ложно и ложно если истинно. NOT является функцией с аргументом и результатом, но с приоритетом низшим чем у других функций, поэтому скобки можно не употреблять если аргумент не содержит AND или OR.

NOT A=B означает A<>B

Истина и ложь имеют числовые значения: истина = 1, ложь = 0.

Расположение символов. CODE, CHR$, POKE, PEEK, USR, BIN

В SPECTRUM 128 существует таблица символов, которые кодируются числами от 0 до 255. Перевод символов в коды и наоборот возможен при помощи функций CODE и CHR$.
Параметром функции CODE является строка, а результатом является код первого символа.
Параметром функции CHR$ является код, а результатом строка из одного символа, соответствующего этому коду.
Следующая программа распечатывает на экране все символы

10 FOR A=32 TO 255: PRINT CHR A:: NEXT A

Символы со 128 по 134 являются символами мозаичной графики и используются в режиме <GRAPH>, печатаются с нажатием на клавиши 1, 2, 3, 4, 5, 6, 7, 8 и на те же клавиши с <CAPS SHIFT>.
Нажатие на <9> — выход из графического режима, так же как и вторичное нажатие на <GRAPH>.
После графических символов располагается копия алфавита от A до S, которые можно переопределить самостоятельно, тогда они могут вызываться с клавиатуры в графическом режиме с нажатием на соответствующие клавиши.
Символы состоят из таблицы точек 8*8, которые либо светлые, либо темные.
USR — функция, конвертирующая строковый аргумент в адрес первого байта графического изображения символа определяемого пользователем.
Команда POKE сохраняет числа непосредственно в памяти по адресу указанному USR. Обратная функция peek принимает значение байта по указанному в качестве аргумента абсолютному адресу памяти.
После определяемых символов идут коды ключевых слов (TOKENS)
С 0 по 31 располагаются управляющие коды. Так CHR$8 сдвигает курсор назад на одну позицию. Код 13 (CHR$13) перемещает курсор в начало новой строки экрана. Для управления экраном используются также коды 16-23.

Цвета. INK, PAPER, FLASH, BRIGHT, INVERSE, OVER, BORDER

Вы можете управлять восемью цветами и двумя уровнями яркости, цвета имеют следующие номера:

0 — черный
1 — синий
2 — красный
3 — малиновый
4 — зеленый
5 — голубой
6 — желтый
7 — белый

Возможна также инверсия фона и текста, и мигание. для управления цветами существуют следующие команды:

PAPER	— цвет фона	— число 0 — 7
INK	— цвет шрифта	— число 0 — 7
BRIGHT	— яркость	— число 0/1
FLASH	— мигание	— 0/1

С командами PAPER и INK возможен параметр 9, означающий контрастирование, т.е. со светлыми цветами — черный, а с темными — белый.

Существуют также команды INVERSE и OVER с параметрами:

0 — выключить
1 — включить

Так INVERSE 1 приводит к печати инверсированным цветом, OVER 1 вызывает наложение символов друг на друга:

```
10 OVER 1
20 PRINT ''W'';CHR$ 8;''_'';
```

в результате чего происходит подчеркивание символа ''W''. Команды INK и PAPER можно вставлять в команду PRINT

```
PRINT PAPER 6;''X'';:PRINT ''Y''
```

в данном случае только ''x'' будет печататься на желтом фоне.

BORDER номер цвета — меняет цвет нижнего экрана. Цвета можно менять при помощи управляющих кодов:

CHR$	код
16 —	INK
17 —	PAPER
18 —	FLASH
19 —	BRIGHT
20 —	INVERSER
21 —	OVER

Функция ATTR имеет форму:

```
ATTR(строка, колонка)
```

Число, которое получается в результате, является суммой четырех чисел:
128 если символ мигает, 0 — если нет
64 если символ яркий, 0 — если нормальный
8 умноженное на код цвета фона
1 умноженное на код цвета шрифта.

Графика. PLOT, DRAW, CIRCLE

Команда PLOT X координата, Y координата ставит точку на экране. Следующая программа ставит точки на экране по случайным координатам:

```
10 PLOT INT (RND*256), INT (RND*176); INPUT A$:
GO TO 10
```

Команда DRAW X, Y предназначена для рисования прямых линий.

Первой точкой является последняя точка предыдущей команды графики: PLOT, DRAW или CIRCLE. Команды RUN, CLEAR, CLS и NEW перемещают эту точку в нижний левый угол (0, 0).

```
PLOT 0, 100:DRAW 80. -35
```

Команды PAPER, INK, FLASH, BRIGHT, INVERS и OVER могут применяться с PLOT или DRAW между ключевым словом и координатами.

При помощи DRAW X, Y, а можно рисовать элементы окружности, так как третий параметр определяет угол поворота, а X и Y — координаты. A — число в радианах, если оно положительное, то поворот налево, если отрицательное — поворот направо. Полная окружность получается, если A=2*PI.

Команда CIRCLE X, Y, радиус — чертит полный круг, управление цветом такое же как и в предыдущих командах.

PAUSE, INKEY, PLAY

Команда PAUSE N останавливает компьютер. Значение N может быть от 0 до 65535, около 22 минут максимум, если N=0, то пауза длится бесконечно. Пауза может быть прервана нажатием на клавишу.

Для измерения времени может быть использовано следующее:

```
(65536*PEEK 23674+256*PEEK 23673+PEEK
23672)/50
```

Это выражение дает количество секунд с момента последнего включения компьютера или последнего RESET. точность внутренних часов около 0.01%, использование команды BEEP или периферийных устройств останавливает на время внутренние часы.

Можно поставить внутренние часы в реальное время, достаточно записать в три ячейки памяти число секунд *50. Так время 10.00 будет иметь форму 1800000 пятидесятых секунды.

```
1800000=65536*27+119+64
```

т.о. в ячейки следует записать числа 27, 119, 64

```
POKE 23674, 27:POKE 23673, 119:POKE 23672, 64
```

Функция INKEY$ не имеет аргументов, а считывается символ с клавиатуры, если клавиша не нажимается результатом функции является пустая строка.

```
10 IF INKEY$='' '' THEN GO TO 10
20 PRINT INKEY$;
```

Звук

Наиболее простой способ получения звука — команда BEEP.
BEEP продолжительность, высота.
Длительность задается в секундах, высота в числе полутонов.
0 — среднее до, отрицательные числа — звуки нижних октав. т.о. октава имеет 12 числовых значений.

Команда BEEP существует для совместимости с предыдущими версиями SPECTRUM.

9.4. Программирование звуков командой PLAY

Создание музыки и звуковых эффектов при помощи PLAY — просто вы должны ввести серии нот, составляющих мелодию, затем просите SPECTRUM проиграть ее. Кроме того, вы можете включить дополнительные инструкции для уточнения способа звучания.

Можете попробовать ввести 2 следующие программы; они продемонстрируют некоторые из широкого диапазона возможностей SPECTRUM 128. Не пугайтесь, что они выглядят непонятными — ниже все будет подробно рассмотрено.

Музыка

```
10 LET A# = ''T180O6 (CDEC) (5EF7G)
(3GAGF5EC)5CG7C9CGC''
20 LET B# = ''O4 (CDEC) (5EF7G) (3GAGF5EC)
5EB7E9EBE''
30 LET C#=03(7CG)(7CG)(7CG)5GD7G9GDG''
40 PLAY A#,B#,C#
```

Звуковые эффекты.

```
10 LET A#=''M8UX350W507(((C)))'': PLAY A#:
PAUSE 25
20 LET A#=''M56UX5000W103(((C)))'':PLAY A#:
PAUSE 25
30 LET A#=''M56W201N8C'':PLAY A#: PAUSE 25
```

Использование оператора PLAY

В примерах, приведенных выше, вы видите, что оператор PLAY сопровождается 1-3 различными буквами со знаком ''#'' в операторах вида

```
PLAY A#,B#,C#
```

Каждая из этих букв — имя символьной переменной (стринга), которая ранее в программе была определена. Этот стринг говорит SPECTRUM 128 о том, какой звук нужно создать.

PLAY контролирует 3 голоса, названных A, B и C, поэтому в этом операторе может использоваться до 3-х стрингов — по 1-му на каждый канал. В примере ''Музыка'' данном выше, A# определяет мелодию на канале A, B# определяет гармонию на канале B C# определяет басовую партию на канале C. В примере ''Звуковые эффекты'', данном выше, одновременно используется только 1 голос из 3-х, поэтому оператор выглядит просто.

```
PLAY A#
```

Естественно, что каждый из каналов может производить как музыкальные звуки, так и шумовые эффекты, и что можно смешать те и другие в одной мелодии (см. ''Выбор каналов'').

Конструирование стрингов

Сочинять музыку и звуковые эффекты проще всего, создавая стринги, содержащие необходимую информацию. Вы уже видели как это делается в приведенном примере ''Музыка''. Любой стринг создается оператором LET, после которого стоит имя стринга, затем его содержимое, ограниченное кавычками. Попробуйте следующий пример, который проиграет всего 1 ноту — A.

```
LET A#=''A'': PLAY A#
```

Как вы могли заметить по предыдущим примерам, любая музыкальная программа, использующая PLAY, использует также LET, чтобы определить, что нужно играть.

Любой музыкальный звук имеет длительность и высоту. Кроме того, он имеет громкость и качество тона. Стринги содержат информацию об этом. Все возможные команды приведены в таблице.

Список команд оператора PLAY

Все команды оператора PLAY, за исключением нот, вводятся заглавными буквами.

Команда	Функция
C-B или c-b	устанавливает высоту ноты в границах рабочей октавы
$	бемоль

#	диез
O	с последующим числом от 0 до 8 уст. октаву
1-12	устанавливает длительность
&	пауза
N	разделяет 2-е цифры
V	с последующим числом от 0 до 15 устанавливает громкость
W	с последующим числом от 0 до 7 устанавливает звуковой эффект
U	включает звуковой эффект в любом стринге
X	с последующим числом от 0 до 65535 устанавливает длительность звукового эффекта.
T	с последующим числом от 60 до 240 уст. темп
()	ограничивает повторяющуюся фразу
!!	ограничивает комментарий
H	останавливает воспроизведение музыки
M	с послед. числом от 0-63 уст. каналы
Y	с послед. числом от 1-16 вкл. каналы MIDI
Z	с послед. числом засылает контр. коды в MIDI

Установка высоты звука

Как вы уже знаете, высота звука задается буквенным обозначением соответствующей ноты, например:C, E, B. Диез обозначается префиксом # (например #c), а бемоль префиксом $. Ваш ''инструмент'' одновременно может работать в 2-х октавах в скрипичном ключе, буквы C и B используются для низшей октавы, C и B для высшей.

Все ноты этих 2-х октав будут звучать последовательно, одна за другой.

Например:

```
10 LET A$=''CFEDAFGCFEDAFGCC''
20 PLAY A$
```

Если вы хотите использовать более 2-х октав, то можете менять октаву командой ''O'' с последующим числом от 0 до 8. Если вы не задали октаву (как, например, в приведенном выше примере), то по умолчанию установлена 5-я октава. Все ноты, следующие за командой изменения октавы, будут звучать в установленной октаве до тех пор, пока не будет задана другая октава.

Следующая программа даст вам возможность прослушать ту же мелодию, что и в предыдущем примере, но в более высокой октаве (достаточно добавить 07 в вашу предыдущую программу).

```
10 LET A$=''07CFEDAFGCFEDAFGCC''
20 PLAY A$
```

Попробуйте изменять номер октавы, чтобы прослушать весь диапазон частот, воспроизводимых SPECTRUM 128. Стоит заметить, что самые низкие ноты 0-й и 1-й октав не будут воспроизводится без MIDI. SPECTRUM 128 будет воспроизводить их на самой низкой частоте, которую он способен реализовать.

Диапазон высот, охватывающих 2 октавы, перекрывается с 2-я другими. Так, например, верхний участок 04 совпадает с нижним участком 05. Диапазон можно расширить, используя серии диезов (#####) или бемолей ($$$$$), чтобы звучание последующей ноты повысить либо понизить.

Длительность звучания ноты

Если специально не определять длительность звучания ноты, то она будет звучать столько же времени, как и в предыдущих примерах. Вы можете изменить длительность звучания ноты, поставив перед ней цифру от 0 до 12. Следующая программа продемонстрирует вам звучание ноты с разными длительностями (0-9)

```
10 LET A$=''1C2C3C4C5C6C7C8C9C''
20 PLAY A$
```

Первая нота — самая короткая, последняя самая длинная.

Кроме того, вы можете использовать цифры от 10 до 12, которыми можно определить трель (три ноты звучат столько же времени, сколько должны звучать 2), например:

```
10 LET A$=''11ACE''
```

Пауза задается командой & и имеет такую же длительность, как и ноты. Так, например, программа

```
10 LET A$=''07A&B&C&D&E''
```

прозвучит как 5 нот с паузами между ними.

Парные ноты могут задаваться 2-мя длительностями, разделенными символом подчеркивания, например:

```
10 LET A$=''3_5A''
```

Длительность звучания второй ноты распространится и на все последующие ноты.

Команда N

В некоторых примерах вы могли заметить, что буква N используется для разделений серий нот, например

```
10 LET A$=''07N1CDE''
```

Здесь N разделяет номер октавы и длительность звучания. Без команды N SPECTRUM 128 воспринял бы цифры 7 и 1 как номер октавы, что привело бы к ошибке.

Громкость звучания ноты

Общая громкость звучания регулируется регулятором громкости вашего телевизора или усилителя, но громкость отдельных нот или фраз можно определять и программно, используя команду V. Эта команда со следующим числом от 0 до 15, задает громкость последующих нот. Минимальная громкость, прекращающая звучание нот — V0, она может использоваться для временного выключения голоса. V15 — максимальная громкость звучания, которая устанавливается по умолчанию.

Слабые уровни громкости очень слабы, поэтому рекомендуется использовать громкости от 10 до 15, если вы не используете усилитель или MIDI. Попробуйте запустить следующую программу:

```
10 LET A$=''V10CDEFGAB''
20 PLAY A$
```

Теперь попробуйте изменять число после V и увидите разницу.

Звуковые эффекты

Помимо возможности установить уровень каждой ноты, вы можете также изменять ее громкость во время звучания. Так, например, вначале нота может звучать в полную громкость, а к концу постепенно затухать.

Эффекты задаются командой W, которая может быть включена в каждый из стрингов оператора PLAY. Вы также можете использовать команду W в том месте, где вы хотите включить звуковые эффекты. После команды W должен стоять цифровой параметр от 0 до 7, указывающий на построение звукового эффекта. В таблице указаны все варианты звуковых эффектов.

0 единичный спад, затем тихо
1 единичный подъем, затем тихо
2 единичный спад, затем громко
3 единичный подъем, затем громко
4 повторяющийся спад
5 повторяющийся подъем
6 повторяющийся подъем — спад
7 повторяющийся спад — подъем

Следующая программа продемонстрирует разнообразие звуковых эффектов.

```
10 LET
A$=''UX1000W0C&W1C&W2C&W3C&W4C&W5C&W6
C&W7C''
20 PLAY A$
```

Команда U включает звуковые эффекты, далее они определяются серией команд W.

В этом примере используется новая команда — буква X. Она, с последующим числом 0 — 65535, устанавливает длительность звукового эффекта. По умолчанию SPECTRUM 128 сам выберет длительность. Как правило, при повторяющихся эффектах наиболее эффективны значение X1000 и т.п. Попробуйте изменять значение после X в приведенной программе, и вы заметите различие.

Темп

Темп, в котором будет воспроизводится музыка, определяется командой T с последующим числом от 60 до 240. Команда устанавливает темп звучания музыки, но может быть вставлена только на канале A (первый стринг после команды PLAY) иначе она будет игнорирована, например:

```
10 LET A$=''T180CDEFG''
20 PLAY A$
```

По умолчанию темп устанавливается 120.

Повторение фраз

Любая музыкальная фраза может быть повторена, если ее взять в скобки. Так, например:

```
10 LET A$=''ABC(DEFG)''
```

повторит последние 4 ноты. Если количество закрывающихся скобок больше, то фраза будет повторяться до последней скобки.

Если закрывающая скобка единственная, то фраза будет повторяться сначала:

```
10 LET A$="ABCDEFG)"
```

будет повторять все 7 нот. Двойная закрывающая скобка

```
10 LET A$="02CEGA))"
```

будет повторять фразу бесконечно. Это весьма полезно в басовых партиях. Для выключения бесконечных фраз используйте команду H.

Команда H

Команда H, встреченная в каком-либо стринге, немедленно выключит оператор PLAY. Главным образом она используется для выключения бесконечных басовых партий. Вы можете включить их, вставив в конце стринга, задающего мелодию.

Комментарии

В музыкальный стринг можно вставить комментарий, ограничив его восклицательными знаками. Все, что будет написано после первого !, будет игнорироваться до тех пор, пока не встретится второй ! или кавычки, закрывающие стринг, например:

```
10 LET A$="ABCDEFG!ПРИПЕВ!ACEADG"
```

Выбор канала

Команда M служит для определения того, что будет воспроизводить каждый канал: звук или шумовые эффекты.

Вы можете одновременно использовать до 3-х каналов, причем все они могут воспроизводить ноты, шумовые эффекты или то и другое вместе.

Ваш выбор определяется числом, следующей за командой M и обозначающее следующее:

	муз. канал	шумовой канал
	A B C	A B C
число	1 2 4	8 16 32

Выберите то, что вы желаете иметь в каждом канале, и сложите числа. Полученное число нужно поставить после команды M. Так, например, если вы хотите во всех 3-х каналах использовать звук, то сложите числа 1+2+4=7, таким образом команда M7 установит звук на всех 3-х каналах. Точно также, как команда M56 установит на всех 3-х каналах шум.

Несмотря на то, что шумовые эффекты могут использоваться на всех каналах, наиболее широкий диапазон частот на канале A. Для наилучших результатов шумовые эффекты рекомендуется вставлять в стринг, следующий за оператором PLAY.

9.5. Запись данных

LOAD, SAVE, VERIFY, MEGRE

Для сохранения программы на кассете введите:

```
SAVE "FILEMAME"
```

Имя файла может быть до десяти символов длинной.
SPECTRUM 128 отвечает сообщением:

```
START TAPE, THEN PRESS ANY KEY
```

После нажатия соответствующих клавиш магнитофона и любой клавиши компьютера программа сохраняется на ленте. Программа записывается блоками. Перед тем как стирать программу из памяти наберите VERIFY, перемотайте ленту в начало и включите воспроизведение, компьютер проверит качество записи. На экране появится сообщение:

```
PROGRAM: FILENAME
```

А в конце записи "0 OK", что означает удовлетворительное качество записи. Если запись произошла с ошибкой появится сообщение:

```
R TAPE LOADING ERROR
```

Для загрузки используется команда LOAD "FILENAME".

Команда LOAD уничтожает старую программу в компьютере и загружает новую. После загрузки можно запускать программу командой RUN.

Команда MERGE "FILENAME" также загружает программу в память, но при этом старая программа не уничтожается. Две программы сливаются в одну.

Команда SAVE может даваться с параметрами:

```
SAVE "FILENAME" LINE номер строки
```

Сохраненная таким образом программа может быть загружена только командой LOAD (но не MERGE) и запускается указанной строки автоматически.

Можно сохранить также не целиком программу, а только определенные данные.

```
SAVE "FILENAME" DATA имя массива ()
SAVE "BLOGGS" DATA B ()
```

Таким образом сохраняются все элементы массива.

Команда LOAD "BLOGGS" DATA B() находит массив на кассете, загружает его в память и если в программе есть массив B, то он заменяется полученным с ленты.

Команда MERGE с сохранением массива не используется.

Область памяти сохраняется командой

```
SAVE "PICTURE" CODE 16384, 6912
```

Первый параметр — начальный адрес сохраняемой области, второй — длина области.

Загрузка машинного кода производится командой:

```
LOAD "FILENAME" CODE начальный адрес, длина
```

Экранную область можно сохранить командой:

```
SAVE "PICTURE" SCREAN$
```

соответственно загрузка:

```
LOAD "PICTURE" SCREAN$
```

SAVE, LOAD и MERGE работают также со встроенным диском до 64K.

```
SAVE  ! "FILENAME"
LOAD  ! "FILENAME"
MERGE ! "FILENAME"
```

Команда CAT ! показывает содержимое диска.
ERASE ! "FILENAME" — уничтожает файл на диске.

Работа с принтером. LPRINT, LLIST, COPY

Скорость обмена с принтером устанавливается командой:

```
FORMAT "P", скорость в бодах
```

Если скорость не установлена принимается 9600 бод.

Команды LPRINT, LLIST аналогичны PRINT и LIST, но работают с принтером.

Команда COPY передает изображение экрана на принтер.

Работа с портами. IN, OUT

Функции IN и OUT используются для чтения и записи в порты ввода/вывода.

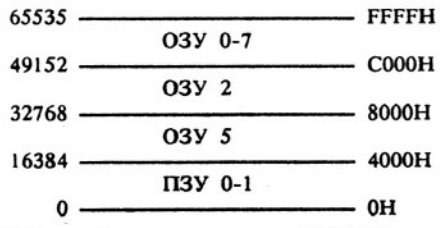
IN	адрес	— чтение
OUT	адрес, значение	— запись.

Адреса: данных и состояния клавиатуры

IN 62278	— <CAPS SHIFT>-V
IN 65022	— A-G
IN 64510	— Q-T
IN 63486	— 1-5 (JOYSTICK 2)
IN 61438	— 0-6 (JOYSTICK 1)
IN 57342	— P-Y
IN 49150	— <ENTER>-H
IN 32766	— <SPACE>-B

Память. CLEAR

Процессор способен адресовать 65536 байт, в компьютере SPECTRUM 128 располагается 131072 байта ОЗУ и 32768 байт ПЗУ. Одновременно адресуется 16к ПЗУ и 48к ОЗУ.



Самый большой адрес используемый BASIC системой указывается переменной RAMTOP. Наибольший адрес может быть изменен командой

```
CLEAR новый адрес RAMTOP
```

Эта команда имеет следующие эффекты:

1. очищаются все переменные
2. очищается экран как при CLS
3. графический курсор устанавливается в 0
4. указатель DATA устанавливается в начало
5. очищается GO SUB стек и устанавливается новый RAMTOP (высший доступный адрес ОЗУ)

9.6. Системные переменные

POKE, PEEK

Область памяти с 23296 до 23733 используется системой. там находятся некоторые системные программы и системные переменные.

В списке системных переменных используются сокращения:

X — нельзя использовать POKE, система может выйти из строя.

N — POKE не приводит к длительному эффекту

R — точка входа подпрограммы. Не переменная.

прим.	адрес	сокр.	описание
R20	23296	SWAP	подпрограмма обмена страниц памяти
R9	23316	YOUNGER	-''-
R18	23325	ONERR	-''-
R5	23343	PIN	подпрограмма ввода RS232
R22	23348	POUT	RS232. Подпрограмма вывода
R14	23370	POUT2	RS232. Вывод символов
N2	23384	TARGET	адрес подпрограммы в ПЗУ 1
X2	23386	RETADDR	адрес возврата в ПЗУ 0
X1	23388	BANK	копия последнего байта посланного в банк
X1	23389	RAMRST	RST 8 инструкция
N1	23390	RAMERR	номер ошибки ПЗУ 1
2	23391	BAUD	RS232 битный период в T/26
N2	23393	SERF	флаг состояния второго принимаемого символа и данные
N1	23395	COL	текущая колонка
1	23396	WIGHT	ширина печатной страницы
1	23397	TVPARS	№ параметра ожидаемого RS232
1	23398	FLAGS3	различные флажки
N10	23399	N STR1	имя файла
1	23409	HD 00	тип файла
2	23410	HD 0B	длина блока
2	23412	HD 0D	начальный адрес блока
2	23414	HD 0F	длина программы
2	23416	HD 11	номер строки
1	23418	SC 00	тип файла
2	23412	SC 08	длина блока
2	23423	SC 0F	длина программы
X2	23425	OLDSP	старый SP при использовании TSTACK
X2	23427	SFNEXT	пойнтер свободного места в DIRECTORY
X3	23429	SFSPACE	количество свободных байт
N1	23423	ROW01	
N1	23433	ROW23	
N1	23434	ROW45	
X2	23435	SYNRET	возврат адреса для ONERR
5	23437	LASTV	последняя длина выводимая при вычислениях
2	23442	RNLINE	номер строки для RENUMBER
2	23444	RNFIRST	новый номер строки начала перенумерации
2	23446	RNSTEP	приращение номера строки для перенумерации
N8	23448	STRIP1,	
N8	23456	STRIP2,	
X	23551	TSTACK	временный стек
N8	23552	KSTATE	устанавливается при чтении клавиатуры
N1	23560	LAST K	код новой нажатой клавиши
1	23561	REPDEL	время задержки на повтор удерживаемой нажатой клавиши
1	23562	REPPER	задержка при повторе на вывод одного символа
N2	23563	DEFADD	адрес аргумента функции пользователя если отсутствует, то равно 0
N1	23565	KDATA	2-й байт — цвет введенного с клавиатуры символа
N2	23566	TVDATA	байт управления цветом, AT и TAB управления выводом на монитор
X38	23568	STRMS	адрес канала зарезервированного под вывод
2	23606	CHARS	указатель буфера вывода
1	23608	RASF	длина предупреждающего сигнала
1	23609	PIP	длительность звука при нажатии клавиши
1	23610	ERR NR	код условия передачи
X1	23611	FLAGS	флаг изменения управления BASIC системой
X1	23612	TVFLAG	флаг управления монитором
X2	23613	ERRSP	адрес машинного стека, устанавливаемый при возврате по ошибке
N2	23615	LISTSP	адрес возврата при автоматическом выводе листинга
N1	23617	MODE	спецификация K, L, C, E, G курсора
2	23618	NEWPPC	номер строки перехода
1	23620	NSPPC	номер оператора в строке, для перехода. Оператор идентифицируется в тексте двумя указателями NEWPPC и NSPPC
2	23621	PPC	номер строки оператора начала выполнения
1	23623	SUBPPC	номер и строка оператора начала выполнения
1	23624	BORDCR	параметр оператора BORDER
2	23625	EPCC	адрес верхней строки
X2	23627	VARS	адреса переменных
N2	23629	DEST	адрес переменной при присваивании
X2	23631	CHANS	адрес канальных данных
X2	23633	CURCHL	адрес информационной области устанавливаемой при вводе и выводе
X2	23637	PROG	адрес BASIC программы
X2	23639	NXTLIN	адрес строки выхода из программы
X2	23641	DATADD	адрес завершающей программы выполняемой после ввода последнего элемента DATA
2	23643	KCUR	адрес курсора
X2	23645	CHADD	адрес интерпретируемого выходного символа после аргумента PEEK или NEWLINE в конце POKE оператора
2	23647	XPTR	адрес символа после [?] маркера
X2	23649	WORKSP	адрес временного рабочего пространства
X2	23651	STKBOT	адрес указателя глубины вычислительного стека
X2	23653	STKEND	адрес конца свободной области стека

прим.	адрес	сокр.	описание
N1	23655	BREG	вычислительный в регистр
N1	23656	MEM	адрес зарезервированной вычислительной памяти
1	23658	FLAGS2	флаги
X1	23659	DFSZ	номер строки в чистой странице
2	23660	S TOP	номер текущей строки программы в листинге
2	23662	OLDPPS	номер строки для CONTINUE перехода
1	23664	OSPCC	номер оператора для CONTINUE перехода
N1	23665	FLAGX	флаг переменных
N2	23666	STRLEN	длинна копии строки при присваивании
N2	23668	T ADDR	адрес выхода из синтаксической ошибки
2	23670	SEED	опция управления RND функцией, устанавливается оператором RANDOMIZE
3	23672	FRAMES	3 байта вычисляются каждые 20 MS
2	23675	UDG	адрес таблицы графики пользователя
1	23677	COORDS	X-координата графики
1	23678		Y-координата графики
1	23679	PPOSN	33-х колоночный номер позиции принтера
1	23680	PR CC	адрес последнего байта выведенного оператором LPRINT
1	23681		не используется
2	23682	ECHOE	адрес последнего выведенного байта на экране (позиция<=33 строка<=24)
2	23684	DF CC	адрес в экранной области позиции оператора PRINT
2	23686	DF CCL	размер нижнего раздела для DF CC
X1	23688	S POSN	33-колоночный номер позиции для PRINT
X1	23689		24-строковый номер позиции для PRINT
X2	23690	SPOSNL	S POSN для нижнего раздела (командной строки)
1	23692	SCR CT	количество SCROLL
1	23693	ATTR P	цветовой префикс
1	23694	MASK P	устанавливает изменения цвета
N1	23695	ATTR T	старое ATTR P до выполнения очистки
N1	23696	MASK T	временное MASK P
1	23697	P FLAG	флаги
N30	23698	MEMBOT	вычислительная область памяти, используется для записи промежуточных результатов которые не могут быть выведены в вычислительный стек
2	23728		не используется
2	23730	RAMTOP	адрес последнего байта системной области BASIC
2	23732	P-RAMT	адрес последнего байта ПЗУ

9.7. BASIC

Числа сохраняются с точностью до 9 или 10 знаков. Наибольшее число около 10 в степени 38, а наименьшее 4*10 в степени -39.

Числа сохраняются как бинарные с плавающей запятой с одним экспонентным байтом e(1<=e<=255), и четыре байта мантиссы (1/2<=M<=1). Это соответствует числу M*2 в степени E-128.

Целые числа представлены следующим образом: первый байт 0, второй знаковый — (0 или FFH), А третий и четвертый само число, наименее значимый байт первый.

Числовые переменные имеют имена любой длинны. Пробелы игнорируются, а все символы превращаются в строчные буквы, они должны обязательно начинаться с символа.

Управляющие переменные циклов FOR...NEXT должны иметь только одну букву.

Числовые массивы обладают именами длинной в один символ, которые могут совпадать с именами простых переменных, размерность может быть любой, как и количество элементов. Элементы начинаются с 1.

Строки могут быть любой длины. Имя строковой переменной должно состоять из одного символа, за которым следует $.

Строковые массивы могут быть многоразмерными. Имя массива — один символ завершенный $. Имена не могут дублироваться с именами других строковых переменных. Все строки массива имеют фиксированную длину определяемую командой DIM. Элементы начинаются с 1.

Функции

Аргументы функций не нуждаются в скобках.

функция	тип аргумента	результат
ABS	число	абсолютная величина
ARC	число	арккосинус в радианах
AND	бинарный оператор правый операнд всегда число левый операнд число левый операнд строка	 A AND B приоритет 3 A IF B<>0 0 IF B=0 A$ AND B A$ IF A<>0 "" IF B=0
ASN	число	арксинус в радианах
ATN	число	арктангенс в радианах
ATTR	X и Y — числа в скобках	число бинарная форма которого кодирует атрибуты строки X колонки Y. Бит 7-1 мигание 0 немигание 6-1 яркий 0 нормальный биты 5-3 цвет фона 2-0 цвет шрифта
BIN		двоичная запись числа
CHR$	число	символ соответствующий коду, округленному до целого
CODE	строка	код первой буквы. 0, если пустая строка
COS	число в рад.	косинус X
EXP	число	e в степени X
FN		FN с символом имени вызывает функцию определенную пользователем. Аргумент в скобках, если аргумента нет — скобки остаются.
IN	число	число находящееся в порту
INKEY$		читает клавиатуру, результат — односимвольная строка, если клавиша была нажата, если нет — пустая строка
INT	число	целая часть (всегда округляется)
LEN	строка	длина
LN	число	натуральный логарифм
NOT	число	0 IF X<>0, 1 IF X=0 приоритет 4
OR	бинарный оператор, оба операнда — числа	A OR B 1 IF B<>0 A IF B=0 приоритет 2

функция	тип аргумента	результат
PEEK	число	значение байта в памяти. Адрес, передаваемый как параметр округляется до ближайшего целого
PI		число пи=3.1415926...
POINT	X и Y числа в скобках	1 — если точка экрана имеет цвет шрифта, и 0 — если цвет фона
RND		следующее псевдослучайное число генерируемой последовательности, получаемой возведением в 75 степень MOD 65537 -1, деленное на 65536. Меньше 1 и больше 0
SCREEN$	X и Y оба числа заключенные в скобки	символ который находится на экране по указанным координатам. При невозможности распознать — пустая строка
SGN	число	знак числа: -1 отрицательное, +1 положительное
SIN	число в рад.	синус X
SQR	число	квадратный корень
STR$	число	строка с изображением числа
TAN	число в рад.	тангенс X
USR	число	адрес битной таблицы с изображением символа определенного пользователем
VAL	строка	строка рассматривается как числовое выражение
VAL$	строка	рассматривается как строковое выражение
-	число	отрицание

Бинарные функции

- \+ сложение чисел или слияние строк
- \- вычитание
- * умножение
- / деление
- ^ возведение в степень
- = равенство
- > больше
- < меньше
- <= меньше или равно
- => больше или равно
- <> не равно

Операции	Приоритет
операции со строками	12
все функции кроме NOT	11
	10
-	9
*,/	8
+,- (минус для вычитания)	6
=,>,<,<=,=>,<>	5
NOT	4
AND	3
OR	2

Операторы

В списке применяются следующие сокращения:

L — для одной буквы

V — переменная

X,Y,Z — численные выражения

M,N — численные выражения, округляемые до ближайшего целого числа

E — выражение

F — строковое выражение

S — последовательность выражений

C — последовательность выражений для установки цветовых определений, разделяемых запятыми или точками с запятой (PAPER, INK, FLASH, BRIGHT, INVERSE, OVER)

Все операторы, кроме INPUT, DEF FN и DATA могут применяться как в программах, так и в командном режиме. Команда или строка программы может содержать несколько операторов разделенных двоеточием.

BEEP X,Y звук длительностью X секунд высотой Y полутонов выше среднего ДО если положительное число, ниже среднего ДО, если отрицательное

BORDER M цвет фона нижнего экрана.
Ошибка K если 0<=M<=7

BRIGHT N яркость символа: N=0 — нормальная; 1 — повышенная; 8 — фиксация

CAT только для гибкого диска

CAT ! каталог встроенного электронного диска

CIRCLE X,Y,Z рисует окружность с центром X,Y и радиусом Z

CLEAR уничтожает переменные, освобождая память выполняются команды RESTORE, CLS, графический курсор устанавливается в левый нижний угол экрана, очищается GO SUB стек

CLEAR N то же что и CLEAR, но также устанавливается системная переменная RAMTOP в N и GO SUB стек (верхний доступный адрес ОЗУ).

CLOSE # только для гибкого диска

CLS очищает экран

CONTINUE продолжение программы после остановки во всех случаях кроме сообщения об ошибке 0. Если сообщение 9 или L продолжение происходит со следующего оператора, в противном случае с того где обнаружена ошибка

COPY посылает 22 строки с экрана на принтер

DATA E1,E2,E3... список данных используемых в программе

DEF FN L(L1...LK) определение функции пользователя. Без аргументов имеет форму DEF FN ()=e

DIM L (N) инициирует массив с именем L, уничтожая предыдущий с тем же именем. N — количество элементов, начиная с 1
DIM L$ (N,M) уничтожает массив с именем L$, и инициализирует новый массив с размером в N элементов, начиная с 1, фиксированной длинны M. Все элементы строкового массива приравниваются к " ". Использованию массива должен предшествовать DIM

DRAW X,Y рисует линию с конечными координатами X,Y

DRAW X,Y,Z рисует дугу с конечными координатами X,Y и углом кривизны Z в радианах

ERASE только для гибкого диска

ERASE! F уничтожает файл с встроенного электронного диска

FLASH устанавливает мигание символов: 0 — немигание, 1 — мигание, 8 — фиксация

FOR L=X TO Y присваивает переменной L значение X и меняет значение L до значения Y по оператору NEXT

FOR L=X TO Y STEP Z присваивает переменной L значение X и меняет значение L до значения Y с шагом Z по оператору NEXT

FORMAT F;N устанавливает скорость обмена с устройством F по интерфейсу RS232 со скоростью N (от 75 до 19200) бод

FORMAT A: только для гибкого диска

GO SUB N аналогично GO TO, с сохранением N в стеке для последующего возврата по оператору RETURN

GO TO N переход на строку с номером N

IF X THEN S выполнение оператора S если выражение X истинно

INK N устанавливает цвет шрифта (графики); N — номер цвета 0-7, 8 — фиксация, 9 — контраст

INPUT... оператор ввода данных с внешнего устройства ряда переменных, разделенных запятой, точкой с запятой или апострофом. Между оператором и списком переменных возможно включение сообщения для вывода на экран, при вводе строковых переменных можно использовать LINE

INVERSE N инверсирует цвет шрифта. N=0 — нормальный, 1 — инверсия фона и изображения

LET V=E присваивается выражение E переменной V

LLIST LLIST 0

LLIST N выводит текст программы на принтер начиная со строки N

LIST LIST 0

LIST N выводит текст программы начиная со строки N

LOAD "F" загрузка программы или переменных

LOAD "F" DATA () загружает числовой массив

LOAD "F" DATA $() загружает символьный массив

LOAD "F" CODE M,N загружает N байт с адреса M

LOAD "F" SCREEN $ загружает экранную область памяти

LOAD! "F" загружает файл с встроенного электронного диска

LPRINT так же как PRINT, но вывод на принтер

MERGE "F" так же как LOAD, но без стирания предыдущей программы, кроме строк совпадающих с загружаемыми

MERGE! "F" то же, но с встроенного электронного диска

MOVE "F1","F2" только для гибкого диска

NEW очищает оперативную память до байта указанного в RAMTOP. Управление передается входному меню

NEXT оператор завершающий цикл FOR...TO

OPEN # только для гибкого диска

OUT M,N выводит байт N в порт M

OVER N управляет печатью символов одного поверх другого. N=0 второй символ стирает первый, 1 — происходит смешение элементов символа

PAPER N так же как INK, но управляет цветом фона

PAUSE N останавливает компьютер на N/50 сек, если N=0, то компьютер останавливается до нажатия любой клавиши

PLAY F(,F,F,..) интерпретирует до 8 звуковых строк и воспроизводит их последовательно, первые три воспроизводятся через усилитель монитора, остальные при помощи MIDI

PLOT C;M,N выводит на экран точку цвета шрифта, перемещает графический курсор в позицию M,N

POKE M,N записать байт N по адресу M

PRINT... выводит последовательность переменных и констант на экран

RANDOMIZE RANDOMIZE 0

RANDOMIZE N устанавливает системную переменную SEED, используемую для генерации следующего значения RND. Если N<>0 то SEED=N, если N=0 то SEED=FRAMES (экранные таблицы)

READ V1,V2,... присваивает переменным значение выражений указанных в списке DATA

REM для комментариев

RESTORE RESTORE 0

RESTORE N устанавливает указатель DATA на строку N

RETURN возвращает из подпрограммы на оператор следующий за GO SUB

RUN RUN 0 RUN N CLEAR, затем GO TO N

SAVE "F" сохраняет программу и переменные под именем F

SAVE "F" LINE M сохраняет программу и переменные под именем F затем переход на строку M

SAVE "F" DATA() так же как и LOAD

SAVE "F" DATA$() — " " —

SAVE "F" CODE M,N — " " —

SAVE "F" SCREEN$ — " " —

SAVE! "F" сохраняет программу и переменные под именем F на встроенном электронном диске

SPECTRUM переключает из 128 BASIC в 48 BASIC, сохраняя программу

STOP останавливает программу

VERIFY сравнивает данные на ленте с данными в оперативной памяти

ABEND — аварийное завершение
ABORT — прерывать выполнение программы
ABSOLUTE CODE — машинный код, программа в машинных кодах
ACCUMULATOR — сумматор, накапливающий регистр
ACTIVATE — активизировать, вызывать
ADDRESS — адрес
ARRAY — массив
ASCII — американский стандартный код для обмена информацией
ASSEMBLE — транслировать
AUTOLAOD — автозагрузка
BASE — база, базовый адрес
BASIC — простой для изучения и применения язык программирования.
BAUD — бод, единица измерения скорости передачи информации (1 бод = 1 бит в секунду)
BINARY — двоичный
BIT — бит, разряд
BOARD — плата
BOOK — книга
BOOLEAN ALGEBRA — булева алгебра
BOOT — начальная загрузка
BRUSH — кисть
BUS — шина, магистраль
BYTE — байт, группа из восьми битов
CALCULATOR — калькулятор
CALL — вызов, обращение к подпрограмме
CARRY — перенос, разряд переноса
CARTRIDGE — кассета
CASSETTE TAPE — кассетная лента
CATALOG — каталог
CELL — ячейка памяти
CHAIN — цепочка, последовательность
CHANNEL — канал ввода-вывода
CHARACTER — символ, знак, литера
CHECK — контроль, ошибка
CHIP — микросхема
CLEAR — очистить
CLOCK — часы
CLOSE — закрыть
CODE — код, система кодирования
COLD BOOT — "холодная" загрузка
COLOR — цвет
COLUMN — столбец
COMMAND — команда
COMMENT — комментарий
COMPILATION — трансляция
COMPUTER GAME — компьютерная игра
CONDITION — условие
CONTROL — управление
COPY — копия, копировать
COUNTER — счетчик
CP/M — операционная система 8-разрядных ПЭВМ
CPU — центральный процессор
CURRENT — текущий
CURSOR — курсор
CYCLE — цикл
DATA — данные, информация
DATA BASE — база данных
DATA INPUT — ввод данных, входные данные
DATA OUTPUT — вывод данных, выходные данные
DATA RECORD — запись данных
DATA TYPE — тип данных
DD — двойная плотность диска
DEBUG — отладка программы
DEC (DECIMAL) — десятичный
DECREMENT — уменьшение
DEFAULT — значение, принимаемое по умолчанию
DELETE — исключать, стирать, удалять.
DEVICE — внешнее устройство
DIGITAL — цифровой
DIRECTORY — каталог
DISABLE — блокировать, отключать
DISASSEMBLER — дизассемблер
DISK — диск
DISK DRIVE (DRIVE) — дисковод
DISPLAY — дисплей
DIVISION — деление
DOS — дисковая операционная система
DOWN — вниз
DRIVER — драйвер (управляющая программа)
DS — двухсторонняя дискета
DUMP — дамп, распечатка содержимого памяти
DYNAMIC MEMORY — динамическое ОЗУ
EDIT — редактировать
EDITOR — редактор
EEPROM — электрически стираемое ППЗУ
EMULATOR — эмулятор
ENABLE — разрешать, включать
END — конец
ENTER — ввод, вводить данные
ENTRY ROINT — точка входа
ERASE — удалять, стирать
ERROR — ошибка
EXECUTION — выполнение
EXIT — выход
EXTENT — экстент, непрерывная область на диске
FATAL ERROR — фатальная ошибка
FAULT — ошибка
FIELD — поле
FILE — файл, множество однотипных записей
FILE NAME — имя файла
FILL — заполнение, закрашивание
FIRE — "огонь"
FLAG — флаг, признак
FLASH — мигать
FLOPPY DISK — гибкий диск, дискета
FONT — шрифт
FORMAT — формат, способ расположения данных
FREE — свободный
FULL — полный, переполнения
GAME — игра
GET — прочитать
GOTO — переход, передача управления
GRAPHICS EDITOR — графический редактор
GROUP — группа
HACKER — хекер, программист, занимающийся поиском доступа к защищенным данным
HARD DISK — жесткий диск
HEAD — головка, первый элемент списка
HEADER — заголовок
HELP — подсказка
HEX — шестнадцатиричный
HIGH — старший, высокий
HOME — начало, левый верхний угол экрана
ID (IDENTIFICATION) —идентификатор, имя
IF — если
IGNORE — пропускать, игнорировать
ILLEGAL — недопустимый
IMAGE — изображение
INCREMENT — увеличивать
INDEX — индекс
INDICATOR — признак, флаг
INITIALIZATION — начальное присваивание значений
INPUT — ввод, входные данные
INSTALL — настраивать, устанавливать
INSTRUCTION — команда, оператор
INTEGER — целое
INTERFACE — интерфейс, стык
INTERNAL — внутренний
INTERPRETER — интерпретатор
INT (INTERRUPT) — прерывание
INVALID — недопустимый, ошибочный
INVERSION — отрицание
JOIN — соединение, слияние
JOYSTICK — джойстик, координатная ручка
JUMP — прыжок, передача управления
JUSTIFY — выравнивать
KEY — ключ
KEYBOARD — клавиатура
KILL — уничтожать, удалять
KILOBYTE — килобайт (1024 байта)
KNOW — знать
LABEL — метка
LAYOUT — размещение, формат

LEADER — начало, заголовок
LEFT — левый, влево
LEGAL — допустимый
LENGTH — длина
LETTER — буква, символ
LIBRARY — библиотека
LIMIT — предел, граница
LINE — строка
LINK — компоновать, связывать
LIST — список
LITERAL — буквальная константа
LOAD — загружать
LOADER — загрузчик
LONG — длинный
LOOK — смотреть
LOOP — цикл
LOW — младший, низкий
MACHINE — машина, ЭВМ
MAIN — главный, важный
MAP — таблица, карта
MARK — метка, маркер
MASK — маска, комбинация разрядов
MASTER — основной
MATCH — сравнивать, сопоставлять
MEGABYTE — мегабайт (1048576 байт)
MEMBER — элемент массива
MEMORY — память
MENU — меню
MERGE — объединять
MESSAGE — сообщение
MODE — режим, вид
MONITOR — управляющая программа, дисплей
MOUSE — "мышь", устройство ввода координат
MOVE — пересылать
MULTI — много
NAME — имя
NIBBLE — полубайт (четыре бита)
NO-OP INSTRUCTION — пустая команда
NORMALIZE — нормализовать
NOT — не, отрицание
NULL — пустой, фиктивный
NUMBER — число, номер
NUMERIC CHARACTER — цифра
OCTAL — восьмеричный
OFF-LINE — выключенный, автономный
ON-LINE — подключенный, доступный
OPEN — открыть (файл)
OPTION — необязательный параметр, вариант, средство
ORDER — порядок, степень
ORIGIN — начальный адрес
OUTPUT — вывод данных, выводное устройство
OVERLAY — перекрытие
PACK — упаковывать
PACKED — упаковывать
PAGE — страница, лист
PAINTBRUSH — программа рисования
PAPER — бумага
PARAMETER — параметр
PARITY — четность, контроль четности
PASS — проход, передавать
PASSWORD — пароль
PASTE — вставлять
PATH — путь доступа
PEN — ручка, перо
PICTURE — изображение, картинка
PIXEL — элемент растра, точка
POINT — точка
POINTER — указатель, ссылка
POKE — записать байт по адресу
PORT — порт, точка подключения ВУ к внутренней шине
PORTRAIT — вертикальный
POWER — мощность, степень
PRESS — нажимать (клавишу)
PRINTER — печатающее устройство
PRINT — печатать, выводить на дисплей
PROBLEM — задача
PROCESSING — обработка, выполнение
PROGRAM — программа
PROMPT — приглашение, вопрос
PURGE — очищать
PUSH — помещать на стек
PUT — выводить
QUERY — запрос
QWERTY KEYBOARD — клавиатура со стандартным американским расположением литер
RAM — оперативная память, ОЗУ
RAM DISK — псевдодиск, логическое устройство, обеспечивающее хранение файлов в специально выделенной области оперативной памяти
RANGE — диапазон, отрезок
RASTER — растр
RASTER UNIT — единица растра, шаг растра
READ — читать, считывать
REAL — вещественный, действительный
REASSIGN — переназначать
RECORD — запись
REGISTER — регистр. Внутреннее запоминающее устройство процессора или адаптера для временного хранения информации.
RELATION — отношение
RENAME — переименовать
RESET — сброс
RESTART — перезапуск, повторный запуск
RETURN — возврат
ROM — постоянное запоминающее устройство, ПЗУ
RUN — выполнение, запуск, счет
SAMPLE — выборка
SAVE — сохранять, записывать
SCALAR — скаляр
SCALE — масштаб
SCAN — просмотр, поиск; сканировать
SCHEMA — схема
SCREEN — экран
SCREEN EDITOR — экранный редактор
SEARCH — поиск, перебор; искать
SECONDARY — вторичный, вспомогательный
SECTOR — сектор, минимальная физически адресуемая единица запоминающего устройства на диске
SECTORING — разбиение на секторы, разметка
SELECT — выбирать, выделять, устанавливать связь
SHIFT — смена регистра, сдвиг
SIMULATOR — модель, имитатор
SIZE — размер, длина
SKIP — пропуск, прогон бумаги
SOFT — программируемый, непостоянный
SOFT KEY — программируемая клавиша
SOFTWARE — программа, программный
SOURCE — источник, исходный
SPACE — пространство, место, пробел
SPARE — запасной, свободный
SPECIFIER — описатель, спецификатор
SPRITE — спрайт. Аппаратное или программное средство формирования динамического графического изображения. Спрайт это растровое графическое изображение небольшого размера.
STACK — стек, магазин. Структура данных, в которой можно добавлять и удалять элементы данных; причем доступен только последний добавленный элемент.
STATE — состояние
STATIC — статический
STORAGE — память
SYMBOL — символ, обозначение, идентификатор
SYSTEM — система, вычислительная система, системный
TAB — символ табуляции, клавиша табуляции
TAPE — лента, ленточный
TAPE DRIVE — накопитель на магнитной ленте
TASK — задача
TEST — тестирование, проверка
TEXT — текст
TIMER — таймер, часы
TRACK — дорожка (ленты, диска, барабана)
TRANSLATOR — конвертер, транслятор. Программа, транслирующая текст на одном языке программирования в текст на другом языке
TRUE — истинный, истина
TYPE — тип (данных), печатать, вводить, набирать, выводить
UNIT — устройство, элемент, модуль
USER — пользователь
VALID — допустимый, правильный
VARIABLE — переменная
VECTOR — вектор, одномерный массив
VERSION — версия
VIDEO RAM — видеопамять, память изображения
WAITING — ждать, ожидание
WINDOW — окно
WORD — (машинное) слово
WORKING — рабочий
WRITE — писать
ZAP — затирать
ZERO — нуль, обнулять

— четность, контроль четности
PASS — проход, передавать
PASSWORD — пароль
PASTE — вставлять
PATH — путь доступа
PEN — ручка, перо
PICTURE — изображение, картинка
PIXEL — элемент растра, точка
POINT — точка
POINTER — указатель, ссылка
POKE — записать байт по адресу
PORT — порт, точка подключения ВУ к внутренней шине
PORTRAIT — вертикальный
POWER — мощность, степень
PRESS — нажимать (клавишу)
PRINTER — печатающее устройство
PRINT — печатать, выводить на дисплей
PROBLEM — задача
PROCESSING — обработка, выполнение
PROGRAM — программа
PROMPT — приглашение, вопрос
PURGE — очищать
PUSH — помещать на стек
PUT — выводить
QUERY — запрос

WORKING
WRITE — п
ZAP — зат
ZERO —