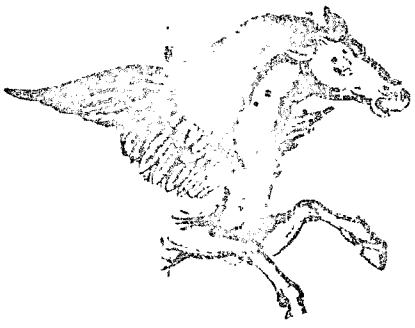
ПЕРСОНАЛЬНЫЙ КОМПЬЮТЕР БК-0010 БК-0011м

Приложение
к журналу
«ИНФОРМАТИКА
И ОБРАЗОВАНИЕ»

Выпуск
2'94 (3)

Издается с 1993 г.

В НОМЕРЕ



Москва «Информатика и образование» 1994

Авторы выпуска

Виницченко А. И.

Зальцман Ю. Л.

Манвелян В. З.

Минцлин М. В.

Прудковский А. Г.

Саяпин А. А.

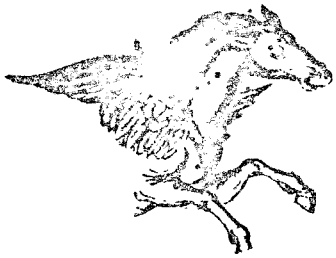
Усенков Д. Ю.

Фролкин Б. Ф.

РЕДАКТОРЫ: *ВАСИЛЬЕВ Б. М.*
УСЕНКОВ Д. Ю.

«Библиотека журнала «Информатика и образование»
Свидетельство о регистрации средства массовой информации № 0110336
от 26 февраля 1993 г.

Телефон: (095) 151-19-40, 208-30-78
E-Mail: mail@infoobr.msk.su
Факс: (095) 208-67-37



ИНФОРМАЦИОННОЕ СООБЩЕНИЕ ОРГАНИЗАЦИОННОГО КОМИТЕТА ПЕРВОГО ОТКРЫТОГО СОРЕВНОВАНИЯ ПРОГРАММИСТОВ РОССИЙСКОЙ ФЕДЕРАЦИИ

Профессиональный союз программистов России при участии Московского клуба пользователей бытовых компьютеров (Клуб БК) и издательства «Информатика и образование» впервые организует ОТКРЫТОЕ ПЕРВЕНСТВО ПРОГРАММИСТОВ.

Первенство проводится в следующих группах (по типу ПЭВМ):

— *ZX-SPECTRUM-совместимые;*

— *БК, УКНЦ, Союз-Неон и пр. (система команд DEC);*

— *ПОИСК (и другие IBM-совместимые класса не выше XT);*

— *профессиональные IBM-совместимые (класс AT и выше).*

Участники делятся на три категории:

A — профессиональные программисты;

B — пользователи и непрофессиональные программисты со стажем работы более 3 лет;

C — не входящие в категории A и B.

Оргкомитет Первенства принимает на магнитном носителе готовые программные разработки и их демонстрационные версии (если таковые существуют), соответствующую документацию, при необходимости — технические описания программ.

Ориентировочная тематика:

— *разработка нового программного обеспечения (общесистемные, инструментальные, прикладные, игровые программы, драйверы, интерфейсы и т.п.);*

— *модификация, расширение возможностей, адаптация существующих разработок.*

Авторские права участников Первенства будут защищены приоритетной справкой или Свидетельством об авторстве на «ноу-хау».

К посылаемому в адрес оргкомитета материалу необходимо прилагать в письменном виде (и на магнитном носителе):

— *краткую справку о представленной работе (историю разработки), аннотацию;*

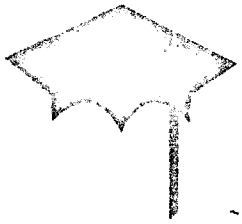
— *сведения об авторе (паспортные данные, почтовый адрес, телефон) и соавторах (с указанием % вклада каждого).*

СРОК ПОДАЧИ МАТЕРИАЛА — НЕ ПОЗДНЕЕ 15 СЕНТЯБРЯ 1994 г.

Подведение итогов будет производиться в конце 1994 г. на основании рейтинга его участников. (Под рейтингом подразумевается пользовательская значимость и профессиональность исполнения заявленного программного продукта в соответствии с категорией, что определяется оргкомитетом.)

ПОБЕДИТЕЛИ и призеры Первенства будут удостоены почетных званий и отмечены призами и наградами. Авторы лучших работ получат возможность бесплатной рекламы и коммерческого распространения своего программного продукта через журналы издательства «Информатика и образование».

Материалы, предложения и идеи направлять по адресу:
129337, Москва, а/я 125, оргкомитет Первенства программистов. Тел.: (095) 183-18-39.
(Обязательно прилагать два чистых конверта с полным набором марок.)



Продолжаем публикацию неформального учебника по программированию на ассемблере для начинающих пользователей БК-0010(.01). Первые уроки этого компьютерного языка см. в №1 за 1994 г.

Ю. А. Зальцман,
г. Алма-Ата

МикроЭВМ БК-0010. Архитектура и программирование на языке ассемблера

Системные регистры

Совершая путешествие по нашему «городу» БК-0010, мы отметили, что «район» ПЗУ тянется почти до конца адресного пространства — до адреса 177577. А что же дальше? С адреса 177600 и до конца «улицы» (по адрес 177777) располагается так называемая ОБЛАСТЬ СИСТЕМНЫХ РЕГИСТРОВ.

Системные регистры микроЭВМ служат для связи с внешними устройствами, а также для некоторых других целей. Каждый регистр занимает в адресном пространстве одно слово, следовательно, имеет 16д разрядов, с нулевого по пятнадцатый. Простой арифметический подсчет показывает, что в области адресов 177600 ...177776 содержится 100 слов, а значит, там можно разместить 100 (64д) регистров. Но далеко не все адреса этой области использованы в нашем компьютере. Рассмотрим имеющиеся в БК-0010 системные регистры в порядке возрастания их адресов. При этом нужно учитывать, что и в имеющихся регистрах далеко не все разряды (биты) используются. Некоторые разряды хранят только определенную, строго фиксированную информацию, которую нельзя изменить, подобно данным, хранимым в ПЗУ. В других информацию меняет только сама ЭВМ, а программист может лишь читать ее и использовать результаты (говорят, что такие разряды ДОСТУПНЫ ПО ЧТЕНИЮ). В некоторые разряды регистров можно, наоборот, только записать информацию, но прочитать то, что записано, нельзя (разряды, ДОСТУПНЫЕ ПО ЗАПИСИ). Наконец, есть разряды, в которые можно как записывать информацию, так и читать, — они ДОСТУПНЫ ПО ЗАПИСИ И ЧТЕНИЮ.

Чем системные регистры (например, доступные по записи и чтению) отличаются от ячеек ОЗУ? Отличие существенное — информацию в ячейках памяти может записывать, читать и использовать только ЦП, он же выводит ее на экран по вашей команде. А разряды системных регистров могут быть связаны непосредственно с внешними устройствами — с клавиатурой, магнитофоном, телеграфной линией, — словом, с внешним миром; могут принимать оттуда сигналы или, наоборот, передавать. Есть, правда, регистры, казалось бы, ни с чем не связанные, например таймер (о котором речь пойдет позже). Но это не более чем заблуждение: таймер тоже связан с внешним миром — через физическое время!

А что же отсутствующие регистры? Это просто резерв для расширения нашей микроЭВМ в будущем, так сказать, пустые пока «участки под застройку» (в машинах серии ДВК, а также в БК-0010Ш, работающих в составе комплекса учебной вычислительной техники КУВТ-86, часть из них уже используется).

Итак, приступим к прогулке по последнему «переулку» адресного пространства. Вы готовы? Тогда — в путь!

Напомним, что каждый регистр имеет 16 разрядов, нумеруемых справа налево, причем нумерация начинается с нуля. Номера разрядов — десятичные. По ходу их перечисления будем также указывать, для чего можно использовать тот или иной разряд при программировании (как их использует ЭВМ — другой вопрос).

1. РЕГИСТР СОСТОЯНИЯ КЛАВИАТУРЫ (адрес 177660). Используются только два разряда:

- РАЗРЯД 06 — маска прерываний от клавиатуры. Если бит равен 0 — прерывание разрешено, 1 — запрещено. Доступен по записи и чтению. Можно использовать, чтобы программно «отключить» клавиатуру ЭВМ.
- РАЗРЯД 07 — флаг состояния клавиатуры. Устанавливается в 1 при поступлении новых данных в РЕГИСТР ДАННЫХ КЛАВИАТУРЫ (см. ниже), например при нажатии очередной клавиши, и сбрасывается в 0 при чтении. Доступен только по чтению. Если запретить автоматическое чтение из регистра данных, этот разряд может использоваться, чтобы определить, была ли нажата клавиша за истекший промежуток времени. В обычных условиях, когда данные читаются сразу после нажатия клавиши, состояние разряда всегда 0, так как системные драйверы БК реагируют на нажатие клавиши быстрее, чем пользовательская программа.

2. РЕГИСТР ДАННЫХ КЛАВИАТУРЫ (адрес 177662). Используются семь младших разрядов:

- РАЗРЯДЫ 00...06 — буфер кода нажатой клавиши. При нажатии клавиши в него заносится 7-разрядный код, на который не влияют клавиши «ПР» («АР2» для БК-0010.01) и «РУС». Полные коды клавиатуры получаются в БК-0010 программным путем. Разряды доступны только по чтению и широко используются для чтения кода клавиатуры независимо от регистра РУС-ЛАТ, для ввода команд без прерывания работы программы (в том числе и на языках высокого уровня) и т.п.

3. РЕГИСТР СМЕЩЕНИЯ (адрес 177664). Используются 9 разрядов:

- РАЗРЯДЫ 00...07 отображают адрес начала экранного ОЗУ, которое организовано по типу рулона, причем их значение указывает количество телевизионных строк дисплея (каждая строка — 100 (64д) байт экранного ОЗУ). В исходном состоянии, когда началу экрана соответствует адрес 40000, содержимое этих разрядов равно 330. По мере сдвига экрана содержимое меняется, причем, так как одна строка текста в БК-0010 содержит 12 (10д) телевизионных строк, содержимое регистра также меняется каждый раз на 12. Вверху экрана, как известно, имеется служебная строка, занимающая в экранном ОЗУ 2000 байт (каждая строка обычного текста — 1200 байт). Эти тонкости приводят к тому, что число сдвигов, необходимое, чтобы начало экрана получило снова адрес 40000, не кратно числу строк текста (24д) и содержимое младших разрядов данного регистра практически все время разное. Разряды доступны по записи и чтению и могут использоваться для плавного сдвига изображения на экране по вертикали. Служебная строка при этом тоже смещается, поэтому данный способ требует специальных приемов получения изображения.

  Нужно отметить, что в режиме «РП» экран организован совсем иначе, чем описано выше, служебная строка (начало экрана) всегда имеет адрес 70000, рулонное смещение не используется, а регистр всегда содержит константу 230.

- РАЗРЯД 09 — задание режима расширенной памяти, что соответствует нулю в данном бите. Единица — обычный режим (в **«Руководстве системного программиста» ошибочно указано наоборот).** Разряд доступен по записи и чтению, может использоваться для обнаружения режима «РП», но включить расширенную память, просто записав туда 0, нельзя (вернее, этого недостаточно).

4. РЕГИСТРЫ СИСТЕМНОГО ТАЙМЕРА (адреса 177706, 177710, 177712). Эти три регистра не описаны ни в одном руководстве по БК-0010, поэтому заслуживают подробного рассмотрения.

Прежде всего, что такое системный таймер? Это специальный счетчик, работающий независимо от ЦП. Если запустить такой счетчик, а потом в какие-либо фиксированные моменты времени читать его показания, то по их разности можно определить прошедший отрезок времени.

Итак, наш таймер имеет три регистра. Первый из них — РЕГИСТР ПРЕДУСТАНОВКИ таймера (адрес 177706) — доступен по записи и чтению, используются все 16 разрядов. Второй — СЧЕТЧИК (177710) — собственно таймер, доступен только по чтению, используются все 16 разрядов. Третий — РЕГИСТР УПРАВЛЕНИЯ (177712) — доступен по записи и чтению, используется младший байт (разряды 00—07).

Попробуем разобраться, как работает таймер. Если мы подадим определенную команду (запишем в регистр 177712 некоторое число), то в регистр счетчика 177710 будет занесена константа, всегда равная содержимому регистра предустановки, и начнется отсчет времени — из константы будет вычитаться по единице до получения в счетчике нуля. Дальнейшее поведение таймера зависит от занесенной в регистр управления команды, а именно

от того, какие разряды младшего байта регистра управления установлены в 0 или 1. Рассмотрим далее их назначение.

- Разряд 00 при установке в 1 запрещает счет и переписывает в регистр счетчика константу из регистра предустановки (РЕЖИМ ПРЕДУСТАНОВКИ).
- Разряд 01 при установке в 1 запрещает фиксацию перехода счетчика через 0 (РЕЖИМ НЕПРЕРЫВНОГО СЧЕТА): досчитав до нуля, таймер продолжает вычитание, в счетчике появляется число 177777, затем 177776 и т.д. Действие разрядов 02 и 03 (см. ниже) при этом, естественно, отменяется, как и повторная перезапись константы из регистра предустановки.
- Разряд 02 при установке в 1 включает индикацию: при очередном переходе таймера через 0 устанавливается в 1 разряд 07 регистра управления, если ранее он был сброшен (РЕЖИМ ИНДИКАЦИИ). Нужно учитывать, что при первом (после включения ЭВМ или «системного сброса») запуске таймера в данном режиме индикация срабатывает только после ВТОРОГО перехода счетчика через 0, причем независимо от того, работал ли таймер до этого в других режимах.
- Разряд 03 при установке в 1 запрещает повторный счет: после первого досчета до 0 устанавливается в 0 разряд 04 (РЕЖИМ ОДНОКРАТНОГО СЧЕТА). Установка данного режима не отменяет режим индикации (при установке в 1 разряда 02).
- Разряд 04 при установке в 1 разрешает счет (РЕЖИМ СЧЕТА). В этом режиме при досчете до 0 в счетчик заново заносится константа из регистра предустановки, следовательно, счет всегда ведется от константы до 0 (если только не запрещена фиксация перехода через 0 разрядом 01). При сбросе разряда 04 в 0 в счетчик переписывается константа предустановки.
- Разряд 05 при установке в 1 снижает скорость счета в 4 раза (РЕЖИМ УМНОЖЕНИЯ ВРЕМЕНИ НА 4).
- Разряд 06 при установке в 1 снижает скорость счета в 16д раз (РЕЖИМ УМНОЖЕНИЯ ВРЕМЕНИ НА 16). При одновременной установке в 1 обоих разрядов — 05 и 06 — скорость счета снижается в 64д раза.
- Разряд 07 используется для индикации перехода счетчика таймера через 0 (при установленном режиме индикации, см. разряд 02).

Таким образом, возможно множество различных команд таймера и, соответственно, вариантов его использования. Рассмотрим некоторые из них.

Пусть в регистр управления записано число устанавливающее в 1 разряд 04 (разрешен счет), а разряды 00...04 установлены в 0. В регистре предустановки имеется некоторая константа. После занесения команды в регистр управления число из регистра предустановки перепишется в счетчик и там начнется вычитание из него по единице в определенном темпе — обратный счет времени. Как только содержимое счетчика 177710 станет равно нулю, в него снова будет переписано число из 177706, снова начнется вычитание и т.д. Задавая в 177706 различные числа, можно изменять ВРЕМЯ ЦИКЛА таймера — это «тонкая регулировка» хода наших часов. Но есть и грубая — в зависимости от записанного в 177712 кода команды можно замедлять его «ход» в 4, 16 и 64 раза, для чего достаточно задать равными 1 один или оба разряда — 05 и 06 — «множители времени». Если разряд 03 равен 1, то таймер выполнит только один цикл счета, а если задать равным 1 разряд 02 и сбросить в 0 разряд 07, то после перехода счетчика через 0 окончание цикла таймера будет отмечено единицей, появившейся в разряде 07. Если же установить в 1 разряд 01, то, выполнив один цикл счета от константы предустановки до 0, счетчик продолжит счет «вкруговую» уже без учета константы предустановки.

При работе таймера отсчитанный промежуток времени определяется умножением разности показаний счетчика за истекшее время на те множители, в разряды которых записаны единицы, и на ЕДИНИЧНЫЙ ПЕРИОД ВРЕМЕНИ. Этот период, измеренный для нескольких БК-0010, в среднем равен 42.9 мкс и может для разных экземпляров БК варьировать в пределах долей процента (реже — нескольких процентов).

Приведем пример. Пусть мы, запуская таймер, записали в регистр 177712 код 160 (единицы в разрядах 04, 05 и 06) и получили разность показаний таймера за отсчитанный промежуток времени, равную 154432 (55578д). Вычислим: 55578 x 4 x 16 x 42.9 = 152594956.8 мкс, или примерно 152.6 с. Естественно, чтобы отсчитать такую разность, необходимо иметь в регистре 177706 число, большее 154432. Приведем таблицу, которая позволит установить максимальное время, отсчитываемое таймером в пределах одного цикла (см. табл. 1).

Как можно понять из вышеизложенного, максимальный цикл задается числом 177777, занесенным в регистр 177706 (вместо этого можно просто обнулить регистр 177706, тогда после первого же вычитания произойдет заем

в старшем разряде, и число станет равно 177777).

Таким образом, таймер позволяет отсчитывать промежутки времени до 3 мин. А если надо больше? Тогда придется написать программу, которая могла бы фиксировать моменты перехода счетчика таймера через ноль и подсчитывать число таких переходов. Цикл таймера при этом целесообразно задать кратным какой-либо целой единице времени. Попробуем задать время цикла равным 1 мин.

Таблица 1

Код	Множитель	Макс.время цикла, с
20	x $1_Д$	2,812
120	x $4_Д$	11,25
60	x $16_Д$	45,00
160	x $64_Д$	180,0

Выберем константу предустановки. Ясно, что подходит лишь код 160 — все другие дают время цикла менее 1 мин. Подсчитаем константу. Известно, что 1 мин = 60 миллионов мкс. Вычислим: (60000000/64) /42.9= 21853 (округленно). Переведем в восьмеричную систему: $21853_Д$ = 52535. Остается записать полученное число в 177706, а код 160 — в 177712, и счетчик таймера будет выполнять цикл длительностью ровно в 1 мин. Эксперименты с таймером удобно проводить в МСД, используя для непрерывного чтения регистра 177710 команду «Ц».

Для фиксации количества циклов таймера (моментов перехода счетчика через 0) можно предложить разнообразные приемы, например периодически опрашивать регистр 177710 (не менее двух раз в течение цикла), использовать разряд индикации регистра 177712 или повторно запускать таймер в режим однократного счета.

Для чего еще может понадобиться таймер, кроме отсчета времени? Это идеальный генератор случайных чисел, если только обращения к нему производятся не программой, а человеком (например, по нажатии клавиши). Такой генератор будет давать не псевдослучайные, а истинно случайные числа, так как момент нажатия оператором клавиши зависит от очень многих факторов, и угадать показание таймера в этот момент невозможно. Если же к таймеру обращается программа, последовательность будет далека от случайной, так как и таймер, и процессор используют одну и ту же ТАКТОВУЮ ЧАСТОТУ, и поэтому процессы в них коррелируют. (Генерация случайных чисел с помощью таймера использована в известной игре для БК STREAP SHOW. — *Прим. ред.*)

В заключение несколько замечаний. Системный таймер, будучи запущенным, работает автономно от процессора, поэтому его счетчик отсчитывает время независимо от того, стоит процессор или работает, есть в БК работающая программа или нет. Таймер одинаково успешно работает в БЕЙСИКе и ФОКАЛе, с программами в кодах, запущенными из МСД и монитора, и даже в тот момент, когда БК-0010 работает с магнитофоном. Но опрос регистра 177710 (если для подсчета числа циклов таймера используется этот способ) необходимо проводить не реже, чем два раза за цикл таймера, иначе можно пропустить переход через 0 и показания «часов» исказятся. Таймер, в свою очередь, «стоит» ли он или «идет», никак не влияет на работу программы пользователя в том числе и на скорость ее исполнения. И наконец, одно не совсем приятное замечание. Поскольку таймер по техническим условиям в состав БК-0010 не входит, то на некоторых экземплярах его регистры могут быть неисправны, что не служит основанием для предъявления претензий заводу-изготовителю. К счастью, эта неисправность, довольно распространенная на БК-0010 3—5 лет назад, сейчас встречается все реже, так что можно смело использовать таймер в своих программных разработках.

При выходе в БЕЙСИК или ФОКАЛ таймер останавливается, поэтому запускать его нужно в той среде, в которой он будет работать.

(Еще одной сферой применения таймера является организация на БК-0010 многопрограммного режима работы. Подробно о выполнении на БК нескольких программ одновременно говорится в журнале «Информатика и образование», №2 за 1992 г. — *Прим. ред.*)

5. РЕГИСТР ПОРТА ВВОДА-ВЫВОДА (адрес 177714). Используются все 16 разрядов, доступные по записи и чтению. Но этот регистр особенный, он как бы составлен из двух отдельных регистров для ввода и вывода. Данные, записываемые по адресу 177714, передаются в регистр вывода, а читаются из регистра ввода. Прочитать данные, записанные в регистр вывода, невозможно, поэтому, если необходимо сохранить к ним доступ, нужно записать их не только в порт, но и в

специально зарезервированную для этого ячейку памяти (адрес 256). Необходимо заметить, что порт вывода инвертирует подаваемый на него код, т.е. при записи в него нулей на выходе будут единицы и наоборот. То же самое можно сказать и о порте ввода: при подаче на него от внешних устройств логических единиц программно читаются нули, при подаче нулей — единицы. При «привязке» к порту внешних устройств нужно это учитывать.

6. РЕГИСТР СИСТЕМНЫХ ВНЕШНИХ УСТРОЙСТВ (адрес 177716). Используются все 16 разрядов, но для разных целей, и поэтому они неравнозначны.

- РАЗРЯДЫ 0...3 предназначены для внутренних нужд ЭВМ, служат для задания режима работы процессора, доступны только по чтению.

- РАЗРЯДЫ 4...7 предназначены для управления системными внешними устройствами ЭВМ (магнитофоном, телеграфным (ТЛГ) каналом, клавиатурой, встроенным динамиком), и на них мы остановимся подробнее. Эти четыре разряда представляют собой внутренний порт ввода-вывода и организованы совершенно так же, как только что описанный внешний порт. Это как бы два отдельных регистра ввода и вывода, поэтому запись и чтение информации по этим разрядам имеют разное значение.

Разряды регистра ВВОДА:

РАЗРЯД 04 служит для чтения информационного сигнала с ТЛГ-линии.

РАЗРЯД 05 — для чтения информации с магнитофона.

РАЗРЯД 06 — индикатор нажатия клавиши: если нажата любая клавиша (кроме «СТОП» и регистровых «ПР», «ПР», «СУ», «СТР», «ЗАГ»), то в этом разряде 0, если ни одна из клавиш не нажата — 1. Широко используется в программах как в кодах, так и на языках высокого уровня для индикации нажатия клавиш, ввода информации «на ходу» (без останова программы), организации циклов автоповтора и т.п. В самой ЭВМ используется для организации режима «повтор».

РАЗРЯД 07 служит для чтения сигнала готовности с ТЛГ-линии. Этот разряд, как и 04, может использоваться как по прямому назначению, так и для других целей, это дополнительный канал связи ЭВМ с внешними устройствами. Для задействования ТЛГ-ввода/вывода на плате ЭВМ необходимо установить перемычки (на БК-0010, выпускаемых некоторыми предприятиями, перемычки установлены, а на других — нет, так что этот вопрос нужно решать применительно к конкретному компьютеру).

Разряды регистра ВЫВОДА:

РАЗРЯД 04 служит для передачи информации на ТЛГ-линию, его исходное состояние — 1.

РАЗРЯД 05 — для передачи сигнала на магнитофон (при записи) или сигнала готовности на ТЛГ-линию, поэтому одновременный обмен информацией с магнитофоном и ТЛГ-линией невозможен. Исходное состояние — 0.

РАЗРЯД 06 — для передачи информации на магнитофон (при записи) и сигнала на пьезодинамик ЭВМ (при записи на МЛ и нажатии клавиши). Очень широко используется в программах в кодах и на БЕЙСИКе для создания звуковых эффектов: если с определенной периодичностью записывать в этот разряд чередующиеся 0 и 1, мы услышим звук, воспроизводимый как встроенным пьезодинамиком, так и подключенным к выходу БК «МАГНИТОФОН» любым усилителем низкой частоты. Исходное состояние — 0.

Здесь необходимо одно пояснение: почему для выдачи информации на магнитофон используются два разряда — 05 и 06. Дело в том, что в БК-0010 для записи на магнитофон был принят двухуровневый сигнал. При записи «нуля» сигнал идет с обоих разрядов и имеет большую амплитуду, а при записи «единицы» — только с разряда 06 и амплитуда его меньше. Это сделано для коррекции частотной характеристики тракта записи магнитофона и получения более качественной записи информации. Но в БК-0010 поздних выпусков от этого отказались, сигнал на магнитофон с разряда 05 уже не передается, в схему компьютера внесены соответствующие изменения, но программа обслуживания выхода на магнитофон осталась той же и по-прежнему обслуживает оба разряда. Эти изменения могут отсутствовать и в БК-0010, выпускаемых некоторыми из предприятий.

РАЗРЯД 07 служит для дистанционного управления двигателем магнитофона: при записи в него нуля двигатель включается, а единицы — отключается. Управление двигателем производится через установленное на плате БК электромагнитное реле.

- РАЗРЯДЫ 08...15 предназначены для задания адреса запуска системы и доступны только по чтению. Адрес запуска в БК-0010 принят равным 100000, он и задан в старшем байте регистра 177716, причем его младший байт принимается равным 0.

Контрольные вопросы и задания

1. Как известными вам способами проследить за изменениями в регистре 177662 при

нажатии разных клавиш? Проверьте это экспериментально.

— Дайте в МСД директиву 177662АЦ, затем нажимайте различные клавиши и наблюдайте на экране их коды.

2. Записывая в регистр 177664 различные числа, проследите за изменениями положения изображения на экране. Желательно предварительно заполнить экран каким-нибудь текстом, чтобы иметь ориентиры помимо служебной строки.

3. Придумайте и осуществите на своем БК проверку работоспособности системного таймера.

— Дайте в МСД директивы: 177706А0,,160-Ц — при этом будет циклически читаться регистр счетчика таймера. Отметьте по секундомеру время между двумя моментами перехода таймера через ноль, этот интервал должен быть с точностью до нескольких секунд равен 3 мин, в противном случае таймер на вашем БК неисправен. Приостановить вывод информации на экран в МСД можно командой СУ/@, повторный пуск — любая клавиша или еще раз СУ/@. Обратите внимание на то, что при прерывании вывода на экран таймер продолжает работать, после возобновления вывода его показания уже другие.

4. Можно ли прочитать информацию, занесенную в МСД по адресу 177714? Придумайте, как все-таки это сделать. Все необходимое имеется в комплекте, прилагаемом к БК-0010.

— Программно прочитать регистр вывода 177714 нельзя, но можно подать информацию с него на регистр ввода, для чего служит входящий в комплект ЭВМ «блок нагрузок». Подключив этот блок к порту ввода-вывода, можно читать из регистра 177714 информацию, записываемую по тому же адресу.

5. Как проверить, не разбирая компьютер, подключен ли на вашей ЭВМ разряд 05 регистра 177716 к выходу на магнитофон?

— Если имеется электронно-лучевой осциллограф, можно посмотреть форму импульсов на выходе «МГ» БК при записи информации на магнитофон. Если разряд задействован, импульсы будут двух разных амплитуд, если нет — только одной.

Как работает ЭВМ

Не пугайтесь, автор не собирается рассказывать вам о триггерах, логических элементах, регистрах сдвига, счетчиках, источниках питания и прочей электронике, это не входит в его задачу. Речь пойдет о том, как компьютер работает с точки зрения программиста, для понимания дальнейшего изложения это необходимо.

Как уже было сказано, в памяти ЭВМ хранится программа — последовательность двоичных кодов. При ее запуске процессор запоминает адрес начала программы в специальном регистре, называемом СЧЕТЧИКОМ КОМАНД, и читает первое слово по этому адресу, причем счетчик команд сразу же увеличивает свое содержимое на 2. В зависимости от того, какая команда прочитана, процессор может или сразу выполнить ее (если в ней заключены все необходимые данные), или заняться «добыванием» указанных в команде данных. Последние либо могут располагаться где-то в памяти (как уже говорилось, для программы и данных используется одно и то же ОЗУ), либо входят в состав самой команды, но не в первое ее слово. Команды процессора БК-0010 могут иметь длину от одного до трех слов, причем первое из них всегда код самой команды, а два последующих, если они есть, — данные (это могут быть как сами числа для выполнения над ними каких-то операций, так и указания на адреса, по которым они расположены). Если команда состоит более чем из одного слова, содержимое счетчика команд в процессе ее выполнения увеличивается не на 2, а на 4 или на 6. Таким образом, счетчик команд всегда содержит адрес следующей команды программы.

После окончания исполнения очередной команды процессор опять читает слово по адресу, указанному в счетчике команд, выполняет следующую команду и т.д. Ясно, что таким образом может исполняться так называемая ЛИНЕЙНАЯ ПОСЛЕДОВАТЕЛЬНОСТЬ, когда команды следуют строго одна за другой. Но из теории программирования известно, что линейная последовательность не является достаточной для решения любых задач, необходимы еще по крайней мере две структуры — УСЛОВИЕ и ЦИКЛ. Тот, кто занимался программированием хотя бы на одном из языков, знает, что исполнение этих структур связано с изменением естественной последовательности команд. В ЭВМ это решается очень просто — достаточно прибавить к счетчику команд какое-либо число (положительное или отрицательное), и процессор послушно начнет выполнять команды уже в другом месте программы, в соответствии с адресом, указанным счетчиком.

Как же «устроена» любая машинная команда? Не вникая в подробности, можно сказать, что в ее 16 разрядах (если команда состоит из одного слова) есть код самой команды и указание на так называемые ОПЕРАНДЫ, т.е. числа, с которыми нужно иметь дело. Если эти числа входят в состав команды,

то, как уже говорилось, они содержатся в ее втором или третьем слове. Кроме того, команда содержит еще и указание на то, нужно ли оперировать со словом или с байтом. Все эти сведения располагаются в специально отведенных для этого разрядах первого слова команды, так называемых ПОЛЯХ. Этим кратким описанием мы пока и ограничимся.

А что представляют собой числа, с которыми оперирует ЭВМ?

Дополнительный код

ЭВМ, как легко догадаться, оперирует с двоичными числами. Положительные двоично-восьмеричные числа от 0 до 7 мы уже рассматривали. Но это еще не все. Для полноценной арифметики нужны и отрицательные числа, а они изображаются в ЭВМ не совсем обычным образом. Чтобы понять, как это делается, произведем несколько действий с двоичными 4-разрядными числами. Сначала сложение:

0001	1
+ 0011	+3
0100	4

А теперь — вычитание:

0001	1
−0011	−3
1110	−2

Но почему у нас получилось 1110, и почему это −2? При вычитании нам потребовалось сделать «заем» в старших разрядах уменьшаемого, а поскольку «цена» каждого разряда двоичного числа равна двойке, то у нас и получилось то, что получилось. При этом мы считали, что число у нас как бы «неограниченной» разрядности, и мы «заняли» единицу в следующем, пятом разряде, который не изображен.

Теперь — внимание! Старший разряд двоичного числа со знаком (в данном случае — четвертый) называется ЗНАКОВЫМ. Если число положительное, он равен 0, а если отрицательное — 1. Значит, раз у нас число 1110 и старший разряд 1, то число это — отрицательное. А как понимать остальные 3 разряда — 110? Почему это 2? Вот это и есть ДОПОЛНИТЕЛЬНЫЙ КОД. Какой в нем смысл? Попробуем выполнить еще два действия:

1110	−2	0101	5
+1110	+−2	+1110	+−2
1100	−4	0011	3

А правда ли, что мы получили в первом случае −4? Проверим:

0000	0
−0100	−4
1100	−4

Результат очевиден — 1100 это действительно −4. Так вот в чем смысл дополнительного кода: мы можем не задумываться о знаках при операциях с положительными или отрицательными числами, они получаются автоматически! Очевидно, что во втором сложении результат тоже верен, 0011 — это 3. Теперь вам ясно, как получается дополнительный код? Чтобы получить отрицательное число в дополнительном коде, нужно вычесть соответствующее по абсолютной величине положительное число из нуля. Потому код и называется дополнительным, что отрицательное и положительное числа, равные по абсолютной величине, ДОПОЛНЯЮТ друг друга до нуля, или, как говорят, ДО ПЕРЕНОСА в разряд за пределы числа. Есть простое правило получения отрицательного числа, которое позволяет делать перевод в дополнительный код в уме. Если вам нужно получить отрицательное число, возьмите положительное, равное ему по абсолютной величине, замените в нем все единицы нулями, а нули — единицами и прибавьте единицу. Попробуем. Берем 4: 0100. Заменяем: 1011. Прибавляем единицу: 1011+0001=1100. Получилось −4 — правило работает. Теперь очень легко получить любое отрицательное число в дополнительном коде. Например, возьмем число 01101100 (154). Заменим нули и единицы (это называется ПОРАЗРЯДНЫМ ИНВЕРТИРОВАНИЕМ): 10010011. Прибавим единицу: 10010100. Это минус 154. Можете проверить:

00000000	000
−01101100	−154
10010100	−154

Если же нам нужно перевести число из дополнительного кода обратно, делаем «все

наоборот»: вычитаем единицу и поразрядно инвертируем полученное двоичное число. Проверьте сами, и получите абсолютную величину отрицательного числа. И так — для чисел любой разрядности. А теперь попробуем выполнить еще одно действие:

```
 0110          6
+0011        + 3
 1001         −7
```

Вот это фокус! Дополнительный код почему-то «отказал»! А дело в том, что произошло так называемое ПЕРЕПОЛНЕНИЕ, перенос единицы в знаковый разряд.То есть простонапросто заданная разрядность оказалась мала для сложения двух предложенных чисел. Все станет на место, если мы добавим еще один разряд слева (и будем теперь его считать знаковым):

```
 00110          6
+ 00011        +3
 01001          9
```

Но наша ЭВМ оперирует не любыми числами, а только 8- и 16-разрядными, т.е. байтами и словами. Поэтому и знаковый разряд здесь фиксирован — это бит 07 для байта и 15 для слова. Как же быть при сложении на ЭВМ, если у нас фиксированный формат числа и добавить разряд мы не можем? Это предусмотрено — компьютер отмечает переполнение и дает о нем знать специальным «флагом» результата, о котором мы поговорим чуть позже. Обнаружив, что этот «флаг» установлен, мы видим, что результат в дополнительном коде неверен и нуждается в коррекции.

Есть в ЭВМ и еще один формат чисел — так называемые ЧИСЛА БЕЗ ЗНАКА. При этом мы включаем знаковый разряд в состав самого числа, которое в этом случае может быть только положительным. Ясно, что при этом число 1001 — это 9. Зачем еще и такое представление чисел? Очень просто — число при этом может быть больше на один разряд. Если для выражения числа 9 в дополнительном коде нам потребовалось 5 разрядов, то для числа без знака — только 4. Если в один байт можно записать числа в дополнительном коде от −128 до +127, то без знака — от 0 до +255. А нередко знак числа нас вовсе не интересует, поэтому не нужен и дополнительный код. Например, если надо сравнить коды символов, то при чем тут знак?

Иное дело — арифметика. Тут ради знака приходится жертвовать разрядностью числа. Но самое главное, что обычный двоичный код и код дополнительный — это, по сути, одно и то же, и мы в принципе можем оперировать числами независимо от того, в дополнительном коде они или нет. А если нам вдруг нужен знак числа, посмотрим на его старший разряд — и все.

Попробуйте поупражняться в сложении и вычитании двоичных чисел со знаком и без него, и вы поймете, что правила все время одни и те же. И еще вы заметите такую деталь: вместо того чтобы вычесть число из другого, можно прибавить к первому числу второе в дополнительном коде, ведь это есть отрицательное число! Вот оно, главное преимущество дополнительного кода, вот зачем он введен на ЭВМ: мы можем не иметь отдельного устройства — «вычитателя», достаточно только сумматора и перевода вычитаемого в дополнительный код. Это намного упрощает устройство процессора и увеличивает скорость вычислений.

А может ли ЭВМ так же просто умножать и делить? Да. Попробуем «сдвинуть» какое-либо число влево:

00010 сдвигаем влево: 00100; еще раз: 01000

У нас было число 2, получили число 4, а затем 8. Но ведь это умножение на 2! Да, так называемый АРИФМЕТИЧЕСКИЙ СДВИГ ВЛЕВО есть умножение на 2. При этом мы считаем, что на место младших разрядов числа пишется 0. Совершенно очевидно, что, сдвигая такое число вправо, по тому же правилу мы получим деление на 2. А если число со знаком минус? Попробуем его умножить на 2 тем же способом:

10011 (−13д) сдвинем влево: 00110

Получили... +6! Разве это правильно? Мы опять столкнулись с тем же явлением — не хватает разрядности. Но на этот раз произошло не переполнение, а ПЕРЕНОС, т.е. выход за пределы числа. В ЭВМ на этот случай тоже есть «флаг», который отметит, что результат ошибочный и нуждается в коррекции. Ну а как с делением чисел со знаком на 2? Тут все в порядке, но... правило сдвига вправо было пока неполным: сдвигая число вправо, мы должны не вписывать слева нули, а РАСШИРЯТЬ ЗНАКОВЫЙ РАЗРЯД, т.е. переписывать его значение в соседний справа бит:

10011 (−13) сдвинем вправо: 11001 (−7)

Все верно, ведь результат округленный! Причем заметим, что округление происходит

из-за отбрасывания младшего разряда. Но что, если нам нельзя округлять число? Ничего страшного, ведь у нас снова произошел перенос за пределы слова и «флаг» переноса отметит, что результат нуждается в коррекции.

Итак, мы в общих чертах разобрались, как ЭВМ выполняет арифметические действия. Нетрудно теперь придумать, как можно умножить или разделить на число, не кратное двум (вспомните, что умножение — это не что иное, как многократное сложение, деление же — многократное вычитание, а также вспомните, как умножают «в столбик» и делят «уголком»). Не будем разбирать соответствующие примеры, так как это заняло бы слишком много места, а пойдем дальше в изучении архитектуры нашей ЭВМ.

Контрольные вопросы и задания

1. Как умножить двоичное число на 128?

— Нужно выполнить 7 арифметических сдвигов влево.

2. Придумайте правило, которое позволило бы умножать двоичные числа на 3, и сформулируйте его словами.

— Чтобы умножить двоичное число на 3, нужно выполнить арифметический сдвиг влево и прибавить исходное число.

3. Пусть нам нужно сделать вычисления по формуле: A/8+B/4+C/2=D. При предварительном делении во всех числах может получиться округление за счет потери младших разрядов и точность результата снизится. Как можно свести потери точности к минимуму?

— Надо привести все дроби к общему знаменателю, получим: (A+2*B+4*C)/8=D. При таком порядке вычислений деление будет выполняться только один раз и возможная потеря точности от округления будет минимальной. Но в конкретной программе необходимо, конечно, учитывать возможность переполнения при последовательных умножениях.

Центральный процессор

Теперь пришло время более подробно рассмотреть главную часть ЭВМ — центральный процессор (ЦП). Как уже говорилось, ЦП — это универсальное логическое устройство для операций с числами. Но это не все. Процессор имеет свою собственную память, так называемые РЕГИСТРЫ ОБЩЕГО НАЗНАЧЕНИЯ (РОН). Этих регистров 8, и они, как и ячейки памяти и системные регистры, имеют по 16 разрядов. В них также могут быть записаны или оттуда прочитаны числа. Но в остальном регистры ЦП существенно отличаются от ячеек памяти. Прежде всего, они не входят в общее адресное пространство ЭВМ и обращения к ним особые, с применением специальных команд. Затем операции с числами в регистрах ЦП выполняются намного быстрее, чем в памяти. И наконец, целый ряд команд ЭВМ может быть выполнен только с использованием регистров, и никак иначе. Таким образом, это как бы отдельная специфическая память ЭВМ — «сверхоперативная память», имеющая большие преимущества перед ОЗУ. Жаль только, что этих регистров так мало.

Название «регистры общего назначения» не совсем точно. Действительно, шесть из них совершенно равноправны и вполне могут быть так названы. Они применяются для любых целей, их «имена» — R0, R1, R2, R3, R4, R5. Но два других регистра — специальные, их использование наравне с остальными крайне ограничено или исключено вовсе. Что это за регистры?

R6, или (как его обозначают на языке ассемблера) SP — Stack Pointer, УКАЗАТЕЛЬ СТЕКА. Мы уже говорили, что СТЕК — это специально выделенная область памяти. R6, или SP, всегда содержит адрес ее начала, или, как говорят, ВЕРШИНЫ СТЕКА. Пользуясь этим регистром, удобно записывать в стек или извлекать из него информацию, причем адрес вершины при этом автоматически может меняться, все время указывая на ту ячейку памяти, с которой в данный момент мы работаем. Но главное назначение стека — не столько обслуживать нужды программиста, хотя и это немаловажно, сколько временно сохранять данные, необходимые самой ЭВМ, например при переходе к подпрограмме или при обработке ПРЕРЫВАНИЯ.

R7, или (на языке ассемблера) PC — Program Counter, ПРОГРАММНЫЙ СЧЕТЧИК, или счетчик команд. Мы уже говорили, что он всегда указывает адрес ОЧЕРЕДНОЙ КОМАНДЫ и тем самым позволяет процессору исполнять программу в правильной последовательности, а меняя его содержимое, можно изменять естественный порядок исполнения команд программы. По сути, смысл его тот же, что и у SP, — он указывает определенный адрес памяти, но в отличие от стека это не память данных, а память программ.

Есть в нашей ЭВМ и еще один регистр, уже совсем специфический, он носит сокращенное название PS, от слов Processor Status (или PSW, PSWord). Иногда его называют также СЛОВОМ СОСТОЯНИЯ ПРОЦЕССОРА (ССП). Важность этого регистра трудно переоценить — только с его помощью ЭВМ может выполнять, например, такие необходимые действия, как проверка условий и услов-

ные переходы, а также делать многое другое, влияющее на вычислительный процесс. Поэтому рассмотрим его подробнее.

В БК-0010 используется только младший байт PS. Все его разряды доступны по записи и чтению, хотя запись в некоторые из них возможна только с помощью специальных команд или приемов, а чтение может происходить вовсе не обязательно в том виде, к которому мы привыкли: ЦП сам, с помощью специальных команд читает эти разряды и использует их.

Что же это за разряды? Вначале идут четыре «флага», или разряда УСЛОВИЙ. Они принимают то или иное значение в зависимости от результата исполнения процессором очередной команды. Исходное состояние этих разрядов, если указанные ниже условия НЕ ВЫПОЛНЯЮТСЯ, — нули:

- 00 (разряд C) — ПЕРЕНОС. Устанавливается в 1, если в результате исполнения команды возник перенос за пределы слова или байта. Тут, очевидно, надо оговориться, что возможен перенос только единицы, ноль на результат не влияет. Правильнее было бы сказать, что C-разряд ПРИНИМАЕТ ЗНАЧЕНИЕ РАЗРЯДА ПЕРЕНОСА;
- 01 (разряд V) — ПЕРЕПОЛНЕНИЕ. Устанавливается в 1, если в результате исполнения команды имело место переполнение, т.е. перенос в знаковый разряд слова или байта;
- 02 (разряд Z) — НОЛЬ. Устанавливается в 1, если результатом исполнения команды является нулевое содержимое слова или байта;
- 03 (разряд N) — ОТРИЦАТЕЛЬНОСТЬ. Устанавливается в 1, если результатом исполнения команды является отрицательное число в слове или в байте.

Используя содержимое этих разрядов, можно организовать так называемое ВЕТВЛЕНИЕ программы, т.е. переходы по условию, или, что то же самое, по результатам исполнения команды. Помимо того что эти разряды устанавливаются в 0 или 1 автоматически по результатам исполнения каждой команды, существуют и специальные операторы для их «ручной» установки.

Следующий разряд — 04, или T-РАЗРЯД. Он имеет особое назначение. Если содержимое T-разряда равно 1, то программа приостанавливается каждый раз после выполнения очередной команды (происходит так называемое ПРЕРЫВАНИЕ ПО T-РАЗРЯДУ). Этот режим процессора широко используется в специальных программах-отладчиках, позволяющих исполнять программы по одной команде с целью их отладки (выполнять ТРАССИРОВКУ). Просто так записать в T-разряд единицу нельзя, для этого существует специальный прием, с которым мы познакомимся позже.

И наконец, последние три разряда (05, 06, 07) — это так называемые разряды ПРИОРИТЕТА ПРОЦЕССОРА. Что это такое? В процессе работы ЭВМ ЦП постоянно обрабатывает информацию, или, проще говоря, решает задачи. Но ЭВМ работает и с внешними устройствами, например с клавиатурой. Можно организовать работу с клавиатурой двояко. По первому способу ЦП после выполнения определенного числа команд прерывает решение задачи и обращается к клавиатуре, опрашивая ее. Если ни одна клавиша не нажата, вычисления будут продолжены, затем снова следует опрос клавиатуры и т.д. Этот способ довольно нерационален — независимо от того, нажата ли клавиша, ЦП вынужден все время «отвлекаться» и тратить время на опрос клавиатуры. А если учесть, что на нажатие клавиши ЭВМ должна реагировать мгновенно (чтобы не создавать неудобств в работе), то понятно, что все клавиши должны опрашиваться никак не реже, чем 100 или даже 1000 раз в секунду.

Второй способ куда лучше — пусть ЦП решает задачу, а если будет нажата клавиша, то он получит так называемый ЗАПРОС НА ПРЕРЫВАНИЕ. Это специальный сигнал, сообщающий процессору, что с какого-то из внешних устройств поступила порция информации и оно ждет, чтобы ЦП отреагировал. ЦП прервет решение задачи, ОБРАБОТАЕТ ПРЕРЫВАНИЕ и продолжит вычисления.

Но постойте! Есть ведь такие задачи, когда ЦП никак не может «отвлечься», например работа с магнитофоном. Лента ведь не будет стоять и ждать, пока ЦП обработает прерывание... Вот для этого и служит приоритет. Если приоритет, установленный на данный момент для ЦП (как говорят, ПРИОРИТЕТ ТЕКУЩЕЙ ЗАДАЧИ), ниже, чем установленный ПРИОРИТЕТ ЗАПРОСА НА ПРЕРЫВАНИЕ, то ЦП прервет работу и обработает прерывание. Если же приоритет текущей задачи равен или выше приоритета запроса, то ЦП сначала закончит вычисления, а уже потом обработает прерывание. Сразу поясним, что приоритет запросов на прерывание от внешних устройств (у БК-0010 их, по сути, может быть всего два — клавиатура и внешний таймер) установлен для данной конфигурации ЭВМ раз и навсегда. А вот приоритет ЦП может меняться и определяется содержимым указанных трех разрядов PS.

Таблица 2

Источник прерывания	Приоритет	Адрес вектора	Использование источника
Зависание	1	4	
Ошибочный машинный код	2	10	
Т-бит	3	14	
Сбой питания	4	24	
Радиальное стат. (клавиша «СТОП»)	5	4	Прерывание программы пользователя с пульта оператора
Радиальное дин.	7	100	Прерывание по требованию устройства

Сколько может быть задано различных уровней приоритета процессора? Очевидно, восемь, ведь в три двоичных разряда можно записать восемь различных комбинаций. Минимальный приоритет, когда прервать работу процессора может любое устройство, — 000, максимальный, когда работа процессора непрерываема ничем, кроме клавиши «СТОП» (это — «внеприоритетное» внешнее устройство), — 111.

Для справки приведем таблицу, содержащую сведения о прерываниях, их приоритетах и использовании. Эта таблица взята из внутризаводской «ремонтной документации» по БК-0010 — см. табл. 2.

Не удивляйтесь, что прерывания по зависанию и по клавише «СТОП» имеют один и тот же вектор, но разные приоритеты. «Ремонтная документация» объясняет, в чем дело. При нажатии на клавишу «СТОП» возникает радиальное прерывание IRQ1. Микропрограммная обработка этого прерывания содержит обращение по записи к регистру с адресом @#177676. Однако в данной модификации микроЭВМ ячейка с таким адресом отсутствует, поэтому происходит обработка прерывания по зависанию процессора. Попутно отметим, что кроме зависания и клавиши «СТОП» прерывание по вектору 4 обслуживает также и отработку команды процессора HALT.

Кстати говоря, хотя вектор прерывания по «СТОП» и по зависанию один и тот же, существует простой способ программного определения причины прерывания. Для этого служит разряд 02 регистра @#177716, выставляемый в 1 при нажатии на клавишу «СТОП» (или при отработке HALT):

```
PRST:  . . .    ; п/пр обработки прерывания
       . . .    ; по вектору @#4
       BIT #4, @#177716
                ; проверка разряда 02
       BEQ ZAVIS
                ; на обработку зависания
       . . .    ; обработка нажатия
       . . .    ; на клавишу «СТОП»
```

Этот способ не описан нигде, кроме все той же «ремонтной документации». — Прим. ред.

Векторы прерываний

Мы уже не раз упоминали термин «прерывание». Пора, наконец, разобраться, что же это такое. Для начала попробуем дать общее определение. ПРЕРЫВАНИЕ — ЭТО ПРИОСТАНОВКА ВЫПОЛНЕНИЯ ПРОЦЕССОРОМ РЕШЕНИЯ ТЕКУЩЕЙ ЗАДАЧИ И ПЕРЕХОД К ОБСЛУЖИВАНИЮ ВНЕШНИХ УСТРОЙСТВ ИЛИ ПРОГРАММНЫХ ЗАПРОСОВ. Как происходит так называемая ОБРАБОТКА ПРЕРЫВАНИЯ? Вначале рассмотрим обработку ПРЕРЫВАНИЙ ОТ ВНЕШНИХ УСТРОЙСТВ.

Пусть ЦП выполняет операции по решению текущей задачи, а в это время поступает ЗАПРОС НА ПРЕРЫВАНИЕ от внешнего устройства. Этот запрос обычно подается по одной из отдельных электрических цепей ЦП, следовательно, он не является командой в обычном понимании. Если приоритет ЦП, установленный для текущей задачи, меньше, чем приоритет внешнего устройства, от которого пришел запрос, происходит ПРЕРЫВАНИЕ. При этом процессор завершает выполнение очередной команды и прекращает ввод следующих команд, после чего он записывает

в стек текущее ССП (содержимое PS), а затем содержимое PC, т.е. не что иное, как адрес следующей команды. Эти данные необходимо сохранить, ведь после окончания обработки прерывания процессор должен продолжить решение текущей задачи начиная со следующей команды и с тем же самым приоритетом. Часто требуется сохранить не только содержимое PC и PS, но и другие данные, тогда об этом должен позаботиться уже программист. Записав ССП и PC в стек, ЦП выполняет так называемый ПЕРЕХОД ПО ВЕКТОРУ ПРЕРЫВАНИЯ.

ВЕКТОР ПРЕРЫВАНИЯ — это просто два машинных слова, записанных одно за другим в специально выделенной зоне ОЗУ — СИСТЕМНОЙ ОБЛАСТИ. Первое из этих слов представляет собой АДРЕС начала программы обработки данного прерывания, второе — ССП, установленное для него. Из этого описания совершенно ясно, что каждое прерывание должно иметь свой собственный вектор. Итак, процессор, выполняя переход по вектору прерывания, переписывает его первое слово в PC, а второе — в PS, после чего, естественно, следующей выполняется команда, записанная в памяти по адресу, указанному в PC, т.е. начинается выполнение программы обработки прерывания. Приоритет ЦП при этом будет установлен согласно новому содержимому PS, которое, как мы помним, переписано из второго слова вектора. Если этот приоритет достаточно низкий, т.е. допускает на фоне обрабатываемого новое прерывание, то оно может быть также обработано и т.д. Те, кто занимался практическим программированием, не могут не заметить в этом процессе некоей аналогии с обращением к подпрограмме. Правда, обработка прерывания несколько осложняется изменениями ССП, а следовательно, приоритета и битов («флагов») условий. Но раз уж мы усмотрели эту аналогию, то вспомним, что из любой подпрограммы должен быть возврат к выполнению основной программы. Точно так же существует и ВОЗВРАТ ИЗ ПРЕРЫВАНИЯ, для чего в конце программы его обработки должна быть записана специальная команда. Обнаружив ее, ЦП выполняет действия, обратные только что описанным, — восстанавливает из стека содержимое PC и PS и, естественно, продолжает выполнение прерванной программы, ведь теперь в PC — адрес ее следующей команды. Отметим еще раз, что при этом приоритет процессора (а что еще важнее, «флаги» условий CVZN и, конечно, Т-разряд) также восстанавливается.

Помимо прерываний от внешних устройств существуют и так называемые КОМАНДНЫЕ ПРЕРЫВАНИЯ, или программные запросы. Их обработка отличается от только что описанного процесса лишь тем, что прерывание вызывается особой командой, встретившейся по ходу программы. Ясно, что программный запрос на прерывание удовлетворяется всегда и не зависит от приоритета — не может же процессор игнорировать команду, которую сам прочитал! Командное прерывание, таким образом, еще больше похоже на переход к подпрограмме.

Векторы прерывания располагаются, как было указано выше, в специально отведенной зоне ОЗУ. В БК-0010 эта зона занимает адреса 0...276. Вычисления показывают, что в данной зоне можно разместить до 48 векторов прерывания. Но, разумеется, такого количества векторов в БК нет, а свободные ячейки памяти используются для других целей.

Откуда же берутся в системной области ОЗУ векторы прерывания? Очень просто: они переписываются туда из ПЗУ при запуске системы. А ЦП уже «знает», по какому адресу записан каждый вектор, эти сведения заложены в него аппаратно, и при поступлении соответствующего запроса он ими пользуется.

Перечислим кратко имеющиеся в БК-0010 векторы прерывания. Вектор прерывания обычно получает «имя» от того адреса, по которому записано его первое слово. Эти адреса строго фиксированы для данного типа процессора и поэтому точно определяют вектор.

- ВЕКТОР 4 — ПРЕРЫВАНИЕ ПО ЗАВИСАНИЮ. Переход по этому вектору выполняется как по запросу внешних устройств, так и программно. «Внешним устройством» здесь является клавиша «СТОП», а программный переход по вектору 4 вызывает команда HALT — «останов» (код этой команды — 0). Кроме того, прерывание по этому вектору может вызвать и сам ЦП при зависании (что это такое, уже описано выше), откуда и следует его название. Клавиша «СТОП», в отличие от других внешних устройств, вызывает прерывание всегда (понятие приоритета на нее не распространяется).
- ВЕКТОР 10 — ПРЕРЫВАНИЕ ПО РЕЗЕРВНОМУ КОДУ. Как уже было сказано, не все возможные коды использованы в наборе команд ЦП. Если в процессе выполнения программы встретится «неизвестный» код, происходит прерывание по вектору 10.
- ВЕКТОР 14 — ПРЕРЫВАНИЕ ПО Т-РАЗРЯДУ. Прерывание по данному вектору происходит после выполнения очередной

команды, если Т-разряд ССП установлен равным 1.

- ВЕКТОР 20 — ПРЕРЫВАНИЕ ПО КОМАНДЕ IOT. Иногда используется в операционных системах.
- ВЕКТОР 24 — ПРЕРЫВАНИЕ ПО АВАРИИ СЕТЕВОГО ПИТАНИЯ. В БК данный вектор не используется, хотя программа обработки в ПЗУ имеется. Вектор просто не задействован аппаратно — соответствующий вход процессора (ACLO) не подключен к устройству, вызывающему прерывание при снижении напряжения питания. Это устройство в составе БК попросту отсутствует.
- ВЕКТОР 30 — КОМАНДНОЕ ПРЕРЫВАНИЕ EMT. Этот очень широко используемый вид прерывания, как и следующий (по вектору 34), будет подробно описан в дальнейшем.
- ВЕКТОР 34 — КОМАНДНОЕ ПРЕРЫВАНИЕ TRAP.
- ВЕКТОР 60 — ПРЕРЫВАНИЕ ОТ КЛАВИАТУРЫ. Запрос на прерывание по этому вектору поступает при нажатии любой клавиши (кроме «СТОП» и регистровых). Запрос удовлетворяется, если приоритет процессора менее 4. Если приоритет равен или больше 4, запрос на прерывание «зависает» до окончания обработки процессором текущей задачи или до снижения приоритета, в этом случае говорят, что прерывание от клавиатуры запрещено.
- ВЕКТОР 100 — ПРЕРЫВАНИЕ ПО ТАЙМЕРУ. Помимо внутреннего (системного) таймера, описанного выше, к БК-0010 можно подключить и внешний, или ТАЙМЕР ПО ПРЕРЫВАНИЯМ. Его преимущество в том, что ЭВМ, работая с ним, не должна обязательно периодически опрашивать регистр счетчика, а таймер сам, например один раз в секунду, вызывает прерывание. Приоритет таймера такой же, как у клавиатуры, и прерывание по вектору 100 разрешено, если приоритет процессора ниже 4. Программа обработки данного прерывания предусмотрена в ФОКАЛе, содержимое счетчика таймера вызывается функцией FCLK(), которая, кстати, не описана в руководстве по ФОКАЛу.

(Другое название этого прерывания — «Прерывание пользователя», так как фактически оно может быть вызвано любым пользовательским внешним устройством, подающим сигнал на контакт В1 разъема порта УВВ (справа) или А5 разъема системной шины (слева). Таким образом, при описанном выше использовании прерывания таймер является примером внешнего устройства. Простейший способ вызова прерывания по вектору 100 — подача на контакт В1 порта УВВ сигнала «земля» (или «общий») с контактов А11, В11, А18, В18, А19 или В19 (так как порт УВВ работает с инверсными сигналами). Наиболее часто данное прерывание используется для печати копии экрана на принтер при нажатии замыкающей указанные контакты кнопки, как, например, в программе GRAFIX фирмы ALTEC или резидентном драйвере SCREW ANIRAM-MIRIADA. — *Прим. ред.*)

- ВЕКТОР 274 — ПРЕРЫВАНИЕ ОТ КЛАВИАТУРЫ ПО НИЖНЕМУ РЕГИСТРУ. Как уже было сказано, в регистре данных клавиатуры (адрес 177662) при нажатии клавиши формируется 7-разрядный код. Его преобразование в код нижнего регистра осуществляется благодаря тому, что для обработки нажатия таких клавиш предусмотрен отдельный вектор прерывания. В остальном вектор 274 не имеет отличий от вектора 60, его приоритет тот же самый.

Остальные ячейки в области векторов прерывания БК-0010 используются для других целей. Но нам вполне достаточно и имеющихся векторов. Позже мы рассмотрим подробнее, что и как можно сделать с их помощью, а сейчас перейдем ко второй теме нашего разговора — к языку ассемблера.

Контрольные вопросы и задания

1. Сколько регистров имеется в составе ЦП? Каковы их названия?

— Восемь регистров общего назначения: R0, R1, R2, R3, R4, R5, R6 (или SP) и R7 (или PC), а также специальный регистр PS.

2. Какие «флаги» состояния вы знаете, их буквенные обозначения и краткие наименования?

— C — перенос, V — переполнение, Z — ноль, N — отрицательность.

3. Если в результате выполнения какой-либо команды произошло переполнение и число стало отрицательным, то какое значение примут разряды условий?

— C=0; V=1; Z=0; N=1.

4. Что обеспечивает Т-разряд ССП при его установке в единицу? Зачем это нужно?

— Прерывание программы после выполнения каждой команды, это нужно для обеспечения отладочного режима (трассировки).

5. Что хранится в первом слове вектора прерывания? А во втором?

— Адрес программы обработки прерывания, ССП для данного прерывания.

6. Чем отличаются командные прерывания от прерываний, вызываемых внешними устройствами?

— Командное прерывание вызывается особой командой программы, понятие приоритета на него не распространяется.

7. Учитывая, что разряды приоритета в ССП имеют номера 05, 06 и 07, напишите, какое машинное слово надо записать в регистр PS, чтобы получить приоритет 0, приоритет 4, максимальный приоритет. Считайте при этом, что остальные разряды PS равны нулю. Для облегчения задачи «нарисуйте» и пронумеруйте разряды PS, а получившееся потом число переведите в восьмеричное.

— Приоритет 0 — 000000, приоритет 4 — 000200, максимальный приоритет (7) — 000340.

8. Какой приоритет имеет клавиша «СТОП» БК-0010?

— Это внеприоритетное внешнее устройство.

Языки программирования. Что такое ассемблер

Вы, конечно, знакомы с имеющимися на БК-0010 так называемыми ЯЗЫКАМИ ВЫСОКОГО УРОВНЯ. Это ФОКАЛ и БЕЙСИК-MSX, которые находятся в ПЗУ (такое программное обеспечение, зашитое в ПЗУ, называется РЕЗИДЕНТНЫМ). В составе ПО БК-0010 имеются и другие ЯЗЫКИ ПРОГРАММИРОВАНИЯ, но, в отличие от резидентных, они загружаются в ОЗУ. Все эти языки (вильнюсские БЕЙСИК-85 и БЕЙСИК-87, Т-язык, ФОРТ-83 и т.п.) имеют общую черту — это языки-ИНТЕРПРЕТАТОРЫ. Что это значит?

Программа на языке интерпретирующего типа пишется в виде текста (ЛИСТИНГА), в котором с помощью условных символов, цифр и слов указывается последовательность действий ЭВМ. (Отметим, что под термином «листинг» понимается иногда вообще произвольный список или просто распечатка любых данных. Мы же будем для краткости применять этот термин в смысле «текст программы в условных обозначениях языка программирования».) Но нам известно, что единственный язык, доступный ЭВМ, — это язык машинных команд, двоично-восьмеричный код. Написанная на языке высокого уровня программа может быть исполнена не непосредственно, а лишь с помощью программы-интерпретатора (входящей в состав языка), причем листинг построчно переводится интерпретатором в последовательность машинных команд, которую и исполняет ЭВМ. Программа хранится в памяти ЭВМ практически в том виде, в котором написана (небольшая модификация листинга, происходящая при этом, принципиального значения не имеет). Если нужно еще раз исполнить программу, интерпретатор снова переводит ее в машинные команды, причем не всю сразу, а по частям (обычно по одному оператору). Преимущества интерпретатора — простота языка и удобство программирования (программу можно просмотреть в той же форме, в которой она написана, исправить, дополнить и т.п.), а также компактность представления программ в ОЗУ в виде листинга. Недостатки — низкая скорость исполнения программ (ведь интерпретатор выполняет перевод в машинные коды каждый раз заново) и то, что программа-интерпретатор всегда должна присутствовать в памяти. (Прошитый в ПЗУ БК-0010.01 вильнюсский БЕЙСИК-MSX является своего рода промежуточным звеном между интерпретаторами и компиляторами. Как и последние, он непосредственно перед выполнением программы транслирует ее в последовательность «машинных псевдокоманд» (шитых кодов). Однако, как и в интерпретаторе, отсутствует возможность получения автономной машинной программы и при исполнении в памяти одновременно должен присутствовать листинг, сгенерированный шитый код и сам транслятор БЕЙСИКа (в ПЗУ). Следует тем не менее заметить, что с помощью определенных хитростей все же удается получать «автономные» программы, не содержащие листинга, но для их работы все равно необходимо наличие ПЗУ с БЕЙСИКом. — *Прим.ред.*)

Помимо языков интерпретирующего типа есть и так называемые ТРАНСЛИРУЮЩИЕ языки, или языки-компиляторы (термины «компиляция» и «трансляция» в данном случае означают одно и то же). Обычно это серьезные языки программирования для мощных компьютеров — ФОРТРАН, ПАСКАЛЬ, КОБОЛ, СИ, ПЛ/1, АДА и др. Программа на таком языке пишется также в виде листинга, а затем ТРАНСЛИРУЕТСЯ, т.е. переводится в машинные команды, но при этом не исполняется. При таком переводе, например, на ФОРТРАНе вместо одного оператора могут быть СГЕНЕРИРОВАНЫ десятки и даже сотни машинных команд. В результате трансляции образуется программа в машинных кодах (так называемый ЗАГРУЗОЧНЫЙ МОДУЛЬ). Поскольку это уже не листинг, а готовый машинный код, программа исполняется несравненно быстрее, чем на языке-интерпретаторе, а дальнейшее присутствие транслятора в памяти становится излишним. К недостаткам языков этого типа относится прежде всего то, что программа-транслятор, входящая в состав языка, при переводе листинга на язык машинных команд строит за-

грузочный модуль не всегда оптимально. В зависимости от конкретного типа транслятора, вида языка и реализуемого алгоритма эта «неоптимальность» может приводить к увеличению длины программ в 1.5—3 раза и соответствующему снижению возможного быстродействия ЭВМ. Происходит это потому, что язык высокого уровня по структуре весьма сложен, и при трансляции листинга всегда присутствует некоторая неоднозначность, когда можно сделать перевод «и так, и эдак»

(Более серьезной причиной резкого увеличения объема сгенерированной программы является то, что к ней пристыковывается так называемый библиотечный модуль, содержащий ВСЕ подпрограммы реализации процедур и функций, относящихся к какому-либо классу (например, все операции ввода-вывода). Подобное «техническое решение» обеспечивает унификацию и упрощает работу компилятора, но реально большая часть пристыкованных в составе данного модуля подпрограмм в конкретной программе не используется и является своего рода «балластом». В интерпретаторе же все подпрограммы реализации, как правило, зашиты в самом интерпретирующем модуле, что, в свою очередь, порождает другие недостатки, например невозможность расширения перечня используемых функций, тогда как в компиляторе это можно сделать простым включением новых подпрограмм в состав старого библиотечного модуля. — *Прим. ред.*)

Кроме того, обратный перевод с машинных кодов на язык высокого уровня практически невозможен. Очень сильно на оптимальность трансляции влияет и то, что программы на языках высокого уровня должны быть ПЕРЕНОСИМЫ НА УРОВНЕ ЛИСТИНГОВ, т.е. должны исполняться по возможности одинаково на машинах разных систем. Получается, что стандарт языка единый, а под него должны «приспосабливаться» разные машины. ЭВМ же, как правило, свойственны жесткая архитектура и система команд, не всегда соответствующие средствам языка. Как разрешить это противоречие и как, с одной стороны, использовать все ресурсы ЭВМ, а с другой, сделать программу оптимальной, а средства ее написания — удобными для программиста?

На заре компьютерной техники программы составлялись непосредственно в виде последовательности машинных команд (в «машинном коде»), при этом они, естественно, полностью соответствовали архитектуре ЭВМ и были оптимальны как по длине, так и по быстродействию (конечно, при известной квалификации программиста). Но программирование в машинных кодах — каторжный труд, превращающий человека в придаток машины, большая часть ресурсов человеческого мозга при этом уходит не на собственно программирование, а на запоминание, вычисление и прочие операции, связанные с кодами. Вскоре программисты сообразили, что эти функции можно передать ЭВМ, освободившись тем самым от рутинного труда, а мощность машин стала достаточной, чтобы помочь в этом программисту. Так из машинного языка родился ЯЗЫК АССЕМБЛЕРА. Как и язык высокого уровня, он позволяет пользоваться системой обозначений, удобной и (по сравнению с машинными кодами) легко запоминающейся, но, в отличие от других языков, это ЯЗЫК НИЗКОГО УРОВНЯ, или МАШИННО-ОРИЕНТИРОВАННЫЙ язык. Язык ассемблера всегда приспособлен для машин определенного типа, рассчитан на определенную систему команд. Поэтому он позволяет использовать все, без исключения, ресурсы машины, достигать максимально возможной компактности и быстродействия программ. Каждой команде на языке ассемблера соответствует своя команда в машинных кодах, поэтому возможен и нетруден как прямой (с ассемблера в коды), так и обратный перевод. Структура языка проста, и программа-транслятор получается достаточно компактной. Все эти преимущества обусловили чрезвычайно широкое распространение языков ассемблера (на каждой ЭВМ он свой) и их высокую популярность. Тем более велика роль ассемблера на таких ЭВМ, как БК-0010 с ее далеко не безграничными возможностями как по быстродействию, так и, что особенно важно, по ресурсам памяти.

Итак, что же такое АССЕМБЛЕР? Мы уже знаем, что это машинно-ориентированный язык, система условных обозначений для программирования в машинных кодах. Кроме того, словом «АССЕМБЛЕР» обозначают и специальную программу, обеспечивающую программирование на языке ассемблера, иногда при этом добавляя: «ассемблер-система». И еще одно значение придают этому термину, когда говорят «программа на ассемблере», — под этим обычно понимается программа в машинных кодах, написанная на языке ассемблера и оттранслированная в коды.

А что же машинные коды, их уже знать не требуется? В общем, да. Ассемблер-система обеспечивает перевод листинга программы в машинный код, а другая специальная программа, называемая ДИЗАССЕМБЛЕРОМ, может легко выполнить обратный пере-

вод машинных кодов на язык ассемблера, причем оба преобразования делаются быстро и без ошибок. (На самом деле программы-дизассемблеры не всегда работают без ошибок. Многие из этих программ, реализованных на БК, не распознают входящие в «тело» кодового модуля массивы чисел и символов, пытаясь и их интерпретировать как команды ассемблера. Это может сделать полученный листинг трудночитаемым, особенно для начинающих программистов. — *Прим. ред.*) И тем не менее бывают случаи, когда знание машинных кодов может оказаться весьма полезным, избавляя от необходимости применять специальные программные средства и позволяя экономить время. Но это случается относительно редко, поэтому изучение машинных команд не входит в нашу задачу, а желающие всегда могут ознакомиться с ними по имеющейся литературе. В частности, список машинных команд в кодах и их соответствий на языке ассемблера (так называемый МНЕМОКОД) приведен в «Руководстве системного программиста», прилагаемом к БК-0010. (Более подробные сведения можно найти в кн.: *Осетинский Л.Г., Осетинский М.Г., Писаревский А.Н.* ФОКАЛ для микро- и мини-компьютеров. Л.: Машиностроение, 1988. — *Прим. ред.*) Мы же по ходу изложения будем обращаться к машинным кодам только в редких случаях, по возможности ограничиваясь языком ассемблера.

Ассемблер-системы

Для БК-0010 создано множество ассемблер-систем. Не останавливаясь на ранних, уже устаревших программах, отметим, что все современные ассемблер-системы имеют ряд общих черт. Все они состоят из четырех основных частей, объединенных в одном модуле: МОНИТОР, РЕДАКТОР, ТРАНСЛЯТОР и КОМПОНОВЩИК.

МОНИТОР включается при запуске ассемблер-системы и обеспечивает с помощью ряда директив диалог с пользователем, позволяя считывать тексты программ и записывать их на МЛ, транслировать и компоновать программы, а также (в наиболее совершенных системах) запускать готовый загрузочный модуль.

По особой директиве можно перейти в РЕДАКТОР ТЕКСТА ассемблер-системы. Это, по-видимому, самая важная ее часть, от совершенства которой в большой степени зависит удобство работы в ассемблере. Редактор позволяет вводить с клавиатуры и редактировать тексты (листинги) программ. Мощный редактор многократно ускоряет и облегчает работу в ассемблере, и недооценивать его значение может только тот, кто по-настоящему не работал в ассемблер-системе.

ТРАНСЛЯТОР запускается директивами монитора. Он транслирует (переводит в машинные команды) текст программы, подготовленный в редакторе или загруженный с МЛ, резервирует ячейки памяти под переменные, константы и массивы, вычисляет адреса строк (операторов) программы и адреса переходов, а также выявляет некоторые ошибки, допущенные программистом.

КОМПОНОВЩИК, также работающий под управлением монитора, позволяет так обработать текст уже отранслированной программы, что она становится приспособленной для работы в заданных конкретных адресах ОЗУ. Он также может состыковать несколько модулей, загруженных с магнитной ленты, между собой для их совместной работы как единого целого.

Какие же в настоящее время разработаны ассемблер-системы для БК-0010? Первыми системами, имеющими все вышеперечисленные признаки, были трансляторы МИКРО.С (авторы А.М. Сомов и С.В. Шмытов, Москва). Начиная с МИКРО.3С и кончая МИКРО.1103 и МИКРО.1104 (1989—1990 гг.), эти ассемблер-системы непрерывно совершенствовались, расширялась их система команд, улучшался сервис. Однако основной их недостаток — довольно слабый и примитивный редактор текста — сохранился до самой последней версии.

Заметный шаг вперед в области создания ассемблер-систем для БК-0010 был сделан, когда к работе подключился С.А. Кумандин (г.Коломна). Начав с усовершенствования существовавшей в тот момент версии (МИКРО.8С) и снабдив ее значительно более мощным редактором, что привело к появлению МИКРО.8К, он продолжал в дальнейшем работу над ассемблер-системой, все дальше уходя от исходной версии. Результатом этого явилось создание в 1988—1989 гг. версий МИКРО.10К и МИКРО.11К, которые можно считать уже вполне самостоятельными ассемблер-системами, так далеко они ушли от МИКРО.8С. Если версия МИКРО.8К содержала ряд ошибок, то МИКРО.10К является совершенной и тщательно отлаженной программой, имеет расширенную систему команд, удобные дополнительные форматы записи псевдокоманд, а самое главное — редактор текста, который не уступает, а во многом и превосходит знаменитый редактор EDASP. Не останавливаясь подробно на достоинствах

этого редактора, можно только сказать, что, по мнению автора, сегодня это вообще лучший экранный редактор текста, созданный когда-либо для БК-0010, включая и специальные редакторы, — он имеет все необходимое, в том числе механизм макроопераций, неограниченный буфер текста и т.п.

На базе МИКРО.10К создано еще несколько версий ассемблера — МИКРО.11К, рассчитанный специально на прошивку в ПЗУ и имеющий еще более совершенный редактор, МИКРО.10К-РП, редактор которого может работать как с обычным экраном, так и в режиме «РП», что позволяет при необходимости иметь дело с текстами длиной до 50000 символов, МИКРО.10-01К, специально рассчитанный на работу с клавиатурой БК-0010.01 и также имеющий режим «РП». Следует отметить, что все версии МИКРО.10К (даже те, редактор в которых на работу в «РП» не рассчитан) позволяют в режиме «РП» компоновать в памяти тексты программ большой длины (до 50000), что в основном позволяет обойтись без промежуточных записей на МЛ объектных модулей при компоновке и значительно повышает оперативность и удобство работы.

К редактору ассемблер-системы МИКРО.10К в настоящее время изготовлен, прежде всего автором этой публикации, целый ряд «приставок»-утилит, еще более расширяющих его возможности. Эти утилиты позволяют форматировать тексты и печатать их на принтерах, выполнять произвольную перекодировку, сортировку по алфавиту и т.п. Причем все это делается непосредственно при работе в редакторе, без промежуточной записи текста на МЛ. Все ассемблер-системы МИКРО.К, в отличие от прочих, являются перемещаемыми, могут с равным успехом работать в ОЗУ и ПЗУ, а параметры их буферов (текста и загрузочного модуля) могут быть легко и произвольно изменены пользователем, например с целью использования всей памяти при работе системы в ПЗУ.

В последнее время появился ряд новых, достаточно удобных и мощных разработок ассемблер-систем — MIKRO.SW (В.С. Коренков), пакет MSW (В.В. Савин) и другие, но все они, имея перед версией МИКРО.10К-РП некоторые (не слишком существенные) преимущества, лишены ряда ее достоинств, и прежде всего присущей ей простоты и «открытости», т.е. хоть чем-нибудь да связывают программиста. Поэтому начинать с них знакомство с ассемблером все же не рекомендуется. (Кроме вышеперечисленных сейчас существует и ряд других, не менее удобных ассемблер-систем: М18 (автор А.Г. Прудковский) и ее новая версия М19, ассемблер Турбо2 (А. М. Надежин), имеющий в своем составе модуль плавающей арифметики, и наиболее мощный на сегодня ассемблер-отладчик-дизассемблер PARADISE С.В. Клименкова. — *Прим.ред.*)

В дальнейшем мы будем при описании ассемблера ориентироваться в основном на версию МИКРО.10К-РП, но при необходимости отмечать и особенности других версий. Подробно описывать какую-либо, а тем более все версии ассемблер-систем, созданных для БК-0010, не представляется возможным. Устанавливая конкретную программу (рекомендуется приобрести ее официально, а не украсть с помощью «свободного обмена», ведь это ваш основной рабочий инструмент!), пользователь обычно получает и краткое описание с перечислением команд ассемблера, директив редактора и монитора, а также формата записи операторов и может освоить ее самостоятельно. Мы же будем рассматривать общие вопросы программирования на ассемблере, относящиеся практически ко всем версиям.

Работа с ассемблером. Формат команды

Итак, загрузив и запустив ассемблер-систему МИКРО.10К с адреса 1000, войдем в редактор по команде «EN». Эта команда обеспечивает переход в начало текста, а если нам нужно вернуться на ту страницу, на которой мы были в момент выхода из редактора, подается команда «ED». Теперь мы можем писать текст программы, состоящий из КОМАНД АССЕМБЛЕРА.

В каждой строке текста программы записывается одна команда (исключением являются ПСЕВДОКОМАНДЫ, которых в каждой строке может быть несколько, но об этом позже). КОМАНДНАЯ СТРОКА текста программы условно разделена на четыре ПОЛЯ, расположенных слева направо: МЕТКА, ОПЕРАТОР, ОПЕРАНДЫ и КОММЕНТАРИЙ. Поля в редакторе МИКРО.10К отделяются друг от друга любым числом пробелов (код 40) или символов «ТТ» (код 11). Удобнее всего разделять поля нажатием клавиши «ТТ» (ее эквивалент для клавиатуры БК-0010.01 в редакторе МИКРО.10К — СУ/Т или клавиша «СБР»), при этом одноименные поля размещаются друг под другом, программа имеет аккуратный вид и легко читается. Оканчивается каждая строка символом «ВВОД» («ВК», код 12). Рассмотрим каждое из полей командной строки.

Первое — поле МЕТКИ. Метка — это условное символьное имя, присваиваемое данной строке. После трансляции каждая строка текста превращается в последовательность кодов, размещаемую начиная с определенного (текущего) адреса. Метке, стоящей в начале строки, присваивается этот адрес, который заносится в формируемую транслятором ТАБЛИЦУ МЕТОК. Зачем нужны метки? Как вы помните, в БЕЙСИКе или ФОКАЛе каждой строке присваивается номер, пользуясь которым можно обратиться к данной строке (передать ей управление) из любой точки программы. Точно так же в ассемблере можно обратиться к данной строке (оператору), передав ему управление по имени метки. Но есть и отличия метки от номера строки.

Во-первых, поле метки не является обязательным, т.е. помечаются не все строки, а только те, к которым нам придется обращаться по ходу исполнения программы. Во-вторых, метками могут помечаться не только исполняемые строки программы (содержащие операторы), а вообще любые адреса (ячейки памяти). Таким образом, по метке можно не только передать управление, но и просто обратиться к любому заданному адресу, например записать в помеченную ячейку памяти какое-либо значение или прочитать из нее число. Следовательно, метка — это еще и имя переменной или константы. С другой стороны, как передать управление, так и обратиться к ячейке памяти можно и по АДРЕСУ. Метки же в общем случае служат для присваивания отдельным адресам имен, что значительно облегчает процесс программирования на ассемблере.

Метка может состоять из латинских заглавных букв и цифр, причем допустимое количество символов в имени метки не более трех (в некоторых других версиях ассемблеров — до шести). Если метка начинается с буквы, она называется ОБЫЧНОЙ и обращение к ней возможно из любого места программы с помощью любого оператора, имеющего поле операндов или передающего управление (заметим, что в серьезных ассемблер-системах на больших ЭВМ такие метки носят название ГЛОБАЛЬНЫХ, но автор считает «механический» перенос терминологии на БК-0010 неправильным). Если же метка начинается с цифры, она называется ЛОКАЛЬНОЙ и обращение к ней возможно с помощью лишь некоторых операторов, а действительна она только до ближайшей обычной метки (вперед или назад). Имя обычной метки, как правило, должно быть для данной программы уникальным, т.е. его повторение не практикуется (при повторении имеет смысл только последняя из одноименных меток). Имена локальных меток могут по ходу программы повторяться многократно, лишь бы одноименные локальные метки были разделены обычными (в ранних версиях ассемблеров допускалось употребление только обычных меток). Для отличия имен меток от прочих идентификаторов ассемблера после них ставится символ «:». Имена меток и операторов могут совпадать, это не приводит к ошибкам. Если в строке имеется несколько меток, то всем им присваивается один и тот же адрес. Приведем примеры.

Обычные метки:

A: AB: MET: N9: T75: TR4:

Локальные метки:

2: 94: 545: 7N: 4A7: 8AR:

Запись нескольких меток в одной строке:

1:LD:M1: 1:12: A:17:14:

Второе поле командной строки — поле ОПЕРАТОРА. Оно является обязательным, без него строка лишена смысла. Оператором называется собственно машинная команда, записанная на языке ассемблера. Набор операторов в ассемблере строго соответствует набору машинных команд ЭВМ. Таким образом, оператор сообщает процессору, ЧТО НАДО СДЕЛАТЬ. Подробно каждый из операторов будет рассмотрен в дальнейшем, пока же мы опишем только два из них, которые будем использовать в приводимых примерах:

- CLR — ОЧИСТКА (обнуление) операнда (заданной тем или иным способом ячейки памяти или регистра);
- MOV — ПЕРЕСЫЛКА содержимого первого операнда во второй (т.е. перезапись содержимого одной ячейки памяти во вторую). При этой операции содержимое первой ячейки не меняется, поэтому название «пересылка» следует признать не совсем удачным, правильнее было бы говорить «копирование».

В состав оператора входит также указание, работает ли он со словом или с байтом. Если имя оператора обычное, то это указывает на работу со словом. Для работы с байтом к имени оператора прибавляется буква «B», например: CLR — CLRB, MOV — MOVB. Имена операторов в ассемблере записываются латинскими заглавными символами и, в отличие от некоторых команд БЕЙСИКа, не допускают никаких вариаций и сокращений.

Третье поле командной строки — поле ОПЕРАНДОВ. В зависимости от стоящего в строке оператора операндов может быть два,

один или ни одного (т.е. поле операндов не является обязательным). Если оператор указывает процессору, ЧТО ему надо сделать, то операнды — указание на то, С ЧЕМ ЭТО НАДО СДЕЛАТЬ, т.е. операнды играют роль переменных или констант. Если поле операндов отсутствует, это значит, что выполняются строго фиксированные для данного оператора-действия (например, команда останова процессора). Способы указания операндов весьма разнообразны, они носят название СПОСОБОВ АДРЕСАЦИИ и будут подробно рассмотрены далее.

Последнее поле командной строки — поле КОММЕНТАРИЯ. Как известно, комментариями называются пояснения к программе, применяющиеся с целью сделать ее текст более понятным для чтения. Транслятор игнорирует комментарии, они сохраняются только в листинге программы и в загрузочный модуль (программу в кодах), разумеется, не включаются, поэтому в состав комментариев могут входить любые символы. Комментарии могут следовать после оператора или операндов без всяких разделителей, транслятор распознает их просто по месту расположения в строке. Если же комментарии записаны в пустой строке или сразу после метки (это тоже возможно, тогда данная метка считается относящейся к первой последующей строке с оператором), они должны начинаться с символа «;». В некоторых других версиях ассемблеров отделение комментариев символом «;» обязательно во всех случаях. Приведем пример записи полных командных строк ассемблера и комментариев:

```
; Программа начальной очистки счетчиков
BEG: CLR M1        Очистка счетчика единиц
     CLR M2        Очистка счетчика десятков
     CLR M3        Очистка счетчика сотен
OST: MOVB #1,PRO   Записать признак останова
                   ; в ячейку признака
CON:               ; Продолжение программы
     MOV M1,M4     ; Запись единиц
                   ; и т.д., дальнейший текст
                   ; программы
```

Разумеется, не все еще в этой записи вам понятно, но это станет ясно в дальнейшем. Пока же обратите внимание на расположение полей меток, операторов и операндов и на правила записи комментариев.

* * *

Теперь вы в общих чертах знаете, как писать текст программы. А что с ним делать потом? Разумеется, в свое время об этом будет рассказано подробно. Но по ходу изложения мы будем приводить примеры, которые многие читатели захотят проверить, а может быть, изменить и посмотреть, что получится. Такие эксперименты можно только приветствовать, для освоения ассемблера (как и любого другого языка программирования) нужна практика. Но те, кто столкнулся с ассемблером впервые, наверняка встретят трудности при решении вопроса, что делать с написанным текстом программы. Придется нам, забегая вперед, дать самые общие и схематические рекомендации, не объясняя пока, «что для чего и зачем». Эти рекомендации строго конкретны и относятся только к ассемблер-системе МИКРО.10К и ее модификациям. Если же у вас другая ассемблер-система, самостоятельно найдите в ее описании эквивалентные директивы.

* * *

Итак, текст программы готов. В конце, в отдельной строке, поставьте псевдооператор END. Все, что следует за ним, транслятор игнорирует. Теперь нужно выйти из редактора, нажав клавишу «КТ» (или СУ/Е для МИКРО.10-01К). Вы в мониторе. Если хотите, запишите текст программы на МД директивой «ST». Оттранслируйте листинг директивой «CO», по окончании трансляции будет выдана длина загрузочного модуля в байтах. Если транслятор обнаружит ошибку, то на запрос «E/C?» нажмите клавишу «E», транслятор передаст управление в редактор, и курсор покажет ошибку («ткнет в нее носом»). Частая ошибка, которую на первый взгляд трудно распознать начинающему, — случайное употребление русского символа вместо одинакового по начертанию латинского.

Но вот текст оттранслирован. Определим, нужна ли компоновка. Если все метки (имена и адреса) транслятор выдал в обычной форме, компоновка не требуется. Если же есть метки, выданные инверсно, то она нужна. Для компоновки дайте директиву «LS», последует запрос «Addr=». Введите число 13000 и нажмите «ВВОД». Во вторично выданной таблице инверсных меток быть не должно. Если они есть, значит, в программе имеются ошибки (мы пока даем указания, касающиеся только приводимых далее примеров, при реальном программировании все несколько сложнее). Если все в порядке, программу в кодах (загрузочный модуль) можно записать на МД директивой «SA» и запустить директивой «RU». Посмотрев, как работает программа, остановите ее клавишей «СТОП», ЭВМ выйдет в монитор или МСД. Теперь вы можете с по-

мощью директив МСД просмотреть полученную программу в кодах, она расположена начиная с адреса 13000. При желании можно еще раз запустить программу с этого адреса. Если вы захотите внести в листинг изменения, войдите в ассемблер по адресу «повторного входа» (1002G), выйдите в редактор («ED» или «EN») и модифицируйте текст, а если его нужно уничтожить и начать все сначала, то либо запустите ассемблер с адреса 1000, либо в мониторе ассемблер-системы дайте директиву «RS». Текст с МЛ загружается директивой «LO», а если вы хотите подгрузить его к прежнему тексту в памяти, дайте директиву «LF». Остальные директивы монитора еще долго вам не понадобятся (а если вы не освоите ассемблер всерьез, то вообще никогда).

Вот и все. Как вы считаете, все это настолько сложнее ФОКАЛа или БЕЙСИКа, чтобы стоило об этом говорить? А куда как часто автору приходилось встречать людей, которые заявляли, что на ассемблере ОЧЕНЬ трудно работать, что он УЖАСНО сложен и что освоить его может разве лишь гений... Скромность, конечно, хороша, но не тогда, когда она мешает прогрессу.

Контрольные вопросы и задания

1. Каково основное отличие языка ассемблера от языков ФОКАЛ и БЕЙСИК БК-0010?

— ФОКАЛ и БЕЙСИК — языки интерпретирующего типа, а ассемблер — компилирующего.

2. Какая ассемблер-система самая лучшая?

— МИКРО.10К-РП или МИКРО.10-01К. Это, конечно, шутка. Лучшая ассемблер-система — это та, которая вам нравится, или даже просто та, которая у вас есть. Некоторые же утверждают, что лучшая программа — это та, которая еще не написана.

3. Из скольких символов может состоять метка?

— Не более чем из трех в МИКРО.10К и из шести — в некоторых других системах.

4. Локальная метка может начинаться с буквы?

— Нет, только с цифры.

5. Какие операторы ассемблера вы уже знаете?

— CLR, CLRB, MOV, MOVB, END.

6. Обязательно ли комментарии должны начинаться с символа «;»?

— В МИКРО.10К — не обязательно, если они следуют за оператором или операндами.

7. Наберите в ассемблер-системе программу, приведенную в качестве примера в этом разделе (комментарии можно не переписывать), и попробуйте ее оттранслировать и скомпоновать (запускать не надо). Что обращает на себя внимание после компоновки?

— Метки М1, М2, М3, М4, РКО после компоновки остались инверсными, так как они НЕ ОПРЕДЕЛЕНЫ — отсутствуют в исходном тексте.

8. Является ли программирование на ассемблере сложной процедурой?

— Автор утверждает, что нет, и в ваших интересах не пытаться его опровергнуть.

(Продолжение следует)

Эта публикация представляет собой несколько переработанный вариант внутризаводскнх инструкций по послесборочному контролю и наладке, известных пользователям БК как «ремонтная документация». Следует оговориться, что все приведенные ниже рекомендации относятся к БК-0010 (с ФОКАЛом), для более поздних моделей возможны некоторые отличия. Кроме того, поскольку большинство инструкций рассчитано иа использование специального контрольиого стенда, далеко не все неисправности могут быть выявлены и устранены самостоятельно. Тем не менее, редакция надеется, что эти рекомендацни окажутся полезны БКмаиам.

Ремонт на дому

Поиск и устранение некоторых неисправностей БК-0010

Отыскание неисправностей следует начинать с анализа внешних признаков, различное сочетание которых позволяет с определенной вероятностью установить блок, узел или элемент, подлежащий ремонту. При этом можно рекомендовать следующую последовательность действий:

при включенной системе (микроЭВМ + телевизионный приемник) убедитесь в надежности контактов в соединительных разъемах путем их легкого покачивания с одновременным нажимом на вдвижение внутрь;

после разборки компьютера проведите тщательный внешний осмотр, обращая внимание на любые дефекты монтажа и деталей.

Таблица 1

Признак	Вероятная прнчина	Способ отыскания и устранения
1. Блок вычислнтеля		
1.1 Повышенное потребление в цепи +5В	Короткие замыкания между сигнальными и питающими шинами	Проведите осмотр (прозвонку) монтажа, устраните возможные касания токоведущих элементов, возникшие из-за некачественных паек и изгибов элементов монтажа. Проверьте радиодетали по цепям на пробой
1.2. Пониженное потребление в цепи +5В	Обрывы в разводке шин питания	Осмотрите и прозвоните монтаж, устраните дефект (см. п. 1 и 2)*
1.3. При включении питания отсутствует обнуление памяти экрана (на экране ТВ наблюдаются вертикальные полосы)	Неисправны БИС D8.2, D14, DS17, DS1—DS16 (одна или несколько). Возможны обрывы или короткие замыкания связей	
1.4. При включении питания отсутствует выход на режим диалога	Неисправна БИС DS18. Возможны также обрывы связей	Проверьте и при необходимости замените БИС или восстановите связи
1.5. Нет перехода на тесты контроля	Неисправна БИС DS19. Возможны обрывы связи	Проверьте контрольные суммы. Восстановите связи
1.6. Нет реакции на нажатие клавиш (не проходит тест 2 в режиме ТК)	Обрывы, замыкания цепей платы клавиатуры и кабеля связи, неисправность БИС D4	

* Здесь и далее в таблице указаны пункты разделов «Локализация неисправного узла микроЭВМ» и «Локализация неисправного элемента микроЭВМ».

1.7. Отсутствует изображение, либо изображение есть, но синхронизация сбита	Неисправна схема контроллера ТВ	Выполните действия по п. 4 и 5
1.8. Не проходит тест 1 в режиме ТК	Неисправны одна (несколько) ячеек ОЗУ	Определите неисправную ячейку с помощью теста 1 в режимах ТК и ТС, замените неисправную БИС DS1—DS16
1.9. Не проходит тест 3 в режиме ТК	Неисправны линии программируемого интерфейса	Проведите тест 3 в режиме ТС, локализуйте и устраните неисправность
	Не установлен (неисправен) блок нагрузок	Установите исправный блок нагрузок
1.10. Нет записи информации на ленту или чтения с ленты (не проходит тест 5)	Неисправна схема контроллера магнитофона (КНМЛ) или БИС D14	Выполните действия по п. 26
1.11. Нет обмена информацией в канале связи. (Процессор включен, нарушены временные диаграммы обмена)	Неисправен канал связи, задающий генератор или микросхема D14	
2. Блок питания		
2.1. Блок не включается, светодиод «вкл» не горит, выходные напряжения отсутствуют	Перегорел сетевой предохранитель	Замените предохранитель
	Если предохранитель перегорает повторно — короткое замыкание выходной цепи одного из номиналов напряжения (пробой конденсатора или замыкание). Напряжение на фильтре выпрямителя 2C5, 2C6 присутствует	Прозвоните тестером выходные цепи блока, устраните замыкание, в случае пробоя конденсатора замените его
	Из-за неисправности транзистора 2VT1 или микросхемы 2D2 (отсутствие напряжения на выводе 06) нет напряжения +5В всп	Найдите и устраните неисправный элемент
	Отсутствует напряжение +5В из-за неисправности транзисторов 2VT2, 2VT3 или микросхем 2D1, 2D2	Найдите и устраните неисправный элемент. D2 исправна, если при наличии питающих напряжений на выводах 05 и 10, опорного напряжения на выводе 09 (1.6—2.3 В) и отсутствии запрета на выводе 04 (1.3 В) напряжение на выводах 02 и 03 относительно 08 равно 0.6 В
2.2. Блок не включается, выходные напряжения отсутствуют. Не горит светодиод	Перегорел предохранитель 2А	Замените предохранитель
	Сработала защита по перенапряжению из-за пробоя регулирующих транзисторов	Проверьте и замените поврежденный транзистор

2.3. Выходное напряжение «+5В» не соответствует номинальному значению	Неисправна микросхема 2D2 или транзисторы 2VT3, 2VT4	Проверьте исправность транзисторов и микросхемы. Измерьте падение напряжения на резисторе 2R14 — оно не должно превышать 0.45 В. Установите с помощью переменного резистора 2R18 выходное напряжение равным +5±0.05 В
2.4. Выходное напряжение «+12В» не соответствует номинальному значению	Неисправны транзисторы 2VT5, 2VT6, 2VT8, 2VT9	Проверьте исправность транзисторов. Измерьте падение напряжения на резисторе 2R13 — оно не должно превышать 0.45 В. Установите с помощью переменного резистора 2R30 выходное напряжение +12 В

Примечание. При остановленном процессоре и подаче питания на микроЭВМ от штатного блока питания на экране телевизионного приемника отображается состояние ОЗУ: чередующиеся вертикальные темные и светлые полосы. При включенном процессоре вид информации на экране определяется действиями пользователя.

Ремонт микроЭВМ

Ремонт микроЭВМ заключается в локализации неисправности, замене неисправного элемента и контроле после устранения неисправности.

Локализация неисправного узла микроЭВМ производится с помощью тестов 0—7 в режимах ТК, ТД и ТС (режим ТД может быть использован только при контроле микроЭВМ на стенде).

Порядок запуска тестов и соответствующие сообщения оператору изложены в руководстве оператора, прилагаемом к БК.

Определение неисправного элемента микроЭВМ проводится с помощью тестов и директив отладки режимов ТС и ТД и описано в разделе «Локализация неисправного элемента».

Локализация неисправного узла микроЭВМ

Напряжения на выходных цепях блока питания, используемого при контроле микроЭВМ, должны соответствовать +5±0.05 В и +12 В, сопротивление изоляции между выходными цепями — не менее 20 МОм.

1. Проверка монтажных цепей микроЭВМ. Проведите осмотр монтажа, выявите и устраните возможные касания токоведущих элементов между собой, возникшие из-за их изгибов и некачественных паек.

Измерьте сопротивление в цепи питания +5 В (вольтметр В7—27А в режиме омметра): плюс прибора подключается к точке XT10/03, общая шина — XT10/02, сопротивление должно быть 40—50 Ом. Если измеренное значение меньше приведенного, то проведите визуальный контроль шины питания на короткие замыкания. Если же результат измерения превышает приведенное значение, то проверьте отсутствие обрывов в шинах питания.

2. Подключение питающего напряжения. Включите микроЭВМ в сеть. Проконтролируйте ток потребления. Он не должен превышать 1.1 А.

3. Локализация неисправного узла с помощью тестов. При включении источника питания на экране ТВ контролируемой микроЭВМ должны наблюдаться обнуление экрана и индикация сообщения:

ГОТОВНОСТЬ К РАБОТЕ

*

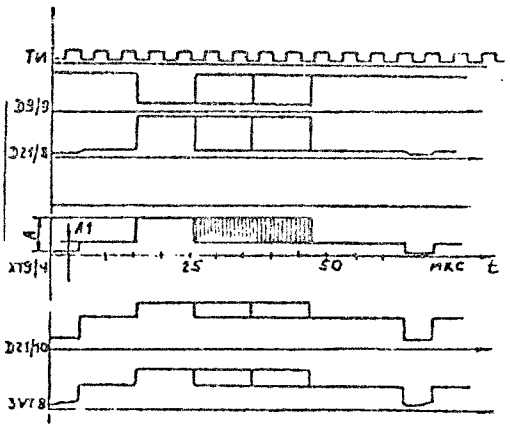


Рис.1

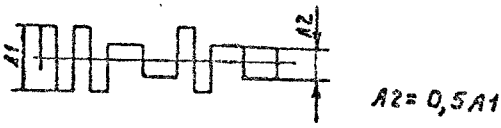


Рис. 2

Это сообщение указывает на работоспособность основных цепей микроЭВМ и на факт запуска процессора (ПРЦ).

Проверьте перезапуск процессора переключателем «СТОП-ПУСК» (SA1). Перейдите к п. 4. Если запуск ПРЦ при включении питания или при перезапуске не состоялся, перейдите к разделу «Локализация неисправного элемента».

4. Переход в режим ТК для запуска теста 1. При отсутствии на экране приглашения «+» выполните п. 17.

5. Запуск теста ОЗУ (тест 1). При сообщении «тест 1 завершен» сделайте заключение о прохождении теста. Если были ошибки, перейдите к п. 18.

6. Запуск теста 2. При наличии ошибок перейдите к п. 21.

7. Запуск теста 3. При ошибках перейдите к п. 22.

8. Запуск теста 4. Проконтролируйте сигнал в точке XT9/4 на соответствие временной диаграмме рис.1 (цена деления на осциллографе 10 мкс на клетку; $A \geq 1$ В, $A1=(0.3 \pm 0.1)*A$). При несоответствии сигнала перейдите к п. 24, при отсутствии прерывания теста — к п. 23.

9. Запуск теста 5. Если в процессе прохождения теста выявлены ошибки, перейдите к п. 26. (Если магнитофон оборудован дистанционным управлением, по окончании теста движение ленты должно быть автоматически остановлено.)

10. Проверка сигнала записи на магнитофон. Отсоедините кабель магнитофона от разъема МГ. Подключите осциллограф к выводу 5 разъема XT4. С помощью директив отладки подготовьте контрольный массив:

```
2000A 4000Д 125252P M3
```

Нажмите клавиши магнитофона «Пуск» и «Запись»

```
адрес=2000 <ввод>
длина=4000 <ввод>
имя=1 <ввод>
```

Проконтролируйте амплитуду сигнала A1, которая должна быть в пределах $1 \pm 0{,}3$ В. Регуляторами «усиление» на развертке с ценой деления 1 установите размах сигнала A1 4 клетки. Проконтролируйте наличие меандра в точках D2/12, XT2/12. Временная диаграмма должна соответствовать рис. 2.

К разъему XT4 подключите RC-цепочку (рис. 3). Проверьте работу усилителя приема данных с линейного выхода магнитофона (повторите директиву M3). Амплитуда сигнала на выводе 3 XT4 должна быть не менее 0.2 В. На выходе усилителя D12/7 проверьте наличие сигнала с амплитудой около 5 В и скважностью $1.8 \leq Q \leq 2.2$. Если амплитуда или скважность не соответствуют указанным значениям, перейдите к п. 26.

11. Проверка работы ПЗУ пользователя. Переставьте ПЗУ с языком ФОКАЛ (из гнезда XT6) в гнездо пользователя (XT8). Перезапустите микроЭВМ. При отсутствии сообщения о готовности к работе верните ПЗУ в гнездо XT6 и перейдите к п. 3.

Локализация неисправного элемента микроЭВМ

12. Контроль генератора тактовых импульсов. Проведите контроль схемы генератора тактовых импульсов на соответствие временной диаграмме (рис. 4).

Проследите цепь прохождения сигнала в последовательности: D5/6, D8/5, D8/6, D8/7, D14/1, D11/8, D9/11, D10/12, D10/11. Устраните неисправность. Повторите контроль схемы и при соответствии сигналов временным диаграммам перейдите к п. 3.

13. Контроль схемы запуска ПРЦ. Проверьте наличие высокого уровня в соответствии с временной диаграммой (рис.5) на выводах D14/29, D14/30 и D14/34 при включении SA1. (Фазы работы ПРЦ: I — начальная установка; II — ожидание снятия сигнала аварии сетевого питания; III — работа ПРЦ по программе.)

При соответствии временной диаграмме выключите SA1 и перейдите к п. 14. При несоответствии проследите цепи D6, D2/15 и

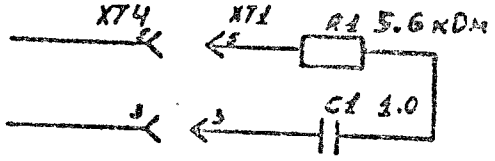


Рис. 3

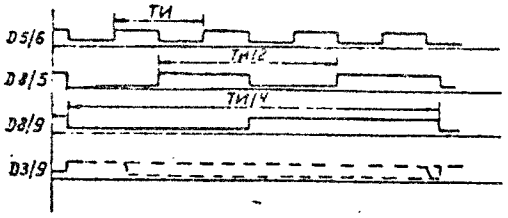


Рис. 4

D11/8. Проверьте соседние линии связи на короткое замыкание. Устраните неисправность. Повторите контроль в точках D14, включите SA1 и перейдите к п. 3.

14. Контроль уровней статических сигналов блока ПРЦ. Проверьте наличие уровня «лог. 1» в точке D11/6 и на контактах D14: 2, 3, 5, 6, 26, 27, 31—35; уровня «лог. 0» в точках D11/1 и D11/9. При соответствии перейдите к п. 15, иначе проверьте неисправную линию на обрыв или короткое замыкание, устраните неисправность. Повторите проверку уровней. При отсутствии ошибок перейдите к п. 3.

15. Контроль регистров микроЭВМ. Полный контроль возможен только в ремонтной мастерской. В домашних условиях может быть произведена проверка лишь разряда РНП2 (@#177716). В режиме ТС занесите в ОЗУ с адреса 1000 и запустите программу:

012700 177716 012701 002000
012702 177777 011021 010210
011021 011021 000137 163412

Эта программа выполняет следующие действия:

1) занести адрес РНП в R0;

2) занести адрес ячейки в R1;

3) занести число 177777 в R2;

4) переслать содержимое РНП в ячейку с адресом, указанным в R1, и увеличить значение этого адреса на 2;

5) записать содержимое R2 в РНП;

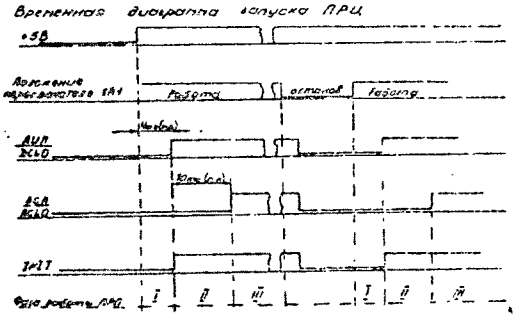


Рис.5

6) еще дважды прочесть РНП (в ячейки 2000+2 и 2002+2);

7) вернуться в режим тестов.

После запуска прочтите ячейки 2000, 2002, 2004 и обратите внимание на разряд ДА02. «Лог. 1» из этого разряда можно прочитать только в первом цикле чтения после обращения к РНП по записи (т.е. она должна быть только в ячейке с адресом 2002).

16. Порядок обращения к регистрам клавиатуры. По адресу 177662 прочтите произвольное число. На данном этапе контроля важно отсутствие зависания при обращении. При зависании проверьте цепь: D4/30, D19/38, D4/28. Возможна неисправность линий связи, БИС D19 или D14.

17. Контроль модулей ПЗУ. Директивой отладки X подсчитайте контрольные суммы и сравните их с табличными значениями, приведенными в руководстве оператора. При совпадении значений перейдите к п. 18. При зависании или несовпадении проверьте правильность установки БИС, целостность связей, наличие контакта с модулями ПЗУ. Проверьте связи D19/37, D19/1, D2/10, D7/3, D5/4. Если неисправность обнаружить не удалось, обратитесь в ремонтную мастерскую.

18. Контроль ячеек ОЗУ. При ошибке в тесте 1 в режиме ТК распечатывается значение адреса первой неисправной ячейки. С помощью директив отладки обратитесь к ней по записи/чтению. Проверьте наличие контакта с помощью вольтметра или осциллографа. В первую очередь это касается канальных связей БИС D19 (выводы 1—17, 34, 39, 40, 41). Проверьте наличие сигнала тактовой частоты на выводе D19/33. Устраните неисправности связей, в результате чего при обращении по адресу ОЗУ (по записи и по чтению) должен появляться сигнал на выводе XT3/B20. На данном этапе настройки информация может считываться неправильно (не соответствовать записанной).

Проверьте на обрывы и короткие замыкания адресные шины БИС D19 ДА00—ДА15 и А0—А6.

Проконтролируйте наличие пачек импульсов на адресных выходах БИС D19 в очередности: 18, 19, 20, 22, 23, 24, 25. Период следования пачек на каждом последующем контакте должен удваиваться. При отсутствии сигнала (пачек) возможна неисправность БИС D19.

Проверьте наличие импульсов на выводах D19: 32, 36, 35, 31, 28. При их отсутствии возможны неисправности связей или БИС D19, DS1—DS16 (развертка 0,2 мкс).

Проверьте наличие импульсов на выводах 14 микросхем DS1—DS16. При их отсутствии возможны неисправности одной или нескольких из этих микросхем. (Для синхронизации используйте сигнал WT D19/31 (развертка 0,2 мкс). В ОЗУ должна находиться информация, заносимая туда при включении питания.)

Для надежного тестирования отдельной ячейки ОЗУ с конкретным адресом <A> необходим испытательный стенд, однако по адресу <A> можно обратиться, записывая (считывая) в циклическом режиме некоторое число с помощью директивы отладки Ц, например командой 10Ц. (В этом примере проверяется запись «лог. 1» в разряд ДА3.)

Возможны неисправности микросхем DS1—DS16 по входам D1, обрывы ДА00—ДА15, неисправность БИС D22 и D25, нарушения в цепях управления БИС D22, D23: CS1, CS2, MD, EW, CLR, а также БИС DS1—DS16: RAS, CAS, WE.

19. Тест ОЗУ в режиме ТС. Вызовите режим ТС. При отсутствии перехода в ТС выполните п. 21. Запустите тест 1 (команда Т1). Если тест прошел без ошибок, перейдите к п. 20, иначе к п. 27.

20. Контроль ПЗУ (ТС). С помощью директив отладки Х, А, Д проверьте контрольную сумму (КС) модулей (аналогично п. 17). При совпадении КС перейдите к п. 21, при зависании, не совпадении и т.д. — к п. 27.

21. Проверка функционирования клавиатуры. Тест предназначен для оценки работоспособности пульта оператора и используется для уточнения диагностики, проведенной в режимах ТС и ТК. Для его проведения необходим стенд, поэтому в домашних условиях можно выполнить лишь частичную проверку.

Снимите перемычку S1 *. С помощью осциллографа проверьте уровни на выводах БИС D4 при ненажатых клавишах и сравните результаты с табличными.

Назначение вывода	Номер вывода	Уровень
Шины X	6—15	верхний
Шины Y	1—5, 40, 41	нижний
Канал	19, 23—30, 32—39	верхний
Регистровые	16—18	верхний
	31	нижний
Выводы подключения RC-цепочки	20	нижний
	22	верхний

При несоответствии проверьте наличие контактов в разъемах XT1 и XT2, состояние резисторов, задающих уровень, отсутствие коротких замыканий. Устраните неисправность.

Нажмите на любую клавишу (например, «РУС»). Проверьте сигнал на шинах X_i и Y_j, в пересечении которых находится нажатая клавиша, а также на соответствующих выводах БИС D4 X_i и Y_j (для клавиши «РУС» это X_1 и Y_6).

Проверьте наличие сигналов, соответствующих временной диаграмме рис.6. (Параметры сигналов приведены для справки.) При несоответствии проверьте наличие связей в цепях X и Y, в том числе контакт в колодках XT1 и XT2 кабелей межплатной связи, отсутствие коротких замыканий, работоспособность кнопочных переключателей. Проверьте также отсутствие обрывов в связях BS—CS (D4/30) и исправность БИС D4. Проконтролируйте длительность переходных процессов при коммутации кнопочных переключателей — она не должна превышать 0,3R3*C3. В противном случае возможны нарушения протоколов обмена информацией и потеря управления БИС D4. Устраните неисправность.

22. Тест программируемого интерфейса. Вызовите тест 3 (ТС). При отсутствии ошибок перейдите к п. 27 для контроля временной диаграммы регистров РНП и регистра смещения. Если тест не прошел, перейдите к п. 27 для контроля временной диаграммы порта.

23. Проверка функционирования клавиши «СТОП». Для проверки прерывания теста 4 клавишей «СТОП» проследите формирова-

* Перемычка S1 соединяет между собой выводы INIT микросхем D4 (вывод 19) и D14 (вывод 24). На принципиальной схеме (см. Приложение) она изображена над микросхемой D9.1. На плате вычислителя перемычка S1 расположена рядом с контактами A1, B1 разъема порта ввода-вывода (XT5), как это показано на монтажной схеме (см. Приложение). — *Прим. ред.*

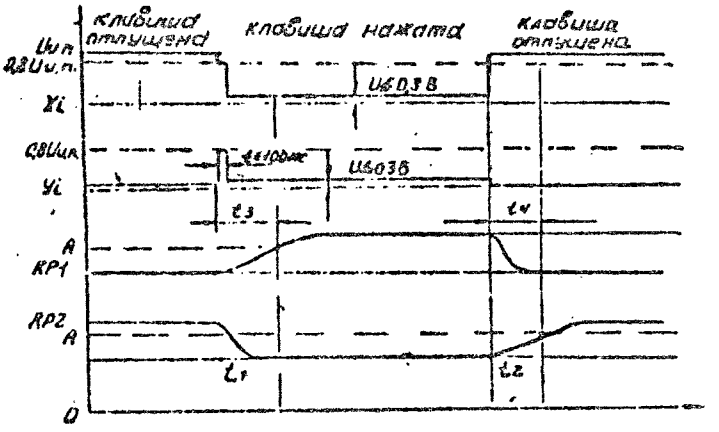


Рис. 6. Примечания:
1. t — определяется быстродействием БИС D4.
2. t1 и t2 — моменты коммутации клавишных переключателей. При коммутации допускаются переходные процессы (дребезг) в интервале не более 3.0 мс.
3. t3 и t4 — длительности переходных процессов (не более 3.0 мс).
4. A — уровень переключения входных цепей БИС D19.

ние импульса IRQ1 длительностью не менее 1 мкс: D1/2, D11/2, D11/12, D14/31. Проверьте отсутствие обрывов (замыканий) в цепях, работоспособность комплектующих.

24. Проверка схемы формирования видеосигнала. Установите переключатель «длительность» осциллографа в положение «2 мс». Отрегулируйте амплитуду сигнала в точке XT9/4 с помощью потенциометра R57 на соответствие временной диаграмме рис.7 и перейдите к п. 26. При отсутствии импульсов проверьте работу схемы формирования виде-

Ю. Березкин, И. Березенцев,
г. Пермь

Ремонт входного порта БК-0010.01

В компьютерах БК-0010.01 довольно часто выходит из строя входной порт, при этом программы перестают работать с джойстиком или мышью. Для проверки порта можно использовать тесты, зашитые в МСТД, но проще воспользоваться командами БЕЙСИКа. Подключите «блок нагрузок» из комплекта БК и введите: ?BIN$(PEEK(&O177714)). Если ответом будет 0, то порт исправен, иначе единицы в полученном результате укажут на неисправные разряды. (Номер неисправного разряда равен количеству знаков справа от единицы. Например, число 1001 говорит о том, что неисправны третий и нулевой разряды.) Для определения неисправного разряда порта можно также воспользоваться тестом 3 блока МСТД.

Вывод порта	Разряд	
	D17	D18
04	00	08
06	01	09
08	02	10
10	03	11
15	04	12
17	05	13
19	06	14
21	07	15

Если в разъеме порта нет механических повреждений (например, загнут штырек в разъеме, замыкание между дорожками), то скорее всего неисправна одна из микросхем К589ИР12. Вскрыв компьютер, вы увидите, что около разъема порта установлены четыре такие микросхемы. Младшие разряды входного порта (с нулевого по седьмой) обслуживает микросхема, обозначенная как D17, а старшие — D18. Если смотреть на компьютер спереди, то микросхема D17 вторая слева в ряду К589ИР12, а D18 — крайняя справа. Остальные обслуживают выходной порт. Неисправную микросхему нужно заменить.

(Указанный прием в редакции не проверялся. — *Прим. ред.*)

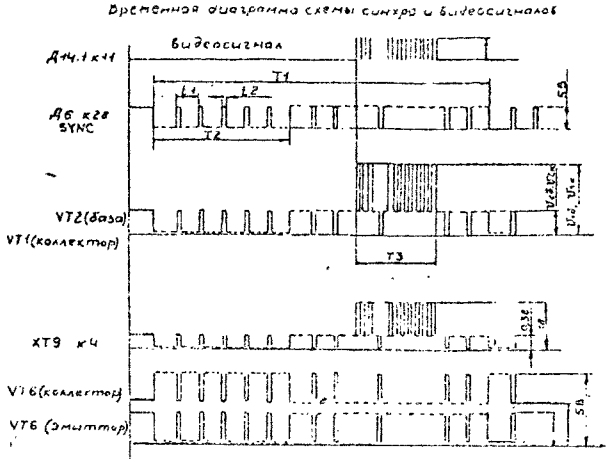


Рис.7

Примечания к рис. 7

1. t1=64 мкс — период строчного синхросигнала.
2. t2=4 мкс — длительность строчного синхросигнала.
3. T1=20 мкс — период кадрового синхроимпульса.
4. T2=6*t1 — длительность кадрового синхросигнала.
5. T3=20 мс — длительность информационного кадра.

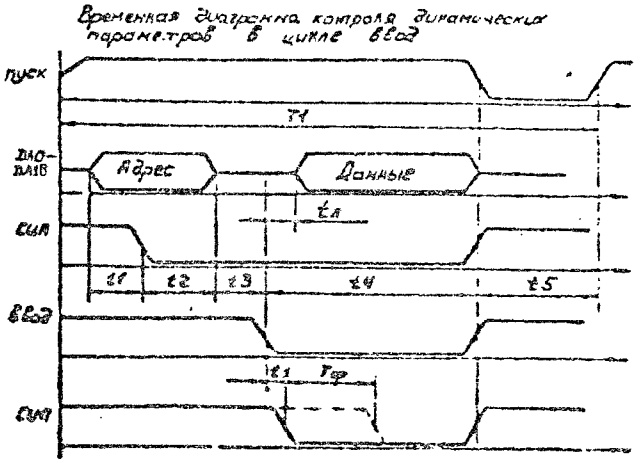


Рис.8

Примечания к рис. 8 и 9

1. Тср — среднее время обращения активного устройства (контрольный стенд) к ячейкам ОЗУ.
2. t — время ответа пассивного устройства.
3. tл — время установления данных.
4. t1...t4 — регулируемые задержки формирования интерфейсных сигналов.
5. Т1 — период повторения временной диаграммы (Т1=t1+t2+t3+t4+t5).

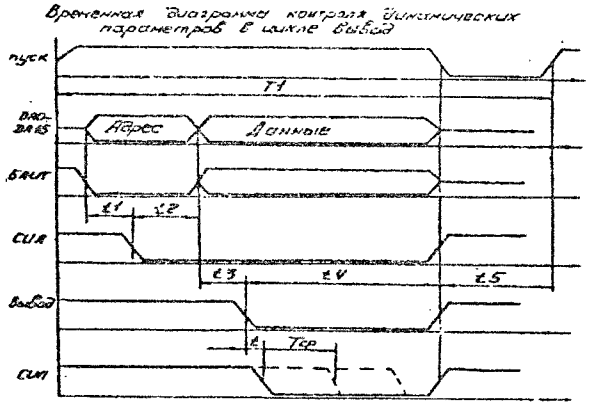


Рис.9

осигнала на соответствие временной диаграмме рис.7. Для этого подготовьте ОЗУ экранной памяти объекта контроля в режиме ТС: **20000А20Д177777Р 20020А125252Р 20040А52525Р 20000А0Р 20000А40000Д20100П.**

При исправно работающем контроллере ТВ-приемника на всей площади экрана должны наблюдаться четыре вертикальные полосы. Проведите контроль в следующей последовательности:

на входах 1, 2, 13, 22 и 23 D24 и D25 проверьте наличие высокого уровня;

на входах D0—D7, D24 и D25 — информация ОЗУ;

на выходах Q0 D24 и D25 — последовательный код;

на выходах дешифратора D10/6, D5/12, D20, D10/3, D10/4, D9/9 — импульсный сигнал;

замкните выводы 3 и 5 разъема XT9 и проверьте D21/8, D21/10, VT8, XT9/4;

замкните выводы 1 и 3 разъема XT9 и проверьте D21/2, D21/4, D21/6, XT9/4.

Если считанные данные не равны записанным, обратитесь в ремонтную мастерскую для контроля на стенде. Возможны неисправности связей, неисправность D19.

26. Проверка схем магнитофонного интерфейса. При несоответствии сигнала в точке XT4/5 диаграмме рис.2 проверьте номиналы резисторов R29—R31 и связи в цепи D12/5, D12/11, D13/3, D13/4, D13/12, D13/11, XT4/5, D14/7.

Если в ходе теста 5 магнитофон не включается (не выключается), подключите вольтомметр в режиме омметра к контактам 1 и 4 разъема XT4 и наберите директивы МП, МО:

МП — контакты реле замкнуты;

МО — разомкнуты.

Проверьте связи D12/14, D13/5, 10VT3. Возможна неисправность БИС D12, D13, VT3.

При несоответствии сигнала в точке D12/7 требованиям п. 10 проверьте связи, элементы схемы усилителя приема данных с магнитной ленты. Устраните неисправность.

Таблица 2

Данные для записи	Реакция на запись	Считанные данные
1330	исходное состояние	1330
1342	смещение информации вверх на одну строку	1342
330	смещение информации вниз на одну строку, гашение нижней части экрана	330
1330	исходное состояние	1330

При несоответствии сигналов временной диаграмме рис.7 проверьте соответствующие связи. Устраните неисправность.

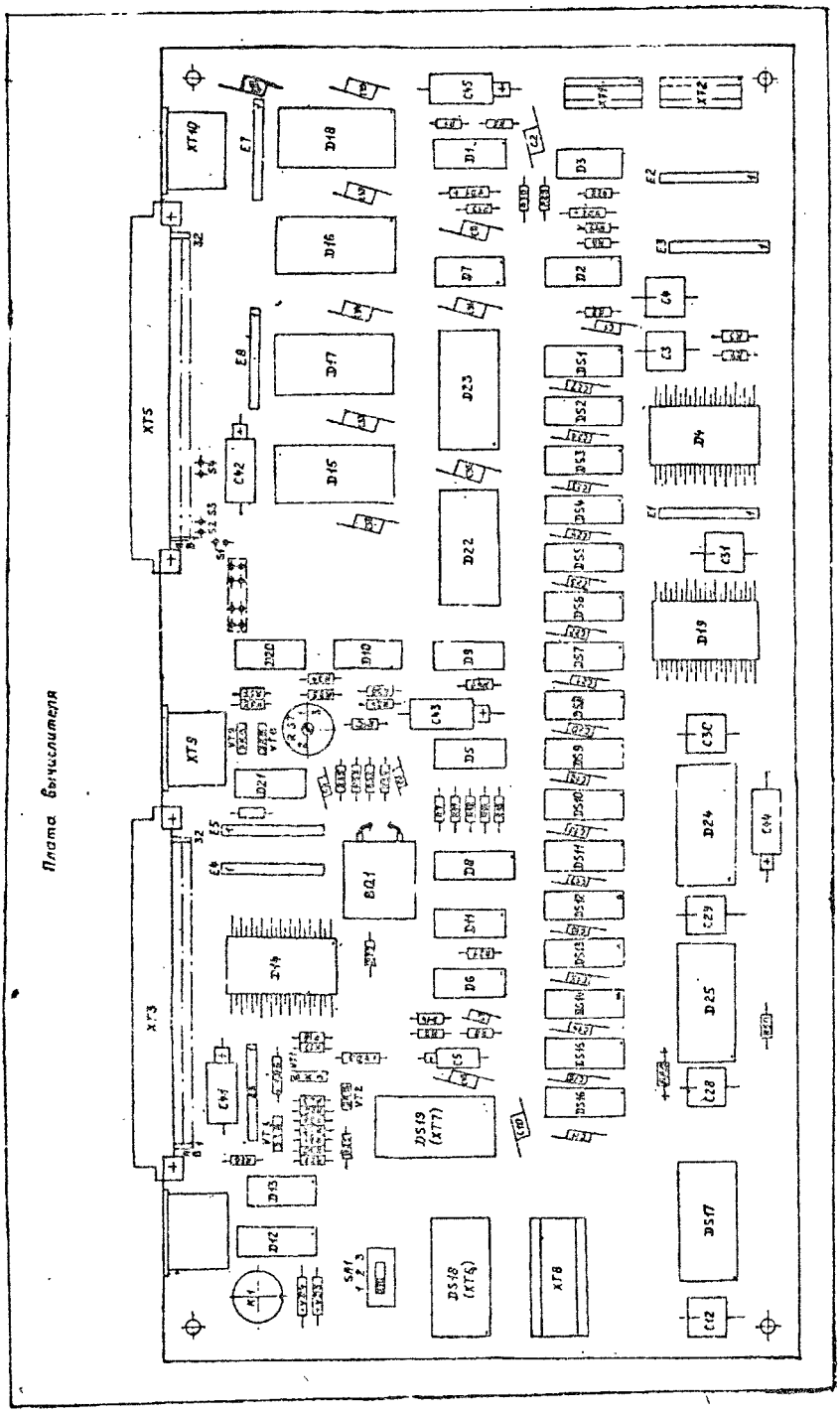
25. Контроль регистра рулонного смещения. При отсутствии изображения в нижней части экрана или неравномерной печати строк (неравные расстояния между строками, сбои при печати) проверьте регистр сдвига (адрес 177664), записывая в него и считывая последовательно в режиме ТС следующие данные (команда **177664А <записываемое число>И И<выдается результат>**) — см. табл.2.

перейдите к п. 3.

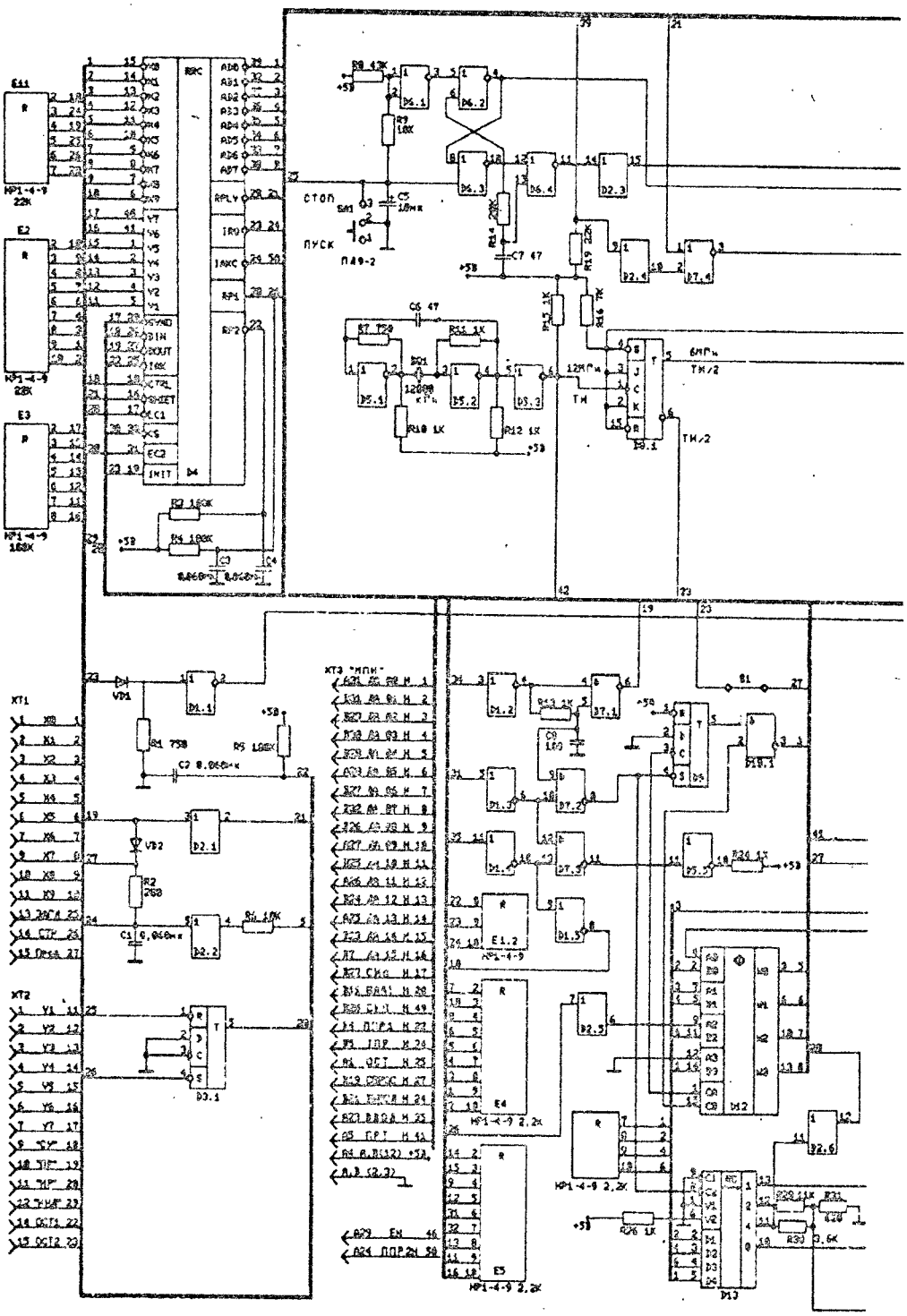
27. Проверка микроЭВМ в рабочем режиме. С помощью директив отладки задайте адрес контролируемой ячейки ОЗУ или регистра и данные (при записи). Проверьте импульсные сигналы в контрольных точках на соответствие временным диаграммам рис.8, рис.9. Проверьте номиналы элементов, подключенных к шине, на которой отмечено превышение значения задержки. Возможна неисправность БИС. Устраните неисправность (замените БИС), перейдите к п. 3

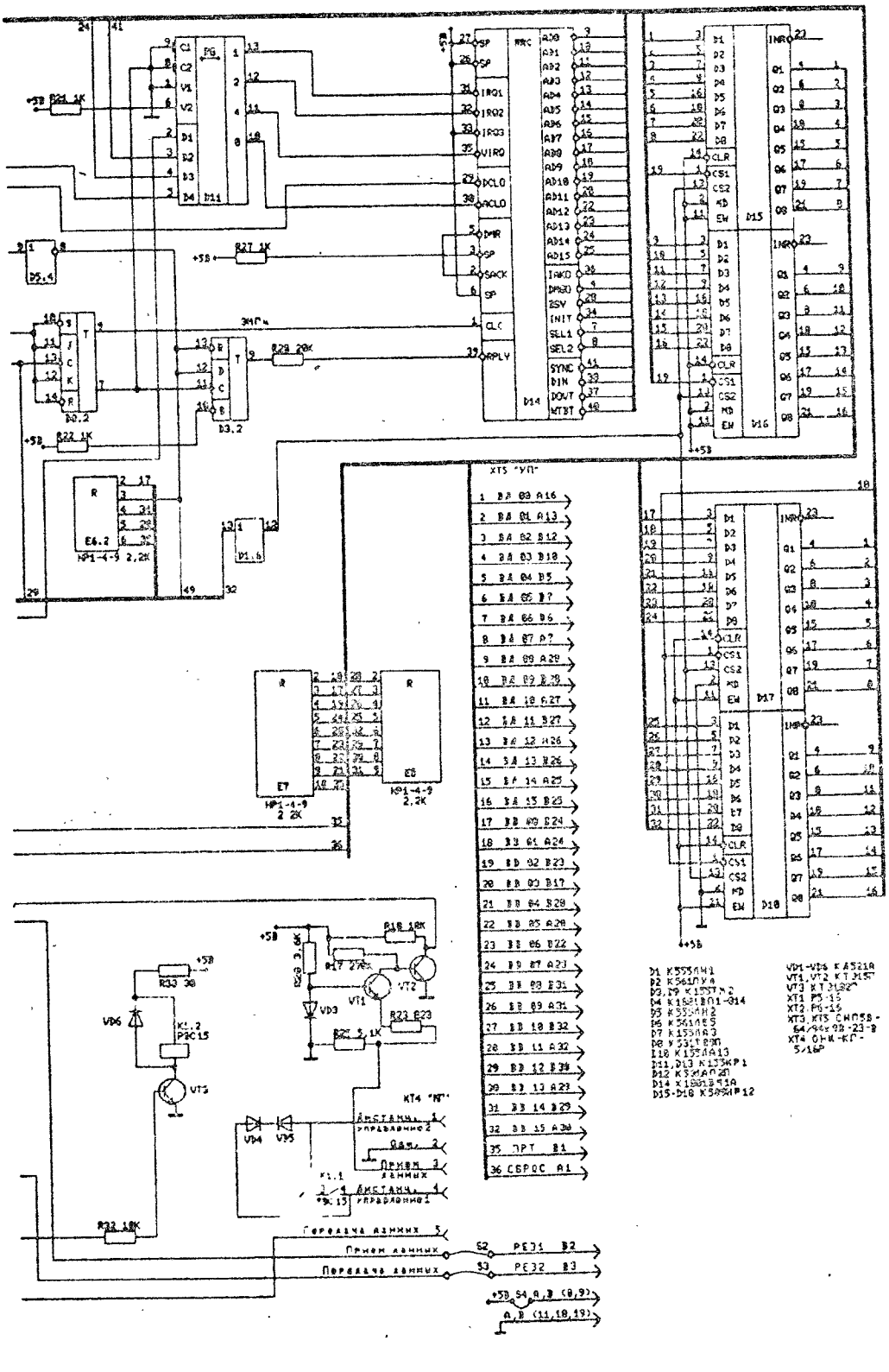
Приложения

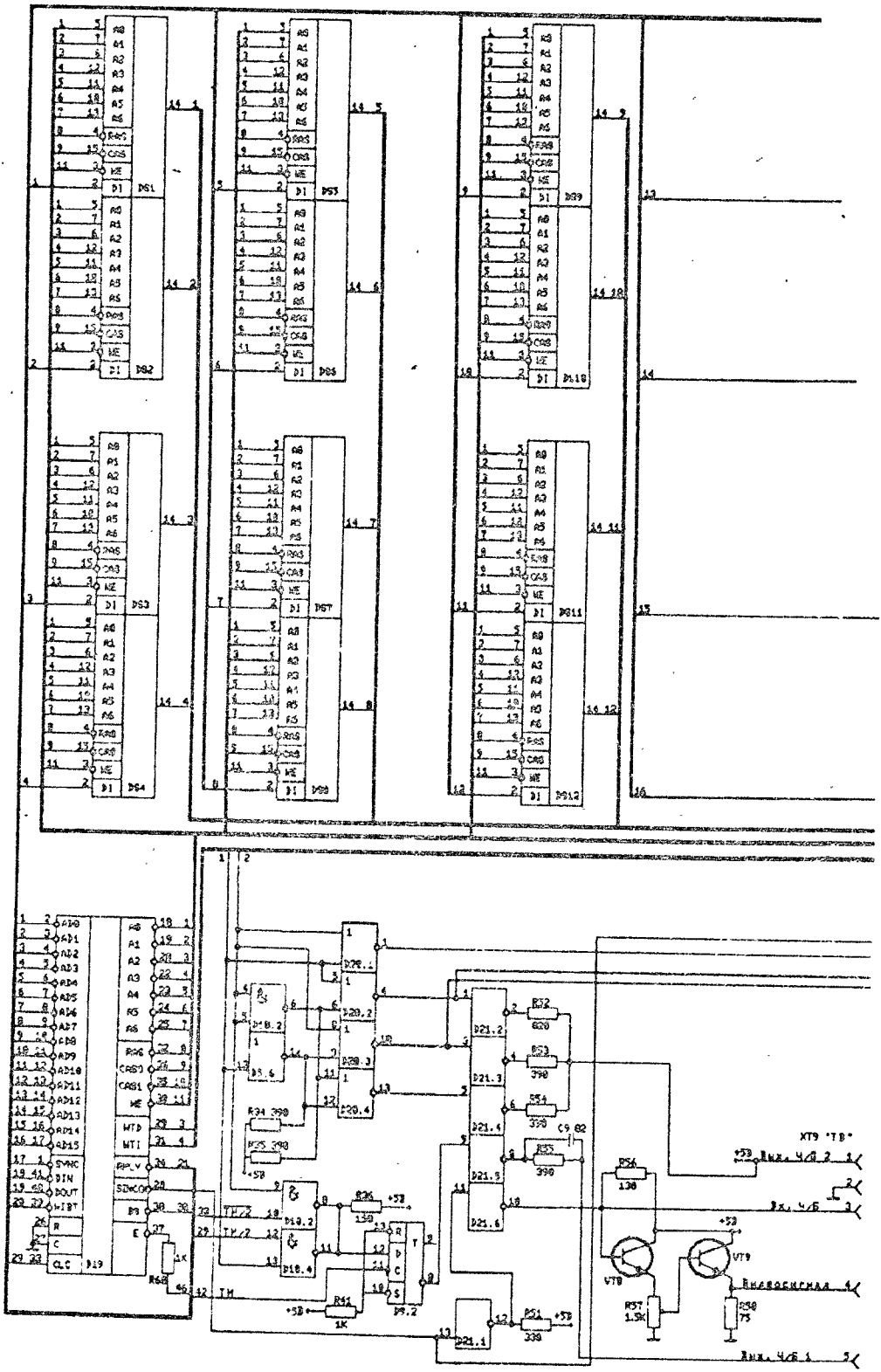
Монтажная схема платы вычислителя

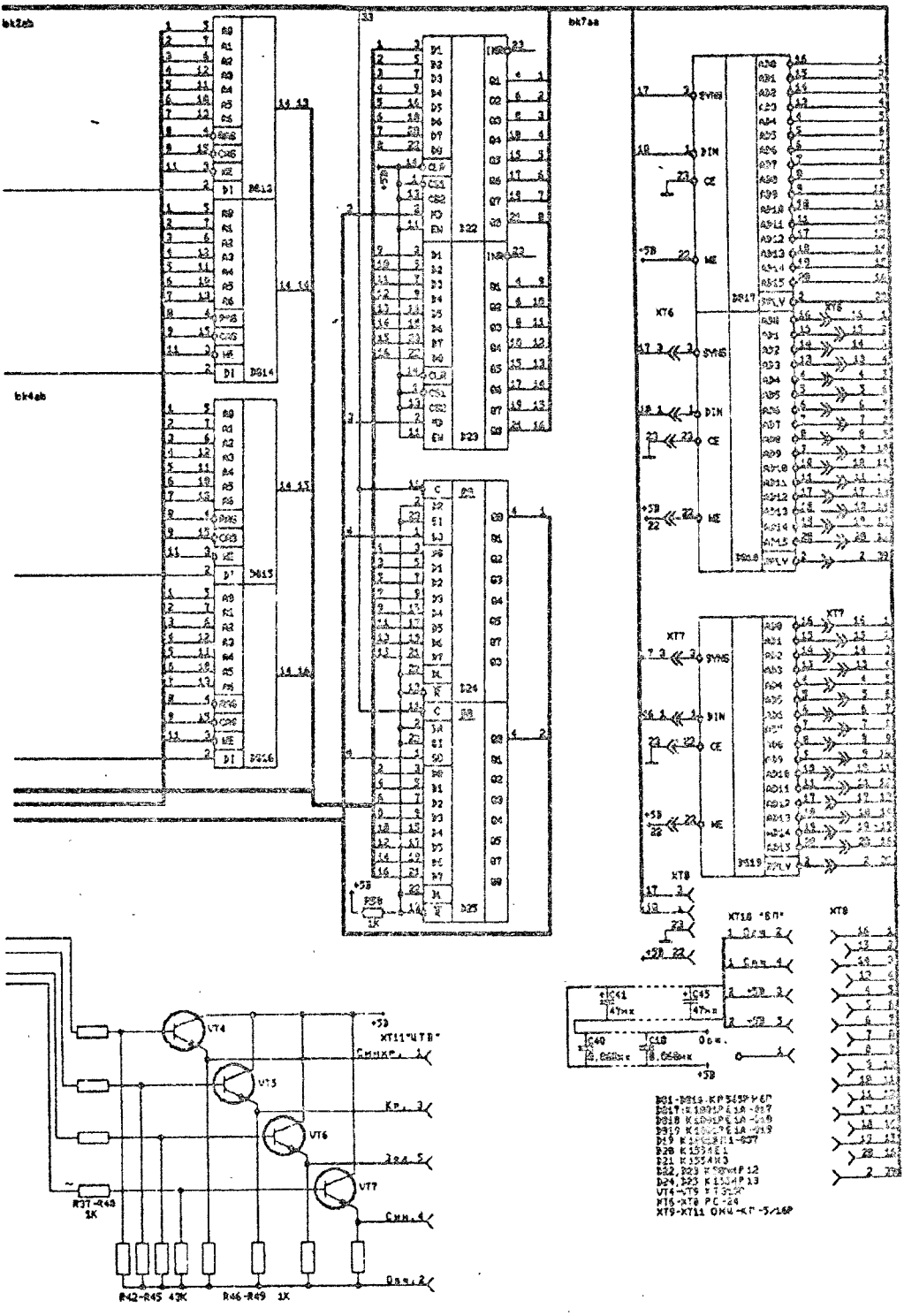


Принципиальная схема БК-0010

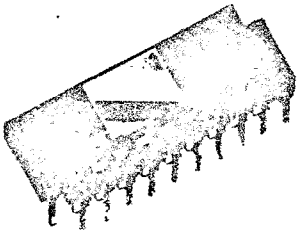



+5В
R21 1К
R27 1К
R29 20К
R22 1К
ЗИПч
D5.4
D0.2
D3.2
D1.6
R
E6.2
КР1-4-9 2,2К
E7
E8
КР1-4-9 2 2К
КР1-4-9 2,2К
XT5 "УП"
D15
D16
D17
D18
R33 30
R18 18К
R17 270
R24 2,2К
R23 623
R32 1К
VT1
VT2
VT3
VD3
VD4
VD5
VD6
КТ4 "М"
Передача данных
Прием данных
Передача данных
РЕ31 В2
РЕ32 В3
+5В 54 А,В (8,9)
А,В (11,18,19)




bk2ab
bk4ab
bk7aa
XT6
XT7
XT8
XT10 "ВП"
XT11"ЧТВ"
Синхр.
Кр.
Зел.
Син.
Общ.
VT4
VT5
VT6
VT7
R37-R43
1K
R42-R45 47K
R46-R49 1K

СПРАВОЧНОЕ БЮРО



О том, что некоторые из программ, прошитых в ПЗУ БК, можно использовать в своих разработках, знают многие. Но вот что собой представляет само это ПЗУ, каковы его особенности, известно далеко не всем. Надеемся, что справочные сведения, приведенные в данной статье, помогут заполнить этот пробел.

Д. Ю. Усенков,
г. Москва

ПЗУ ДЛЯ БК

Наиболее «чувствительным» отличием БК-0011 и БК-0011М от семейства БК-0010 является больший объем памяти. Те же, кто имеет БК-0010 и не надеется купить БК-0011, пытаются любыми доступными способами «выжать» из «десятки» все, что возможно. В ход идут любые средства, начиная от чисто программных (сверхоптимизация программ, кодовые «вставки» в БЕЙСИКе, использование подпрограмм из «базовых» ПЗУ БК) до расширения памяти аппаратным способом. И несмотря на все возрастающую популярность расширенного ОЗУ, сравнительно более дешевые модули ПЗУ (дополнительно устанавливаемые в отсек пользователя, блок МСТД, либо изготавливаемые в виде дополнительных блоков) все еще прочно «держат свои позиции». В связи с этим возрастает важность справочной информации о микросхемах ПЗУ. Как правило, эти сведения разрозненны, разбросаны по разным источникам, тогда как хотелось бы иметь их «в концентрированном виде».

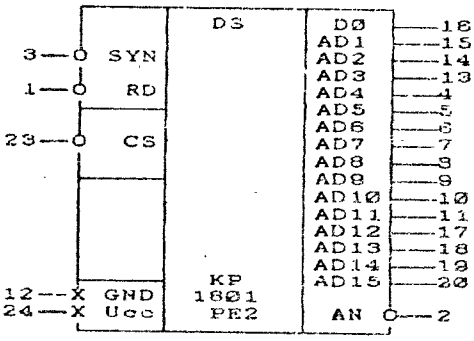
На БК могут быть использованы микросхемы ПЗУ трех типов: КР1801РЕ2 (РЕ1), К1801РР1 и К573РФ3. ПЗУ КР1801РЕ2 используются, как правило, в заводских блоках БК, тогда как К1801РР1 и К573РФ3 популярны среди пользователей БК, имеющих собственный программатор.

Характеристики ПЗУ КР1801РЕ2

КР1801РЕ2 представляют собой неперепрограммируемые ПЗУ масочного типа, в которых прошивка двоичной информации в виде структуры ячеек, содержащих или не содержащих проводящие перемычки, выполняется при изготовлении кристалла микросхемы.

Емкость: 4Кб×16.

Исполнение: полимерный корпус 31,5×15,3×4,0 мм (239.24-1).



Нумерация и назначение выводов:

№ вывода	Обозначение	Функциональное назначение
1	RD	Сигнал «Чтение» (DIN)
2	AN	Сигнал «Ответ» (RPLY)
3	SYN	Сигнал «Синхронизация» (SYNC)
4—11	AD4—AD11	Разряды адреса/данных
12	GND	Общий
13—16	AD3—D0	Разряды адреса/данных
17—20	AD12—AD15	Разряды адреса/данных
23	CS	Сигнал выбора микросхемы
24	Ucc	Питание +5В
21	A16	Дополнительные входы для КР1801РЕ1
22	A17	

Характеристики ПЗУ К573РФ3 и К1801РР1

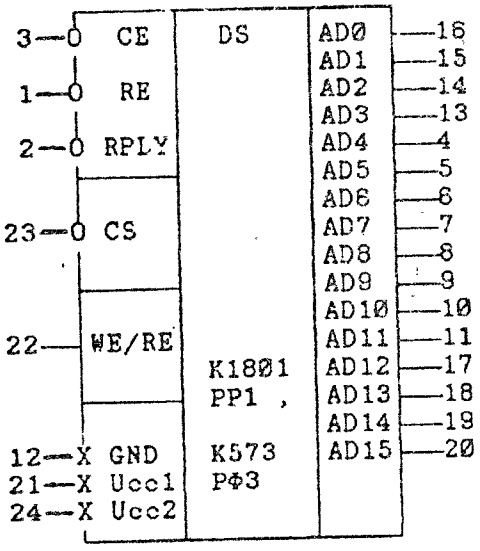
К573РФ3 — ПЗУ с ультрафиолетовым стиранием, позволяющее при необходимости перепрограммировать содержащуюся в нем информацию. Принцип действия ПЗУ данного типа основан на удержании и хранении после программирования в полупроводниковых ячейках остаточного заряда, снимаемого (при необходимости повторного программирования) под действием ионизирующих излучений. Недостатком таких микросхем по сравнению с масочными является меньшая надежность хранения информации. Причина этого, с одной стороны, заключается в чувствительности к излучениям достаточно большой мощности, разрушающим хранимую информацию, а с другой — в эффекте релаксации (постепенного самопроизвольного рассасывания остаточных зарядов при длительном хранении ПЗУ без подачи питания).

К1801РР1 также является перепрограммируемым ПЗУ (ППЗУ), однако, в отличие от К573РФ3, здесь стирание информации производится электрическим путем. В остальном же эти два типа микросхем аналогичны. Так как К573РФ3 в настоящее время достаточно дефицитна, К1801РР1 сегодня наиболее популярна среди пользователей БК.

Емкость: 4К6Х16.

Исполнение: металлокерамический корпус 29,5Х15,0Х6,0 мм (210Е.24-5) — для К573РФ3.

Нумерация и назначение выводов:



№ вывода	Обозначение	Функциональное назначение
1	RE	Сигнал «Чтение» (DIN)
2	RPLY	Сигнал «Ответ» (RPLY)
3	CE	Сигнал «Синхронизация» (SYNC)
4—11	AD4—AD11	Разряды адреса/данных
12	GND	Общий
13—16	AD3—AD0	Разряды адреса/данных
17—20	AD12—AD15	Разряды адреса/данных
21	Ucc1	Питание 1 (чтение: +5В, программирование: К573РФ3 — +18В, К1801РР1 — +24В; стирание: (К1801РР1) —18В)
22	WE/RE	Сигнал «Программирование /чтение»
23	CS	Сигнал выбора микросхемы
24	Ucc2	Питание +5В

Особенности микросхем ПЗУ КР1801РЕ2, К1801РР1 и К573РФ3

Указанные ПЗУ имеют две особенности, отличающие их от микросхем других типов. Первое отличие ясно из приведенных выше таблиц. Выводы AD0—AD15 используются двояко — и как адресные входы, и как выходы данных (что согласуется с идеологией общей шины БК, также служащей для передачи и адресов и данных). При этом текущее «назначение» выводов AD0—AD15 определяется управляющими сигналами. (В КР1801РЕ2, предназначенных для хранения двухбайтных машинных слов, значение нулевого бита адреса всегда принимается равным «0» — четные адреса. Поэтому указанный вывод обозначается как D0 и используется только как нулевой разряд данных.)

Вторая особенность не столь очевидна, но обнаруживается в результате логических рассуждений. Если объем ПЗУ равен 4Кб, то для адресации к его ячейкам достаточно 13 разрядов адреса, тогда как в указанных микросхемах задействованы все 16 разрядов. Для чего же нужны лишние 3 разряда? Оказывается, кроме обычного входа CS (выбор микросхемы), предназначенного для «включения/выключения» использования микросхемы «командой извне» (именно так обеспечивается отключение БЕЙСИКа в БК-0010.01 при подключении блока МСТД), в микросхе-

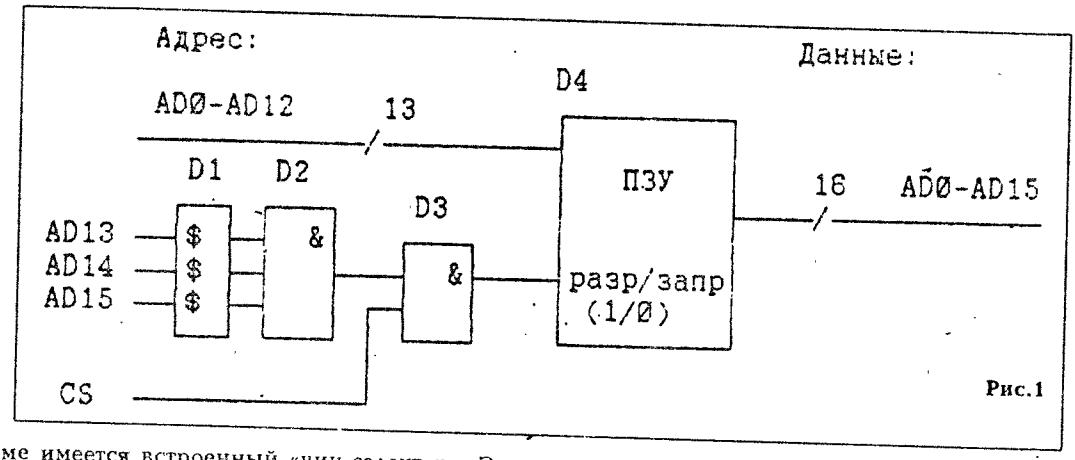


Рис.1

ме имеется встроенный «чип-селектор». Эта дополнительная ячейка, также программируемая наряду с ячейками памяти, служит своего рода «фильтром» адресного сигнала. Если сигналы на входах AD13—AD15 соответствуют хранящейся в этой ячейке триаде битов, микросхема «открывается» и значение адреса, установленное на входах AD0—AD12, используется для выбора ячейки данных. В противном случае микросхема «запирается» и адрес на входах AD0—AD12 игнорируется. Упрощенно логику работы встроенного «чип-селектора» можно представить в виде следующей схемы (рис.1).

Знаки $ в составе D1 указывают на инвертирование сигнала (если в соответствующем бите триады запрограммирован 0) или на неинверсный сигнал (если бит триады равен 1).

Таким образом, если в ячейке «чип-селектора» установлена триада 000, данная микросхема будет «откликаться» на адреса от 0 до 17777, триада 100 — от 100000 до 117777, 111 — от 160000 до 177777 и т.д.

Такая немного сложная на первый взгляд структура микросхемы позволяет значительно упростить конструкцию БК. Попытаемся подключить несколько ПЗУ «одну над другой», запараллелив все выводы микросхем, как это показано на схеме (рис.2).

Подобная схема способна привести в ужас любого дипломированного конструктора цифровых устройств. Действительно, если на все четыре ПЗУ поданы одни и те же управляющие сигналы (соответственно, все четыре активны), сигналы адреса будут одновременно «приняты к рассмотрению» на всех четырех микросхемах, и каждая будет «стараться» выдать на выходы данных «свое» значение. Что при этом получит процессор от такого блока ПЗУ — один Бог ведает. Поэтому-то и используются в блоках ПЗУ дополнительные микросхемы — дешифраторы, принимающие на входе те самые «лишние» разряды адреса и генерирующие в соответствии с ними ряд сигналов выбора микросхемы (CS1, CS2, CS3, CS4, ...), что обеспечивает

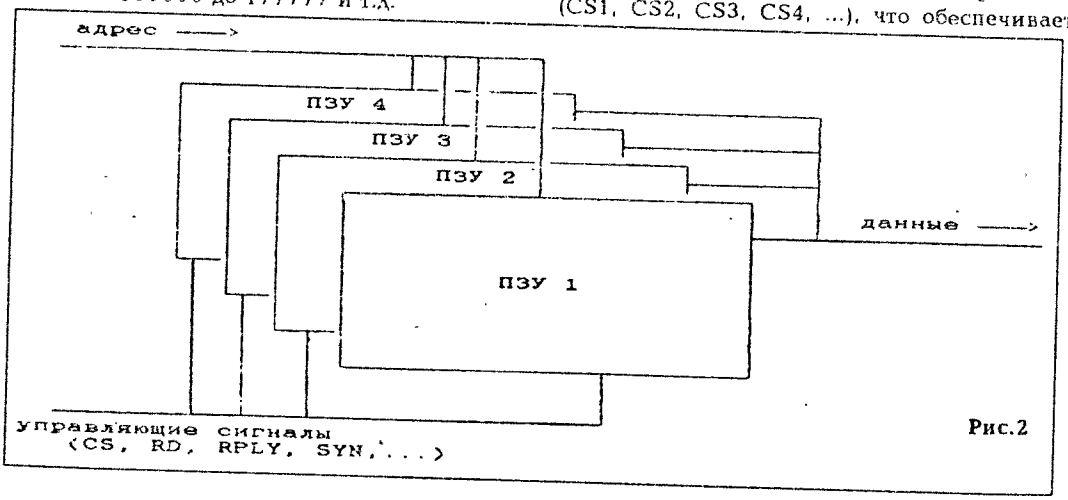


Рис.2

действие в каждый момент времени только одной микросхемы из набора.

А вот в нашем случае простейший дешифратор (ранее названный нами «чип-селектором») встроен в каждую микросхему, и «неправильная», казалось бы, схема подключения будет нормально работать (и работает, например, в блоке МСТД).

Какие же практические рекомендации следуют из всех этих теоретических рассуждений? Если у вас имеется дополнительная микросхема ПЗУ типа КР1801РЕ2 (РЕ1), К1801РР1 или К573РФ3 с прошитой в ней программой, ее можно смело устанавливать в гнездо пользователя (БК-0010) или в резервное место в блоке МСТД (БК-0010.01). (Так как микросхемы плохо переносят перегрев при пайке, необходимо распаять в контактные отверстия платы блока МСТД розетку типа РС—24 и затем вставлять микросхемы ПЗУ в нее.) Если же у вас имеется ПЗУ любого другого типа (например, РФ2 или РФ5), необходимо разрабатывать и изготавливать схему блока ПЗУ «по всем правилам», с применением дешифраторов и пр.

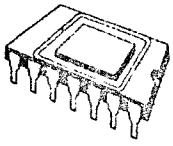
И в заключение приведем соответствие номеров прошивки ПЗУ заводского изготовления (КР1801РЕ2) их содержимому.

017 — монитор БК-0010 (БК-0010.01)
018— ФОКАЛ БК-0010 (ФОКАЛ-МСТД БК-0010.01)
019— тесты и отладочный монитор «ТС» БК-0010 (МСТД БК-0010.01)
084— ФОКАЛ БК-0010Ш для работы с локальной сетью
106 — БЕЙСИК БК-0010.01, первая микросхема
107 — БЕЙСИК БК-0010.01, вторая микросхема
108 — БЕЙСИК БК-0010.01, третья микросхема
198 — БЕЙСИК БК-0011, первая микросхема
199 — БЕЙСИК БК-0011, вторая микросхема
200 — БЕЙСИК БК-0011, третья микросхема
201 — монитор БК-0011, первая микросхема
202 — монитор БК-0011, вторая микросхема
253 — контроллер дисковода БК-0011
324 — монитор БК-0011М, первая микросхема
325 — монитор БК-0011М, вторая микросхема
326 — контроллер дисковода БК-0011М
327 — БЕЙСИК БК-0011М, первая микросхема
328 — БЕЙСИК БК-0011М, вторая микросхема
329 — БЕЙСИК БК-0011М, третья микросхема

Кроме того, разработаны ПЗУ с прошивками «ПРОЛОГа-Д» (см.: ИНФО. 1992. №3—4. С.38), «Менестрель» (программно-аппаратный музыкальный комплекс для БК), некоторых дисковых операционных систем и т.п.

Литература:

Микропроцессоры: Справочное пособие /Под ред. Ю. А. Овечкина. Л.: Судостроение, 1987



Бесспорно, работать с дисководом намного удобнее, чем с магнитофоном. Тем не менее стандартный дисковод от БК-0011М, который чаще всего и подключают к БК-0010(.01), нередко «радует» своих владельцев разнообразными «мелкими пакостями». А чтобы от них избавиться, нужно знать, как работает схема КНГМД, какова роль тех или иных ее элементов. Об этом, а также о ряде рекомендаций по повышению качества работы с диском и идет речь в данной статье.

Борис Ф. Фролкин,
г. Москва

КОНТРОЛЛЕР ДИСКОВОДА БК-0010/11 И ЕГО ДОРАБОТКА

Контроллер дисковода для БК-0010 существует в нескольких вариантах: «стандартный» («пустой»), «зеленоградский», «армянский» и некоторые другие. Наиболее распространен «стандартный», поставляемый в комплекте с БК-0011(М). Он дорабатывается для работы с БК-0010 установкой двух резисторов аналогично тому, как это сделано в блоке МСТД.

Вторым по распространенности можно назвать «зеленоградский», выпускаемый МП «ComCon» и МП «Универсал». Схемотехнически он построен на основе контроллера дисковода КМД УКНЦ и поэтому отличается от «стандартного» битами выбора дисковода и включения предкомпенсации в регистре управления. В число положительных черт входят имеющееся «с рождения» встроенное

ОЗУ на 4 Кб, зашитая в ПЗУ ОС БК (имитация RT-11) и кнопка «сброс».

«Армянский» контроллер выпускался МП «Ширакаци». Число его пользователей, по видимому, невелико, во всяком случае в России. Он существенно отличается по схемотехнике от двух описанных выше моделей, так как построен на основе микросхемы КР1818ВГ93 (все остальные контроллеры — КР1801ВП1-128). К числу его недостатков можно отнести слишком большое энергопотребление и необходимость подключения дополнительных источников питания на 5 и 12 В (все остальные контроллеры берут питание 5 В непосредственно от БК).

Встречаются также контроллеры, переделанные из КМД УКНЦ. Однако такое сооружение можно назвать контроллером лишь с большой долей условности. Чтобы его запустить, необходимо навесным монтажом собрать синхрогенератор на 4 МГц, доработать плату для блокировки ПЗУ в области адресов 170000—177777, а кроме того, «навесить» разъем-переходник для подключения к БК. Полученная в результате конструкция совершенно не удовлетворяет никаким технологическим нормам. По поводу же аргумента о невысокой стоимости КМД УКНЦ и дефицитности КНГМД БК можно заметить, что сегодня дело обстоит с точностью до наоборот.

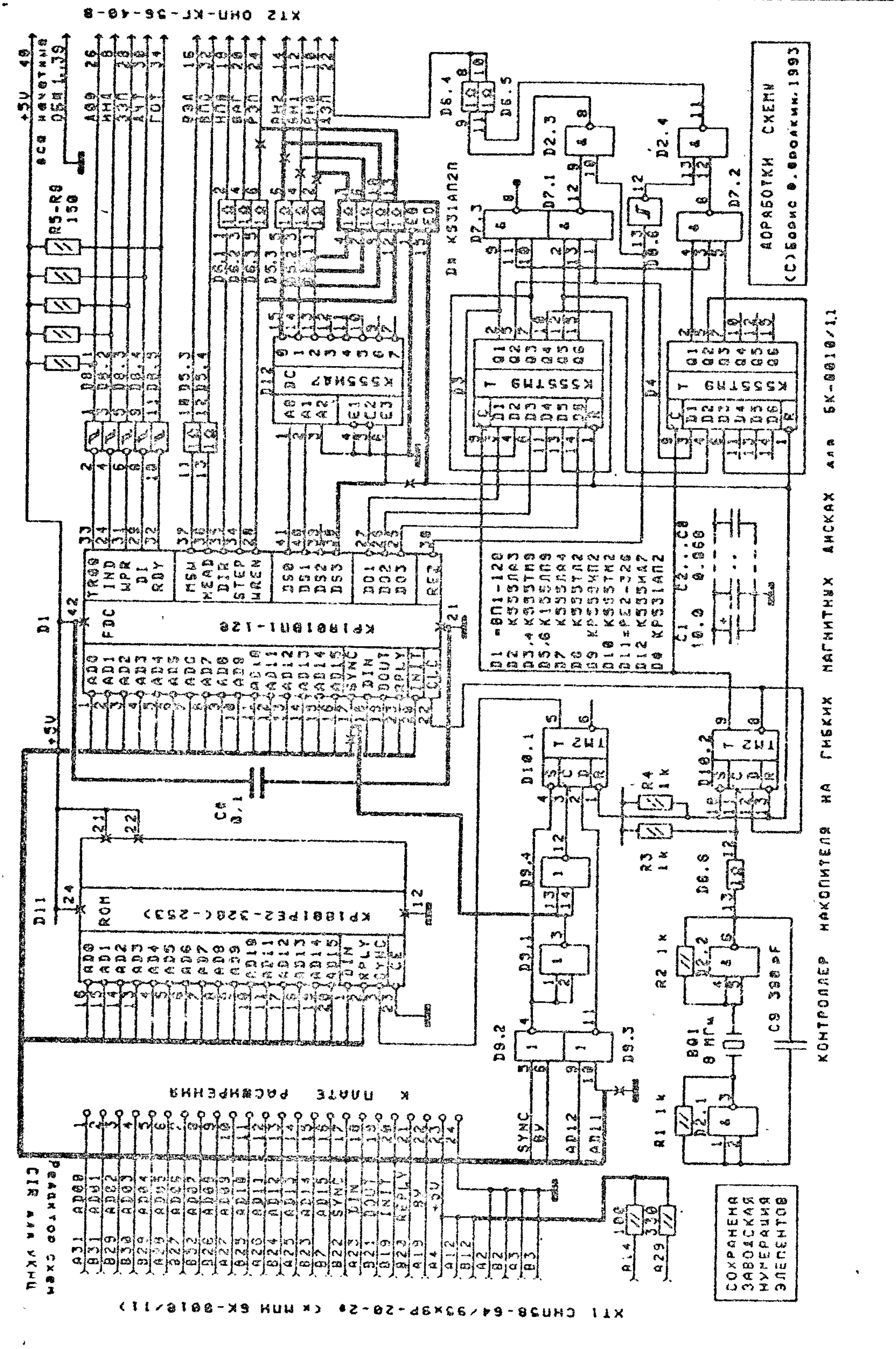
Периодически можно услышать о том, что где-то появилась какая-то новая модификация контроллера, но эти слухи, как правило, оказываются несколько преувеличенными, и единичные экземпляры новинок отмирают сами собой. Поэтому остановимся на стандартном контроллере и рассмотрим его более подробно (см. схему).

«Сердцем» контроллера, несомненно, является БИС КР1801ВП1-128. Именно она занимается перекачиванием животворного потока информации от дисковода к БК и обратно, следя попутно за отсутствием ошибок. БИС подключается через разъем непосредственно к шине БК и позволяет процессору оперировать с той информацией, к которой он «привык», т.е. с 16-разрядными словами. Дисководу же она поставляет тоже привычную для него информацию — закодированный специальным образом (а именно по способу МЧМ — Модифицированной Частотной Модуляции) поток битовых данных. Попутно БИС подсчитывает Контрольный, Циклический Код (КЦК или CRC) для обнаружения возможной ошибки. Кроме того, БИС содержит в своем составе специальную схему, позволяющую осуществить предкомпенсацию при записи (о чем будет сказано ниже) и устройство для инерционной цифровой фазовой автоматической подстройки частоты при чтении (подробнее см. [2, с.482]).

К процессору БК, как уже упоминалось, КР1801ВП1-128 подключается непосредственно, но к дисководу напрямую подключиться невозможно — требуются буферные элементы. Эту роль для всех сигналов, приходящих с дисковода (сигналы от датчиков нулевой дорожки, индексного отверстия, защиты от записи (заклейка на дискете), данные чтения с дисковода и сигнал готовности), играет микросхема D8 (К555ТЛ2). Для сигналов управления, подаваемых на дисковод, аналогичную задачу выполняют микросхемы D5 и D6 (К155ЛП9). Они буферизуют сигналы включения электродвигателя, выбора поверхности (нижняя/верхняя сторона дискеты), направления шагов магнитной головки (от центра/к центру), шага (одиночный импульс — один шаг), разрешения записи (чтение/запись), выбора накопителя 0...3 (не допускается одновременный выбор нескольких накопителей) и, наконец, данных для записи на дискету.

Дешифратор D12 (К555ИД7) включен в схему контроллера с единственной целью — сделать аппаратно невозможным одновременный выбор нескольких дисководов. Это решение представляет собой некоторую перестраховку на случай сбоев в программном обеспечении. Но можно было обойтись и без него. Дело в том, что все дисководы по стандарту имеют выходы на шлейф с открытым коллектором, и, даже если все они будут выбраны одновременно, ничего страшного не произойдет. Кстати, резисторы R5...R9 необходимы именно для нагрузки этих выходов с открытым коллектором.

Более интересно сообщество микросхем D3, D4 (К555ТМ9), D7 (К555ЛА4), D8.6 (К555ТЛ2) и D2.3, D2.4 (К555ЛА3). Это схема прекоррекции. Дело в том, что при чтении данных, записанных на любой магнитный носитель, магнитные домены как бы отталкиваются один от другого, т.е. короткие по времени паузы между импульсами увеличиваются, а длинные — уменьшаются. На внешних дорожках дискеты, где плотность записи невысока, это явление несущественно. Но для внутренних дорожек, чтобы получить при чтении стандартное во времени расположение импульсов, необходимо при записи немножко «сдвинуть» близкие импульсы и «раздвинуть» далеко стоящие друг от друга. На одном из выходов DO1...DO3 БИС КР1801ВП1-128 в зависимости от записываемой кодовой комбинации, формируется импульс данных для



КОНТРОЛЛЕР НАКОПИТЕЛЯ НА ГИБКИХ МАГНИТНЫХ ДИСКАХ для БК-0010/11

записи на дискету. Причем, если импульс при включенной предкомпенсации должен остаться «на своем месте», он выдается по выходу DO2, если немножко задержан — DO3, а если с небольшим опережением, то DO1. На триггерах микросхем D3 и D4 реализованы линии задержки (каждый триггер — по 250 нс), тактируемые сигналом 4 МГц в противофазе с КР1801ВП1-128.

Поскольку физически реализовать опережение невозможно, используется следующий прием. Импульс с DO1 передается без задержки, с DO2 — с задержкой и с DO3 — с двойной задержкой. При этом (относительно DO2) получаем как раз то, что нужно. Элементы микросхемы D7 работают как «ИЛИ» (в отрицательной логике), собирая воедино три канала прохождения импульсов. Тогда на выходе D7.3 сигнал записываемых данных соответствует отсутствию предкомпенсации, на D7.1 — предкомпенсации 250 нс и на D7.2 — предкомпенсации 500 нс. Выход REZ КР1801ВП1-128 управляет включением/выключением предкомпенсации. Легко заметить, что на самом деле происходит лишь переключение между различными ее уровнями — 250 и 500 нс.

Задающий генератор на 8 МГц собран по стандартной схеме на элементах D2.1 и D2.2 и каких-либо особенностей не имеет. Его частота делится на два триггером D10.2 (К555ТМ2) и в противофазе подается на FDC и линию задержки.

Если КР1801ВП1-128 является «сердцем» контроллера, то ПЗУ КР1801РЕ2-326 (или КР1801РЕ2-253) можно назвать его «мозгом». Именно в этом ПЗУ «зашиты» подпрограммы низшего уровня для поддержки обмена данными с дисководом. Именно от качества находящегося там драйвера зависит надежность работы с диском. И если «сердце» можно оценить как великолепное, а схемное обрамление — посредственное или терпимое, то с «мозгом» явно не все в порядке. (Подробнее об этом см. [3].) Конечно, имеющиеся в ПЗУ ошибки не смертельны и с ним можно работать, но именно с этими ошибками связаны периодические «нечтения» хороших дискет на вполне исправных дисководах.

Оставшиеся микросхемы D9 и D10.1 предназначены для единственной цели — ограничить область адресов ПЗУ D11 до диапазона 160000—167777. При непосредственном подключении входа SYNC КР1801РЕ2-326 к одноименному сигналу магистрали процессора эта область была бы 160000—177777, что могло бы привести к конфликтам как с системными регистрами БК, так и с КР1801ВП1-128 (регистры 177130 и 177132). Попутно следует заметить, что «висящий в воздухе» при подключении к БК-0010 вход 6 микросхемы D9.2 (КР559ИП2) воспринимается как заземленный (в отличие от серий К155 и К555).

Теперь перейдем к доработкам (на схеме они показаны утолщенными линиями). Прежде всего следует повысить помехозащищенность контроллера. Достичь этого очень просто — достаточно припаять керамический малогабаритный конденсатор емкостью 0.1 мкФ с обратной стороны платы непосредственно к выводам питания 21 и 42 БИС КР1801ВП1-128. С той же целью полезно подать сигнал SYNC на БИС не непосредственно с шины компьютера, а через буфер, т.е. с выхода 3 D9.1 (К559ИП2), предварительно разорвав существующую цепь.

Далее, раз уж в схеме присутствует дешифратор D12 (К555ИД7), грешно было бы не воспользоваться им и не обеспечить возможность подключения третьего дисковода (цепь от ножки 15 этой микросхемы, пробуферизованная свободным элементом микросхемы D5 К155ЛП9). Такое подключение не поддерживается старым драйвером (тем, что находится в КР1801РЕ2-326), зато допускается всеми новыми программами для работы с дисководом. Подключение же дополнительного дисковода с сохранением совместимости с «326» прошивкой потребовало бы введения изрядного числа «порезов» на печатной плате. (Следует заметить, что для первых двух дисководов при этом ничего менять не нужно.)

Полезно (но необязательно) подключить вход 6 D12 не к уровню логической единицы, а к выходу DS3 БИС D1. Это облегчит дальнейшее расширение системы.

Те, кто имеет отдельные блоки питания для БК и дисковода, не могли не столкнуться со следующей неприятной особенностью контроллера. Если выключить питание БК при включенном питании дисководов, все они разом клацают головками и затирают часть информации на недальновидно оставленных в них дискетах. Как показали эксперименты, в этом «повинна» микросхема К155ЛП9, замыкающая свой выход на землю при выключении питания (вместо того чтобы делать его «отключенным»). Такая же ситуация наблюдается и в «зеленоградском» контроллере (в КМД УКНЦ такой проблемы не существует). Чтобы раз и навсегда покончить с подобными «подарками судьбы», следует заменить буферы выбора дисководов и разрешения записи на аналогичные из установленной дополнительно, как показано на схеме, микросхемы КР531АП2 (К531АП2П). (Такими микросхемами в изо-

(Окончание на стр. 47)

СПРАВОЧНЫЙ ЛИСТОК

Параллельный программируемый регистр ввода-вывода (разъем «УП» — вилка СНП58-64/94Х9 В-23-В)

Номер контакта разъема	Обозначение сигнала	Примечание
	Регистр вывода	Управляются программно;
А16	ВД00	$U_{IL} \leq 0{,}4$ В;
А13	ВД01	$U_{IH} = (2{,}4—5{,}0)$ В;
В12	ВД02	$U_{OH} = (3{,}65—5{,}0)$ В;
В10	ВД03	$U_{OL} \leq 0{,}5$ В;
В5	ВД04	$J_{IL} \leq -2{,}75$ мА;
В7	ВД05	$J_{IH} \leq 0{,}01$ мА;
В6	ВД06	$J_{OH} \leq 1{,}0$ мА;
А7	ВД07	$J_{OL} \leq 15{,}0$ мА
А28	ВД08	
В28	ВД09	
А27	ВД10	
В27	ВД11	
А26	ВД12	
В26	ВД13	
А25	ВД14	
В25	ВД15	
	Регистр ввода	Управляются программно;
В24	ВВ00	$U_{IL} \leq 0{,}4$ В;
А24	ВВ01	$U_{IH} \geq 2{,}4$ В;
В23	ВВ02	$U_{OH} \geq 3{,}65$ В;
В17	ВВ03	$U_{OL} \leq 0{,}5$ В;
В20	ВВ04	$J_{IL} \leq 2{,}75$ мА;
А20	ВВ05	$J_{IH} \leq 0{,}01$ мА;
В22	ВВ06	$J_{OH} \leq 1{,}0$ мА;
А23	ВВ07	$J_{OL} \leq 15{,}0$мА
В31	ВВ08	
А31	ВВ09	
В32	ВВ10	
А32	ВВ11	
В30	ВВ12	
А29	ВВ13	
В29	ВВ14	
А30	ВВ15	
В1	ПРТ	
А1	СБРОС	
	Питание	
А8	+5 В	Jmax=150 мА
В8	То же	
А9	»	
В9	»	
А11	Общий	
В11	»	
А18	»	
В18	»	
А19	»	
В19	»	

Соответствие контактов разъема ХТ3 сигналам системной магистрали

Номер монтажа разъема ХТ3	Обозначение сигиала в магистрали	Обозначение в соответствии с условным обозначением процессора	Наименование сигнала магистрали
A1	ОСТ Н	IRQ1	Останов
A2	Общий		Общий
A3	»		»
A4	+5 В (контроль)		Контроль источника питания +5 В
A5	ПРТ Н	IRQ2	Требование прерывания
A23	ВВОД Н	DIN	Ввод данных
A25	ДА13 Н	AD13	Линия адреса данных
A26	ДА11 Н	AD11	» » »
A27	ДА09 Н	AD9	» » »
A28	ДА05 Н	AD5	» » »
A31	ДА00 Н	AD0	» » »
B2	Общий		Общий
B3	»		»
B4	ППР1 Н	IAK1	Входной сигнал предоставления прерывания
B4	ППР1 Н	IAK0	Выходной сигнал предоставления прерывания
B5	ТПР Н	VIRQ	Требование прерывания
B7	ДА15 Н	AD15	Линия адреса данных
B11	БАЙТ Н	WTBT	Вывод байта
B19	СБРОС Н	INIT	Первоначальная установка магистрали
B20	СИП Н	RPLY	Сигнал синхронизации пассивного устройства (ответ)
B21	ВЫВОД Н	DOUT	Вывод данных
B22	СИА Н	SYNC	Сигнал синхронизации активного устройства
B23	ДА14 Н	AD14	Линия адреса данных
B24	ДА12 Н	AD12	» » »
B25	ДА10 Н	AD10	» » »
B26	ДА08 Н	AD8	» » »
B27	ДА06 Н	AD6	» » »
B28	ДА04 Н	AD4	» » »
B29	ДА02 Н	AD2	» » »
B30	ДА03 Н	AD3	» » »
B31	ДА01 Н	AD1	» » »
B32	ДА07 Н	AD7	» » »

билии «усыпленные» платы ДВК, так что найти их будет не так сложно.)

Последняя доработка — расширение области адресов ПЗУ D11 с диапазона 160000—167777 до 160000—173777, производимое подачей на вход 10 D9.3 не заземления, а сигнала с линии AD11 входного разъема. Для ПЗУ КР1801РЕ2-326 (как и -253) это не актуально, ибо находящийся там драйвер вполне умещается в выделенном адресном пространстве. Однако существуют альтернативные драйверы (зашитые в микросхемы КР1801РР1 или К573РФ3 и устанавливаемые в панельку, припаянную вместо стандартного ПЗУ), для которых требуется подобная доработка.

В заключение следует сказать об одной периодически встречающейся на дисководах (чаще всего болгарского или армяиского производства) особенности. Бывает, что сигнал индекса имеет завышенную по отношению к стандарту длительность (что связано с плохой схемотехникой или настройкой). В этом случае при подключении к контроллеру БК или УКНЦ или к подавляющему большинству контроллеров для ZX-Spectrum (Sinclair) наблюдается любопытный эффект: запись и чтение выполняются великолепно, но вот форматирование дискет не удается. Для тех, кто уже успел с этим намучиться или занят подобной «борьбой», сообщаю: достаточно поставить одновибратор на, например, К555АГ3 для укорочения данного импульса (в схеме контроллера БК это следовало бы сделать в разрыве цепи «выход 10 D8.5 — вход 32 D1»), и эту «головную боль» как рукой снимет.

Литература

1. Персональный компьютер БК-0010 — БК-0011М. 1993. №1.

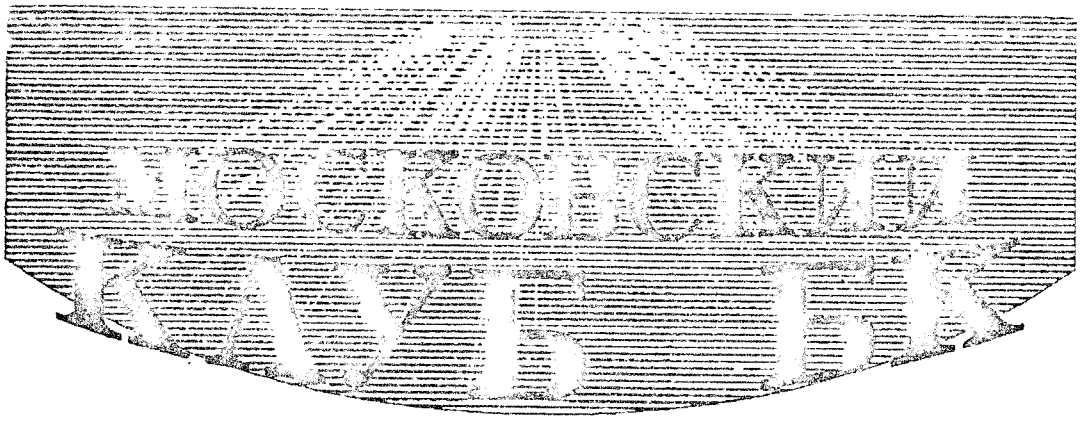
2. *Шевкопляс Б.В.* Микропроцессорные структуры. Инженерные решения: Справочник. М.: Радио и связь, 1990.

3. *Борис Ф. Фролкин.* Контроллер дисковода для БК-0010/11 М.: Самиздат, 1993.

*** Внимание! Опечатка ***

В статье Р. Аскерова «О некоторых способах генерации шрифтов для БК» («Персональный компьютер БК-0010 — БК-0011М» №1 за 1994 г.) ассемблерный листинг на с. 55 содержит ошибки. Ниже приведен исправленный фрагмент:

```
        .....
MET:    MOVB (R1)+,(R2)      ; Вывести очередную строчку
        BICB R3,(R2)+        ; и сменить цвет. R3:
        MOVB (R1)+,(R2)      ; 0 - красный,
        BICB R3,(R2)+        ; 52525 - зеленый,
        .....
```



А. Г. Прудковский,
г. Москва

Операционная система NORD – плюсы и минусы

В настоящее время существуют две модификации дисковой операционной системы NORD: NORD-БК11 v2.16 — для БК-0011(М) и NORD-БК10 v2.16 — для БК-0010(.01) с дополнительным ОЗУ на 16 Кб (еще большие возможности предоставляются, если компьютер оснащен дополнительным ОЗУ на 32 Кб с конфигурацией, предложенной П.В. Петровым). Готовится также новая версия NORD-3.0.

Система NORD имеет удобный NORTON-подобный интерфейс с 20 ключами. Работа с ней возможна от клавиатуры, мышки или джойстика. Поддерживается многоуровневая структура подкаталогов и логических дисков. На БК-0011(М) может использоваться виртуальный диск на 64 Кб.

NORD надежно работает с четырьмя типами дискет и дисководов (80- и 40-дорожечными, двусторонними и односторонними), автоматически распознавая тип установленной в данный момент дискеты. Две копии каталога позволяют уверенно работать даже на низкокачественных дисководах.

Вышеуказанные достоинства системы, а также простота общения с ней обсуждаются в большинстве статей, однако мне как создателю NORDa хотелось бы рассказать об этой системе более подробно.

АРХИТЕКТУРА СИСТЕМЫ NORD

При создании системы первым был поставлен вопрос выбора формата каталога диска. Можно было остановиться на одном из общепринятых стандартов — RT11 или IBM PC, но это влекло за собой достаточно неприятные последствия, а именно необходимость воссоздавать на малой машине громоздкую структуру, пригодную только при большом объеме памяти. Обратите внимание на медленную работу систем, пользующихся форматом RT11, а также на неимоверные трудности, которые испытывают создатели ANDOS, когда пытаются «научить» свою систему работать с подкаталогами (ведь на IBM PC длина подкаталога может быть больше любой рабочей области, которую способна высвободить система на малом компьютере типа БК). По этой причине был выбран формат MicroDOS, который легче стандартизировать под малый компьютер.

Кратко обсудим некоторые технические подробности по системе NORD.

- Каталог дискеты в формате NORD состоит из 12(восьм.) секторов с номерами 0-11 (начало нулевой дорожки диска), его копия хранится в секторах 12-23 (на односторонних дисках копия отсутствует). Логический диск имеет одну копию каталога, записанную в области данных дискеты (номер его начального сектора хранится в 26-й ячейке «основного» каталога, благодаря чему возможно копирование логического диска как целого).

- В ОЗУ БК-0010 каталог размещается начиная с адреса 146000, на БК-0011(М) — с 66000. (Это область скрытой страницы, до которой можно добраться, выполнив команду MOV @#100076,@#177716. Однако часто это делать не рекомендуется, так как возможно зависание компьютера. После работы на скрытой странице необходимо вернуться в исходное

состояние с помощью команды MOV @#117776,@#177716).

- Стандартные ячейки каталога (адреса восьмеричные):

26 — ноль для обычного каталога либо номер сектора для каталога логического диска (это значение используется для автоматической обработки последнего, когда он скопирован как целое на другое физическое устройство);

30 — общее количество имен в каталоге (включая удаленные файлы и имена подкаталогов);

32 — первый пустой сектор на диске;

34 — ячейка защиты диска от записи (если она содержит код 123456, то система NORD считает диск защищенным);

42 — тип диска:

0 — 80 дорожек, двусторонний,
1 — 40 дорожек, двусторонний,
2 — 80 дорожек, односторонний,
3 — 40 дорожек, односторонний;

44 — количество и состояние копий каталогов на диске:

0,1 — одна копия каталога, начальный сектор 0,
2 — одна копия, 12-й сектор,
3 — две копии, 0 и 12-й секторы;

46 — признак заполнения виртуального диска E:. Если виртуальный диск заполнен более чем наполовину, то ячейка 46 в его каталоге обнуляется. Это сигнал, что системная страница БК-0011(М) испорчена и файлы, написанные для этого компьютера, запускать нельзя. В противном случае система NORD может запускать файлы в собственном мониторе БК-0011(М), если в каталоге они помечены расширением .11 или .11M. Например, таким способом система NORD поддерживает дисковый вариант БЕЙСИКа из ПЗУ БК-0011(М);

100 — байт идентификации каталога (равен нулю);

400 — ячейка идентификации каталога (равна 123456);

466 — объем диска в секторах;

500 — начало собственно каталога, на каждую запись отводится 30(восьм.) байт (далее указано смещение от начала записи):

0 — байт атрибута файла:

0 — обычный,
1 — защищенный,
2 — плохой,
200—377 — удаленный (для имен подкаталогов — их номера);

1 — байт-номер каталога, которому принадлежит файл или подкаталог;

2—16 — имя файла или подкаталога (имя подкаталога начинается с кода 177);

20 — начальный сектор файла;

22 — длина файла в секторах;

24 — начальный адрес файла;

26 — длина файла в байтах (по модулю 200000).

Еще одним достоинством системы NORD является то, что она оставляет свободной память по адресам 120000—137010, куда пользователи могут загружать свои программы. Так, например, к системе NORD без особого труда удалось приспособить стандартный ФОКАЛ, туда же можно поместить редактор или отладчик. Большинство же других систем, например ANDOS или MKDOS, используют эту область памяти для собственных нужд.

СОВМЕСТИМОСТЬ СИСТЕМЫ NORD С КОМПЬЮТЕРАМИ IbM PC

Существуют значительные технические сложности при чтении и особенно записи на БК дисков, отформатированных на IBM-совместимых компьютерах, поэтому для работы с ними был создан некий «полуБК-шный» стандарт ANDOS. (Указанные сложности сегодня успешно преодолены. Программы ALTEK и INTERSERVER позволяют нормально работать на БК-0010(.01) и БК-0011(М) с дискетами, отформатированными на IBM, однако на БК-0010(.01) для этого требуется замена ПЗУ контроллера дисковода. — *Прим. ред.*) Но формат ANDOS не универсален, он не совместим с большинством отечественных ПЭВМ («ПОИСК», «МС-1502» и т.д.), а также с западными компьютерами, оснащенными DOS 3.0 или дисководами на 360 Кб. Для всех этих случаев мной создан формат «POISK». На отечественных компьютерах, а также на западных, оснащенных дисководами на 360 Кб, дискеты в формате «POISK» даже не требуют включения драйвера 800.COM. Из моих слов, однако, не следует, что этот формат более универсальнен, чем ANDOS. Так, диски на 80 дорожек, размеченные в формате «POISK», обычно не читаются на западных IBM-совместимых компьютерах, оснащенных системой DOS 4.0. Для владельцев же односторонних дисководов можно использовать аналогичный формат «SINGL».

Таким образом, система NORD имеет утилиты для совместимости с компьютерами серии IBM PC в трех форматах по выбору пользователя. Правда, работа с ними возможна только в корневом каталоге диска, зато задачи копирования, перекодировки,

редактирования и выдачи на принтер текстовых файлов ЛЮБОЙ длины в системе NORD решены полностью. А как обстоит дело с копированием БОЛЬШИХ файлов в системе ANDOS?

Остроумно решена и задача копирования одноименных файлов. На дисках для IBM PC, записываемых в системе NORD, одноименные файлы получают расширения вида .000, .001 и т.д., причем чем раньше создан файл, тем больший номер он получает. Последняя же версия сохраняет первоначальное расширение.

КОМАНДНЫЙ ПРОЦЕССОР СИСТЕМЫ NORD

Система NORD изначально создавалась для управления длительными оверлейными вычислительными процессами. Для этой цели она имеет встроенный язык, подобный реализации .BAT-файлов в MS-DOS на IBM PC, что позволяет подгружать в память компьютера оверлейные кодовые подпрограммы с расширением .COM или запускать любые другие программы (в последнем случае без возврата в вызывающий файл).

Передача числовой информации производится через двухбайтовую переменную @, значение которой при обращении к кодовой подпрограмме содержится в регистре R0.

Передача символьной информации выполняется через текстовую переменную ¤. При обращении к кодовой подпрограмме в регистре R1 находится адрес этой переменной (обычно 120004), а в R2 — длина текста (не более 177 восьм.).

Переменная K (регистр R3) сигнализирует о той или иной ошибке процесса, а L (регистр R4) — о ее типе.

Регистр R5 содержит адрес первой свободной ячейки, начиная с которой оверлейная программа может организовать свою рабочую область.

Командный язык системы NORD является полноценным языком высокого уровня с метками, целочисленной арифметикой, операторами безусловной и условной передачи управления и т.д. Проиллюстрируем его возможности на примере создания пользовательского меню (выбор запускаемого файла на системном диске):

РАБОТА С ДИСКОВОДОМ В СИСТЕМЕ NORD

Работа с дисководом в системе NORD также рассчитана на функционирование по возможности без участия человека. В связи с этим пришлось отказаться от каких-либо запросов к пользователю и при этом

```
@ec;REM оператор @ECHO - отмена трассировки;
ALL;REM включение режима ALLDIR - доступа во все каталоги;
IF @#56 :000 ;00 :000 ;REM проверка, включен ли курсор?
:00;chrn #232;REM выключить курсор;
:000
POS 0 7;REM установка курсора в позицию X=0, Y=7;
got :M1;REM переход на метку :M1
:M1ANS
pos 15 15;REM установка курсора в позицию X=15, Y=15;
chrn #232;REM включение курсора;
:010;inp A;REM ввод числа в ячейку A;
case A :1 :2 :3 :4 :5 ;REM переход на метку по значению A;
goto :010;
:1;CLS;call "M19ADN"          ;pau;out;REM  запуск файлов
:2;CLS;call "DEBU10DN"        ;pau;out;REM  в соответствии
:3;CLS;call "SQ18.035EAGLE"   ;pau;out;REM  с выбором
:4;CLS;call "F:START-FOC  "   ;pau;cut;REM  в меню;
:5;END;REM возврат в вызывающую программу
:M1;@TYPE +
```

```
        User menu:
 1  Ассемблер/редактор
 2  Отладчик
 3  Сквизирование диска
 4  Фокал
 5  Выход из user menu
```

```
+; GOT :M1ANS;REM печать меню и переход на метку :M1ANS;
```

максимально обеспечить защиту информации на диске, что и привело к запрету стирания одноименных файлов при записи (при чтении всегда доступен вариант, записанный последним). Это свойство системы многим не нравится, но, к сожалению, для автономной работы больших программ оно необходимо. Впрочем, нетрудно перейти и на другой режим записи, для чего в комплект входит специальная утилита DEL-11-32.COM, которая переводит систему в режим удаления одноименных файлов. Ту же операцию может производить и малый сквизер SQM-NORD-v1, созданный в конце 1993 г.

Для работы с дисководом система NORD использует два прерывания: EMT 36 и EMT 54, устанавливая некоторый стандарт пользования ими.

Функция EMT 36 в системе NORD во многом похожа на аналогичную функцию монитора БК-0010 для работы с магнитофоном:

- используется блок параметров, на который указывает регистр R1 (не путайте его с блоком параметров дисковода, который хранится внутри системы по адресам 137700 — 137777);
- формат блока параметров совпадает с «магнитофонным» для БК-0010. Обычно он создается по адресу 320 (восьм.).

Однако в системе NORD прерывание EMT 36 имеет больше возможностей, чем его магнитофонный аналог:

- кроме известных функций чтения (код 3), записи (2) и фиктивного чтения (4) появились две новые: создание файла (6) и удаление/защита (7);
- при задании имени файла можно указать устройство, с которым производится работа: «A:имя» — работа с дисководом A, «B:имя» — с дисководом B, «E:имя» — с виртуальным диском, «F:имя»...«Z:имя» — с логическими дисками, «M:имя» — с магнитофоном;
- если необходимо войти в подкаталог, то достаточно в качестве имени задать «/имя» и провести операцию чтения. Если надо выйти из подкаталога, то следует задать обратную дробную черту. Можно создать и новый подкаталог, для чего нужно провести операцию записи с именем «/имя»;
- тип операции с диском определяется первыми тремя битами первого слова блока параметров, остальные биты несут следующую дополнительную информацию:

3 бит (маска 10) — включение режима ALLDIR, т.е. чтение, тестирование и удаление файлов из всех подкаталогов, а запись и создание — только в корневом каталоге;

4 бит (20) — поиск файла по первым нескольким буквам имени;

5 бит (40) — после операции не записывается новая версия каталога (этот режим необходим для ускорения групповых операций с файлами);

6 бит (100) — использование содержимого ячейки 264: при чтении и удалении — для нахождения адреса строки каталога, при записи — для нахождения адреса файла в ОЗУ (в каталог при этом, как и раньше, заносится адрес из блока параметров);

7 бит (200) — «не читать каталог», используется при групповых операциях с файлами;

15 бит (100000) — не выключать мотор дисковода.

В новой версии NORD-3.0 добавлены две команды:

маска 106 — изменение атрибутов файла (адреса и длины), при этом адрес соответствующей строки каталога должен быть указан в ячейке 264;

команда 200 — запись на диск текущего каталога (в отличие от использования маски 200 здесь все биты, кроме седьмого, должны быть нулевыми);

- при создании файлов (режим 6) используется следующая информация (указаны смещения от начала блока параметров):

  2 — адрес файла в каталоге,

  4 — длина (по модулю 200000),

  6—24 — имя файла,

  30 — длина в секторах (можно задать значение в отрицательном виде, это гарантирует, что файл будет создан в конце каталога, а не на месте удаленного файла).

Если операция завершена без ошибки, то номер первого сектора созданного файла помещается в ячейку 250, та же информация заносится в нее при чтении и записи уже имеющегося файла;

- при удалении/защите файлов (режим 7) используется информация в ячейке со смещением 2:

  0 — удаление,

  1 — защита файла от удаления,

  2 — отметка файла как «плохого» (.BAD).

Если удаляется подкаталог, то все файлы, содержащиеся в нем, переходят в подкаталог предыдущего уровня;

- при тестировании (режим 4) выдается следующая дополнительная информация (указан адрес в ОЗУ):

236 — номер начального сектора на диске для текущего каталога,

240 — адрес строки каталога в ОЗУ, соответствующей найденному файлу (в новой версии системы эта ячейка может быть использована не только при тестировании, но и в других случаях),

242 — адрес в ОЗУ конца каталога,

244 — номер первого свободного сектора на текущем устройстве (ячейкой 244 очень удобно пользоваться для определения, найден ли вообще на диске каталог; если он не найден или не в том формате, то содержимое этой ячейки будет равно нулю),

246 — общее количество секторов на текущем устройстве, а также следующие данные (указано смещение от начала блока параметров):

2 — номер начального сектора найденного файла,

4 — длина в секторах,

26 — начальный адрес файла,

30 — длина в байтах (по модулю 200000).

В новой версии системы ячейки 244 и 246 содержат границы (начало и конец) максимального свободного участка на диске. Кроме того, добавлены ячейки 226 и 230, в которых хранится количество свободных секторов и общий объем диска соответственно (для логического диска в ячейке 230 содержится значение, на 1 большее номера его последнего сектора);

- для защиты области системного монитора БК-0011(М) от затирания функция EMT 36 запрещает чтение файлов с переходом от положительных к отрицательным адресам (т.е. через адрес 100000);
- функция EMT 36 поддерживает следующие условные имена:

DIR — распечатка на экране текущего каталога,

\ — выход из подкаталога,

BACK — возврат к предыдущей конфигурации панелей дисковода (в новой версии — возврат к начальной конфигурации, которая была установлена перед запуском программы из NORDa),

SWAP — обмен панелей дисковода (здесь следует пояснить, что система NORD считает левую панель рабочей, а правую — вспомогательной; при запуске файлов с правой панели она автоматически делает «SWAP», что вызывает справедливые нарекания пользователей);

- все регистры функция EMT 36 сохраняет в стеке;
- результат работы EMT 36 отображается в байте с адресом 1(R1), где R1 — адрес начала блока параметров. Этот байт может быть равен 0 (операция прошла без ошибок), 4 (при ошибке или после чтения DIR), 5 (при безошибочном входе в подкаталог либо при операциях SWAP или BACK);
- имеется возможность задания постоянного режима работы функции EMT 36. Для этой цели в ячейке 137300 для БК-0011(М) или 140236 для БК-0010 «пустую» команду BIS #0,(R1) нужно изменить так, чтобы она задавала постоянный режим работы прерывания. Например, BIS #10,(R1) заставит функцию EMT 36 работать только в режиме ALLDIR, а BIC #200,(R1) — производить чтение каталога диска даже в том случае, когда пользовательская программа от этого отказывается.

Функция EMT 54 предназначена для прямого обращения к диску, при этом должна быть задана следующая информация:

R2 — адрес массива для чтения или записи;

R1 — длина в словах (положительная — для чтения, отрицательная — для записи);

R0 — номер начального сектора.

Функция EMT 54 сохраняет только регистр R5, а при ошибке выставляет бит C в единицу и устанавливает стандартный код ошибки в ячейке 52.

Опишем теперь, как можно организовать операции открытия и закрытия файлов в системе NORD.

ВАРИАНТ 1 — используется в случае открытия одного файла и при условии, что других операций записи до его закрытия проводиться не будет:

1) производим операцию фиктивного чтения с любым именем и определяем в ячейке 244 первый пустой сектор на диске, а в 246 — общее количество секторов. Теперь (с помощью функции EMT 54) мы имеем доступ ко всем секторам в пределах, заданных в ячейках 244 — 246;

2) после записи информации в пустые сектора определяем количество занятых из них, и, если оно отлично от нуля, помещаем его с обратным знаком в ячейку 30(R1) (с адресом 350, если блок параметров создается по адресу 320);

3) заносим адрес и длину файла в ячейки 2(R1) и 4(R1), а имя — в ячейки 6(R1)-24(R1) и проводим операцию создания файла (отрицательное число в ячейке 30(R1) обеспечивает формирование записи о нем в

конце каталога, а не вместо удаленного файла).

ВАРИАНТ 2 — используется в более сложных случаях:

1) выбираем имя будущего файла и некоторую его минимальную длину D;

2) добавляем к имени расширение .000 и создаем первый файл с длиной, равной D;

3) из ячейки 250 узнаем начальный сектор созданного файла и с помощью функции EMT 54 заполняем его информацией по мере ее поступления в компьютер;

4) если объема файла не хватает, то создаем новый с тем же именем и с расширением .001, и т.д;

5) если окажется, что информация отсутствует и что мы зря открыли очередной файл, то его следует удалить с помощью команды удаления (7).

ВАРИАНТ 3 — новая версия системы NORD-3.0 позволяет более корректно производить операции открытия и закрытия файлов. Поскольку в ней производится вычисление объема свободного места на диске, пользователь имеет возможность, протестировав диск, определить максимальную длину создаваемого файла.

Открыв файл и записав туда информацию, с помощью новой операции (10б) можно изменить его атрибуты так, чтобы они соответствовали реальным.

НЕДОСТАТКИ СУЩЕСТВУЮЩЕЙ ВЕРСИИ СИСТЕМЫ NORD

1. Ошибки в утилитах связи с IBM-совместимыми компьютерами, проявляющиеся при большом количестве файлов на диске. Эта ошибка исправлена в версиях PC-ANDOS6, PC-POISK6, PC-SINGL6.

2. Произвольная перестановка панелей NORTON-таблиц системы.

3. Несохранение позиции курсора в списке имен файлов при смене активной панели или файловых операциях.

4. Появление «мусора» на экране при прокрутке в NORTON-таблицах имен файлов, содержащих букву g. Ошибка исправлена в последних версиях системы и утилит PC-ANDOS6, PC-POISK6, PC-SINGL6.

В новой версии NORD-3.0 все эти недостатки исправлены.

По вопросам приобретения ОС NORD v2.16 обращайтесь в редакцию по телефону: 151-19-40 E-Mail: mail@infoobr.msk.su

В. З. Манвелян,
г. Москва

УНИВЕРСАЛЬНЫЙ ПРОГРАММАТОР МИКРОСХЕМ ППЗУ ДЛЯ БК

Бытовой компьютер «Электроника БК-0010(.01)», несмотря на ограниченные ресурсы, может использоваться не только как игровая приставка к телевизору, но и как универсальное инструментальное средство, например при построении различных устройств управления, измерительных приборов, автоматизированных систем сбора и предварительной обработки данных. Это возможно благодаря низкой стоимости, значительной надежности и быстродействию для такого класса машин. Большое распространение в последнее время дисководов и простота их подключения к БК значительно расширяют

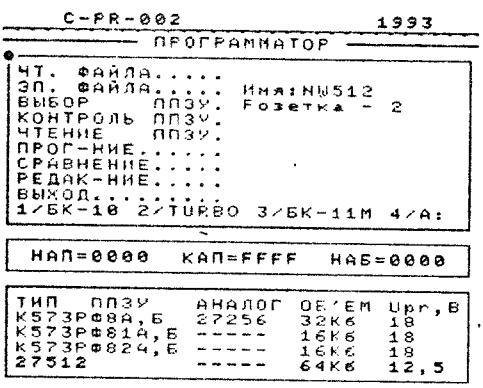


Рис.

сферу возможного использования этого бытового компьютера.

Описываемое в данной статье устройство выполнено в виде приставки к БК и предназначено для программирования основных типов ППЗУ (перепрограммируемых постоянных запоминающих устройств) с ультрафиолетовым (УФ) и электрическим стиранием информации. Эти микросхемы в настоящее время получили широкое распространение благодаря доступности, большой информационной емкости и достаточному для большинства применений быстродействию. Они часто используются в современной вычислительной и бытовой технике для хранения программ и кодов знакогенераторов для дисплеев и принтеров.

Процесс записи в эти микросхемы необходимой информации называется программированием ППЗУ, а устройство записи — программатором. Внутреннее строение и принципы программирования микросхем ППЗУ достаточно подробно описаны в цикле статей «Учись работать с ППЗУ» [1].

Простейшие программаторы обычно не содержат буферной памяти и обеспечивают копирование информации из «эталонного» ППЗУ, что удобно при массовом производстве. В случае же разработки оригинальных конструкций необходим программатор, обладающий значительно большими возможностями.

Особенностью предлагаемого устройства является удобный пользовательский интерфейс и наличие виртуальной буферной памяти, позволяющей программировать ППЗУ большой емкости за один прием. Перечень программируемых микросхем ППЗУ приведен в таблице.

Программное обеспечение, созданное для работы с программатором, выполняет следующие функции: ЧТЕНИЕ/ЗАПИСЬ файла на диск, ВЫБОР типа ППЗУ, КОНТРОЛЬ ППЗУ, ЧТЕНИЕ содержимого ППЗУ в буфер, ПРОГРАММИРОВАНИЕ, СТИРАНИЕ информации, СРАВНЕНИЕ содержимого ППЗУ и буфера, РЕДАКТИРОВАНИЕ информации в буфере, РАСПЕЧАТКА содержимого буфера, РАЗДЕЛЕНИЕ файла на два, содержащих младшие и старшие байты, а также РАБОТА С ОТДЕЛЬНЫМИ ЯЧЕЙКАМИ ППЗУ и НАСТРОЙКА на тактовую частоту процессора.

После запуска программы на экране монитора формируется три окна. Левая половина первого окна содержит список команд, доступных пользователю. Правая — служит для сообщений о результатах их выполнения. Второе окно предназначено для выдачи запросов пользователю при выполнении команд и для индикации содержимого буфера или ППЗУ. Третье — содержит перечень ППЗУ, обозначение их функциональных аналогов, информацию о емкости и напряжении программирования.

Выбор команды осуществляется при помощи клавиш «стрелка вверх» и «стрелка вниз», а выполнение начинается после нажатия на «ввод» или «пробел» и сопровождается звуковым сигналом. Выбранная команда выделяется на экране повышенной яркостью. Отменить ее можно клавишей «КТ».

Несколько слов о буферах памяти. Их два. Программа создает буфер требуемых размеров на диске, а необходимая в данный момент информация автоматически подгружается в промежуточный буфер в ОЗУ. Емкость буфера в ОЗУ равна 8 Кб, а на диске устанавливается равной емкости выбранной микросхемы ППЗУ. Учитывая разнообразие операционных систем для БК, а также отсутствие в них общего стандарта для средств создания файлов, их удаления, резервирования места на диске и в каталоге под файл требуемой длины, а также отсутствие ряда других системных средств, необходимых пользовательской программе для работы с диском, программатор создает дисковый буфер в обход каталога. Буфер может быть создан на любом приводе, номер которого отображается в последней строке первого окна. Изменить его можно клавишей «4», при этом номер привода, установленный операционной системой для чтения/записи файлов, не изменяется.

Рассмотрим основные команды программатора.

- ЧТЕНИЕ/ЗАПИСЬ файла. Файл читается или записывается только из буфера. Максимальная длина его имени равна 10 символам. Имя задается без расширения, оно добавляется автоматически при выполне-

Таблица

Тип ППЗУ	Аналог	Объем, кбит	U_{pr}, В
С ультрафиолетовым стиранием			
К573РФ2, К573РФ5	2716	16 (2×8)	25
К573РФ21, РФ22	б/а	8 (1×8)	25
К573РФ3, РФ31, РФ32, РФ33, РФ34	б/а	64 (4×16)	24
К573РФ4А,Б, К573РФ6А,Б	2764	64 (8×8)	21
К573РФ41А,Б, РФ42А,Б		32 (4×8)	21
КС1626РФ1А,Б	27С64	64 (8×8)	12,5
КС1626РФ11А,Б, РФ12А,Б		32 (4×8)	12,5
К573РФ7, К573РФ8А,Б, (27С256)	27256	256 (32×8)	18
К573РФ81А,Б, РФ82А,Б	б/а	128 (16×8)	18
27128, 27С128		128 (16×8)	12,5
27512, 27С512		512 (64×8)	12,5
С электронным стиранием			
К558РР3	б/а	64 (8×8)	24
К1801РР1	б/а	64 (4×16)	24

Принятые обозначения: U_{pr} - напряжение программирования, б/а - без аналога. В графе «Объем» в скобках указана организация микросхемы.

нии операций чтения/записи. Максимальный объем буфера равен 8 Кб, поэтому при записи создается необходимое количество файлов длиной по 8 Кб, в зависимости от емкости выбранной ППЗУ. Так, для ППЗУ типа 27256 информационная емкость которой равна 32 Кб (256 кбит), будет создано четыре файла. Отметим, что эти ограничения также связаны с отсутствием в операционных системах средств, позволяющих создавать файлы произвольной длины.В случае ошибки при работе с диском выдается сообщение «Ошибка ЧТ/ЗП». Для выхода из команды необходимо нажать клавишу «КТ», для повтора — «ввод».

- ВЫБОР ППЗУ. По этой команде активным становится третье окно, содержащее перечень доступных для программирования микросхем. Выбор ППЗУ осуществляется так же, как и выбор команд. После этого активным становится первое окно, а напротив команды ВЫБОР ППЗУ выводится номер розетки для установки микросхемы.
- КОНТРОЛЬ ППЗУ. Микросхемы ППЗУ с ультрафиолетовым стиранием поставляются незапрограммированными и имеют на выходах высокий уровень напряжения, что соответствует исходному состоянию. Такое же состояние должно быть и после стирания [2]. По команде КОНТРОЛЬ осуществляется проверка микросхемы в режиме считывания на соответствие всех выходов по всем адресам высокому уровню напряжения. В случае несоответствия выдается сообщение «Ошибка», а во втором окне выводится адрес и содержимое ошибочной ячейки ППЗУ. При успешном завершении выдается сообщение «Годен». Выход из команды — клавиша «КТ», продолжение выполнения — «ввод».
- ЧТЕНИЕ ППЗУ. По этой команде содержимое ППЗУ записывается в буфер на диске. Перед выполнением выдается запрос на ввод начального (НАП) и конечного адреса ППЗУ (КАП), а также начального адреса буфера (НАБ). Формат запроса: **НАП=XXXX КАП=YYYY НАБ=ZZZZ**, где XXXX — начальный адрес ППЗУ (по умолчанию 0000), YYYY — конечный адрес ППЗУ (по умолчанию равен конечному адресу для выбранной микросхемы), ZZZZ — начальный адрес буфера (по умолчанию 0000)

- Адреса вводятся в шестнадцатеричной системе счисления. Их задание позволяет считывать произвольный фрагмент содержимого ячеек ППЗУ и записывать его в буфер с указанного адреса, не изменяя при этом остального содержимого буфера.

Для сохранения диапазона адресов, указанного по умолчанию, необходимо нажать на клавишу «ввод». Клавиша «КТ» используется для отмены команды или для перехода к вводу предыдущего адреса. Корректность ввода адресов контролируется программой. Выполнение команды завершается сообщением «Данные в буфере».

- ПРОГРАММИРОВАНИЕ. В программаторе использован адаптивный алгоритм записи информации в ППЗУ, рекомендованный фирмой INTEL [1], который позволяет значительно ускорить процесс записи без ухудшения качества. В процессе программирования осуществляется контроль записываемой информации. В случае ошибки выдается сообщение «Ошибка», выводятся адрес и данные ячейки ППЗУ и данные буфера. Клавишей «ввод» можно повторить запись в ту же ячейку, а «П» — продолжает программирование, игнорируя ошибку. Выход из режима — по клавише «КТ».

Как и в предыдущей команде, перед программированием предлагается ввести адреса НАП, КАП и НАБ, после чего запрашивается дополнительное подтверждение на выполнение команды. Кроме того, для ППЗУ с электрическим стиранием выдается запрос на стирание содержимого микросхемы, а также на ввод и программирование кода для ППЗУ К1801РР1.

Кроме того, следует отметить, что программа анализирует содержимое текущей ячейки буфера и если байт содержит исходную информацию (для выбранного типа ППЗУ), то он не программируется. Это, с одной стороны, уменьшает затраты времени, а с другой — позволяет маскировать отдельные ячейки ППЗУ в процессе записи.

При успешном программировании выдается сообщение «Годен».

- СРАВНЕНИЕ. Эта команда предназначена для сравнения содержимого буфера и ППЗУ в указанном диапазоне адресов. При несовпадении данных выводятся сообщение об ошибке, адрес, данные ППЗУ и буфера. Для продолжения команды необходимо нажать «ввод», а для выхода — «КТ».
- РЕДАКТОР. Встроенный в программу редактор позволяет оперативно корректировать информацию в буфере, а также выводить ее на печать. На экран данные выводятся блоками по 256 байт. В верхней части экрана содержится информация о номере текущего блока и адрес ячейки буфера, на которую указывает курсор.

Редактор выполняет следующие команды: «↑» — на одну строку вверх, «↓» — на одну строку вниз, «←» — влево на одно знакоместо, «→» — вправо на одно знакоместо, «N» — переход в начало буфера, «К» — в конец буфера, «;» — к следующему блоку, «—» — к предыдущему блоку, «СУ/Б» — переход по указанному номеру блока, «СУ/Р» — вывод на принтер содержимого буфера, «СУ/Е» — выход из редактора без записи информации в буфер на диске, «КТ» — выход с записью информации в буфер.

При распечатке буфера запрашивается номер конечного блока и режим печати: с паузой после распечатки очередного блока («Д») или без паузы («Н»). После выхода из редактора восстанавливаются все параметры и режимы работы, установленные перед входом в него.

Необходимость настройки на тактовую частоту процессора вызвана тем, что временная диаграмма в режиме программирования ППЗУ формируется программно и с изменением тактовой частоты меняется один из важных параметров — длительность импульса программирования. Поэтому в программаторе предусмотрена возможность коррекции длительности этого импульса в зависимости от используемой машины: БК-0010(.01) или БК-0011(М). Вся необходимая для настройки информация выводится в последнюю строку первого окна.

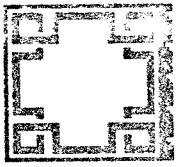
Конструктивно программатор выполнен в виде прямоугольной пластмассовой коробки из-под блока МСТД и подключается к порту пользователя.

Литература

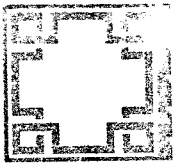
1. Микропроцессорные средства и системы. 1985. № 3. С. 71—88.

2. *Гордонов А.Ю., Бекин Н.В., Цыркин В.В. и др.* Большие интегральные схемы запоминающих устройств: Справочник. М.:Радио и связь, 1990.

По вопросам приобретения программатора можно обратиться в редакцию по телефону: 151-19-40 E-Mail: mail@infoobr.msk.su



HARD & SOFT



А. И. Винниченко,
г. Днепропетровск

DOSB10 v2.0: новая версия ОС для БК

Закончена разработка новой версии операционной системы DOSB10 (2.0). По заложенным в ней принципам, организации, структуре она вплотную приблизилась к профессиональным ОС, аналогов ей пока нет. Что же она собой представляет, чем отличается от предыдущей версии, от других систем?

Как и предыдущая версия (см. «Персональный компьютер БК-0010 — БК-0011М», №1 за 1993 г.), DOSB10 2.0 может работать и на БК-0010(.01) с дополнительным ОЗУ 8 или 16 Кб, и на БК-0011(М). Адаптация к конкретному типу компьютера при загрузке и дальнейшей работе осуществляется автоматически.

Главное преимущество новой версии — открытая структура. Ядро системы — файловая компонента с перехватчиком прерываний и командным монитором. Связь с внешними устройствами, количество которых в DOSB10 2.0 может быть неограниченно, осуществляется с помощью драйверов, системных таблиц и вызовов. Таким образом, операционная система и программы, работающие под ее управлением, полностью независимы от внешних устройств. Достаточно в названии файла указать имя устройства, и все операции ввода-вывода будут перенаправлены на него. Например, из любого текстового редактора (или другой прикладной программы) можно произвести вывод на принтер, плоттер и т.п.

Файловая система DOSB10 2.0 полностью совместима с предыдущими версиями (1.0 — 1.4). С помощью загружаемых драйверов-эмуляторов можно также работать с дискетами, записанными в других системах (RT—11, MicroDOS, ANDOS и т.д.).

Базовый вариант DOSB10 2.0 занимает 8 Кб расширенного ОЗУ и, подобно предыдущим версиям, поддерживает два типа устройств — дисковод (A, B) и магнитофон (T). Сохранена встроенная файловая оболочка, аналогичная той, к которой привыкли пользователи DOSB10 V1.3. Драйверы дополнительно обслуживаемых устройств, драйверы-эмуляторы других систем, а также различные расширения загружаются в свободную область дополнительного ОЗУ 16 Кб на БК-0010(.01) или в одну из скрытых страниц памяти БК-0011(М).

Одной из особенностей DOSB10 2.0, выгодно отличающей ее от предыдущих версий, является возможность использования «перекрытий». Встроенную файловую оболочку, драйвер магнитофона, целый ряд обслуживающих подпрограмм можно выгрузить и тем самым освободить значительную часть занимаемой системой памяти для загрузки утилит, драйверов, прикладных программ. Это особенно актуально для пользователей БК-0010 с ДОЗУ 8 Кб.

В новой версии DOSB10 решен вопрос передачи параметров непосредственно из системы утилитам и прикладным программам (параметры задаются в командной строке). Это позволяет неограниченно расширять набор команд с помощью загружаемых утилит.

Еще одно существенное отличие — возможность работы с подкаталогами. При этом предусмотрено создание на диске произвольного количества подкаталогов, копирование файлов из одного подкаталога в другой, пересылка (с удалением файла-оригинала), задание имени файла с учетом подкаталогов в командном режиме или из программ пользователя через EMT 36 (можно указать произвольный путь к файлу — от текущего подкаталога или от корневого, как и в MS-DOS). Все операции с подкаталогами выполняются просто и наглядно. Количество уровней подкаталогов ограничено только длиной командной строки. Но, в отличие от MS-DOS, в DOSB10 время доступа к файлу не зависит от уровня вложенности подкаталогов. Совместимость с дискетами, записанными в предыдущих вер-

сиях, полностью обеспечивается на уровне корневых каталогов.

Как и во многих профессиональных ОС, в DOSB10 2.0 реализованы стартовый командный файл и командные файлы пользователя. При этом в них можно задать последовательность команд не только для ОС, но и для прикладных программ, использующих ввод с клавиатуры по EMT-6.

При работе в DOSB10 2.0 на компьютерах БК-0011 и БК-0011М предоставляется целый ряд дополнительных удобств. Большие ресурсы памяти с успехом могут быть использованы для создания электронного диска емкостью 144—160 блоков (драйвер входит в комплект поставки системы). Одна из скрытых страниц памяти может быть использована для загрузки утилит, прикладных программ и драйверов. В комплект DOSB10 2.0 входит дисковый ФОКАЛ, отладчик MIRAG120, работающий в адресах 120000—132000 и не использующий основную и экранную память, утилита MODE, позволяющая установить одну из 16 палитр экрана, изменить или отключить звук («щелчок») при нажатии клавиш, установить новый шрифт, в том числе цветной. Кроме того, автором разработаны драйвер логических дисков, настраиваемый драйвер дисковода, позволяющий работать одновременно с односторонними и двухсторонними дискетами на одном и том же дисководе, модуль коррекции программ для БК-0011, обеспечивающий возможность работы на нем ПО для БК-0010 со звуком.

Наряду со значительным расширением возможностей системы, в DOSB10 2.0 принят ряд мер по повышению надежности и качества ее работы. Полностью отсутствует свопинг (в DOSB10 V1.3 он использовался при выборочном копировании из оболочки), и теперь архивные дискеты можно заклеивать для защиты от случайного повреждения файлов.

В ближайшее время для DOSB10 2.0 будет разработана загружаемая оболочка, по своим возможностям почти не уступающая Norton Commander (две панели, система меню, управление от клавиатуры и мыши, встроенный редактор и т.д.). Планируется также создание комплексной утилиты обслуживания дисков и каталогов BK-DUP, новой утилиты EDIT и целого ряда полезных драйверов.

Более подробную информацию о системе DOSB10 можно получить непосредственно у автора по адресу: 320055 Украина г. Днепропетровск-55, а/я 1351

А. А. Саяпин,
«ИнтерСервер»

Драйвер виртуального диска для ОС БК-0011М

Логическое устройство VM: представляет собой виртуальный диск объемом до 121 блока, организованный в дополнительных страницах ОЗУ БК-0011М. Виртуальный диск работает существенно быстрее, чем накопитель на дискетах, и свободен от сбоев и ошибок. Он может применяться для временного хранения часто используемых программ и файлов, для размещения временных файлов компиляторов, в качестве промежуточного устройства для копирования файлов и для других целей. Особенно заметно облегчается работа в однодисковых конфигурациях (становится возможным даже использование программы РСВК с одним дисководом). Драйвер можно настроить на работу с определенными страницами ОЗУ.

Условия применения

Для правильного и безопасного использования виртуального диска необходимо знать некоторые аспекты работы с расширенной памятью БК-0011М в операционной системе ОС БК-0011М и с некоторыми системными и прикладными программами. (Предполагается, что пользователь знаком с организацией памяти БК-0011М.)

Некоторые страницы памяти используются всегда одинаково. Так, страница 5 является основным экраном, страница 7 частично используется для нужд базовой ОС, хранения знакогенератора и при работе локальной сети, страницы 0, 1 и 2 образуют стандартную память для операционной системы и прикладных программ.

Страницы 3, 4, 6 и частично 7 могут быть использованы прикладными программами или виртуальным диском. Страница 6 может работать также и как дополнительный экран.

ОС БК-11 V 4.0 имеет два режима работы: стандартный и виртуальный. В виртуальном режиме ОС использует страницы 3, 4 и 6 для хранения подгружаемых частей и свопинга, что ускоряет работу больших программ, не использующих расширенную память. В стандартном режиме эти страницы освобождаются, но ОС требует постоянного присутствия системного диска в дисководе.

Так как виртуальный диск создается в расширенной памяти, драйвер VM требует стандартного режима работы ОС. Однако необходимо помнить, что некоторые программы (например, EXE10P) сами используют расширенную память, поэтому, пока у вас на виртуальном диске хранятся файлы, избегайте запуска таких программ. В крайнем случае, если вам известно, что программа использует только одну страницу, например дополнительный экран, можно настроить драйвер VM так, что он не будет обращаться к этой странице. При этом размер виртуального диска уменьшится на соответствующую величину.

Использование драйвера VM.SYS

Если ОС находилась в виртуальном режиме, перейдите в стандартный командой **SETSWP Disk**, а затем загрузите драйвер: **LOAD VM**. Если вы еще не работали с VM после перезагрузки системы, также проинициализируйте виртуальный диск командой **INIT VM:** .

После этого виртуальный диск готов к работе. Вы можете записывать на него файлы, считывать их, просматривать оглавление, в общем, делать все то же, что и с обычным диском. Информация, хранимая на VM:, не теряется, пока не произойдет:

а) выключение питания;

б) переход в виртуальный режим ОС;

в) запуск программы, использующей расширенную память;

г) изменение конфигурации страниц.

Настройка драйвера

Любая страница из четырех (3, 4, 6, 7) может быть разрешена или запрещена для использования драйвером. Для этого применяется команда **SET VM [NO]3 [NO]4 [NO]6 [NO]7** . Например, **SET VM 3 4 NO6 NO7** разрешает страницы 3 и 4 и запрещает 6 и 7.

Изменение конфигурации страниц влияет только на дисковую копию драйвера SY:VM.SYS и не на драйвер, загруженный в данный момент в ОЗУ, и делает информацию на виртуальном диске недостоверной. Если драйвер был загружен во время команды SET, выгрузите его командой **UNLOAD VM**, а затем снова загрузите и проинициализируйте. При этом будет установлен соответствующий размер виртуального диска.

Существует также команда **SET VM INFO**, которая выводит на экран краткую информацию о драйвере.

Примеры и рекомендации по использованию

При работе в стандартном режиме ОС рекомендуется, если позволяет размер запускаемых программ, запретить свопинг командой **SET USR NOSWAP**

Копирование файлов в однодисковой конфигурации

COPY/W SY:(список файлов) VM:
Mount input volume in BY0:; Continue? Y
— поставить исх. диск
Mount output volume in VM0:; Continue? Y
Files copied:
.....
Mount system volume in BY0:; Continue? Y
— поставить сист. диск
COPY/W VM: SY:
Mount input volume in VM0:; Continue? Y
Mount output volume in BY0:; Continue? Y
— поставить вых. диск
Files copied:
.....
Mount system volume in BY0:; Continue? Y
— поставить сист. диск

Использование совместно с InterCommander

При использовании версий 0.4 и 0.8 нужно настроить драйвер командой **SET VM 3 4 NO6 NO7** .

После этого загрузите и проинициализируйте VM:. В одной из панелей надо нажать AP2/7 и ввести в качестве имени устройства VM:. При смене дискет в системном устройстве обязательно запрещение свопинга.

Использование совместно с PCBK

Выгрузите все неиспользуемые драйверы для освобождения максимального количества памяти и запретите свопинг. Загрузите VM и запустите PCBK. В однодисковой конфигурации выньте системную дискету и вставьте дискету IBM PC. На первый запрос введите номер привода, на котором стоит дискета IBM PC. На второй — имя устройства VM:. После окончания копирования поставьте обратно системный диск, выйдите из PCBK и перези-

шите файлы с VM: на дискету. Если нужно скопировать файлы из ОС БК на дискету IBM PC, перепишите нужные файлы на VM: до запуска РСВК.

Использование совместно с ассемблер-транслятором MACRO

При трансляции больших файлов MACRO создает временный файл (по умолчанию на устройстве DK:). Если разместить его на виртуальном диске, скорость трансляции повысится. Это можно сделать командой **ASSIGN VM WF** .

Универсальное использование

Если вы интенсивно работаете с небольшим числом файлов, например редактируете, транслируете, компонуете и отлаживаете программу, можно скопировать нужные файлы на виртуальный диск и назначить его устройством по умолчанию: **ASSIGN VM DK** . В этом случае скорость работы существенно повысится. Но не забудьте скопировать файлы на дискету, иначе при выключении питания они будут утеряны.

Сообщения драйвера

Если какая-либо системная или прикладная программа выдает сообщение об ошибке при обращении к устройству VM:, это может быть вызвано одной из двух причин: драйвер не загружен или система находится в виртуальном режиме. Выполните действия, описанные в разделе «Использование драйвера»:

VM-F-Используйте команду SETSWP Disk — попытка загрузить драйвер в виртуальном режиме системы.

VM-W-Программы, исп-е доп. страницы, могут запортить VM:. Используйте команду SET VM INFO для получения доп. информации — предупреждающее сообщение при нормальной загрузке драйвера.

Примечания

1. Первые шесть блоков на каждом устройстве используются для хранения загрузчика ОС. Для экономии памяти эти блоки на устройстве VM: сделаны фиктивными. При их чтении выдаются нули, а запись игнорируется.

2. Если вам надоело сообщение, выдаваемое при нормальной загрузке, замените с помощью любой программы для коррекции двоичных файлов в программе SY:VM.SYS слово по адресу 1474 с 240 на 207.

Д. Ю. Усенков,
г. Москва

Компактное хранение БЕЙСИК-кодовых программ

В последнее время среди пользователей БК-0010.01 распространено большое количество БЕЙСИК-программ (прежде всего игр), написаных с использованием различных кодовых вставок: USR-подпрограмм, спрайтов, музыкальных фрагментов и т.п. Подобный прием позволяет создавать высококачественные программы, с одной стороны, делая доступной всю полноту возможностей БЕЙСИКа как языка высокого уровня (плавающая арифметика, вычисление математических функций и т.п.), а с другой — обеспечивая присущую ассемблеру высокую скорость исполнения программ при наличии достаточно богатой графики.

Однако у БЕЙСИК-кодовых программ есть и недостатки. Одним из них является то, что работа БЕЙСИКа с магнитной лентой и отсутствие программно исполняемых операторов подгрузки дополнительных модулей не позволяют автоматически загружать кодовые вставки. А выполнять рутинную поисковую работу на кассете и ручную подгрузку всех USR-программ и спрайтов перед каждым запуском БЕЙСИК-программы никому не хочется. Приходится либо формировать необходимые «добавки» в виде списка кодов в операторах DATA и записывать их в память в цикле операторами POKE, либо, сформировав вручную «раз и навсегда» БЕЙСИК-кодовый модуль и «подключив» все USR-функции, записать оператором BLOAD на магнитную ленту все содержимое пользовательского ОЗУ с адреса 2000 по 40000 (т.е. листинг на БЕЙСИКе, все кодовые вставки и все системные ячейки БЕЙСИК-транслятора).

В первом случае (с операторами DATA) резко возрастает объем листинга на БЕЙСИКе и скромные ресурсы ОЗУ БК-0010.01 не позволяют записать таким способом длинные

кодовые модули одновременно с достаточно большой БЕЙСИК-частью.

При втором способе (который чаще всего и применяется при создании БЕЙСИК-кодовых игр, тиражируемых для продажи) программист имеет возможность наиболее гибко использовать ресурсы БК, но... при хранении программы на кассете часть магнитной ленты тратится впустую. Ведь чтобы записать все «сюжетно необходимые части» такой программы — от области системных переменных БЕЙСИКа и БЕЙСИК-листинга в младших адресах ОЗУ до таблицы адресов строк БЕЙСИК-программы и кодовых модулей в старших, приходится сохранять и расположенные посередине пустые ячейки памяти.

При работе с программами в машинных кодах проблема компактного хранения на носителе информации решается достаточно просто — с помощью архиваторов. Сегодня программы-архиваторы, позволяющие «упаковывать» любые файлы (программы, тексты, картинки и др.), на БК уже имеются. В качестве примеров можно назвать BKZIP и ВКРАСК. Из них ВКРАСК (автор — А. Ходулев) наиболее удобен, так как создаст самораспаковывающиеся архивы: к началу архивного файла добавляется подпрограмма распаковки, и при его запуске с начального адреса происходит автоматическая распаковка. При этом в программе ВКРАСК предусмотрены два режима. В режиме «ПРОГРАММА» сразу же после распаковки файла управление передается на запуск распакованной программы (с учетом автозапуска, если он установлен). А в режиме «ДАННЫЕ» происходит только распаковка загруженного файла в оперативной памяти, а затем отрабатывается команда останова HALT.

В результате проведенных экспериментов обнаружилось, что архиватор ВКРАСК можно использовать для упаковки БЕЙСИК-кодовых программ. Общая методика его использования следующая:

создать БЕЙСИК-кодовую программу (загрузить БЕЙСИК-часть и все кодовые вставки, «подключить» USR-функции, можно установить нужные значения программируемых ключей клавиатуры и нарисовать на экране заставку или просто текст-подсказку типа «Нажмите АР2+0») и записать ее на магнитную ленту командой **BSAVE «имя»,&O2000,&O40000** (если необходимо сохранить и экранную картинку-заставку, число &O40000 нужно заменить на соответствующий больший адрес, а перед рисованием самой заставки «нормализовать» рулонное смещение экрана двухкратным нажатием АР2+СБР);

перейдя в монитор, запустить архиватор ВКРАСК и заархивировать ранее полученный файл <имя>.BIN в режиме «ДАННЫЕ» (имя для файла, полученного после архивации, должно быть приемлемым для БЕЙСИКа, т.е. иметь шесть символов и расширение .BIN);

запуск хранящегося на ленте упакованного файла БЕЙСИК-кодовой программы производится командой **BLOAD «имя»,R** (обязательно с R!). При этом файл загружается в ОЗУ и распаковывается. После этого на экран выдается надпись «СТОП», и можно запускать БЕЙСИК-часть, как обычно (оператором RUN либо ключами АР2+5 или АР2+0, если эти ключи не были изменены при создании программного файла).

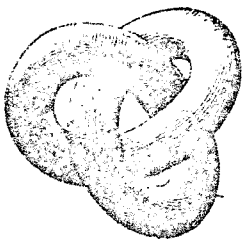
Уважаемые читатели!

В выпуске 1'94 после статьи Юрова В. П. и Юрова В. В. «Каталогизатор ДИСКАТ2.МВИ» редакцией ошибочно сообщалось о поставке программы ДИСКАТ2.МВИ в комплекте операционной системы ANDOS v2.30. Для приобретения этой программы следует обращаться по адресу:

127349, Москва, а/я 9, Юров Вячеслав Петрович.

Условия можно узнать по телефону:

(095) 908-22-12 с 10 до 21 часа ежедневно.



Самым заветным желанием любого, кто имеет доступ к какому-либо персональному компьютеру, почти всегда является стремление узнать о нем как можно больше. К сожалению, семейство БК в понимании многих незаслуженно считается всего лишь электронной игрушкой. По этой причине БКманам не приходится рассчитывать на сколько-нибудь заметное разнообразие книг, посвященных этому компьютеру, а следовательно, нужно по крупицам собирать сведения о нем в различных журналах. Чтобы помочь пользователям БК в подобной «поисковой работе», редакцией подготовлен список статей об этом ПК в наиболее известных изданиях, помещавших когда-либо на своих страницах материалы о БК.

Материалы о БК, опубликованные в журналах *

Справочные сведения

Информатика и образование

(издается с 1986 г.)

- ☐ Справочный листок (язык Фокал) №3/87—63.
- ☐ Информацня о Московском клубе БК №5/88—89.
- ☐ *Борисов А.* Пленочные переключатели №5/88—93.
- ☐ *Гриценко А.* Устранение «дребезга» клавиш у БК №5/88—95.
- ☐ *Алексин Б.* Где взять программы для БК? №6/88—97.
- ☐ Новые возможности БК №3/89—71.
- ☐ Модем для ПЭВМ — реальность №3/89—73.
- ☐ *Погаевский А.* Как записать картинку №3/89—97.
- ☐ *Антонов Д.* Увеличение тактовой частоты (турбо-режим) №2/90—46.
- ☐ *Канивец И.* Знаете ли вы, что... (советы) №3/90—53.
- ☐ *Панченков И.* Приручение БК (в том числе принципиальные схемы узлов БК) №4/90—68, №5/90—75.
- ☐ *Василенко В.* Спасение программ при зависании (Фокал) №6/90—71.
- ☐ *Соколов А.* Принципы работы БК-0010 №1/91—66.
- ☐ *Молчанов А.* Оптимальный драйвер магнитофона для БК (HELP7) №3/91—52.
- ☐ *Губаренко В.* Волшебная сила двух ячеек (ключи) №3/91—60.
- ☐ Рекомендацин по подключению внешних устройств к порту ввода-вывода БК-0010 №3/91—64.
- ☐ *Ким В.* Важные восьмеричные числа №3/91—65.
- ☐ БК-информ (цоколевка разъемов) №4/91—66.
- ☐ *Фролкин Б.* О «дребезге» клавиатуры БК-0010 №6/91—85.
- ☐ *Карагиоз С.* Попробуйте так! (Как избавиться от блока МСТД) №1/92—76.
- ☐ *Панченков И.* Подключение блока клавиатуры МС7008.01 к БК №2/92—98.
- ☐ Московский клуб БК №5—6/92—68.
- ☐ *Конюшенко А.* Система машинных команд БК-0010 №2/93—96.
- ☐ Калейдоскоп №1/90—73, №4/90—79, №5/90—89, №1/91—75, №5/91—78, №5—6/92—61.
- ☐ Советы и наблюдения №2/90—51.
- ☐ Программная орбита №4/88—96, №5/88—91, №6/88—98, №1/89—101, №3/89—98.
- ☐ Маленькие хитрости №4/88—97, №6/88—96, №1/89—104, №3/89—100.
- ☐ Обмен опытом №6/88—98.
- ☐ Кирпичики ваших программ №3/91—61, №1/92—83.

Вычислительная техника и ее применение

(издавался с 1988 по 1993 г.)

- ☐ *Кузьмин Ю. Я.* Бытовой компьютер БК-0010 — надежды и возможности №7/88—3.

* Обозначение: <номер>/<год>—<стр.>

- *Косенков С.М.* Новые модели семейства БК №7/88—13.
- *Синягин С.Ю.* Вывод текстов на экран №5/90—34.
- *Саттаров Н.М.* Состояние программного обеспечения микроЭВМ №8/90—3.
- *Мелентьев А.В.* Советы №8/90—42.
- *Синягин С.Ю.* Советы №8/90—45.
- *Усенков Д.Ю.* О мониторе и вильнюсской версии Бейсика №12/90—31.
- *Алексеев А.В.* О текстах подпрограмм в кодах №12/90—36.
- *Макаров Н.Я.* Копирование программ в турбо-формате №7/91—31.
- *Конюшенко А.М.* Что может делать процессор 1801ВМ1 №7/91—32.
- *Арбузов Н.* Совет №10/91—45.
- Советы читателей №11/91—18.
- *Усенков Д.Ю.* Дополнительные возможности EMT-прерывания БК-0010 №11/91—43.
- *Рабинович В.М., Дмитровский М.* Выход из отладки [Ø] в Фокал №11/91—47.
- *Шубин А.А.* Использование специальных прерываний БК-0010 №12/91—34.
- *Новак В.Е.* БК-0010.01: рекомендации фирмы ALT по улучшению клавиатуры №12/91—36.
- *Ивашинников С.А.* Новый канал обмена для БК-0010 №1/92—38.
- *Жданов В.С., Окунев С.А., Горелов В.Ю.* Команды микропроцессоров К580 и микроЭВМ «Электроника»» №3/92—3.
- *Чумак А.В.* Системные ячейки ПЭВМ БК-0010 №3/92—16.
- *Березенцев И.П.* Загрузка кодовых программ в КУВТ-86 №3/92—46.
- Советы читателей №4/92—45.

Наука и жизнь

(публикации о БК с 1991 г. прекращены)

- Получение кодов клавиш БК №4/86—83.
- Годится любой магнитофон №6/86—109.
- БК-0010: два года спустя №4/87—72.
- Несколько слов о прерывании №4/88—122, №12/88—91.
- Полезные советы, обмен опытом №4/87—74, №10/87—104, №4/88—122,123, №4/89—117, №12/90—130.

Персональный компьютер БК-0010 — БК-0011М

(приложение к журналу ИНФО, издается с 1993 г.)

- *Усенков Д.Ю.* Два совета пользователям БК №1/93—138.
- Блок дополнительного ОЗУ 32К для БК-0010 №1/93—140
- БК-0011М — 64 знака в строке №1/94—69.
- *Новиков А.В.* Кнопка RESET для БК-0010(.01) №1/94—71.

Микропроцессорные средства и системы

(издается с 1984 г., с 1991 г название изменено на «Средства и системы информации»)

- *Джхунян В.Л., Борщенко Ю.И. и др.* Однокристальные микропроцессоры комплекта БИС серии К1801 №4/84—12.
- *Косенков С.М., Полосин А.Н. и др.* Бытовая персональная микроЭВМ «Электроника БК-0010» №1/85—22.
- *Глушкова Г.Г., Иванов Е.А.* МикроЭВМ семейства «Электроника» №4/86—7.
- *Монахов В.Т* Программы любителей для бытовой персональной ЭВМ «Электроника БК-0010» №4/87—81.
- Рекомендации по применению микросхем серии К1801 №4/88—№2/89.
- *Монахов В.Т.* Программное обеспечение ПЭВМ серии «Электроника БК-0010» №3/89—60.
- *Розенштейн Э.П., Овчинников А.В.* Синхронизация работы микропроцессоров серии К1801 №5/89—17.
- Постоянное запоминающее устройство КР1801РЕ2 №5/89—94.

Для владельцев БК-0011(М)

Информатика и образование

- *Усенков Д.* От БК-0010 до БК-0011М №2/93—87.
- *Саяпин А., Вормсбехер* Знакомьтесь: БК-0011М №2/93—93.

Вычислительная техника и ее применение

- *Таланов С.П* Использование программ от БК-0010 на БК-0011 №1/92—34.

Персональный компьютер БК-0010 — БК-0011М

- *Саяпин А.А., Вормсбехер В.Р.* БК-0011М — первые шаги №1/93—10.
- *Саяпин А.А.* Программа копирования файлов IBM PC РСВК v1.1 №1/93—23.
- *Колосов А.Н.* Подключение мыши к БК-0011(М) №1/94—72.

Байтик

- *Прудковский А.* От БК-0010 к БК-0011 и далее №1/92—10.

Язык программирования Бейсик

Информатика и образование

- *Баронов Д., Николаев И.* Нуждается в доработке (Бейсик) №1/89—100.
- *Пилин А., Казан А.* Осваиваем Бейсик №1/89—102.
- И снова о ключах №1/89—102.
- *Козлов А.* Чего нет в описании (Бейсик) №3/89—100.
- *Зильберштейн С.* Структурный Бейсик для БК-0010 №5/89—113.
- *Авсеев В., Авсеев А.* Особенности транслятора с языка Бейсик для БК-0010.01 №2/90—42.
- *Бочаров А.* Хитрости Бейсика ЕК №2/90—49.
- *Гармашов А.* Удобный аналог команды MERGE №2/91—69.
- *Комаров С., Монин М.* Про Бейсик, быстродействие и кое-что еще №3/91—32.
- *Чернолясов А.* О символьных переменных. № 3/91—32
- *Молдавский Э.* Работа с символьными величинами №4/91—62
- *Муминов А.* Программное исполнение системных команд Бейсика БК №1/92—82.
- *Аляев Ю., Кузнецов С.* О возможностях функции MID$ при работе с файлами на КУВТ-86 №1/92—85.
- *Якошвили Д.* Обработка прерываний по клавише СТОП средствами Бейсика №2/92—94.
- *Гусейнов А.* Форматировка результатов вычислений в Бейсике БК №3—4/92—93.
- *Соловьев К.* Занимательные игры со строковыми фуикциями №5—6/92—69.
- *Белозеров О., Добряков В.* Ну и что, что мала память! №1/93—80.
- *Сапегин И.А.* О приручении клавиши «СТОП» №4/93—124.

Вычислительная техника и ее применение

- *Баронов Д.* Включение подпрограмм в машинных кодах в программы на Бейсике для БК-0010 №9/88—25.
- *Сарычихин П.И., Фролов С.А.* Большие программы на БК-0010 №3/89—35 (см. №7/89—17, №2/90—35, №2/90—40, №4/90—45, №8/90—37, №3/91—45).
- *Ермаков В.В.* Многомерные массивы на Бейсике №8/90—9.
- *Гармашов А.П.* О текстах подпрограмм в кодах для Бейсика БК-0010 №8/90—12 (см. №12/90—36).
- *Авсеев В.В., Авсеев А.В.* Использование возможностей Бейсика в программах на языке ассемблера №12/90—24.
- *Котов Ю.В.* Совместное использование Бейсика и машинных кодов №3/92—25.

Наука и жизнь

- На смену Фокалу №10/86—112.
- Фокал или Бейсик №6/88—113.
- Бейсик БК-0010.01 (описание языка) №3/89—109, №5/89—127.

Байтик

- *Юров В.* Мой путь в освоении Бейсика №1/91—23.

Язык программирования Фокал

Информатика и образование

- *Ашкенази В.* Литерные величины на БК-0010 (Фокал) №4/87—69.
- *Брик И.* Простые программы на Фокале №4/88—68.
- *Прис Г.* Супер-Фокал или Фонд БК №5/88—94.
- *Монахов В.* Фокал БК-0010 и его расширения №1/89—50.
- *Гвоздев С., Эрнестонс Г.* Использование плавающей арифметики Фокала в БК-0010 №5/89—60 (см. №3/90—49).
- *Бузитко В.* Вмешательство в работу программы (Фокал) №2/90—50.
- *Ларкин М.* Экономия памяти БК-0010.01 №3/90—44.

- *Гвоздев С., Эрнестонс Г.* Еще раз о плавающей арифметике Фокала №3/90—49.

Вычислительная техника и ее применение

- *Гвоздев С., Эрнестонс Г.* Использование плавающей арифметики Фокала в БК-0010. №5/89—31 (см. №4/90—22).

Наука и жизнь

- Таймерная функция в Фокале №4/87—74.
- Как засеять двумерное поле (о генераторе случайных чисел Фокала) №10/87—103.
- Кофок №4/89—117.

Микропроцессорные средства и системы

- *Казанцев А.П.* Интерфейс внешних функций интерпретатора Фокал—БК-0010 №4/87—86.
- *Куправа Т.А.* Интерпретатор языка Фокал в микросхеме ПЗУ №6/87—85.
- *Полянский П.В.* Эффективное хранение данных в языке Фокал ПЭВМ «Электроника БК-0010» №3/89—57.
- *Казанцев А.П., Данилов А.Б.* Интерфейс пользователя ПЭВМ «Электроника БК-0010» №3/87—57.
- Обзор программных продуктов системы «Интерфейс внешних функций интерпретатора Фокал БК-0010» №5/89—32.

Ассемблер и машинные коды

Информатика и образование

- *Зальцман Ю.* Архитектура и программирование на языке ассемблера БК-0010 №4/90—61, №5/90—79, №6/90—61, №1/91—59, №2/91—57, №3/91—43.
- *Зальцман Ю.* О системных ячейках БК-0010 №3/91—80.
- *Зальцман Ю.* Архитектура и ассемблер БК №4/91—53, №5/91—67.
- *Зальцман Ю.* Продолжительность исполнения команд на БК-0010 №6/91—73.
- *Еремин Е.* Как работает команда MARK №6/91—75.
- *Гиматов Ю.* Самомодифицирующиеся программы №2/92—91.
- *Булитко В.* Псевдопараллельное исполнение программ на БК-0010 №2/92—96.
- *Аскеров Р.* Параллельные процессы на БК-0010 №2/92—98.

Вычислительная техника и ее применение

- Слободчук В.В. Ассемблер для ЭВМ «Электроника БК-0010» №9/88—27.

Персональный компьютер БК-0010 — БК-0011М

- *Зальцман Ю.А.* Микро-ЭВМ БК-0010. Архитектура и программирование на языке ассемблера (переработанное и дополненное) №1/94—8.

Микропроцессорные средства и системы

- *Борзов Г.В., Ляпунов М.М.* Программные трюки на Макро-11 №5/87—37.

Прочие языки программирования

Информатика и образование

- *Кузьмин Ю.* Т-язык №2/87—64.
- *Григорьев С.* Работа системы Пролог-Д (язык Пролог для БК) №4/90—43, №5/90—51, №6/90—47, №1/91—41, №3—4/92—38.
- *Ингорь А.* Квазиподпрограммы №6/90—70.
- *Ингорь А.* Конвертор Бейсик-ассемблер для БК №1/91—69.
- *Чураков А.* Реализация ЛОГО для КУВТ-86 №1/92—108.
- *Андронов И.* Ода Форту №5/93—113.

Вычислительная техника и ее применение

- *Маслов В.В.* ФОРТ для БК-0010 №11/91—19.

Наука и жизнь

- Форт для БК №6/88—113.

Микропроцессорные средства и системы

- *Завилов В.Н., Константинов М.Ю., Померанец М.В.* Программирование на языке Паскаль для микроЭВМ «Электроника БК-0010» №1/87—37.

Периферийные устройства

Информатика и образование

- «Электроника БК-0010» работает с магнитофоном №5/88—89.
- *Камнев С.* Не только цифровые... (ЦАП и АЦП для БК) №5/88—92.
- *Чирков П., Булыч Р.* Копирование без проблем (приставка для копирования программ с магнитофона на магнитофон без участия БК-0010) №5/89—111.
- *Настасенко В.* Игровой пульт (как сделать джойстик; хотя это материал о «Ямахе», но он полезен и БКманам) №2/90—40.
- Полку периферии прибыло (телевизионный сканер для БК) №6/90—73.
- *Кумандин С., Соколов А.* «Электронный диск» для БК-0010 №1/91—72.
- *Диков А., Калашников А., Кулаков А.* «TermOC» (ОС для управления магнитофонным обменом) №2/91—64, №3/91—49.
- *Барсуков А.* Энергонезависимое ОЗУ №2/91—66.
- *Ермаков А.* Программатор микросхем ППЗУ 558РР3 для БК-0010 №3/91—57.
- *Коренков В.* Диспетчер ОЗУ и операционная система RAMDOS для БК-128К №6/91—76.
- *Буланов О.* Операционная система для БК-0010 №1/92—81.
- *Зальцман Ю.* О многочастотной кодовой модуляции №3—4/92—87.
- *Малашенко А., Аскеров Р.* Джойстик? Нет проблем! (Подключение к БК) №5—6/92—66.

Вычислительная техника и ее применение

- *Герман Н.* Рога для БК: доработки и самоделки №11/89—30.
- *Барташюс Р.* Компьютер БК-0010 в роли программатора ПЗУ №2/90—36 (см. №8/90—16).
- *Жариков А.Н.* Выбор магнитофона для БК-0010 №4/90—42.
- *Усенков Д.Ю.* О некоторых периферийных устройствах для БК-0010 №8/90—12.
- *Жариков А.Н.* Программа для программатора ППЗУ и ее оптимизация №8/90—16.
- *Сапогов В.В.* Система автоматического управления накопителем на магнитной ленте (бытовым магнитофоном) на базе БК-0010 №8/90—23.
- *Суяргулов М.Л.* Интерес к БК затухает №9/90—34.
- *Чирков П.А.* Преобразование интерфейсов в системе Бейсик-БК №7/91—39.
- *Усенков Д.Ю.* О некоторых периферийных устройствах для БК-0010 №8/91—38.
- *Герман Н.* Еще о рогах для БК №9/91—37.
- *Хабибулин Ю.Д.* Кассета ОЗУ-ПЗУ для БК-0010 №11/91—40.

Наука и жизнь

- Пульт для игры №10/86—113.
- Компьютер под елкой (использование порта БК для управления электрогирляндами) №12/86—92.
- БК плюс телефон №10/87—104.

Байтик

- *Юров В.* Что и как подключать к БК-0010 №1/92—4.
- *Малашенко А., Аскеров Р.* Джойстик?.Нет проблем! №1/92—14.

Микропроцессорные средства и системы

- *Лапиров А.В., Рудометов Е.А., Харазов Б.Г.* АЦП на БИС К1113ПВ1 для персональной ЭВМ «Электроника БК-0010» №4/87—85.
- *Егоров В.А.* Нестандартные внешние устройства для ПЭВМ «Электроника БК-0010» №3/89—33.
- *Меш М.Г.* Программный счетчик прерываний по таймеру для ПЭВМ БК-0010 №4/90—87.
- *Рудометов Е.А., Коленников А.Ю.* Программно-аппаратное обеспечение функции таймера ПЭВМ БК-0010 №4/90—88.

Зарубежная радиоэлектроника

(издается с 1947 г.)

- *Рудометов Е.А.* Аппаратно-программные средства управления микродвигателями РД09 от «Электроники БК-0010.01» и «Электроники МС0513» №10, 11, 12/93—46.
- *Рудометов Е.А.* Десятиразрядный двухдиапазонный АЦП с дифференциальным входом и активным фильтром нижних частот на ИС К1113ПВ1 №10,11,12/93—49.

- *Рудометов Е.А.* Цифро-аналоговые и аналого-цифровые преобразователи на ИС 572ПА1 №10,11,12/93—51.

Дисковод

Информатика и образование

- *Борисов А.* Магнитофон или дисковод? №4/89—81.
- *Надежин А.М.* Дисковая операционная система ANDOS №3/93—103.

Вычислительная техника и ее применение

- *Надежин А.М.* БК-0010: работа с дисководом №4/92—38.

Персональный компьютер БК-0010 — БК-0011М

- *Юров В.П.* БК-0010(.01) с дисководом №1/93—30.
- *Надежин А.М.* Дисковая операционная система ANDOS №1/93—50.
- *Ермаков А.М.* РАМОН — расширение монитора БК-0010 №1/93—58.
- *Винниченко А.И.* Дисковая операционная система «DOSB10» для БК-0010(.01) №1/93—72.
- *Бандалетов П.А., Бандалетов Е.М.* Программно-аппаратный комплекс (ПАК) БК-DISK №1/93—77.
- *Макаров В.* Подключение к ПЭВМ БК-0011, БК-0011М накопителя на жестких магнитных дисках (винчестера) №1/93—79.
- *Усенков Д.Ю.* БК-0010: опыт работы с диском №1/93—87.
- О подключении контроллера дисковода к БК-0010.01 (КМД-УК от УК НЦ) №1/93—118.
- *Юров В.В., Юров В.П.* Каталогизатор ДИСКАТ2.МВИ №1/94—64.

Байтик

- *Малахов В.* БК-0010 с дисководом №1/91—20.

Принтер

Информатика и образование

- Информация (подключение УВВП4 30/04 к БК-0010) №5/87—122.
- О пересылке информации на принтер в КУВТ-86 №5/88—58 (см. №3/89—68).
- Картинка — на принтер (ROBOTRON CM6329.02-M с КУВТ-86) №3/89—68.
- *Горвиц Ю., Глазко А.* Печать рисунков в КУВТ-86 №2/91—47.
- *Барсуков А.* Подключение принтеров «Электроника МС-6312» и «Электроника МС-6313» к БК №3/91—55 (см. №5/91—77).
- *Хоружий В.* Подключение принтера «Электроника МС-6312» к БК №4/91—61.
- *Кошелев В.* Подключение термопринтера 15ВВП80-002 к БК №6/91—86.
- *Панченков И.* Подключение принтеров к БК №1/92—78.
- *Никонов П.* СМ6337+БК №3—4/92—91.
- *Рогов В., Кузнецов А.* Эксплуатация принтера МС-6312 №5—6/92—61.
- *Барсуков А.* Печать графической информации на принтерах МС-6312 и МС-6313 №5—6/92—62.
- *Таланов С.* БК-0010 и телетайп №2/93—101.

Вычислительная техника и ее применение

- *Кузнецов О.Г.* Функции твердой копии для Фокала БК-0010 №5/90—32.
- *Ничипуренко Э.В.* Принтер к БК №3/91—15.
- *Бенке В., Бенке Э.* Программа LPRINT №12/91—10.
- *Захаров П.И.* Программа печати текстов PRINTPZ №12/91—12.
- *Усенков Д.Ю.* Использование возможностей вильнюсской версии Бейсика для вывода текста на принтер №12/91—21.

Наука и жизнь

- Подключение к БК принтера УВВП4 30/04 №12/88—90.

Микропроцессорные средства и системы

- *Черников Д.А., Черников К.А.* Пакет программ для работы микроЭВМ «Электроника БК-0010» в комплекте с печатающим устройством №3/90—81.
- *Лисенков Б.М.* Сопряжение пишущей машины Консул-260 с ПЭВМ типа БК-0010 №3/90—84.

Телевизор и видеомонитор

Информатика и образование

- ☐ *Асташенок С.* Рекомендую, проверено на практике! (Подключение к ТВ) №4/89—82.
- ☐ *Чирков П.* Подключение БК к телевизорам 3УСЦТ №1/90—69.
- ☐ *Луцкевич Ю.* Подключение БК к телевизорам 4УСЦТ №6/90—72.
- ☐ *Михайлов П.* Улучшение сопряжения БК и ТВ №5/91—75.
- ☐ *Луцкевич Ю.* 4УСЦТ-монитор №6/91—86.
- ☐ *Сулейманов Т. и др.* Цветной монитор как черно-белый №6/91—87.
- ☐ *Аскеров Р.* Если нет принтера... №4/93—125.

Вычислительная техника и ее применение

- ☐ *Усенков Д.Ю.* О подключении БК-0010 к телевизору №11/91—27.

Наука и жизнь

- ☐ Подключение БК к телевизору №10/86—111, №12/88—89
- ☐ Пересъемка с экрана телевизора (фотоаппарат вместо принтера) №10/86—113.
- ☐ БК помогает настроить телевизор №10/87—105.

Микропроцессорные средства и системы

- ☐ *Казанцев А.П., Майоров Л.Н., Данилов А.Б.* Указатель информации и интерфейс цветного телевизора для микроЭВМ «Электроника БК-0010» №3/89—54.
- ☐ *Врублевский Ю.Н., Осипенко Ю.П.* ТВ-приемники — в качестве видеомониторов персональных компьютеров №1/90—54.

Машинная графика

Информатика и образование

- ☐ *Рязанский М., Яцюк О.* Палитра компьютерной графики (графические средства Фокала) №1/87—58.
- ☐ *Аншаков О., Писаренко Н.* Методические рекомендации по использованию машинной графики БК-0010 (Фокал) №4/88—26.
- ☐ *Виноградов В.* Гармония чисел (графика Фокала) №1/89—104.
- ☐ *Зильберштейн С.* Радуга на экране БК №2/89—101 (см. №4/93—115).
- ☐ *Зильберштейн С.* Графика в мировых координатах №6/89—82.
- ☐ *Кузнецов А.* Еще раз о восьмицветном БК №1/90—70.
- ☐ *Смирнов С.* Восьмицветная приставка №1/90—71.
- ☐ *Белова Л., Белов Ю.* Изображение поверхностей №2/90—32.
- ☐ *Умников Е.* Спрайты для БК (Бейсик) №2/90—46.
- ☐ *Комаров С.* Строки вместо матриц (кодирование цветов БК) №2/90—48.
- ☐ *Гинзбург М.* Графика БК на уроке №3/90—40.
- ☐ *Матвийчук С.* Увеличение символов на экране №3/91—59.
- ☐ *Никонов П.* Спрайты на БК №3/91—63.
- ☐ *Садовников М., Петрук О.* Упрощение организации движения изображений в Бейсике №5/91—76.
- ☐ *Погожих С.* На экране — поверхность №5—6/92—64.
- ☐ *Усенков Д.Ю.* Реализация многоцветной закраски на Бейсике №4/93—115 (см. №5/93—122).

Вычислительная техника и ее применение

- ☐ *Герман Н.* От прямых линий на дисплее — к произвольным кривым №3/90—19.
- ☐ *Павлова А.А.* Базовая графика версий языка Бейсик, близких к MSX №1/91—20 (см. №5/91—21).
- ☐ *Любутов О.Д.* Редактор спрайтов для БК-0010 №1/91—33.
- ☐ *Надежин А.М.* Графические средства БК-0010 №1/91—42.
- ☐ *Шаповалов В.А.* Графика Бейсика №1/94—47.
- ☐ *Королев Д.В.* Простейший графический редактор №5/91—18.
- ☐ *Авсеев А.В., Авсеев В.В.* Динамические изображения на БК-0010 №5/91—33.
- ☐ *Усенков Д.Ю.* Электронное зеркало №10/91—44.

Наука и жизнь

- ☐ Мультфильм на экране (использование операций ГРАФ, ЗАП, СТИР) №10/87—102.

- ☐ Стереоизображение на экране БК №12/90—132.

Персональный компьютер БК-0010 — БК-0011М

- ☐ *Юров В.П.* Графические редакторы для БК-0010(.01) №1/94—24.
- ☐ *Усенков Д.Ю.* Генератор узоров для вязания №1/94—36.
- ☐ *Усенков Д.Ю.* Библиотека графических функций для ассемблера БК-0010(.01) №1/94—37.
- ☐ *Милюков А.В.* О нетрадиционном использовании знакогенератора БК №1/94—48.
- ☐ *Лядвинский М.В.* Новый набор символов в программах на Бейсике БК-0010.01 №1/94—51.
- ☐ *Аскеров Р.* О некоторых способах генерации шрифтов для БК №1/94—53.
- ☐ Векторный шрифт для Бейсика БК-0010.01 №1/94—57.
- ☐ *Рахманкулов Р.А.* Большие символы с тенями на БК №1/94—61.
- ☐ *Рогов В.И., Кузнецов А.В.* Увеличение символов на Фокале №1/94—62.
- ☐ *Константинов В.В.* Программа «Кружевница» №1/94—63.
- ☐ *Ерохин С.Л.* Исправление ошибки реализации оператора LINE в Бейсике БК-0011 №1/94—72.
- ☐ Использование графических средств Бейсика БК-0010 в программах на ассемблере №1/94—70.

Байтик

- ☐ *Юров В.* Делаем свои шрифты №4/91—8.
- ☐ *Юров В., Бабешко М.* Многоцветный графический редактор GRAF36 1/92—14.

Микропроцессорные средства и системы

- ☐ *Кузнецов А.Ф.* Устройство ввода графической информации в ЭВМ №4/89—58.
- ☐ *Лавровский В.Л.* Графика локальной сети микроЭВМ БК-0010 №3/90—82.

Компьютерная графика

(издается с апреля 1992 г.)

- ☐ *Ермак Е.* Работа со спрайтами №1/92—46.
- ☐ *Усенков Д.Ю.* Генератор узоров для вязания №1/92—47.
- ☐ *Павлова А.А.* О сочетании ПЭВМ разных классов в учебном процессе №2/92—34.

Музыкальные и звуковые эффекты, синтез речи

Информатика и образование

- ☐ *Чистяков А.* Здравствуйте, говорит «Электроника БК-0010»! №4/88—95.
- ☐ *Ивашинников С.* И снова «голос» домашнего компьютера №2/89—101.
- ☐ *Захребетков И.* Электромузыкальный инструмент с памятью №3/89—99.
- ☐ *Семенов О.* Если надоел звук (Бейсик) №4/89—83.
- ☐ *Бунцельман А.* Программа «Музыкальная заставка» №3/90—53.
- ☐ *Ивашинников С.* Многоголосие на БК-0010 №4/90—79.
- ☐ *Зальцман Ю., Михайлов В.* БК-0010 — речь и слух: возможности и реальность №2/91—93.
- ☐ *Усенков Д.* Музыкальный редактор для Бейсик-Вильнюса БК-0010.01 №2/93—71.
- ☐ *Пименов Г., Пименов Д.* Четырехголосный музыкальный редактор на Бейсике для БК-0010.01 (БК-11М) №2/93—74 (см. №4/93—91).
- ☐ *Самойлов В.* Программирование мелодий на БК-0011М №2/93—79.
- ☐ *Маслов В.* Музыка на Форте №2/93—80.
- ☐ *Рахманкулов Р.* Звуковые эффекты на БК-0010 №2/93—81.
- ☐ *Ивашинников С.* О создании музыкального оформления №2/93—83.
- ☐ *Новиков Ф.* Подпрограмма в ПЗУ №2/93—86.
- ☐ *Николаев В.* Рев БК №2/93—86.

Вычислительная техника и ее применение

- ☐ *Семенов О.Ю.* Драйвер бесшумной клавиатуры №6/89—45.
- ☐ *Усенков Д.Ю.* Синтезатор речи №2/90—34 (см. №9/91—36).

Наука и жизнь

- ☐ Голос домашнего компьютера №4/87—73.

Текстовые редакторы

Информатика и образование

- *Гриценко А.* Система подготовки текстов для КУВТ-86 (EDASP) №3/89—55.
- *Коренков В.* Текстовый редактор для БК: новые принципы и возможности №1/92—72.
- *Бакулин А., Черняков В.* BIGRED — текстовый редактор для БК-0010(.01) №1/92—75.
- *Булитко В.* Формульный редактор для БК №3/93—97.
- *Романов Д.А.* «VorteX!» — издательская система для БК №3/93—98.

Вычислительная техника и ее применение

- *Надежин А.М.* Использование БК-0010 для работы с текстами №3/91—16.
- *Усенков Д.Ю.* Текстовый редактор №3/91—19 (см. №9/91—36).

Микропроцессорные средства и системы

- *Киселев О.Е.* Универсальный текстовый редактор для микроЭВМ «Электроника БК-0010» №3/89—56.

Игры

Вычислительная техника и ее применение

- *Звездин Д., Рогусский А.* Игра «Лабиринт» №3/90—41.
- *Ермаков В.В.* Игра «Слалом» №8/90—10.
- *Ласкин С.А.* Игра «Монте-Карло» №10/90—46.
- *Воронин Е.В.* Игра в покер на БК №12/90—41.
- *Любутов О.Д.* Игра «Арканоид» №9/91—6.
- *Любутов О.Д.* Игра «Вертолет» №9/91—10.
- *Милюков А.В.* Все о Тетрисе №9/91—21.

Персональный компьютер БК-0010 — БК-0011М

- *Андросов Г.А., Герасимов А.Ю.* Две игры на Бейсике №1/93—141.
- Усенков Д.Ю. Игра «Жизнь» №1/93—144.
- *Иванов В.Е.* Игра «STEP» №1/94—73.

Разное

Информатика и образование

- *Кузьмин Ю., Гвардина Т., Кузьмина Л.* Система «Рита» №2/88—42.
- *Бойко А.* БК считает периоды №5/88—90.
- *Соколов Н.* Ваш новый «Блокнот» №6/88—95.
- *Каплун А.* Поздравьте своих учителей (Фокал) №1/89—98.
- *Комаров С.* Пишем в служебной строке №4/89—83.
- *Гужа П.* Расширение возможностей КУВТ-86 (организация рабочего места пользователя, доработка монитора) №6/89—76.
- *Яковлев В.* Отсчет времени на БК-0010.01 №1/90—65 (см. №2/90—53, №6/91—81).
- *Тереховский А.* Универсальный каталогизатор №3/90—50
- *Барсуков А.* Убыстрение поиска файлов (на кассете БК) №3/90—52.
- *Кумандин С.* Новый подход к построению баз данных на БК-0010 №6/90—72.
- *Парсаданов О., Янушко А.* Защита программ от несанкционированного доступа (Бейсик) №1/91—74.
- *Инжрь А.* Просмотрщик памяти №2/91—67.
- *Козлов О.* Куплю БК... №4/91—63.
- *Барсуков А.* БК-0010.01 в роли осциллографа №5/91—76.
- *Карагиоз С.* Многоформатный верификатор-каталогизатор №6/91—83.
- *Попов А., Кондратенко А.* Исследование ИМС с помощью БК-0010.01 №2/92—90.
- *Ежов М.* Еще раз о защите №3—4/92—89.
- *Лобанов А.* Каталожная информационно-поисковая система №3—4/92—92.
- *Сулханов В., Ходулев А.* «Электроника БК-0010» и компьютеры, совместимые с IBM PC №5—6/92—67.
- *Володин Е.* БК — стихотворец №1/93—81.
- *Вормсбехер В., Саяпин А.* Опыт рационального решения локальной сети с использованием IBM PC и БК-0011М №1/93—82.
- *Котов Ю.* Простое соединение ПЭВМ типа IBM PC и БК-0010 №2/93—104.
- *Котов Ю.* Экономное программирование для БК-0010.01 и БК-0011М №5/93—111.

Вычислительная техника и ее применение

- *Герман Н.* Все — за компьютеры! №7/88—13.
- *Герман Н.* Случайность, необходимость и прямолинейность №3/89—22.
- *Попов В.В.* Быстрый ввод операторов №12/90—35 (см. №9/91—36).
- *Саттаров Н.М.* Несколько доводов в защиту БК №3/91—13.
- *Сапогов В.В.* HELP-драйвер для БК-0010 №8/91—43 (см. №12/91—31).
- *Михулин М.В.* Программа DEC №4/92—34.

Наука и жизнь

- Эксперимент на БК-0010 №6/86—108.
- Компьютер БК в роли модельера №10/86—111.
- Случайны ли случайные числа №10/86—112.
- БК-0010 дома и в школе №8/87—33.
- Программа-«счетчик» для «разметки» пустой кассеты №4/88—121.
- Радиолюбитель поневоле №12/88—88.
- Мини-ОС «Мираж» №4/89—117.
- БК-0010 рассчитывает обмен квартир №12/90—130.

Персональный компьютер БК-0010 — БК-0011М

- *Степанов В.С.* Полвека в пути (заводу «Экситон» — 50 лет) №1/93—5.
- *Воронин Е.В.* Программа «DIR» №1/93—116.
- *Каймин В.А., Нечаев А.М.* Экспертиза программных средств для вычислительной техники, для компьютеров БК №1/93—119.
- *Жевак А.А.* Оценка производительности БК, и не только... №1/93—125.
- *Мальцев С.А.* Нестандартное применение команды RESET №1/93—129.
- *Сулханов В.* «Электроника БК-0010» и УК НЦ №1/93—133.
- *Страхов А.Ю.* Шаг назад и... прыжок в будущее №1/94—4.

Байтик

- *Цыганков В.* Бытовой нейрокомпьютер «Эмбрион» №2/91—26, №4/91—24.

Микропроцессорные средства и системы

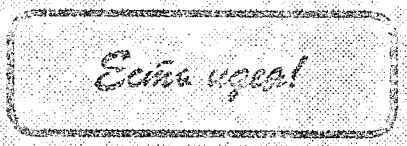
- *Фролов Г.И., Косенков С.М.* и др. Комплектный класс технических средств на базе микроЭВМ «Электроника БК-0010Ш» и ДВК-2МШ №4/86—65.
- *Обновленский П.А., Рудометов Е.А.* и др. МикроЭВМ «Электроника БК-0010» в системе управления производством кварцевого стекла №2/87—48.
- *Полянский П.В., Ширковский Н.А.* «Электроника БК-0010» в системах управления технологическими объектами №4/87—I сторона вкладки
- *Водянкин А.Г., Моисеенко В.И.* Учебная локальная сеть микроЭВМ №4/87—83.
- *Водяник А.Г.* Адаптация операционной системы ФОДОС к микроЭВМ «Электроника БК-0010» №3/89—55.
- *Полянский П.В.* «Электроника БК-0010» в системах исследования объектов с распределенными параметрами №3/89—58.
- *Ефимов А.С., Котов С.В* ПЭВМ «Электроника БК-0010» в качестве терминала №3/89—63.
- *Чернышев Ю.Н. и др.* Лаборатория МП техники с локальной сетью ПЭВМ «Электроника БК-0010» №5/89—II сторона обложки.
- *Полянский П.В.* Организация прикладных баз данных на ПЭВМ с малой емкостью ОЗУ №5/89—20.
- *Коврига Д.С., Гладченко С.М.* Микропроцессорная приставка для станков с ЧПУ на базе микроЭВМ «Электроника БК» №3/90—57.
- *Ситников С.Ю* Рабочее место ученика на базе микроЭВМ с кассетным накопителем №3/90—93.
- *Гвоздев С.В., Эрнестонс Г.А.* Универсальный отладчик ГРОТ №5/90—17.
- *Круглова Н.А.* МикроЭВМ «Электроника БК-0010» в системах автоматизации эксперимента №5/90—31

Анонс новой рубрики

«Есть идея для хорошей программы. Только реализовать ее не хватает времени и умения. Вот если бы кто-нибудь из сильных программистов.».

«Знаю компьютер как свои пять пальцев, могу запрограммировать все что угодно. Но ума не приложу, что бы еще такое придумать...»

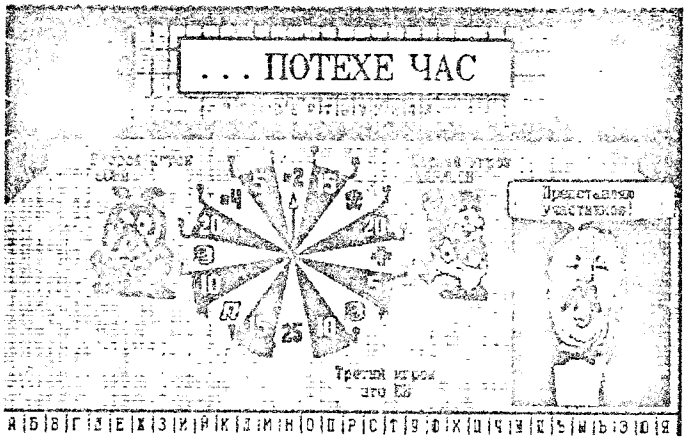
Вероятно, любой пользователь как скромных бытовых, так и мощных профессиональных компьютеров хоть раз да был обуреваем подобными мыслями. Желая помочь и тем, и другим, редакция предполагает открыть новую рубрику:



Приглашаем всех, у кого есть интересные задумки и заделы в написании прикладных, игровых, расчетных и прочих программ и кто не может самостоятельно довести их «до ума», поделиться своими идеями с другими читателями журнала. Лучшие предложения будут опубликованы в новой рубрике и, надеемся, помогут многоопытным, но страдающим недостатком фантазии программистам в их творчестве, дадут им четкое представление о первоочередных пожеланиях пользователей.

Программистам, сумевшим реализовать предложенные в новой рубрике идеи, редакция предоставит возможность публикации статей, рекламы и коммерческого распространения их разработок.

Ждем ваших писем и телефонных звонков.



Работая играть, играючи работать

М.В. Михулин,
г.Таллинн

Игра «Поле чудес»

Предлагаемая игровая программа «Поле чудес» подобна игре PUZZLE для Amstrad и почти полностью аналогична популярной телеигре. Однако имеются и некоторые отличия в правилах. Играют от одного до четырех человек, что определяется соответствующим указанием по запросу программы. Далее требуется выбрать способ задания фразы или слова. Возможны два варианта: фразу задает или компьютер, или «ведущий», не принимающий затем участия в игре. Фраза может включать в себя до четырех слов вместе с предлогами. Длина каждого слова — до 16 знаков. После компьютерного выбора или ввода фразы с клавиатуры на экране появляется игровое поле. В его верхней части выводятся имена игроков и их очки в процессе игры. Далее идет загаданная фраза или слово. Потом комментарий, т. е. краткая подсказка о том, что может означать зашифрованная фраза. (При ручном вводе желательно писать комментарий одним словом.) В правой нижней части располагается поле алфавита, в левой — игровой барабан. Две самые нижние строки поля отведены для диалога ЭВМ с игроками.

После «вращения» барабана и выпадания очков машина требует ввести согласную букву (но не гласную!), что определяется подсказкой в строке диалога. Время для раздумий ограничено. После нажатия на клавишу с выбранной буквой в случае ее наличия в слове (фразе) она появится на соответствующем месте в зашифрованной строке, а изображение этой буквы исчезнет из поля алфавита. После этого следует новый запрос компьютера.

При выпадании «0000» или «БАНКРОТ» право хода передается другому игроку с соответствующей поправкой в очках. Имя играющего в данный момент пишется инверсным текстом. Начинающего игру компьютер выбирает случайным образом.

Если играющий знает задуманное слово (фразу), он может ввести его, не вращая барабан. Для этого необходимо нажать клавишу «?». В нижней строке игрового поля появится подсказка о дальнейших действиях.

Если сумма очков у игрока больше 1000, компьютер предлагает купить гласную за 1000 очков.

Если все согласные буквы в слове (фразе) исчерпаны, предлагается или ввести его целиком, или купить гласную.

При правильном вводе фразы играющему прибавляется еще 1000 очков и он становится победителем, даже если сумма его очков меньше, чем у других. Если же фраза введена с ошибкой, компьютер выдает об этом сообщение и пишет правильную фразу. Естественно, в этом случае победителем становится компьютер.

Программа предлагает достаточно большой выбор слов и фраз (до 35 вариантов). Их можно подобрать по определенной тематике, например «литература», «география» и т.д. Игра может также использоваться при обучении алфавиту, правописанию, иностранным

языкам. Правда, в последнем случае русский алфавит надо заменить на латинский, а в операторах DATA писать слова на изучаемом языке.

Теперь немного о самой программе. Она написана на вильнюсском Бейсике для БК-0010.01 и занимает около 14 Кб, т. е. практически все ОЗУ пользователя. В связи с этим при наличии длинных фраз в операторах DATA возможно переполнение памяти во время игры. Тогда нужно уменьшить или длину фраз, или их количество. В последнем случае необходимо изменить число 35 в строке 1740 (количество фраз). Возможны также сбои при выводе имен игроков на экран.

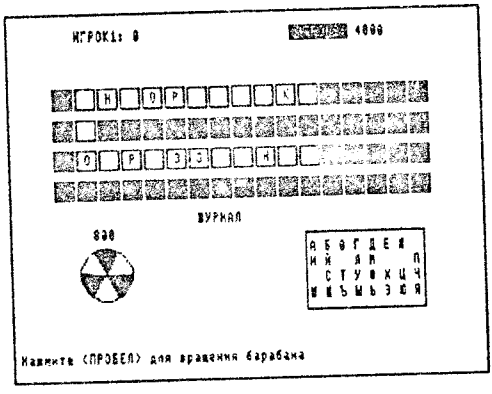
Фразы и комментарии записываются в операторах DATA в следующем формате: фраза, комментарий...

Следует заметить, что в программе нет защиты от повторного нажатия клавиши уже отгаданной буквы. Если это произойдет, очки игрока могут увеличиться.

В целях экономии памяти программа набирается без пробелов и с сокращением слов команд.

Основные блоки программы:

60—170 — ввод количества и имен игроков;
180—210 — выбор режима задания слова (фразы);
220—650 — рисование игрового поля;
660—940 — «вращение барабана» и случайный выбор числа очков;
980—1040 — ввод согласной буквы и проверка, что это действительно согласная. Временное ограничение на ввод буквы — 5 секунд (цикл 990—1020). Если время истекло, происходит переход хода;
1050—1190 — проверка на наличие введенной буквы в слове (фразе) и подсчет очков S%(JGD%) в зависимости от количества угаданных букв Z%;
1200—1240 — подпрограмма вывода угаданных букв в соответствующих клетках, подсчет количества угаданных согласных и гласных;
1250—1310 — подпрограмма удаления использованных букв из поля алфавита;
1320—1410 — альтернативный выбор согласной или гласной в зависимости от числа очков игрока;
1420—1460 — выбор: написать фразу (слово) или купить гласную. Запрос происходит, когда все согласные буквы слова (фразы) уже отгаданы;
1470—1590 — запрос по нажатию «?» и ввод слова (фразы) целиком. Подсчет числа очков игрока;
1600—1650 — подпрограмма первоначального ввода слова (фразы) и комментария ведущим;
1660—1830 — выбор слова (фразы) компьютером, подсчет общего числа согласных и гласных;
1890—2050 — подпрограмма вывода имен игроков, очередности хода, количества очков;
2090—2190 — подпрограмма смены хода и проверки условий выигрыша по угаданным буквам;
2230—2250 — определение победителя, вывод его имени на экран.



Основные переменные программы:

JUG% — количество игроков;
S%(JGD%) — очки каждого игрока;
Z% — количество угаданных букв;
FR% — количество согласных в слове (фразе);
LT% — количество гласных в слове (фразе);
GL% — количество угаданных гласных;
FL% — количество угаданных согласных;
N$ — имя игрока;
F$ — загаданное слово (фраза);
K$ — комментарий;
S$ — очки игрока (для вывода на экран);
G$ — имя победителя;
FF$ — слово (фраза), введенное игроком при попытке угадать всю фразу.

```
10 CLEAR 400
20 POKE 112%,15360%+PEEK(132%)
30 ? "        ПОЛЕ ЧУДЕС"
40 IF PEEK(&O40)<>0%TH?CHR$(155%)
50 DIM N$(3%),F$(3%),A$(3%),L$(16%)
60 CLS
70 ? AT(1%,10%)"Число игроков (1-4): ";
80 A$=INKEY$
90 IF A$<"1"ORA$>"4"TH70
100 JUG%=VAL(A$)-1%
110 CLS
120 ? "ИГРОКИ: "
130 FOR J%=0%TOJUG%
140 ? "ИМЯ ИГРОКА #";J%+1%;
150 INPUT N$(J%)
160 NEXT J%
170 JGD%=INT(RND(1)*JUG%)
180 CLS
190 ? AT(1%,8%)"Нажмите <F>, если желаете
    сами записать"AT(1%,9%)"фразу, или
    клавишу <ПРОБЕЛ>"
200 C$=INKEY$
210 IF C$="F"ORC$="Ф"TH1600ELIF
    C$=" "THGOSUB1660EL200
220 CLS
230 ? CHR$(155%)
240 LINE (0%,0%)-(255%,239%),1%,B
250 GOSUB 1890
260 J%=0%
270 A$=" "
280 FOR J%=0%TOP%
290 A$=""+A$+F$(J%)
300 NEXT J%
310 FOR J%=0%TO3%
320 FOR I%=1%TO17%
330 A$(J%)=" "+F$(J%)+STRING$(17%-
    LEN(F$(J%))," ")
340 A%=48%+J%*20%
350 B%=12%+I%*12%
360 LINE (B%,A%)-(E%+10%,A%+13),1%,B
370 IF MID$(A$(J%),I%,1%)<>" "TH410
380 PAINT (B%+1%,A%+1%),1%,1%
390 GOTO 420
410 PAINT (B%+1%,A%+1%),4%,1%
420 NEXT I%,J%
430 ? CHR$(155%)
440 LINE (316%,145%)-(444%,195%),,B
450 L$(0%)="А Б В Г Д Е Ж З"
460 L$(1%)="И Й К Л М Н О П"
470 L$(2%)="Р С Т У Ф Х Ц Ч"
480 L$(3%)="Ш Щ Ъ Ы Ь Э Ю Я"
490 FOR J%=0%TO3%
500 FOR I%=1%TO15%
510 ? AT(40%,15%+J%)L$(J%)
520 NEXT I%,J%
530 ? AT(25%,13%)K$
540 ? CHR$(155%)
550 CIRCLE (50%,170%),17%,2%,,,1.2
560 FOR V%=0%TO300%ST60%
570 W!=V%*PI/180%
580 X%=50%+(14*COS(W!))
590 Y%=170%+(17*SIN(W!))
600 LINE (50%,170%)-(X%,Y%),2%
610 NEXT V%
620 PAINT (58%,171%),1%,2%
630 PAINT (41%,182%),1%,2%
640 PAINT (47%,155%),1%,2%
650 ? CHR$(155%)
660 ? AT(1%,21%)STRING$(38%," ")AT(1%,22%)
    "Нажмите <ПРОБЕЛ> для вращения
    барабана";
670 F%=0%
680 Q$=INKEY$
690 IF Q$=" "TH700ELIF Q$="?"TH1470EL680
700 ? AT(11%,14%)
710 ? CHR$(155%);
720 I%=INT(RND(1)*8%)
730 II%=INT(RND(1)*15%)
740 X%=0%
750 Y%=1%
760 FOR J%=1%TOI%
770 FOR T%=1%TO2%
780 PAINT (58%,171%),X%,2%
790 PAINT (53%,181%),Y%,2%
800 PAINT (41%,182%),X%,2%
810 BEEP
820 PAINT (38%,168%),Y%,2%
830 PAINT (47%,155%),X%,2%
840 BEEP
850 IF X%=0%THX%=1%ELX%=0%
860 IF Y%=1%THY%=0%ELY%=1%
870 NEXT T%,J%
880 ? CHR$(155%)
890 RESTORE 930
900 FOR J%=0%TOII%
910 READ D$
920 NEXT J%
930 DATA300,600,100,1000,0000,500,100,900,800,
    БАНКРОТ,5000,700,200,400,3000
940 ? AT(11%,14%)D$
950 IF D$="БАНКРОТ"TH2090ELIF
    D$="0000"TH2110
960 IF S%(JGD%)>1000%ANDGL%<LT%TH1320
980 ? AT(1%,21%)STRING$(28%," ")AT(1%,22%)
    "Выберите согласную букву:          "
990 FOR T=1TO20
1000 B$=INKEY$
```

```
1010 IF B$<>""ANDF%=0%TH1040ELIF
     B$<>""ANDF%=1%TH1050
1020 NEXT T
1030 GOTO 2100
1040 IF B$<"Ю"ORB$>"Ъ"ORB$="А"ORB$=
     "Е" ORB$="И"ORB$="О" ORB$="У"OR
     B$="Ы"ORB$="Э"ORB$="Ю"ORB$="Я"OR
     B$=" " TH980
1050 N%=0%
1060 Z%=0%
1070 FOR J%=0%TOP%
1080 FOR I%=1%TOLEN(F$(J%))
1090 Y%=5%+J%*2%
1100 X%=8%+2%*I%+N%
1110 N%=N%+1%
1120 IF Q$="?"THB$=""+MID$(F$(J%),I%,1%)
1130 IF MID$(F$(J%),I%,1%)=B$THGOSUB1200
1140 IF I%=LEN(F$(J%))THN%=0%
1150 NEXT I%,J%
1160 IF Q$="?"TH2060
1170 S%(JGD%)=S%(JGD%)+VAL(D$)*Z%
1180 GOSUB 1250
1190 IF Z%=0%TH2100EL2150
1200 ? AT(X%,Y%)B$;
1210 Z%=Z%+1%
1220 IF F%=0%THFL%=FL%+1%
1230 IF F%=1%THGL%=GL%+1%
1240 RETURN
1250 FOR J%=0%TO3%
1260 FOR I%=1%TO15%ST2%
1270 M$=""+MID$(L$(J%),I%,1%)
1280 IF M$=B$TH?AT(39%+I%,15%+J%)" "
1290 IF M$=B$TH1310
1300 NEXT I%,J%
1310 RETURN
1320 ? AT(1%,21%)"Выберите согласную
     букву или"AT(1%,22%) "нажмите
     <ПРОБЕЛ>,чтобы купить гласную      ";
1330 FOR WT=1TO20
1340 B$=INKEY$
1350 IF B$=" "TH1380ELIF B$""TH1040
1360 NEXT T
1370 GOTO 2110
1380 ? AT(1%,21%)STRING$(38%," ")
     AT(1%,22%) "Выберите гласную (-1000):"
1390 F%=1%
1400 S%(JGD%)=S%(JGD%)-1000%
1410 GOTO 990
1420 ? CHR$(155%)
1430 D$="0"
1440 ? AT(1%,21%)"Нажмите<F> ,чтобы
     написать фразу," AT(1%,22%)"или
     <ПРОБЕЛ>,чтобы купить гласную      ";
1450 B$=INKEY$
1460 IF B$="F"ORB$="Ф"TH1500ELIF
     B$=" "TH1380EL1450
1470 ? AT(1%,21%)"Нажмите<F>, чтобы
     написать фразу," AT(1%,22%)"или
     <ПРОБЕЛ>,чтобы это сделал Я      ";
1480 A$=INKEY$
1490 IF A$="F"ORA$="Ф"TH1500ELIF
     A$=" "TH1050EL1480
1500 ? CHR$(158%);AT(1%,21%)"Напишите
     свою фразу"CHR$(156%)
1510 ? AT(1%,22%)STRING$(38%," ")
1520 Q$="?"
1530 LOCATE 1%,22%
1540 INPUT "Фраза: ";FF$
1550 LOCATE 1%,22%
1560IF FF$=F$TH?CHR$(7)+CHR$(7)+CHR$(7);
     EL?CHR$(156%)AT(1%,21%)"НЕТ!!! Ваша
     фраза неверна!"
     AT(1%,22%)STRING$(38%," ")
     AT(1%,22%)"Фраза: ";F$;CHR$(156%);
1570 IF FF$=F$THS%(JGD%)=S%(JGD%)+1000%
1580 IF FF$=F$THG$=N$(JGD%)
1590 GOTO 1050
1600 ? AT(1%,12%)"Фраза должна иметь не
     более"AT(1%,13%)"4-х слов вместе с
     предлогами"
1610 ?
1620 INPUT "Фраза: ";F$
1630 INPUT "Комментарий: ";K$
1640 GOSUB 1710
1650 GOTO 220
1660 RESTORE 1840
1670 I%=INT(RND(1)*35%)
1680 FOR J%=0%TOI%
1690 READ F$,K$
1700 NEXT J%
1710 J%=0%
1720 F$(J%)=""
1730 FOR I%=1%TOLEN(F$)
1740 B$=""+MID$(F$,I%,1%)
1750 F$(J%)=""+F$(J%)+MID$(F$,I%,1%)
1760IF B$="А"ORB$="Е"ORB$="И"OR
     B$="О"ORB$="У"ORB$="Ы"ORB$="Э"OR
     B$="Ю"ORB$="Я"THLT%=LT%+1%
1770 IF B$=" "THJ%=J%+1%
1780 IF J%>3TH250
1790 IF B$=" "TH1810
1800 FR%=LEN(F$)-LT%-J%
1810 NEXT I%
1820 P%=J%
1830 RETURN
1840 DATA АРГЕНТИНА,СТРАНА,АЛЬБЕРТ
     ЭЙНШТЕЙН,УЧЕНЫЙ,ФЕДОР
     МИХАЙЛОВИЧ ДОСТОЕВСКИЙ,
     ПИСАТЕЛЬ,ВАШИНГТОН,СТОЛИЦА,
```

```
    НОРБЕРТ ВИНЕР,УЧЕНЫЙ,
    ТЕРАПЕВТ,ВРАЧ,ПОЛЕ ЧУДЕС,
    ТЕЛЕИГРА,АНДРЕЙ РОСТОЦКИЙ,
    АРТИСТ
1850 DATA МЕРСЕДЕС,АВТОМОБИЛЬ,
    СНЕЖНЫЙ БАРС,ЖИВОТНОЕ,
    ГАЙ ЮЛИЙ ЦЕЗАРЬ,ИМПЕРАТОР,
    ДЕД ЩУКАРЬ,ПЕРСОНАЖ,
    ТИХИЙ ДОН,РОМАН,
    ШРИ АУРОБИНДО,ФИЛОСОФ,
    ЮЖНАЯ АМЕРИКА,МЕСТО,
    ВЕЛИКАЯ КИТАЙСКАЯ СТЕНА,
    СООРУЖЕНИЕ
1860 DATA АРТУР КОНАН ДОЙЛ,ПИСАТЕЛЬ,
    МАШИНА ВРЕМЕНИ,РОК-ГРУППА,
    ВЛАДИМИРСКИЙ ТЯЖЕЛОВОЗ,
    ЖИВОТНОЕ,БАЛАШИХА,ГОРОД,
    АЛЕКСАНДР ПОПОВ,УЧЕНЫЙ,
    А ЗОРИ ЗДЕСЬ ТИХИЕ,ФИЛЬМ
1870 DATA ЛУИ ДЕ ФЮНЕС,АРТИСТ,
    БИРЖЕВОЙ МАКЛЕР, ПРОФЕССИЯ,
    МАЛАЯ КУРИЛЬСКАЯ ГРЯДА,ОСТРОВА,
    АВРААМ ЛИНКОЛЬН,ПРЕЗИДЕНТ,
    СПЯЩАЯ КРАСАВИЦА,БАЛЕТ,
    БОРИС ГОДУНОВ,ОПЕРА,
    ВЕСЕЛЫЕ КАРТИНКИ,ЖУРНАЛ
1880 DATA РЫБА ГНИЕТ С ГОЛОВЫ,
    ВЫРАЖЕНИЕ, КАНАРСКИЕ
    ОСТРОВА,МЕСТО,АЛЬБЕРВИЛЬ,ГОРОД,
    ГАЛИНА ВИШНЕВСКАЯ, ПЕВИЦА,
    ПРЕСТУПЛЕНИЕ И НАКАЗАНИЕ,КНИГА,
    КАЗАНСКИЙ СОБОР,СООРУЖЕНИЕ
1890 IF PEEK(&O40)<>0%TH?CHR$(155%);
1900 H%=8%
1910 FOR J%=0TOJUG%
1920 GOSUB 2000
1930 ? AT(H%,J%/2%+1%)" «;N$(J%);»:";
1940 GOSUB 2000
1950 GOSUB 2020
1960 IF H%=8%THH%=38%ELH%=8%
1970 NEXT J%
1980 ? CHR$(155%)
1990 RETURN
2000 IF J%=JGD%TH?CHR$(156%)
2010 RETURN
2020 LOCATE H%+LEN(N$(J%))+2%,J%/2%+1%
2030 IF S%(JGD%)=0%THS$=STR$(S%(JGD%))
     +"       «EL S$=STR$(S%(JGD%))
2040 IF J%=JGD%TH?S$;
```

Уважаемые читатели!

Подписаться на журнал «Персональный компьютер БК-0010 — БК-0011М» можно в любом отделении связи или непосредственно в редакции. В каталоге ЦРПА «Роспечать» данные о журнале следует искать на букву «Б» — «Библиотечка журнала «Информатика и образование».

Чтобы приобрести отдельные выпуски журнала через редакцию:

- частным лицам необходимо перечислить за каждый выпуск 4300 руб. включая стоимость почтовых расходов (по пересылке бандероли из Москвы в пункт назначения) и 300 руб. (орграсходы);
- предприятиям и организациям необходимо перечислить на расчетный счет редакции за каждый выпуск 7300 руб. включая стоимость почтовых расходов (по пересылке бандероли из Москвы в пункт назначения) и 300 руб. (орграсходы).

Расчетный счет для Москвы и Московской области: 609602 в ММКБ филиал «Интеллект», МФО 212199, уч.1Е

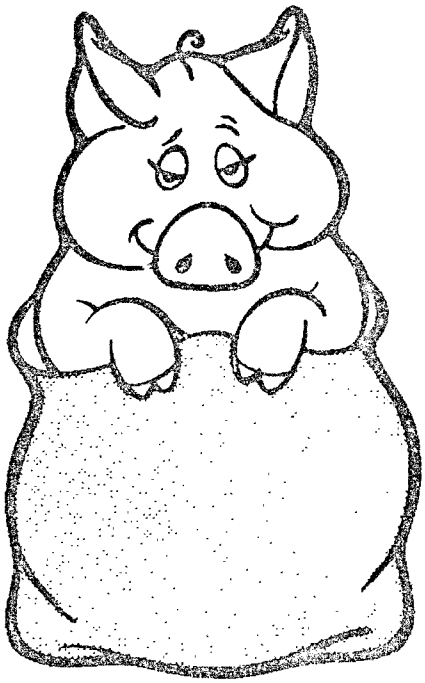
Расчетный счет для других городов России и ближнего зарубежья: 609602 в ММКБ филиал «Интеллект», корр. счет 216161800 в ЦРКЦ ГУ ЦБ РФ, уч.С3, МФО 211004.

Перечисление денег необходимо подтвердить письмом с вложенной в конверт заявкой (см. на обороте) по адресу:

103051, Москва, ул.Садовая Сухаревская, д.16, комн.9

Редакция журнала «Информатика и образование».

```
2050 RETURN
2060 IF S%(JGD%)<0%THS%=0%
2070 GOSUB 1890
2080 GOTO 2200
2090 S%(JGD%)=0%
2100 GOSUB 1890
2110 JGD%=JGD%+1%
2120 BEEP
2130 BEEP
2140 IF JGD%>JUG%THJGD%=0%
2150 GOSUB 1890
2160 IF FR%+LT%=FL%+GL%THG$=N$(JGD%)
2170 IF FR%+LT%=FL%+GL%TH2060
2180 IF FR%=FL%TH1420
2190 GOTO 650
2200 FOR T=0TO1500
2210 NEXT T
2220 CLS
2230 IF FF$=F$ORFR%+LT%=FL%+GL%AND
     S%(JGD%)=>0%TH?AT(2%,11%)
     «* * * ВЫИГРАЛ(А) »;G$;" * * * «
     EL?AT(3%,11%)»* * * ВЫИГРАЛ
     КОМПЬЮТЕР * * *"
2240 LOCATE 1%,22%
2250 END
```



ЗАЯВКА

на журнал «Персональный компьютер БК-0010 — БК-0011М»

_____

(адрес подписчика с почтовым индексом)

_____

(фамилия, имя, отчество полностью)

_____

(номер выпуска и год издания)

_____

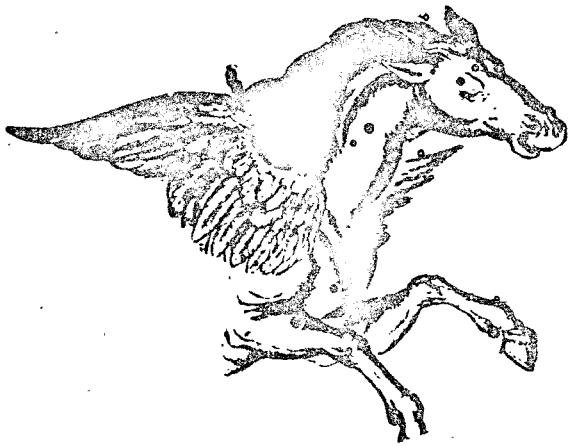
(общее количество экземпляров)

Перечислено на расчетный счет _____

_____ руб.

(общее количество экземпляров х стоимость одного экземпляра)

Платежное поручение № _____ от _____ 199 г.



СОДЕРЖАНИЕ