О. БЕЛОЗЕРОВ, В. ДОБРЯКОВ

Ну и что, что мала память!

Одним из эффективных приемов решения проблемы малого объема ОЗУ БК-0010 (в составе КУВТ-86) является оверлейное построение программ, так что в нужный момент в ОЗУ подгружаются блоки, необходимые в текущий момент времени для работы.

Основная часть ППС создается на Бейсике (Вильнюс-86), к числу очевидных недостатков которого следует отнести и невозможность использования операторов обращения к диску в программном режиме.

Вопросам памяти БК уделяется большое внимание. В частности, предложена следующая идея: обеспечить выполнение Бейсик-программы по заданному сценарию (Вычислительная техника и ее применение. 1989. № 3, 4; 1990. № 8). К сожалению, сценарий имеет жесткую структуру, не предусматривающую ее изменения в ходе выполнения.

Нам удалось устранить данный недостаток. Взяв за основу программу И. П. Березенцева и незначительно изменив способ управления, мы практически забыли о малой памяти БК, так как получили возможность создавать на Бейсике достаточно разветвленные блочные учебные программы, имеющие возможность из одного блока загружать другой, запускать его, задавать последовательность выполнения программных модулей.

На листинге управляющая подпрограмма начинается со строки 5000. Перед обращением к ней необходимо определить переменную A$, придав ей значение последовательности выполняемых команд с включением управляющих кодов, например код 10 — **ВК** (здесь и далее все значения кодов и адресов десятичные).

Учитывая возможные ошибки, возникающие в Бейсик-системе при работе с символьными переменными, лучше всего это сделать, определив соответствующую функцию пользователя (строки 90, 100).

В приводимом листинге из базовой программы вызывается модуль «TERMO 1», при этом символьная переменная A$ принимает следующее значение (последовательность команд):

```
10 REM ********************************
20 REM *                              *
30 REM * ОСНОВНАЯ ЧАСТЬ ПРОГРАММЫ     *
40 REM *                              *
50 REM ********************************
60 ......
70 ......
80 REM <<< ЗАДАНИЕ
          ПОСЛЕДОВАТЕЛЬНОСТИ КОМАНД >>>
90 DEF FNA$(A$)="NEW"+CHR$(10)+
    "CLOAD"+CHR$(34)+A$+CHR$(34)+",R"+
                        CHR$(10)+CHR$(3)
100 A$=FNA$("TT:TERMO1")
110 GOTO 5000
120 ......
130 ......
5000 REM <<< ПОДПРОГРАММА ЗАГРУЗКИ >>>
5010 DEF USR1=16128
5020 DEF USR2=16142
5030 DATA 5443,4930,5569,16222,
                        -27503,32450,135
5040 DATA 2551,2,289,5623,16222,66,
                        5599,16162,24
5050 DATA 135,9718,-18418,16,770,
                        95,-32694,5047
5060 DATA 40,-6665,2,34,9727,-30714,
                        28,757,-24640,24
5070 DATA 2743,20,9664,3,769,2,5599,
                        -32694,24,95,-24420,0
5080 I%=16128
5090 READ A%
5100 POKE I%,A%
5110 I%=I%+2%
5120 IF A% THEN 5090
5130 A$=USR1(A$) 'ПЕРЕСЫЛКА
                        ПЕРЕМЕННОЙ A$
5140 A%=USR2(A%) 'ЗАПУСК КОМАНД
                        НА ВЫПОЛНЕНИЕ
```

О. БЕЛОЗЕРОВ, В. ДОБРЯКОВ

Ну и что, что мала память!

Одним из эффективных приемов решения проблемы малого объема ОЗУ БК-0010 (в составе КУВТ-86) является оверлейное построение программ, так что в нужный момент в ОЗУ подгружаются блоки, необходимые в текущий момент времени для работы.

Основная часть ППС создается на Бейсике (Вильнюс-86), к числу очевидных недостатков которого следует отнести и невозможность использования операторов обращения к диску в программном режиме.

Вопросам памяти БК уделяется большое внимание. В частности, предложена следующая идея: обеспечить выполнение Бейсик-программы по заданному сценарию (Вычислительная техника и ее применение. 1989. № 3, 4; 1990. № 8). К сожалению, сценарий имеет жесткую структуру, не предусматривающую ее изменения в ходе выполнения.

Нам удалось устранить данный недостаток. Взяв за основу программу И. П. Березенцева и незначительно изменив способ управления, мы практически забыли о малой памяти БК, так как получили возможность создавать на Бейсике достаточно разветвленные блочные учебные программы, имеющие возможность из одного блока загружать другой, запускать его, задавать последовательность выполнения программных модулей.

На листинге управляющая подпрограмма начинается со строки 5000. Перед обращением к ней необходимо определить переменную A%, придав ей значение последовательности выполняемых команд с включением управляющих кодов, например код 10 — ВК (здесь и далее все значения кодов и адресов десятичные).

Учитывая возможные ошибки, возникающие в Бейсик-системе при работе с символьными переменными, лучше всего это сделать, определив соответствующую функцию пользователя (строки 90, 100).

В приводимом листинге из базовой программы вызывается модуль «TERMO 1», при этом символьная переменная A% принимает следующее значение (последовательность команд):

```
10 REM ××××××××××××××××××××××××××××
20 REM ×                            ×
30 REM × ОСНОВНАЯ ЧАСТЬ ПРОГРАММЫ  ×
40 REM ×                            ×
50 REM ××××××××××××××××××××××××××××
60 ......
70 ......
80 REM <<< ЗАДАНИЕ
          ПОСЛЕДОВАТЕЛЬНОСТИ КОМАНД >>>
90 DEF FNA$(A$)="NEW"+CHR$(10)+
    "CLOAD"+CHR$(34)+A$+CHR$(34)+",R"+
                        CHR$(10)+CHR$(3)
100 A$=FNA$("TT:TERMO1")
110 GOTO 5000
120 ......
130 ......
5000 REM <<< ПОДПРОГРАММА ЗАГРУЗКИ >>>
5010 DEF USR1=16128
5020 DEF USR2=16142
5030 DATA 5443,4930,5569,16222,
                        -27503,32450,135
5040 DATA 2551,2,289,5623,16222,66,
                        5599,16162,24
5050 DATA 135,9718,-18418,16,770,
                        95,-32694,5047
5060 DATA 40,-6665,2,34,9727,-30714,
                        28,757,-24640,24
5070 DATA 2743,20,9664,3,769,2,5599,
                        -32694,24,95,-24420,0
5080 I%=16128
5090 READ A%
5100 POKE I%,A%
5110 I%=I%+2%
5120 IF A% THEN 5090
5130 A$=USR1(A$) 'ПЕРЕСЫЛКА
                        ПЕРЕМЕННОЙ A$
5140 A%=USR2(A%) 'ЗАПУСК КОМАНД
                        НА ВЫПОЛНЕНИЕ
```

Если после выполнения вызываемого программного модуля необходим возврат в исходную программу, то символьную переменную A$ необходимо определить так:

```
NEW <BK>
CLOAD "TERMO1",R <BK>
```

Внимание! В конце значения переменной A$ обязательно должен стоять управляющий код 3!

Базовый программный модуль, их которого передается управление, должен сохранить свою работоспособность (или восстановить работоспособность) при выполнении команды NEW. Для этого кодовый блок необходимо располагать либо в буфере ввода-вывода с адреса 16128 (см. листинг), либо в младших адресах стековой области с адреса 256. Выбор места следует производить исходя из следующих соображений: если используется буфер ввода-вывода, то в вызываемом программном модуле должны отсутствовать операторы PRINT и INPUT#, нельзя также использовать команды LOAD и SAVE. Если же предпочтение отдано стековой области, необходимо учесть, что разрушение кодового блока может произойти при закрашивании сложных контуров оператором PAINT.

Чтобы перейти в стековую область, в приведенном листинге число 16222 (строки 5030 и 5040) нужно заменить на 350 и число 16162 (строка 5040) — на 290.

Кроме того, надо изменить следующие строки:

```
NEW <BK>
CLOAD "TERMO1",R <BK>
NEW <BK>
CLOAD "<имя исходного файла>",R <BK>

5010 DEF USR1=256
5020 DEF USR2=270
5080 I%=256
```

Если возникает необходимость в применении «жестких» сценариев, следует исключить строку 5140, задать символьную переменную A$ и после запуска подпрограммы записать сформированный новый управляющий блок на диск. При размещении в буфере ввода-вывода или в стековой области использовать соответственно команды

```
BSAVE "TT:<имя>",16142,16384
BSAVE "TT:<имя>",270,512
```


Более точно адрес конца программы подсчитывается для конкретного случая.

Запуск сценариев на выполнение производится командой

```
BLOAD "TT:<имя>",R
```

Е. ВОЛОДИН

БК-стихотворец

«Все мне пророчат счастье, но...
Вечная осень, багровые тона.
Устала я, устала я,
Хочу любить, все жду чего-то.
Вечная осень. Что ж — судьба!
Устала я. Надежда блекнет...»

Научить БК писать стихи совсем несложно, даже если он имеет только встроенный Фокал, не предусматривающий обработку символьных переменных.

Предлагаю программу, которая доставит немало удовольствия начинающим поэтам и графоманам. Она состоит из двух частей. Первая (строки 1.3—1.7) — создание словаря, вторая — «сочинение» и вывод на экран дисплея готовых стихотворений.

На запрос программы «СОЗДАНИЕ [1] или ИСПОЛНЕНИЕ [2]?:» выберите режим 1 и начинайте вводить слова, не забывая после каждого слова нажимать клавишу ⟨ВВОД⟩. Клавиша ⟨Q⟩ — конец работы.

```
1.10 X FCHR(12);A "СОЗДАНИЕ [1] ИЛИ
     ИСПОЛНЕНИЕ [2]?",A;X FCHR(12)
1.20 I (1-A)1.8
1.30 S I=0;S J=1;S W=0;S P=0;S L=1
1.40 S I=I+1;S A(I,J)=FCHR(FCHR(-1))
1.50 I (A(I,J)-81)1.60,2.60
1.60 I (A(I,J)-13)1.40,1.70,1.40
1.70 S M(J)=I-1;S J=J+1;S I=0;T " ";G 1.40
1.80 A "ФОРМАТ",X,"ДЛИНА",Y;X FCHR(12)
1.90 T "   ОПУС N ",%2.00,L,!!
2.10 S J=FABS(FITR(FRAN()*N))+1
2.20 F I=1,M(J);X FCHR(A(I,J))
2.30 S W=W+1;T " ";I (W-X)2.10;T !
2.40 S P=P+1;S W=0;I (P-Y)2.10;T !
2.50 S L=L+1;S P=0;I (FCHR(-1)-13)2.60,1.90
2.60 Q
```

Созданный массив сразу же запишите на магнитную ленту командой L О ИМЯ. Затем по команде М 2.1 войдите в режим редакт и рования строки и вместо переменной N подставьте число введенных слов. Снова запустите программу и переходите в режим 2. Запрос «ФОРМАТ»: обозначает требуемое количество слов в строке, а «ДЛИ-

Если после выполнения вызываемого программного модуля необходим возврат в исходную программу, то символьную переменную A$ необходимо определить так:

```
NEW <ВК>
CLOAD "TERMO1",R <ВК>
```

Внимание! В конце значения переменной A$ обязательно должен стоять управляющий код 3!

Базовый программный модуль, их которого передается управление, должен сохранить свою работоспособность (или восстановить работоспособность) при выполнении команды NEW. Для этого кодовый блок необходимо располагать либо в буфере ввода-вывода с адреса 16128 (см. листинг), либо в младших адресах стековой области с адреса 256. Выбор места следует производить исходя из следующих соображений: если используется буфер ввода-вывода, то в вызываемом программном модуле должны отсутствовать операторы PRINT и INPUT#, нельзя также использовать команды LOAD и SAVE. Если же предпочтение отдано стековой области, необходимо учесть, что разрушение кодового блока может произойти при закрашивании сложных контуров оператором PAINT.

Чтобы перейти в стековую область, в приведенном листинге число 16222 (строки 5030 и 5040) нужно заменить на 350 и число 16162 (строка 5040) — на 290.

Кроме того, надо изменить следующие строки:

```
NEW <ВК>
CLOAD "TERMO1",R <ВК>
NEW <ВК>
CLOAD "<имя исходного файла>",R <ВК>

5010 DEF USR1=256
5020 DEF USR2=270
5080 I%=256
```

Если возникает необходимость в применении «жестких» сценариев, следует исключить строку 5140, задать символьную переменную A$ и после запуска подпрограммы записать сформированный новый управляющий блок на диск. При размещении в буфере ввода-вывода или в стековой области использовать соответственно команды

```
BSAVE "TT:<имя>",16142,16384
BSAVE "TT:<имя>",270,512
```


Более точно адрес конца программы подсчитывается для конкретного случая.

Запуск сценариев на выполнение производится командой

```
BLOAD "TT:<имя>",R
```

Е. ВОЛОДИН

БК-стихотворец

«Все мне пророчат счастье, но...
Вечная осень, багровые тона.
Устала я, устала я,
Хочу любить, все жду чего-то.
Вечная осень. Что ж — судьба!
Устала я. Надежда блекнет...»

Научить БК писать стихи совсем несложно, даже если он имеет только встроенный Фокал, не предусматривающий обработку символьных переменных.

Предлагаю программу, которая доставит немало удовольствия начинающим поэтам и графоманам. Она состоит из двух частей. Первая (строки 1.3—1.7) — создание словаря, вторая — «сочинение» и вывод на экран дисплея готовых стихотворений.

На запрос программы «СОЗДАНИЕ [1] или ИСПОЛНЕНИЕ [2]?:» выберите режим 1 и начинайте вводить слова, не забывая после каждого слова нажимать клавишу ⟨ВВОД⟩. Клавиша ⟨Q⟩ — конец работы.

```
1.10 X FCHR(12);A "СОЗДАНИЕ [1] ИЛИ
     ИСПОЛНЕНИЕ [2]?",A;X FCHR(12)
1.20 I (1-A)1.8
1.30 S I=0;S J=1;S W=0;S P=0;S L=1
1.40 S I=I+1;S A(I,J)=FCHR(FCHR(-1))
1.50 I (A(I,J)-81)1.60,2.60
1.60 I (A(I,J)-13)1.40,1.70,1.40
1.70 S M(J)=I-1;S J=J+1;S I=0;T " ";G 1.40
1.80 A "ФОРМАТ",X,"ДЛИНА",Y;X FCHR(12)
1.90 T "   ОПУС N ",%2.00,L,!!
2.10 S J=FABS(FITR(FRAN()*N))+1
2.20 F I=1,M(J);X FCHR(A(I,J))
2.30 S W=W+1;T " ";I (W-X)2.10;T !
2.40 S P=P+1;S W=0;I (P-Y)2.10;T !
2.50 S L=L+1;S P=0;I (FCHR(-1)-13)2.60,1.90
2.60 Q
```

Созданный массив сразу же запишите на магнитную ленту командой L О ИМЯ. Затем по команде М 2.1 войдите в режим редакт и рования строки и вместо переменной N подставьте число введенных слов. Снова запустите программу и переходите в режим 2. Запрос «ФОРМАТ»: обозначает требуемое количество слов в строке, а «ДЛИ-