Приложение 4. Пример организации связей с языком MSX BASIC. "Универсальное меню"

Приведенная ниже программа на языке MSX BASIC вызывает подпрограмму на языке ассемблера и передает ей адрес строкового массива, используя функцию `VARPTR`. Массив представляет собой меню. Строка массива, начинающаяся с символа пробел, считается комментарием. Пустая строка ("") является признаком конца меню.

Ассемблирование в системе DUAD описано здесь.

Подпрограмма на ассемблере рисует на экране (параметры `SCREEN 0:WIDTH 80` — устанавливаются заранее) окно, размеры которого определяются программно, исходя из параметров переданного массива (Y — по количеству элементов массива, т.е. до пустой строки, X — по наиболее длинной строке). Координаты окна программа на языке MSX BASIC передаёт функцией `LOCATE X,Y`, используя тот факт, что интерпретатор записывает значения X и Y в системные ячейки.

Запуск программы:

```
run"wnd.bas"
```

Строка-курсор устанавливается на первый возможный вариант выбора меню (комментарии пропускаются). Эту строку можно перемещать в пределах окна с помощью клавиш вверх/вниз. Выбор осуществляется нажатием клавиши ввод.

Выбранная строка отмечается галочкой, и программе на языке MSX BASIC возвращается её номер. Если в момент выбора были нажаты клавиши CTRL+Stop, возвращается ноль. Обратите внимание:

1. Пользователь MSX BASIC должен сам следить за размещением окна на экране.
2. В меню должен быть хотя бы один возможный выбор.

Архив с готовыми файлами

- `wnd.asm` — исходный код программы на ассемблере
- `wnd.lst` — листинг программы на ассемблере
- `wnd.bas` — программа на MSX BASIC
- `wnd.obj` — оттранслированная программа на ассемблере

1. BASIC-программа

wnd.bas

```
 0 rem (c) 1990 И.Бочаров
10 CLEAR 200,&HDC00              ' Резервируем память для подпр.
20 DIM A$(17)                    ' Определяем массив-меню
30 BLOAD"wnd.obj"                ' Загружаем подпрограмму
40 VDP(13)=&HA4:COLOR = (4,0,0,4)' Устанавливаем цвета окна
50 VDP(14)=&HF0
60 DATA "     Select:"," ===============",Brief,Full,Info,Tree,
On/Off        ^F1," ---------------",Name,Extension,Size,Time,
UnsORted," ---------------",Help infORmation,Exit to main menu,""
70 FOR I=0 TO 16                 ' Считываем меню
80    READ A$(I)
90 NEXT
100 DEFUSR=&HDC00                ' Адрес подпрограммы-меню
110 DEFUSR1=&HDC03               ' Подпрограмма очистки экрана
130 :
140 LOCATE 2,1                   ' Устанавливаем координаты окна
150 GOSUB 230                    ' Вызываем подпрограмму
160 LOCATE 20,6
170 GOSUB 230
180 LOCATE 26,0
190 GOSUB 230
```

```
200 GOTO 140 ' Зацикливаемся
210 :
230 A=USR1(0) ' Очищаем цвета на экране
240 A=USR(VARPTR(A$(0))) ' Вызываем подпрограмму
250 IF A THEN LOCATE 75,22:PRINT A;:RETURN ELSE CLS:PRINT USR1(
"<Ctrl><Stop>"):END              ' Обрабатываем результат
260 rem The end.
```

2. Подпрограмма на языке ассемблера

wnd.asm

```
;                (c) 1990 by IgOR BOCHAROV.
;
;-------------- Константы и адреса -------------------------
;
cr      EQU     13              ; Клавиша "Конец выбора"
Up      EQU     30              ; Клавиша вверх
Down    EQU     31              ; Клавиша вниз
TheBeg  EQU     0DC00h          ; Отсюда наша программа трансл.
GetHL   EQU     2F8Ah           ; Подпрограммы передачи значен.
PutHL   EQU     2F99h
Base1   EQU     0F3B5h          ; Здесь хран. адрес табл. цвет.
CsrX    EQU     0F3DDh          ; Сюда  LOCATE  пишет  X
CsrY    EQU     0F3DCh          ; А сюда  Y
RdVram  EQU     004Ah           ; Чтение из  VRAM
WrVrHL  EQU     004Dh           ; Запись во  VRAM
SetWrt  EQU     0053h           ; Установка  VDP  для записи
ChSns   EQU     009Ch           ; Опрос состояния буфера клав.
ChGet   EQU     009Fh           ; Взятие символа из буфера
BreakX  EQU     00B7h           ; Опрос  <CTRL>+<Stop>
;-----------------------------------------------------------------
        ORG     TheBeg
;------------------ Точки входа ----------------------------
        JP      Menu            ; Универсальное меню
        JP      ClMenu          ; Очищение таблицы цветов
;------------- Вычисляем размер окна ------------------------
Menu:   CALL    GetHL           ; [HL] - адрес первого элемента
        PUSH    HL              ; Адрес первого элемента
        LD      IY,Variab       ; Адрес блока переменных
        XOR     a               ; Инициируем переменный
        LD      (IY+dy),a       ; Число строк
        LD      (IY+dx),a       ; dx
l001:   LD      a,(HL)          ; Длина очередной строки
        AND     a               ; Конец?
        JR      Z,l002          ; Если да, то выходим
        INC     (IY+dy)         ; Увеличиваем число строк
        INC     HL              ; + длина
        INC     HL              ; + адрес
        INC     HL
        CP      (IY+dx)         ; Больше dx?
        JR      C,l001          ; Если нет, то dx не меняем
        LD      (IY+dx),A
        JR      l001
;-------------- Вычисляем координаты окна -------------------
l002:   LD      HL,(CsrY)       ; x y
        DEC     H
        DEC     L
        LD      C,H             ; x
        LD      B,0
        LD      H,B
        ADD     HL,HL           ; * 2
        ADD     HL,HL           ; * 4
```

```
        ADD     HL,HL           ; * 8
        ADD     HL,HL           ; * 16
        LD      E,L
        LD      D,H
        ADD     HL,HL           ; * 32
        ADD     HL,HL           ; * 64
        ADD     HL,DE           ; * 80
        ADD HL,BC
        EX      DE,HL
        INC     (IY+dx)         ; dx
        INC     (IY+dx)         ; dx
        LD      b,(IY+dx)       ; dx
        LD      c,(IY+dy)       ; dy
        PUSH    DE              ; Для печати текста
;------------------ Рисуем окно ----------------------------
; [DE] - x + y*80, [b] - dx, [c] - dy
        PUSH    BC
        PUSH    DE              ; Для подсветки
; Рисование окантовки окна
        LD      a,17h           ;  ─
        LD      HL,1819h        ; ┌ ┐
        CALL    WndL01
WndL02:
        LD      a,' '           ; ' '
        LD      HL,1616h        ; │ │
        CALL    WndL01
        DEC     c
        JR      NZ,WndL02
        LD      a,17h           ;  ─
        LD      HL,1A1Bh        ; └ ┘
        CALL    WndL01
;--- Теперь подсвечиваем ------------
NEXtColORInWindow:
        POP     HL              ; x,y
        POP     DE              ; dx,dy
        INC     D
        INC     D
        INC     D               ; dx = dx + 4
        INC     D
        INC     E               ; dy = dy + 2
        INC     E
WndLp1:
        PUSH    DE
        PUSH    HL
        CALL    FillColORTable  ; Заполняем строку
        POP     HL
        LD      DE,80           ; [y] = [y] + 1
        ADD     HL,DE
        POP     DE
        DEC     E               ; [dy] = [dy] - 1
        JR      NZ,WndLp1
;------------------ Пишем текст в окне ---------------------
        POP     HL
        LD      DE,80*1+3       ; Откуда начать писать
        ADD     HL,DE
        LD      D,H
        LD      E,L
        EX      (SP),HL         ; Адрес первой строки
l003:   LD      A,(HL)          ; Длина очередной строки
        AND     A               ; Конец?
        JR      Z,l004          ; Если да, то выходим
        LD      B,A             ; Текущий  dx
        PUSH    HL
        PUSH    DE
```

```
        DI
        LD      A,E             ; Младший байт адреса
        OUT     (99h),A
        LD      A,D             ; затем старший байт
        OR      40h             ; выставив 6 бит в 1
        OUT     (99h),A
        INC     HL
        LD      A,(HL)          ; Получаем адрес строки
        INC     HL
        LD      H,(HL)
        LD      L,A
        LD      C,98h           ; загрузить номер порта  VDP
        OTIR                    ; вывести блок
        EI
        POP     HL
        LD      DE,80
        ADD     HL,DE
        EX      DE,HL           ; [DE] -  следующий адрес
        POP     HL
        INC     HL
        INC     HL
        INC     HL
        JR      l003            ; Повторяем вывод
;------------------ Выбор в меню ----------------------------
l004:   POP     HL              ; Адрес первой строки
        LD      C,0             ; Текущее состояние
        LD      D,C
        LD      E,D
l007:   PUSH    HL
l008:   ADD     HL,DE
        CALL    RdVram
        CP      ' '             ; Пробел?
        JR      NZ,l006
        LD      DE,80
        JR      l008
l006:   POP     DE
        CP      17h             ; —?
        JR      NZ,l009
        EX      DE,HL
        DEC     C
l009:   INC     C
        PUSH    BC              ; Состояние
        PUSH    HL              ; Стираем засветку
        LD      D,(IY+dx)
        DEC     HL
        CALL    ClearColORTable
        POP     HL
        POP     BC
;-------------------
InputKey:
        CALL    BreakX
        JR      NC,Input
        LD      C,0
        JR      Exit
Input:  CALL    ChSns           ; Есть что-нибудь в буфере?
        JR      Z,Inputkey
        CALL    ChGet           ; Берем символ
;-------------------
        CP      cr
        JR      NZ,CheckUp
        LD      A,'√'
        DEC     HL
        CALL    WrVrHL
Exit:   LD      H,0
```

```
        LD      L,C             ; Текущее состояние
        JP      PutHL
;-------------------
CheckUp:
        CP      Up
        JR      NZ,CheckDown
        LD      A,C
        DEC     A               ; А можно ли наверх?
        JR      Z,InputKey
        LD      C,A
        PUSH    BC
        PUSH    HL
        LD      D,(IY+dx)
        DEC     HL
        CALL    FillColORTable  ; Стираем курсор
        POP     HL
        LD      DE,-80          ; Смещение на строку вверх
l010:   ADD     HL,DE           ; Новая позиция курсора
        CALL    RdVram
        CP      ' '
        JR      Z,l010          ; Сканируем пробелы
        LD      D,(IY+dx)
        PUSH    HL
        DEC     HL
        CALL    ClearColORTable
        POP     HL
        POP     BC
        JR      InputKey
;-------------------
CheckDown:CP    Down
        JR      NZ,InputKey
        PUSH    HL
        LD      D,(IY+dx)
        PUSH    BC
        DEC     HL
        CALL    FillColORTable  ; Стираем курсор
        POP     BC
        POP     HL
        LD      DE,80
        JR      l007
;---------------------------------------------------------------
WndL01:
        PUSH    BC
        PUSH    DE
        PUSH    AF
        PUSH    BC
        LD      A,' '
        CALL    WrVram
        LD      A,H
        CALL    WrVram
        POP     BC
        POP     AF
        CALL    FillVm
        LD      A,L
        CALL    WrVram
        LD      A,' '
        CALL    WrVram
        POP     HL
        LD      BC,80
        ADD     HL,BC
        EX      DE,HL
        POP     BC
        RET
;---------------------------------------------------------------
```

```
; Рисует цветную линию по координатам и размеру
; [HL] - y *80+ x; [d] - dx
; MoDIfy: AF, BC, DE, HL
FillColORTable:
        LD      E,10000000b
        CALL    GetMaskColOR    ; Получить адрес и маску
WndLp4: CALL    RdVram          ; считать текущий байт
WndLp3: OR      E
        DEC     D               ; [dx] = [dx] - 1
        JP      z,WrVrHL
        RRC     E               ; Следующая маска
        JR      NC,WndLp3       ; Если не вышли из байта, то не
                ; пишем
        CALL    WrVrHL          ; Иначе записываем
        INC     HL              ; Следующий адрес  VRAM
        JR      WndLp4
;-----------------------------------------------------------------
; Очищает цветную линию
; [HL] - y *80+ x; [d] - dx
; MoDIfy: AF, BC, DE, HL
ClearColORTable:
        LD      E,01111111b
        CALL    GetMaskColOR    ; Получить адрес  VRAM  и маску
WndLp5: CALL    RdVram          ; считать текущий байт
WndLp6: AND     E
        DEC     D               ; [dx] = [dx] - 1
        JP      Z,WrVrHL
        RRC     E               ; Следующая маска
        JR      C,WndLp6        ; Если не вышли из байта, то не
                ; пишем
        CALL    WrVrHL          ; Иначе записываем
        INC     HL              ; Следующий адрес  VRAM
        JR      WndLp5
;-----------------------------------------------------------------
; По координатам выдает адрес  VRAM  и маску
; [HL] - y*80 + x;
; [HL] - адрес Vram; [e] - маска
GetMaskColOR:
        LD      A,L             ; Получаем из коор. физ. адрес
        LD      B,3
        SRL     H               ; [HL] = [HL] / 8
        RR      L
        DJNZ    $-4
        LD      BC,(Base1)
        ADD     HL,BC           ; Физический адрес  VRAM
        CPl                     ; [a] = not([a])
        AND     00000111b       ; Бит, который надо установить
        LD      B,A
        INC     B
        RLC     E
        DJNZ    $-2
        RET
;-----------------------------------------------------------------
; Очистка цветной таблицы
ClMenu: LD      DE,(Base1)
        LD      B,24*10
        XOR     A
;-----------------------------------------------------------------
; заполнение VRAM const [a], len [b], adr [DE]
FillVm: DI
        PUSH    AF
        LD      A,E
        OUT     (99h),A
        LD      A,D
```

```
        OR      40h
        OUT     (99h),A
        POP     AF
        EX      (SP),HL
        EX      (SP),HL
        OUT     (98h),A
        INC     DE
        djNZ    $-3             ; повторить, если надо
        RET
;-----------------------------------------------------------------
WrVram: EX      DE,HL
        CALL    WrVrHL
        EX      DE,HL
        INC     DE
        RET
;-----------------------------------------------------------------
Variab  EQU     $
dy      EQU     1
dx      EQU     2
        END
```

---

http://sysadminmosaic.ru/msx/assembler_programming_guide-fakhrutdinov_bocharov/14

2020-11-25 09:55