Московский государственный технический университет им. Н.Э.Баумана

Защищено:
Большаков С.А.

"__"_____2024 г.

Демонстрация ЛР:
Большаков С.А.

"__"_____2024 г.

Отчет по лабораторной работе № 5 по курсу Системное программирование

"Ввод/вывод в адреса и числа"

(есть ли дополнительные требования - НЕТ)

8
(количество листов)
Вариант № 11

ИСПОЛНИТЕЛЬ:

студент группы **ИУ5-41Б** _____
(подпись)

Ларкин Б. В.

"__"_____2024 г.

Москва, МГТУ - 2024

СОДЕРЖАНИЕ

1. Цель выполнения лабораторной работы № 5

Лабораторная работа №5 выполняется для получения навыков разработки программ и процедур на Ассемблере, использующих массивы, вложенные циклы и буферизацию ввода.

2. Порядок и условия проведения работы № 5

Разработать и отладить программу на языке Ассемблер для **ввода и буферизации строки символов** с клавиатуры (последовательности символов) и затем последовательного их вывода на экран в шестнадцатеричном представлении (через пробел). В данной программе для корректной работы необходимо предусмотреть запоминание строки символов в байтовом массиве. Программа и блок-схема должны содержать вложенные циклы (двойные циклы). Программу оформить в виде исполнимого ***.COM файла.**

Признак завершения ввода отдельной строки с клавиатуры – это символ "$" (он вводится с клавиатуры для завершения ввода строки). Между введенной строкой символов и их шестнадцатеричным представлением должен располагаться знак равенства (“=”). Максимальное число вводимых символов не должно превышать 20-ти. В данной программе цикл ввода (с клавиатуры) организуется с помощью команд условного (JE, JNE) перехода и команды безусловного перехода (JMP). После завершения ввода строки выполняется ее автоматический вывод. Организовать цикл ввода строк до ввода специального символа ('*'). Пример результата работы одного цикла программы показан ниже:

АБВ$ = 80 81 82

Требования к процедурам и их именованию совпадают с требованием предыдущих ЛР. Программа должна работать в циклическом режиме ввода строк (для внешнего цикла используется команда LOOP): после ввода одной строки запрашивается следующая (максимальное число вводимых строк для одного запуска программы равно 10). Завершение цикла ввода строк может быть выполнено при вводе символа звездочка (“*”), который должен быть введен в первой позиции строки. Вводимые символы строки записываются в символьный массив (буфер символов), максимальное число введенных символов равно 20-ти. Цикл ввода строки организуется командами условного и безусловного перехода. При вводе нужно подсчитать число введенных символов, включая символ доллара (“$”). Для вывода организуется цикл с помощью команды цикла (LOOP). В программе использовать процедуры предыдущих лабораторных данного цикла (ввода символа, печати, перевода строки и др.).

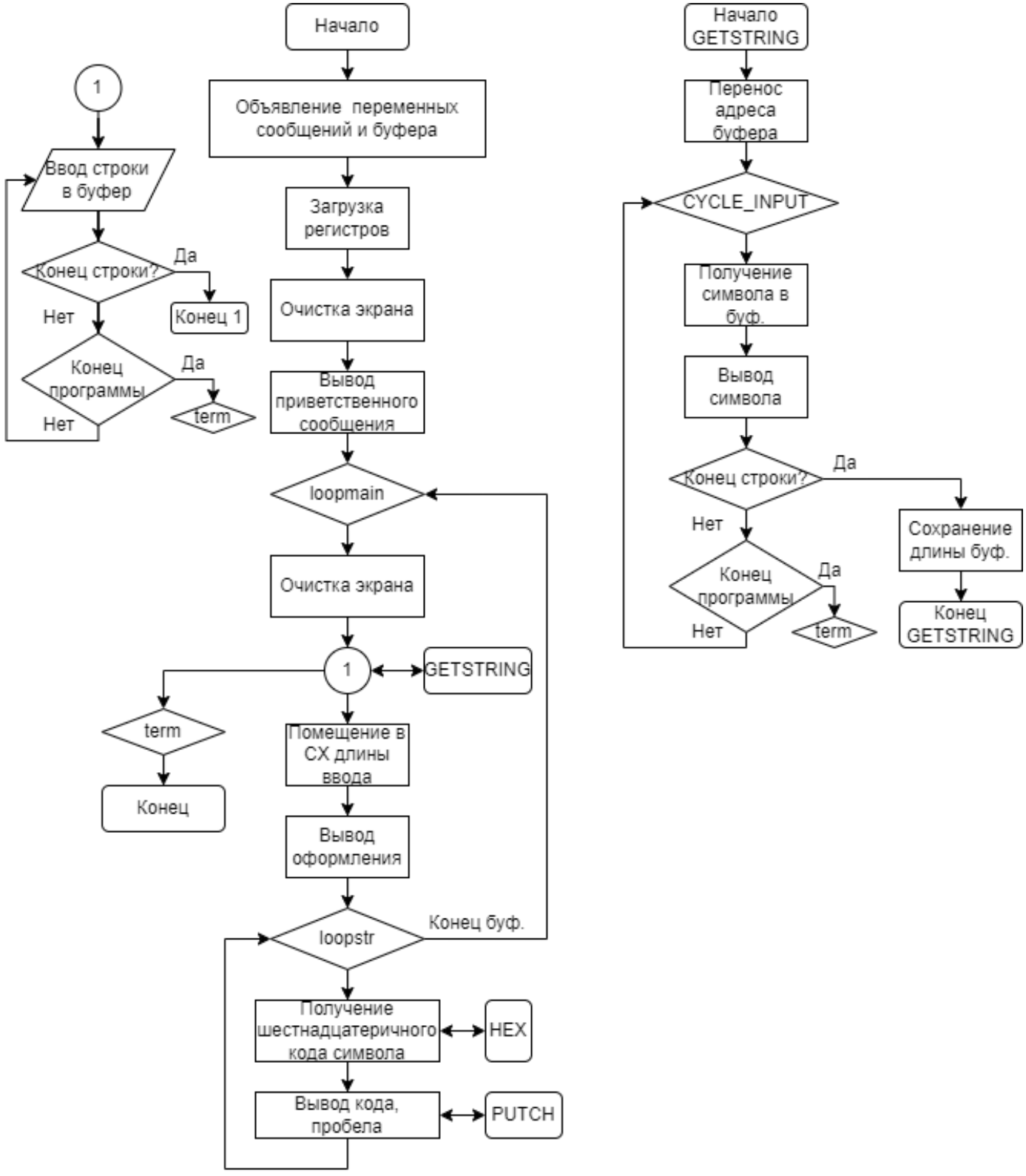
Для ввода/вывода строки и ее шестнадцатеричного представления разрабатываются дополнительная процедура **HEX** (см. ЛР №4). Организовать очистку экрана до начала работы программы, а также после ее завершения (С помощью специальной процедуры - CLRSCR).

В данной программе необходимо отдельно объявить отдельно сегмент данных (**DTSEG**) и сегмент стека (**STSEG**). Проверить загрузку сегментного регистра данных (DS) с помощью команды пересылки (MOV), но через промежуточный регистр (AX).

3. Описание ошибок, возникших при отладке ЛР № 5

№ п/п	Проявление ошибки	Причина ошибки	Способ устранения
1.	Несоответствие типов при проведении операции MOV	Попытка сложения WORD с BYTE	MOV через временный регистр AH, операция byte[]
2.	Отсутствие функционала объявленных в DATASEGMENT переменных	Отсутствие объявления DATASEG после CODESEG, отсутствие его в ASSUME	Добавление объявления в конце и в ASSUME.

4. Блок-схема программы



5. Текст программы на языке Ассемблера (.LST)

Turbo Assembler Version 3.1 04/22/24 24:44:01
l5.asm

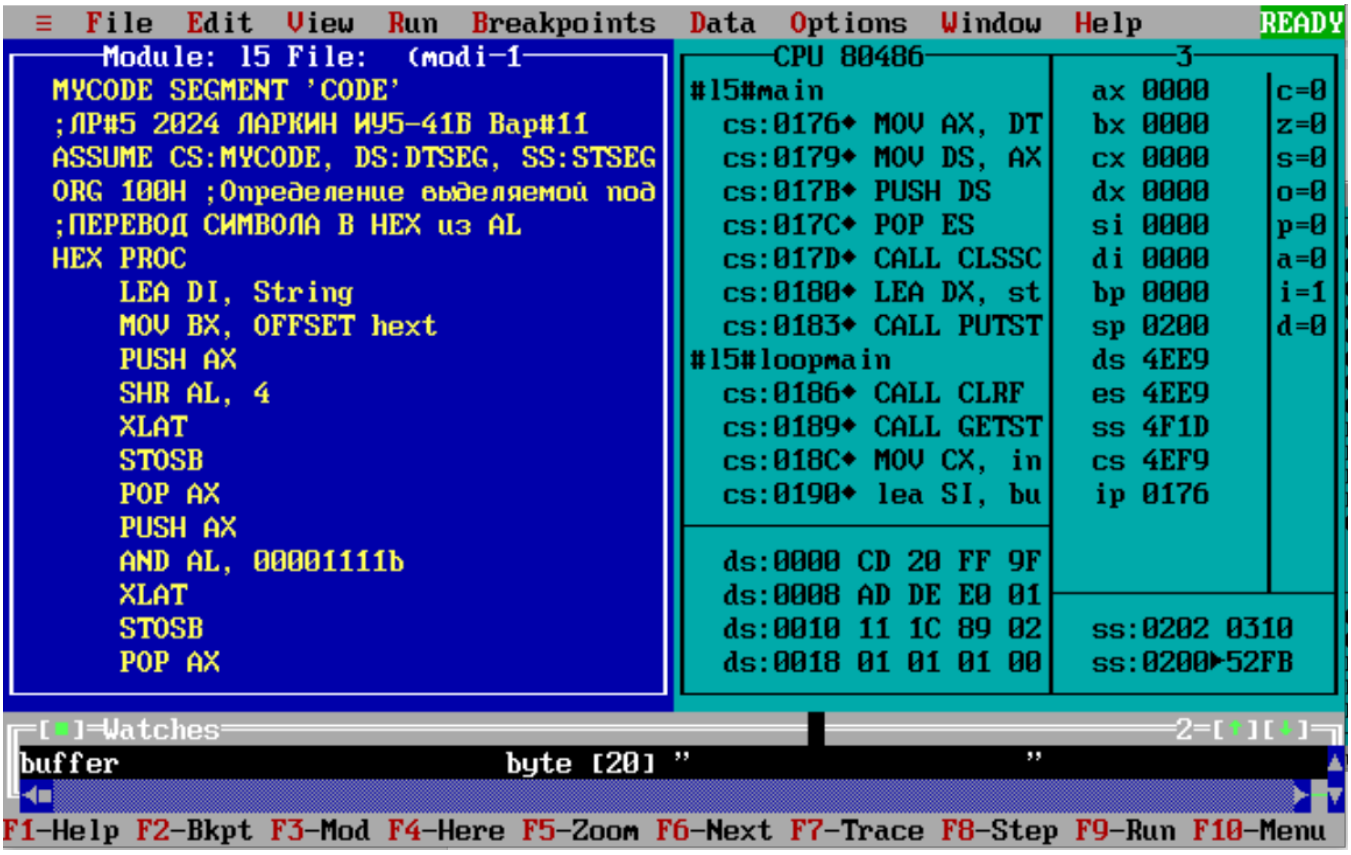
```
1    0000                    MYCODE SEGMENT 'CODE'
   2                         ;ЛР#5 2024   ЛАРКИН ИУ5-41Б Вар#11
   3                         ASSUME CS:MYCODE, DS:DTSEG, SS:STSEG
   4                         ORG 100H ;Определение выделяемой под PSP памяти
   5                         ;ПЕРЕВОД СИМВОЛА В    HEX из AL
   6    0100                 HEX PROC
```

```
 7    0100  BF 0000r                  LEA DI, String
 8    0103  BB 000Dr                  MOV BX, OFFSET        hext
 9    0106  50                  PUSH AX
10    0107  D0 E8 D0 E8 D0 E8          D0+       SHR AL, 4
11          E8
12    010F  D7                        XLAT
13    0110  AA                        STOSB
14    0111  58                  POP AX
15    0112  50                  PUSH AX
16    0113  24 0F                     AND AL, 00001111b
17    0115  D7                        XLAT
18    0116  AA                        STOSB
19    0117  58                  POP AX
20    0118  B8 0068                   MOV AX, 'h'
21    011B  AB                        STOSW
22    011C  B8 0024                   MOV AX, '$'
23    011F  AB                        STOSW
24    0120  BA 0000r                  LEA DX, STRING
25    0123  C3                        RET
26    0124                      HEX ENDP
27
28    0124                      GETCH PROC
29    0124  B4 08                    MOV AH,          08H
30    0126  CD 21                    INT 21H
31    0128  C3                       RET
32    0129                      GETCH ENDP
33
34    0129                      CLSSCR PROC
35    0129  53                 PUSH BX
36    012A  51                        PUSH CX
37    012B  50                        PUSH AX
38    012C  B8 0600                   MOV   AX,0600H          ;Запрос на очистку экрана
39    012F  B7 07                     MOV   BH,07     ;ЧБ гамма
40    0131  B9 0000                   MOV   CX,0000  ;Верхняя левая позиция.
41    0134  BA 184F                   MOV   DX,184FH          ;Нижняя правая позиция.
42    0137  CD 10                     INT   10H    ;БИОС
43    0139  58                  POP AX
44    013A  59                        POP CX
45    013B  5B                        POP BX
46    013C  C3                        RET
47    013D                      CLSSCR ENDP
48
49    013D                      PUTCH PROC
50    013D  B4 02                     MOV AH, 02H
51    013F  CD 21                     INT 21H
52    0141  C3                        RET
53    0142                      PUTCH ENDP
54
55    0142                      CLRF PROC
56    0142  B2 0A                     MOV DL, 10
57    0144  E8 FFF6                   CALL PUTCH
58    0147  B2 0D                     MOV DL, 13
59    0149  E8 FFF1                   CALL PUTCH
60    014C  C3                        RET
61    014D                      CLRF ENDP
62
63    014D                      PUTSTR PROC
64    014D  B4 09                     MOV AH, 09h
65    014F  CD 21                     INT 21H
66    0151  C3                        RET
67    0152                      PUTSTR ENDP
68
69                              ;Возвращает по адресу DS:DX буфер строки
70    0152                      GETSTRING PROC
71    0152  BF 001Fr                  LEA DI, buffer  ;DS:DX указывает на буфер строки
72    0155  B9 0014                   MOV CX, 20
```

```
   73   0158                            CYCLE_INPUT:
   74   0158  E8 FFC9                     CALL GETCH
   75   015B  8B D0                      MOV DX, AX
   76   015D  E8 FFDD                    CALL PUTCH
   77   0160  3C 24                      CMP AL, '$'
   78   0162  74 09                      JE     endofstring
   79   0164  3A 06 0033r                CMP AL, BREAK_SYMBOL
   80   0168  74 43                      JZ     term
   81   016A  AA                         STOSB
   82   016B  E2 EB                    LOOP CYCLE_INPUT
   83   016D                            endofstring:
   84   016D  B8 0014                    MOV AX, 20
   85   0170  2B C1                      SUB AX, CX
   86   0172  A3 001Dr                   MOV inputLength, AX
   87   0175  C3                       RET
   88   0176                         GETSTRING ENDP
   89
   90   0176                         MAIN PROC
   91                                ; Загрузка      регистров данных
   92   0176  B8 0000s                  MOV AX, DTSEG
   93   0179  8E D8                     MOV DS, AX
   94   017B  1E                        PUSH DS
   95   017C  07                        POP ES
   96                                ; Вывод символов на экран
   97   017D  E8 FFA9                   CALL CLSSCR
   98   0180  BA 0034r                  LEA DX, start
   99   0183  E8 FFC7                   CALL PUTSTR
  100   0186                            loopmain:
  101   0186  E8 FFB9                    CALL CLRF
  102   0189  E8 FFC6                    CALL GETSTRING
  103   018C  8B 0E 001Dr                MOV CX, inputLength ;Число  символов
  104   0190  BE 001Fr                   lea SI, buffer
  105   0193  B2 3D                      MOV DL, '='
  106   0195  E8 FFA5                    CALL PUTCH
  107   0198                            loopstr:
  108   0198  AC                             LODSB
  109   0199  E8 FF64                        CALL HEX
  110   019C  B4 09                          MOV AH, 09h
  111   019E  CD 21                          INT 21H
  112   01A0  E8 FF9A                        CALL PUTCH
  113   01A3  E2 F3                     loop loopstr
  114   01A5  E8 FF7C                   CALL GETCH
  115   01A8  E8 FF7E                   CALL CLSSCR
  116   01AB  EB D9                     jmp loopmain
  117   01AD                         ENDP MAIN
  118                                ; Выход из     программы
  119   01AD                         term:
  120                                     ;Выход с кодом 0
  121   01AD  B0 00                       MOV AL, 0
  122   01AF  B4 4C                       MOV AH, 4CH
  123   01B1  CD 21                       INT 21H
  124   01B3                         MYCODE ENDS
  125
  126   0000                         DTSEG SEGMENT
  127   0000  20 20 20 20 20 20   20+     String   db '             $',0
  128         20 20 20 20 24 00
  129   000D  30 31 32 33 34 35   36+     hext DB '0123456789ABCDEF'
  130         37 38 39 41 42 43   44+
  131         45 46
  132   001D  0014                        inputLength dw  20  ; number of    read characters
  133   001F  14*(00)                     buffer   db 20 DUP(0) ; actual buffer
  134   0033  2A                          BREAK_SYMBOL DB '*'           ;Символ, по которому будет производится
выход
  135   0034  82 A2 A5 A4 A8 E2        A5+     start DB "Введите доллар для окончания строки;  '*' - для окончания
работы$"
  136         20 A4 AE AB AB A0 E0+
```

```
  137        20 A4 AB EF 20 AE   AA+
  138        AE AD E7 A0 AD A8   EF+
  139        20 E1 E2 E0 AE AA   A8+
  140        3B 20 27 2A 27 20   2D+
  141        20 A4 AB EF 20 AE   AA+
  142        AE AD E7 A0 AD A8   EF+
  143        20 E0 A0 A1 AE E2   EB+
  144        24
  145  0074                      DTSEG ENDS
  146
  147  0000                      STSEG SEGMENT STACK 'STACK'
  148  0000  0100*(0000)         DW 256 DUP(0)
  149  0200                      STSEG ENDS
  150
  151                            END MAIN

Symbol Name             Type   Value                    Cref    (defined at #)

??DATE                          Text   "04/22/24"
??FILENAME              Text   "l5          "
??TIME                  Text   "24:44:01"
??VERSION               Number 030A
@CPU                    Text   0101H
@CURSEG                         Text   DTSEG                    #1  #128  #132
@FILENAME               Text   L5
@WORDSIZE               Text   2                    #1  #128  #132
BREAK_SYMBOL                    Byte   DTSEG:0033                79  #140
BUFFER                          Byte   DTSEG:001F                71  106  #139
CLRF                    Near   MYCODE:0142              #55    103
CLSSCR                          Near   MYCODE:0129               #34    99  117
CYCLE_INPUT             Near   MYCODE:0158              #73    82
ENDOFSTRING             Near   MYCODE:016D              78  #83
GETCH                   Near   MYCODE:0124              #28    74  116
GETSTRING               Near   MYCODE:0152              #70    104
HEX                     Near   MYCODE:0100              #6  111
HEXT                    Byte   DTSEG:000D              8  #135
INPUTLENGTH             Word   DTSEG:001D              86  105  #138
LOOPMAIN                Near   MYCODE:018B              #102    118
LOOPSTR                         Near   MYCODE:019D               #109    115
MAIN                    Near   MYCODE:0176              #90    153
PUTCH                   Near   MYCODE:013D              #49    57  59    76  108   114
PUTSTR                          Near   MYCODE:014D               #63    101
START                   Byte   DTSEG:0034              100    #141
STRING                          Byte   DTSEG:0000                7  24  #133
TERM                    Near   MYCODE:01B2              80  #121

Groups & Segments       Bit Size Align     Combine          Class       Cref    (defined at #)

DTSEG                   16  0074 Para      none                3  92  #132
MYCODE                          16  01B8 Para      none      CODE      #1  3
STSEG                   16  0100 Para      Stack               3  94  #128
```

6. Скриншот программы в TD.exe



7. Результаты работы программы

```
Введите доллар для окончания строки; '*' - для окончания работы
wdwr$ = 77h 64h 77h 72h
wefuyer$ = 77h 65h 66h 75h 79h 65h 72h
rijgreintno$ = 72h 69h 6Ah 67h 72h 65h 69h 6Eh 74h 6Eh 6Fh
rdbt3847nd584$ = 72h 64h 62h 74h 33h 38h 34h 37h 6Eh 64h 35h 38h 34h
abc123$ = 61h 62h 63h 31h 32h 33h
*
```

8. Выводы по ЛР № 5

Разработан файл .ASM и соответствующие файлы приложения и листинга на языке Ассемблер. Программа выполняется в циклическом режиме до ввода ‘*’, выводя по каждой введенной последовательности, заканчивающейся знаком ‘$’, шестнадцатеричные кодировки каждого из символов этой последовательности, разделенные пробелами. Программа работает корректно, мы изучили буферизацию ввода и работу с адресами в ней.