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 № 3 по курсу**

**Системное программирование**

#### "Вывод трех символов"

#### (есть ли дополнительные требования - ДА)

8

(количество листов)

Вариант № 20

| ИСПОЛНИТЕЛЬ: |  |
| --- | --- |
| студент группы **ИУ5-41Б** | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| **Цыпышев Т.А.** | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Москва, МГТУ - 2024

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**СОДЕРЖАНИЕ**

[1. Цель выполнения лабораторной работы № 3 3](#_gjdgxs)

[2. Порядок и условия проведения работы № 3 3](#_30j0zll)

[3. Описание ошибок, возникших при отладке № 3 3](#_1fob9te)

[4. Блок-схема программы 4](#_3znysh7)

[5. Текст программы на языке Ассемблера (.LST) 5](#_2et92p0)

[6. Скриншот программы в TD.exe 8](#_tyjcwt)

[7. Результаты работы программы 8](#_3dy6vkm)

[8. Выводы по ЛР № 3 8](#_1t3h5sf)

# Цель выполнения лабораторной работы № 3

Лабораторная работа №3 выполняется для получения навыков разработки и отладки программ на ЯП, получения базовых знаний об использовании прерываний, процедур и регистров на Ассемблере, изучения и использования компонентов системы программирования Ассемблер (компилятора, редактора связей, отладчика) и получения навыков оформления документации по программным разработкам, реализуемым на языке.

# Порядок и условия проведения работы № 3

Разработать и отладить программу на языке Ассемблер для вывода на экран дисплея трех первых заглавных русских букв (A, Б, В), на трех отдельных строках дисплея подряд (отдельно программируется перевод строки и возврат каретки!).

После завершения вывода букв на экран организовать ожидание ввода любого символа с клавиатуры (нажатие клавиши).

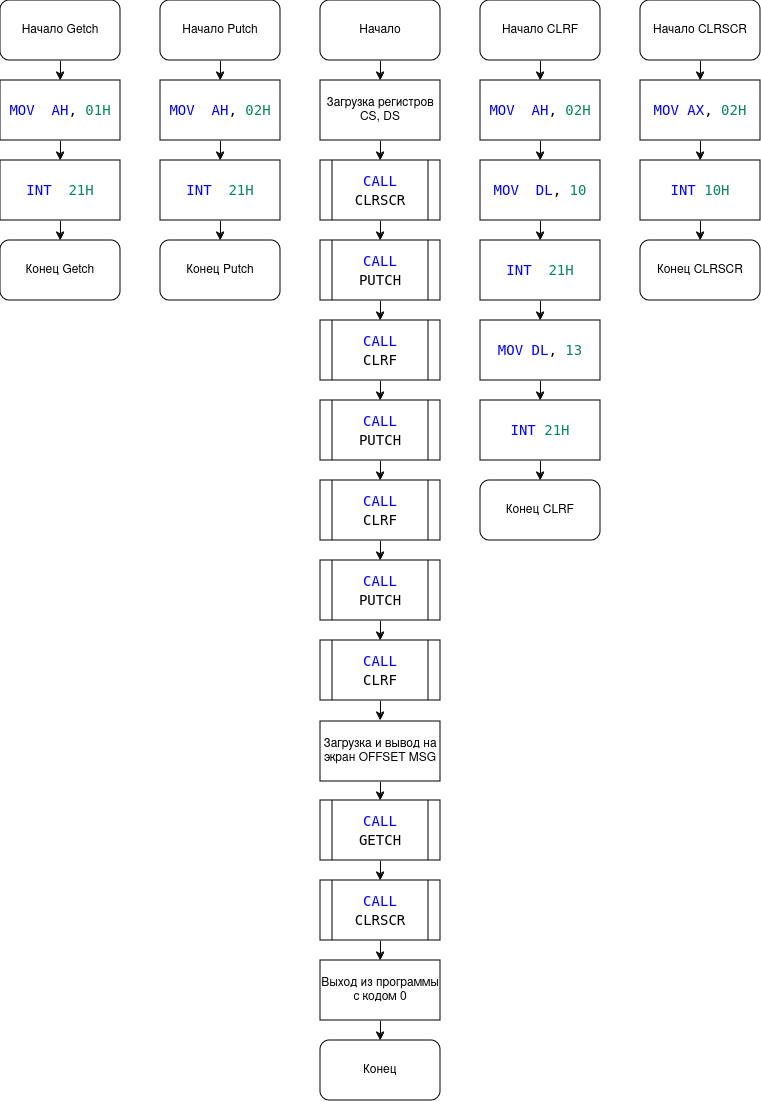
Необходимо использовать процедуры при разработке программы. Предусмотреть минимально три процедуры: для ввода символа (1-я процедура - **GETCH** название процедуры ввода символа желательно взять такое название), для вывода одного символа (2-я процедура - **PUTCH**) и для перевода строки с возвратом каретки (3-я процедура - **CLRF**) на дисплее (оформление процедур - **PROC - ENDP**, вызов процедур - **CALL**).

В программе организовать очистку экрана до начала вывода символов, а также после завершения работы программы. Очистка экрана должна выполняться отдельной дополнительной процедурой на языке Ассемблер (название ее - **CLRSCR**). Очистка экрана должна быть выполнена без организации циклов вывода символов с помощью соответствующего прерывания (найденного вами в справочнике). При выполнении дополнительных требований в текст программы добавляется специальный комментарий, подтверждающий их выполнение. На титульном листе отчета нужно отметить факт выполнения ЛР с дополнительными требованиями.

# Описание ошибок, возникших при отладке № 3

| № п/п | Проявление ошибки | Причина ошибки | Способ устранения |
| --- | --- | --- | --- |
|  | Некорректная работа функции очистки строки | Отсутствие символа возврата каретки перед перед прерыванием для вывода символа | Добавление строки:  MOV DL, 13  перед:  INT 21H |

# Блок-схема программы



# Текст программы на языке Ассемблера (.LST)

Turbo Assembler Version 3.1 04/02/24 20:59:48 Page 1

c:\lab3\lab.asm

1 ;Цыпышев Т.А. ИУ5-41 Вар. 20

2 ;=================================================================================

3 0000 PRGR SEGMENT

4 ASSUME CS:PRGR, DS:DATA ; Устанавливаем соответствие +

5 сегментов кода и данных

6

7 0000 BEGIN:

8 0000 B8 0000s MOV AX, DATA ; Загружаем адрес сегмента +

9 данных в регистр AX

10 0003 8E D8 MOV DS, AX ; Устанавливаем сегмент данных

11 0005 E8 0046 CALL CLRSCR ; Вызываем процедуру очистки +

12 экрана

13

14 0008 8A 16 0000r MOV DL, SYMB1 ; Загружаем символ SYMB1 в +

15 регистр DL

16 000C E8 002F CALL PUTCH ; Выводим символ

17 000F E8 0031 CALL CLRF ; Очищаем строку

18 0012 8A 16 0001r MOV DL, SYMB2 ; Загружаем символ SYMB2 в +

19 регистр DL

20 0016 E8 0025 CALL PUTCH ; Выводим символ

21 0019 E8 0027 CALL CLRF ; Очищаем строку

22 001C 8A 16 0002r MOV DL, SYMB3 ; Загружаем символ SYMB3 в +

23 регистр DL

24 0020 E8 001B CALL PUTCH ; Выводим символ

25 0023 E8 001D CALL CLRF ; Очищаем строку

26

27 0026 B4 09 MOV AH, 09H ; Устанавливаем функцию +

28 вывода строки

29 0028 BA 0003r MOV DX, OFFSET MSG ; Загружаем адрес сообщения в DX

30 002B CD 21 INT 21H ; Вызываем прерывание для +

31 вывода строки

32

33 002D E8 0009 CALL GETCH ; Ждем нажатия любой клавиши

34 0030 E8 001B CALL CLRSCR ; Очищаем экран

35

36 0033 B4 4C MOV AH, 4CH ; Устанавливаем функцию +

37 завершения программы

38 0035 B0 00 MOV AL, 0 ; Устанавливаем код xвозврата 0

39 0037 CD 21 INT 21H ; Вызываем прерывание для +

40 завершения программы

41

42

43 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

44 0039 GETCH PROC

45 0039 B4 01 MOV AH, 01H ; Устанавливаем функцию ввода +

46 символа с клавиатуры

47 003B CD 21 INT 21H ; Вызываем прерывание для +

48 ввода символа

49 003D C3 RET ; Возвращаемся из процедуры

50 003E GETCH ENDP

51

52 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

53 003E PUTCH PROC

54 003E B4 02 MOV AH, 02H ; Устанавливаем функцию +

55 вывода символа

56 0040 CD 21 INT 21H ; Вызываем прерывание для +

57 вывода символа

Turbo Assembler Version 3.1 04/02/24 20:59:48 Page 2

c:\lab3\lab.asm

58 0042 C3 RET ; Возвращаемся из процедуры

59 0043 PUTCH ENDP

60

61 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

62 0043 CLRF PROC

63 0043 B4 02 MOV AH, 02H ; Устанавливаем функцию +

64 вывода символа

65 0045 B2 0A MOV DL, 10 ; Загружаем символ перевода +

66 строки

67 0047 CD 21 INT 21H ; Вызываем прерывание для +

68 вывода символа

69 0049 B2 0D MOV DL, 13 ; Загружаем символ возврата +

70 каретки

71 004B CD 21 INT 21H ; Вызываем прерывание для +

72 вывода символа

73 004D C3 RET ; Возвращаемся из процедуры

74 004E CLRF ENDP

75

76 ; - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -

77 004E CLRSCR PROC

78 004E B8 0002 MOV AX, 02H ; Устанавливаем функцию +

79 очистки экрана

80 0051 CD 10 INT 10H ; Вызываем прерывание для +

81 очистки экрана

82 0053 C3 RET ; Возвращаемся из процедуры

83 0054 CLRSCR ENDP

84

85 0054 PRGR ENDS

86 ;=================================================================================

87

88 0000 DATA SEGMENT

89 0000 41 SYMB1 DB 65 ; Определяем символы

90 0001 42 SYMB2 DB 66

91 0002 43 SYMB3 DB 67

92 0003 50 72 65 73 73 20 61+ MSG DB 'Press any key to exit... $' ; Определяем сообщение

93 6E 79 20 6B 65 79 20+

94 74 6F 20 65 78 69 74+

95 2E 2E 2E 20 24

96 001D DATA ENDS

97 ;=================================================================================

98

99 0000 STK SEGMENT STACK

100 0000 0100\*(00) DB 256 DUP (0) ; Объявляем стек

101 0100 STK ENDS

102 ;=================================================================================

103

104 END BEGIN ; Завершаем программу

Turbo Assembler Version 3.1 04/02/24 20:59:48 Page 3

Symbol Table

Symbol Name Type Value Cref (defined at #)

??DATE Text "04/02/24"

??FILENAME Text "lab "

??TIME Text "20:59:48"

??VERSION Number 030A

@CPU Text 0101H

@CURSEG Text STK #3 #88 #99

@FILENAME Text LAB

@WORDSIZE Text 2 #3 #88 #99

BEGIN Near PRGR:0000 #7 104

CLRF Near PRGR:0043 17 21 25 #62

CLRSCR Near PRGR:004E 11 34 #77

GETCH Near PRGR:0039 33 #44

MSG Byte DATA:0003 29 #92

PUTCH Near PRGR:003E 16 20 24 #53

SYMB1 Byte DATA:0000 14 #89

SYMB2 Byte DATA:0001 18 #90

SYMB3 Byte DATA:0002 22 #91

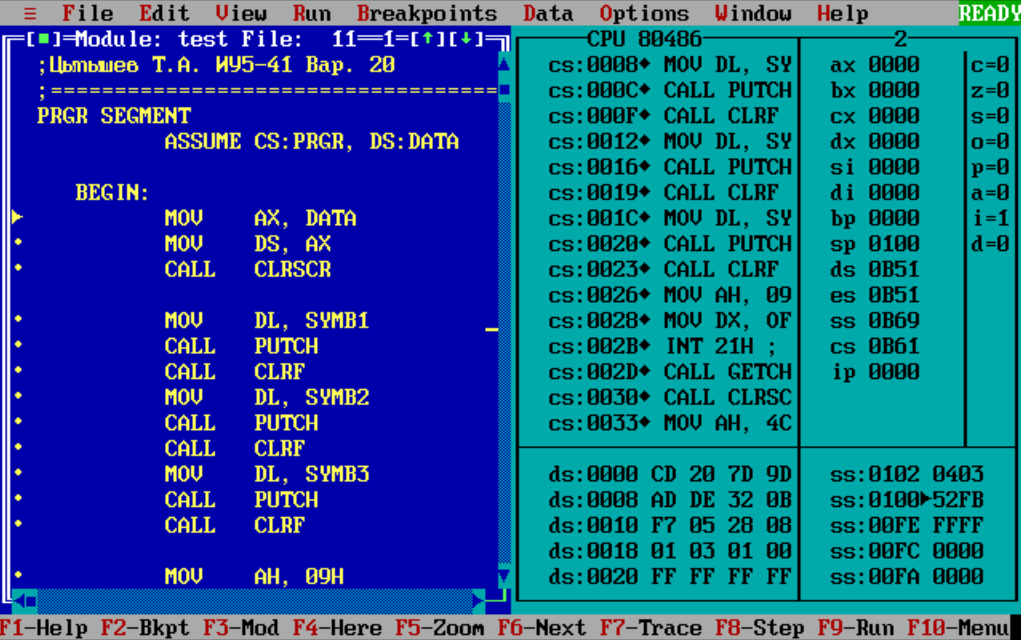
Groups & Segments Bit Size Align Combine Class Cref (defined at #)

DATA 16 001D Para none 4 8 #88

PRGR 16 0054 Para none #3 4

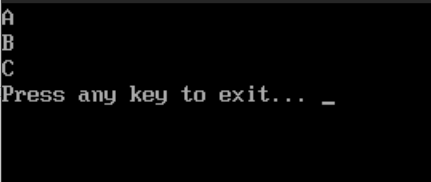
STK 16 0100 Para Stack #99

# Скриншот программы в TD.exe



# Результаты работы программы

Для завершения программы необходимо нажать любую клавишу



# Выводы по ЛР № 3

Разработан файл .ASM и соответствующие файлы приложения и листинга на языке Ассемблер. Программа корректно выводит единичные символы при помощи перемещений данных в регистрах, использования прерываний системы. Так мы изучили основы языка Ассемблер.