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 - ДА)

10

(количество листов)

Вариант № 20

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

Москва, МГТУ - 2024

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

**СОДЕРЖАНИЕ**

[1. Цель выполнения лабораторной работы № 5 3](#_gjdgxs)

[2. Порядок и условия проведения работы № 5 3](#_30j0zll)

[3. Описание ошибок, возникших при отладке № 5 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 № 5 8](#_1t3h5sf)

# Цель выполнения лабораторной 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).

# Порядок и условия проведения работы № 5

1. **Завершение ввода строки**: Признаком завершения ввода отдельной строки с клавиатуры является символ "$", который вводится с клавиатуры. Между введенной строкой символов и их шестнадцатеричным представлением должен быть знак равенства "=".
2. **Ограничение на количество символов**: Максимальное число вводимых символов не должно превышать 20.
3. **Организация цикла ввода**: Ввод строк осуществляется в цикле до ввода специального символа "\*", который завершает цикл. Для этого используется команда LOOP.
4. **Обработка строки и вывод результата**: После завершения ввода строки ее автоматически выводится.
5. **Именование процедур**: Требования к процедурам и их именованию совпадают с требованиями предыдущих лабораторных работ.
6. **Дополнительные процедуры**: Для ввода/вывода строки и ее шестнадцатеричного представления разрабатываются дополнительные процедуры, например, HEX.
7. **Очистка экрана**: Организуется очистка экрана до начала и после завершения работы программы с помощью специальной процедуры CLRSCR.
8. **Объявление сегментов**: Необходимо отдельно объявить сегмент данных (DTSEG) и сегмент стека (STSEG). Проверка загрузки сегментного регистра данных (DS) осуществляется через промежуточный регистр (AX) с помощью команды MOV.
9. **Оформление отчета**: Для оформления отчета студенту необходимо знать или найти способ вывода результата работы программы в текстовый файл. Рекомендуется использовать копирование текста из окна командной строки.
10. **Лучшие практики**: Рекомендуется избегать снятия графической картинки с экрана для оформления отчета, предпочтительнее использовать текстовые данные.

# Описание ошибок, возникших при отладке № 5

| № п/п | Проявление ошибки | Причина ошибки | Способ устранения |
| --- | --- | --- | --- |
|  | Неправильное срабатывание ограничения в 20 символов | Неправильная установка счетчика символов | MOV CX,20 - Установка счетчика символов в 20  MOV SI,3 - Установка индекса начала строки в 3 |

# Блок-схема программы

# Текст программы на языке Ассемблера (.LST)

Turbo Assembler Version 3.1 05/08/24 05:27:11 Page 1

lab.asm

1 ;Лабораторная работа №5

2 ;Цыпышев Т.А. ИУ5-41 Вар. 20

3 ;===============================================================================

4

5 0000 PRGR SEGMENT

6 ASSUME CS:PRGR, DS:DATA

7

8 0000 BEGIN:

9 0000 B8 0000s MOV AX, DATA ; Инициализация сегмента данных

10 0003 8E D8 MOV DS, AX

11 0005 E8 0066 CALL INFO ; Вызов процедуры INFO для вывода приветствия

12

13 0008 INIT:

14 0008 B9 0014 MOV CX, 20 ; Установка счетчика символов в 20

15 000B BE 0003 MOV SI, 3 ; Установка индекса начала строки в 3

16 000E BB 0000r MOV BX, OFFSET hextb ; Загрузка таблицы шестнадцатеричных символов

17

18 0011 INPUT:

19 0011 E8 008D CALL GETCH ; Получение символа

20 0014 3C 24 CMP AL, '$' ; Проверка на конец строки

21 0016 74 0C JE END\_LINE

22 0018 3C 2A CMP AL, '\*' ; Проверка на символ завершения ввода

23 001A 74 30 JE EXIT\_PROGRAM

24 001C E8 009A CALL HEX ; Преобразование символа в шестнадцатеричное +

25 представление

26 001F 49 DEC CX ; Уменьшение счетчика символов

27 0020 74 02 JZ END\_LINE ; Если счетчик стал равен нулю, завершаем ввод

28 0022 EB ED JMP INPUT ; Повторяем ввод

29

30 0024 END\_LINE:

31 0024 53 PUSH BX ; Сохранение регистров

32 0025 51 PUSH CX

33 0026 BB 0001 MOV BX, 1 ; Установка начального индекса для вывода строки

34 0029 4E DEC SI ; Уменьшение индекса строки до последнего +

35 символа

36 002A 8B CE MOV CX, SI ; Установка счетчика на количество введенных +

37 символов

38 002C B4 40 MOV AH, 40h ; Услуга для вывода строки

39 002E BA 0010r MOV DX, OFFSET line

40 0031 CD 21 INT 21h

41 0033 B9 0014 MOV CX, 20 ; Восстановление счетчика символов

42 0036 BE 0003 MOV SI, 3 ; Восстановление индекса строки

43

44 0039 CLEAR:

45 0039 C6 84 0010r 20 90 MOV line[SI], 32 ; Очистка символов в строке

46 003F 46 INC SI

47 0040 E2 F7 LOOP CLEAR

48 0042 59 POP CX ; Восстановление регистров

49 0043 5B POP BX

50 0044 B9 0004 MOV CX, 4

51 0047 E8 005C CALL CLRF ; Очистка экрана

52 004A E2 BC LOOP INIT ; Повторное начало ввода

53

54 004C EXIT\_PROGRAM:

55 004C E8 0057 CALL CLRF ; Очистка экрана

56 004F E8 0054 CALL CLRF ; Очистка экрана

57 0052 B4 09 MOV AH, 09h ; Вывод сообщения о завершении программы

Turbo Assembler Version 3.1 05/08/24 05:27:11 Page 2

lab.asm

58 0054 BA 00CFr LEA DX, msg\_input\_name

59 0057 CD 21 INT 21h

60 0059 E8 004A CALL CLRF ; Очистка экрана

61 005C BA 004Cr MOV DX, OFFSET msg

62 005F CD 21 INT 21h

63 0061 E8 003D CALL GETCH ; Ожидание нажатия любой клавиши

64 0064 B8 0002 MOV AX, 02h ; Выход из программы

65 0067 CD 10 INT 10h

66 0069 B8 4C00 MOV AX, 4C00h

67 006C CD 21 INT 21h

68

69 006E INFO PROC

70 006E B8 0002 MOV AX, 02h ; Установка видеорежима

71 0071 CD 10 INT 10h

72 0073 B4 09 MOV AH, 09h ; Вывод приветствия

73 0075 BA 0066r MOV DX, OFFSET greet

74 0078 CD 21 INT 21h

75 007A E8 0029 CALL CLRF ; Очистка экрана

76 007D B4 09 MOV AH, 09h ; Вывод информации о программе

77 007F BA 008Ar MOV DX, OFFSET defl

78 0082 CD 21 INT 21h

79 0084 E8 001F CALL CLRF ; Очистка экрана

80 0087 B4 02 MOV AH, 02h ; Установка курсора в позицию

81 0089 B2 24 MOV DL, 36

82 008B CD 21 INT 21h

83 008D B4 09 MOV AH, 09h ; Вывод дополнительной информации

84 008F BA 00ADr MOV DX, OFFSET info1

85 0092 CD 21 INT 21h

86 0094 E8 000F CALL CLRF ; Очистка экрана

87 0097 B4 09 MOV AH, 09h

88 0099 BA 00BEr MOV DX, OFFSET info2

89 009C CD 21 INT 21h

90 009E E8 0005 CALL CLRF ; Очистка экрана

91 00A1 INFO ENDP

92

93 00A1 GETCH PROC

94 00A1 B4 01 MOV AH, 01h ; Получение символа

95 00A3 CD 21 INT 21h

96 00A5 C3 RET

97 00A6 GETCH ENDP

98

99 00A6 CLRF PROC

100 00A6 B4 02 MOV AH, 02h ; Вывод символов перевода строки

101 00A8 B2 0A MOV DL, 10

102 00AA CD 21 INT 21h

103 00AC B4 02 MOV AH, 02h

104 00AE B2 0D MOV DL, 13

105 00B0 CD 21 INT 21h

106 00B2 C3 RET

107 00B3 CLRF ENDP

108

109 00B3 CLRSCR PROC

110 00B3 B8 0002 MOV AX, 02H ; Очистка экрана

111 00B6 CD 10 INT 10H

112 00B8 C3 RET

113 00B9 CLRSCR ENDP

114

Turbo Assembler Version 3.1 05/08/24 05:27:11 Page 3

lab.asm

115 00B9 HEX PROC

116 00B9 50 PUSH AX ; Сохранение регистра AX

117 00BA D0 E8 D0 E8 D0 E8 D0+ SHR AL, 4 ; Сдвиг вправо на 4 бит

118 E8

119 00C2 D7 XLAT ; Перевод символа в шестнадцатеричный эквивалент

120 00C3 88 84 0010r MOV line[SI], AL ; Сохранение первой половины шестнадцатеричного+

121 представления

122 00C7 46 INC SI ; Увеличение индекса текущего символа

123 00C8 58 POP AX ; Восстановление регистра AX

124 00C9 24 0F AND AL, 00001111b ; Очистка старших битов

125 00CB D7 XLAT ; Перевод символа в шестнадцатеричный эквивалент

126 00CC 88 84 0010r MOV line[SI], AL ; Сохранение второй половины шестнадцатеричного+

127 представления

128 00D0 46 INC SI ; Увеличение индекса текущего символа

129 00D1 B0 20 MOV AL, 32 ; Установка пробела в строку

130 00D3 88 84 0010r MOV line[SI], AL ; Сохранение пробела

131 00D7 46 INC SI ; Увеличение индекса текущего символа

132 00D8 C3 RET

133 00D9 HEX ENDP

134

135 00D9 PRGR ENDS

136

137 0000 DATA SEGMENT

138 0000 30 31 32 33 34 35 36+ hextb DB '0123456789ABCDEF' ; Таблица +

139 37 38 39 41 42 43 44+ шестнадцатеричных символов

140 45 46

141 0010 14\*(20 3D 20) line DB 20 DUP (' = ') ; Буфер для хранения +

142 строки

143 004C 50 72 65 73 73 20 61+ msg DB 'Press any key to exit... $' ; Сообщение о выходе

144 6E 79 20 6B 65 79 20+

145 74 6F 20 65 78 69 74+

146 2E 2E 2E 20 24

147 0066 54 79 70 65 20 61 6E+ greet DB 'Type any chars, maximum length - 20$' ; Приветствие

148 79 20 63 68 61 72 73+

149 2C 20 6D 61 78 69 6D+

150 75 6D 20 6C 65 6E 67+

151 74 68 20 2D 20 32 30+

152 24

153 008A 2D 2D 2D 2D 2D 2D 2D+ defl DB '----------------------------------$' ; Разделитель

154 2D 2D 2D 2D 2D 2D 2D+

155 2D 2D 2D 2D 2D 2D 2D+

156 2D 2D 2D 2D 2D 2D 2D+

157 2D 2D 2D 2D 2D 2D 24

158 00AD 20 2D 20 65 6E 64 20+ info1 DB ' - end of string$' ; Информация о конце +

159 6F 66 20 73 74 72 69+ строки

160 6E 67 24

161 00BE 2A 20 2D 20 65 6E 64+ info2 DB '\* - end of input$' ; Информация о конце +

162 20 6F 66 20 69 6E 70+ ввода

163 75 74 24

164 00CF 54 53 59 50 59 53 48+ msg\_input\_name DB 'TSYPYSHEV T.A. UI5-41 Var-20$' ; Имя автора и номер +

165 45 56 20 54 2E 41 2E+ варианта

166 20 55 49 35 2D 34 31+

167 20 56 61 72 2D 32 30+

168 24

169 00EC DATA ENDS

170

171 0000 STK SEGMENT STACK

Turbo Assembler Version 3.1 05/08/24 05:27:11 Page 4

lab.asm

172 0000 0100\*(00) DB 256 DUP (0) ; Стек

173 0100 STK ENDS

174

175 END BEGIN

Turbo Assembler Version 3.1 05/08/24 05:27:11 Page 5

Symbol Table

Symbol Name Type Value Cref (defined at #)

??DATE Text "05/08/24"

??FILENAME Text "lab "

??TIME Text "05:27:11"

??VERSION Number 030A

@CPU Text 0101H

@CURSEG Text STK #5 #137 #171

@FILENAME Text LAB

@WORDSIZE Text 2 #5 #137 #171

BEGIN Near PRGR:0000 #8 175

CLEAR Near PRGR:0039 #44 47

CLRF Near PRGR:00A6 51 55 56 60 75 79 86 90 #99

CLRSCR Near PRGR:00B3 #109

DEFL Byte DATA:008A 77 #153

END\_LINE Near PRGR:0024 21 27 #30

EXIT\_PROGRAM Near PRGR:004C 23 #54

GETCH Near PRGR:00A1 19 63 #93

GREET Byte DATA:0066 73 #147

HEX Near PRGR:00B9 24 #115

HEXTB Byte DATA:0000 16 #138

INFO Near PRGR:006E 11 #69

INFO1 Byte DATA:00AD 84 #158

INFO2 Byte DATA:00BE 88 #161

INIT Near PRGR:0008 #13 52

INPUT Near PRGR:0011 #18 28

LINE Byte DATA:0010 39 45 120 126 130 #141

MSG Byte DATA:004C 61 #143

MSG\_INPUT\_NAME Byte DATA:00CF 58 #164

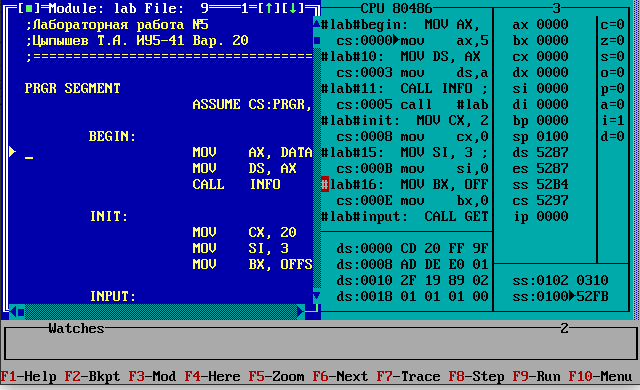
Groups & Segments Bit Size Align Combine Class Cref (defined at #)

DATA 16 00EC Para none 6 9 #137

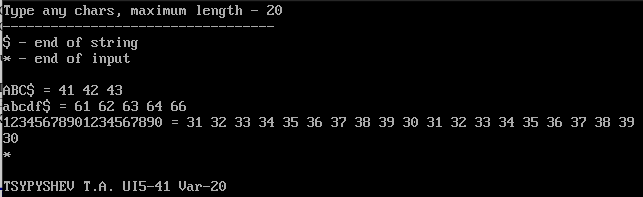
PRGR 16 00D9 Para none #5 6

STK 16 0100 Para Stack #171

# Скриншот программы в TD.exe



# Результаты работы программы



# Выводы по ЛР № 5

Изучив разработку программы на языке Ассемблер для ввода и буферизации строки символов с последующим выводом на экран в шестнадцатеричном представлении, я усвоил несколько важных принципов и навыков:

1. Освоил основы работы с языком Ассемблер, включая структуру программы, инструкции, работу с регистрами и операциями ввода-вывода.

2. Понял необходимость работы с байтовыми массивами для хранения и обработки введенных символов.

3. Получил опыт работы с вложенными циклами, что позволяет эффективно обрабатывать и трансформировать данные в программе.

4. Приобрел навыки отладки программы, что позволяет выявлять и исправлять ошибки в коде для достижения правильного функционирования.

5. Осознал важность оптимизации кода и его структурирования для повышения производительности и улучшения читаемости.

6. Приобрел знания по форматированию вывода данных для удобства восприятия пользователями.

Все это в совокупности позволяет мне эффективно разрабатывать программы на языке Ассемблер для обработки данных, а также лучше понимать принципы работы компьютерных систем.