|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  Калужский филиал  федерального государственного бюджетного  образовательного учреждения высшего образования  ***«Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет)»***  ***(КФ МГТУ им. Н.Э. Баумана)*** |

|  |  |
| --- | --- |
| **ФАКУЛЬТЕТ** | **ИУК «Информатика и управление»** |
| **КАФЕДРА** | **ИУК4 «Программное обеспечение ЭВМ,** |
| **информационные технологии»** | |

**ЛАБОРАТОРНАЯ РАБОТА №1**

**«Технология подготовки и выполнения программ на ASSEMBLER»**

**ДИСЦИПЛИНА: «Системное программирование»**



|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Выполнил: студент гр. ИУК4-31Б | |  |  | ( | Суриков Н.С. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |
| Проверил: | |  |  | ( | Амеличева К.А. | ) |
|  |  |  | (подпись) |  | (Ф.И.О.) |  |

|  |  |
| --- | --- |
| Дата сдачи (защиты):  Результаты сдачи (защиты): | |
|  | - Балльная оценка:  - Оценка: |

**Цель:** изучение процесса разработки программы на ассемблере, создание исходного файла, объектного и загрузочного модулей программы. Изучение основных возможностей, отладчика TDEBUG.EXE.

**Задачи:**

1. Создать рабочую папку для текстов программ на ассемблере и записать в нее файлы tasm.exe, tlink.exe, rtm.exe и td.exe. из пакета tasm, а также файл с исходным текстом программы на ассемблере, который необходимо сохранить с именем prog1.asm.
2. Coздать загрузочный модуль, загрузить его в отладчик и выполнить программу в пошаговом режиме.
3. Изучить возможности отладчика tdebug, режимы просмотра регистров, файлов, памяти.

**Листинг программы:**

1 .model small

2 .stack 100h

3 .data

4 message db 'IUK4 RULIT','$'

5 perem\_1 db 0ffh

6 perem\_2 dw 3a7fh

7 perem\_3 dd 0f54d567ah

8 mas db 10 dup (' ')

9 pole\_1 db 5 dup (?)

10 adr dw perem\_3

11 adr\_full dd perem\_3

12 numbers db 11, 34, 56, 23

13 fin db 'End of the program data segment $'

14

15 .code

16 start:

17 mov ax,@data

18 mov ds,ax

19

20 mov ah,09h

21 mov dx,offset message

22 int 21h

23

24 mov ah,08h

25 mov ah,7h

26 int 21h

27

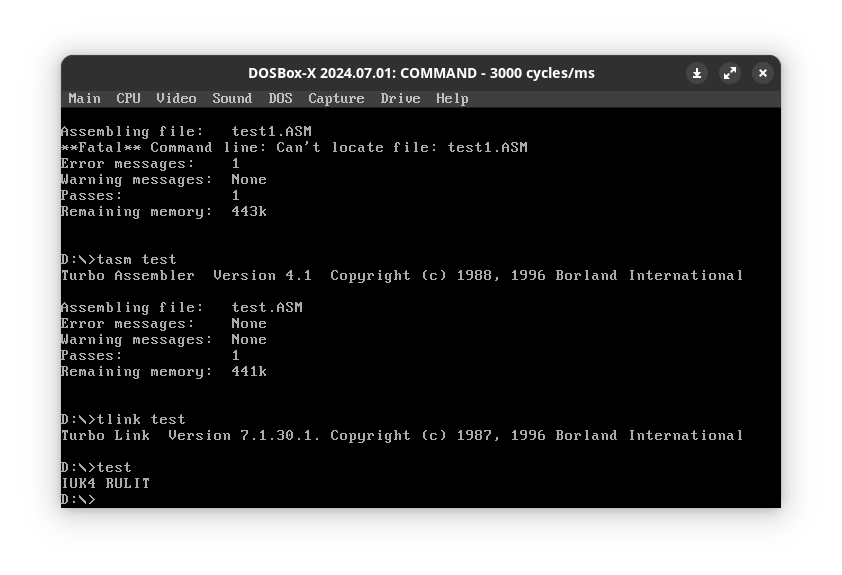
28 mov ax,4c00h

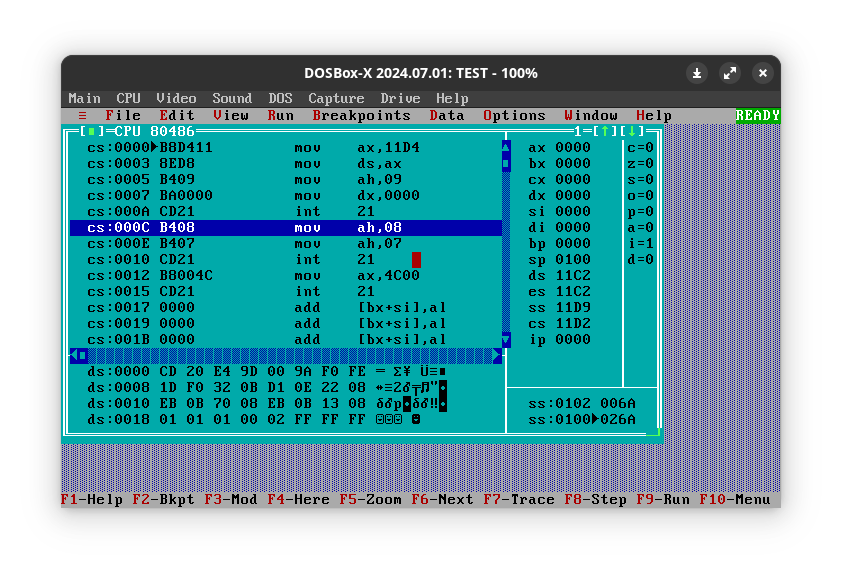
29 int 21h

30 end start

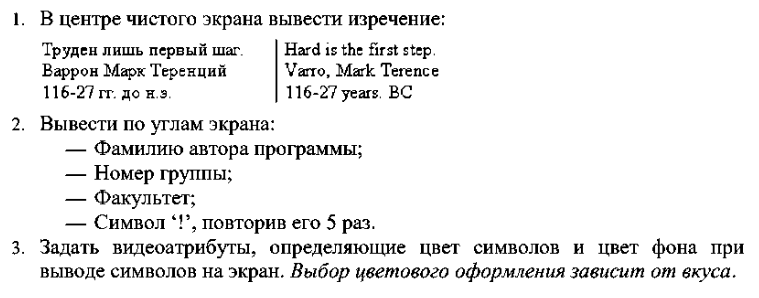
31

**Результат работы:**



**Отладчик:**

**Задание:**



**Листинг:**

1 .model small *; Определяет модель памяти как "small", где код и данные помещаются в один сегмент.*

2 .stack 100h *; Определяет стек размером 256 байт (100h).*

3

4 *; Начало сегмента данных.*

5 .data

6 message db 'Hard is the first step','$', 10, 13

7 message1 db 'Varro, Mark Ternce','$', 10, 13

8 message2 db '116-27 years. BC','$', 10, 13

9 message3 db 'Surikov','$',10,13

10 message4 db 'IUK4-31B','$',10,13

11 message5 db 'IUK4','$',10,13

12

13 .code *; Начало сегмента кода.*

14 start: *; Метка начала программы.*

15 mov ax,@data *; Загружает адрес сегмента данных в регистр AX.*

16 mov ds,ax *; Устанавливает сегмент данных (DS) в значение AX.*

17

18 *; Очистка экрана*

19 mov ax,0600h *; Подготавливает код для очистки экрана (функция 0).*

20 mov bh, 2Ch *; Устанавливает цвет фона и шрифта*

21 mov cx, 0000 *; Указывает количество строк для очистки (все).*

22 mov dx, 184FH *; Указывает адрес экрана (184FH — адрес видеопамяти).*

23 int 10h *; Вызывает прерывание BIOS для выполнения очистки экрана.*

24

25 *; Центральное сообщение*

26 mov ah, 2 *; Подготавливает функцию установки курсора.*

27 mov dh, 10 *; Устанавливает вертикальную позицию курсора (строка 10).*

28 mov dl, 30 *; Устанавливает горизонтальную позицию курсора (столбец 30).*

29 mov bh, 0 *; Указывает страницу экрана (0).*

30 int 10h *; Вызывает прерывание BIOS для установки курсора.*

31

32 mov ah, 09h *; Подготавливает функцию вывода строки.*

33 mov dx, offset message *; Загружает адрес строки message в DX.*

34 int 21h *; Вызывает прерывание DOS для вывода строки.*

35

36 mov ah, 2 *; Подготавливает функцию установки курсора.*

37 mov dh, 11 *; Устанавливает вертикальную позицию курсора (строка 11).*

38 mov dl, 30 *; Устанавливает горизонтальную позицию курсора (столбец 30).*

39 mov bh, 0 *; Указывает страницу экрана (0).*

40 int 10h *; Вызывает прерывание BIOS для установки курсора.*

41

42 mov ah, 09h *; Подготавливает функцию вывода строки.*

43 mov dx, offset message1 *; Загружает адрес строки message1 в DX.*

44 int 21h *; Вызывает прерывание DOS для вывода строки.*

45

46 mov ah, 2 *; Подготавливает функцию установки курсора.*

47 mov dh, 12 *; Устанавливает вертикальную позицию курсора (строка 12).*

48 mov dl, 30 *; Устанавливает горизонтальную позицию курсора (столбец 30).*

49 mov bh, 0 *; Указывает страницу экрана (0).*

50 int 10h *; Вызывает прерывание BIOS для установки курсора.*

51

52 mov ah, 09h *; Подготавливает функцию вывода строки.*

53 mov dx, offset message2 *; Загружает адрес строки message2 в DX.*

54 int 21h *; Вызывает прерывание DOS для вывода строки.*

55

56 *; Вывод информации по углам экрана*

57

58 *; Левый верхний угол*

59 mov ah, 2 *; Подготавливает функцию установки курсора.*

60 mov dh, 0 *; Устанавливает вертикальную позицию курсора (строка 0).*

61 mov dl, 0 *; Устанавливает горизонтальную позицию курсора (столбец 0).*

62 mov bh, 0 *; Указывает страницу экрана (0).*

63 int 10h *; Вызывает прерывание BIOS для установки курсора.*

64

65 mov ah, 09h *; Подготавливает функцию вывода строки*

66 mov dx, offset message3 *; Загружает адрес строки message3 в DX.*

67 int 21h *; Вызывает прерывание DOS для вывода строки.*

68

69 *; Правый верхний угол*

70 mov ah, 2 *; Подготавливает функцию установки курсора.*

71 mov dh, 0 *; Устанавливает вертикальную позицию курсора (строка 0).*

72 mov dl, 72 *; Устанавливает горизонтальную позицию курсора (столбец 72).*

73 mov bh, 0 *; Указывает страницу экрана (0).*

74 int 10h *; Вызывает прерывание BIOS для установки курсора.*

75

76 mov ah, 09h *; Подготавливает функцию вывода строки.*

77 mov dx, offset message4 *; Загружает адрес строки message4 в DX.*

78 int 21h *; Вызывает прерывание DOS для вывода строки.*

79

80 *; Левый нижний угол*

81 mov ah, 2 *; Подготавливает функцию установки курсора.*

82 mov dh, 24 *; Устанавливает вертикальную позицию курсора (строка 24).*

83 mov dl, 0 *; Устанавливает горизонтальную позицию курсора (столбец 0).*

84 mov bh, 0 *; Указывает страницу экрана (0).*

85 int 10h *; Вызывает прерывание BIOS для установки курсора.*

86

87 mov ah, 09h *; Подготавливает функцию вывода строки.*

88 mov dx, offset message5 *; Загружает адрес строки message5 в DX.*

89 int 21h *; Вызывает прерывание DOS для вывода строки.*

90

91 *; Правый нижний угол*

92 mov ah, 2 *; Подготавливает функцию установки курсора.*

93 mov dh, 24 *; Устанавливает вертикальную позицию курсора (строка 24).*

94 mov dl, 75 *; Устанавливает горизонтальную позицию курсора (столбец 74).*

95 mov bh, 0 *; Указывает страницу экрана (0).*

96 int 10h *; Вызывает прерывание BIOS для установки курсора.*

97

98 mov ah, 09h *; Подготавливает функцию вывода строки.*

99 mov al, '!' *; Выводимый символ*

100 mov bl, 10101100b *; Атрибут(цвет, фон, мерцание)*

101 mov cx, 5 *; Коэффицент повторения*

102 int 10h *; Вызывает прерывание BIOS для установки курсора.*

103

104 mov ah, 7h *; Подготавливает функцию для ожидания нажатия клавиши.*

105 int 21h *; Вызывает прерывание DOS для ожидания нажатия клавиши.*

106

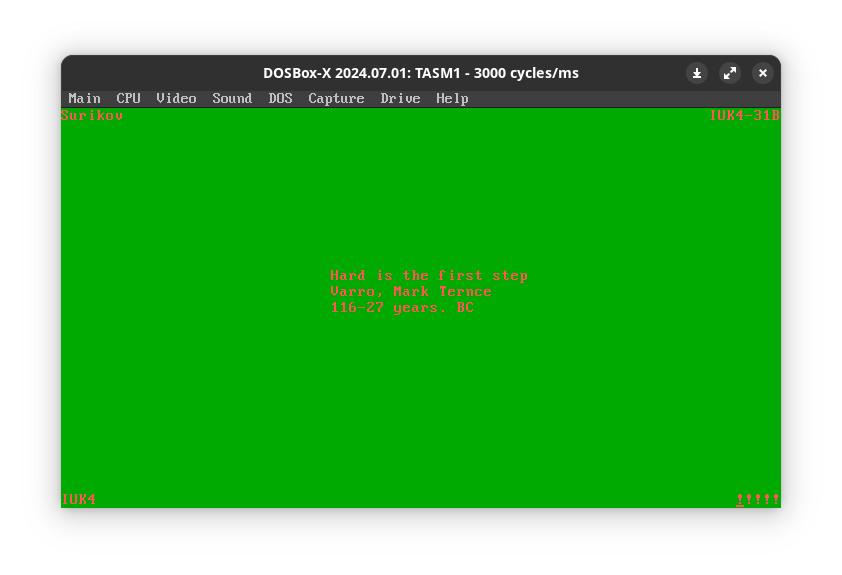
107 mov ax, 4c00h *; Завершает программу и возвращает управление операционной системе.*

108 int 21h *; Вызывает прерывание DOS для завершения программы.*

109

110 end start

**Результат работы:**



**Вывод:** в ходе выполнения данной лабораторной работы были изучены процесс разработки программы на ассемблере, создание исходного файла, объектного и загрузочного модулей программы, основные возможности отладчика TDEBUG.EXE