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

«Этапы создания программы на ASSEMBLER»

ДИСЦИПЛИНА: «Машинно-зависимые языки программирования»

Выполнил: студент гр. ИУК4-32Б _____ (Карельский М.К.)
(Подпись)

Проверил: _____ (Ameличева К.А.)
(Подпись)

Дата сдачи (защиты):

Результаты сдачи (защиты):

- Балльная оценка:
- Оценка:

Калуга, 2021

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

Задание:

1. Используя текстовый редактор, создать и отредактировать исходный модуль программы **PRG_1.asm**, текст которого приведен ниже.

Prg_1.asm. Пример использования директив резервирования и инициализации данных

```
.model small
.stack 100h
.data
message db 'Запустите эту программу в отладчике','$'
perem_1    db    0ffh
perem_2    dw    3a7fh
perem_3    dd    0f54d567ah
mas        db    10 dup (' ')
pole_1     db    5 dup (?)
adr        dw    perem_3
adr_full   dd    perem_3
numbers    db    11, 34, 56, 23
fin        db    'Конец сегмента данных программы $'
.code
start:
      mov   ax,@data
      mov   ds,ax
      mov   ah,09h
      mov   dx,offset message
      int   21h
      mov   ah, 7h
      int   21h
      mov   ax,4c00h
      int   21h
end   start
```

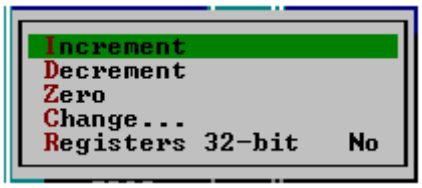
2. Используя компилятор Турбо Ассемблера **tasm.exe** создать файлы **PRG_1.obj** и **PRG_1.lst**.

3. Убедиться в работоспособности программы **PRG_1**, запустив ее из командной строки

4. Запустите программу в отладчике. Выполните программу до команды int 21h и просмотрите содержимое регистров процессора.

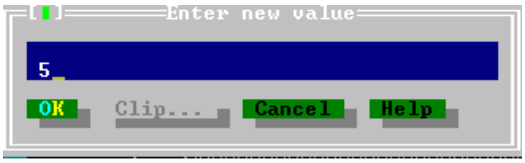
- Вы должны увидеть, что в старшей половине регистра AX находится число 09h - номер вызываемой функции DOS. Младшая половина регистра AX заполнена остатком от выполнения последней операции с регистром AX.
- В регистре DX будет 0000h - относительный адрес первого байта строки message в сегменте команд.
- Изменим этот относительный адрес. Для этого надо:

1) Перейти в окно регистров, поместить курсор на строку, отображающую содержимое регистра DX, и ввести команду Alt+F10, открывающую внутреннее меню окна регистров



Increment, Decrement,- уменьшить, увеличить значение регистра на 1
Zero – обнулить регистр;
Change – заменить значение в регистре, на любое заданное значение

2) Выбрав пункт Change, занесем **в регистр DX число 5**;



3) Выполнить очередную команду (int 21h), DOS выведет на экран строку, начало которой расположено в байте 5 сегмента данных.

В нашей фразе "Запустите эту программу в отладчике» байт 5 приходится на букву **т** (нумерация байтов в строке, естественно, начинается с нуля).

В результате на экран будет выведена строка «тите эту программу в отладчике»;

5. Внести изменения в программу **PRG_1**, которые заставят ее выводить на экран еще две строки символов: «My name is Family name» и «My group ITD-31». Для этого создайте новый исходный модуль **PRG_2.asm,** выполните ассемблирование и компоновку, после чего убедитесь в работоспособности программы.

Решение:

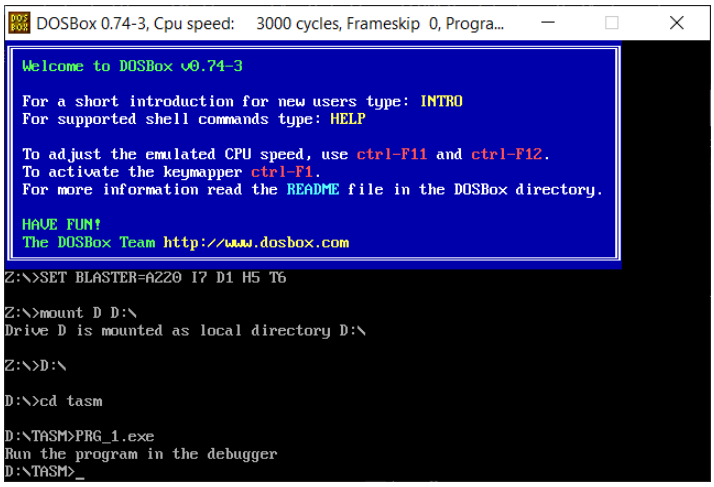


Рис. 1. Пункт 3

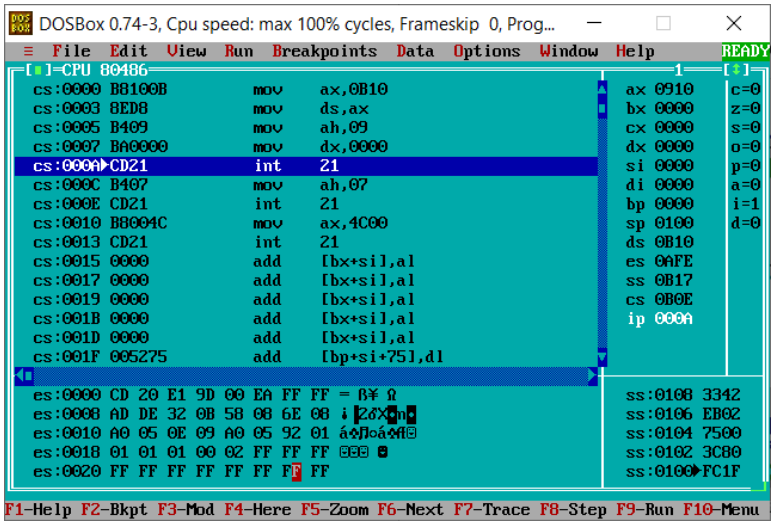


Рис. 2.1. Пункт 4

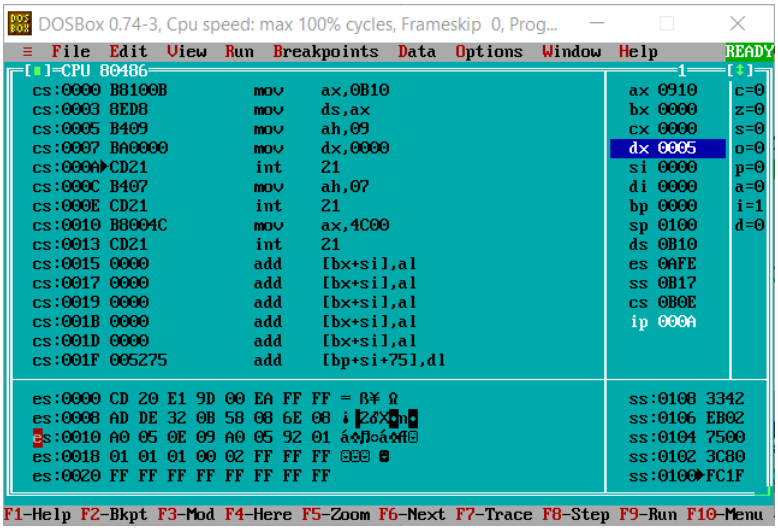


Рис. 2.2. Пункт 4

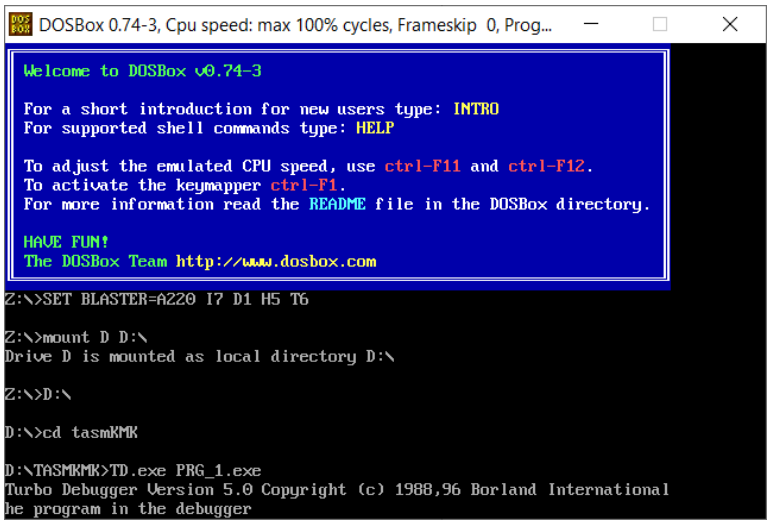


Рис. 2.3. Пункт 4

Листинг PRG_2:

```
.model small
.stack 100h
.data
message   db 'Run the program in the debugger', '$'
nm        db   10, 13, 'My name is Karelsky', '$'
grp       db   10, 13, 'My group is IUK4-32B', '$'
perem_1   db   0ffh
perem_2   dw   3a7fh
perem_3   dd   0f54d567ah
mas       db   10 dup (' ')
pole_1    db   5 dup (?)
adr       dw   perem_3
adr_full  dd   perem_3
numbers   db   11, 34, 56, 23
fin       db   'End of the program data segment $'
.code
start:
     mov   ax, @data
     mov   ds, ax
     mov   ah, 09h
     mov   dx, offset message
     int   21h
     mov   ah, 7h
     int   21h
     mov   ah, 09h
     mov   dx, offset nm
     int   21h
     mov   ah, 7h
     int   21h
     mov   ah, 09h
     mov   dx, offset grp
     int   21h
     mov   ah, 7h
     int   21h
     mov   ax, 4c00h
     int   21h
end start
```

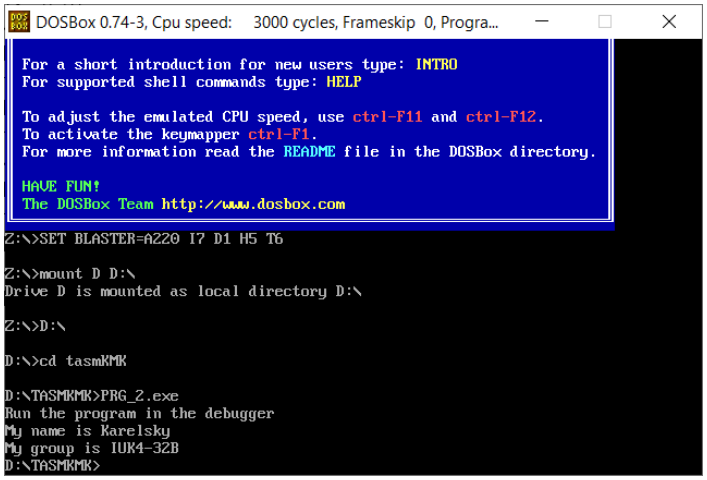


Рис. 3.1. Пункт 5

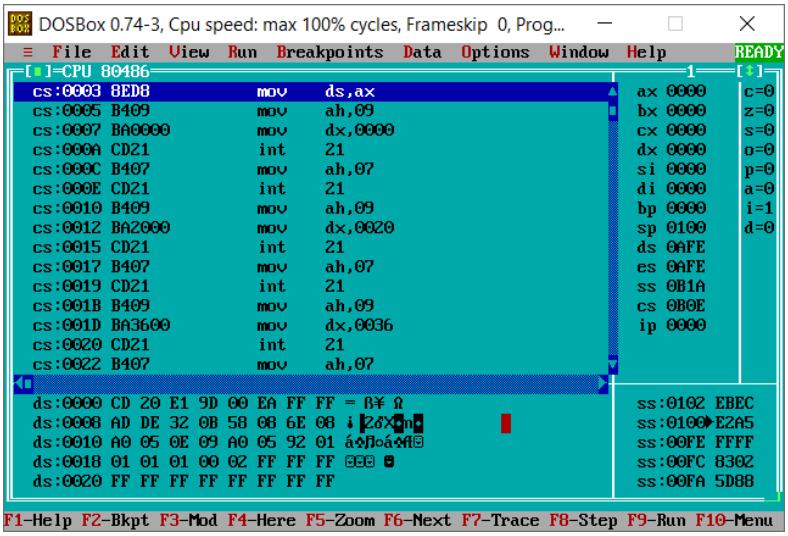


Рис. 3.2. Пункт 5

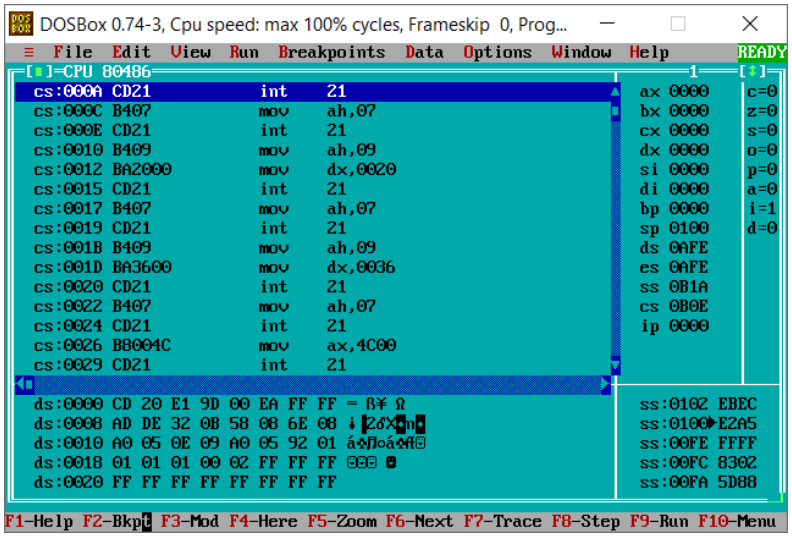


Рис. 3.3. Пункт 5

Контрольные вопросы:

1. Предложите два типа программ или приложений, которые удобно писать на языке ассемблера.

Операционные системы, драйверы.

2. Почему программа, написанная для процессора Intel 8086, не будет работать с процессорами Intel 80386 или Intel 80486?

Ассемблер – машинно-зависимый язык программирования. Программы, написанные для одного процессора, не будут работать с другим, ведь у процессоров разный набор команд.

3. В чем особенность программы на языке ассемблер?

Ассемблер – язык низкого уровня, позволяющий использовать мнемоники.

4. Из чего состоит процесс ассемблирования программы?

Исходный код превращается в объектный модуль.

5. Чем отличается трансляция от компоновки?

Во время трансляции исходный код превращается в объектный модуль, а во время компоновки объектные модули комбинируются в выполняемую программу.

6. Объясните назначение каждого из перечисленных файлов, получаемых при создании исполняемой программы на языке ассемблер *.asm, *.lst, *.map, *.exe.

.asm – исходный код

.lst – листинг

.map – таблица адресов

.exe – выполняемая программа

7. Что такое "Турбоотладчик", расскажите основы пользования им?

Турбоотладчик – программа, разработанная для поиска и исправления логических ошибок. Для его запуска следует прописать в терминале следующую команду: *TD.exe YourProgram.exe*. Чтобы перейти на следующий шаг исполнения программы, следует нажать F8.

8. Перечислите основные команды для работы с ячейками памяти и регистрами отладчика Turbo Debugger?

Increment – увеличить значение регистра на 1

Decrement – уменьшить значение регистра на 1

Zero – обнулить регистр

Change – заменить значение в регистре на любое заданное значение

9. Для чего предназначена программа DOSBox?

DOSBox – эмулятор, необходимый для запуска программ, написанных под MS-DOS.

Вывод: в ходе выполнения лабораторной работы были получены навыки работы с машинно-зависимым языком программирования Assembler, эмулятором DOSBox, изучены этапы создания программы на Assembler.