## Отчет по лабораторной работе №5

Дисциплина: архитектура компьютера

Киселева Елизавета Александровна

## Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
	4.1 Основы работы с тс	9
	4.2 Структура программы на языке ассемблера NASM	11
	4.3 Подключение внешнего файла	14
	4.4 Выполнение заданий для самостоятельной работы	17
5	Выводы	20
Сг	писок литературы	21

# Список иллюстраций

4.1	Открытый mc	9
4.2	Перемещение между директориями	10
4.3	Создание каталога	10
4.4	Создание файла	11
4.5	Открытие файла для редактирования	12
4.6	Редактирование файла	13
4.7	Компиляция файла и передача на обработку компоновщику, испол-	
	нение файла	13
4.8	Скачиваение и перемещение файла	14
4.9	Копирование файла	14
4.10	Редактирование файла	15
4.11	Исполнение файла	15
4.12	Отредактированный файл	16
4.13	Исполнение файла	16
4.14	Копирование и редактирование файла	17
4.15	Исполнение файла	18
4.16	Копирование файла	18
4.17	Редактирование файла	19
4.18	В Исполнение файла	19

# Список таблиц

## 1 Цель работы

Целью данной лабораторной работы является приобретение практических навыков работы в Midnight Commander, освоение инструкций языка ассемблера mov и int.

## 2 Задание

- 1. Основы работы с тс
- 2. Структура программы на языке ассемблера NASM
- 3. Подключение внешнего файла
- 4. Выполнение заданий для самостоятельной работы

### 3 Теоретическое введение

Midnight Commander (или просто mc) — это программа, которая позволяет просматривать структуру каталогов и выполнять основные операции по управлению файловой системой, т.е. mc является файловым менеджером. Midnight Commander позволяет сделать работу с файлами более удобной и наглядной. Программа на языке ассемблера NASM, как правило, состоит из трёх секций: секция кода программы (SECTION .text), секция инициированных (известных во время компиляции) данных (SECTION .data) и секция неинициализированных данных (тех, под которые во время компиляции только отводится память, а значение присваивается в ходе выполнения программы) (SECTION .bss). Для объявления инициированных данных в секции .data используются директивы DB, DW, DD, DQ и DT, которые резервируют память и указывают, какие значения должны храниться в этой памяти: - DB (define byte) — определяет переменную размером в 1 байт; - DW (define word) — определяет переменную размеров в 2 байта (слово); - DD (define double word) — определяет переменную размером в 4 байта (двойное слово); - DQ (define quad word) — определяет переменную размером в 8 байт (учетве- рённое слово); - DT (define ten bytes) — определяет переменную размером в 10 байт. Директивы используются для объявления простых переменных и для объявления массивов. Для определения строк принято использовать директиву DB в связи с особенностями хранения данных в оперативной памяти. Инструкция языка ассемблера mov предназначена для дублирования данных источника в приёмнике.

mov dst,src

Здесь операнд dst — приёмник, а src — источник. В качестве операнда могут выступать регистры (register), ячейки памяти (memory) и непосредственные значения (const). Инструкция языка ассемблера intпредназначена для вызова прерывания с указанным номером.

#### int n

Здесь n— номер прерывания, принадлежащий диапазону 0–255. При программировании в Linux с использованием вызовов ядра sys\_calls n=80h (принято задавать в шестнадцатеричной системе счисления).

### 4 Выполнение лабораторной работы

### 4.1 Основы работы с тс

Открываю Midnight Commander, введя в терминал mc (рис. 4.1).

```
eakiseleva1@dk3n55-... × eakiseleva1@dk3n55-l... × eakiseleva1@dk3n55-... × ▼
eakiseleva1@dk3n55 ~ $ mc
```

Рис. 4.1: Открытый тс

Перехожу в каталог ~/work/study/arch-pc, используя файловый менеджер mc (рис. 4.2)

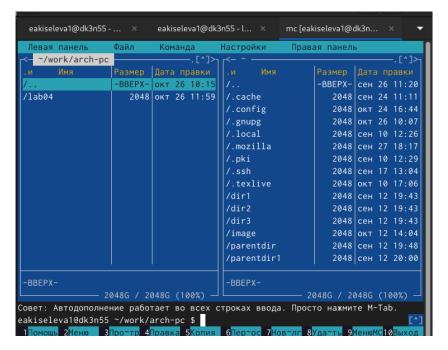


Рис. 4.2: Перемещение между директориями

С помощью функциональной клавиши F7 создаю каталог lab05 (рис. 4.3).

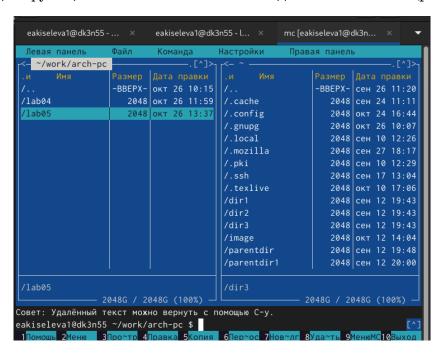


Рис. 4.3: Создание каталога

Перехожу в созданный каталог. В строке ввода прописываю команду touch lab5-1.asm, чтобы создать файл, в котором буду работать (рис. 4.4).

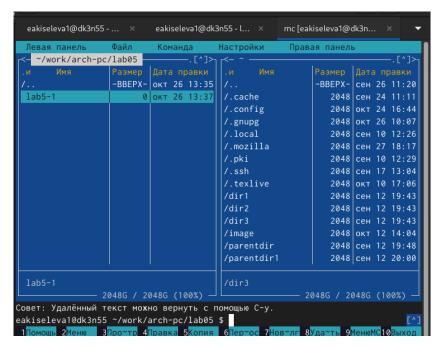


Рис. 4.4: Создание файла

#### 4.2 Структура программы на языке ассемблера NASM

С помощью функциональной клавиши F4 открываю созданный файл для редактирования в редакторе nano (рис. 4.5).

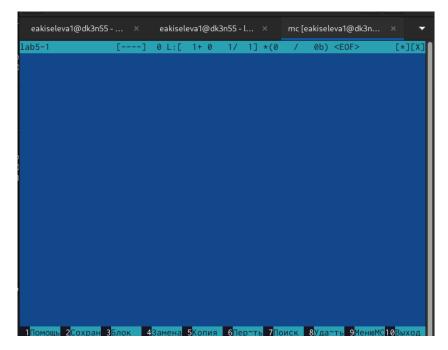


Рис. 4.5: Открытие файла для редактирования

Ввожу в файл код программы для запроса строки у пользователя (рис. 4.6). Далее выхожу из файла (Ctrl+X), сохраняя изменения (Y, Enter). С помощью функциональной клавиши F3 открываю файл для просмотра, чтобы проверить, содержит ли файл текст программы.

Рис. 4.6: Редактирование файла

Транслирую текст программы файла в объектный файл командой nasm -f elf lab5-1.asm. Создался объектный файл lab5-1.o. Выполняю компоновку объектного файла с помощью команды ld -m elf\_i386 -o lab5-1 lab5-1.o. Создался исполняемый файл lab5-1.Запускаю исполняемый файл. Программа выводит строку "Введите строку:" и ждет ввода с клавиатуры, я ввожу свои ФИО, на этом программа заканчивает свою работу (рис. 4.7).

```
eakiseleva1@dk4n59 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1.asm
eakiseleva1@dk4n59 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1 lab5-1.o
eakiseleva1@dk4n59 ~/work/arch-pc/lab05 $ ./lab5-1
Введите строку
Киселева Елизавета
eakiseleva1@dk4n59 ~/work/arch-pc/lab05 $
```

Рис. 4.7: Компиляция файла и передача на обработку компоновщику, исполнение файла

#### 4.3 Подключение внешнего файла

скачиваю файл in\_out.asm со страницы курса в ТУИС. Он сохранился в каталог "загрузки". С помощью функциональной клавиши F5 копирую данный файл в созданный каталог lab05 (рис. 4.8).

<- ~/work/arch-pc/	[^]> <sub>7 [</sub> <- ~/work/arch-pc/lab05[^]> <sub>7</sub>								
.и Имя	Размер	Дата			.и Имя	Размер	Дата		равки
1	-BBEPX-	окт	26	13:35	1	-BBEPX-	окт	26	13:35
in_out.asm		ноя				3942	ноя		16:00
*lab5-1	8740	ноя		15:56		8740	ноя		15:56
lab5-1.asm		ноя			lab5-1.o	752	ноя		15:55
lab5-1.o	752	ноя		15:55		323	ноя		15:55
lab5-2.asm		ноя							

Рис. 4.8: Скачиваение и перемещение файла

С помощью функциональной клавиши F5 копирую файл lab5-1 в тот же каталог, но с другим именем, для этого в появившемся окне mc прописываю имя для копии файла (рис. 4.9).

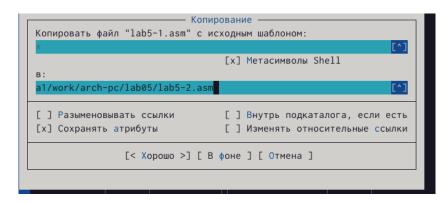


Рис. 4.9: Копирование файла

Изменяю содержимое файла lab5-2.asm во встроенном редакторе nano (рис. 4.10), чтобы в программе использовались подпрограммы из внешнего файла in\_out.asm.

```
/afs/.dk.sci.pfu.edu.ru~ch-pc/lab05/lab5-2.asm 261
%include 'in_out.asm'

SECTION .data
msg: DB 'Введите строку: ',0h

SECTION .bss
buf1: RESB 80

SECTION .text
    GLOBAL _start
    _start:
    mov eax, msg
    call sprintLF

    mov ecx, buf1
    mov edx, 80

call sread

call quit
```

Рис. 4.10: Редактирование файла

Транслирую текст программы файла в объектный файл командой nasm -f elf lab5-2.asm. Создался объектный файл lab5-2.o. Выполняю компоновку объектного файла с помощью команды ld -m elf\_i386 -o lab5-2 lab5-2.o Создался исполняемый файл lab5-2. Запускаю исполняемый файл (рис. 4.11).

```
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
киселева Лиза
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку: Киселева лиза
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 4.11: Исполнение файла

Открываю файл lab5-2.asm для редактирования в nano функциональной кла-

вишей F4. Изменяю в нем подпрограмму sprintLF на sprint. Сохраняю изменения и открываю файл для просмотра, чтобы проверить сохранение действий (рис. 4.12).

```
/afs/.dk.sci.pfu.edu.ru~ch-pc/lab05/lab5-2.asm 259
%include 'in_out.asm'

SECTION .data
msg: DB 'Введите строку: ',0h

SECTION .bss
buf1: RESB 80

SECTION .text
   GLOBAL _start
   _start:
   mov eax, msg
   call sprint

mov edx, 80

call sread
call quit
```

Рис. 4.12: Отредактированный файл

Снова транслирую файл, выполняю компоновку созданного объектного файла, запускаю новый исполняемый файл (рис. 4.13).

```
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку:
киселева Лиза
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-2.asm
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2 lab5-2.o
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-2
Введите строку: Киселева лиза
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 4.13: Исполнение файла

Разница между первым исполняемым файлом lab5-2 и вторым, что запуск первого запрашивает ввод с новой строки, а программа, которая исполняется при запуске второго, запрашивает ввод без переноса на новую строку, потому что в этом заключается различие между подпрограммами sprintLF и sprint.

#### 4.4 Выполнение заданий для самостоятельной работы

Создаю копию файла lab5-1.asm с именем lab5-1-1.asm с помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 4.14).

```
Lab5-1-1.asm [-M--] 9 L:[ 1+16 17/ 34] *(218 / 390b) 0010 0х00А [*][X]
SECTION .data
msg: DB 'Введите строку', 10

msgLen: EQU $-msg

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
   _start:
...
mov eax,4
mov ebx,1
mov ecx,msg
mov edx,msgLen
int 80h

mov eax, 3
mov ebx, 0
mov ecx, buf1
mov edx, 80
```

Рис. 4.14: Копирование и редактирование файла

Создаю объектный файл lab5-1-1.о, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-1-1, запускаю полученный исполняемый файл. Программа запрашивает ввод, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. 4.15).

```
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ nasm -f elf lab5-1-1.asm
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $ ./lab5-1-1
Введите строку
Киселева Лиза
Киселева Лиза
eakiseleva1@dk3n55 ~/work/arch-pc/lab05 $
```

Рис. 4.15: Исполнение файла

Создаю копию файла lab5-2.asm с именем lab5-2-2.asm с помощью функциональной клавиши F5 (рис. 4.16).

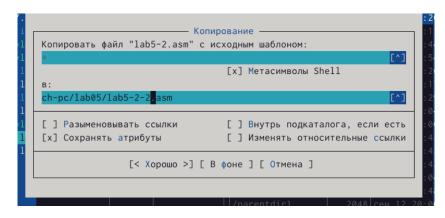


Рис. 4.16: Копирование файла

С помощью функциональной клавиши F4 открываю созданный файл для редактирования. Изменяю программу так, чтобы кроме вывода приглашения и запроса ввода, она выводила вводимую пользователем строку (рис. 4.17).

```
lab5-2-2.asm [----] 0 L:[ 6+21 27/ 27] *(316 / 316b) <EOF> [*][X]
SECTION .bss
buf1: RESB 80

SECTION .text
    GLOBAL _start
    _start:
    mov eax, msg
    call sprint
...
    mov ecx, buf1
    mov edx, 80
...
    call sread
...
    mov eax,4
    mov ebx,1
    mov ecx,buf1
    int 80h
...
    call quit
```

Рис. 4.17: Редактирование файла

Создаю объектный файл lab5-2-2.о, отдаю его на обработку компоновщику, получаю исполняемый файл lab5-2-2, запускаю полученный исполняемый файл. Программа запрашивает ввод без переноса на новую строку, ввожу свои ФИО, далее программа выводит введенные мною данные (рис. 4.18).

```
nasm -f elf lab5-2-2.asm eakiseleva1@dk3n55 -/work/arch-pc/lab05 $ nasm -f elf lab5-2-2.asm eakiseleva1@dk3n55 -/work/arch-pc/lab05 $ ld -m elf_i386 -o lab5-2-2 lab5-2-2.o eakiseleva1@dk3n55 -/work/arch-pc/lab05 $ ./lab5-2-2
Введите строку: Киселева Лиза
Киселева Лиза
eakiseleva1@dk3n55 -/work/arch-pc/lab05 $
```

Рис. 4.18: Исполнение файла

## 5 Выводы

При выполнении данной лабораторной работы я приобрела практические навыки работы в Midnight Commander, а также освоила инструкции языка ассемблера mov и int.

# Список литературы

1. Лабораторная работа №5