Отчёт по лабораторной работе №5

Дисциплина: Архитектура компьютера

Абакумов Тимофей Александрович

Содержание

# 1 Цель работы

Приобретение практических навыков работы в Midnight Commander. Освоение инструкций языка ассемблера mov и int

# 2 Задание

**Порядок выполнения лабораторной работы**

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

# 3 Выполнение лабораторной работы

## 3.1 Основы работы с mc

1. Для начала откроем Midnight Commander с помощью команды mc (рис. 1).

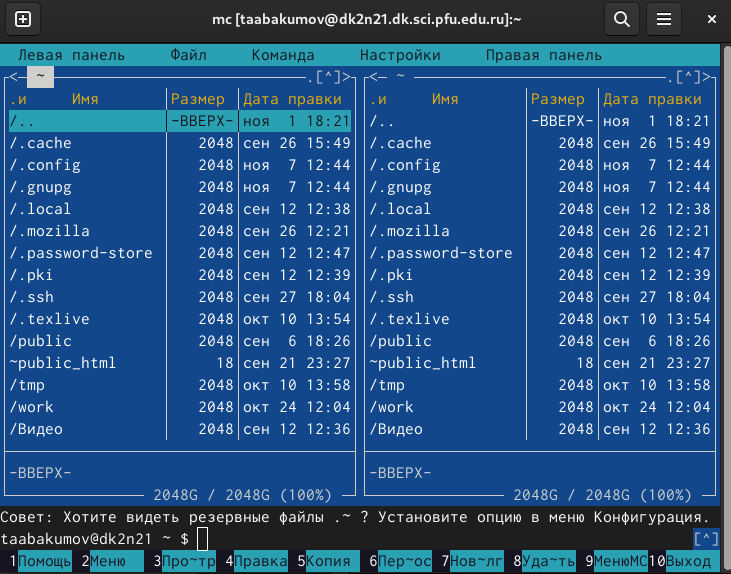


Рис. 1: Midnight Commander

1. Пользуясь клавишами ↑ , ↓ и Enter перейдём в каталог ~/work/study/2024-2025/arch-pc/labs/lab05 (рис. 2).

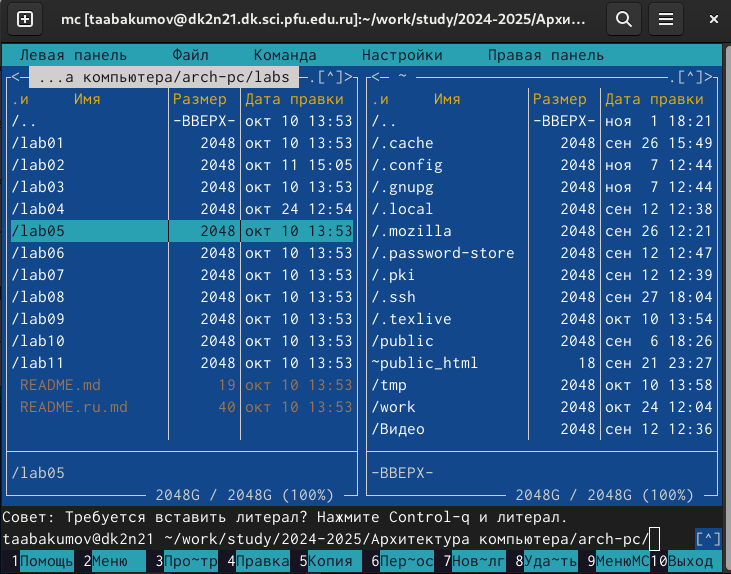


Рис. 2: Перемещение между директориями

1. Пользуясь строкой ввода и командой touch создадим файл lab5-1.asm (рис. 3).

Рис. 3: Создание файла

Рис. 3: Создание файла

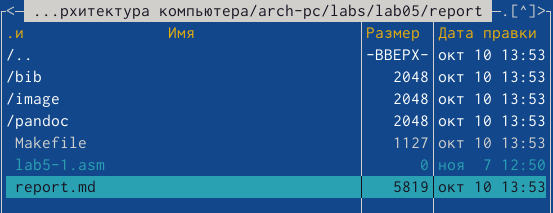


Рис. 4: Проверка создания файла

## 3.2 Структура программы на языке ассемблера NASM

1. С помощью функциональной клавиши F4 откройте файл lab5-1.asm для редактирования во встроенном редакторе (рис. 5).

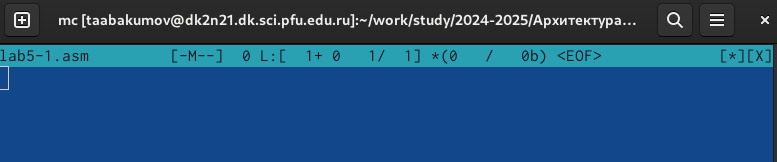


Рис. 5: Открытие файла

1. Введём текст программы из листинга 5.1, затем сохраним изменения и закроем файл (рис. 6).

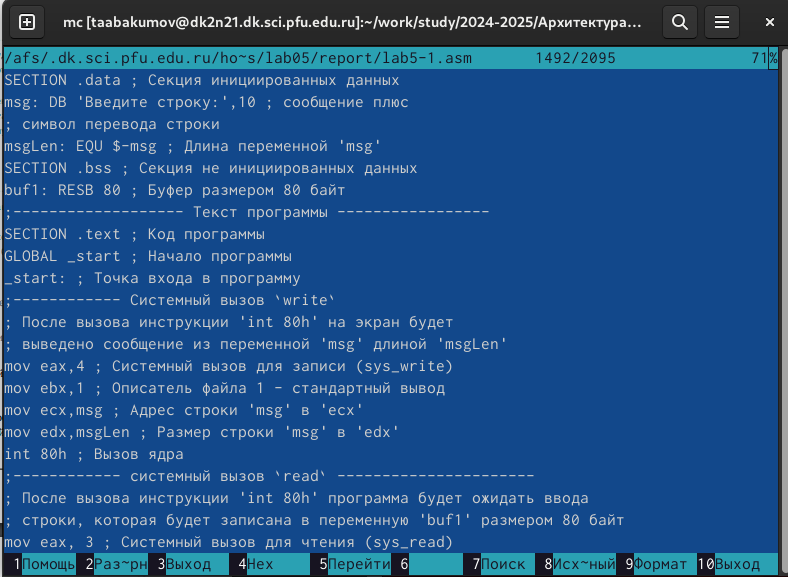


Рис. 6: Ввод текста

1. Оттранслируем текст программы lab5-1.asm в объектный файл. Выполним компоновку объектного файла и запустим получившийся исполняемый файл. Программа выводит строку ‘Введите строку:’ и ожидает ввода с клавиатуры. На запрос введём свою ФИО (рис. 7).

Рис. 7: Компановка файла

Рис. 7: Компановка файла

Рис. 8: Запуск файла

Рис. 8: Запуск файла

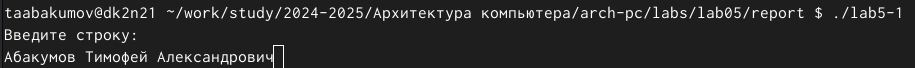


Рис. 9: Вывод результата

## 3.3 Подключение внешнего файла

1. Скачаем файл in\_out.asm со страницы курса в ТУИС (рис. 10).

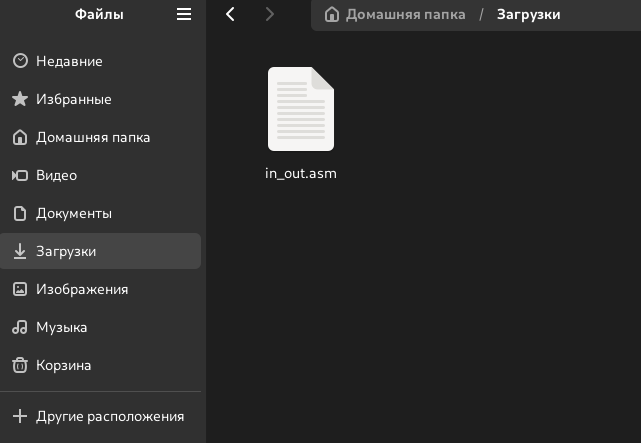


Рис. 10: Скачивание файла

1. Подключаемый файл in\_out.asm должен лежать в том же каталоге, что и файл с про- граммой, в которой он используется, поэтому перемещаем его из папки Загрузки (рис. 11).

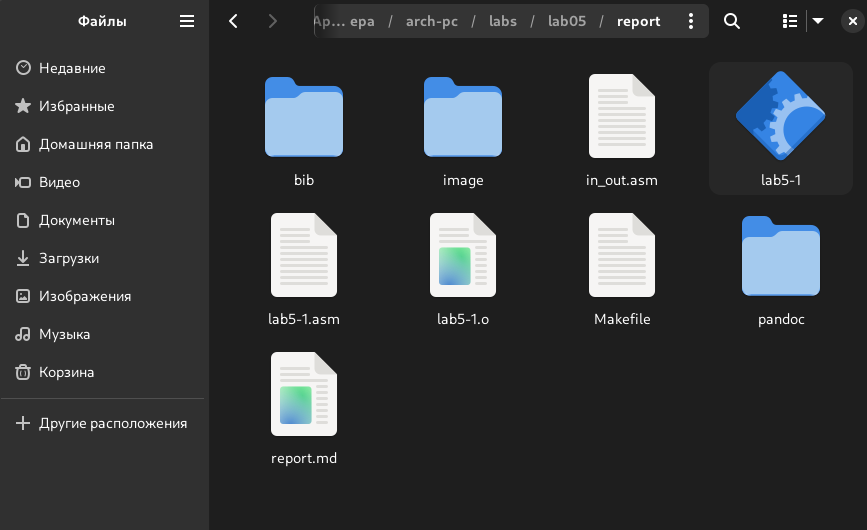


Рис. 11: Перемещение файла

* 1. С помощью функциональной клавиши F5 создадим копию файла lab5-1.asm с именем lab5-2.asm. Выделим файл lab5-1.asm, нажмём клавишу F5, введём имя файла lab5-2.asm и нажмём клавишу Enter (рис. 12).

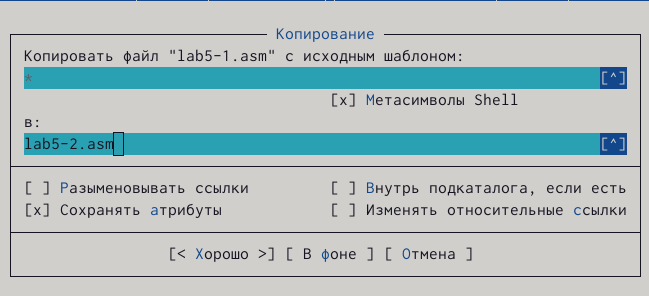


Рис. 12: Копирование файла

(рис. 13).

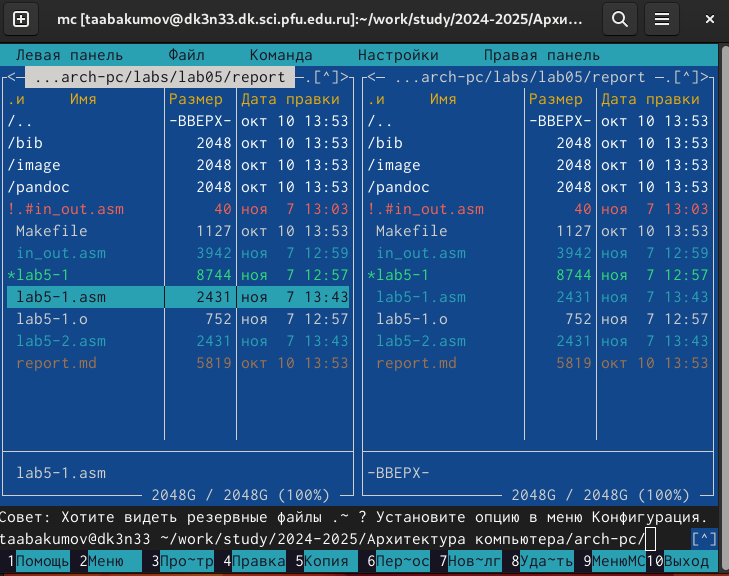


Рис. 13: Проверка успешности копировня

1. Исправим текст программы в файле lab5-2.asm с использованием подпрограмм из внешнего файла in\_out.asm (используем подпрограммы sprintLF, sread и quit) в соответствии с листингом 5.2. Создадим исполняемый файл и проверим его работу (рис. 14).

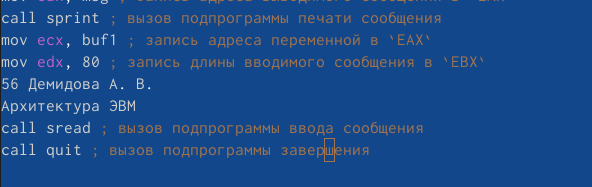


Рис. 14: Исправление текста



Рис. 15: Создание и проверка исполняемого файла

## 3.4 Задания для самостоятельной работы

1. Создадим копию файла lab5-1.asm. Внесём изменения в программу (без использования внешнего файла in\_out.asm), так чтобы она работала по следующему алгоритму: • вывести приглашение типа “Введите строку:”; • ввести строку с клавиатуры; • вывести введённую строку на экран. Проверим работу исполняемого файла. На приглашение ввести строку введём свою ФИО.(рис. 16).

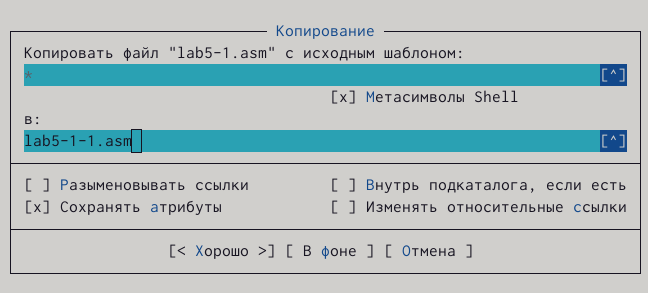


Рис. 16: Создание копии файла

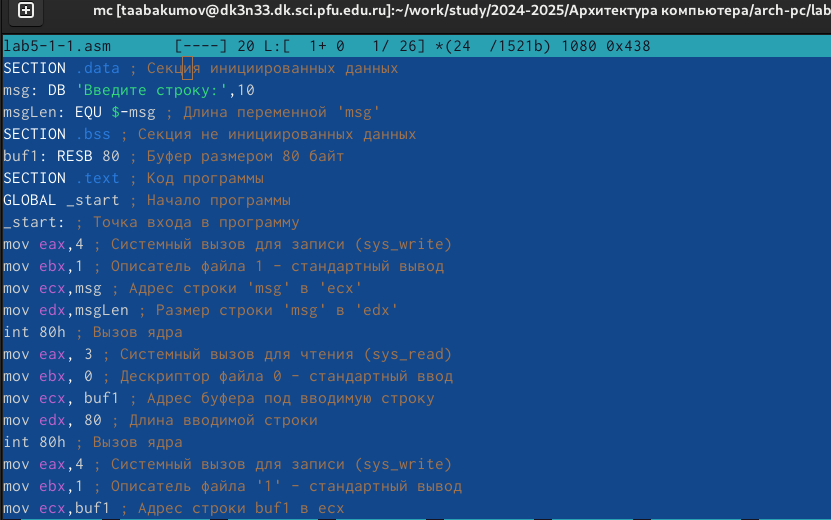


Рис. 17: Изменения программы



Рис. 18: Проверка работы файла

Код программы из пункта 1:

SECTION .data ; Секция инициированных данных

msg: DB ‘Введите строку:’,10

msgLen: EQU $-msg ; Длина переменной ‘msg’

SECTION .bss ; Секция не инициированных данных

buf1: RESB 80 ; Буфер размером 80 байт

SECTION .text ; Код программы

GLOBAL \_start ; Начало программы

\_start: ; Точка входа в программу

mov eax,4 ; Системный вызов для записи (sys\_write)

mov ebx,1 ; Описатель файла 1 - стандартный вывод

mov ecx,msg ; Адрес строки ‘msg’ в ‘ecx’

mov edx,msgLen ; Размер строки ‘msg’ в ‘edx’

int 80h ; Вызов ядра

mov eax, 3 ; Системный вызов для чтения (sys\_read)

mov ebx, 0 ; Дескриптор файла 0 - стандартный ввод

mov ecx, buf1 ; Адрес буфера под вводимую строку

mov edx, 80 ; Длина вводимой строки

int 80h ; Вызов ядра

mov eax,4 ; Системный вызов для записи (sys\_write)

mov ebx,1 ; Описатель файла ‘1’ - стандартный вывод

mov ecx,buf1 ; Адрес строки buf1 в ecx

mov edx,buf1 ; Размер строки buf1

int 80h ; Вызов ядра

mov eax,1 ; Системный вызов для выхода (sys\_exit)

mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)

int 80h ; Вызов ядра

1. Создадим копию файла lab5-2.asm. Исправим текст программы с использование подпрограмм из внешнего файла in\_out.asm, так чтобы она работала по следующему алгоритму: • вывести приглашение типа “Введите строку:”; • ввести строку с клавиатуры; • вывести введённую строку на экран (рис. 19).

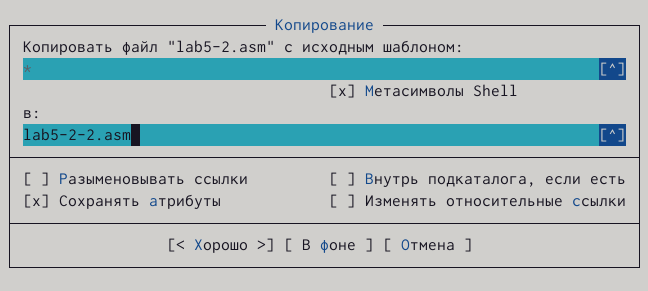


Рис. 19: Копирование файла

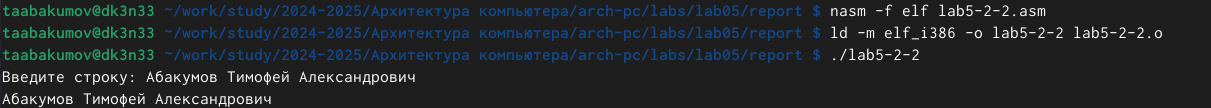


Рис. 20: Проверка работы программы

Код программы из пункта 2:

%include ‘in\_out.asm’

SECTION .data ; Секция инициированных данных

msg: DB ‘Введите строку:’,0h ; сообщение

SECTION .bss ; Секция не инициированных данных

buf1: RESB 80 ; Буфер размером 80 байт

SECTION .text ; Код программы

GLOBAL \_start ; Начало программы

\_start: ; Точка входа в программу

mov eax, msg ; запись адреса выводимого сообщения в EAX

call sprint ; вызов подпрограммы печати сообщения

mov ecx, buf1 ; запись адреса переменной в EAX

mov edx, 80 ; запись длины вводимого сообщения в EBX

call sread ; вызов подпрограммы ввода сообщения

mov eax,4 ; Системный вызов для записи (sys\_write)

mov ebx,1 ; Описатель файла ‘1’ - стандартный вывод

mov ecx,buf1 ; Адрес строки buf1 в ecx

int 80h ; Вызов ядра

call quit ; вызов подпрограммы завершения

# 4 Выводы

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