

Лабораторная работа №5

Основы работы с Mindight Commander. Структура программы на языке ассемблера NASM. Системные вызовы в ОС GNU Linux

Славинский Владислав Вадимович

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Выполнение задания для самостоятельной работы	12
4	Вывод	17

Список иллюстраций

2.1	Создание lab5-1	6
2.2	Создание программы	7
2.3	Создание исполняемого файла	8
2.4	Подключение внешнего файла	9
2.5	Новый код программы	10
2.6	Создание исполняемого файла	10
2.7	Замена <code>sprintf</code> на <code>sprint</code>	11
3.1	Изменения	13
3.2	Запуск	14
3.3	Изменения2	15
3.4	Запуск2	16

Список таблиц

1 Цель работы

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

2 Выполнение лабораторной работы

1) Создал новый каталог lab05, перешел в него и создал файл lab5-1.asm: (Рис. 2.1)

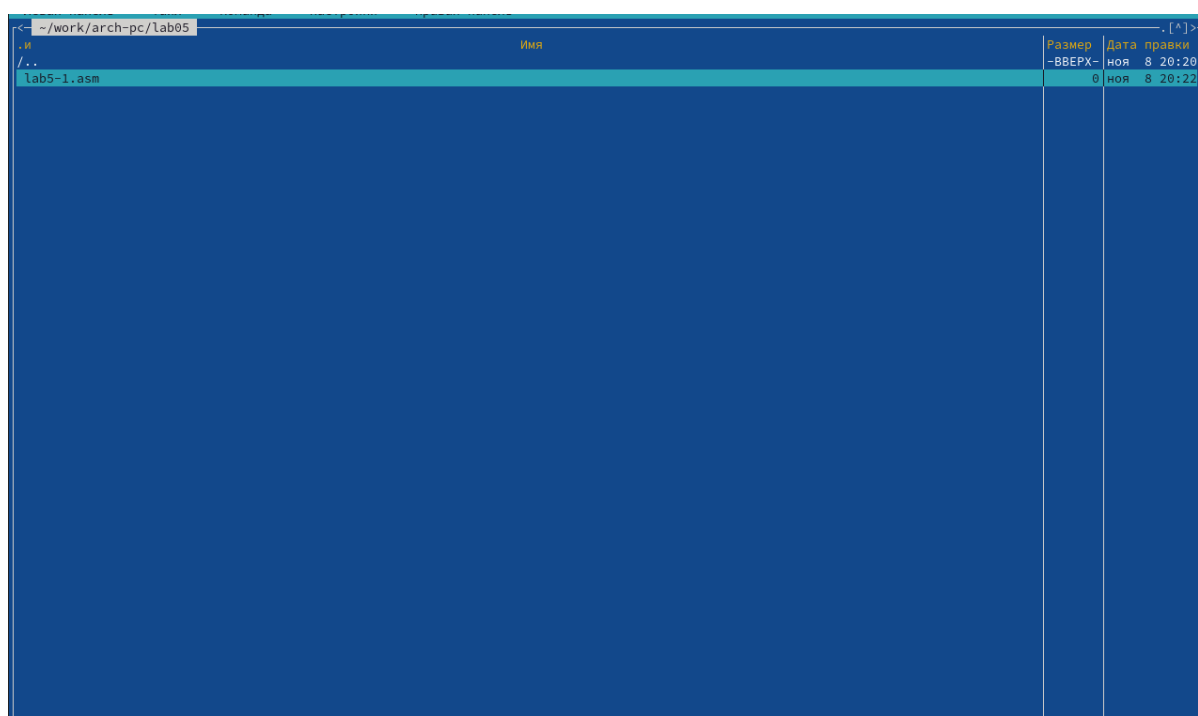


Рис. 2.1: Создание lab5-1

2) Открыл файл с помощью клавиши f4 и заполнил в него программу вывода сообщения на экран и ввода строки с клавиатуры: (Рис. 2.2)

```

;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов `write` -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
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
;----- системный вызов `read` -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
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
;----- Системный вызов `exit` -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

```

Рис. 2.2: Создание программы

3) Убедился, что файл содержит текст программы, а после оттранслировал и ввел в строку своё ФИО: (Рис. 2.3)

```
flory@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-1.asm
flory@vbox:~/work/arch-pc/lab05$ mc

flory@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
ld: невозможно найти lab5-1.o: Нет такого файла или каталога
flory@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-1 lab5-1.o
flory@vbox:~/work/arch-pc/lab05$ ./lab5-1.nasm
bash: ./lab5-1.nasm: Нет такого файла или каталога
flory@vbox:~/work/arch-pc/lab05$ ./lab5-1
bash: ./lab5-1: Нет такого файла или каталога
flory@vbox:~/work/arch-pc/lab05$ ./lab5-1
Введите строку:
Славинский Владислав Вадимович
flory@vbox:~/work/arch-pc/lab05$
```

Рис. 2.3: Создание исполняемого файла

4) Подключил внешний файл in_out.asm, после чего скопировал его в папку, где находится файл lab5-1. Потом скопировал файл lab5-1 и переименовал его в lab5-2: (Рис. 2.4)

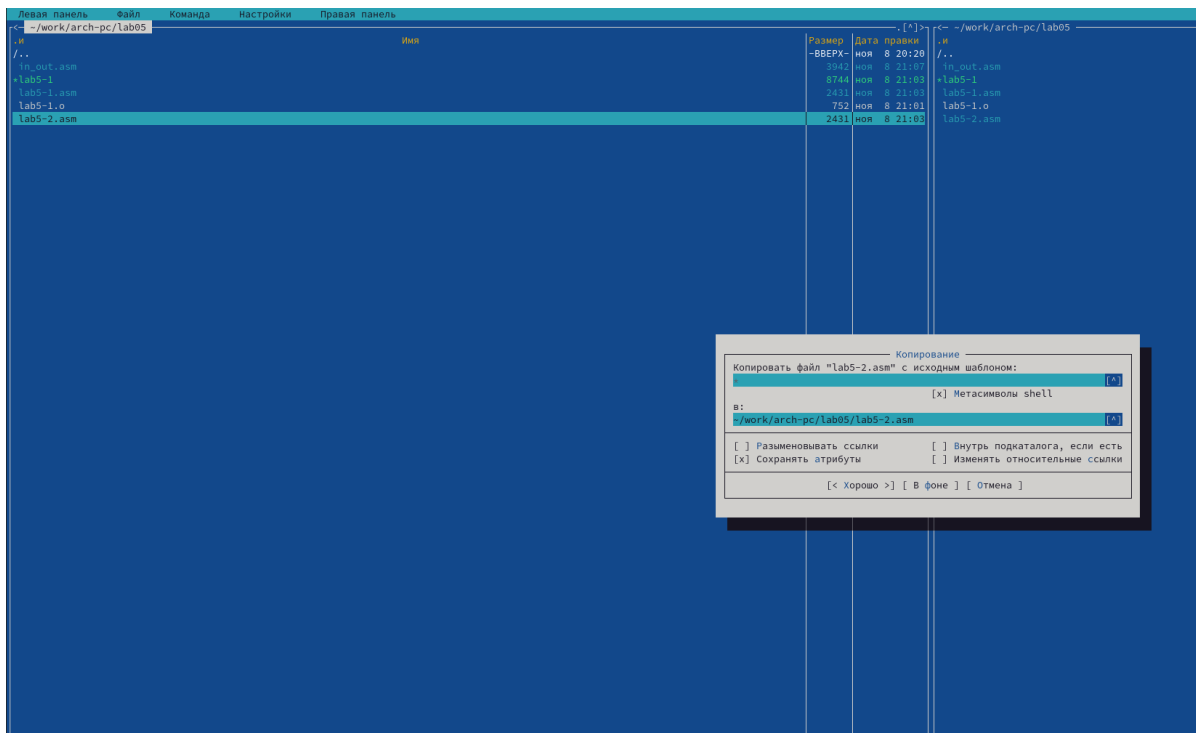


Рис. 2.4: Подключение внешнего файла

5) Ввел код программы вывода сообщения на экран и ввода строки с клавиатуры с использованием файла in_out.asm (Рис. 2.5)

```

lab5-2.asm          [=====] 41 L: 1*16 17/ 17] * (1224/1224b) <EOF>
-----
; Программа вывода сообщения на экран и ввода строки с клавиатур
-----
#include "in_out.asm" ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB "Введите строку: \n" ; сообщение
SECTION .bss ; Секция не инициализированных 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 sprintf ; вызов подпрограммы печати сообщения
mov ecx, buf1 ; запись адреса переменной в 'ECX'
mov edx, 80 ; запись длины вводимого сообщения в 'EDX'
call read ; вызов подпрограммы ввода сообщения
call quit ; вызов подпрограммы завершения

```

Рис. 2.5: Новый код программы

6)Оттранслировал lab5-2 (Рис. 2.6)

```

flory@vbox:~/work/arch-pc/lab05$ nasm -f elf
in_out.asm lab5-1      lab5-1.asm lab5-1.o   lab5-2.asm
flory@vbox:~/work/arch-pc/lab05$ nasm -f elf lab5-2.asm
flory@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o lab5-2 lab5-2.o
flory@vbox:~/work/arch-pc/lab05$ ./lab5-2
Введите строку:
Славинский Владислав Вадимович
flory@vbox:~/work/arch-pc/lab05$

```

Рис. 2.6: Создание исполняемого файла

7)Заменил в lab5-2 sprintf на printf и заметил, что после вывода сообщения нет перехода на новую строку.(Рис. 2.7)

```

;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
%include 'in_out.asm' ; подключение внешнего файла
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
call quit ; вызов подпрограммы завершения

```

Рис. 2.7: Замена sprintLF на sprint

3 Выполнение задания для самостоятельной работы

1)Сделал копию файла lab5-1 и внес изменения по заданию (Рис. 3.1)

```

;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
;----- Объявление переменных -----
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку:',10 ; сообщение плюс
; символ перевода строки
msgLen: EQU $-msg ; Длина переменной 'msg'
SECTION .bss ; Секция не инициализированных данных
buf1: RESB 80 ; Буфер размером 80 байт
;----- Текст программы -----
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
;----- Системный вызов `write` -----
; После вызова инструкции 'int 80h' на экран будет
; выведено сообщение из переменной 'msg' длиной 'msgLen'
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
;----- системный вызов `read` -----
; После вызова инструкции 'int 80h' программа будет ожидать ввода
; строки, которая будет записана в переменную 'buf1' размером 80 байт
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
mov ebx, 1
mov ecx, buf1
mov edx, 80
int 80h
;----- Системный вызов `exit` -----
; После вызова инструкции 'int 80h' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 80h ; Вызов ядра

```

Рис. 3.1: Изменения

2)Оттранслировал текст программы и проверил его на работу (Рис. 3.2)

```
flory@vbox:~/work/arch-pc/lab05$ nc
flory@vbox:~/work/arch-pc/lab05$ nasm -f elf laba5-1.asm
flory@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o laba5-1 laba5-1.o
flory@vbox:~/work/arch-pc/lab05$ ./laba5-1
Введите строку:
Славинский Владислав Вадимович
Славинский Владислав Вадимович
flory@vbox:~/work/arch-pc/lab05$
```

Рис. 3.2: Запуск

3)Так же сделал и с файлом lab5-2 и внес изменения, учитывая файл in_out.asm(Рис. 3.3)

```

Laba5-2.asm [----] 11 L: [ 1+17 18/ 19] *(1179/1248b) 0010 0x00A
;-----
; Программа вывода сообщения на экран и ввода строки с клавиатуры
;-----
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; Секция инициализированных данных
msg: DB 'Введите строку: ',0h ; сообщение
SECTION .bss ; Секция не инициализированных 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, buf1
call sprint
call quit ; вызов подпрограммы завершения

```

Рис. 3.3: Изменения2

4)Оттранслировал текст программы и также проверил его на работу(Рис. 3.4)

```
flory@vbox: ~/work/arch-pc/lab05$  
flory@vbox:~/work/arch-pc/lab05$ nasm -f elf laba5-2.asm  
flory@vbox:~/work/arch-pc/lab05$ ld -m elf_i386 -o laba5-2 laba5-2.o  
flory@vbox:~/work/arch-pc/lab05$ ./laba5-2  
Введите строку: Славинский Владислав Вадимович  
Славинский Владислав Вадимович  
flory@vbox:~/work/arch-pc/lab05$
```

Рис. 3.4: Запуск2

4 Вывод

В ходе выполнения лабораторной работы я приобрел практические навыки для работы в Midnight Commander. Также я освоил инструкции языка ассемблера `mov` и `int`.