

Отчет по лабораторной работе №5

Дисциплина: Архитектура компьютера

Стрижов Дмитрий Павлович

Содержание

1	Цель работы	3
2	Задание	4
3	Выполнение лабораторной работы	5
3.1	Начальная работа в Midnight Commander	5
3.2	Подключение внешнего файла in_out.asm	7
3.3	Задание для самостоятельной работы	8
4	Выводы	10
	Список литературы	11

1 Цель работы

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

2 Задание

1. Начальная работа в Midnight Commander
2. Подключение внешнего файла in_out.asm
3. Задание для самостоятельной работы

3 Выполнение лабораторной работы

3.1 Начальная работа в Midnight Commander

Открываю Midnight Commander и перехожу в ~/work/arch-pc и создаю каталог lab05, нажимая на клавишу F7 (рис. 3.1).

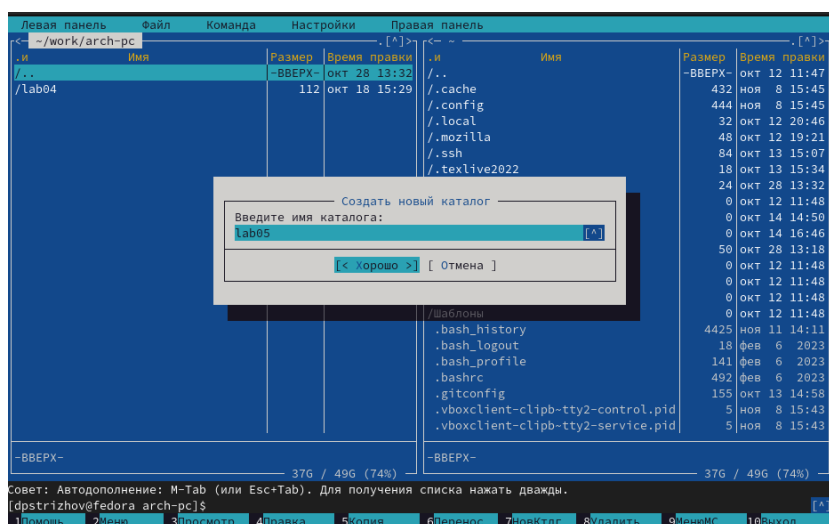


Рис. 3.1: Создание каталога lab05

Создаю lab5-1.asm (рис. 3.2).

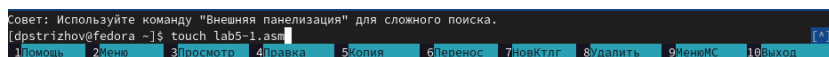


Рис. 3.2: Создание lab5-1.asm

Открываю lab5-1.asm и ввожу текст программы (рис. 3.3).

```

mc [dpstrizhov@fedora:~/work/arch-pc/lab05]
GNU nano 7.2 /home/dpstrizhov/work/arch-pc/lab05/lab5-1.asm
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 0x80' 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 0x80; Вызов ядра
----- системный вызов 'read' -----
; После вызова инструкции 'int 0x80' 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 0x80; Вызов ядра
----- Системный вызов 'exit' -----
; После вызова инструкции 'int 0x80' программа завершит работу
mov eax, 1; Системный вызов для выхода (sys_exit)
mov ebx, 0; Выход с кодом возврата 0 (без ошибок)
int 0x80; Вызов ядра

```

Рис. 3.3: Работа с lab5-1.asm

Убеждаюсь в том, что в файле имеется код (рис. 3.4).

```
dpstrizhov@fedora:~/work/study/2023-2024/Архитекту...  x  mc [dpstrizhov@fedora]:~/work/arch-pc/lab05  x
/home/dpstrizhov/work/arch-pc/lab05/, #lab5-1.asm  2402/2433  98%
; 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 0xh' 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 0xh ; Вызов ядра
;----- Системный вызов 'read' -----
; После вызова инструкции 'int 0xh' 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 0xh ; Вызов ядра
;----- Системный вызов 'exit' -----
; После вызова инструкции 'int 0xh' программа завершит работу
mov eax,1 ; Системный вызов для выхода (sys_exit)
mov ebx,0 ; Выход с кодом возврата 0 (без ошибок)
int 0xh ; Вызов ядра
```

Рис. 3.4: Проверка lab5-1.asm

3.2 Подключение внешнего файла in_out.asm

Скачиваю необходимый файл с ТУИСа, перемещаю его в директорию, где находятся мои программы. Создаю копию файла lab5-1.asm (рис. 3.5).

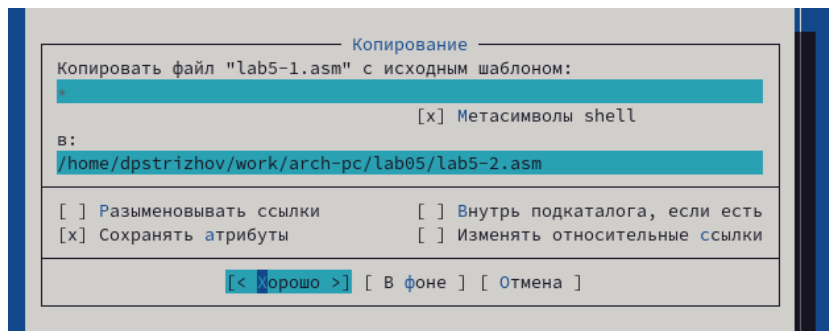


Рис. 3.5: Копия lab5-1.asm

Исправляю текст программы lab5-2.asm, а затем меняю sprintLF на sprint (рис. 3.6).



Рис. 3.6: Текст программы lab5-2.asm

Разница между первой и второй командой заключается в том, что при первой идет переход на следующую строку после вывода какой-то строки.

3.3 Задание для самостоятельной работы

№1 Создаю копию файла lab5-1.asm. Вношу изменения в программу (без использования внешнего файла in_out.asm), так чтобы она работала по следующему алгоритму(рис. 3.7):

- вывести приглашение типа “Введите строку:”;
- ввести строку с клавиатуры;
- вывести введенную строку на экран.

```
-----системный вызов 'write'-----  
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
```

Рис. 3.7: Изменения в копии lab5-1.asm

№2 Получаю исполняемый файл и проверяю его работу(рис. 3.8).

```
[dpstrizhov@fedora lab05]$ nasm -f elf lab5-1-1.asm  
[dpstrizhov@fedora lab05]$ ld -m elf_i386 -o lab5-1-1 lab5-1-1.o  
[dpstrizhov@fedora lab05]$ ./lab5-1-1  
Введите строку:  
Стрижов Дмитрий  
Стрижов Дмитрий  
[dpstrizhov@fedora lab05]$
```

Рис. 3.8: Проверка изменений

№3 Создаю копию файла lab5-2.asm. Исправляю текст программы с использование подпрограмм из внешнего файла in_out.asm, так чтобы она работала по следующему алгоритму(рис. 3.9):

- вывести приглашение типа “Введите строку:”;
- ввести строку с клавиатуры;
- вывести введенную строку на экран.

```
call sread ; вызов подпрограммы ввода сообщения  
call sprint; вызов подпрограммы для печати сообщения  
call quit ; вызов подпрограммы завершения
```

Рис. 3.9: Работа с копией lab5-2.asm

№4 Получаю исполняемый файл и проверяю его работу(рис. 3.10).


```
[dpstrizhov@fedora lab05]$ nasm -f elf lab5-2-1.asm
[dpstrizhov@fedora lab05]$ ld -m elf_i386 -o lab5-2-1 lab5-2-1.o
[dpstrizhov@fedora lab05]$ ./lab5-2-1
Введите строку: Стрижов Дмитрий
Стрижов Дмитрий
[dpstrizhov@fedora lab05]$
```

Рис. 3.10: Работа копии lab5-2.asm

4 Выводы

За время выполнения данной лабораторной работы я освоил инструкции ассемблера `mov` и `int`, а также получил навыки работы в Midnight Commander

Список литературы

Осваиваем эффективную работу в Midnight Commander. Ссылка: https://interface31.ru/tech_it/2effektivnuyu-rabotu-v-midnight-commander.html