

Отчёт по лабораторной работе №10

Дисциплина: Архитектура компьютера

Осина Виктория Александровна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	11
5	Выполнение задания для самостоятельной работы	15
6	Выводы	19
7	Список литературы	20

Список иллюстраций

4.1	Создание каталога lab010 и файлов в нем	11
4.2	Копирование файла in_out.asm	11
4.3	Проверка, что файл находится в нужном каталоге	12
4.4	Открытие файла в редакторе	12
4.5	Ввод текста программы в файл	12
4.6	Создание исполняемого файла и его запуск	13
4.7	Изменение прав доступа к исполняемому файлу	13
4.8	Изменение прав доступа к файлу	13
4.9	Предоставление прав для файла readme-1.txt	14
4.10	Предоставление прав для файла readme-2.txt	14
5.1	Ввод текста программы в файл	16
5.2	Выполнение программы, проверка наличия файла и его содержимого	16

Список таблиц

1 Цель работы

Целью данной лабораторной работы является приобретение навыков написания программ для работы с файлами.

2 Задание

1. Выполнение лабораторной работы
2. Выполнение задания для самостоятельной работы

3 Теоретическое введение

Права доступа к файлам

Права доступа определяют набор действий (чтение (r), запись (r), выполнение (x)), разрешённых для выполнения пользователям системы над файлами, дефис (-) - право не дано. Для каждого файла пользователь может входить в одну из трех групп: владелец (u), член группы владельца (g), все остальные (o). Для каждой из этих групп может быть установлен свой набор прав доступа. Для предоставления прав доступа другому пользователю или другой группе командой

```
chown [ключи] <новый_пользователь>[:новая_группа] <файл>
```

или

```
chgrp [ключи] < новая_группа > <файл>
```

Права доступа rw- (чтение и запись, без исполнения) понимаются как три двоичные цифры 110 или как восьмеричная цифра 6

Полная строка прав доступа в символьном представлении имеет вид:

```
<права_владельца> <права_группы> <права_остальных>
```

Свойства (атрибуты) файлов и каталогов можно вывести на терминал с помощью команды ls с ключом -l.

Тип файла определяется первой позицией, это может быть: каталог — d, обычный файл — дефис (-) или символьная ссылка на другой файл — l.

Для изменения прав доступа служит команда chmod, которая понимает как символьное, так и числовое указание прав.

Формат символьного режима:

```
chmod <категория><действие><набор_прав><файл>
```

Работа с файлами средствами Nasm

Обработка файлов в операционной системе Linux осуществляется за счет использования определенных системных вызовов. Для корректной работы и доступа к файлу при его открытии или создании, файлу присваивается уникальный номер (16-битное целое число) – дескриптор файла.

Общий алгоритм работы с системными вызовами в Nasm можно представить в следующем виде:

1. Поместить номер системного вызова в регистр EAX;
2. Поместить аргументы системного вызова в регистрах EBX, ECX и EDX;
3. Вызов прерывания (int 80h);
4. Результат обычно возвращается в регистр EAX.

Открытие и создание файла

Для создания и открытия файла служит системный вызов `sys_creat`, который использует следующие аргументы: права доступа к файлу в регистре ECX, имя файла в EBX и номер системного вызова `sys_creat` (8) в EAX.

Для открытия существующего файла служит системный вызов `sys_open`, который использует следующие аргументы: права доступа к файлу в регистре EDX, режим доступа к файлу в регистр ECX, имя файла в EBX и номер системного вызова `sys_open` (5) в EAX. Среди режимов доступа к файлам чаще всего используются:

- (0) – `O_RDONLY` (открыть файл в режиме только для чтения);
- (1) – `O_WRONLY` – (открыть файл в режиме только записи);
- (2) – `O_RDWR` – (открыть файл в режиме чтения и записи).

Системный вызов возвращает файловый дескриптор открытого файла в регистр EAX. В случае ошибки, код ошибки также будет находиться в регистре EAX.

Запись в файл

Для записи в файл служит системный вызов `sys_write`, который использует следующие аргументы: количество байтов для записи в регистре `EDX`, строку содержимого для записи `ECX`, файловый дескриптор в `EBX` и номер системного вызова `sys_write` (4) в `EAX`. Системный вызов возвращает фактическое количество записанных байтов в регистр `EAX`. В случае ошибки, код ошибки также будет находиться в регистре `EAX`. Прежде чем записывать в файл, его необходимо создать или открыть, что позволит получить дескриптор файла.

Чтение файла

Для чтения данных из файла служит системный вызов `sys_read`, который использует следующие аргументы: количество байтов для чтения в регистре `EDX`, адрес в памяти для записи прочитанных данных в `ECX`, файловый дескриптор в `EBX` и номер системного вызова `sys_read` (3) в `EAX`. Как и для записи, прежде чем читать из файла, его необходимо открыть, что позволит получить дескриптор файла.

Закрытие файла

Для правильного закрытия файла служит системный вызов `sys_close`, который использует один аргумент – дескриптор файла в регистре `EBX`. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов возвращает код ошибки в регистр `EAX`.

Изменение содержимого файла

Для изменения содержимого файла служит системный вызов `sys_lseek`, который использует следующие аргументы: исходная позиция для смещения `EDX`, значение смещения в байтах в `ECX`, файловый дескриптор в `EBX` и номер системного вызова `sys_lseek` (19) в `EAX`.

Значение смещения можно задавать в байтах. Значения обозначающие исходную позицию могут быть следующими:

- (0) – `SEEK_SET` (начало файла)
- (1) – `SEEK_CUR` (текущая позиция);

- (2) – SEEK_END (конец файла).

В случае ошибки, системный вызов возвращает код ошибки в регистр EAX.

Удаление файла

Удаление файла осуществляется системным вызовом `sys_unlink(10)`, который использует один аргумент – имя файла в регистре EBX.

4 Выполнение лабораторной работы

Создаю каталог lab10 для программ лабораторной работы №10, перехожу в него и создаю файл lab10-1.asm, readme-1.txt и readme-2.txt (рис. 4.1)

```
[vaosina@fedora ~]$ mkdir ~/work/arch-pc/lab10
[vaosina@fedora ~]$ cd ~/work/arch-pc/lab10
[vaosina@fedora lab10]$ touch lab10-1.asm readme-1.txt readme-2.txt
```

Рис. 4.1: Создание каталога lab010 и файлов в нем

Перед работой с программами копирую файл in_out.asm в каталог и проверяю, что файл находится в нужном каталоге (рис. 4.2) (рис. 4.3)

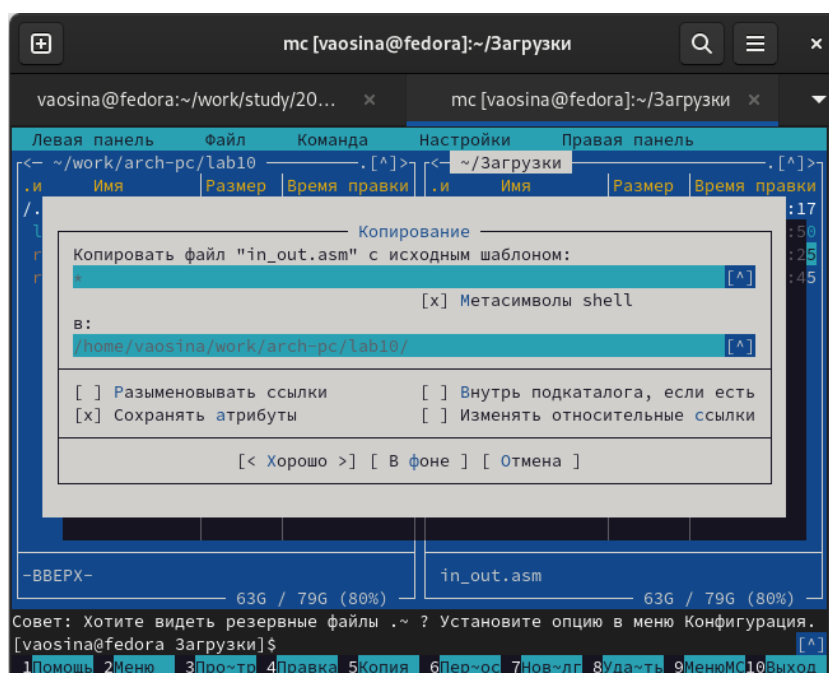


Рис. 4.2: Копирование файла in_out.asm

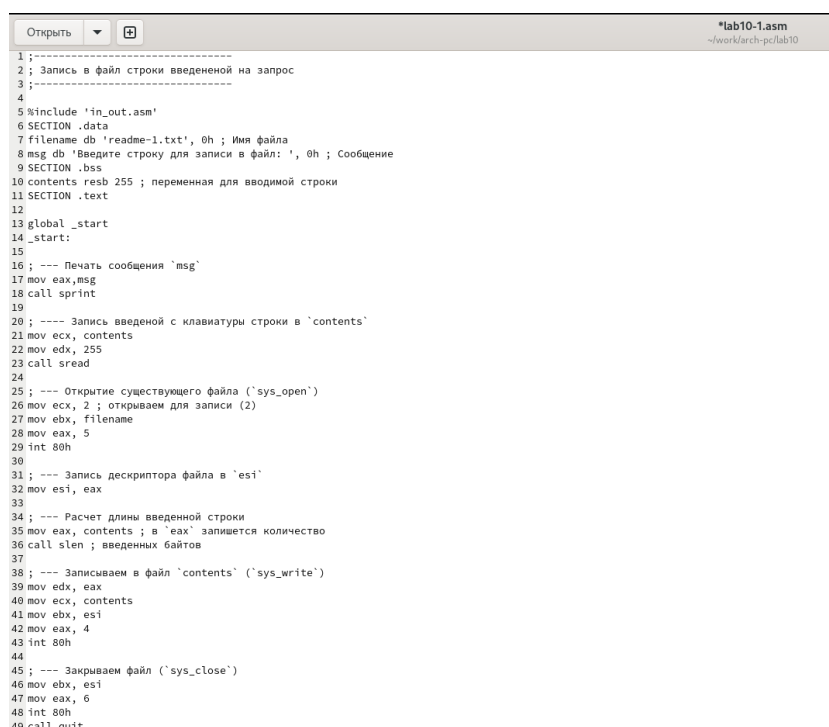
```
[vaosina@fedora lab10]$ ls
in_out.asm  lab10-1.asm  readme-1.txt  readme-2.txt
vaosina@fedora lab10$
```

Рис. 4.3: Проверка, что файл находится в нужном каталоге

Открываю lab10-1.asm в редакторе и ввожу в него текст программы записи в файл сообщения (рис. 4.4) и (рис. 4.5).

```
[vaosina@fedora lab10]$ gedit lab10-1.asm
```

Рис. 4.4: Открытие файла в редакторе



```

1 ;-----
2 ; Запись в файл строки введенной на запрос
3 ;-----
4
5 %include 'in_out.asm'
6 SECTION .data
7 filename db 'readme-1.txt', 0h ; Имя файла
8 msg db 'Введите строку для записи в файл: ', 0h ; Сообщение
9 SECTION .bss
10 contents resb 255 ; переменная для вводимой строки
11 SECTION .text
12
13 global _start
14 _start:
15
16 ; --- Печать сообщения 'msg'
17 mov eax, msg
18 call sprint
19
20 ; --- Запись введенной с клавиатуры строки в 'contents'
21 mov ecx, contents
22 mov edx, 255
23 call sread
24
25 ; --- Открытие существующего файла ('sys_open')
26 mov ecx, 2 ; открываем для записи (2)
27 mov ebx, filename
28 mov eax, 5
29 int 80h
30
31 ; --- Запись дескриптора файла в 'esi'
32 mov esi, eax
33
34 ; --- Расчет длины введенной строки
35 mov ecx, contents ; в 'eax' запишется количество
36 call slen ; введенных байтов
37
38 ; --- Записываем в файл 'contents' ('sys_write')
39 mov edx, eax
40 mov ecx, contents
41 mov ebx, esi
42 mov eax, 4
43 int 80h
44
45 ; --- Закрываем файл ('sys_close')
46 mov ebx, esi
47 mov eax, 6
48 int 80h
49 call quit

```

Рис. 4.5: Ввод текста программы в файл

Создаю исполняемый файл и запускаю его (рис. 4.6). Проверяю, записано ли сообщение в нужный файл, и вижу, что сообщение, введенное с клавиатуры, теперь действительно находится в файле readme-1.txt

```
[vaosina@fedora lab10]$ nasm -f elf -g -l lab10-1.lst lab10-1.asm
[vaosina@fedora lab10]$ ld -m elf_i386 -o lab10-1 lab10-1.o
[vaosina@fedora lab10]$ ./lab10-1
Введите строку для записи в файл: Hello world!!!
[vaosina@fedora lab10]$ cat readme-1.txt
Hello world!!!
```

Рис. 4.6: Создание исполняемого файла и его запуск

С помощью команды `chmod a-x` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение. Пытаюсь выполнить файл, после чего выводится надпись “Отказано в доступе”. Так выходит, потому что введённая мной команда, где “a” - “все пользователи”, “-” - отмена набора прав, а “x” - право на исполнение, и подразумевает собой для всех пользователей запрет на исполнение файла, поэтому я и не могу выполнить файл. (рис. 4.7)

```
[vaosina@fedora lab10]$ chmod a-x lab10-1
[vaosina@fedora lab10]$ ./lab10-1
bash: ./lab10-1: Отказано в доступе
```

Рис. 4.7: Изменение прав доступа к исполняемому файлу

С помощью команды `chmod a+x` изменяю права доступа к файлу `lab10-1.asm` с исходным текстом программы, добавив права на исполнение, и пытаюсь исполнить его. В этом случае также выдается ошибка, т.к. в файле нет команд для терминала. (рис. 4.8)

```
[vaosina@fedora lab10]$ chmod a+x lab10-1.asm
[vaosina@fedora lab10]$ ./lab10-1.asm
./lab10-1.asm: строка 1: синтаксическая ошибка рядом с неожиданным маркером «;»
./lab10-1.asm: строка 1: `;-----'
```

Рис. 4.8: Изменение прав доступа к файлу

В соответствии со своим вариантом (вариант 7) предоставляю права доступа `rw- rw- rw-` к файлу `readme-1.txt`, представленные в символьном виде. Проверяю правильность выполнения при помощи команды `ls -l`. Права предоставлены корректно. (рис. 4.9)

```
[vaosina@fedora lab10]$ chmod u=rw readme-1.txt
[vaosina@fedora lab10]$ chmod g=rwx readme-1.txt
[vaosina@fedora lab10]$ chmod o=rw readme-1.txt
[vaosina@fedora lab10]$ ls -l readme-1.txt
-rw-rwxrwx. 1 vaosina vaosina 15 дек 16 14:53 readme-1.txt
```

Рис. 4.9: Предоставление прав для файла readme-1.txt

Для файла readme-2.txt предоставляю права доступа 101 111 111, но т.к. я не разобралась, как их предоставить, используя двоичное представление, то я использую восьмеричное представление, т.е. 577. Проверяю правильность выполнения при помощи команды `ls -l`. Права предоставлены корректно. (рис. 4.10)

```
[vaosina@fedora lab10]$ chmod 577 readme-2.txt
[vaosina@fedora lab10]$ ls -l readme-2.txt
-r-xrwxrwx. 1 vaosina vaosina 0 дек 15 20:03 readme-2.txt
```

Рис. 4.10: Предоставление прав для файла readme-2.txt

5 Выполнение задания для самостоятельной работы

Создаю файл lab10-2.asm и ввожу в него текст программы (рис. 5.1), работающей по следующему алгоритму:

- Вывод приглашения “Как Вас зовут?”
- ввести с клавиатуры свои фамилию и имя
- создать файл с именем name.txt
- записать в файл сообщение “Меня зовут”
- дописать в файл строку введенную с клавиатуры
- закрыть файл

```

Открыть  *lab10-2.asm
~workbench-pc/lab10
3 filename db 'name.txt', 0h ; имя файла
4 msg1 db 'Как Вас зовут?', 0h ; сообщение
5 msg2 db 'Меня зовут ', 0h
6 SECTION .bss
7 contents resb 255 ; переменная для вводимой строки
8 SECTION .text
9
10 global _start
11 _start:
12
13 mov eax, msg1
14 call sprint
15
16 mov ecx, contents
17 mov edx, 255
18 call sread
19
20 mov ecx, 0777o
21 mov ebx, filename
22 mov eax, 8
23 int 80h
24
25 mov ecx, 2
26 mov ebx, filename
27 mov eax, 5
28 int 80h
29
30 mov esi, eax
31
32 mov eax, contents ; в 'eax' запишется количество
33 call slen ; введенных байтов
34
35 mov eax, msg2 ; в 'eax' запишется количество
36 call slen ; введенных байтов
37
38 mov edx, eax
39 mov ecx, msg2
40 mov ebx, esi
41 mov eax, 4
42 int 80h
43
44 mov edx, eax
45 mov ecx, contents
46 mov ebx, esi
47 mov eax, 4
48 int 80h
49
50 mov ebx, esi
51 mov eax, 6
52 int 80h

```

Рис. 5.1: Ввод текста программы в файл

Создаю исполняемый файл и запускаю его. Проверяю наличие созданного при исполнении программы файла и вижу, что он действительно создан и находится в текущем каталоге. После этого проверяю содержимое файла при помощи команды `cat`. Программа сработала корректно. (рис. 5.2).

```

[vaosina@fedora lab10]$ nasm -f elf lab10-2.asm
[vaosina@fedora lab10]$ ld -m elf_i386 -o lab10-2 lab10-2.o
[vaosina@fedora lab10]$ ./lab10-2
Как Вас зовут? Вика
[vaosina@fedora lab10]$ ls
in_out.asm  lab10-1.asm  lab10-1.o  lab10-2.asm  name.txt      readme-2.txt
lab10-1     lab10-1.lst  lab10-2   lab10-2.o   readme-1.txt
[vaosina@fedora lab10]$ cat name.txt
Меня зовут Вика

```

Рис. 5.2: Выполнение программы, проверка наличия файла и его содержимого

Текст программы:

```

#include 'in_out.asm'

SECTION .data

```



```

filename db 'name.txt', 0h ; Имя файла
msg1 db 'Как Вас зовут? ', 0h ; Сообщение
msg2 db 'Меня зовут ', 0h
SECTION .bss
contents resb 255 ; переменная для вводимой строки
SECTION .text

global _start
_start:

mov eax,msg1
call sprint

mov ecx, contents
mov edx, 255
call sread

mov ecx, 0777o
mov ebx, filename
mov eax, 8
int 80h

mov ecx, 2
mov ebx, filename
mov eax, 5
int 80h

mov esi, eax

```

```
mov eax, contents
call slen
```

```
mov eax, msg2
call slen
```

```
mov edx, eax
mov ecx, msg2
mov ebx, esi
mov eax, 4
int 80h
```

```
mov edx, eax
mov ecx, contents
mov ebx, esi
mov eax, 4
int 80h
```

```
mov ebx, esi
mov eax, 6
int 80h
call quit
```

6 Выводы

При выполнении данной лабораторной работы я приобрела навыки написания программ для работы с файлами.

7 Список литературы

1. Архитектура ЭВМ