

Лабораторная работа №7

дисциплина: Архитектура компьютера

Серёгина Ирина Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Ответы на вопросы	17
5	Задания для самостоятельной работы	18
6	Выводы	20

Список иллюстраций

3.1	создаю папку и файл	7
3.2	копирую in_out.asm	7
3.3	программа вывода значения регистра eax	8
3.4	запуск исполняемого файла	9
3.5	изменение содержимого файла	9
3.6	запуск исполняемого файла	9
3.7	создание файла lab7-2.asm	10
3.8	файл lab7-2.asm	10
3.9	запуск исполняемого файла	11
3.10	изменяю файл lab7-2.asm	11
3.11	запуск исполняемого файла	11
3.12	изменения в файле lab7-2.asm	12
3.13	запуск файла lab7-2.asm	12
3.14	создание файла lab7-3.asm	12
3.15	изменения в файле lab7-3.asm	13
3.16	запуск файла lab7-3.asm	13
3.17	изменения в файле lab7-3.asm	14
3.18	запуск lab7-3.asm	14
3.19	создание variant.asm	15
3.20	variant.asm	15
3.21	запуск программы для вычисления варианта	16
5.1	программа для вычисления выражения	18
5.2	запускаю программу	19

Список таблиц

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Задания для самостоятельной работы

3 Выполнение лабораторной работы

1. Открываю терминал, перехожу в каталог курса и создаю папку lab07, в ней создаю файл lab7-1.asm (рис. 3.1).

```
[irina@fedora ~]$ mkdir ~/work/study/2022-2023/Архитектура\ компьютера/study_2022-2023_arh-pc/lab07
[irina@fedora ~]$ cd ~/work/study/2022-2023/Архитектура\ компьютера/study_2022-2023_arh-pc/lab07
[irina@fedora lab07]$ touch lab7-1.asm
[irina@fedora lab07]$ ls
lab7-1.asm
[irina@fedora lab07]$
```

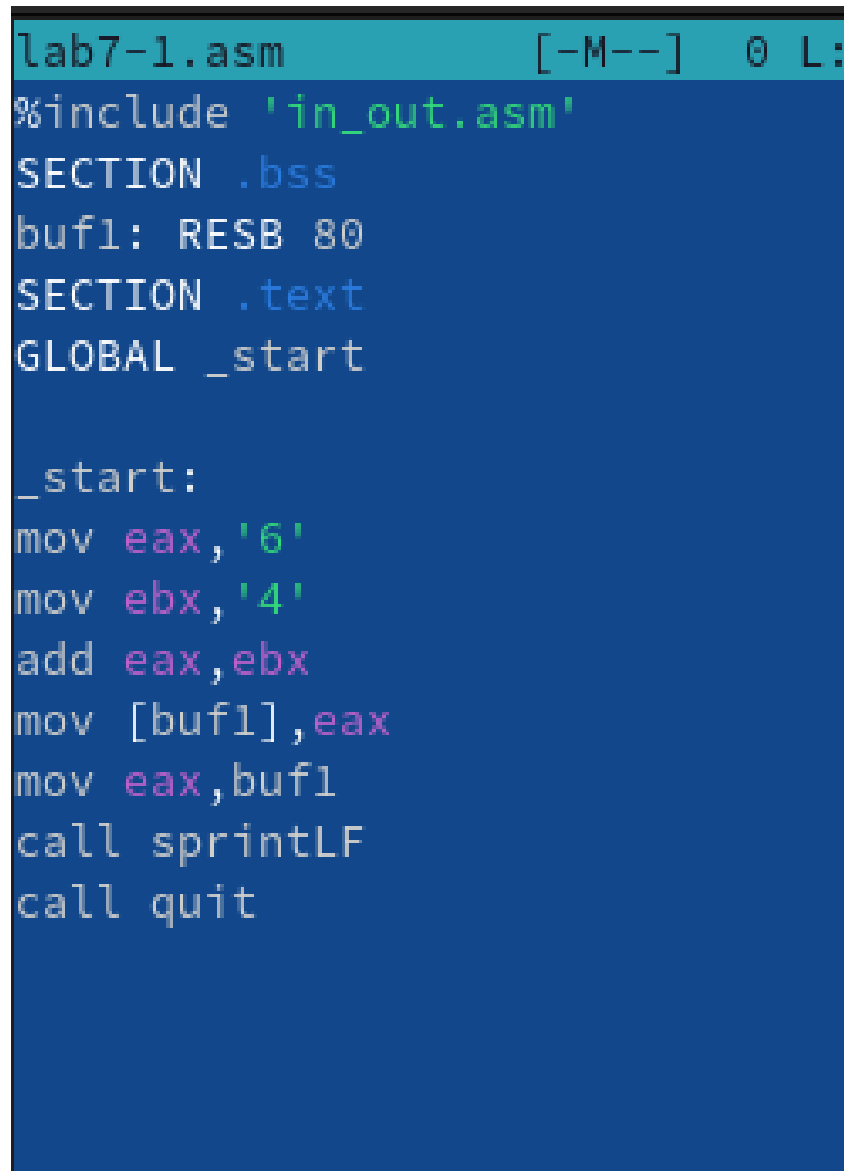
Рис. 3.1: создаю папку и файл

2. Копирую в каталог файл in_out.asm (рис. 3.2).

```
[irina@fedora lab07]$ cp ~/Загрузки/in_out.asm in_out.asm
[irina@fedora lab07]$ ls
in_out.asm  lab7-1.asm
[irina@fedora lab07]$
```

Рис. 3.2: копирую in_out.asm

3. Открываю файл lab7-1.asm и изменяю содержимое файла на программу вывода значения регистра eax (рис. 3.3).



```
lab7-1.asm      [-M--]  0  L:
#include 'in_out.asm'
SECTION .bss
buf1: RESB 80
SECTION .text
GLOBAL _start

_start:
mov  eax,'6'
mov  ebx,'4'
add  eax,ebx
mov  [buf1],eax
mov  eax,buf1
call sprintfLF
call quit
```

Рис. 3.3: программа вывода значения регистра eax

4. Создаю объектный, а потом исполняемый файл, запускаю его. (рис 3.4)
Выводится “j”, потому что в регистр записалась сумма кодов, которая соответствует этой букве (рис. 3.4).


```
[irina@fedora lab07]$ nasm -f elf lab7-1.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[irina@fedora lab07]$ ./lab7-1
j
```

Рис. 3.4: запуск исполняемого файла

5. Меняю содержимое файла, убирая кавычки у цифр 4 и 6 (рис. 3.5).

```
_start:
mov eax,6
mov ebx,4
add eax,ebx
```

Рис. 3.5: изменение содержимого файла

6. Заново создаю объектный и исполняемый файл, запускаю. Код выводимого элемента - 10, это символ переноса строки, поэтому отображаться он не будет (рис. 3.6).

```
[irina@fedora lab07]$ nasm -f elf lab7-1.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[irina@fedora lab07]$ ./lab7-1
```

Рис. 3.6: запуск исполняемого файла

7. Создаю файл lab7-2.asm (рис. 3.7).

```
[irina@fedora lab07]$ touch lab7-2.asm  
[irina@fedora lab07]$
```

Рис. 3.7: создание файла lab7-2.asm

8. Редактирую содержимое файла, вставляя туда программу вывода значения регистра `eax` (рис. 3.8).

```
lab7-2.asm [-M  
%include 'in_out.asm'  
SECTION .text  
GLOBAL _start  
_start:  
mov eax, '6'  
mov ebx, '4'  
add eax, ebx  
call iprintLF  
call quit
```

Рис. 3.8: файл lab7-2.asm

9. Создаю объектный и исполняемый файл, запускаю. Получаю 106, потому что программа выводит получившееся число, а не его символ (рис. 3.9).

```
[irina@fedora lab07]$ nasm -f elf lab7-2.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[irina@fedora lab07]$ ./lab7-2
106
```

Рис. 3.9: запуск исполняемого файла

10. Редактирую файл lab7-2.asm убирая кавычки у чисел 4 и 6 (рис. 3.10).

```
_start:
mov  eax,6
mov  ebx,4
```

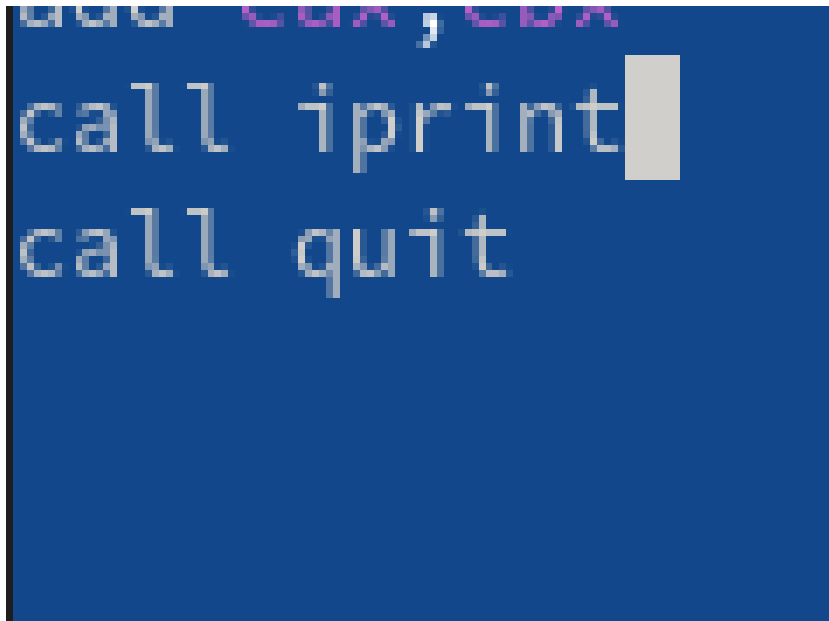
Рис. 3.10: изменяю файл lab7-2.asm

11. Заново создаю объектный и исполняемый файл, запускаю. Получаю 10, потому что в этот раз программа суммирует числа, а не их коды, как в предыдущий раз (рис. 3.11).

```
[irina@fedora lab07]$ nasm -f elf lab7-2.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[irina@fedora lab07]$ ./lab7-2
10
```

Рис. 3.11: запуск исполняемого файла

12. Опять редактирую файл lab7-2.asm, меняя `iprintLF` на `iprint` (рис. 3.12).



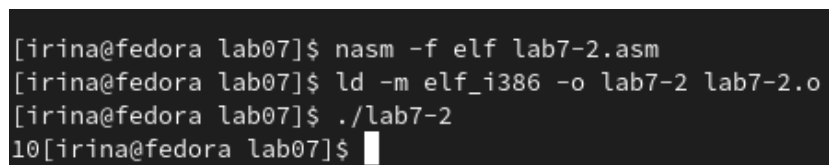
```

mov ecx, ecx
call iprint
call quit

```

Рис. 3.12: изменения в файле lab7-2.asm

13. Создаю новый объектный, затем исполняемый файл, запускаю. На этот раз отсутствует символ переноса строки, поэтому его нет (рис. 3.13).



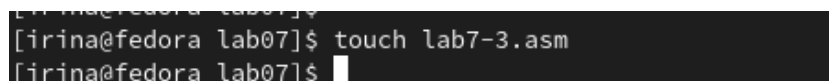
```

[irina@fedora lab07]$ nasm -f elf lab7-2.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[irina@fedora lab07]$ ./lab7-2
10[irina@fedora lab07]$

```

Рис. 3.13: запуск файла lab7-2.asm

14. Создаю файл lab7-3.asm (рис. 3.14).



```

[irina@fedora lab07]$ touch lab7-3.asm
[irina@fedora lab07]$

```

Рис. 3.14: создание файла lab7-3.asm

15. Редактирую его, вставляя туда программу вычисления выражения $f(x) = (5 * 2 + 3)/3$ (рис. 3.15).

```

lab7-3.asm      [-M--]  0 L:[ 1+ 0  1/ 30] *(0  /1366b) 005
;-----
; Программа вычисления выражения
;-----
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,5 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX

add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/3, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 3.15: изменения в файле lab7-3.asm

16. Создаю объектный, исполняемый файл, запуская его (рис. 3.16).

```

[irina@fedora lab07]$ nasm -f elf lab7-3.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[irina@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1

```

Рис. 3.16: запуск файла lab7-3.asm

17. Редактирую файл так, чтобы теперь он вычислял выражение $f(x) = (4 * 6 + 2)/5$ (рис. 3.17).

```

lab7-3.asm      [-M--] 19 L:[ 1+18 19/ 30] *(581 /136
;-----
; Программа вычисления выражения
;-----
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax,4 ; EAX=4
mov ebx,6 ; EBX=6
mul ebx ; EAX=EAX*EBX

add eax,2 ; EAX=EAX+2
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=5
div ebx ; EAX=EAX/5 EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax,div ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
mov eax,rem ; вызов подпрограммы печати
call sprint ; сообщения 'Остаток от деления: '
mov eax,edx ; вызов подпрограммы печати значения
call iprintLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Рис. 3.17: изменения в файле lab7-3.asm

18. Заново создаю объектный и исполняемый файл, запускаю. Всё работает верно (рис. 3.18).

```

[irina@fedora lab07]$ nasm -f elf lab7-3.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[irina@fedora lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1

```

Рис. 3.18: запуск lab7-3.asm

19. Создаю файл variant.asm (рис. 3.19).

```
остаток от деления: 1
[irina@fedora lab07]$ touch variant.asm
[irina@fedora lab07]$ ls
in_out.asm  lab7-1.asm  lab7-2      lab7-2.o  lab7-3.asm  variant.asm
lab7-1      lab7-1.o    lab7-2.asm  lab7-3    lab7-3.o
[irina@fedora lab07]$
```

Рис. 3.19: создание variant.asm

20. В файл вставляю программу для вычисления варианта, в зависимости от номера студенческого билета (рис. 3.20).

```
variant.asm  [-M--] 25 L:[ 1+ 6 7/ 28] *(270 / 617b) 0010 0x00
;-----
; Программа вычисления варианта
;-----
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите No студенческого билета: ',0
rem: DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit
```

Рис. 3.20: variant.asm

21. Создаю объектный файл, исполняемый, запускаю. Мой вариант - 7 (рис. 3.21).

```
[irina@fedora lab07]$ nasm -f elf variant.asm
[irina@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[irina@fedora lab07]$ ./variant
Введите No студенческого билета:
1132227126
Ваш вариант: 7
```

Рис. 3.21: запуск программы для вычисления варианта

4 Ответы на вопросы

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? `mov eax, rem` и `call sprint`
2. Для чего используются следующие инструкции? `push mov ecx, x` `mov edx, 80` `call sread`? первая используется для записи адреса в регистр, вторая - для определения длины вводимой строки, а третья - для привязки программы, позволяющей вводить данные с клавиатуры
3. Для чего используется инструкция “`call atoi`”? для вызова подпрограммы из внешнего файла
4. Какие строки листинга 7.4 отвечают за вычисления варианта? `xor edx,edx` `mov ebx,20` `div ebx` `inc edx`
5. В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”? в `edx`
6. Для чего используется инструкция “`inc edx`”? прибавляет к `edx` единицу
7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычисления? `mov eax,edx` `call iprintLF`

5 Задания для самостоятельной работы

1. Создаю файл lab7-4.asm. В соответствии с номером полученного варианта выбираю выражение, для вычисления которого мне нужно составить программу $(5(x - 1)^2)$, редактирую файл, пишу программу (рис. 5.1).

```
lab7-4.asm      [----] 12 L:[ 10+21 31/ 31] *(912 /
SECTION .text
GLOBAL _start
_start:
; ---- Вычисление выражения
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
sub eax, 1
mul eax
mov ebx, 5
mul ebx
mov edi, eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax, rem ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 5.1: программа для вычисления выражения

2. `sub eax` - я вычитаю из `x` единицу, `mul eax` - возвожу $(x-1)$ в квадрат, `mov ebx` - присваиваю `ebx` значение 5, `mul ebx` - умножаю `eax` на `ebx`.

3. Создаю объектный и исполняемый файл, запускаю, ввожу два различных значения из таблицы. Результаты верны (рис. 5.2).

```
[irina@fedora lab07]$ nasm -f elf lab7-4.asm
[irina@fedora lab07]$ ld -m elf_i386 -o lab7-4 lab7-4.o
[irina@fedora lab07]$ ./lab7-4
Переменная x: 3
Результат: 20
[irina@fedora lab07]$ ./lab7-4
Переменная x: 5
Результат: 80
```

Рис. 5.2: запускаю программу

6 Выводы

Я приобрела практические навыки в работе с арифметическими операциями в NASM.