

Отчет по лабораторной работе №7.

Дисциплина: архитектура компьютера.

Лобанова Полина Иннокентьевна.

Содержание

1	Цель работы	3
2	Выполнение лабораторной работы	4
3	Вопросы	10
4	Выполнение самостоятельной работы	11
5	Выводы	13

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

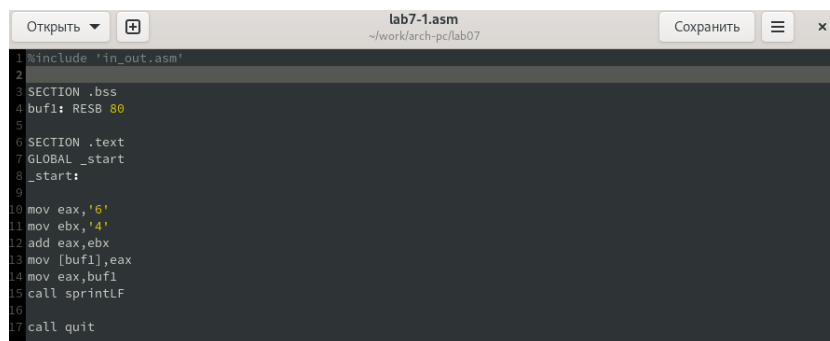
2 Выполнение лабораторной работы

1. Создадим каталог lab07 и файл lab7-1.asm.

```
[pilobanova@i0 ~]$ mkdir ~/work/arch-pc/lab07
[pilobanova@i0 ~]$ cd ~/work/arch-pc/lab07
[pilobanova@i0 lab07]$ touch lab7-1.asm
[pilobanova@i0 lab07]$ ls
lab7-1.asm
[pilobanova@i0 lab07]$
```

Рис. 2.1: Создание каталога lab07 и файла lab7-1.asm.

2. Введем текст программы в файл lab7-1.asm.



```
1 %include 'in_out.asm'
2
3 SECTION .bss
4 buf1: RESB 80
5
6 SECTION .text
7 GLOBAL _start
8 _start:
9
10 mov eax, '6'
11 mov ebx, '4'
12 add eax, ebx
13 mov [buf1], eax
14 mov eax, buf1
15 call sprintf
16
17 call quit
```

Рис. 2.2: Текст программы в файле lab7-1.asm.

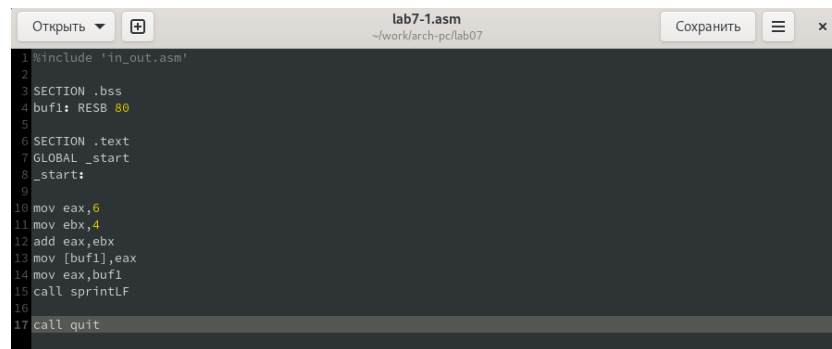
3. Создадим исполняемый файл и запустим программу.

```
[pilobanova@i0 lab07]$ nasm -f elf lab7-1.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[pilobanova@i0 lab07]$ ./lab7-1
j
```

Рис. 2.3: Создание и запуск программы.

В результате получим символ j.

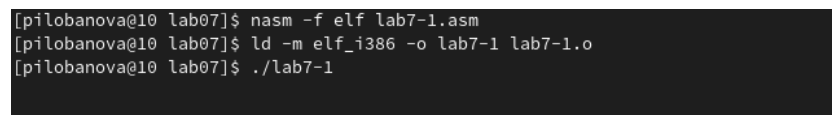
4. Изменим текст программы и вместо символов запишем в регистры числа.



```
1 %include 'in_out.asm'
2
3 SECTION .bss
4 buf1: RESB 80
5
6 SECTION .text
7 GLOBAL _start
8 _start:
9
10 mov eax,6
11 mov ebx,4
12 add eax,ebx
13 mov [buf1],eax
14 mov eax,buf1
15 call printf
16
17 call quit
```

Рис. 2.4: Измененный текст программы.

5. Создадим исполняемый файл и запустим программу.

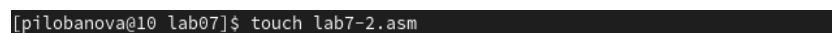


```
[pilobanova@i0 lab07]$ nasm -f elf lab7-1.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[pilobanova@i0 lab07]$ ./lab7-1
```

Рис. 2.5: Создание и запуск программы.

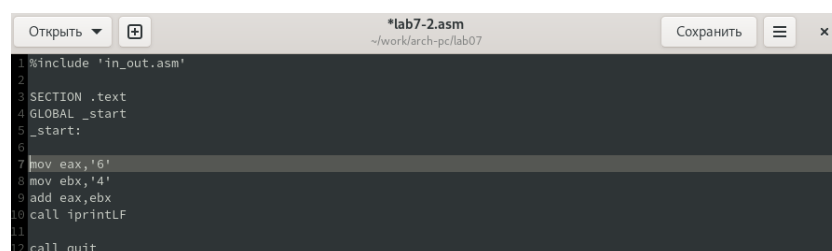
В результате получаем пустоту, как и должно быть.

6. Создадим файл lab7-2.asm и заполним его.



```
[pilobanova@i0 lab07]$ touch lab7-2.asm
```

Рис. 2.6: Создание файла lab7-2.asm.



```
1 %include 'in_out.asm'
2
3 SECTION .text
4 GLOBAL _start
5 _start:
6
7 mov eax,'6'
8 mov ebx,'4'
9 add eax,ebx
10 call iprintLF
11
12 call quit
```

Рис. 2.7: Текст программы в файле lab7-2.asm.

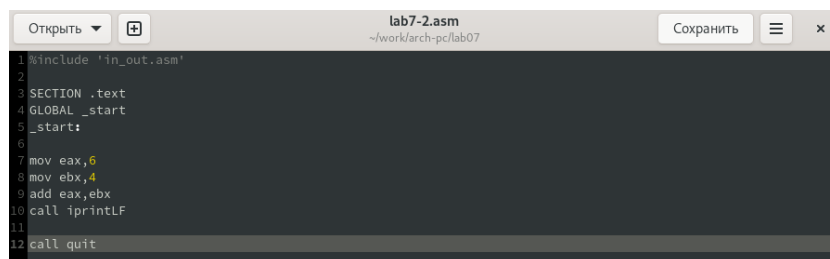
7. Создадим исполняемый файл и запустим программу.

```
[pilobanova@i0 lab07]$ nasm -f elf lab7-2.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[pilobanova@i0 lab07]$ ./lab7-2
106
```

Рис. 2.8: Создание и запуск программы.

В результате работы программы получим число 106.

8. Отредактируем текст программы в файле lab7-2.asm, заменив символы на числа.



```
1 %include 'in_out.asm'
2
3 SECTION .text
4 GLOBAL _start
5 _start:
6
7 mov eax,6
8 mov ebx,4
9 add eax,ebx
10 call iprintLF
11
12 call quit
```

Рис. 2.9: Измененный текст программы в файле lab7-2.asm.

9. Создадим исполняемый файл и запустим программу.

```
[pilobanova@i0 lab07]$ nasm -f elf lab7-2.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[pilobanova@i0 lab07]$ ./lab7-2
10
```

Рис. 2.10: Создание и запуск программы.

Результатом является 10.

10. Заменим функцию iprintLF на iprint в файле lab7-2.asm.



```
1 %include 'in_out.asm'
2
3 SECTION .text
4 GLOBAL _start
5 _start:
6
7 mov eax,6
8 mov ebx,4
9 add eax,ebx
10 call iprint
11
12 call quit
```

Рис. 2.11: Измененный текст программы.

11. Создадим исполняемый файл и запустим программу.

```
[pilobanova@i10 lab07]$ nasm -f elf lab7-2.asm
[pilobanova@i10 lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[pilobanova@i10 lab07]$ ./lab7-2
i10[pilobanova@i10 lab07]$
```

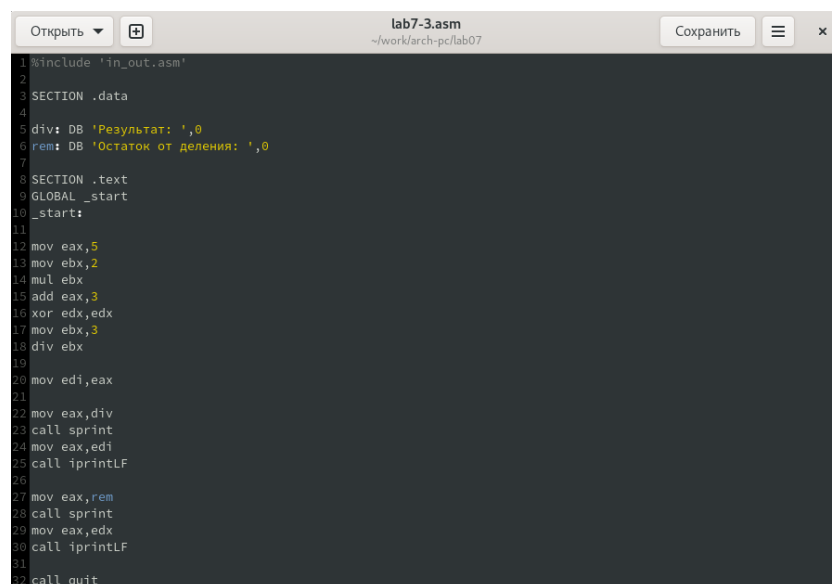
Рис. 2.12: Создание и запуск программы.

Вывод функции `iprintLF` отличается от функции `iprint` тем, что при использовании команды `iprintLF` мы начинаем вводить следующую команду на следующей строке, а при использовании команды `iprint` мы вводим на той же строке.

12. Создадим файл `lab7-3.asm` и заполним его.

```
[pilobanova@i10 lab07]$ touch lab7-3.asm
```

Рис. 2.13: Создание файла `lab7-3.asm`.



```
1 %include 'in_out.asm'
2
3 SECTION .data
4
5 div: DB 'Результат: ',0
6 rem: DB 'Остаток от деления: ',0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 mov eax,5
13 mov ebx,2
14 mul ebx
15 add eax,3
16 xor edx,edx
17 mov ebx,3
18 div ebx
19
20 mov edi,eax
21
22 mov eax,div
23 call sprint
24 mov eax,edi
25 call iprintLF
26
27 mov eax,rem
28 call sprint
29 mov eax,edx
30 call iprintLF
31
32 call quit
```

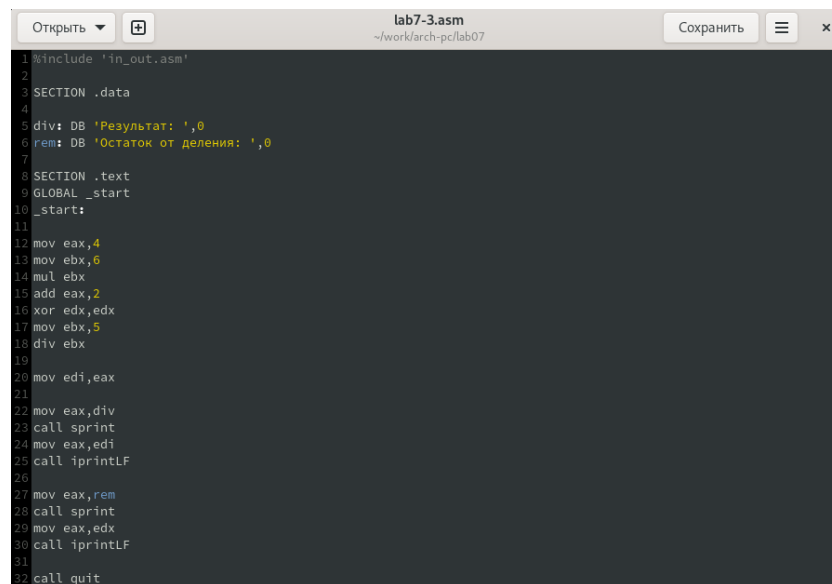
Рис. 2.14: Текст программы в файле `lab7-3.asm`.

13. Создадим исполняемый файл и запустим программу.

```
[pilobanova@i10 lab07]$ nasm -f elf lab7-3.asm
[pilobanova@i10 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[pilobanova@i10 lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
```

Рис. 2.15: Создание и запуск программы.

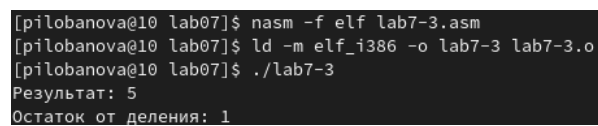
14. Изменим текст программы для вычисления выражения $f(x)=(4*6+2)/5$.



```
1 %include 'in_out.asm'
2
3 SECTION .data
4
5 div: DB 'Результат: ',0
6 rem: DB 'Остаток от деления: ',0
7
8 SECTION .text
9 GLOBAL _start
10 _start:
11
12 mov eax,4
13 mov ebx,6
14 mul ebx
15 add eax,2
16 xor edx,edx
17 mov ebx,5
18 div ebx
19
20 mov edi,eax
21
22 mov eax,div
23 call sprint
24 mov eax,edi
25 call iprintLF
26
27 mov eax,rem
28 call sprint
29 mov eax,edx
30 call iprintLF
31
32 call quit
```

Рис. 2.16: Измененный текст программы в файле *lab7-3.asm*.

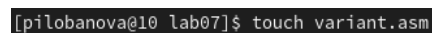
15. Создадим исполняемый файл и запустим программу.



```
[pilobanova@i0 lab07]$ nasm -f elf lab7-3.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[pilobanova@i0 lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1
```

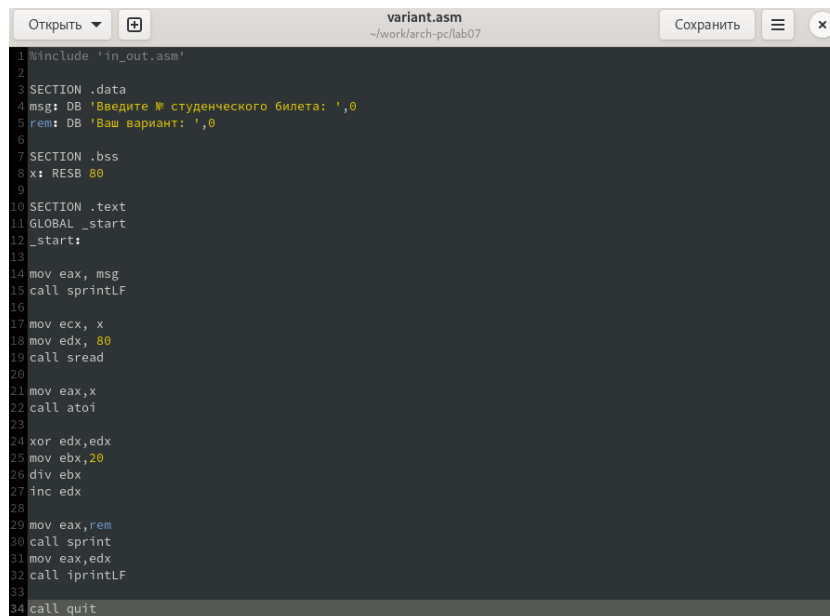
Рис. 2.17: Создание и запуск программы.

17. Создадим файл *variant.asm* и заполним его.



```
[pilobanova@i0 lab07]$ touch variant.asm
```

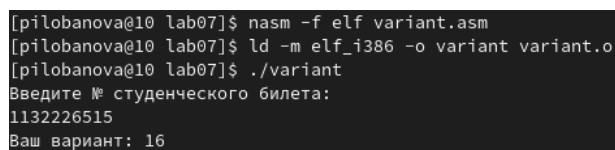
Рис. 2.18: Создание файла *variant.asm*.



```
1 %include 'in_out.asm'
2
3 SECTION .data
4 msg: DB 'Введите № студенческого билета: ',0
5 rem: DB 'Ваш вариант: ',0
6
7 SECTION .bss
8 x: RESB 80
9
10 SECTION .text
11 GLOBAL _start
12 _start:
13
14 mov eax, msg
15 call sprintf
16
17 mov ecx, x
18 mov edx, 80
19 call sread
20
21 mov eax, x
22 call atoi
23
24 xor edx, edx
25 mov ebx, 20
26 div ebx
27 inc edx
28
29 mov eax, rem
30 call sprintf
31 mov eax, edx
32 call iprintLF
33
34 call quit
```

Рис. 2.19: Текст программы в файле *variant.asm*.

18. Создадим исполняемый файл и запустим программу, введем номера студенческого билета для определения варианта.



```
[pilobanova@i0 lab07]$ nasm -f elf variant.asm
[pilobanova@i0 lab07]$ ld -m elf_i386 -o variant variant.o
[pilobanova@i0 lab07]$ ./variant
Введите № студенческого билета:
1132226515
Ваш вариант: 16
```

Рис. 2.20: Создание и запуск программы.

3 Вопросы

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’?

```
mov eax,rem call sprint
```

2. Для чего используются следующие инструкции? `mov ecx, x` `mov edx, 80` `call sread`

Для ввода сообщения с клавиатуры.

3. Для чего используется инструкция “`call atoi`”?

Для преобразования ASCII-кодф в целое число.

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`”?

Остаток записывается в регистр `ebx`.

6. Для чего используется инструкция “`inc edx`”?

Для увеличения значения регистра `edx` на 1.

7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычисления?

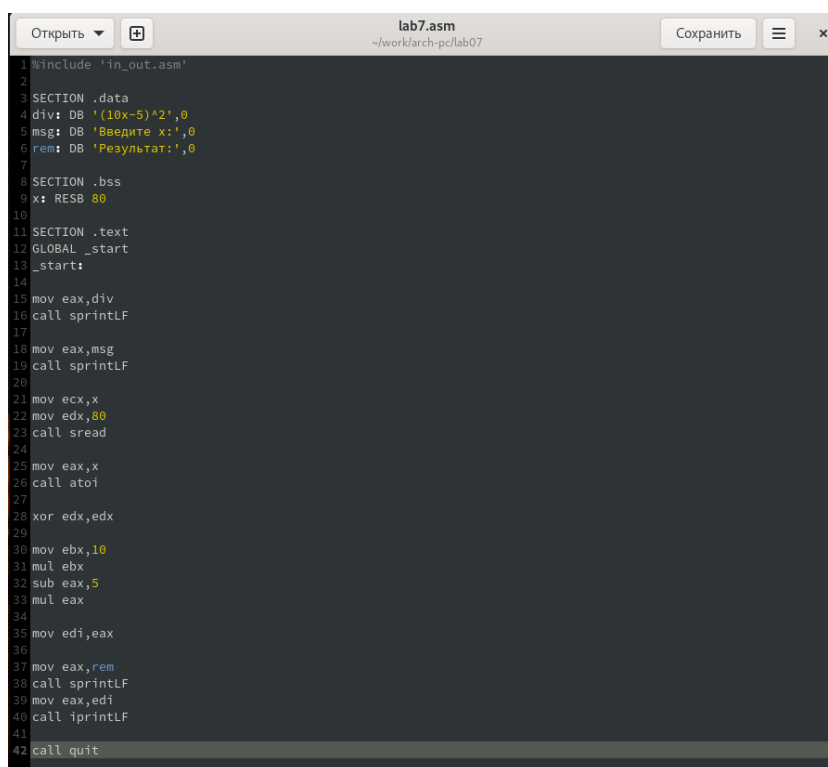
```
mov eax,rem call sprint mov eax,edx call iprintLF
```

4 Выполнение самостоятельной работы

1. Создадим файл lab7.asm и заполним его так, чтобы программа вычисляла функцию в соответствии с номером варианта. Поскольку в задании 7 лабораторной работы мне выпал вариант 16, необходимая функция - $(10x-5)^2$.

```
[pilobanova@10 lab07]$ touch lab7.asm
```

Рис. 4.1: Создание файла lab7.asm.



```
1 %include 'in_out.asm'
2
3 SECTION .data
4 div: DB '(10x-5)^2',0
5 msg: DB 'Введите x:',0
6 rem: DB 'Результат:',0
7
8 SECTION .bss
9 x: RESB 80
10
11 SECTION .text
12 GLOBAL _start
13 _start:
14
15 mov eax,div
16 call sprintf
17
18 mov eax,msg
19 call sprintf
20
21 mov ecx,x
22 mov edx,80
23 call sread
24
25 mov eax,x
26 call atoi
27
28 xor edx,edx
29
30 mov ebx,10
31 mul ebx
32 sub eax,5
33 mul eax
34
35 mov edi,eax
36
37 mov eax,rem
38 call sprintf
39 mov eax,edi
40 call iprintf
41
42 call quit
```

Рис. 4.2: Текст программы в файле lab7.asm.

2. Создадим исполняемый файл и запустим его.

```
[pilobanova@10 lab07]$ nasm -f elf lab7.asm
[pilobanova@10 lab07]$ ld -m elf_i386 -o lab7 lab7.o
[pilobanova@10 lab07]$ ./lab7
(10x-5)^2
Введите x:
3
Результат:
625
[pilobanova@10 lab07]$ ./lab7
(10x-5)^2
Введите x:
1
Результат:
25
[pilobanova@10 lab07]$
```

Рис. 4.3: Создание и запуск исполняемого файла.

5 Выводы

Я научилась работать с арифметическими инструкциями языка ассемблера NASM.