

Отчёт по лабораторной работе №6

Дисциплина: Архитектура компьютера

Кузьмина Мария Константиновна

Содержание

1	Цель работы	4
2	Задание	5
3	Выполнение лабораторной работы	6
3.1	Символьные и численные данные в NASM	6
3.2	Выполнение арифметических операций в NASM	10
3.3	Выполнение заданий для самостоятельной работы	15
4	Выводы	18

Список иллюстраций

3.1	снимок экрана	6
3.2	снимок экрана	7
3.3	снимок экрана	7
3.4	снимок экрана	8
3.5	снимок экрана	8
3.6	снимок экрана	8
3.7	снимок экрана	9
3.8	снимок экрана	9
3.9	снимок экрана	9
3.10	снимок экрана	10
3.11	снимок экрана	10
3.12	снимок экрана	10
3.13	снимок экрана	11
3.14	снимок экрана	11
3.15	снимок экрана	11
3.16	снимок экрана	12
3.17	снимок экрана	13
3.18	снимок экрана	13
3.19	снимок экрана	14
3.20	снимок экрана	16
3.21	снимок экрана	17

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. Выполнение заданий для самостоятельной работы.

3 Выполнение лабораторной работы

3.1 Символьные и численные данные в NASM

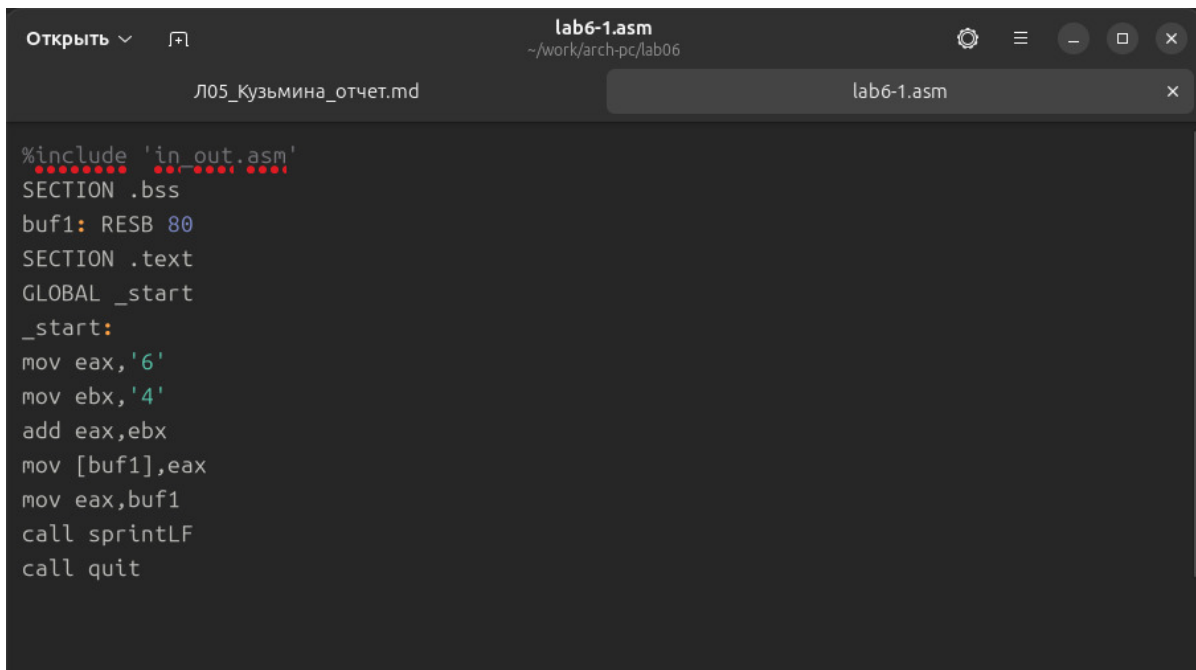
С помощью `mkdir` создаем директорию для создания файлов лабораторной работы, переходим в созданный каталог(рис. 3.1):



```
mkkuzjmina@VirtualBox:~$ mkdir ~/work/arch-pc/lab06
mkkuzjmina@VirtualBox:~$ cd ~/work/arch-pc/lab06
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ touch lab6-1.asm
```

Рис. 3.1: снимок экрана

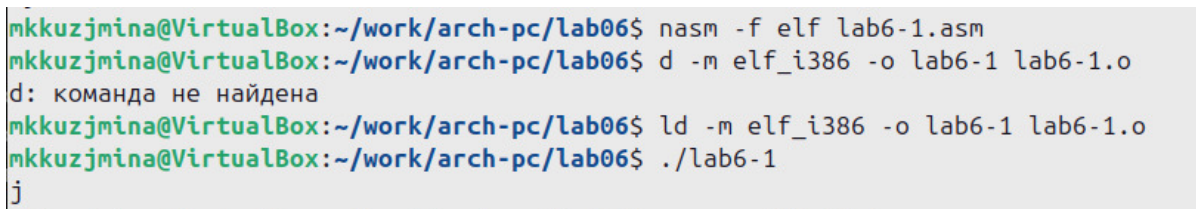
С помощью утилиты `touch` создаем файл `lab6-1.asm`, копируем в текущий каталог файл `in_out.asm` и открываем созданный файл; вставляем в него программу вывода значения регистра `eax` (рис. 3.2):

A screenshot of a text editor window titled 'lab6-1.asm' with a path '~/.work/arch-pc/lab06'. The editor contains assembly code for a program. The code includes a header file 'in_out.asm', defines a buffer 'buf1' of size 80 in the .bss section, and places the main logic in the .text section. The logic moves the character '6' into 'eax', '4' into 'ebx', adds them, and prints the result using 'sprintf' before calling 'quit'.

```
%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 sprintf
call quit
```

Рис. 3.2: снимок экрана

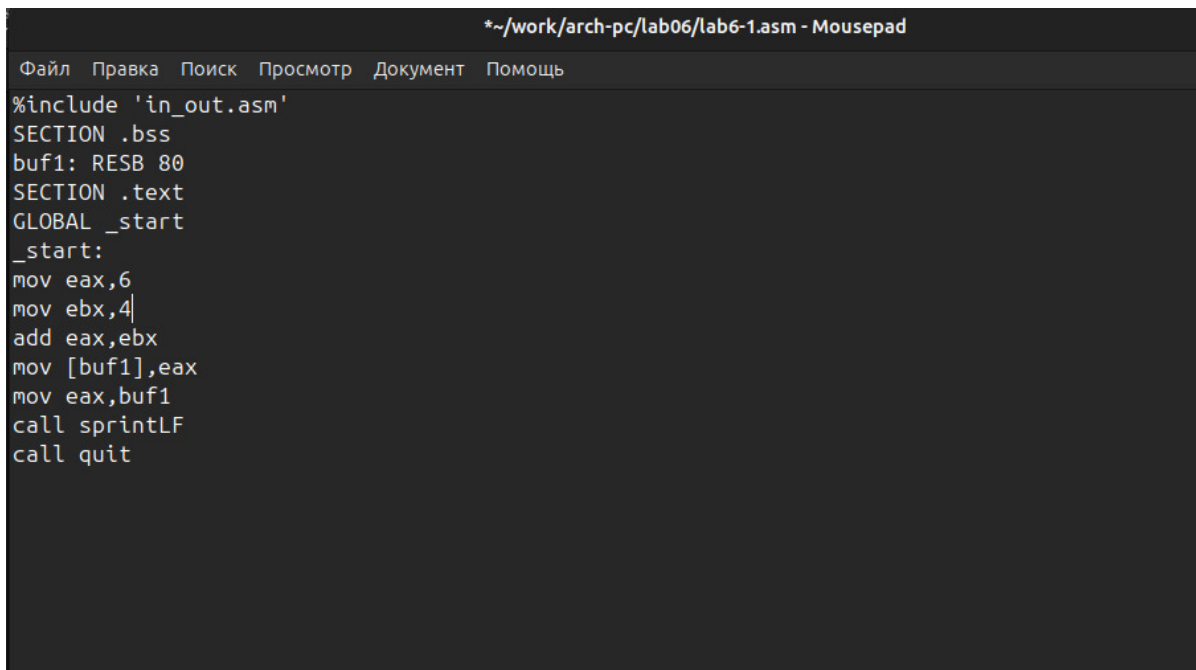
Сохраняем исполняемый файл программы и запускаем его (рис. 3.3):

A screenshot of a terminal window showing the compilation and execution of the assembly program. The user runs 'nasm -f elf lab6-1.asm' to create 'lab6-1.o', then 'ld -m elf_i386 -o lab6-1 lab6-1.o' to create the executable 'lab6-1', and finally './lab6-1' to run it. The output shows the sum of 6 and 4, which is 10.

```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
d: команда не найдена
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-1
10
j
```

Рис. 3.3: снимок экрана

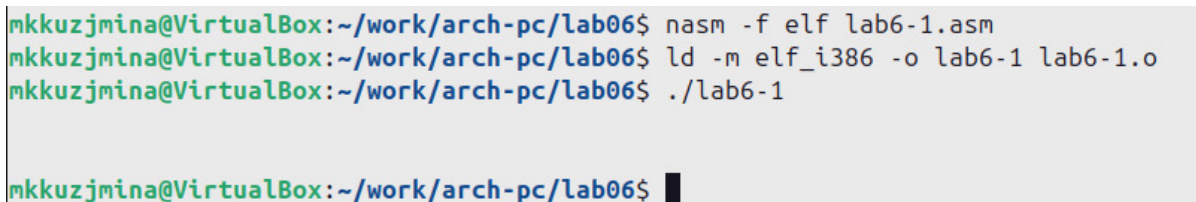
Изменяем в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. 3.4):



```
*~/work/arch-pc/lab06/lab6-1.asm - Mousepad
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%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 sprintf
call quit
```

Рис. 3.4: снимок экрана

Создаем новый исполняемый файл программы и запускаем его. При исполнении программы мы не получаем число 10. В данном случае выводится символ с кодом 10. Пользуясь таблицей ASCII определяем, что 10 соответствует символу перевода строки. (рис. 3.5):

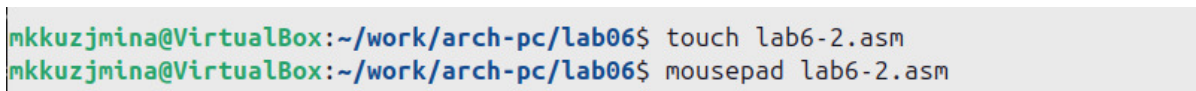


```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-1

mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.5: снимок экрана

С помощью touch создаем новый файл lab6-2.asm, открываем mousepad и вводим текст программы для вывода значения регистра eax (рис. 3.6):



```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ touch lab6-2.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ mousepad lab6-2.asm
```

Рис. 3.6: снимок экрана

(рис. 3.7):

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

Рис. 3.7: снимок экрана

Создаем и запускаем исполняемый файл lab6-2. На выходе получаем число 106, потому что программа позволяет вывести число. (рис. 3.8):

```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-2
106
```

Рис. 3.8: снимок экрана

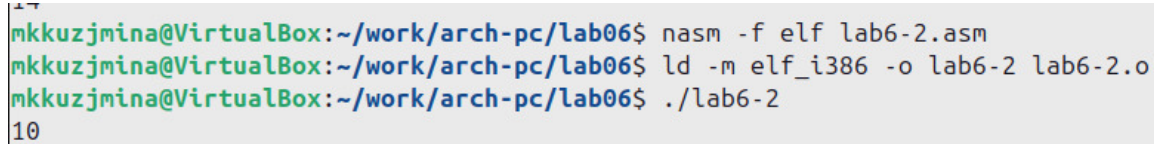
Заменяем в тексте программы файла lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. 3.9):

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

Рис. 3.9: снимок экрана

Создаем и запускаем новый исполняемый файл. Программа складывает непосредственно сами числа, поэтому вывод 10.

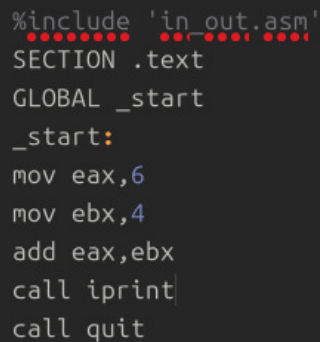
(рис. 3.10):



```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-2
10
```

Рис. 3.10: снимок экрана

Заменяем в тексте программы функцию `iprintLF` на `iprint` (рис. 3.11):



```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprint
call quit
```

Рис. 3.11: снимок экрана

Создаем и запускаем новый исполняемый файл. Вывод изменился, потому что `iprintLF` добавил перенос строки, а `iprint` нет. (рис. 3.12):



```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-2
10mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$
```

Рис. 3.12: снимок экрана

3.2 Выполнение арифметических операций в NASM

Создаем файл `lab6-3.asm` с помощью `touch` (рис. 3.13):

```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ touch lab6-3.asm
```

Рис. 3.13: снимок экрана

Вводим в созданный файл текст программы для вычисления значения данного выражения (рис. 3.14):

```
%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
```

Рис. 3.14: снимок экрана

Создаем и запускаем файл (рис. 3.15):

```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ touch lab6-3.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
```

Рис. 3.15: снимок экрана

Изменяем текст, чтобы программа вычисляла значение нового выражения $(4 * 6 + 2)/5$ (рис. 3.16):

```

; ---- 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'

```

Рис. 3.16: снимок экрана

```

#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

```

Создаем и запускаем файл (рис. 3.17):

```

mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-3 lab6-3.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис. 3.17: снимок экрана

Создаем файл variant.asm, вводим в файл текст программы для вычисления варианта по номеру студенческого билета(рис. 3.18):

```

;-----
; Программа вычисления варианта
;-----
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread
mov eax,x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, `eax=x`

```

Рис. 3.18: снимок экрана

Создаем и запускаем исполняемый файл. Вводим номер своего студенческого билета. (рис. 3.19):

```
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf variant.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132246734
Ваш вариант: 15
```

Рис. 3.19: снимок экрана

1. Какие строки листинга 6.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’? Ответ:

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

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

Ответ: `mov ecx, x` — загружает адрес буфера, куда будет сохранён ввод. `mov edx, 80` — указывает максимальное количество символов для ввода (80). `call sread` — вызывает функцию для чтения строки с клавиатуры.

3. Для чего используется инструкция “`call atoi`”? Ответ: преобразует введённую строку в целое число.

4. Какие строки листинга 6.4 отвечают за вычисления варианта? Ответ:

```
xor edx,edx
mov ebx,20
div ebx
inc edx
```

5. В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”? Ответ: остаток от деления записывается в регистр `edx`

6. Для чего используется инструкция “inc edx”? Ответ: увеличивает значение, находящееся в регистре edx, на 1.
7. Какие строки листинга 6.4 отвечают за вывод на экран результата вычисления? Ответ:

```
mov eax,edx  
call iprintLF
```

3.3 Выполнение заданий для самостоятельной работы

Создаю и открываем файл lab6-4.asm , вводим в него текст программы для вычисления значения $(5 + x)^2 - 3$, (вариант 15) (рис. 3.20):

```

%include 'in_out.asm'
SECTION .data
msg: DB 'Введите значение x: ',0
rem: DB 'Результат: ',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`
add eax, 5
mov ebx, eax
mul ebx
add eax, -3
mov edi, eax
mov eax, rem
call sprintf
mov eax, edi
call iprintf
call quit

```

Рис. 3.20: снимок экрана

```

%include 'in_out.asm'
SECTION .data
msg: DB 'Введите значение x: ',0
rem: DB 'Результат: ',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`
add eax,5
mov ebx,eax
mul ebx
add eax,-3
mov edi,eax
mov eax,rem
call sprint
mov eax,edi
call iprintLF
call quit

```

Создаем и запускаем исполняемый файл, после вводим значения x1, x2 (рис. 3.21):



```

mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-4
Введите значение x:
5
Результат: 97
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ ./lab6-4
Введите значение x:
1
Результат: 33
mkkuzjmina@VirtualBox:~/work/arch-pc/lab06$ █

```

Рис. 3.21: снимок экрана

4 Выводы

При выполнении лабораторной работы были освоены арифметические инструкции языка ассемблера NASM.