

# **Отчет по лабораторной работа №7**

**Группа НПИбд-02-22**

**Стариков Данила Андреевич**

# Содержание

<b>1</b>	<b>Цель работы</b>	<b>3</b>
<b>2</b>	<b>Основная часть</b>	<b>4</b>
2.1	Выполнение лабораторной работы . . . . .	4
2.2	Выполнение заданий для самостоятельной работы. . . . .	9
<b>3</b>	<b>Выводы</b>	<b>15</b>

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

## 2 Основная часть

### 2.1 Выполнение лабораторной работы

Для выполнения лабораторной работы создали каталог ~/work/study/arch-rc/lab07. В нем создали файл lab7-1.asm и ввели текст из Листинга 2.1. Также положили в каталог файл in\_out.asm, использованный в лабораторной работе №6.

---

**Листинг 2.1** Программа для вывода значения регистра eax

---

```
%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
```

---

Создали исполняемый файл и запустили его (Рис. 2.1). На выводе получили символ 'j', так как в двоичном представлении код символа '6' - 00110110 (54 в

десятичном), а код символа '4' - 00110100 (52). Команда `add eax, ebx` записала в регистр `eax` сумму кодов - 01101010 (106), что соответствует символу 'j' в ASCII.

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

Рис. 2.1: Результат запуска файла `lab7-1`.

Заменяли текст программы и вместо символов записали в регистры числа:

```
mov eax, '6'
```

```
mov ebx, '4'
```

на строки

```
mov eax, 6
```

```
mov ebx, 4
```

Создали исполняемый файл и запустили (Рис. 2.2). В результате выполнения на экран выводится число с кодом 10, что соответствует символу переноса строки.

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

[dastarikov@fedora lab07]$ |
```

Рис. 2.2: Результат запуска файла `lab7-1` после изменения текста программы.

Чтобы напечатать числа вместо ASCII символов изменили текст программы, создали файл `lab7-2.asm`, воспользовавшись подпрограммами из файла `in_out.asm` (Листинг 2.2).

---

## Листинг 2.2 Программа вывода значения регистра eax

---

```
%include 'in_out.asm'
```

```
SECTION .bss
```

```
    buf1: RESB 80
```

```
SECTION .text
```

```
GLOBAL _start
```

```
    _start:
```

```
    mov eax, '6'
```

```
    mov ebx, '4'
```

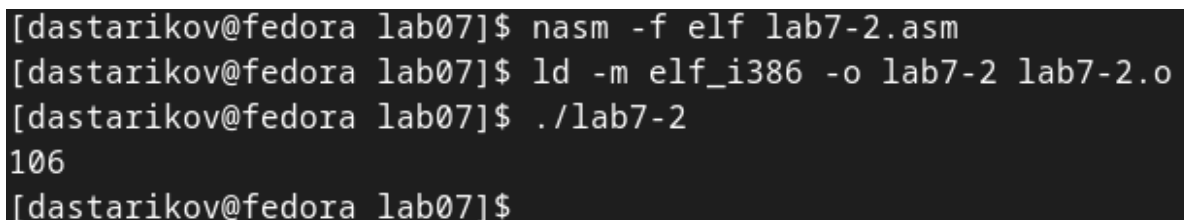
```
    add eax, ebx
```

```
    call iprintLF
```

```
    call quit
```

---

Создали исполняемый файл и запустили (Рис. 2.3). В результате выполнения на экран выводится число 106, так как в регистры были записаны символы '6' и '4'. Подпрограмма `iprintLF` позволила вывести число, а не символ, кодом которого является это число.



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

Рис. 2.3: Результат запуска файла lab7-2

Заменяли текст программы и вместо символов записали в регистры числа:

```
mov eax, '6'
```

```
mov ebx, '4'
```

на строки

```
mov eax, 6
```

```
mov ebx, 4
```

Создали исполняемый файл и запустили (Рис. 2.4). В результате выполнения на экран выводится число 10.

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

Рис. 2.4: Результат запуска файла lab7-2 после изменения текста программы.

Функция `iprint`, в отличие от `iprintLF`, не печатает символ переноса строки (Рис. 2.5).

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

Рис. 2.5: Результат запуска файла lab7-2 с функцией `iprint`

Для примера выполнения арифметических операций в NASM создали программу вычисления арифметического выражения  $f(x) = (5 * 2 + 3) / 3$ . Создали файл lab7-3.asm и ввели текст из Листинга 2.3.

Создали исполняемый файл и запустили (Рис. 2.6). Заметим изменили программу в соответствии с Листингом 2.4, чтобы она вычисляла выражение  $f(x) = (4 * 6 + 2) / 5$  (Рис. 2.7).

```
[dastarikov@fedora lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
[dastarikov@fedora lab07]$
```

Рис. 2.6: Результат запуска файла lab7-2 после изменения текста программы.

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

Рис. 2.7: Результат запуска файла lab7-2 после изменения текста программы.

В качестве другого примера рассмотрели задачу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

- вывести запрос на введение № студенческого билета
- вычислить номер варианта по формуле:  $(S_n \bmod 20) + 1$ , где  $S_n$  – номер студенческого билета (В данном случае  $a \bmod b$  – это остаток от деления  $a$  на  $b$ ).
- вывести на экран номер варианта.

Номер студенческого билета, т. е. число, надо которым будут проводиться арифметические вычисления, вводится с клавиатуры, необходимо преобразовать введенные символы в число, так как ввод с клавиатуры осуществляется в символьном виде. В файле in\_out.asm для этого есть функция atoi.

Создали файл variant.asm и ввели текст из Листинга 2.5. При выполнении программы ввели номер студенческого билета (1132226531) и получили Вариант 12 (Рис. 2.8), что соответствует подсчетам в уме.

```
[dastarikov@fedora lab07]$ touch variant.asm
[dastarikov@fedora lab07]$ nasm -f elf variant.asm
[dastarikov@fedora lab07]$ ld -m elf_i386 -o variant variant.o
[dastarikov@fedora lab07]$ ./variant
Введите № студенческого билета:
1132226531
Ваш вариант: 12
[dastarikov@fedora lab07]$
```

Рис. 2.8: Результат запуска файла lab7-2 после изменения текста программы.



Ответы на вопросы:

1. За вывод сообщения “Ваш вариант:” отвечают строки:

```
mov eax,rem
```

```
call sprint
```

2. Эти строки используются для чтения ввода с клавиатуры. В регистр ECX передает адрес буфера x, а в EDX - длина вводимой строки.
3. Инструкция “call atoi” используется для вызова подпрограммы atoi, преобразующей символы в числа.
4. За вычисление варианты отвечают строки:

```
xor edx,edx
```

```
mov ebx,20
```

```
div ebx
```

```
inc edx
```

5. Остаток от деления записывается в регистр EDX.
6. Инструкция увеличивает значение регистра EDX на 1.
7. За вывод на экран результата вычислений отвечают строки:

```
mov eax,edx
```

```
call iprintLF
```

## 2.2 Выполнение заданий для самостоятельной работы.

Варианту 12 соответствует выражение  $f(x) = (8x - 6)/2$ . Для проверки корректности работы программы ввели 2 числа  $x_1 = 1, x_2 = 5$ . Программа lab7-var12 принимает ввод с клавиатура в качестве значения x и выводит результата вычисления выражения  $f(x) = (8x - 6)/2$ , под делением принимается целочисленное деление с отбрасываем остатка (Рис. 2.9). Текст файла lab7-var12.asm соответствует Листингу 2.6.

```
[dastarikov@fedora lab07]$ nasm -f elf lab7-var12.asm
[dastarikov@fedora lab07]$ ld -m elf_i386 -o lab7-var12 lab7-var12.o
[dastarikov@fedora lab07]$ ./lab7-var12
Выражение  $f(x) = (8 \cdot x - 6) / 2$ 
Введите значение x:1
Результат: 1
[dastarikov@fedora lab07]$ ./lab7-var12
Выражение  $f(x) = (8 \cdot x - 6) / 2$ 
Введите значение x:5
Результат: 17
[dastarikov@fedora lab07]$
```

Рис. 2.9: Результат работы программы (Вариант 12)

Созданные файлы \*.asm скопировали в каталог ~/work/study/2022-2023/“Архитектура компьютера”/archpc/labs/lab07/ и загрузили на Github.

---

**Листинг 2.3** Программа вычисления выражения  $f(x) = (5*2+3)/3$ 

---

```
;-----  
; 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
```

---

---

**Листинг 2.4** Программа вычисления выражения  $f(x) = (4*6+2)/5$ 

---

```
;-----  
; 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
```

---

---

**Листинг 2.5** Программа вычисления варианта задания по номеру студенческого билета

---

```
;-----  
; 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`  
    xor edx,edx  
    mov ebx,20  
    div ebx  
    inc edx  
  
    mov eax,rem  
    call sprint  
    mov eax,edx  
    call iprintLF  
  
    call quit
```

---

---

**Листинг 2.6** Программа вычисления выражения  $f(x) = (8x-6)/2$ 

---

```
%include "in_out.asm"
```

```
SECTION .data
```

```
msg: DB "Выражение f(x)= (8*x-6)/2", 10, "Введите значение x:", 0
div: DB 'Результат: ', 0
```

```
SECTION .bss
```

```
x: RESB 80 ; буфер размером 80 байт
```

```
SECTION .text
```

```
GLOBAL _start
```

```
_start:
```

```
;Вывод строки с выражением f(x)
```

```
mov eax, msg
call sprint
```

```
;Ввод значения x
```

```
mov ecx, x
mov edx, 80
call sread
```

```
;Преобразование ввода в числовой вид
```

```
mov eax, x
call atoi; EAX=x как число
```

```
;Вычисление выражения
```

```
mov ebx, 8
mul ebx; EAX=EAX*EBX
sub eax, 6; EAX= EAX-6
```

```
xor edx,edx; обнуляем значение регистра EDX для корректной работы команды div
```

```
mov ebx, 2
```

```
div ebx; EAX=EAX/EBX - частное, EDX= EAX % EBX - остаток от деления
```

```
mov [x], eax; Сохраняю результат вычислений в адрес x
```

```
;Вывод результата вычислений
```

```
mov eax, div
call sprint
mov eax, [x]
call iprintLF
```

```
call quit
```

## **3 Выводы**

В рамках лабораторной работы получили практические навыки использования арифметических инструкций языка ассемблера NASM.