

# **Отчет по лабораторной работе №6.**

**Арифметические операции в NASM**

Зайцева П. Е.

# Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Выводы	12
	Список литературы	13

## Список иллюстраций

3.1	рис.1	. . . . .	7
3.2	рис.2	. . . . .	7
3.3	рис.3	. . . . .	8
3.4	рис.5	. . . . .	8
3.5	рис.6	. . . . .	8
3.6	рис.7	. . . . .	8
3.7	рис.8	. . . . .	9
3.8	рис.9	. . . . .	9
3.9	рис.10	. . . . .	9
3.10	рис.15	. . . . .	10
3.11	рис.12	. . . . .	11
3.12	рис.13	. . . . .	11
3.13	рис.14	. . . . .	11

## Список таблиц

# 1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

## 2 Теоретическое введение

Существует три основных способа адресации:

- Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: `mov ax,bx`.
- Непосредственная адресация – значение операнда задается непосредственно в команде, Например: `mov ax,2`.
- Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

### 3 Выполнение лабораторной работы

Создала каталог для программ по лабораторных работ №6, перешла в него и создала файл lab6-1.asm

```
pezayjceva@dk8n80 ~ $ mkdir ~/work/arch-pc/lab06
pezayjceva@dk8n80 ~ $ cd ~/work/arch-pc/lab06
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $ touch lab6-1.asm
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $
```

Рис. 3.1: рис.1

Ввела в файл lab6-1.asm текст программы из листинга 6.1. Далее создала исполняемый файл и запустила его.

```
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $ d -m elf_i386 -o lab6-1 lab6-1.o
bash: d: команда не найдена
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
pezayjceva@dk8n80 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис. 3.2: рис.2

В данном случае при выводе значения регистра eax мы ожидаем увидеть число 10. Однако результатом будет символ j.

После изменила текст программы и вместо символов, записала в регистры числа.

```

/afs/.dk.sci.pfu.edu.ru/home/p/e/pezayjceva/work/arch-pc/lab06/lab6-1.asm
#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 sprintLF
call quit

```

Рис. 3.3: рис.3

Создала исполняемый файл и запустила его.

Создала файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 .

```

pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $

```

Рис. 3.4: рис.5

Ввела в него текст программы из листинга 6.2.

```

/afs/.dk.sci.pfu.edu.ru/home/p/e/pezayjceva/work/arch-pc/lab06/lab6-2.asm
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit

```

Рис. 3.5: рис.6

Создала исполняемый файл и запустила его.

```

pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ ./lab6-2
106j 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00

```

Рис. 3.6: рис.7



В результате работы программы получила число 106.

Аналогично предыдущему примеру изменила символы на числа.

Создала исполняемый файл и запустила его. Результат 10.

```
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ ./lab6-2
10
```

Рис. 3.7: рис.8

Создала файл lab6-3.asm в каталоге ~/work/arch-pc/lab06

```
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-3.asm
pezayjceva@dk1n22 ~/work/arch-pc/lab06 $
```

Рис. 3.8: рис.9

Из листинга 6.3 ввела текст в lab6-3.asm

Создала исполняемый файл и запустила его.

Далее изменила текст программы под выражение  $\text{X}(\text{X}) = (4 \times 6 + 2)/5$  и запустила.

```
GNU nano 7.2 /afs/.dk.sci.pfu.edu.ru/home/p/e/pezayjceva/work/arch-pc/lab06/lab6-3.asm И
;-----
#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

^G Справка ^O Записать ^W Поиск ^K Вырезать ^T Выполнить M-U Отмена M-A Устан
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^C Позиция M-E Повтор M-6 Копир
```

Рис. 3.9: рис.10

Создала файл variant.asm в каталоге ~/work/arch-pc/lab06

Запустила файл и получила свой номер варианта.

Ваш вариант: 1132239116  
17 ? ? ? ? ? ? ? ?

Рис. 3.10: рис.15

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

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

2. Инструкция mov ecx, x используется, чтобы положить адрес вводимой строки x в регистр ecx mov edx, 80 - запись в регистр edx длины вводимой строки call sread - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры
3. call atoi используется для вызова подпрограммы из внешнего файла, которая преобразует ascii-код символа в целое число и записывает результат в регистр eax

4. За вычисления варианта отвечают строки:

```
xor edx,edx ; обнуление edx для корректной работы div mov ebx,20 ; ebx = 20 div  
ebx ; eax = eax/20, edx - остаток от деления inc edx ; edx = edx + 1
```

5. При выполнении инструкции div ebx остаток от деления записывается в регистр edx
6. Инструкция inc edx увеличивает значение регистра edx на 1
7. За вывод на экран результатов вычислений отвечают строки:

mov eax,edx call iprintLF

Написать программу вычисления выражения  $18(x + 1)/6$

```
GNU nano 7.2 /afs/.dk.sci.pfu.edu.ru/home/p/e/pezayjceva/work/arch-pc/lab06/lab6-4.asm
#include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; секция инициализированных данных
msg: DB 'Введите значение переменной x: ',0
res: DB 'Результат: ',0
SECTION .bss ; секция не инициализированных данных
x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный размер - 80 байт
SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу
; ---- Вычисление выражения
mov eax, msg ; запись адреса выводимого сообщения в eax
call sprint ; вызов подпрограммы печати сообщения
mov ecx, x ; запись адреса переменной в ecx
mov edx, 80 ; запись длины вводимого значения в edx
call sread ; вызов подпрограммы ввода сообщения
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII кода в число, 'eax=x'
add eax,1; eax = eax+1 = x + 1
mov ebx,18 ; запись значения 18 в регистр ebx
mul ebx; EAX=EAX*EBX = (x+1)*18
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,6 ; EBX=6
div ebx ; EAX=EAX/6, EDX=остаток от деления
mov edi,eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax, res ; вызов подпрограммы печати
```

Рис. 3.11: рис.12

Вывод при  $x=3$

```
Введите значение переменной x: Результат: 3
Результат: 3
12
```

Рис. 3.12: рис.13

Вывод при  $x=1$

```
pehayjceva@dk1n22 ~/work/arch-pc/lab06 $ ./lab6-4
Введите значение переменной x: Результат: 1
Результат: 1
```

Рис. 3.13: рис.14

## 4 Выводы

Освоила арифметические инструкции языка ассемблера NASM.

## **Список литературы**