

# **Отчет по лабораторной работе №6**

**Архитектура компьютера**

Иванов Александр Олегович

# Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	7
4	Ответы на вопросы	12
5	Самостоятельная работа	13
6	Выводы	14
	Список литературы	15

# Список иллюстраций

3.1	Использование команды mkdir и touch . . . . .	7
3.2	Использование Midnight Commander для написания кода . . . . .	8
3.3	Запуск файла и его результат . . . . .	8
3.4	Редактирование изначального файла . . . . .	8
3.5	Вывод программы . . . . .	9
3.6	Использование команды touch . . . . .	9
3.7	Вывод программы . . . . .	9
3.8	окно Midnight Commndner . . . . .	9
3.9	Вывод программы . . . . .	9
3.10	Использование команды touch . . . . .	10
3.11	окно Midnight Commndner . . . . .	10
3.12	вывод результата программы . . . . .	10
3.13	окно Midnight Commndner . . . . .	10
3.14	вывод результата программы . . . . .	10
3.15	Использование команды touch . . . . .	10
3.16	Использование Midnight Commander для написания кода и вывод результата команды . . . . .	11
5.1	Использование команды touch . . . . .	13
5.2	окно Midnight Commndner . . . . .	13
5.3	вывод результата программы . . . . .	13

## **Список таблиц**

# 1 Цель работы

Цель данной лабораторной работы - освоение арифметических инструкций языка ассемблера NASM.

## 2 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти.

Регистровая адресация – операнды хранятся в регистрах и в команде используются имена регистров.

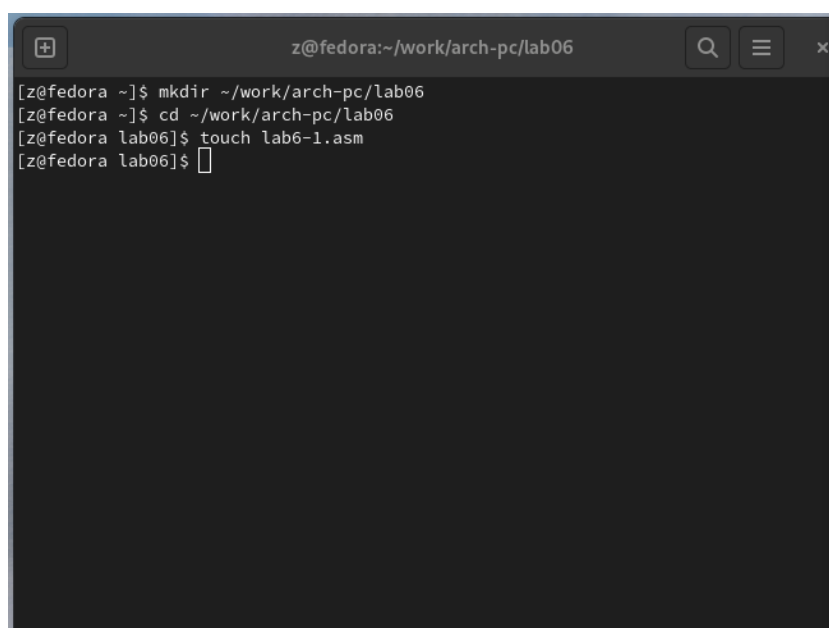
Непосредственная адресация – значение операнда задается непосредственно в команде.

Адресация памяти – операнд задает адрес в памяти. В команде указывается символическое имя ячейки памяти.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно.

### 3 Выполнение лабораторной работы

Создание рабочего каталога и рабочего файла для выполнения 6 лабораторной работы (рис. 3.1).



```
z@fedora:~/work/arch-pc/lab06
[z@fedora ~]$ mkdir ~/work/arch-pc/lab06
[z@fedora ~]$ cd ~/work/arch-pc/lab06
[z@fedora lab06]$ touch lab6-1.asm
[z@fedora lab06]$
```

Рис. 3.1: Использование команды mkdir и touch

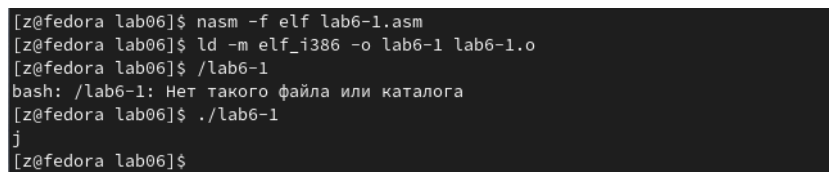
Создание программы вывода значения регистра eax(рис. 3.2).



```
GNU nano 7.2 /home/z/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. 3.2: Использование Midnight Commander для написания кода


Создание исполняемого файла и его запуск(рис. 3.3).



```
[z@fedora lab06]$ nasm -f elf lab6-1.asm
[z@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[z@fedora lab06]$ ./lab6-1
bash: ./lab6-1: Нет такого файла или каталога
[z@fedora lab06]$ ./lab6-1
j
[z@fedora lab06]$
```

Рис. 3.3: Запуск файла и его результат

Замена символов на регистры чисел(рис. 3.4).



```
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintLF
call quit
```

Рис. 3.4: Редактирование изначального файла

Создание файла новой программы и его запуск(рис. 3.5).



```
[z@fedora lab06]$ nasm -f elf lab6-1.asm
[z@fedora lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[z@fedora lab06]$ ./lab6-1

[z@fedora lab06]$
```

Рис. 3.5: Вывод программы

Создание исполняемого файла lab6-2.asm (рис. 3.6).

```
[z@fedora lab06]$ touch ~/work/arch-pc/lab06/lab6-2.asm
[z@fedora lab06]$
```

Рис. 3.6: Использование команды touch

Создание программы вывода значения регистра eax с использованием in out.asm и ее запуск(рис. 3.7).

```
[z@fedora lab06]$ nasm -f elf lab6-2.asm
[z@fedora lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[z@fedora lab06]$ ./lab6-2
106
[z@fedora lab06]$
```

Рис. 3.7: Вывод программы

Редактирование программы вывода значения регистра eax с использованием in out.asm(рис. 3.8).



Рис. 3.8: окно Midnight Commander

Создание исполняемого файла и его запуск(рис. 3.9).



Рис. 3.9: Вывод программы

Создание исполняемого файла lab6-3.asm (рис. 3.10).

```
[z@fedora lab06]$ touch ~/work/arch-pc/lab06/lab6-3.asm  
[z@fedora lab06]$
```

Рис. 3.10: Использование команды touch

Создание программы вычисления выражения  $(5 * 2 + 3)/3$  (рис. 3.11).



Рис. 3.11: окно Midnight Commander

Запуск программы вычисления выражения  $(5 * 2 + 3)/3$  (рис. 3.12).



Рис. 3.12: вывод результата программы

Создание программы вычисления выражения  $(4 * 6 + 2)/5$  (рис. 3.13).



Рис. 3.13: окно Midnight Commander

Запуск программы вычисления выражения  $(4 * 6 + 2)/5$  (рис. 3.14).



Рис. 3.14: вывод результата программы

Создание исполняемого файла variant.asm (рис. 3.15).

```
[z@fedora lab06]$ touch ~/work/arch-pc/lab06/variant.asm  
[z@fedora lab06]$
```

Рис. 3.15: Использование команды touch

Создание программы вычисления варианта задания по номеру студенческого билета и ее запуск(рис. 3.16).

```
[z@fedora lab06]$ nasm -f elf variant.asm
[z@fedora lab06]$ ld -m elf_i386 -o variant variant.o
[z@fedora lab06]$ ./variant
Введите № студенческого билета:
1132236088
Ваш вариант: 9
[z@fedora lab06]$
```

Рис. 3.16: Использование Midnight Commander для написания кода и вывод результата команды

## 4 Ответы на вопросы

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`

## 5 Самостоятельная работа

Создание исполняемого файла ans.asm (рис. 5.1).



```
[z@fedora lab06]$ touch ans.asm
```

Рис. 5.1: Использование команды touch

Создание программы вычисления выражения  $(x * 31 - 5) + 10$  (рис. 5.2).



Рис. 5.2: окно Midnight Commndner

Запуск программы вычисления выражения  $(x * 31 - 5) + 10$  (рис. 5.3).



Рис. 5.3: вывод результата программы

## **6 Выводы**

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

## **Список литературы**