

Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Хрисанова Ксения Олеговна

Содержание

1	Цель работы	1
2	Задание.....	1
3	Теоретическое введение	1
4	Выполнение лабораторной работы.....	2
4.1	Символьные и численные данные в NASM	2
4.2	Выполнение арифметических операций в NASM	5
4.3	Ответы на контрольные вопросы	7
4.4	Задание для самостоятельной работы	8
5	Вывод	9
6	Список литературы.....	9

1 Цель работы

Цель данной лабораторной 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 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. Далее рассмотрены все существующие способы задания адреса хранения операндов – способы адресации. 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 – operand задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

4 Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

Создаю каталог для программ лабораторной работы №6 и перехожу в него, создаю там файл (рис. 1).

```
kseニアhisanova@fedora:~$ mkdir ~/work/arch-pc/lab06
kseニアhisanova@fedora:~$ cd ~/work/arch-pc/lab06
kseニアhisanova@fedora:~/work/arch-pc/lab06$ touch lab6-1.asm
kseニアhisanova@fedora:~/work/arch-pc/lab06$ █
```

Рис. 1: Создание нового каталога

В созданном файле ввожу программу из листинга (рис. 2).



```
mc [kseニアhisanova@fedora:~/work/arch-pc/lab06 — /usr/bin/mc -P /tmp/mc.pwd.G7iBsE ~/work/arch-pc/lab06
GNU nano 8.3          /home/kseニアhisanova/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. 2: Сохранение новой программы

Создаю исполняемый файл и запускаю его, вывод программы отличается от предполагаемого, потому что коды символов вместе дают символ j по таблице ASCII.

```
kseニアhisanova@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
kseニアhisanova@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
kseニアhisanova@fedora:~/work/arch-pc/lab06$ ./lab6-1
j
kseニアhisanova@fedora:~/work/arch-pc/lab06$ █
```

Рис. 3: Запуск изначальной программы

Изменяю текст программы, убрав кавычки (рис. 4).

```
GNU nano 8.3          /home/kseniahrisanova/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. 4: Измененная программа

Программа выводит пустую строку, потому что в буфер записывается значение 10, которое в ASCII соответствует символу «LF» (перевод строки). (рис. 5).

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

```
kseniahrisanova@fedora:~/work/arch-pc/lab06$
```

Рис. 5: Запуск измененной программы

Создаю новый файл для будущей программы и записываю в нее код из листинга (рис. 6).

```
GNU nano 8.3          /home/kseniahrisanova/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

```

^G Справка ^O Записать ^F Поиск ^K Вырезать ^T Выполнить ^C Позиция M-U Отмена
^X Выход ^R ЧитФайл ^\ Замена ^U Вставить ^J Выровнять ^/ К строке M-E Повтор

Рис. 6: Вторая программа

Создаю исполняемый файл и запускаю его. В программе используются символьные константы '6' и '4'. Ассемблер подставляет вместо них их ASCII-коды: 54 и 52 соответственно. Команда add eax, ebx выполняет сложение этих кодов, в результате чего в регистре EAX получается значение 106. Процедура iprintLF интерпретирует содержимое EAX как целое число и выводит его в десятичном виде, поэтому на экран выводится число 106, а не сумма цифр $6 + 4 = 10$. (рис. 7).

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

Рис. 7: Вывод второй программы

Убираю кавычки в программе. (рис.8)



```
GNU nano 8.3          /home/kseniahrisanova/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

[ Прочитано 9 строк ]
```

The screenshot shows the nano text editor interface. The file path is /home/kseniahrisanova/work/arch-pc/lab06/lab6-2.asm. The assembly code has been modified from the previous version by removing the quotes around the constant values. The menu bar at the top includes 'Справка', 'Выход', 'Записать', 'Читать', 'Поиск', 'Замена', 'Вырезать', 'Вставить', 'Выполнить', 'Позиция', 'Отмена', and 'Повтор'. The status bar at the bottom indicates 'Прочитано 9 строк'.

Рис. 8: Измененная программа

И снова ее запускаю и получаю предполагаемый изначально результат. (рис. 9).

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

Рис. 9: Вывод измененной второй программы

Заменив функцию вывода на iprint, я получаю тот же результат, но без переноса строки (рис. 10).

```

GNU nano 8.3                               /home/kseniahrisanova/work/arch-pc/lab06/lab6-2.asm      Изменён
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
    mov eax,6
    mov ebx,4
    add eax,ebx
    call iprint
    call quit

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

```

Рис. 10: Замена функции вывода во второй программе

4.2 Выполнение арифметических операций в NASM

Создаю новый файл и копирую в него содержимое листинга (рис. 11).

```

GNU nano 8.3                               /home/kseniahrisanova/work/arch-pc/lab06/lab6-3.asm
%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

^G Справка      ^O Записать      ^F Поиск      ^X Вырезать      ^T Выполнить      ^Q Позиция      M-U Отмена
^X Выход      ^R Читайфайл      ^L Замена      ^U Вставить      ^J Выровнять      ^V К строке      M-E Повтор
M-A Установить M-V На скобку      M-G Копировать      M-B Обр. поиск

```

Рис. 11: Третья программа

Программа выполняет арифметические вычисления и выводит результат выражения и его остаток от деления (рис. 12).

```

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

Рис. 12: Запуск третьей программы

Заменив переменные в программе для выражения $f(x) = (4*6+2)/5$ (рис. 13).

Рис. 13: Изменение третьей программы

Запуск программы дает корректный результат (рис. 14).

```

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

Рис. 14: Запуск измененной третьей программы

Создаю новый файл и помещаю текст из листинга (рис. 15).

```
GNU nano 8.3
/home/kseniahrisanova/work/arch-pc/lab06/variant.asm

SECTION .data
msg: DB 'Введите номер студенческого: ',0
rem: DB 'Ваш вариант: ',0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start

_start:
    mov eax,msg
    call sprintLF

    mov ebx,x
    mov edx,80
    call sread

    mov eax,x
    call atoi

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

    mov eax,rem
    call sprint

    mov eax,edx
    call iprintf

%
```

Справка Выход Записать ЧитФайл Поиск Замена Вырезать Вставить Выполнить Позиция К строке Отмена Повтор Установить На скобку Копировать Обр. поиск Предыдущий Следующий

Рис. 15: Программа для подсчета варианта

Запустив программу и указав свой номер студенческого билета, я получила свой вариант для дальнейшей работы. (рис. 16).

```
kseniahrisanova@fedora:~/work/arch-pc/lab06$ nasm -f elf variant.asm
kseniahrisanova@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o variant variant.o

kseniahrisanova@fedora:~/work/arch-pc/lab06$ ./variant
Введите номер студенческого:
1032253563
Ваш вариант: 1
kseniahrisanova@fedora:~/work/arch-pc/lab06$
```

Рис. 16: Запуск программы для подсчета варианта

4.3 Ответы на контрольные вопросы

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

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

2. • **mov ecx, x**

Загружает в регистр ECX адрес буфера x, куда будет записываться введённая строка.

- **mov edx, 80**

Указывает максимальную длину вводимой строки — 80 символов.

- **call sread**

Вызывает подпрограмму sread из файла in_out.asm, которая считывает строку с клавиатуры и сохраняет её в буфер по адресу ECX.

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. inc edx увеличивает значение регистра edx на 1.

7. За вывод на экран результатов отвечают строки:

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

4.4 Задание для самостоятельной работы

В соответствии с выбранным вариантом, я вычисляю значение функции $f(x) = (10+2x)/3$.

Проверка на нескольких переменных показывает корректное выполнение программы (рис. 17).

The screenshot shows a terminal window with the following content:

```
GNU nano 8.3 /home/kseniahrisanova/work/arch-pc/lab06/lab6-4.asm  
#include "in_out.asm"  
  
SECTION .data  
msg: DB 'Введите значение переменной x: ',0  
rem: DB 'Результат: ',0  
  
SECTION .bss  
x: RESB 80  
  
SECTION .text  
GLOBAL _start  
  
.start:  
    mov eax, msg  
    call sprint  
  
    mov ecx, x  
    mov edx, 80  
    call sread  
  
    mov eax, x  
    call atoi  
  
    mov ebx, 2  
    mul ebx  
  
    add eax, 10  
  
    mov ebx, 3  
    xor edx, edx  
    div ebx  
    mov edi, eax  
  
    mov eax, rem  
    call sprint  
  
    mov eax, edi  
    call iprint  
  
    call quit
```

At the bottom of the terminal window, there is a menu bar with Russian labels and a status bar indicating the file is saved.

```
kseniahrisanova@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm  
kseniahrisanova@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 -o lab6-4 lab6-4.o  
kseniahrisanova@fedora:~/work/arch-pc/lab06$ ./lab6-4  
Введите значение переменной x: 1  
Результат: 4  
kseniahrisanova@fedora:~/work/arch-pc/lab06$ ./lab6-4  
Введите значение переменной x: 10
```

Рис. 17: Запуск и проверка программы

Код моей программы:

```
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
    mov eax, msg
    call sprint
    mov ecx, x
    mov edx, 80
    call sread
    mov eax, x
    call atoi
    mov ebx, 2
    mul ebx
    add eax, 10
    mov ebx, 3
    xor edx, edx
    div ebx
    mov edi, eax
    mov eax, rem
    call sprint
    mov eax, edi
    call iprint
    call quit
```

5 Вывод

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

6 Список литературы

1. Курс на ТУИС
2. Лабораторная работа №6