

Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Глущенко Евгений Игоревич

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 Выполнение лабораторной работы

3.1 Символьные и численные данные в NASM

С помощью утилиты `mkdir` создаю директорию, в которой буду работать (рис. 1).
Перехожу в созданный каталог с помощью `cd`.

```
[eigluthenko@38 ~]$ mkdir work/study/2023-2024/архитектура\ компьютера/stady_2023-2024_arch_pc/lab06  
[eigluthenko@38 ~]$ cd work/study/2023-2024/архитектура\ компьютера/stady_2023-2024_arch_pc/lab06  
[eigluthenko@38 lab06]$
```

Рис. 1: Создание директории

С помощью `touch` создаю файл `lab6-1.asm` (рис. 2).

```
[eigluthenko@38 lab06]$ touch lab6-1.asm  
[eigluthenko@38 lab06]$ ls  
lab6-1.asm  
[eigluthenko@38 lab06]$
```

Рис. 2: Создание файла

Копирую в текущий каталог файл `in_out.asm` (рис. 3).

```
[eigluthenko@38 lab06]$ cp ~/Загрузки/in_out.asm in_out.asm
[eigluthenko@38 lab06]$ ls
in_out.asm  lab6-1.asm
[eigluthenko@38 lab06]$
```

Рис. 3: Создание копии файла

Открываю созданный файл lab6-1.asm, вставляю в него программу вывода(рис. 4).



```
*lab6-1.asm
~/work/study/2023-2024/архитектура компьютера/stady_2023-2024_arch_pc/lab06

1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax,'6'
8 mov ebx,'4'
9 add eax,ebx
10 mov [buf1],eax
11 mov eax,buf1
12 call sprintf
13 call quit
```

Рис. 4: Редактирование файла

Создаю исполняемый файл программы и запускаю его (рис. 5). Вывод программы: символ j, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6.

```
[eigluthenko@38 lab06]$ nasm -f elf lab6-1.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[eigluthenko@38 lab06]$ ./lab6-1
j
[eigluthenko@38 lab06]$
```

Рис. 5: Запуск исполняемого файла

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4 (рис. 6).

```

1 %include 'in_out.asm'
2 SECTION .bss
3 buf1: RESB 80
4 SECTION .text
5 GLOBAL _start
6 _start:
7 mov eax,6
8 mov ebx,4
9 add eax,ebx
10 mov [buf1],eax
11 mov eax,buf1
12 call sprintf
13 call quit

```

Рис. 6: Редактирование файла

Создаю новый исполняемый файл программы и запускаю его (рис. 7). Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран.

```

[eigluthenko@38 lab06]$ gedit lab6-1.asm
[eigluthenko@38 lab06]$ nasm -f elf lab6-1.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-1 lab6-1.o
[eigluthenko@38 lab06]$ ./lab6-1

[eigluthenko@38 lab06]$

```

Рис. 7: Запуск исполняемого файла

Создаю новый файл lab6-2.asm с помощью touch (рис. 8).

```

[eigluthenko@38 lab06]$ touch lab6-2.asm
[eigluthenko@38 lab06]$

```

Рис. 8: Создание файла

Ввожу в файл текст другой программы для вывода значения регистра eax (рис. 9).

```

1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax, '6'
6 mov ebx, '4'
7 add eax, ebx
8 call iprintLF
9 call quit

```

Рис. 9: Редактирование файла

Создаю и запускаю исполняемый файл lab6-2 (рис. 10). Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”.

```

[eigluthenko@38 lab06]$ nasm -f elf lab6-2.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[eigluthenko@38 lab06]$ ./lab6-2
106
[eigluthenko@38 lab06]$

```

Рис. 10: Запуск исполняемого файла

Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4 (рис. 11).

```

1 %include 'in_out.asm'
2 SECTION .text
3 GLOBAL _start
4 _start:
5 mov eax, 6
6 mov ebx, 6
7 add eax, ebx
8 call iprintLF
9 call quit

```

Рис. 11: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис. 12).. Теперь программа складывает не соответствующие символам коды в системе ASCII, а сами числа, поэтому вывод 12.

```
[eigluthenko@38 lab06]$ nasm -f elf lab6-2.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[eigluthenko@38 lab06]$ ./lab6-2
12
[eigluthenko@38 lab06]$
```

Рис. 12: Запуск исполняемого файла

Заменяю в тексте программы функцию `iprintLF` на `iprint` (рис. 13).

```
5 mov eax,0
6 mov ebx,6
7 add eax,ebx
8 call iprint
9 call quit
```

Рис. 13: Редактирование файла

Создаю и запускаю новый исполняемый файл (рис. 14). Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполнялась с функцией `iprintLF`, а `iprint` не добавляет к выводу символ переноса строки, в отличие от `iprintLF`.

```
[eigluthenko@38 lab06]$ nasm -f elf lab6-2.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-2 lab6-2.o
[eigluthenko@38 lab06]$ ./lab6-2
12[eigluthenko@38 lab06]$
```

Рис. 14: Запуск исполняемого файла

3.2 Выполнение арифметических операций в NASM

Создаю файл `lab6-3.asm` с помощью утилиты `touch` (рис. 15).

```
12[eigluthenko@38 lab06]$ touch lab6-3.asm
[eigluthenko@38 lab06]$
```

Рис. 15: Создание файла

Ввожу в созданный файл текст программы для вычисления значения выражения $f(x) = (5 * 2 + 3) / 3$ (рис. 16).

```

1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data
3 div: DB 'Результат: ',0
4 rem: DB 'Остаток от деления: ',0
5 SECTION .text
6 GLOBAL _start
7 _start:
8 ; ---- Вычисление выражения
9 mov eax,5 ; EAX=5
10 mov ebx,2 ; EBX=2
11 mul ebx ; EAX=EAX*EBX
12 add eax,3 ; EAX=EAX+3
13 xor edx,edx ; обнуляем EDX для корректной работы div
14 mov ebx,3 ; EBX=3
15 div ebx ; EAX=EAX/3, EDX=остаток от деления
16 mov edi,eax ; запись результата вычисления в 'edi'
17 ; ---- Вывод результата на экран
18 mov eax,div ; вызов подпрограммы печати
19 call sprint ; сообщения 'Результат: '
20 mov eax,edi ; вызов подпрограммы печати значения
21 call iprintLF ; из 'edi' в виде символов
22 mov eax,rem ; вызов подпрограммы печати
23 call sprint ; сообщения 'Остаток от деления: '
24 mov eax,edx ; вызов подпрограммы печати значения
25 call iprintLF ; из 'edx' (остаток) в виде символов
26 call quit ; вызов подпрограммы завершения

```

Рис. 16: Редактирование файла

Создаю исполняемый файл и запускаю его (рис. 17).

```

[eigluthenko@38 lab06]$ nasm -f elf lab6-3.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[eigluthenko@38 lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Рис. 17: Запуск исполняемого файла

Изменяю программу, чтобы она вычисляла значение $f(x) = (4 * 6 + 2)/5$ (рис. 18).

```

%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

```

Рис. 18: Изменение программы

Создаю и запускаю новый исполняемый файл (рис. 19). Как видим, все верно.

```

[eigluthenko@38 lab06]$ nasm -f elf lab6-3.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[eigluthenko@38 lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[eigluthenko@38 lab06]$

```

Рис. 19: Запуск исполняемого файла

Создаю файл variant.asm (рис. 20).

```
[eigluthenko@38 lab06]$ touch variant.asm  
[eigluthenko@38 lab06]$ gedit variant.asm
```

Рис. 20: Создание файла

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета (рис. 21).

```
1 %include 'in_out.asm'  
2 SECTION .data  
3 msg: DB 'Введите № студенческого билета: ',0  
4 rem: DB 'Ваш вариант: ',0  
5 SECTION .bss  
6 x: RESB 80  
7 SECTION .text  
8 GLOBAL _start  
9 _start:  
0 mov eax, msg  
1 call sprintf  
2 mov ecx, x  
3 mov edx, 80  
4 call sread  
5 mov eax,x ; вызов подпрограммы преобразования  
6 call atoi ; ASCII кода в число, `eax=x`  
7 xor edx,edx  
8 mov ebx,20  
9 div ebx  
0 inc edx  
1 mov eax,rem  
2 call sprintf  
3 mov eax,edx  
4 call iprintLF  
5 call quit
```

Рис. 21: Редактирование файла

Создаю и запускаю исполняемый файл (рис. 22). Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 11.


```
[eigluthenko@38 lab06]$ nasm -f elf variant.asm
[eigluthenko@38 lab06]$ ld -m elf_i386 -o variant variant
[eigluthenko@38 lab06]$ ./variant
Введите № студенческого билета:
1132239110
Ваш вариант: 11
[eigluthenko@38 lab06]$
```

Рис. 22: Запуск исполняемого файла

3.2.1 Ответы на вопросы по программе

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 read` - 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
```

3.3 Выполнение заданий для самостоятельной работы

Создаю файл `lab7-4.asm` (рис. 23).

```
[eigluthenko@38 lab06]$ touch lab6-4.asm
[eigluthenko@38 lab06]$
```

Рис. 23: Создание файла

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения $(11 + x) * 2 - 6$ (рис. 24).

```
~/work/study/2023-2024/архитектура компьютера/stady_2023-2024_arch_pc/lab06
1 %include 'in_out.asm' ; подключение внешнего файла
2 SECTION .data ; секция иницированных данных
3 msg: DB 'Введите значение переменной x: ',0
4 rem: DB 'Результат: ',0
5 SECTION .bss ; секция не иницированных данных
6 x: RESB 80 ; Переменная, значение к-рой будем вводить с клавиатуры, выделенный размер - 80 байт
7 SECTION .text ; Код программы
8 GLOBAL _start ; Начало программы
9 _start: ; Точка входа в программу
0 ; ---- Вычисление выражения
1 mov eax, msg ; запись адреса выводимого сообщения в eax
2 call sprint ; вызов подпрограммы печати сообщения
3 mov ecx, x ; запись адреса переменной в ecx
4 mov edx, 80 ; запись длины вводимого значения в edx
5 call sread ; вызов подпрограммы ввода сообщения
6 mov eax,x ; вызов подпрограммы преобразования
7 call atoi ; ASCII кода в число, eax=x
8 add eax,11; eax = eax+11 = x + 11
9 mov ebx,2 ; запись значения 2 в регистр ebx
0 mul ebx; EAX=EAX*EBX = (x+11)*2
1 add eax,-6; eax = eax-6 = (x+11)*2-6
2 mov edi,eax ; запись результата вычисления в 'edi'
3 ; ---- Вывод результата на экран
4 mov eax,rem ; вызов подпрограммы печати
5 call sprint ; сообщения 'Результат: '
6 mov eax,edi ; вызов подпрограммы печати значения
7 call iprint ; из 'edi' в виде символов
8 call quit ; вызов подпрограммы завершения
```

Рис. 24: Написание программы

Создаю и запускаю исполняемый файл (рис. 25). При вводе значения 3, вывод - 22.

```
eigluthenko@38 lab06]$ nasm -f elf lab6-4.asm
eigluthenko@38 lab06]$ ld -m elf_i386 -o lab6-4 lab6-4.o
eigluthenko@38 lab06]$ ./lab6-4
```

Рис. 25: Запуск исполняемого файла

Провожу еще один запуск исполняемого файла для проверки работы программы с другим значением на входе (рис. 26). Программа отработала верно.

```
eigluthenko@38 lab06]$ ./lab6-4
Введите значение переменной x: 3
Результат: 22[eigluthenko@38 lab06]$
```

Рис. 26: Запуск исполняемого файла

Программа для вычисления выражения $(11 + x) * 2 - 6$.

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data ; секция иницированных данных
```

```

msg: DB 'Введите значение переменной x: ',0
rem: DB 'Результат: ',0
SECTION .bss ; секция не инициированных 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, 11;  $eax = eax + 11 = x + 11$ 
mov ebx, 2 ; запись значения 2 в регистр ebx
mul ebx;  $EAX = EAX * EBX = (x + 11) * 2$ 
add eax, -6;  $eax = eax - 6 = (x + 11) * 2 - 6$ 
mov edi, eax ; запись результата вычисления в 'edi'
; ---- Вывод результата на экран
mov eax, rem ; вызов подпрограммы печати
call sprint ; сообщения 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call iprint ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения

```

4 Выводы

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