

Лабораторная работа №7

Дисциплина: Архитектура компьютера

Александрова Ульяна Вадимовна

Содержание

1	Цель работы	5
2	Выполнение лабораторной работы	6
3	Задание для самостоятельной работы	10
4	Выводы	12

Список иллюстраций

2.1	Запуск программы lab7-1	6
2.2	Запуск новой программы	6
2.3	Запуск программы lab7-2	7
2.4	Запуск измененной программы	7
2.5	Запуск измененной программы	7
2.6	Запуск программы lab7-3	7
2.7	Редактирование текста файла	8
2.8	Запуск измененной программы	8
2.9	Запуск программы variant	8
3.1	Запуск программы	11

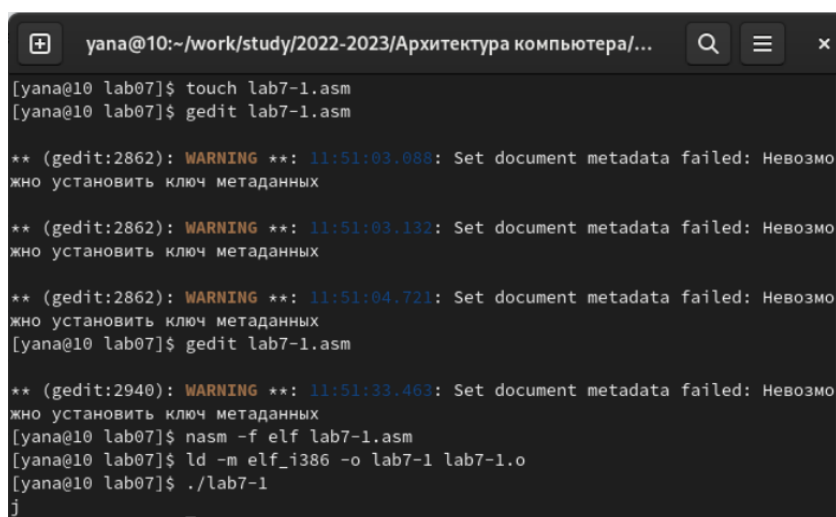
Список таблиц

1 Цель работы

Целью лабораторной работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

Создаю каталог lab07 и перехожу туда. В каталоге создаю файл *lab7-1.asm* и редактирую в текстовом редакторе так, чтобы текст соответствовал предложенному листингу. Создаю исполняемый файл и запускаю программу (рис. 2.1).



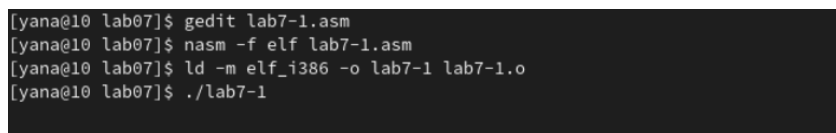
```
yana@10:~/work/study/2022-2023/Архитектура компьютера/...
[yana@10 lab07]$ touch lab7-1.asm
[yana@10 lab07]$ gedit lab7-1.asm

** (gedit:2862): WARNING **: 11:51:03.088: Set document metadata failed: Невозможно установить ключ метаданных
** (gedit:2862): WARNING **: 11:51:03.132: Set document metadata failed: Невозможно установить ключ метаданных
** (gedit:2862): WARNING **: 11:51:04.721: Set document metadata failed: Невозможно установить ключ метаданных
[yana@10 lab07]$ gedit lab7-1.asm

** (gedit:2940): WARNING **: 11:51:33.463: Set document metadata failed: Невозможно установить ключ метаданных
[yana@10 lab07]$ nasm -f elf lab7-1.asm
[yana@10 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[yana@10 lab07]$ ./lab7-1
j
```

Рис. 2.1: Запуск программы lab7-1

Далее заменяю в программе символы '6' и '4' на 6 и 4, а затем запускаю программу. На экран был выведен символ перевода строки с кодом 10 (рис. 2.2).



```
[yana@10 lab07]$ gedit lab7-1.asm
[yana@10 lab07]$ nasm -f elf lab7-1.asm
[yana@10 lab07]$ ld -m elf_i386 -o lab7-1 lab7-1.o
[yana@10 lab07]$ ./lab7-1
```

Рис. 2.2: Запуск новой программы

Создаю новый файл *lab7-2.asm* и ввожу в него текст программы из листинга 7.2. В результате работы программы получаю число 106 (рис. 2.3).

```
[yana@10 lab07]$ touch lab7-2.asm
[yana@10 lab07]$ gedit lab7-2.asm
[yana@10 lab07]$ nasm -f elf lab7-2.asm
[yana@10 lab07]$ ld -m elf_i386 -o lab7-2 lab7-2.o
[yana@10 lab07]$ ./lab7-2
106
```

Рис. 2.3: Запуск программы lab7-2

Заменяю символы на числа и при запуске программы получаю число 10, тк система складывает числа, а не коды, соответствующие данным символам (рис. 2.4).

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

Рис. 2.4: Запуск измененной программы

Меняю функцию `iprintLF` на `iprint`. В результате запуска программы видно, что система не перешла на новую строку (рис. 2.5).

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

Рис. 2.5: Запуск измененной программы

Создаю файл *lab7-3.asm* и ввожу в него текст из листинга 7.3 (рис. 2.6).

```
[yana@10 lab07]$ gedit lab7-3.asm
[yana@10 lab07]$ nasm -f elf lab7-3.asm
[yana@10 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[yana@10 lab07]$ ./lab7-3
Результат: 4
Остаток от деления: 1
```

Рис. 2.6: Запуск программы lab7-3

Изменяю текст файла для соответствия заданию (рис. 2.7).

```

1 ;-----
2 ; Программа вычисления выражения
3 ;-----
4
5 %include 'in_out.asm' ; подключение внешнего файла
6
7 SECTION .data
8 div: DB 'Результат: ',0
9 rem: DB 'Остаток от деления: ',0
10
11 SECTION .text
12 GLOBAL _start
13 _start:
14
15 ; ---- Вычисление выражения
16
17 mov eax,4 ; EAX=4
18 mov ebx,6 ; EBX=6
19 mul ebx ; EAX=EAX*EBX
20 add eax,2 ; EAX=EAX+2
21 xor edx,edx ; обнуляем EDX для корректной работы div
22 mov ebx,5 ; EBX=5
23 div ebx ; EAX=EAX/5, EDX=остаток от деления
24 mov edi,eax ; запись результата вычисления в 'edi'
25
26 ; ---- Вывод результата на экран
27
28 mov eax,div ; вызов подпрограммы печати
29 call sprint ; сообщения 'Результат: '
30 mov eax,edi ; вызов подпрограммы печати значения
31 call iprintLF ; из 'edi' в виде символов
32 mov eax,rem ; вызов подпрограммы печати
33 call sprint ; сообщения 'Остаток от деления: '
34 mov eax,edx ; вызов подпрограммы печати значения
35 call iprintLF ; из 'edx' (остаток) в виде символов
36
37 call quit ; вызов подпрограммы завершения

```

Рис. 2.7: Редактирование текста файла

Запускаю файл и получаю результат (рис. 2.8).

```

[yana@10 lab07]$ gedit lab7-3.asm
[yana@10 lab07]$ nasm -f elf lab7-3.asm
[yana@10 lab07]$ ld -m elf_i386 -o lab7-3 lab7-3.o
[yana@10 lab07]$ ./lab7-3
Результат: 5
Остаток от деления: 1

```

Рис. 2.8: Запуск измененной программы

Создаю файл *variant.asm* и вношу текст из листинга 7.4 в него. В результате запуска программы получаю свой вариант (рис. 2.9).

```

[остаток от деления: 1
[yana@10 lab07]$ touch variant.asm
[yana@10 lab07]$ gedit variant.asm
[yana@10 lab07]$ nasm -f elf variant.asm
[yana@10 lab07]$ ld -m elf_i386 -o variant variant.o
[yana@10 lab07]$ ./variant
Введите № студенческого билета:
1132226444
Ваш вариант: 5

```

Рис. 2.9: Запуск программы variant

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

```
mov eax,rem
```

```
call sprint
```


2. *mov ecx, x* используется для того, чтобы положить адрес вводимой пользователем строки в регистр *ecx*. *mov edx, 80* – запись длины видимой строки. *call sread* – вызов подпрограммы из файла.
3. Инструкция используется для вызова подпрограммы, преобразующей код символа (ASCII) в целое число, а также записывает результат в регистр *eax*.
4. За вариант 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. Остаток от деления записывается в регистр *edx*.
6. инструкция *inc edx* используется для увеличения значения в регистре *edx* на один.
7. За вывод на экран результатов отвечают строки

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

3 Задание для самостоятельной работы

Я создала файл *zadanie.asm* и заполнила программу в соответствии с моим вариантом (рис. 3.1).

```
%include 'in_out.asm'

SECTION .data
msg: DB 'Введите x: ',0
rem: DB 'Ответ: ',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`
mov ebx,9
```

```

mul ebx
sub eax,8
mov ebx,8
div ebx
xor ebx, ebx
mov edi, eax

mov eax, rem
call sprint ; сообщения 'Результат: '
mov eax,edi ; вызов подпрограммы печати значения
call iprintLF ; из 'edi' в виде символов

call quit ; вызов подпрограммы завершения

```

```

[yana@10 lab07]$ gedit zadanie.asm
[yana@10 lab07]$ nasm -f elf zadanie.asm
[yana@10 lab07]$ ld -m elf_i386 -o zadanie zadanie.o
[yana@10 lab07]$ ./zadanie
Введите x:
8
Ответ: 8
[yana@10 lab07]$ ./zadanie
Введите x:
64
Ответ: 71

```

Рис. 3.1: Запуск программы

4 Выводы

Я освоила арифметические инструкции языка ассемблер NASM.