

Отчет по лабораторной работе №6

Архитектура компьютера

Бурлакова Алина Андреевна

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	13
	Список литературы	14

Список иллюстраций

3.1	Создание файла	7
3.2	введение текста в файл	7
3.3	Запуск файла	7
3.4	Изменение текста	8
3.5	Запуск файла	8
3.6	Создание файла	8
3.7	Ввод текста	9
3.8	Запуск файла	9
3.9	Изменение	9
3.10	Запуск файла	10
3.11	Создание файла	10
3.12	Ввод текста	10
3.13	Создание файла	11
3.14	Ввод текста	11

Список таблиц

1 Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 Задание

Получить навыки в работе с ассемблером NASM.

3 Выполнение лабораторной работы

1. Создадим каталог для программ лабораторной работы No 6, перейдем в него и создадим файл lab6-1.asm (рис. [-fig:001])

```
aaburlakova@dk6n55 ~ $ mkdir ~/work/arch-pc/lab06
aaburlakova@dk6n55 ~ $ cd ~/work/arch-pc/lab06
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ touch lab6-1.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ mc
```

Рис. 3.1: Создание файла

2. Введем в файл lab6-1.asm текст программы из листинга 6.1 (рис. [-fig:002])

```
%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 sprintf
call quit
```

Рис. 3.2: введение текста в файл

3. Создадим исполняемый файл и запустим его. (рис. [-fig:003])

```
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ./lab6-1
j
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $
```

Рис. 3.3: Запуск файла

4. Далее изменим текст программы и вместо символов, запишем в регистры числа. (рис. [-fig:004])

```
%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 sprintf
call quit
```

Рис. 3.4: Изменение текста

5. Создадим исполняемый файл и запустим его. (рис. [-fig:005])

```
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ./lab7-1
bash: ./lab7-1: Нет такого файла или каталога
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ./lab6-1

aaburlakova@dk6n55 ~/work/arch-pc/lab06 $
```

Рис. 3.5: Запуск файла

6. Создадим файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и введем в него текст программы из листинга 7.2. (рис. [-fig:006])

```
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $
```

Рис. 3.6: Создание файла

7. Введем в файл lab7-2.asm текст программы из листинга 6.2. (рис. [-fig:007])


```
lab6-2.asm [-M--] 9 L: [ 1+ 8 9/ 9] *(116 / 116b) <EOF> [*][X]
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, '6'
mov ebx, '4'
add eax, ebx
call iprintLF
call quit
```

Рис. 3.7: Ввод текста

8. Создадим исполняемый файл и запустим его. (рис. [-fig:008])

```
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ./lab6-1

aaburlakova@dk6n55 ~/work/arch-pc/lab06 $
```

Рис. 3.8: Запуск файла

9. Изменим символы на числа в файле. (рис. [-fig:009])

```
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax, 6
mov ebx, 4
add eax, ebx
call iprintLF
call quit
```

1 Помощь 2 Сох-ть 3 Блок 4 Замена 5 Копия 6 Пер-ть 7 Поиск 8 Уда-ть 9 МенюМС 10 Выход

Рис. 3.9: Изменение

10. Создадим исполняемый файл и запустим его. (рис. [-fig:010])

```
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $ ./lab6-2
10
aaburlakova@dk6n55 ~/work/arch-pc/lab06 $
```

Рис. 3.10: Запуск файла

11. Создайте файл lab6-3.asm в каталоге ~/work/arch-pc/lab07. (рис. [-fig:011])

```
aaburlakova@dk6n55 ~ $ touch ~/work/arch-pc/lab07/lab6-3.asm
aaburlakova@dk6n55 ~ $
```

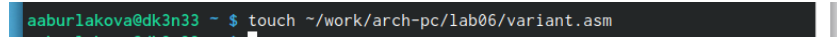
Рис. 3.11: Создание файла

12. Введем текст программы из листинга 6.3 в lab6-3.asm и запустим его. (рис. [-fig:012])

```
touch ~/work/arch-pc/lab07/lab6-3.asm%include "in_out.asm" ; подключение внешнего
%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, EBX=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 iprintfLF ; 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 iprintfLF ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рис. 3.12: Ввод текста

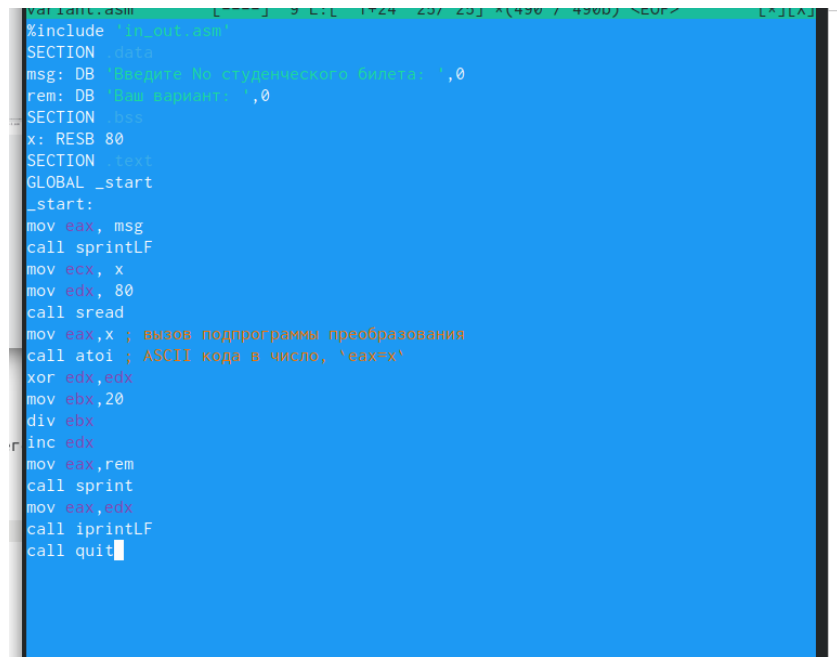
13. Создадим файл variant.asm в каталоге ~/work/arch-pc/lab06 (рис. [-fig:013])



```
aaburlakova@dk3n33 ~ $ touch ~/work/arch-pc/lab06/variant.asm
```

Рис. 3.13: Создание файла

14. Текст программы из листинга 6.4 введем в файл variant.asm. (рис. [-fig:014])



```
variant.asm
%include "io.inc"
SECTION .data
msg: DB "Введите № студенческого билета: ",0
rem: DB "Ваш вариант: ",0
SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x ; вызов подпрограммы преобразования
call atoi ; ASCII коды в число, 'eax=x'
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
call quit
```

Рис. 3.14: Ввод текста

15. Ответы на вопросы.

1) Какие строки листинга 7.4 отвечают за вывод на экран сообщения 'Ваш вариант:'?

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

2) Для чего используются следующие инструкции? `nasm mov ecx, x` `mov edx, 80` `call sread`

`mov ecx, x` - запись входной переменной в регистр `ecx`; `mov edx, 80` - запись размера переменной в регистр `edx`; `call sread` - вызов процедуры чтения данных;

3) Для чего используется инструкция “call atoi”?

Вызов atoi – функции преобразующей ascii-код символа в целое число и записывающий результат в регистр eax.

4) Какие строки листинга 7.4 отвечают за вычисления варианта?

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

5) В какой регистр записывается остаток от деления при выполнении инструкции “div ebx”?

В регистр ebx.

6) Для чего используется инструкция “inc edx”?

Инструкция INC используется для увеличения операнда на единицу.

7) Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений?

```
mov eax,rem call sprint mov eax,edx call iprintLF
```

4 Выводы

Во время выполнения лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

Список литературы