

Отчёт по лабораторной работе №6

Лобанова Екатерина Евгеньевна

Содержание

1	1. Цель работы	5
2	2. Порядок выполнения лабораторной работы	6
2.1	2.1 Символьные и численные данные в NASM	6
2.2	2.2 Выполнение арифметических операций в NASM	10
3	Задание для самостоятельной работы	15
4	Выводы	18

Список иллюстраций

2.1	Создание каталога и файла для лабораторной работы 6	6
2.2	Используя команду nano вводим текст листинга в файл.	7
2.3	Компилируем файл и проверяем корректность его работы.	7
2.4	Меняем программу	7
2.5	Скомпилировали файл и смотрим на результат изменения программы	8
2.6	Создаем файл использую touch и проверяем что он есть	8
2.7	Заполняем файл	8
2.8	Запускаем файл и смотрим на результат	9
2.9	Меняем программу	9
2.10	Запускаем файл	9
2.11	Замена подпрограммы	10
2.12	Запускаем файл и смотрим что изменилось	10
2.13	Используя touch создаем lab6-3.asm	10
2.14	Заполняем файл	11
2.15	Рис 15	11
2.16	Изменение кода для вычисления выражения	12
2.17	Смотрим на результат работы кода	12
2.18	Создаем новый файл	13
2.19	Редактируем файл	13
2.20	Проверяем корректность работы программы	13
3.1	Создаем файл	15
3.2	Заполняем файл	16
3.3	Подставляем x=3	17
3.4	Подставляем x=1	17

Список таблиц

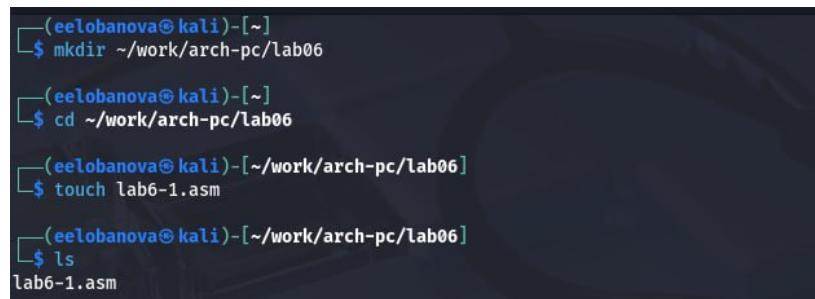
1 1. Цель работы

Освоение арифметических инструкций языка ассемблера NASM.

2 2. Порядок выполнения лабораторной работы

2.1 2.1 Символьные и численные данные в NASM

Создаем каталог для программ лабораторной работы № 6 lab06, переходим в него и создаем файл lab6-1.asm (рис. 2.1).



```
(eelobanova㉿kali)-[~]
└$ mkdir ~/work/arch-pc/lab06
(eelobanova㉿kali)-[~]
└$ cd ~/work/arch-pc/lab06
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ touch lab6-1.asm
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ ls
lab6-1.asm
```

Рисунок 2.1: Создание каталога и файла для лабораторной работы 6

Рассмотрим примеры программ вывода символьных и численных значений. Программы будут выводить значения записанные в регистр eax.

Вводим в файл lab6-1.asm текст программы из листинга 6.1. (рис. 2.2).

```
GNU nano 8.4          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
```

Рисунок 2.2: Используя команду nano вводим текст листинга в файл.

Копируем файл in_out.asm в новый каталог lab06. Создаем исполняемый файл и запускаем его, видим что результатом компиляции является символ j. (рис. 2.3).

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-1.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-1 lab6-1.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-1
j
```

Рисунок 2.3: Компелируем файл и проверяем корректность его работы.

Меняем текст программы, убрав ковычки mov eax,,6“ mov ebx,,4“ в данных строках.(рис. 2.4).

```
GNU nano 8.4          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
```

Рисунок 2.4: Меняем программу

Снова создаем исполняемый файл и запустите его, смотрим на изменения.
Выводится пустое поле. (рис. 2.5).

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-1.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-1 lab6-1.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-1

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
```

Рисунок 2.5: Скомпилировали файл и смотрим на результат изменения программы

Пользуясь таблицей ASCII определили, что код 10 соответствует символу перевода строки, поэтому он не отображается на экране.

Создаем файл lab6-2.asm в каталоге ~/work/arch-pc/lab06. (рис. 2.6).

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ touch lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2.asm
```

Рисунок 2.6: Создаем файл использую touch и проверяем что он есть

Вводим в файл lab6-2.asm текст листинга 2.(рис. 2.7).

```
GNU nano 8.4                               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
```

Рисунок 2.7: Заполняем файл

Создаем исполняемый файл и запускаем его.(рис. 2.8).

```

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nano lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-2
106

```

Рисунок 2.8: Запускаем файл и смотрим на результат

Программы выводит число 106.

Каки в прошлом случаем убираем ковычки из двух строк меняя текст программы.(рис. 2.9).

```

GNU nano 8.4                               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

```

Рисунок 2.9: Меняем программу

Создаем исполняемый файл и запускаем его.(рис. 2.10).

```

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nano lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-2
10

```

Рисунок 2.10: Запускаем файл

Результатом работы программы является число 10.

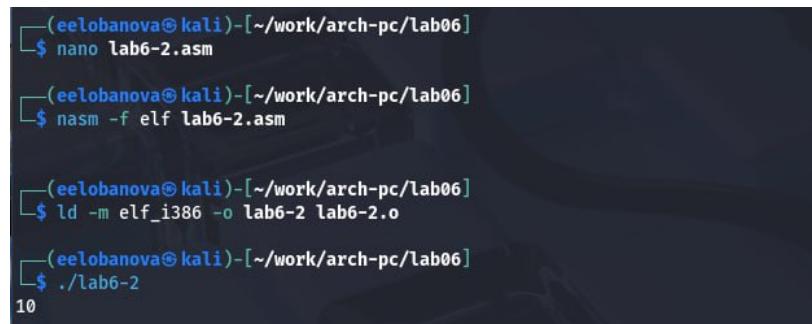
Заменяем подпрограмму iprintLF на iprint.(рис. 2.11).



```
GNU nano 8.4                               lab6-2.asm *
%include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
    mov eax,6
    mov ebx,4
    add eax,ebx
    call iprint
    call quit
```

Рисунок 2.11: Замена подпрограммы

Создаем исполняемый файл и запускаем его. (рис. 2.12)



```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nano lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-2.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-2 lab6-2.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-2
10
```

Рисунок 2.12: Запускаем файл и смотрим что изменилось

При использовании iprintLF выводимые строчки видны с новой строки, а с iprint на той же самой.

2.2 2.1 Выполнение арифметических операций в NASM

Создаем файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 (рис. 2.13)



```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ touch lab6-3.asm
```

Рисунок 2.13: Используя touch создаем lab6-3.asm

Вводим в новый файл текст листинга 3.(рис. 2.14).

```

GNU nano 8.4                               lab6-3.asm *
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
66 Демидова А. В.
Архитектура ЭВМ
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

```

Рисунок 2.14: Заполняем файл

Создаем исполняемый файл и запускаем его. (рис. 2.15)

```

[eelobanova@kali]~/work/arch-pc/lab06]$ nano lab6-3.asm
[eelobanova@kali]~/work/arch-pc/lab06]$ nasm -f elf lab6-3.asm
[eelobanova@kali]~/work/arch-pc/lab06]$ ld -m elf_i386 -o lab6-3 lab6-3.o
[eelobanova@kali]~/work/arch-pc/lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Рисунок 2.15: Рис 15

Программа работает корректно.

Вносим изменения в код, чтобы программа вычисляла выражение $\square(\square) = (4 * 6 + 2)/5$. (рис. 2.15)

```
GNU nano 8.4                               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,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/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 iprintf ; 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 iprintf ; из 'edx' (остаток) в виде символов
call quit ; вызов подпрограммы завершения
```

Рисунок 2.16: Изменение кода для вычисления выражения

Создаем исполняемый файл и проверяем работу. (рис. 2.17)

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

Рисунок 2.17: Смотрим на результат работы кода

Создаем файл variant.asm в каталоге ~/work/arch-pc/lab06 (рис. 2.18)

```

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ cd ~/work/arch-pc/lab06
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ touch ~/work/arch-pc/lab06/variant.asm
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ nano variant.asm

```

Рисунок 2.18: Создаем новый файл

Вводим текст листинга 6.4 в созданный файл.(рис. 2.19)

```

GNU nano 8.4                               variant.asm *
%include 'in_out.asm'
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 sprintLF
    mov ecx, x
    mov edx, 80
    call sread
    mov eax,x ; вызов подпрограммы преобразования
    call atoi ; ASCII кода в число, `eax=x`
    xor edx,edx
    mov ebx,20
    div ebx
    inc edx

```

Рисунок 2.19: Редактируем файл

Компилируем файл и смотрим на результат его работы. (рис. 2.20)

```

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ nano lab6-4.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ nasm -f elf lab6-4.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ ld -m elf_i386 -o lab6-4 lab6-4.o

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└$ ./lab6-4
Вычисляем: у = 18(х+1)/6
Введите х:
1
Результат у: 6

```

Рисунок 2.20: Проверяем корректность работы программы

Ответы на вопросы:

1. За вывод сообщения „Ваш вариант“ отвечают строки: `mov eax,rem call sprint`
2. Конструкции отвечают за чтение строки и ввод строки от пользователя. `mov eshx, x` - загрузка адреса буфера для сохранения вводимых данных
`mov edx, 80` - установка максимального размера вводимых данных (80 символов)
`call sread` - вызов функции ввода строки из стандартного потока ввода
3. Инструкция `call atoi` используется для преобразования строки в число, принимая адрес строки в регистре `eax` и возвращает полученное число в `eax`.
4. За вычисление вырианта отвечают следующие строки: `xor edx,edx ; Обнуление edx` для деления `mov ebx,20 ; Загрузка делителя (20) в ebx` `div ebx ; Деление (eax) / (ebx)`. Частное в `eax`, остаток в `edx`. `inc edx ; Увеличение остатка на 1`
5. При выполнении инструкции `div ebx` остаток от деления записывается в регистр `edx`
6. Инструкция `inc edx` увеличивает значение в регистре `edx` на 1.
7. Стока „`mov eax,edx`“ передает значение остатка от деления в `eax`. Стока „`call iprintLF`“ вызывает процедуру `iprintLF` для вывода значения на экран вместе с переводом строки.

3 Задание для самостоятельной работы

Создаем новый файл lab6-4.asm использую touch.(рис. 3.1)

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└─$ touch lab6-4.asm

(eelobanova㉿kali)-[~/work/arch-pc/lab06]
└─$ mc
```

Рисунок 3.1: Создаем файл

Редактируем его и вводим программу которая будет вычислять заданное выражение.(рис. 3.2)

```
GNU nano 8.4          lab6-4.asm
#include 'in_out.asm'
SECTION .data
formula: DB 'Вычисляем: у = 18(х+1)/6',0
msg: DB 'Введите х: ',0
rem: DB 'Результат у: ',0

SECTION .bss
x: RESB 80
SECTION .text
GLOBAL _start
_start:
; Показываем формулу
mov eax, formula
call sprintLF

; Запрашиваем x
mov eax, msg
call sprintLF
mov ecx, x
mov edx, 80
call sread

; Преобразуем в число
mov eax,x
call atoi

; Вычисляем у = 18(х+1)/6
add eax,1
mov ebx,18
mul ebx
mov ebx,6
xor edx,edx
div ebx
; Выводим результат
mov ebx, eax
mov eax, rem
call sprint
mov eax,ebx
call iprintLF

call quit
```

Рисунок 3.2: Заполняем файл

Запускаем программу и проверяем ее работу для $x=3$. (рис. 3.3)

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06] (#fig-023)
$ nasm -f elf lab6-4.asm
(eelobanova㉿kali)-[~/work/arch-pc/lab06] (#fig-023 width=705)
$ ld -m elf_i386 -o lab6-4 lab6-4.o
  Входная информация для компоновщика: вектора, таблицы символов и т.д.
  Входные файлы: lab6-4.o
  Выходной файл: lab6-4
  Время компоновки: 0.000 секунд
(eelobanova㉿kali)-[~/work/arch-pc/lab06] (#fig-023 width=705)
$ ./lab6-4
Вычисляем: y = 18(x+1)/6
Введите x: проверяем ее работу для x=3. (рис. #fig-023)
3
Результат y: 12
```

Рисунок 3.3: Подставляем x=3

Запускаем программу и проверяем ее работу для x=1. (рис. 3.4)

```
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nano lab6-4.asm
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ nasm -f elf lab6-4.asm
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ld -m elf_i386 -o lab6-4 lab6-4.o
(eelobanova㉿kali)-[~/work/arch-pc/lab06]
$ ./lab6-4
Вычисляем: y = 18(x+1)/6
Введите x:
1
Результат y: 6
```

Рисунок 3.4: Подставляем x=1

4 Выводы

В ходе лабораторной работы были освоены арифметические инструкции NASM и их практическое применение. Изучены операции сложения, вычитания, умножения и деления с учетом особенностей работы с регистрами. Освоено преобразование данных между символьным и числовым представлением. Разработаны программы вычисления арифметических выражений и определения варианта задания.