

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ

ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

дисциплина: Архитектура компьютера

Студент: Моисеев Дмитрий Вячеславович

Группа: НБИбд-02-25

МОСКВА

2025 г.

1. Цель работы: Освоение арифметических инструкций языка ассемблера NASM.

2. Выполнение лабораторной работы:

6.3.1. Символьные и численные данные в NASM

1. Создадим каталог для программ лабораторной работы № 6, перейдем в него и создадим файл lab6-1.asm:

```
mkdir ~/work/arch-pc/lab06
```

```
cd ~/work/arch-pc/lab06
```

touch

```
dvmoiseev@dk7n03 ~ $ mkdir ~/work/arch-pc/lab06
dvmoiseev@dk7n03 ~ $ cd ~/work/arch-pc/lab06
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ touch lab6-1.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

lab6-
1.asm

2. Рассмотрим примеры программ вывода символьных и численных значений.

Программы будут выводить значения записанные в регистр `eax`. Введем в файл lab6-1.asm текст программы из листинга 6.1. В данной программе в регистр `eax` записывается символ 6 (`mov eax,'6'`), в регистр `ebx` символ 4 (`mov ebx,'4'`).

Далее к значению в регистре `eax` прибавим значение регистра `ebx` (`add eax,ebx`, результат сложения запишется в регистр `eax`). Далее выведем результат. Так как для работы функции `sprintLF` в регистр `eax` должен быть записан адрес, необходимо использовать дополнительную переменную. Для этого запишем значение регистра `eax` в переменную `buf1` (`mov [buf1],eax`), а затем запишем адрес переменной `buf1` в регистр `eax` (`mov eax,buf1`) и вызовем функцию `sprintLF`.

Листинг 6.1. Программа вывода значения регистра `eax`

```
%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
```



```
GNU nano 8.6
#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
```

Создадим исполняемый файл и запустим его.

```
nasm -f elf lab6-1.asm
```

```
ld -m elf_i386 -o lab6-1 lab6-1.o
```

./lab6-1

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-1
j
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

3. Далее изменим текст программы и вместо символов, запишем в регистры числа.

Исправим текст программы (Листинг 6.1) следующим образом: заменим строки

```
mov eax,'6'
mov ebx,'4'
```

на строки

```
mov eax,6
mov ebx,4
```

```
GNU nano 8.6
#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
```

Создадим исполняемый файл и запустите его.

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-1

dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

4. Для работы с числами в файле `in_out.asm` реализованы подпрограммы для преобразования ASCII символов в числа и обратно. Преобразуем текст программы из Листинга 6.1 с использованием этих функций. Создадим файл `lab6-2.asm` в каталоге `~/work/arch-pc/lab06` и введем в него текст программы из листинга 6.2.

```
touch ~/work/arch-pc/lab06/lab6-2.asm
```

Листинг 6.2. Программа вывода значения регистра `eax`

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

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

```
GNU nano 8.6
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

Создадим исполняемый файл и запустим его.

```
nasm -f elf lab6-2.asm
```

```
ld -m elf_i386 -o lab6-2 lab6-2.o
```

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nano lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-2
106
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

./lab6-2

5. Аналогично предыдущему примеру изменим символы на числа. Заменяем строки

```
mov eax,'6'
```

```
mov ebx,'4'
```

на строки

```
mov eax,6
```

```
mov ebx,4
```

```
GNU nano 8.6
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Создадим исполняемый файл и запустим его.

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nano lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-2
10
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nano lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-2
10dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

Заменяем функцию `iprintLF` на `iprint`. Создадим исполняемый файл и запустим его.

6.3.2. Выполнение арифметических операций в NASM

6. В качестве примера выполнения арифметических операций в NASM приведем программу вычисления арифметического выражения $\square(\square) = (5 * 2 + 3)/3$.

Создадим файл `lab6-3.asm` в каталоге `~/work/arch-pc/lab06`:

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-3.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

`touch ~/work/arch-pc/lab06/lab6-3.asm`

Внимательно изучим текст программы из листинга 6.3 и введем в `lab6-3.asm`.

Листинг 6.3. Программа вычисления выражения $\square(\square) = (5 * 2 + 3)/3$

```

;-----
; Программа вычисления выражения
;-----

%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

```



```

GNU nano 8.6 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

```

Создадим исполняемый файл и запустим его. Результат работы программы должен быть следующим:

```
user@dk4n31:~$ ./lab6-3
```

Результат: 4

Остаток от деления: 1

```
user@dk4n31:~$
```

```

dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $

```

Изменив текст программы для вычисления выражения $\square(\square) = (4 * 6 + 2)/5$. Создадим

```

GNU nano 8.6 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/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

```

исполняемый файл и проверим его работу

```

dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nano lab6-3.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ █

```

7. В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

- вывести запрос на введение № студенческого билета
- вычислить номер варианта по формуле: $(\square\square \bmod 20) + 1$, где $\square\square$ – номер студенческого билета (В данном случае $\square \bmod \square$ – это остаток от деления \square на \square).
- вывести на экран номер варианта.

В данном случае число, над которым необходимо проводить арифметические операции, вводится с клавиатуры. Ввод с клавиатуры осуществляется в символьном виде и для корректной работы арифметических операций в NASM символы необходимо преобразовать в числа. Для этого может быть использована функция `atoi` из файла `in_out.asm`. Создадим файл `variant.asm` в каталоге `~/work/arch-pc/lab06`:

```
touch ~/work/arch-pc/lab06/variant.asm
```

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/variant.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

Внимательно изучим текст программы из листинга 6.4 и введем в файл variant.asm.

Листинг 6.4. Программа вычисления вычисления варианта задания по номеру студенческого билета

```
;-----
; Программа вычисления варианта
;-----
%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
mov eax,rem
```

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

```
GNU nano 8.6 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
mov eax, rem
call sprint
mov eax, edx
call iprintLF
call quit
```

Создадим исполняемый файл и запустим его.

```
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nano variant.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1032253845
Ваш вариант: 6
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

3. Ответы на вопросы

1. За вывод сообщения «Ваш вариант:» отвечают строки:

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

2. - `mov ecx, x` — записывает в регистр `ecx` адрес переменной `x` (буфера), куда будут помещены введенные пользователем символы.

- `mov edx, 80` — записывает в регистр `edx` максимальное количество байт (80), которое можно прочитать.

- `call sread` — вызывает подпрограмму `sread`, которая непосредственно выполняет системный вызов для чтения данных из потока ввода (клавиатуры) в указанный буфер.

3. Инструкция `call atoi` используется для преобразования строки, содержащей цифры в числовое значение, с которым компьютер может выполнять арифметические операции

4. За вычисление варианта отвечают строки:

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

5. При выполнении инструкции `div ebx` остаток от деления всегда записывается в регистр `edx`.

6. Инструкция `inc edx` увеличивает значение в регистре `edx` на единицу.

7. За вывод на экран числового результата (номера варианта) отвечают строки:

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

4. Выполнение заданий для самостоятельной работы:

1. Напишем программу вычисления выражения $\square = \square(\square)$. Программа должна

ВЫВОДИТЬ

выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. Вид функции $f(x)$ выберем из таблицы вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создадим исполняемый файл и проверим его работу для значений x_1 и x_2 из таблицы.

```
GNU nano 8.6
%include 'in_out.asm'
SECTION .data
msg: DB 'Введите x:',0
rem: DB 'Ответ:',0
SECTION .bss
x_str_buffer: RESB 80
x_val: RESD 1
SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x_str_buffer
mov edx, 80
call sread
mov eax, x_str_buffer
call atoi
mov [x_val], eax
mov eax, 12 ; EAX = 12
imul dword [x_val] ; EAX = 12*x
add eax, 3 ; EAX = EAX + 3
mov ebx, 5 ; EBX = 5
imul ebx ; EAX = EAX * EBX
mov edx, eax
mov eax, rem
call sprintf
mov eax, edx
call iprintLF
call quit
```

```
dvmois -4.asm
dvmois lf lab6-4.asm
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-4 lab6-4.o
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-4
Введите x:
2
Ответ:135
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $ ./lab6-4
Введите x:
5
Ответ:315
dvmoiseev@dk7n03 ~/work/arch-pc/lab06 $
```

Выводы:

В рамках
данной
работы
мы

приобрели практические навыки работы с арифметическими инструкциями языка

а

с

с

е

м

б

л

е

р

а

N

A

S