

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: Игнатенкова Виктория Станиславовна

Группа: НММбд-02-24

МОСКВА

2024 г.

СОДЕРЖАНИЕ

1. ЦЕЛЬ РАБОТЫ.....	3
2. ЗАДАНИЕ.....	4
3. ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ.....	5
4. ВЫВОД.....	6

1. ЦЕЛЬ РАБОТЫ

Изучение арифметических команд языка ассемблера NASM.

2. ЗАДАНИЕ

1. Изучение адресации в NASM

- Ознакомьтесь с различными методами адресации в языке ассемблера NASM и их применением.

2. Целочисленное сложение (команда add)

- Напишите программу, использующую команду add для выполнения операции целочисленного сложения.

3. Целочисленное вычитание (команда sub)

- Создайте программу, которая применяет команду sub для выполнения операции целочисленного вычитания.

4. Использование команд инкремента и декремента

- Реализуйте программу, демонстрирующую работу команд инкремента (inc) и декремента (dec).

5. Изменение знака операнда (команда neg)

- Напишите программу, использующую команду neg для изменения знака целочисленного операнда.

6. Умножение (команды mul и imul)

- Создайте программу, которая применяет команды mul и imul для выполнения целочисленного умножения.

7. Деление (команды div и idiv)

- Реализуйте программу, демонстрирующую использование команд div и idiv для выполнения целочисленного деления.

8. Перевод символа числа в десятичную символьную запись

- Напишите программу, которая преобразует символ числа в его десятичную символьную запись.

9. Порядок выполнения лабораторной работы

- Ознакомьтесь с порядком выполнения лабораторной работы и следуйте ему.

10. Самостоятельная работа

- Выполните задание для самостоятельной работы, связанное с изученными темами, и подготовьте отчет о выполнении.

3. ВЫПОЛНЕНИЕ ЛАБОРАТОРНОЙ РАБОТЫ

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

```
vsignatenkova@dk3n55 ~ $ mkdir ~/work/arch-pc/lab06
vsignatenkova@dk3n55 ~ $ cd ~/work/arch-pc/lab06
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-1.asm
```

Рис.3.1. Создание файла

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ls
lab6-1.asm
```

Рис.3.2. Проверка

Создаем исполняемый файл и запускаем его:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис.3.3. Файл

Изменяем информацию в файле(заменяем '6' и '4' на 6 и 4) и заново запускаем его:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-1
```

Рис.3.4. Измененный файл

Создаем файл lab6-2.asm в каталоге ~/work/arch-pc/lab06 и вводим в него текст программы из листинга 6.2:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-2.asm
```

Рис.3.5. Второй файл

Запускаем полученный файл:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
106
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.3.6. Запуска второго файла

Вносим аналогичные изменения во втором файле(убираем кавычки):

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
10
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.3.7. Вывод второго файла

Заменяем функцию `iprintLF` на `iprint`. Можем заменить, что при выводе изменилось расположение ответа:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-2
10vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.3.8. Изменения

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

```
10vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm lab6-1 lab6-1.asm lab6-1.o lab6-2 lab6-2.asm lab6-2.o lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.3.9. Создание третьего файла

Изучаем текст программы из листинга 6.3 и вводим в `lab6-3.asm`:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.3.10. Вывод программы

Заменяем первое уравнение на второе и заново запускаем файл:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
```

Рис.3.11. Вывод ответа на второе уравнение

Создаем новый файл:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ls
in_out.asm lab6-1 lab6-1.asm lab6-1.o lab6-2 lab6-2.asm lab6-2.o lab6-3 lab6-3.asm lab6-3.o variant.asm
```

Рис.3.12. Создание файла

Вводим текст и запускаем программу. После вводим номер студенческого билета:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132246774
Ваш вариант: 15
```

Рис.3.13. Вывод

Ответы на вопросы:

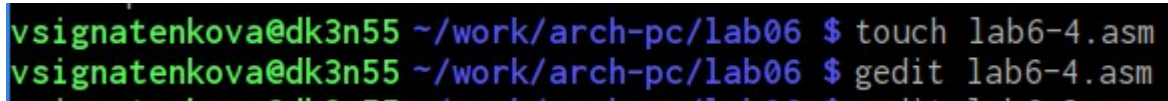
1. Вывод сообщения "Ваш вариант" осуществляется с помощью следующих строк кода: `mov eax, rem` `call sprint`.
2. Инструкция `mov esx, x` используется для загрузки адреса вводимой строки `x` в регистр `esx`. Инструкция `mov edx, 80` записывает в регистр `edx` длину вводимой строки.
3. `call sread` — это вызов подпрограммы из внешнего файла, отвечающей за ввод сообщения с клавиатуры. `call atoi` применяется для вызова подпрограммы из внешнего файла, которая преобразует ASCII-код символа в целое число и сохраняет результат в регистре `eax`.
4. Вычисления варианта выполняются с помощью следующих строк кода: “ `xor edx, edx` ; обнуляем `edx` для правильной работы `div` `mov ebx, 20` ; присваиваем `ebx` значение 20 `div ebx` ; выполняем деление `eax` на 20, остаток помещается в `edx` `inc edx` ; увеличиваем `edx` на 1 “

5. Во время выполнения инструкции `div ebx` остаток от деления сохраняется в регистре `edx`.
6. Инструкция `inc edx` увеличивает значение регистра `edx` на единицу.
7. Вывод результатов вычислений на экран осуществляется с помощью следующих строк кода: “ `mov eax, edx call iprintLF` “

САМОСТОЯТЕЛЬНАЯ РАБОТА

Мне выпал 15 вариант. Условие: $(5 + x)^2 - 3$, $x_1=5$, $x_2=1$

Создаем файл:



```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ touch lab6-4.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ gedit lab6-4.asm
```

Рис.1. Создание файла

Пишем программу для уравнения:

```
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
10 ; ---- Вычисление выражения
11 mov eax, msg ; запись адреса выводимого сообщения в eax
12 call sprint ; вызов подпрограммы печати сообщения
13 mov ecx, x ; запись адреса переменной в ecx
14 mov edx, 80 ; запись длины вводимого значения в edx
15 call sread ; вызов подпрограммы ввода сообщения
16 mov eax,x ; вызов подпрограммы преобразования
17 call atoi ; ASCII кода в число, 'eax=x'
18 add eax,5; eax = eax+5 = x + 5
19 mov ebx,eax ; запись значения eax в регистр ebx
20 mul ebx; EAX=EAX*EBX = (x+5)**2
21 add eax,-3; eax = eax-3 = (x+5)**2-3
22 mov edi,eax ; запись результата вычисления в 'edi'
23 ; ---- Вывод результата на экран
24 mov eax,rem ; вызов подпрограммы печати
25 call sprint ; сообщения 'Результат: '
26 mov eax,edi ; вызов подпрограммы печати значения
27 call iprint ; из 'edi' в виде символов
28 call quit ; вызов подпрограммы завершения
```

Рис.2. Программа уравнения

Подставляем первое значение x:

```
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-4.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-4 lab6-4.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-4
Введите значение переменной x: 5
Результат: 97vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-4.asm
```

Рис.3. Первое решение

Подставляем второе значение x:

```
Результат: 97vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ nasm -f elf lab6-4.asm
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-4 lab6-4.o
vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $ ./lab6-4
Введите значение переменной x: 1
Результат: 33vsignatenkova@dk3n55 ~/work/arch-pc/lab06 $
```

Рис.4. Второе решение

4. ВЫВОД

Освоение арифметических команд языка ассемблера NASM позволит эффективно выполнять математические операции на низком уровне, что является основой для разработки программ и оптимизации алгоритмов в системном программировании.