РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ № 6

дисциплина: Архитектура компьютера

Освоение арифметических инструкций языка ассемблера NASM

Студент: ТУЙИШИМЕ Тьерри

Группа: НКАбд-05-25

Оглавление

1. Цель работы

Освоение арифметических инструкций языка ассемблера NASM. Изучение принципов работы с символьными и численными данными, выполнение базовых арифметических операций и преобразований между различными форматами данных.

2. Задание

1. Изучить символьные и численные данные в NASM, включая различия в их обработке.

2. Выполнить арифметические операции в NASM с использованием различных типов данных.

3. Реализовать программу для вычисления арифметического выражения в рамках задания для самостоятельной работы.

4. Изучить принципы ввода-вывода данных и их преобразования между ASCII и числовым представлением.

3. Теоретическое введение

Ассемблер NASM предоставляет низкоуровневый доступ к процессору и памяти, что требует глубокого понимания способов адресации и обработки данных.

Основные способы адресации:

Регистровая адресация: операнды хранятся в регистрах процессора (например, ‘mov ax, bx‘). Это наиболее быстрый способ обработки данных.

Непосредственная адресация :значение операнда задается непосредственно в команде (например, ‘mov ax, 2‘). Используется для константных значений.

Адресация памяти: операнд указывает на адрес в памяти. Требует больше времени для выполнения, но позволяет работать с большими объемами данных.

Особенности работы с данными в NASM:

Ввод и вывод информации в NASM осуществляются в символьном виде с использованием кодировки ASCII. Это создает необходимость преобразования данных между символьным и числовым представлениями.

Критически важные моменты

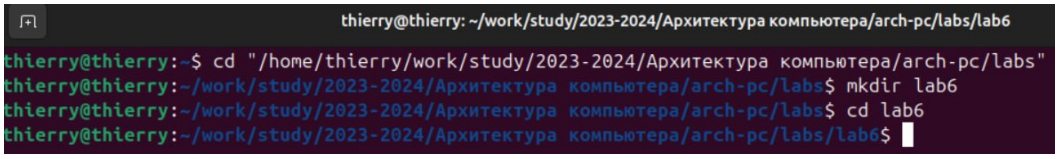
- При выводе числа без преобразования оно интерпретируется как последовательность ASCII-символов
- Для корректных арифметических операций необходимо преобразовывать введенные символы в числа
- Функции вроде ‘atoi‘ (ASCII to integer) и ‘iprint‘ (integer print) обеспечивают это преобразование

4. Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

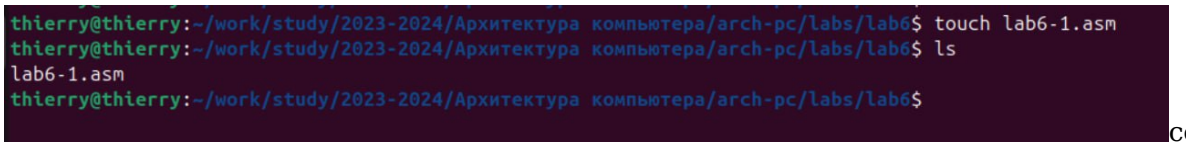
Шаг 1: Создание рабочей среды

1. Создайте директорию для лабораторной работы:



создание директории и переход в неё.

2. Создайте файл ‘lab6-1.asm‘:



создани файла и проверка его наличия.

3. Скопируйте файл ‘in_out.asm‘ в текущую директорию:

копирование файла.

Шаг 2: Демонстрация различий символьных и численных данных

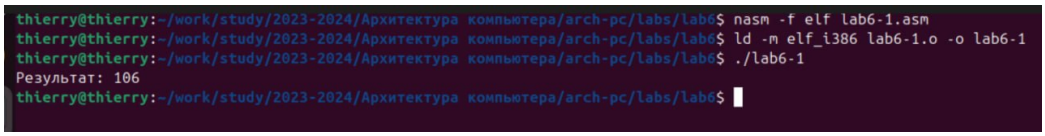
1. Откройте файл ‘lab6-1.asm‘ и введите следующий код:

Open • lab6-1.asm
~/work/arch-pc/lab06

```
%include 'in_out.asm'
SECTION .data
msg db "Result: ",0
SECTION .text
global _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call sprintLF
call quit
```

редактирование файла.

2. Создайте исполняемый файл и запустите его:



запуск программы и вывод символа 'j'.

ASCII-код 106 соответствует символу 'j'

3. Измените символы на числа в файле ‘lab6-1.asm‘:

lab6-1.asm

```
mov edi, eax

; Print "Result: "
mov eax, msg
call sprint

; Print the result character
mov eax, edi
call sprintLF

; Exit properly
mov eax,6
mov ebx,4
add eax,ebx
```

изменение кода.

4. Создайте и запустите новый исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-1.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-1.o -o lab6-1
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-1
Результат: 106
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

Сделайте скриншот: вывод пустой строки (символ перевода строки с кодом 10).

Шаг 3: Программа с корректным выводом чисел

1. Создайте файл ‘lab6-2.asm‘:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ touch lab6-2.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

создание файла.

2. Введите в файл следующий код:

Open

lab6-2.asm

~/work/study/2023-2024/Архитектура компьютера/arch

```
%include 'in_out.asm'
SECTION .data
msg db "Result: ",0
SECTION .text
global _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call iprintLF
call quit
```

редактирование файла.

3. Создайте и запустите исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-2.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-2.o -o lab6-2
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-2
Result: 106
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

вывод числа 106.

4. Замените символы на числа в файле ‘lab6-2.asm‘:

Open • lab6-2.asm
~/work/study/2023-2024/Архитектура компьютера/

```
%include 'in_out.asm'
SECTION .data
msg db "Result: ",0
SECTION .text
global _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call iprintLF
call quit
```

изменение кода.

5. Создайте и запустите новый исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-2.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-2.o -o lab6-2
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-2
Result: 10
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

вывод числа 10.

6. Замените ‘iprintLF‘ на ‘iprint‘ в файле ‘lab6-2.asm‘:

Open • lab6-2.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6

```
mov ebx,4
add eax,ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call call iprint
call quit
```

изменение кода.

7. Создайте и запустите новый исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-2.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-2.o -o lab6-2
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-2
Result: 10
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

вывод числа 10 без перевода строки.

4.2 Выполнение арифметических операций в NASM

Шаг 1: Программа вычисления арифметического выражения

1. Создайте файл ‘lab6-3.asm‘:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ touch lab6-3.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

создание файла.

2. Введите в файл следующий код для вычисления f(x) = (5 * 2 + 3)/3:

Open — lab6-3.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6

```
%include 'in_out.asm'
SECTION .data
msg db "Result: ",0
SECTION .text
global _start
_start:
mov eax,5
mov ebx,2
mul ebx
add eax,3
mov ebx,3
div ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call iprintLF
call quit
```

редактирование файла.

3. Создайте и запустите исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-3.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-3.o -o lab6-3
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-3
Result: 4
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

вывод результата вычисления (4).

4. Измените программу для вычисления выражения ‘ (4 * 6 + 2) / 5‘:

Open — lab6-3.asm — ~/work/study/2023-2024/Архитектура компьютера/arch-pc/lab

```
%include 'in_out.asm'
SECTION .data
msg db "Result: ",0
SECTION .text
global _start
_start:
mov eax,4
mov ebx,6
mul ebx
add eax,2
mov ebx,5
div ebx
mov edi,eax
mov eax,msg
call sprint
mov eax,edi
call iprintLF
call quit
```

изменение кода.

5. Создайте и запустите новый исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-3.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 lab6-3.o -o lab6-3
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-3
Result: 5
```

вывод результата вычисления (5).

Программа вычисления варианта задания

1. Создайте файл ‘variant.asm‘:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ touch variant.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

создание файла.

2. Введите в файл следующий код:

Open • variant.asm
~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6

```
%include 'in_out.asm'
SECTION .data
msg db "Enter your student ID number: ",0
rem db "Your variant: ",0
SECTION .bss
x resb 10
SECTION .text
global _start
_start:
mov eax,msg
call sprint
mov ecx,x
mov edx,10
call sread
mov eax,x
call atoi
xor edx,edx
mov ebx,20
div ebx
inc edx
mov eax,rem
call sprint
mov eax,edx
call iprintLF
```

редактирование файла.

3. Создайте и запустите исполняемый файл:

```
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf variant.asm
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 variant.o -o variant
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$ ./variant
Enter your student ID number:1132255025
Your variant: 6
thierry@thierry:~/work/study/2023-2024/Архитектура компьютера/arch-pc/labs/lab6$
```

ввод номера студенческого билета и вывод варианта.

4.3 Вопросы по программе

Анализ программы variant.asm:

1. Вывод сообщения ”Ваш вариант”
   - **mov eax, rem**
   - **call sprint**

использование функции sprint для вывода строки*

2, Организация ввода данных:

   - mov ecx, x – загрузка адреса буфера для вводимой строки
   - mov edx, 80 – установка максимальной длины ввода
   - call sread – вызов функции чтения строки

3.Преобразование данных:

   - call atoi – преобразует ASCII-строку в целое число

4. Вычисление варианта:
   - xor edx,edx
   - mov ebx,20
   - div ebx
   - inc edx

алгоритм равномерного распределения вариантов

5. Особенности операции div:
   - При ‘div ebx‘ остаток сохраняется в регистре edx

6. Инструкция inc:
   - ‘inc edx‘ – увеличивает значение регистра на 1

7. Вывод результата:
   - mov eax,edx
   - call iprintLF

4.4 Задание для самостоятельной работы

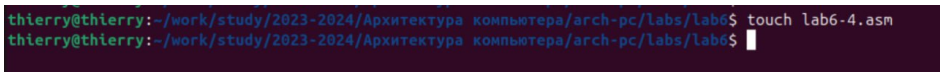
Шаг 1: Создание программы для вычисления выражения $5(x - 1)^2$

1. Создайте файл ‘lab6-4.asm‘:

создание файла.

создание программы для индивидуального задания*

2. Введите в файл следующий код:



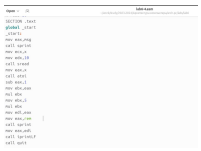
редактирование файла.

3. Создайте и запустите исполняемый файл:

ввод значения 3 и вывод результата 20.

проверка корректности вычислений (5*(3-1)² = 5*4 = 20)

4. Проверьте программу с другим значением:



ввод значения 5 и вывод результата 80.

дополнительная проверка (5*(5-1)² = 5*16 = 80)

5. Выводы

В ходе выполнения лабораторной работы были достигнуты следующие результаты:

1. Освоены арифметические инструкции: языка ассемблера NASM, включая сложение, вычитание, умножение и деление.

2. Изучены различия: между символьными и численными данными, а также принципы их преобразования с помощью функций ‘atoi‘ и ‘iprint‘.

3. Реализованы программы: различной сложности - от простых арифметических операций до интерактивных приложений с пользовательским вводом.

4. Освоены принципы работы: с функциями ввода-вывода из внешнего файла ‘in_out.asm‘.

5.Получен практический опыт: отладки ассемблерных программ и анализа результатов выполнения.

Работа продемонстрировала важность понимания низкоуровневых операций для эффективного программирования на ассемблере и необходимость тщательного контроля за форматом данных при выполнении арифметических операций.

6. Список литературы

1. Лабораторная работа №6 - Методические указания
2. NASM Documentation - Official NASM Manual
3. Assembly Language Programming - Fundamentals and Techniques