

РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ
Факультет физико-математических и естественных наук
Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ №7
дисциплина: Архитектура компьютеров и операционные системы

Студент: Галиев Самир Салаватович

Группа: НКАбд-02-25

МОСКВА

2025 г.

Оглавление

1. Цель работы — стр. 3
2. Результаты выполнения лабораторной работы — стр. 3
 - 2.1 Программа с безусловными переходами — стр. 3
 - 2.2 Программа с условными переходами — стр. 4
 - 2.3 Работа с файлом листинга — стр. 5
3. Задание для самостоятельной работы — стр. 7
 - 3.1 Программа нахождения наименьшего из трех чисел — стр. 7
 - 3.2 Программа вычисления функции $f(x)$ для варианта 11 — стр. 8
4. Выводы - стр. 10
5. Список литературы — стр. 11

1. Цель работы

Изучение команд условного и безусловного переходов. Приобретение навыков написания программ с использованием переходов. Знакомство с назначением и структурой файла листинга.

2. Результаты выполнения лабораторной работы

2.1 Программа с безусловными переходами

Задание: Изучение работы инструкции `jmp` для организации безусловных переходов между различными участками программы.

Выполнение:

Результат работы программы

```
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nasm -f elf lab7-1.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ld -m elf_i386 lab7-1.o -o lab7-1
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-1
Сообщение № 2
Сообщение № 3
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % █

ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nasm -f elf lab7-1.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ld -m elf_i386 lab7-1.o -o lab7-1
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-1
Сообщение № 1
Сообщение № 3
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % █

ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nasm -f elf lab7-1.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ld -m elf_i386 lab7-1.o -o lab7-1
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

Рисунок 1, 2 и 3: Программа выводит сообщения в порядке: №3, №2, №1, что подтверждает корректную работу безусловных переходов. Управление передается между метками с помощью инструкций `jmp`.

Комментарии: Программа содержит три метки с выводом различных сообщений. Инструкция `jmp _label3` в начале обеспечивает переход к выводу третьего сообщения, пропуская первые два.

2.2 Программа с условными переходами

Задание: Освоение команд условных переходов на примере программы поиска наибольшего из трех чисел.

Выполнение:

```
%include 'in_out.asm'

section .data
msg1 db 'Введите B: ',0h
msg2 db "Наибольшее число: ",0h
A dd '20'
C dd '50'

section .bss
max resb 10
B resb 10

section .text
global _start
_start:
    mov eax, msg1
    call sprint

    mov ecx, B
    edx, 10
    call sread

    mov eax, B
    call atoi
    mov [B], eax

    mov ecx, [A]
    mov [max], ecx

    cmp ecx, [C]
    jg check_B
    mov ecx, [C]
    mov [max], ecx

check_B:
    mov eax, max
    call atoi
    mov [max], eax

    mov ecx, [max]
    cmp ecx, [B]
    jg fin
    mov ecx, [B]
    mov [max], ecx

fin:
    mov eax, msg2
    call sprint
    mov eax, [max]
    call iprintLF
    call quit
```

```
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-2
Введите B: 30
Наибольшее число: 50
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-2
Введите B: 60
Наибольшее число: 60
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 %
```

Рисунок 5, 6: При $V=30$ наибольшее число 50 (значение C), при $V=60$ наибольшее число 60 (значение B). Программа корректно определяет наибольшее число среди трех переменных.

2.3 Работа с файлом листинга

Задание: Изучение структуры и назначения файла листинга, создаваемого транслятором NASM.

Выполнение:

1		<u>section .data</u>
2	00000000 11000000	a dd 17
3	00000004 17000000	b dd 23
4	00000008 <u>2D000000</u>	c dd 45
5		
6		<u>section .text</u>
7		<u>global _start</u>
8		<u>_start:</u>
9	00000000 <u>A1[00000000]</u>	mov eax, [a]
10	00000005 <u>8B1D[04000000]</u>	mov ebx, [b]
11	<u>0000000B 8B0D[08000000]</u>	mov ecx, [c]
12		
13	00000011 <u>39D8</u>	cmp eax, ebx
14	00000013 <u>7C02</u>	jl check_c
15	00000015 <u>89D8</u>	mov eax, ebx
16		
17		<u>check_c:</u>
18	00000017 <u>39C8</u>	cmp eax, ecx
19	00000019 <u>7C02</u>	jl print
20	<u>0000001B 89C8</u>	mov eax, ecx
21		
22		<u>print:</u>
23		call <u>iprintLF</u>
23	*****	error: symbol <u>`iprintLF'</u> not defined
24		call <u>quit</u>
24	*****	error: symbol <u>`quit'</u> not defined

Рисунок 7: Файл листинга содержит номера строк, смещения в сегменте, машинный код и исходные инструкции. Видны адреса инструкций и их шестнадцатеричное представление.

```
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nano lab7-2.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nasm -f elf -l lab7-2.lst lab7-2.asm
lab7-2.asm:20: error: label, instruction or prefix expected at start of line, found `edx'
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % █
```

Рисунок 8: При наличии синтаксических ошибок транслятор выводит сообщения об ошибках и не создает файл листинга, что демонстрирует важность корректного синтаксиса.

3. Задание для самостоятельной работы

3.1 Программа нахождения наименьшего из трех чисел

Задание: Написать программу нахождения наименьшей из 3 целочисленных переменных a, b и c для варианта 11.

Выполнение:

```
%include 'in_out.asm'      ; Подключение внешнего файла с функциями

section .data
a dd 21
b dd 28
c dd 34

section .text
global _start
_start:
    mov eax, [a]
    mov ebx, [b]
    mov ecx, [c]

    ; Сравниваем a и b
    cmp eax, ebx
    jl check_c
    mov eax, ebx

check_c:
    ; Сравниваем текущий минимум (eax) с c
    cmp eax, ecx
    jl print
    mov eax, ecx

print:
    call iprintLF      ; Вывод числа
    call quit          ; Завершение программы
```

Рисунок 8: Для варианта 11 используются значения $a=21$, $b=28$, $c=34$. Программа последовательно сравнивает значения и выбирает минимальное с помощью условных переходов `jl`.

```

ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nano lab7-3.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % nasm -f elf lab7-3.asm
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ld -m elf_i386 lab7-3.o -o lab7-3
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-3
21
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % █

```

Рисунок 9: Результат работы программы.

3.2 Программа вычисления функции $f(x)$ для варианта 11

Задание: Написать программу вычисления значения функции $f(x)$ для введенных с клавиатуры значений x и a .

Выполнение:

```

section .data
msg_x db "Введите x: ",0
msg_a db "Введите a: ",0
msg_r db "Результат: ",0

section .bss
x resb 10
a resb 10

section .text
global _start
_start:
    ; Ввод x
    mov eax, msg_x
    call sprintf
    mov ecx, x
    mov edx, 10
    call sread
    mov eax, x
    call atoi
    mov [x], eax

    ; Ввод a
    mov eax, msg_a
    call sprintf
    mov ecx, a
    mov edx, 10
    call sread
    mov eax, a
    call atoi
    mov [a], eax

    ; Проверка условия
    mov ebx, [x]
    mov ecx, [a]

    ; Сравниваем x с 0
    cmp ebx, 0
    jne else_part    ; если x ≠ 0

    ; если x = 0: f(x) = 4a
    mov eax, ecx     ; eax = a
    shl eax, 2       ; eax = 4a (умножение на 4)
    jmp print_result

else_part:
    ; если x ≠ 0: f(x) = 4a + x
    mov eax, ecx     ; eax = a
    shl eax, 2       ; eax = 4a
    add eax, ebx     ; eax = 4a + x

print_result:
    mov ebx, eax     ; сохраняем результат
    mov eax, msg_r
    call sprintf
    mov eax, ebx
    call iprintf
    call quit

```

Рисунок 10: Исходный код программы lab7-4.asm

```

ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ld -m elf_i386 lab7-4.o -o lab7-4
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-4
Введите x: 0
Введите a: 3
Результат: 12
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % ./lab7-4
Введите x: 1
Введите a: 2
Результат: 9
ssgaliev@linux ~/work/study/2025-2026/Архитектура компьютера/arch-pc/lab07 % █

```

Рисунок 11: При $x=0$ программа вычисляет $f(x)=4a=12$. Умножение на 4 реализовано через сдвиг `shl eax,2`, что эквивалентно умножению на 4. При $x=0$ программа вычисляет $f(x)=4a=12$. Умножение на 4 реализовано через сдвиг `shl eax,2`, что эквивалентно умножению на 4.

4. Выводы

В ходе выполнения лабораторной работы были достигнуты следующие результаты:

Освоены команды переходов - Практически изучены безусловные переходы (jmp) и условные переходы (je, jne, jg, jl, jge и др.), их синтаксис и применение.

Изучена работа с флагами - Освоено влияние арифметических операций на регистр флагов и использование флагов для организации условных переходов.

Реализованы программы с ветвлениями - Созданы и протестированы программы с использованием переходов для реализации различных алгоритмов.

Освоена работа с файлом листинга - Изучена структура файла листинга, его создание и анализ для отладки программ.

Реализованы сложные алгоритмы - Написаны программы для поиска экстремальных значений и вычисления кусочно-заданных функций с обработкой пользовательского ввода.

Выполнено задание для самостоятельной работы - Созданы и протестированы программы для нахождения наименьшего числа и вычисления функции $f(x)$ для варианта 11.

Цель работы достигнута полностью: Изучены механизмы ветвления в ассемблере NASM и успешно применены на практике для решения различных задач. Полученные навыки позволяют создавать сложные программы с нелинейной структурой выполнения.

5. Список литературы

- 1) NASM Assembly Language Tutorials — 2021. — URL: <https://asmtutor.com/>
- 2) The NASM documentation — 2021. — URL: <https://www.nasm.us/docs.php>
- 3) Расширенный ассемблер: NASM — 2021. — URL: <https://www.opennet.ru/docs/RUS/nasm/>
- 4) Столяров А. Программирование на языке ассемблера NASM для ОС Unix. — 2-е изд. — М. : МАКС Пресс, 2011. — URL: http://www.stolyarov.info/books/asm_unix
- 5) Колдаев В. Д., Lupin С. А. Архитектура ЭВМ. — М. : Форум, 2018.