РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 8

дисциплина: Архитектура компьютера

Студент: Ким Илья 1032253496

Группа: НКАбд-03-25

МОСКВА

2025 г.

Содержание

Список иллюстраций

1. Цель работы

Приобретение навыков написания программ с использованием циклов и обработкой аргументов командной строки.

2. Задание

1. Реализация циклом в NASM
2. Обработка аргументов командной строки
3. Самостоятельное написание программы по материалам лабораторной работы

3. Теоретическое введение

Стек — это структура данных, организованная по принципу LIFO («Last In — First Out» или «последним пришёл — первым ушёл»). Стек является частью архитектуры процессора и реализован на аппаратном уровне. Для работы со стеком в процессоре есть специальные регистры (ss, bp, sp) и команды. Основной функцией стека является функция сохранения адресов возврата и передачи аргументов при вызове процедур. Кроме того, в нём выделяется память для локальных переменных и могут временно храниться значения регистров.

4. Выполнение лабораторной работы

4.1 Реализация циклов в NASM

Создаю каталог для программ лабораторной работы №8 (рис. 4.1).

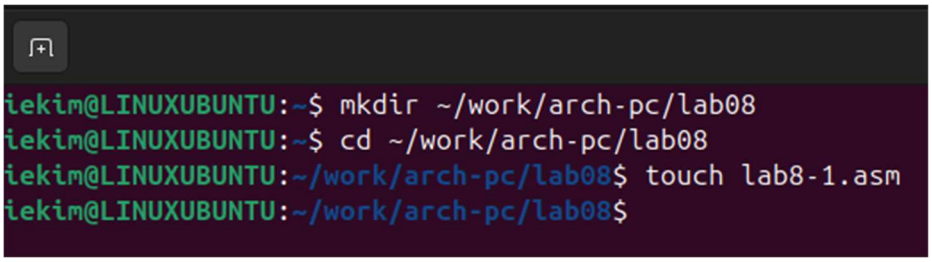

Рис. 4.1: Создание каталога

Копирую в созданный файл программу из листинга. (рис. 4.2).

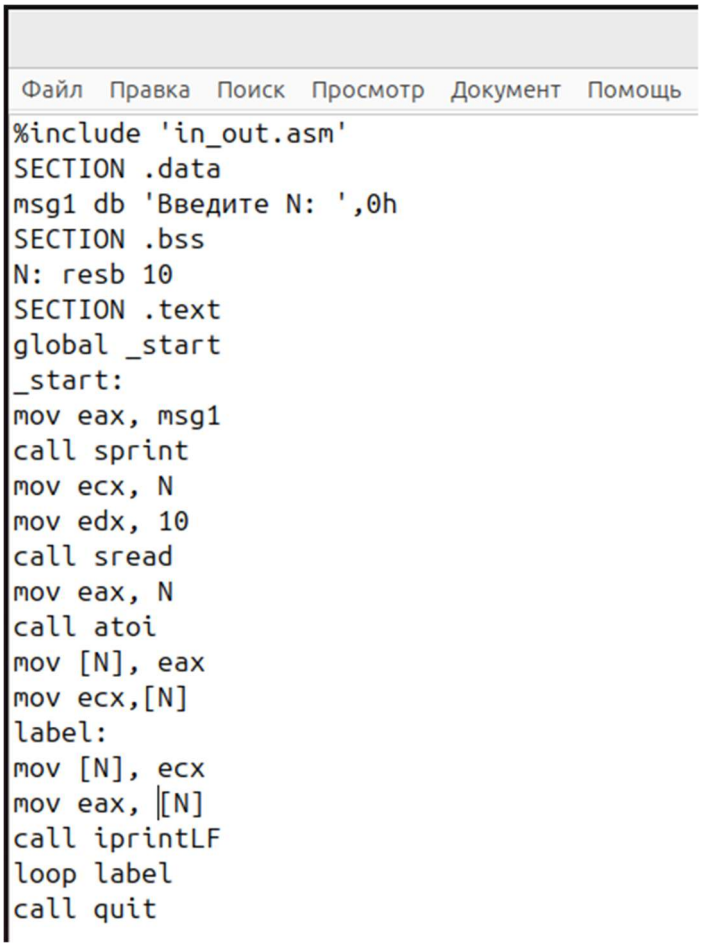

Рис. 4.2: Копирование программы из листинга

Запускаю программу, она показывает работу циклов в NASM (рис. 4.3).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ mousepad lab8-1.asm

(mousepad:5757): GLib-CRITICAL **: 07:41:33.547: g_strjoinv: assertion 'str_array != NULL' failed

(mousepad:5757): GLib-CRITICAL **: 07:41:33.547: g_strjoinv: assertion 'str_array != NULL' failed
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-1 lab8-1.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-1
```

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 20
20
19
18
17
16
15
14
13
12
11
10
9
8
7
6
5
4
3
2
1
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$
```

Рис. 4.3: Запуск программы

Заменяю программу изначальную так, что в теле цикла я изменяю значение регистра ecx (рис. 4.4).

```
Файл  Правка  Поиск  Просмотр  Документ  Помощь
%include 'in_out.asm'
SECTION .data
msg1 db 'Введите N: ',0h
SECTION .bss
N: resb 10
SECTION .text
global _start
_start:
mov eax, msg1
call sprint
mov ecx, N
mov edx, 10
call sread
mov eax, N
call atoi
mov [N], eax
mov ecx, [N]
label:
sub ecx, 1
mov [N], ecx
mov eax, [N]
call iprintLF
loop label
call quit
```

Рис. 4.4: Изменение программы

Из-за того, что теперь регистр ecx на каждой итерации уменьшается на 2 значения, количество итераций уменьшается вдвое (рис. 4.5).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ mousepad lab8-1.asm

(mousepad:6020): GLib-CRITICAL **: 07:47:33.650: g_strjoinv: assertion 'str_array != NULL' failed

(mousepad:6020): GLib-CRITICAL **: 07:47:33.652: g_strjoinv: assertion 'str_array != NULL' failed
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-1 lab8-1.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
7
5
3
1
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$
```

Рис. 4.5: Запуск измененной программы

Добавляю команды push и pop в программу (рис. 4.6).

Файл Правка Поиск Просмотр Документ Помощь

```
%include 'in_out.asm'

SECTION .data
msg1 db 'Введите N: ',0h

SECTION .bss
N: resb 10

SECTION .text
global _start
_start:

mov eax, msg1
call sprint

mov ecx, N
mov edx, 10
call sread

mov eax, N
call atoi
mov [N], eax

mov ecx, [N]

label:
push eax
sub ecx, 1
mov [N], ecx
mov eax, [N]
call iprintLF
pop ecx
loop label

call quit
```

Рис. 4.6: Добавление push и pop в цикл программы

Теперь количество итераций совпадает введенному N, но произошло смещение выводимых чисел на -1 (рис. 4.7).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-1.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-1 lab8-1.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-1
Введите N: 10
9
8
7
6
5
4
3
2
1
0
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$
```

Рис. 4.7: Запуск измененной программы

4.2 Обработка аргументов командной строки

Создаю новый файл для программы и копирую в него код из следующего листинга (рис. 4.8).

```
Файл  Правка  Поиск  Прос
%include 'in_out.asm'

SECTION .text
GLOBAL _start

_start:
pop ecx
pop edx
sub ecx, 1

next:
cmp ecx, 0
jz _end
pop eax
call sprintLF
loop next

_end:
call quit
```

Рис. 4.8: Копирование программы из листинга

Компилирую программу и запускаю, указав аргументы. Программой было обработано то же количество аргументов, что и было введено (рис. 4.9).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-2.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-2 lab8-2.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-2 arg1 arg 2 'arg 3'
arg1
arg
2
arg 3
```

Рис. 4.9: Запуск второй программы

Создаю новый файл для программы и копирую в него код из третьего листинга (рис. 4.10).

```
%include 'in_out.asm'
 SECTION .data
 msg db "Результат: ", 0
 SECTION .text
 global _start
 _start:
pop ecx
 pop edx
 sub ecx, 1
 mov esi, 0
 next:
 cmp ecx, 0h
 jz _end
 pop eax
 call atoi
 add esi, eax
 loop next
 _end:
 mov eax, msg
 call sprint
 mov eax, esi
 call iprintLF
 call quit
```

Рис. 4.10: Копирование программы из третьего листинга

Компилирую программу и запускаю, указав в качестве аргументов некоторые числа, программа их складывает (рис. 4.11).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-3 lab8-3.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-3 12 13 7 10 5
Результат: 47
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$
```

Рис. 4.11: Запуск третьей программы

Изменяю поведение программы так, чтобы указанные аргументы она умножала, а не складывала (рис. 4.12).

```
%include 'in_out.asm'
 SECTION .data
 msg db "Результат: ", 0
 SECTION .text
 global _start
 _start:
pop ecx
 pop edx
 sub ecx, 1
 mov esi, 1
 next:
 cmp ecx, 0h
 jz _end
 pop eax
 call atoi
 mul esi
mov esi, eax
 loop next
 _end:
 mov eax, msg
 call sprint
 mov eax, esi
 call iprintLF
 call quit
```

Рис. 4.12: Изменение третьей программы

Программа действительно теперь умножает данные на вход числа (рис. 4.13).

```
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ nasm -f elf lab8-3.asm
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ld -m elf_i386 -o  lab8-3 lab8-3.o
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$ ./lab8-3 111 1 7
Результат: 777
iekim@LINUXUBUNTU:~/work/arch-pc/lab08$
```

Рис. 4.13: Запуск измененной третьей программы

4.3 Задание для самостоятельной работы

Пишу программу, которая будет находить сумма значений для функции f(x) = 10x-4, которая совпадает с моим девятым вариантом (рис. 4.14).

```
%include 'in_out.asm'
SECTION .data
msg_func db "Функция: f(x) = 10x - 4", 0
msg_result db "Результат: ", 0
SECTION .text
GLOBAL _start
_start:
mov eax, msg_func
call sprintLF
pop ecx
pop edx
sub ecx, 1
mov esi, 0
next:
cmp ecx, 0h
jz _end
pop eax
call atoi
mov ebx, 10
mul ebx
sub eax, 4
add esi, eax
loop next
_end:
mov eax, msg_result
call sprint
mov eax, esi
call iprintLF
call quit
```

Рис. 4.14: Написание программы для самостоятельной работы

Код программы:

```
%include 'in_out.asm'
SECTION .data
msg_func db "Функция: f(x) = 10x - 4", 0
msg_result db "Результат: ", 0
SECTION .text
GLOBAL _start
_start:
mov eax, msg_func
call sprintLF
pop ecx
pop edx
sub ecx, 1
mov esi, 0
next:
cmp ecx, 0h
jz _end
pop eax
call atoi
mov ebx, 10
mul ebx
sub eax, 4
add esi, eax
loop next
_end:
mov eax, msg_result
call sprint
mov eax, esi
call iprintLF
call quit
```

Проверяю работу программы, указав в качестве аргумента несколько чисел (рис. 4.15).

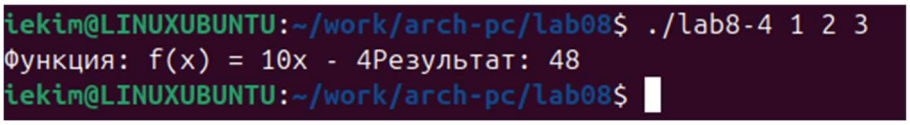

Рис. 4.15: Запуск программы для самостоятельной работы

5. Выводы

В результате выполнения данной лабораторной работы я приобрел навыки написания программ с использованием циклов, а также научился обрабатывать аргументы командной строки.

6. Список литературы

1. Курс на ТУИС
2. Лабораторная работа №8
3. Программирование на языке ассемблера NASM Столяров А. В.