Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Никуленков Степан Сергеевич

Содержание

1 Цель работы **5**

2 Задание **6**

3 Теоретическое введение **7**

4 Выполнение лабораторной работы **8**
4.1 Реализация переходов в NASM 8
4.2 Изучение структуры файла листинга 9
4.3 Задания для самостоятельной работы.Вариант 5 10

5 Выводы **17**

Список литературы **18**

Список иллюстраций

4.1 Создание каталога и файла для программы 8
4.2 Копирование 8
4.3 Изменение программы 8
4.4 Изменение программы 9
4.5 Проверка программы из листинга 9
4.6 Проверка файла листинга 10
4.7 Просмотр ошибки в файле листинга 10
4.8 Проверка работы первой программы 13
4.9 Проверка работы второй программы 16

Список таблиц

1 Цель работы

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

2 Задание

1. Реализация переходов в NASM
2. Изучение структуры файлов листинга
3. Самостоятельное написание программ по материалам лабораторной работы

3 Теоретическое введение

Для реализации ветвлений в ассемблере используются так называемые команды передачи управления или команды перехода. Можно выделить 2 типа переходов: • условный переход – выполнение или не выполнение перехода в определенную точку программы в зависимости от проверки условия. • безусловный переход – выполнение передачи управления в определенную точку программы без каких-либо условий.

4 Выполнение лабораторной работы

4.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы №7.

```
root@vbox:/home/ssnikulenkov/work/arch-pc# mkdir lab07
root@vbox:/home/ssnikulenkov/work/arch-pc# ls
lab04  lab05  lab06  lab07
root@vbox:/home/ssnikulenkov/work/arch-pc# touch lab7-1.asm
root@vbox:/home/ssnikulenkov/work/arch-pc#
```

Рис. 4.1: Создание каталога и файла для программы

Копирую inout.asm в рабочий каталог.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab06# cp in_out.asm /home/ssnikulenko
v/work/arch-pc/lab07
```

Рис. 4.2: Копирование

Изменяю программу таким образом, чтобы поменялся порядок выполнения функций.Запускаю программу и проверяю, что примененные изменения верны.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-1
Сообщение № 2
Сообщение № 1
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07#
```

Рис. 4.3: Изменение программы

Теперь изменяю текст программы так, чтобы все три сообщения вывелись в обратном порядке.Работа выполнена корректно, программа в нужном мне порядке выводит сообщения.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# nasm -f elf lab7-1.asm
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ld -m elf_i386 -o lab7-1 lab7-1
.o
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07#
```

Рис. 4.4: Изменение программы

Создаю новый рабочий файл и вставляю в него код из следующего листинга.Программа выводит значение переменной с максимальным значением, проверяю работу программы с разными входными данными.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# nasm -f elf lab7-2.asm
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ld -m elf_i386 -o lab7-2 lab7-2
.o
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-2
Введите B: 3
Наибольшее число: 50
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-2
Введите B: 55
Наибольшее число: 55
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-2
Введите B: -32
Наибольшее число: 50
```

Рис. 4.5: Проверка программы из листинга

4.2 Изучение структуры файла листинга

Создаю файл листинга с помощью флага -l команды nasm и открываю его с помощью текстового редактора mousepad.

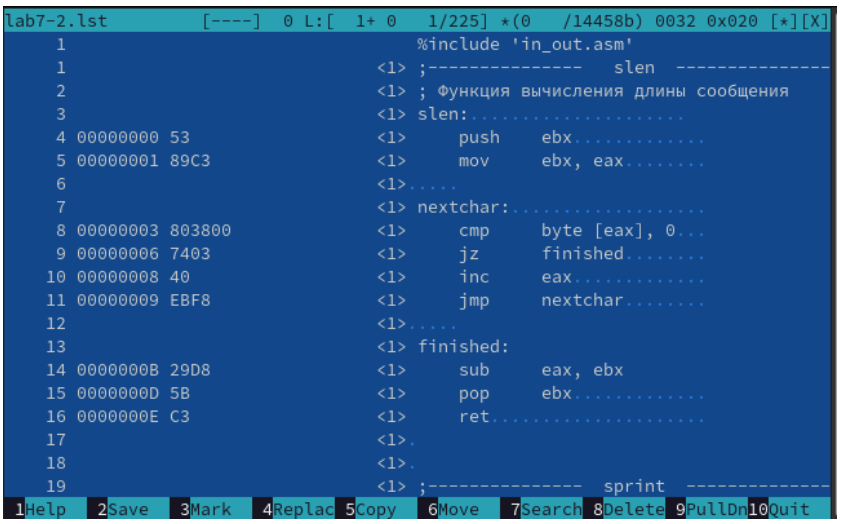

Рис. 4.6: Проверка файла листинга

Первое значение в файле листинга - номер строки, и он может вовсе не совпадать с номером строки изначального файла. Второе вхождение - адрес, смещение машинного кода относительно начала текущего сегмента, затем непосредственно идет сам машинный код, а заключает строку исходный текст прогарммы с комментариями.

Удаляю один операнд из случайной инструкции, чтобы проверить поведение файла листинга в дальнейшем В новом файле листинга показывает ошибку, которая возникла при попытке трансляции файла. Никакие выходные файлы при этом помимо файла листинга не создаются.

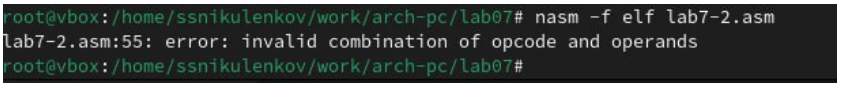

Рис. 4.7: Просмотр ошибки в файле листинга

4.3 Задания для самостоятельной работы.Вариант 5

Возвращаю операнд к функции в программе и изменяю ее так, чтобы она выводила переменную с наименьшим значением. Код первой программы:

```
%include 'in_out.asm'
section .data
msg1 db 'Введите A: ',0h
msg2 db 'Введите B: ',0h
msg3 db 'Введите C: ',0h
msg4 db "Наименьшее число: ",0h
section .bss
min resb 10
A resb 10
B resb 10
C resb 10
section .text
global _start
_start:
; ---------- Вывод сообщения 'Введите A: '
mov eax,msg1
call sprint
; ---------- Ввод 'A'
mov ecx,A
mov edx,10
call sread
; ---------- Преобразование 'A' из символа в число
mov eax,A
call atoi ; Вызов подпрограммы перевода символа в число
mov [A],eax ; запись преобразованного числа в 'A'


; ---------- Вывод сообщения 'Введите B: '
mov eax,msg2
```

```
call sprint
; ---------- Ввод 'B'
mov ecx,B
mov edx,10
call sread
; ---------- Преобразование 'B' из символа в число
mov eax,B
call atoi ; Вызов подпрограммы перевода символа в число
mov [B],eax ; запись преобразованного числа в 'B'


; ---------- Вывод сообщения 'Введите C: '
mov eax,msg3
call sprint
; ---------- Ввод 'C'
mov ecx,C
mov edx,10
call sread
; ---------- Преобразование 'C' из символа в число
mov eax,C
call atoi ; Вызов подпрограммы перевода символа в число
mov [C],eax ; запись преобразованного числа в 'C'


; ---------- Записываем 'A' в переменную 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ---------- Сравниваем 'A' и 'C'
mov ecx,[min]
```

```
cmp ecx,[C] ; Сравниваем 'A' и 'C'
jl check_B ; если 'A<C', то переход на метку 'check_B',
mov ecx,[C] ; иначе 'ecx = C'
mov [min],ecx ; 'min = C'
jmp prefin
; ---------- Преобразование 'max(A,C)' из символа в число
check_B:
mov ecx,[A]
mov [min],ecx ; запись преобразованного числа в `min`
; ---------- Сравниваем 'min(A,C)' и 'B' (как числа)
prefin:
mov ecx,[min]
cmp ecx,[B] ; Сравниваем 'min(A,C)' и 'B'
jl fin ; если 'min(A,C)<B', то переход на 'fin',
mov ecx,[B] ; иначе 'ecx = B'
mov [min],ecx
; ---------- Вывод результата
fin:
mov eax, msg4
call sprint ; Вывод сообщения 'Наименьшее число: '
mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход
```

Проверяю корректность написания первой программы.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-2
Введите A: 54
Введите B: 62
Введите C: 87
Наименьшее число: 54
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07#
```

Рис. 4.8: Проверка работы первой программы

Пишу программу, которая будет вычислять значение заданной функции согласно моему варианту для введенных с клавиатуры переменных a и x.

Код второй программы:

```
%include 'in_out.asm'
SECTION .data
msg_x: DB 'Введите значение переменной x: ', 0
msg_a: DB 'Введите значение переменной a: ', 0
res: DB 'Результат: ', 0
SECTION .bss
x: RESB 80
a: RESB 80
SECTION .text
GLOBAL _start
_start:
mov eax, msg_x
call sprint
mov ecx, x
mov edx, 80
call sread
mov eax, x
call atoi
mov edi, eax

mov eax, msg_a
call sprint
mov ecx, a
mov edx, 80
call sread
mov eax, a
```

```
call atoi
mov esi, eax

cmp edi, esi
jge print_15

; x < a
sub esi, edi
mov eax, esi
mov ebx, 2
mul ebx
mov edi, eax
jmp print_result

print_15:
mov edi, 15

print_result:
mov eax, res
call sprint
mov eax, edi
call iprintLF
call quit
```

Транслирую и компоную файл, запускаю и проверяю работу программмы для различных значений a и x.

```
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-3
Введите значение переменной x: 1
Введите значение переменной a: 2
Результат: 2
root@vbox:/home/ssnikulenkov/work/arch-pc/lab07# ./lab7-3
Введите значение переменной x: 2
Введите значение переменной a: 1
Результат: 15
```

Рис. 4.9: Проверка работы второй программы

5 Выводы

При выполнении лабораторной работы я изучил команды условных и безусловных переходво, а также приобрел навыки написания программ с использованием перходов, познакомился с назначением и структурой файлов листинга.

Список литературы

1. Курс на ТУИС
2. Лабораторная работа №7
3. Программирование на языке ассемблера NASM Столяров А. В.