Отчет по лабораторной работе №7

Дисциплина: архитектура компьютера

Никуленков Степан Сергеевич

Содержание

# 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 про- граммы без каких-либо условий.

# 4 Выполнение лабораторной работы

## 4.1 Реализация переходов в NASM

Создаю каталог для программ лабораторной работы №7.

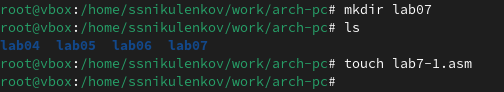


Рис. 1: Создание каталога и файла для программы

Копирую inout.asm в рабочий каталог.

Копирование

Рис. 2: Копирование

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

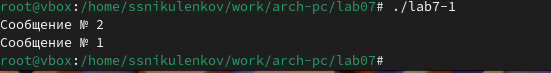


Рис. 3: Изменение программы

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

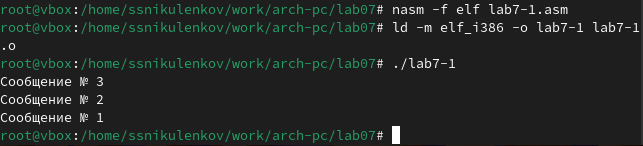


Рис. 4: Изменение программы

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.

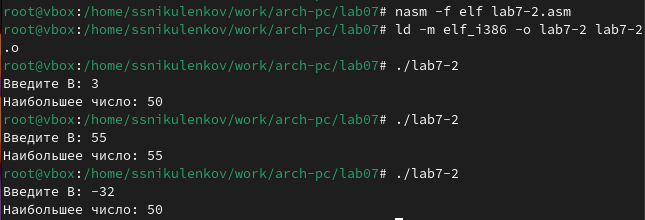


Рис. 5: Проверка программы из листинга

## 4.2 Изучение структуры файла листинга

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

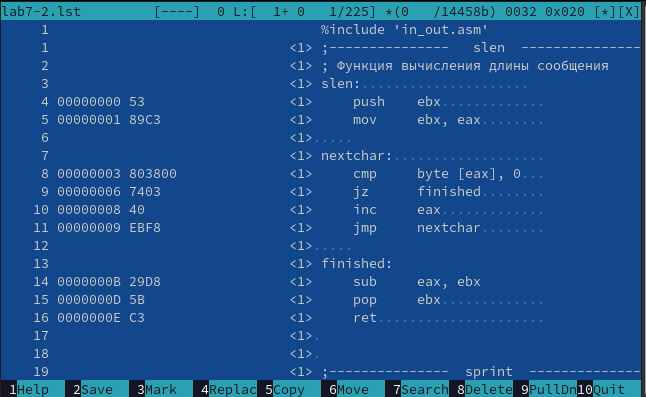


Рис. 6: Проверка файла листинга

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.

Просмотр ошибки в файле листинга

Рис. 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 'С'   
mov ecx,[min]  
cmp ecx,[C] ; Сравниваем 'A' и 'С'  
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.

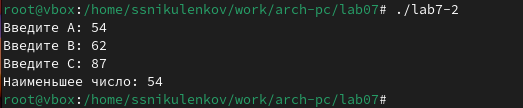


Рис. 8: Проверка работы первой программы

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.

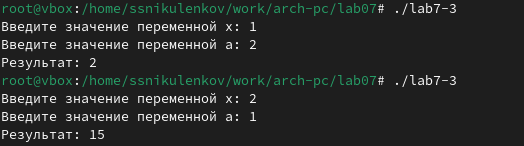


Рис. 9: Проверка работы второй программы

# 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](https://esystem.rudn.ru/course/view.php?id=112)
2. [Лабораторная работа №7](https://esystem.rudn.ru/pluginfile.php/2089087/mod_resource/content/0/%D0%9B%D0%B0%D0%B1%D0%BE%D1%80%D0%B0%D1%82%D0%BE%D1%80%D0%BD%D0%B0%D1%8F%20%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D0%B0%20%E2%84%967.%20%D0%9A%D0%BE%D0%BC%D0%B0%D0%BD%D0%B4%D1%8B%20%D0%B1%D0%B5%D0%B7%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%B8%20%D1%83%D1%81%D0%BB%D0%BE%D0%B2%D0%BD%D0%BE%D0%B3%D0%BE%20%D0%BF%D0%B5%D1%80%D0%B5%D1%85%D0%BE%D0%B4%D0%BE%D0%B2%20%D0%B2%20Nasm.%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%B2%D0%B5%D1%82%D0%B2%D0%BB%D0%B5%D0%BD%D0%B8%D0%B9.pdf)
3. [Программирование на языке ассемблера NASM Столяров А. В.](https://esystem.rudn.ru/pluginfile.php/2088953/mod_resource/content/2/%D0%A1%D1%82%D0%BE%D0%BB%D1%8F%D1%80%D0%BE%D0%B2%20%D0%90.%20%D0%92.%20-%20%D0%9F%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5%20%D0%BD%D0%B0%20%D1%8F%D0%B7%D1%8B%D0%BA%D0%B5%20%D0%B0%D1%81%D1%81%D0%B5%D0%BC%D0%B1%D0%BB%D0%B5%D1%80%D0%B0%20NASM%20%D0%B4%D0%BB%D1%8F%20%D0%9E%D0%A1%20Unix.pdf)