Отчёт по лабораторной работе №7

Дисциплина: архитектура компьютера

Самойлова Софья Дмитриевна

Содержание

# 1 Цель работы

Целью лабораторной работы является 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 файлы листинга
3. Задание для самостоятельной 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

С помощью утилиты mkdir создаю директорию, в которой буду создавать файлы с программами для лабораторной работы №7. Перехожу в созданный каталог с помощью утилиты cd и создаю файл lab7-1.asm (рис. 1). Дополнительно копирую в текущий каталог файл in\_out.asm с помощью утилиты cp, т.к. он будет использоваться в других программах

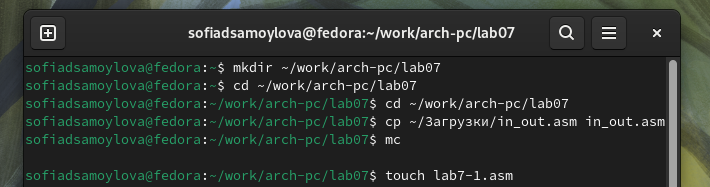


Рис. 1: Создание каталога

Инструкция jmp в NASM используется для реализации безусловных переходов. Рассмотрю пример программы с использованием инструкции jmp. Ввожу в файл lab7-1.asm текст программы (рис. 2).

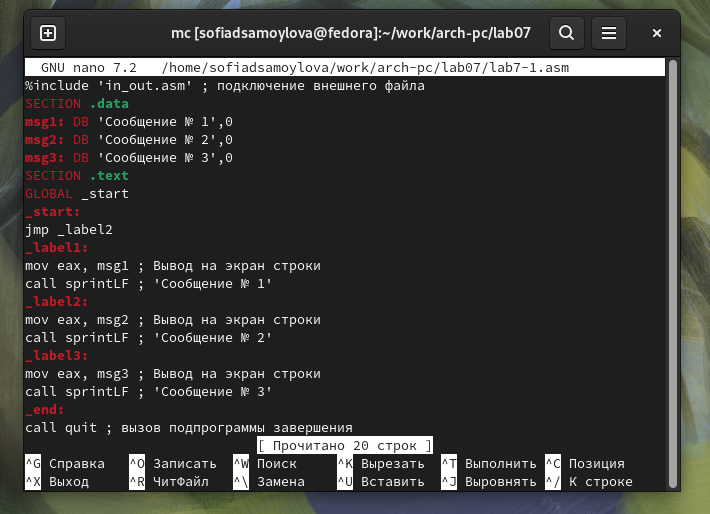


Рис. 2: Ввод программы

Создаю исполняемый файл и запускаю его. Результат работы данной программы следующий (рис. 3).

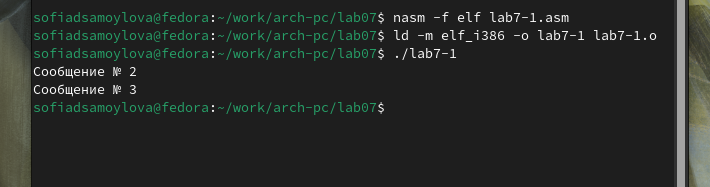


Рис. 3: Результаты работы программы

Таким образом, использование инструкции jmp \_label2 меняет порядок исполнения инструкций и позволяет выполнить инструкции начиная с метки \_label2, пропустив вывод первого сообщения.

Изменяю программу таким образом, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение № 1’ и завершала работу. Для этого в текст программы после вывода сообщения № 2 добавим инструкцию jmp с меткой \_label1 (т.е. переход к инструкциям вывода сообщения № 1) и после вывода сообщения № 1 добавим инструкцию jmp с меткой \_end (т.е. переход к инструкции call quit) (рис. 4).

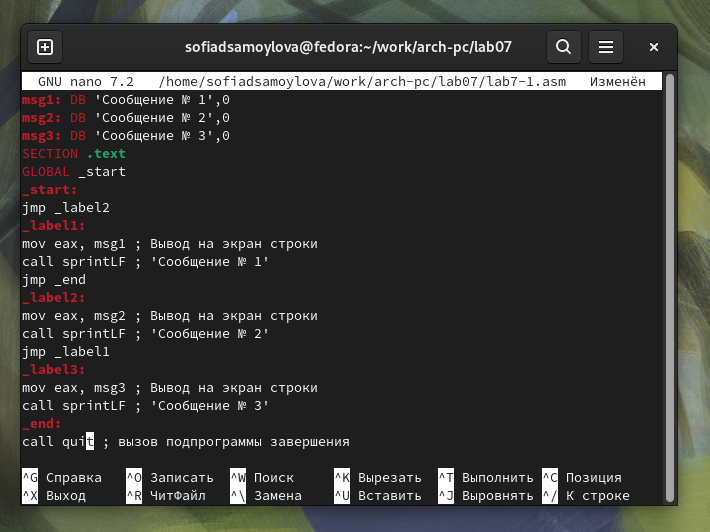


Рис. 4: Редактирование программы

Создаю исполняемый файл и проверяю его работу (рис. 5).

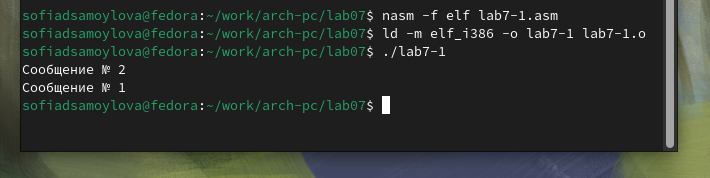


Рис. 5: Работа программы

Изменяю текст программы, корректируя инструкции jmp, чтобы вывод программы был следующим (рис. 6):

user@dk4n31:~$ ./lab7-1  
Сообщение № 3  
Сообщение № 2  
Сообщение № 1  
user@dk4n31:~$

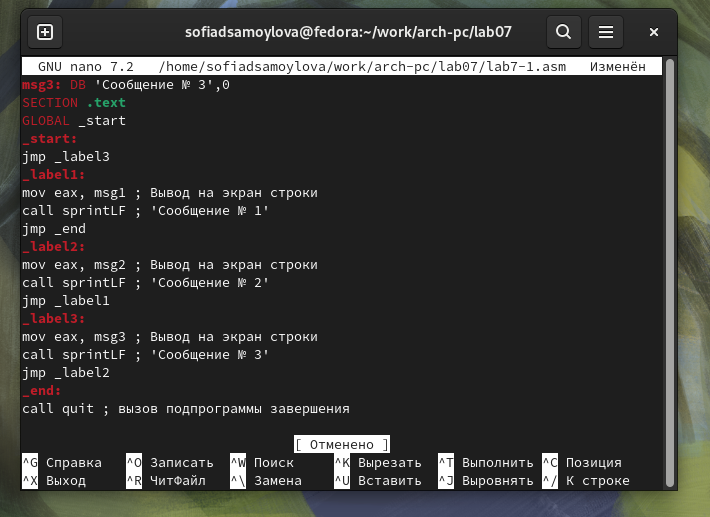


Рис. 6: Редактирование программы

Создаю исполняемый файл и проверяю его работу (рис. 7).

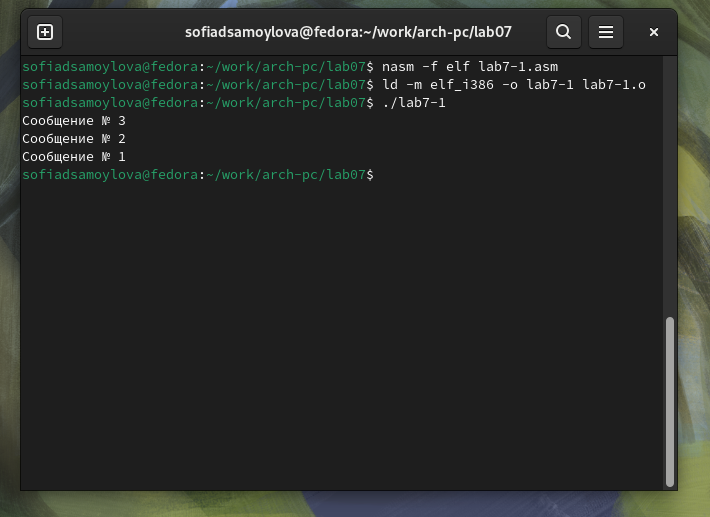


Рис. 7: Результат редактирования

Создаю файл lab7-2.asm в каталоге ~/work/arch-pc/lab07и ввожу код программы в lab7-2.asm (рис. 8).

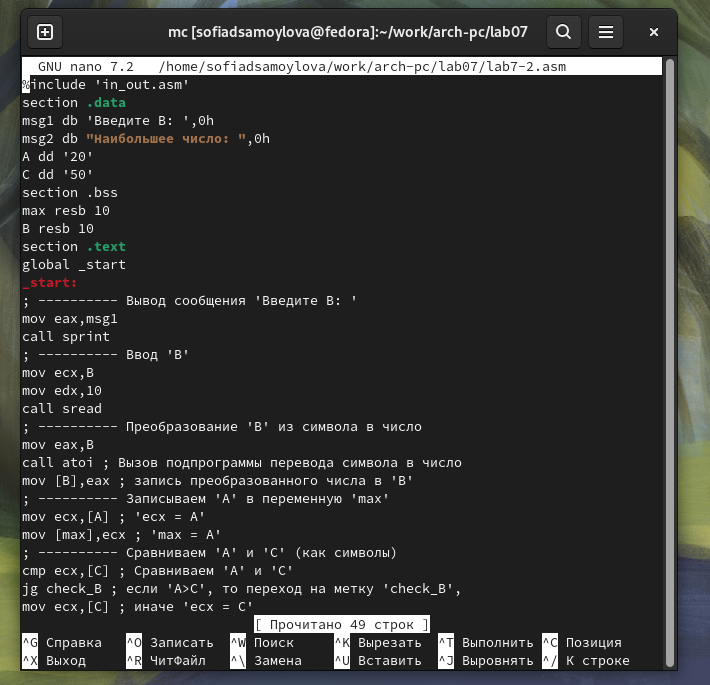


Рис. 8: Ввод программы

Создаю исполняемый файл и проверяю его работу для разных значений В (рис. 9).

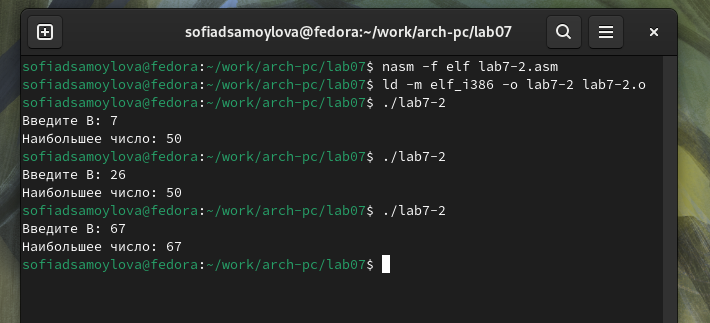


Рис. 9: Результат работы программы

В данном примере переменные A и С сравниваются как символы, а переменная B и максимум из A и С как числа (для этого используется функция atoi преобразования символа в число). Это сделано для демонстрации того, как сравниваются данные.

## 4.2 Изучение структуры файлы листинга

Обычно nasm создаёт в результате ассемблирования только объектный файл. Получить файл листинга можно, указав ключ -l и задав имя файла листинга в командной строке. Создаю файл листинга для программы из файла lab7-2.asm и открываю его с помощью mcedit(рис. 10).

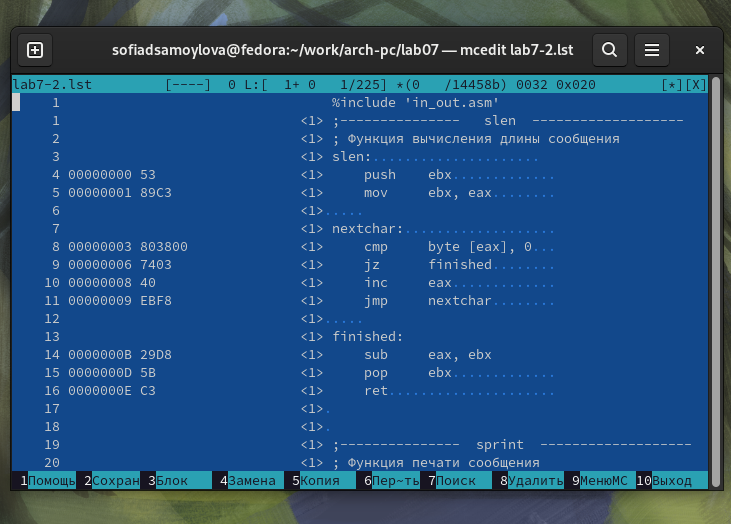


Рис. 10: Открытие файла листинга

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 программы с комментариями.

Попробую изменить следующую строку:

cmp ecx,[C] ; Сравниваем 'A' и 'C'

Удалив один из операндов, изменяю её на:

cmp ecx ; Сравниваем 'A' и 'C'

Последствия изменений

1. Синтаксические ошибки: Удаление одного из операндов в инструкции сравнения cmp приведет к синтаксической ошибке, так как команда cmp требует два операнда (регистры или память).
2. Выходные файлы:

* • Если вы попытаетесь собрать программу с этой ошибкой, компилятор (или ассемблер) не сможет создать исполняемый файл, так как код будет содержать ошибку.
* • В случае успешной сборки (если исправить ошибку), будут созданы выходные файлы, такие как .o (объектный файл) и исполняемый файл (например, a.out или с другим именем, если вы укажете его).

Соответственно, не получится создать и скомпилировать файлы (рис. 11).

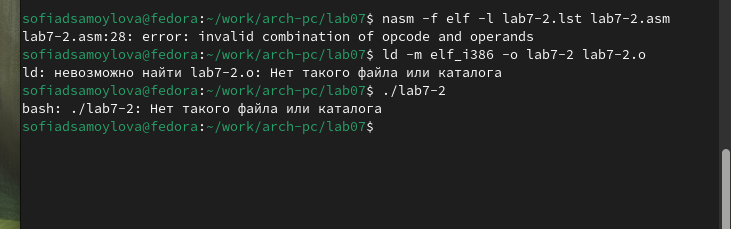


Рис. 11: Ошибка

## 4.3 Задание для самостоятельной работы

Вариант предыдущей лабораторной работы 17.

*Задание 1* Код программы:

%include 'in\_out.asm'  
  
section .data  
msg1 db 'Введите B: ', 0  
msg2 db 'Наименьшее число: ', 0  
A dd 26 ; Значение A  
B dd 12 ; Значение B  
C dd 68 ; Значение C  
  
section .bss  
min resd 1 ; Переменная для хранения наименьшего значения  
  
section .text  
global \_start  
\_start:  
 ; ---------- Начинаем с предположения, что A - наименьшее  
 mov eax, [A]  
 mov [min], eax ; min = A  
  
 ; ---------- Сравниваем min и B  
 mov eax, [B]  
 cmp eax, [min]  
 jl update\_min ; Если B < min, обновляем min  
  
 ; ---------- Сравниваем min и C  
 mov eax, [C]  
 cmp eax, [min]  
 jl update\_min ; Если C < min, обновляем min  
  
 jmp print\_result ; Переход к выводу результата  
  
update\_min:  
 mov eax, [B] ; Если один из предыдущих сравнений был истинным, обновляем min  
 mov [min], eax  
  
print\_result:  
 ; ---------- Вывод результата  
 mov eax, msg2  
 call sprint ; Вывод сообщения 'Наименьшее число: '  
 mov eax, [min]  
 call iprintLF ; Вывод 'min(A,B,C)'  
 call quit ; Выход

Результат работы программы (рис. 12).

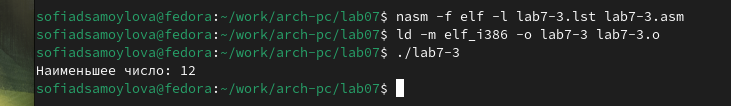


Рис. 12: Работа программы

*Задание 2* Код программы:

%include 'in\_out.asm'  
  
section .data  
msg1 db 'Введите x: ', 0  
msg2 db 'Введите a: ', 0  
msg3 db 'Результат f(x): ', 0  
A dd 20 ; Значение A, не используется в данной задаче  
C dd 50 ; Значение C, не используется в данной задаче  
  
section .bss  
result resb 10 ; Переменная для хранения результата  
x resb 10 ; Переменная для хранения введенного x  
a resb 10 ; Переменная для хранения введенного a  
  
section .text  
global \_start  
\_start:  
 ; ---------- Вывод сообщения 'Введите x: '  
 mov eax, msg1  
 call sprint  
   
 ; ---------- Ввод 'x'  
 mov ecx, x  
 mov edx, 10  
 call sread  
   
 ; ---------- Преобразование 'x' из символа в число  
 mov eax, x  
 call atoi ; Вызов подпрограммы перевода символа в число  
 mov [x], eax ; Запись преобразованного числа в 'x'  
   
 ; ---------- Вывод сообщения 'Введите a: '  
 mov eax, msg2  
 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'  
   
 ; ---------- Сравниваем 'a' с 8  
 mov eax, [a]  
 cmp eax, 8 ; Сравниваем a с 8  
 jl case\_a\_less\_8 ; Если a < 8, переходим к case\_a\_less\_8  
  
 ; Если a >= 8, вычисляем a \* x  
 mov eax, [a]  
 mov ebx, [x] ; Загружаем значение x в ebx  
 imul eax, ebx ; Умножаем a на x (eax = a \* x)  
 jmp store\_result ; Переход к сохранению результата  
  
case\_a\_less\_8:  
 ; Если a < 8, вычисляем a + 8  
 mov eax, [a]  
 add eax, 8 ; eax = a + 8  
  
store\_result:  
 mov [result], eax ; Сохраняем результат в переменной result  
  
 ; ---------- Вывод результата  
 mov eax, msg3  
 call sprint ; Вывод сообщения 'Результат f(x): '  
   
 mov eax, [result]  
 call iprintLF ; Вывод результата f(x)  
   
 call quit ; Выход

Результат работы программы (рис. 13).

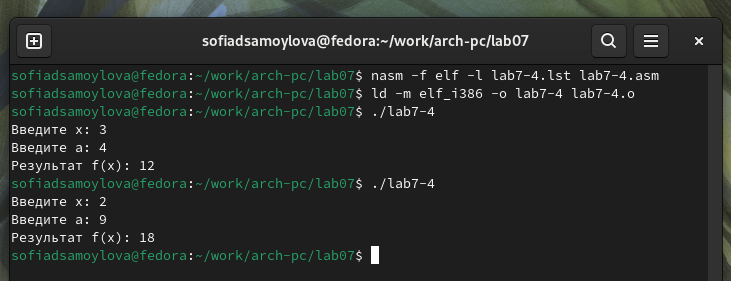


Рис. 13: Работа программы

# 5 Выводы

При выполнении лабораторной работы я изучила команды условных и безусловных переходов, а также приобрела навыки написания программ с использованием переходов, познакомилась с назначением и структурой файлов листинга.