Отчет по лабораторной работе №7

Команды безусловного и условного переходов в Nasm. Программирование ветвлений.

Корчагин Алексей Павлович

Содержание

# 1 Цель работы

Изучить команды условных и безусловных переходов. Преобрести наввыки написания программ с использованием переходов. Познакомится с назаначением и структурой файла

# 2 Теоретическое введение

В Ассемблере безусловный переход выполняется с помощью инструкции jmp, она включает в себя адрес перехода, то-есть куда следует передать управление: jmp

jmp label - переход на метку label jmp (label в квадратных скобках) - переход по адресу в памяти, помеченному меткой label jmp eax - переход по адресу из регистра eax

Как показанно выше, для условного перехода необходима проверка какого-либо условия. В ассемблере команды условного перехода вычисляют условие перехода анализируя флаги из регистра флагов. Флаг – это бит, принимающий значение 1 («флаг установлен»), если выполнено некоторое условие, и значение 0 («флаг сброшен») в противном случае. Флаги работают независимо друг от друга, и лишь для удобства они помещены в единый регистр — регистр флагов, отражающий текущее состояние процессора (такие как CF PF AF ZF SF)

Инструкция cmp является одной из инструкций, которая позволяет сравнить операнды и выставляет флаги в зависимости от результата сравнения. Инструкция cmp является командой сравнения двух операндов: cmp , Команда cmp, так же как и команда вычитания, выполняет вычитание - , но результат вычитания никуда не записывается и единственным результатом команды сравнения является формирование флагов.

Листинг (в рамках понятийного аппарата NASM) — это один из выходных файлов, создаваемых транслятором. Все ошибки и предупреждения, обнаруженные при ассемблировании, транслятор выводит на экран, и файл листинга не создаётся. Структура листинга:

номер строки  
адрес — это смещение машинного кода от начала текущего сегмента  
машинный код представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности  
исходный текст программы

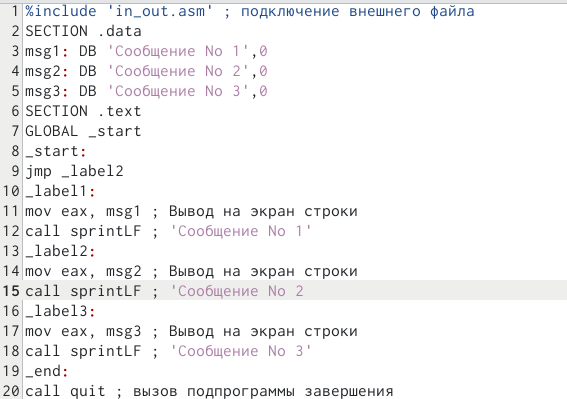
# 3 Выполнение лабораторной работы

Создал каталог для программ необходимых для выполненния лабараторной работы №7, также создал файл lab7-1.asm(рис. ??).



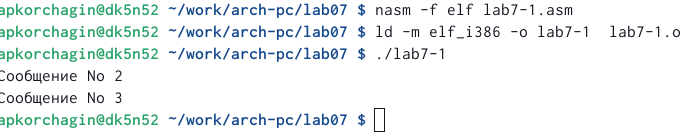
Создание каталога и файла

Ввёл текст программы в файл lab7-1.asm(рис. ??).



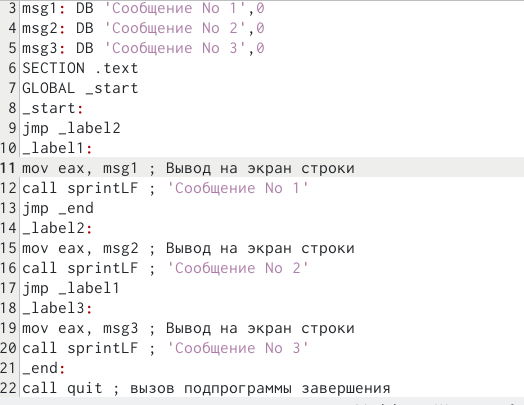
Редактирование файла

Создаю исполняемый файл и запускаю его (рис. ??).



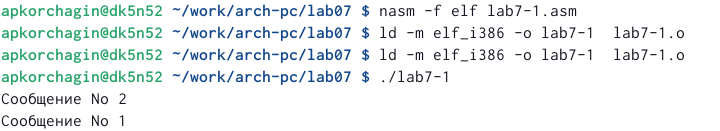
Создание и запуск исполняймого файла

Изменяю текст программы, чтобы она выводила сначала ‘Сообщение № 2’, потом ‘Сообщение№ 1’ и завершала работу(рис. ??).



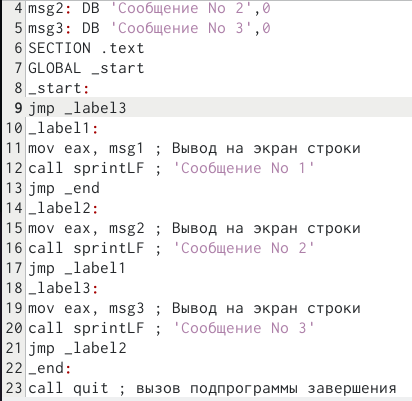
Редактирование файла

Создаю исполняемый файл и запускаю его(рис. ??).



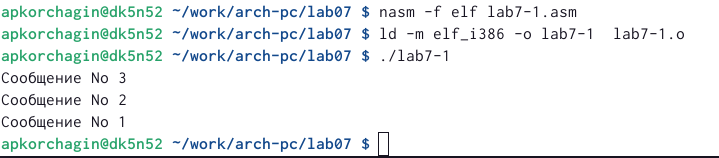
Создание исполняймого файла

Изменяю текст программы, чтобы она выводила сначала ‘Сообщение № 3’, потом ‘Сообщение№ 2’, а затем ‘Сообщение № 1’ (рис. ??).



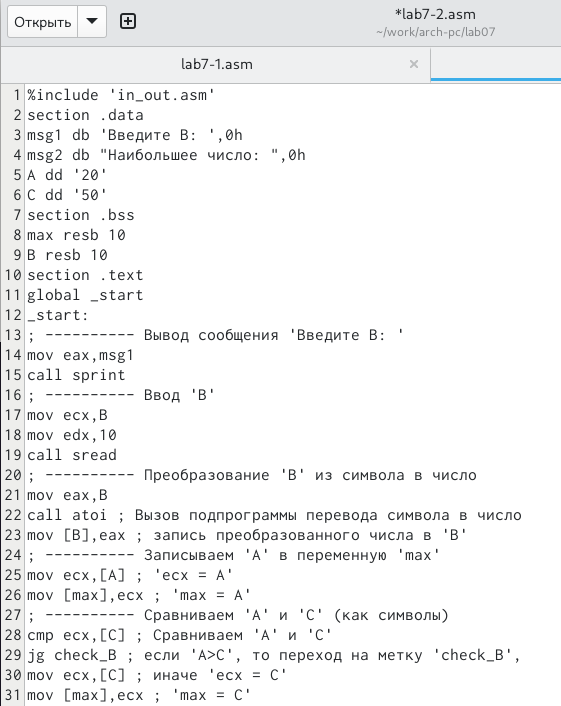
Редактирование файла

Создал и запусти исполняймый файл(рис. ??).



Создание исполняймого файла

Создал файл lab7-2.asm и ввёл в файл lab7-2.asm текст программы, которая определяет и выводит на экран наибольшую из целочисленных переменных: A,B и C(рис. ??).



Создание и реадктирование файла

Создаю исполняемый файл и запускаю его несколько раз для разных значений В, чтобы проверить его работу(рис. ??).



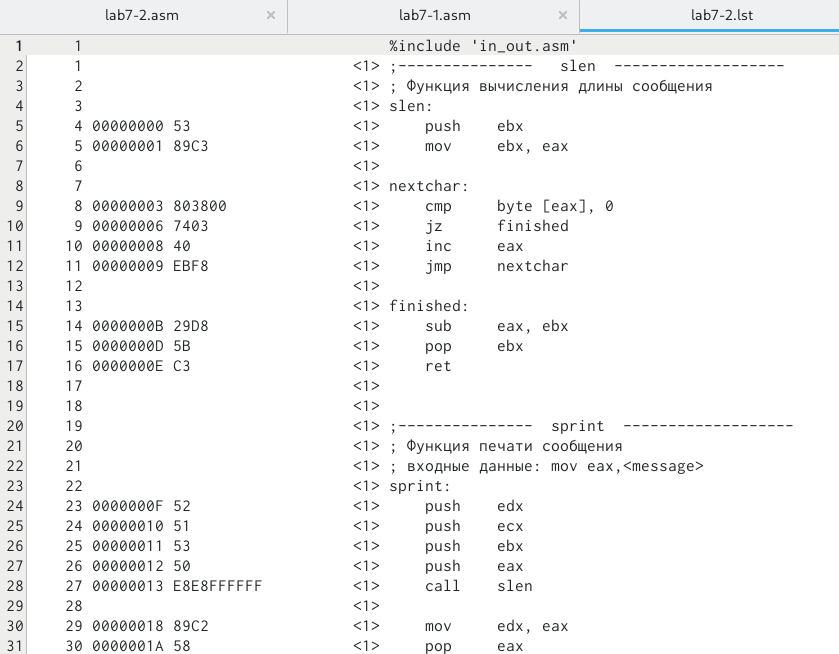
Создание исполняймого файла

Создаю файл листинга для программы из файла lab7-2.asm, указав ключ -l(рис. ??).

Создание файла

Создание файла

Открываю файл листинга lab7-2.lst (рис. ??).



Листинг

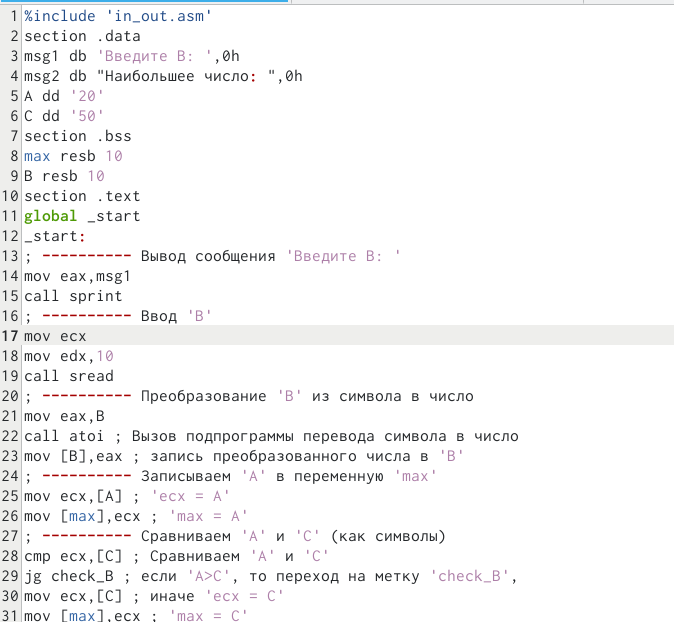
Объяснение содержимого трёх строк файла листинга:

10 строка: Перые цифры [10] - это номер строки файла листинга. Cледующие цифры [00000008] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел. следующие числа [40] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латынского алфавита. следющее [inc eax] - исходный текст программы, которая просто состоит из строкк исходной программы вместе с комментариями

11 строка: Перые цифры [11] - это номер строки файла листинга. Cледующие цифры [00000009] адрес — это смещение машинного кода от начала текущего сегмента, состоит из 8 чисел. следующие числа [EBF8] - это машинный код, который представляет собой ассемблированную исходную строку в виде шестнадцатеричной последовательности, поэтоу и появляются буквы латынского алфавита. следющее [jmp nextchar] - исходный текст программы, которая просто состоит из строкк исходной программы вместе с комментариями

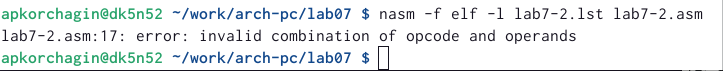
12 строка: содержит номер сторки [11], адресс [00000009], машинный код [EBF8] и содержимое строки кода [jmp nextchar]

Открываю файл с программой lab7-2.asm и в одной из инструкций с двумя операндами удаляю один операнд(рис. ??).



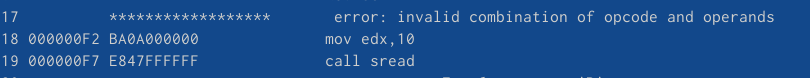
Редактирование файла

Пробую выполнить трансляцию с получением файла листинга(рис. ??).



ошибка

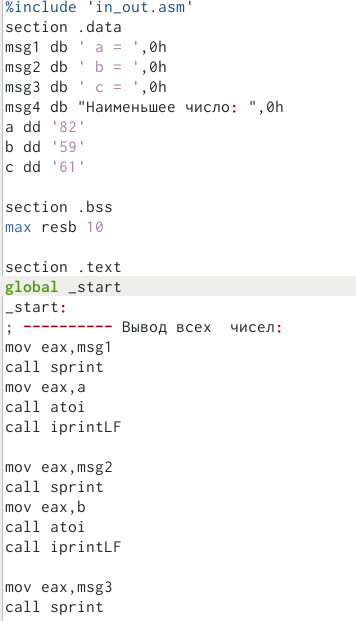
В листинге, как и в коде, появилась ошибка(рис. ??).



ошибка в листинге

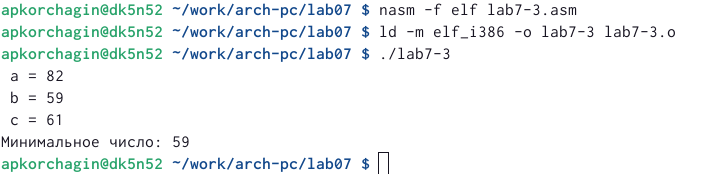
# 4 Задание Самостоятельной работы

Создал файл lab7-3.asm и написал в нём программу, значения взял из примера моего варианта, тоесть из примера второго варианта (рис. ??).



Программа поиска минимума

Создал исполняемый файл и проверил его работу. Все работает коректно (рис. ??).



Создание и тест исполняющего файла

Код программы:

%include ‘in\_out.asm’ section .data msg1 db ’ а = ‘,0h msg2 db’ b = ‘,0h msg3 db’ c = ‘,0h msg4 db “Минимальное число:”,0h a dd ’82’ b dd ‘59’ c dd ‘61’

section .bss max resb 10

section .text global \_start \_start: ; ———- Вывод чисел: mov eax,msg1 call sprint mov eax,a call atoi call iprintLF

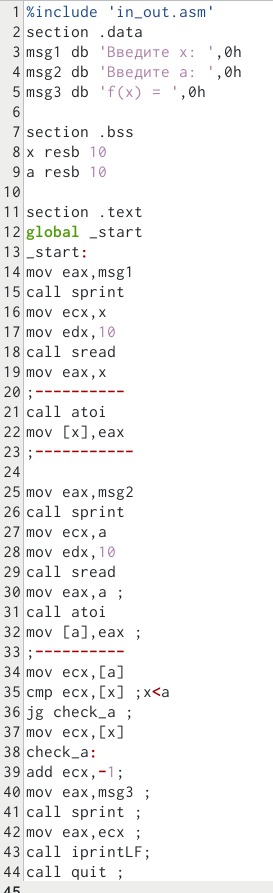
mov eax,msg2 call sprint mov eax,b call atoi call iprintLF

mov eax,msg3 call sprint mov eax,c call atoi call iprintLF

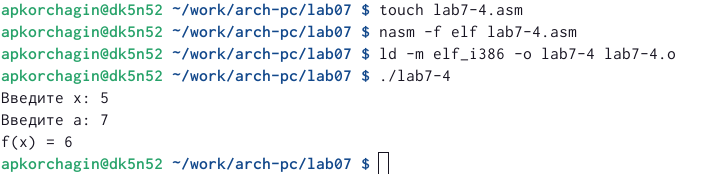
;————-Сравнение чисел mov eax,b call atoi ;перевод символа в число mov [b],eax ; запись преобразованного числа в b ;———— запись b в переменную мах mov ecx,[a] ; mov [max],ecx ; ;————сравнение чисел a c cmp ecx,[c]; if a>c jl check\_b ; то перход на метку mov ecx,[c] ; mov [max],ecx ; ;——-метка check\_b check\_b: mov eax,max ; call atoi mov [max],eax ; ;———— mov ecx,[max] ; cmp ecx,[b] ; jl check\_c ; mov ecx,[b] ; mov [max],ecx ; ;————- check\_c: mov eax,msg4 ; call sprint ; mov eax,[max]; call iprintLF ; call quit

2 Задание самостоятельной работы

Создал файл lab7-4.asm и написал в нйм программу, которая для введенных с клавиатуры значений x и a вычисляет значение функции f(x) и выводит результат вычислений. (рис. [1](#fig:018)).

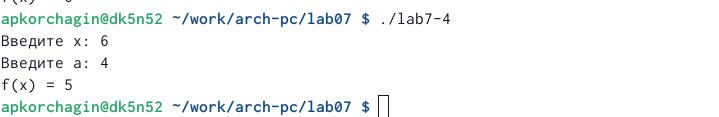


Создал иполняймый файл и провёл тест с первой парой значений для варианта 2(рис. ??).



Тест программы

Тестирую со второй парой занченний(рис. ??).



Тест программы

Код программы %include ‘in\_out.asm’ section .data msg1 db ‘Введите x:’,0h msg2 db ‘Введите a:’,0h msg3 db ‘f(x) =’,0h

section .bss x resb 10 a resb 10

section .text global \_start \_start: mov eax,msg1 call sprint mov ecx,x mov edx,10 call sread mov eax,x ;———- call atoi mov [x],eax ;———–

mov eax,msg2 call sprint mov ecx,a mov edx,10 call sread mov eax,a ; call atoi mov [a],eax ; ;———- mov ecx,[a] cmp ecx,[x] ;x<a jg check\_a ; mov ecx,[x] check\_a: add ecx,-1; mov eax,msg3 ; call sprint ; mov eax,ecx ; call iprintLF; call quit ;

# 5 Выводы

По ходу выполнения лабороторной работы я изучил команды условных и безусловных переходов в Ассемблере.

# Список литературы