Отчёт по лабораторной работе №8

Программирование ветвлений

Мулин Иван Владимирович

Содержание

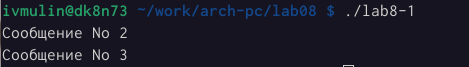
# 1 Цель работы

В ходе выполнения данной лабораторной работы необходимо изучить программирование ветвлений в языке ассемблера NASM. Репозиторий github расположен по адресу <https://github.com/ivmulin/study_2022-2023_arch-pc>.

# 2 Ход работы

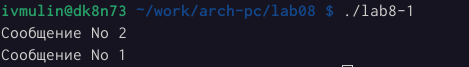
## 2.1 Выполнение лабораторной работы

Программа lab8-1.asm реализует алгоритм выполнения ветвлений при помощи команды jmp, выводя сначала второе сообщение, затем третье:



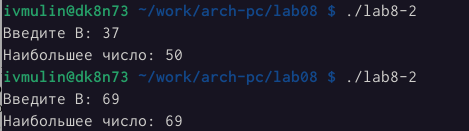
Запуск программы 1, дубль 1

Изменим программу так, что она выводит первое сообщение вслед за вторым:



Запуск изменённой программы 1

Вторая программа находит наибольшее из трёх чисел:



Запуск программы 2, дубль 1

При сборке программы был создан файл листинга. Рассмотрим следующий отрывок из него:

20 000000F2 B9[0A000000] mov ecx, B  
21 000000F7 BA0A000000 mov edx,10  
22 000000FC E842FFFFFF call sread

Команда B9[0A000000] (mov eax, B) располагается по смещению 000000F2. По аналогии команды BA0A000000 (mov edx,10) и E842FFFFFF (call sread) находятся по смещениям 000000F7 и 000000FC соответственно.

Удалим второй операнд в команде mov eax, max и попробуем собрать программу. В итоге создастся файл листинга со следующей ошибкой в соответствующем месте кода:

Запуск программы 2, дубль 2

Запуск программы 2, дубль 2

Объектный файл в этом случае не будет создан.

## 2.2 Выполнение заданий для самостоятельной работы

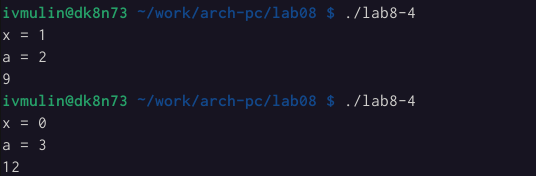
Выполняя задания самостоятельной работы, необходимо написать две программы. Первая программа вычисляет наименьшее из чисел 21, 28, 34:

Поиск наименьшего числа

Поиск наименьшего числа

Вторая программа выводит значение функции

от двух введённых аргументов:



Вычисление значения функции от двух переменных

Программы, как видно, работают исправно.

# 3 Листинги написанных программ

1. lab8-1.asm

%include 'in\_out.asm'  
section .data  
msg1: db 'Сообщение No 1', 0  
msg2: db 'Сообщение No 2', 0  
msg3: db 'Сообщение No 3', 0  
  
section .text  
global \_start  
  
\_start:  
 jmp \_label2  
  
\_label1:  
 mov eax, msg1  
 call sprintLF  
 jmp \_end  
  
\_label2:  
 mov eax, msg2  
 call sprintLF  
 jmp \_label1  
  
\_label3:  
 mov eax, msg3  
 call sprintLF  
  
\_end:  
 call quit

1. lab8-2.asm

%include 'in\_out.asm'  
  
section .data  
 msg1 db 'Введите B: ',0h  
 msg2 db "Наибольшее число: ",0h  
 A dd '20'  
 C dd '50'  
  
section .bss  
 max resb 10  
 B resb 10  
  
section .text  
global \_start  
  
\_start:  
 mov eax, msg1  
 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'  
 ; ---------- Записываем 'A' в переменную 'max'  
 mov ecx, [A] ; 'ecx = A'  
 mov [max], ecx ; 'max = A'  
 ; ---------- Сравниваем 'A' и 'С' (как символы)  
 cmp ecx, [C] ; Сравниваем 'A' и 'С'  
 jg check\_B ; если 'A>C', то переход на метку 'check\_B',  
 mov ecx, [C] ; иначе 'ecx = C'  
 mov [max], ecx ; 'max = C'  
; ---------- Преобразование 'max(A,C)' из символа в число  
  
check\_B:  
 mov eax  
 call atoi ; Вызов подпрограммы перевода символа в число  
 mov [max], eax ; запись преобразованного числа в `max`  
 ; ---------- Сравниваем 'max(A,C)' и 'B' (как числа)  
 mov ecx, [max]  
 cmp ecx, [B] ; Сравниваем 'max(A,C)' и 'B'  
 jg fin ; если 'max(A,C)>B', то переход на 'fin',  
 mov ecx, [B] ; иначе 'ecx = B'  
  
 mov [max], ecx  
  
; ---------- Вывод результата  
fin:  
 mov eax, msg2  
 call sprint ; Вывод сообщения 'Наибольшее число: '  
 mov eax, [max]  
 call iprintLF ; Вывод 'max(A,B,C)'  
 call quit ; Выход

1. lab8-3.asm

%include 'in\_out.asm'  
  
; 21, 28, 34  
  
section .data  
 msg1 db "Наименьшее число - "  
 a dd 20  
 b dd 28  
 c dd 50  
  
section .bss  
 min resb 10  
  
section .text  
global \_start  
  
\_start:  
; --------- печатаем "min(a, b, c) = "  
 mov eax, msg1  
 call sprint  
  
 mov ecx, [a]  
 mov [min], ecx ; 'min = A'  
 ; ---------- Сравниваем 'A' и 'С' (как числа)  
 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'  
; ---------- Преобразование 'min(A,C)' из символа в число  
  
check\_B:  
 ; ---------- Сравниваем 'min(A,C)' и 'B' (как числа)  
 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, [min]  
 call iprintLF ; Вывод 'min(A,B,C)'  
 call quit ; Выход

1. lab8-4.asm

%include 'in\_out.asm'  
  
section .data  
 msgX db 'x = ',0h  
 msgA db "a = ",0h  
  
section .bss  
 x resb 10  
 a resb 10  
 f resb 10  
  
section .text  
global \_start  
  
\_start:  
 ; ---------- Ввод 'B'  
 mov eax, msgX  
 call sprint  
 mov ecx, x  
 mov edx,10  
 call sread  
  
 ; ---------- Ввод 'x'  
 mov eax, msgA  
 call sprint  
 mov ecx, a  
 mov edx,10  
 call sread  
  
 ; ---------- Преобразование 'x' из символа в число  
 mov eax, x  
 call atoi  
 mov [x], eax  
  
 ; ---------- Преобразование 'a' из символа в число  
 mov eax, a  
 call atoi  
 mov [a], eax  
  
 mov ecx, [x]  
 cmp ecx, 0  
  
 je vadim  
  
 mov eax, [a]  
 mov ebx, 4  
 mul ebx  
 add eax, [x]  
 jmp fin  
  
vadim:  
 mov eax, [a]  
 mov ebx, 4  
 mul ebx  
  
fin:  
 call iprintLF ; Вывод 'max(A,B,C)'  
 call quit ; Выход

# 4 Заключение

В ходе выполненя лабораторной работы №8 было изучено программирование условного и безусловного переходов и ветвлений в языке ассемблера NASM, а значит, цель данной лабораторной работы была достигнута.