Отчет по лабораторной работе №7

Архитектура компьютера

Дмитрий Константинович Кобзев

Содержание

Список иллюстраций

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. Напишите программу нахождения наименьшей из 3 целочисленных переменных a,b и . Значения переменных выбрать из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу.
2. Напишите программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений. Вид функции f(x) выбрать из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создайте исполняемый файл и проверьте его работу для значений x и a из 7.6.

3 Выполнение лабораторной работы |

[1–6]

Создаем каталог для программ лабораторной работы № 7, переходим в него и создаем файл lab7-1.asm (рис. 1.1).

```
dkkobzev@dk6n54:~$ mkdir ~/work/arch-pc/lab07
dkkobzev@dk6n54:~$ cd ~/work/arch-pc/lab07
dkkobzev@dk6n54:~/work/arch-pc/lab07$ touch lab7-1.asm
```

Вводим в файл lab7-1.asm текст программы листинга 7.1 (рис. 1.2).

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Создаем исполняемый файл и запускаем его (рис. 1.3).

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ld -m elf_i
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 3
```

Изменяем текст программы в соответствии с листингом 7.2 (рис. 1.4).

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label2
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
_end:
call quit ; вызов подпрограммы завершения
```

Создаем исполняемый файл и запускаем его (рис. 1.5).

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ld -m elf_i
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 2
Сообщение № 1
```

Изменяем текст программы добавив или изменив инструкции jmp (рис. 1.6), (рис. 1.7).

```
%include 'in_out.asm' ; подключение внешнего файла
SECTION .data
msg1: DB 'Сообщение № 1',0
msg2: DB 'Сообщение № 2',0
msg3: DB 'Сообщение № 3',0
SECTION .text
GLOBAL _start
_start:
jmp _label3
_label1:
mov eax, msg1 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 1'
jmp _end
_label2:
mov eax, msg2 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 2'
jmp _label1
_label3:
mov eax, msg3 ; Вывод на экран строки
call sprintLF ; 'Сообщение № 3'
jmp _label2
_end:
call quit ; вызов подпрограммы завершения
```

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf lab7-1.asm
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ld -m elf_i386 -o lab7-1 lab7-1.o
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./lab7-1
Сообщение № 3
Сообщение № 2
Сообщение № 1
```

Создаем файл lab7-2.asm в каталоге ~/work/arch-pc/lab07. Внимательно изучаем текст программы из листинга 7.3 и вводим в lab7-2.asm. (рис. 1.8), (рис. 1.9).

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ touch lab7-2.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:
; ---------- Вывод сообщения 'Введите B: '
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 'C' (как символы)
cmp ecx,[C] ; Сравниваем 'A' и 'C'
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,max
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
```

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$
dkkobzev@dk6n54:~/work/arch-pc/lab07$
dkkobzev@dk6n54:~/work/arch-pc/lab07$
Введите B: 5
Наибольшее число: 50
```

Создаем исполняемый файл и проверяем его работу (рис. 1.10).

Создаем файл листинга для программы из файла lab7-2.asm (рис. 1.11).

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf -l lab7-2.lst lab7-2.asm
```

Открываем файл листинга lab7-2.lst с помощью mcedit (рис. 1.12).

```
dkkobzev@dk6n54:~/work/arch
```

4 Самостоятельная работа

Задание 1. Пишем программу нахождения наименьшей из 3 целочисленных переменных a,b и . Значения переменных выбираем из табл. 7.5 в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создаем исполняемый файл и проверяем его работу

```
%include 'in_out.asm'
section .data
msg2 db "Наименьшее число: ",0h
A dd '26'
B dd '12'
C dd '68'
section .bss
min resb 10
section .text
global _start
_start:
mov ecx,B
; ---------- 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 'min'
mov ecx,[A] ; 'ecx = A'
mov [min],ecx ; 'min = A'
; ---------- Сравниваем 'A' и 'C' (как символы)
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'
; ---------- Преобразование 'min(A,C)' из символа в число
check_B:
mov eax,min
call atoi ; Вызов подпрограммы перевода символа в число
mov [min],eax ; запись преобразованного числа в min
; ---------- 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, msg2
call sprint ; Вывод сообщения 'Наименьшее число: '
mov eax,[min]
call iprintLF ; Вывод 'min(A,B,C)'
call quit ; Выход
```

(рис. 2.1), (рис. 2.2).

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf sr.asm
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ld -m elf_i386 -o sr sr.o
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./sr
Наименьшее число: 12
```

Задание 2. Пишем программу, которая для введенных с клавиатуры значений x и a вычисляет значение заданной функции f(x) и выводит результат вычислений. Вид функции f(x) выбираем из таблицы 7.6 вариантов заданий в соответствии с вариантом, полученным при выполнении лабораторной работы № 7. Создаем исполняемый файл и проверяем его работу для значений x и

а из 7.6 (рис. 2.3), (рис. 2.4).

```
%include 'in_out.asm'
section .data
msg db 'Result: ', 0
msg1 db 'Введите X: ', 0
msg2 db 'Введите A: ', 0
a dd 79
b dd 83
c dd 41
section .bss
x resb 10
result resb 10
section .text
global _start
_start:
    ; Input for x
    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 eax, [a]
    ; Compare x with a
    cmp eax, 8
    jl eight_less_than_a  ; Jump if x < a
    mov ebx, [x]
    mul ebx
    jmp output_result
eight_less_than_a:
    add eax, 8
output_result:
    ; Output the result
    mov [result], eax
    mov eax, msg
    call sprint
    mov eax, [result]
    call iprintLF ; Print the result
    ; Exit
    call quit
```

```
dkkobzev@dk6n54:~/work/arch-pc/lab07$ nasm -f elf sr2.asm
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ld -m elf_i386 -o sr2 sr2.o
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./sr2
Введите X: 3
Введите A: 4
Result: 12
dkkobzev@dk6n54:~/work/arch-pc/lab07$ ./sr2
Введите X: 2
Введите A: 9
Result: 18
```

5 Выводы

В ходе выполнения лабораторной работы мною были изучены команд условного и безусловного переходов. Приобретены навыки написания программ с использованием переходов. Также я познакомился с назначением и структурой файла листинга.

Список литературы

1. GNU Bash Manual [Электронный ресурс]. Free Software Foundation, 2016. URL: https://www.gnu.org/software/bash/manual/.
2. Newham C. Learning the bash Shell: Unix Shell Programming. O'Reilly Media, 2005. 354 с.
3. Zarrelli G. Mastering Bash. Packt Publishing, 2017. 502 с.
4. Robbins A. Bash Pocket Reference. O'Reilly Media, 2016. 156 с.
5. Таненбаум Э. Архитектура компьютера. 6-е изд. СПб.: Питер, 2013. 874 с.
6. Таненбаум Э., Бос Х. Современные операционные системы. 4-е изд. СПб.: Питер, 2015. 1120 с.