

Отчет по лабораторной работе №6

дисциплина: Архитектура компьютера

Бондарь Татьяна Владимировна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	8
4.1	Ответы на вопросы	18
5	Задания для самостоятельной работы	19
6	Выводы	21

Список иллюстраций

4.1	Переход в каталог и создание файла	8
4.2	Программа вывода значения регистра еах	9
4.3	Исполнение программы из листинга 6.1	9
4.4	Исправленный текст программы lab6-1.asm	11
4.5	Исполнение программы lab6-1	12
4.6	Исполнение программы lab6-2	12
4.7	Исправленный текст программы	13
4.8	Запуск программы	13
4.9	Внесение изменений в программу	14
4.10	Запуск исполняемого файла	14
4.11	Текст программы lab6-3.asm	15
4.12	Запуск исполняемого файла	15
4.13	Исправление текста программы	16
4.14	Запуск исполняемого файла lab6-3	16
4.15	Текст программы variant.asm	17
4.16	Запуск программы	18
5.1	Текст программы lab6-4.asm	20
5.2	Запуск программы	20

Список таблиц

1 Цель работы

Целью работы является освоение арифметических инструкций языка ассемблера NASM.

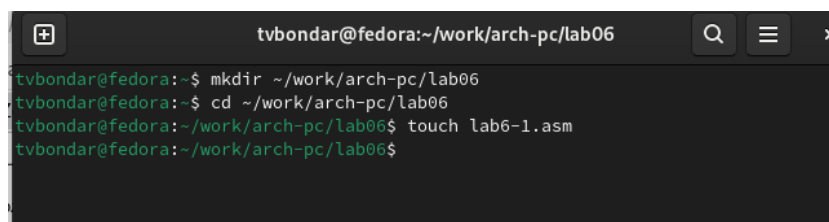
2 Задание

1. Написать программу вычисления выражения $y = f(x)$. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений. Вид функции $f(x)$ выбрать из таблицы 6.3 вариантов заданий в соответствии с номером полученным при выполнении лабораторной работы. Создайте исполняемый файл и проверьте его работу для значений x_1 и x_2 из 6.3. При выполнении задания преобразовывать (упрощать) выражения для $f(x)$ нельзя. При выполнении деления в качестве результата можно использовать только целую часть от деления и не учитывать остаток (т.е. $5 \div 2 = 2$).

3 Теоретическое введение

4 Выполнение лабораторной работы

1. Создаю каталог для программ лабораторной работы №6, перехожу в него и создаю файл lab6-1.asm.



```
tvbondar@fedora:~/work/arch-pc/lab06
tvbondar@fedora:~$ mkdir ~/work/arch-pc/lab06
tvbondar@fedora:~$ cd ~/work/arch-pc/lab06
tvbondar@fedora:~/work/arch-pc/lab06$ touch lab6-1.asm
tvbondar@fedora:~/work/arch-pc/lab06$
```

Рис. 4.1: Переход в каталог и создание файла

2. Ввожу в файл lab6-1.asm текст программы из листинга 6.1. Запускаю исполняемый файл.


```

#include 'in_out.asm'

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintf

call quit

```

Рис. 4.2: Программа вывода значения регистра eax

```

tvbondar@fedora: ~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
lab6-1.asm:1: error: unable to open include file 'in_out.asm': No such file or directory
tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-1.o -o lab6-1
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-1
64
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.3: Исполнение программы из листинга 6.1

3. Изменим текст программы и вместо символов, запишем в регистры числа.

Запустим исправленную программу. Теперь на экран вывелся символ с кодом 10. В соответствии с ASCII таблицей это символ перевода строк и он не отображается при выводе на экран.

```
%include 'in_out.asm'

SECTION .bss
buf1: RESB 80

SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
mov [buf1],eax
mov eax,buf1
call sprintf

call quit
```

Рис. 4.4: Исправленный текст программы lab6-1.asm

```

tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-1.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-1.o -o lab6-1
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-1

tvbondar@fedora:~/work/arch-pc/lab06$ █

```

Рис. 4.5: Исполнение программы lab6-1

4. Создадим файл lab6-2.asm. Преобразуем текст программы из Листинга 6.1 с использованием подпрограмм из файла in_out.asm, которые преобразуют ASCII символы в числа и обратно.

```

tvbondar@fedora:~/work/arch-pc/lab06$ touch lab6-2.asm
tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2  lab6-2.asm  lab6-2.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-2
106
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.6: Исполнение программы lab6-2

5. Аналогично предыдущему примеру исправим символы на числа и запустим программу. Теперь при запуске программы было получено само число 10, а не символ с соответствующим кодом в системе ASCII.

```

#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprintLF

call quit

```

Рис. 4.7: Исправленный текст программы

```

tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2  lab6-2.asm  lab6-2.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-2
10
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.8: Запуск программы

6. Далее заменяю функцию `iprintLF` на `iprint`. Запускаю исполняемый файл. Изменений нет.

```
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax,6
mov ebx,4
add eax,ebx
call iprint

call quit
```

Рис. 4.9: Внесение изменений в программу

```
tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-2.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-2.o -o lab6-2
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1  lab6-1.asm  lab6-1.o  lab6-2  lab6-2.asm  lab6-2.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-2
10
tvbondar@fedora:~/work/arch-pc/lab06$
```

Рис. 4.10: Запуск исполняемого файла

7. Создаю файл `lab6-3.asm`. Ввожу в него текст программы из листинга 6.3. Запускаю исполняемый файл.

```

;Программа вычисления выражения

#include 'in_out.asm'          ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:

    ; Вычисление выражения
    mov eax,5                ; FAX=5
    mov ebx,2                ; FBX=2
    mul ebx                  ; FAX=FAX*FBX
    add eax,3                ; FAX=FAX+3
    xor edx,edx              ; обнуляем EDX для корректной работы div
    mov ebx,3                ; FBX=3
    div ebx                  ; FAX=FAX/3, EDX=остаток от деления
    mov edi,eax              ; запись результата вычисления в 'edi'

    ; Вывод результата на экран
    mov eax,div              ; вызов подпрограммы печати
    call sprint              ; сообщения 'Результат: '
    mov eax,edi              ; вызов подпрограммы печати значения
    call iprintlf            ; из 'edi' в виде символов
    mov eax,rem              ; вызов подпрограммы печати
    call sprint              ; сообщения 'Остаток от деления: '
    mov eax,edx              ; вызов подпрограммы печати значения
    call iprintlf            ; из 'edx' (остаток) в виде символов
    call quit                ; вызов подпрограммы завершения

```

Рис. 4.11: Текст программы lab6-3.asm

```

tvbondar@fedora:~/work/arch-pc/lab06$ touch lab6-3.asm
tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-3.o -o lab6-3
tvbondar@fedora:~/work/arch-pc/lab06$ ды
bash: ды: command not found...
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1.asm  lab6-2      lab6-2.o  lab6-3.asm
lab6-1      lab6-1.o   lab6-2.asm  lab6-3    lab6-3.o
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-3
Результат: 4
Остаток от деления: 1
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.12: Запуск исполняемого файла

8. Изменяю текст программы для вычисления выражения $f(x) = (4*6 + 2)/5$.

```

;Программа вычисления выражения

%include 'in_out.asm'          ; подключение внешнего файла
SECTION .data
div: DB 'Результат: ',0
rem: DB 'Остаток от деления: ',0
SECTION .text
GLOBAL _start
_start:

    ; Вычисление выражения
    mov eax,4                ; EAX=4
    mov ebx,6                ; EBX=6
    mul ebx                  ; EAX=EAX*EBX
    add eax,2                ; EAX=EAX+2
    xor edx,edx              ; обнуляем EDX для корректной работы div
    mov ebx,5                ; EBX=5
    div ebx                  ; EAX=EAX/5, EDX=остаток от деления
    mov edi,eax              ; запись результата вычисления в 'edi'

    ; Вывод результата на экран
    mov eax,div              ; вызов подпрограммы печати
    call sprint              ; сообщения 'Результат: '
    mov eax,edi              ; вызов подпрограммы печати значения
    call iprintlf            ; из 'edi' в виде символов
    mov eax,rem              ; вызов подпрограммы печати
    call sprint              ; сообщения 'Остаток от деления: '
    mov eax,edx              ; вызов подпрограммы печати значения
    call iprintlf            ; из 'edx' (остаток) в виде символов
    call quit                ; вызов подпрограммы завершения

```

Рис. 4.13: Исправление текста программы

```

tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-3.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-3.o -o lab6-3
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-3
Результат: 5
Остаток от деления: 1
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.14: Запуск исполняемого файла lab6-3

- В качестве другого примера рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму.
 - вывести запрос на введение № студенческого билета
 - вычислить номер варианта по формуле: $(S_n \bmod 20) + 1$, где S_n – номер студенческого

билета (В данном случае $a \bmod b$ – это остаток от деления a на b). • вывести на экран номер варианта.

```
        ; Программа вычисления варианта

#include 'in_out.asm'

SECTION .data
msg: DB 'Введите № студенческого билета: ',0
rem: DB 'Ваш вариант: ',0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
_start:
mov eax, msg
call sprintf
mov ecx, x
mov edx, 80
call sread
mov eax, x          ; вызов подпрограммы преобразования
call atoi          ; ASCII кода в число, `eax=x
xor edx, edx
mov ebx, 20
div ebx
inc edx
mov eax, rem
call sprint
mov eax, edx
call iprintf
call quit
```

Рис. 4.15: Текст программы variant.asm

```

tvbondar@fedora:~/work/arch-pc/lab06$ touch variant.asm
tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf variant.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 variant.o -o variant
tvbondar@fedora:~/work/arch-pc/lab06$ ./variant
Введите № студенческого билета:
1132246711
Ваш вариант: 12
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 4.16: Запуск программы

4.1 Ответы на вопросы

1. Для вывода сообщения «Ваш вариант» предоставляются строки кода: `mov eax,rem call sprintLF`
2. Инструкция `mov esx, x` используется, чтобы поставить адрес вводимой строки `x` в регистр `esx`. `mov edx, 80` - запись в регистр `edx` длины вводимой строки `call sread` - вызов подпрограммы из внешнего файла, вводы сообщений с клавиатуры
3. Вызов `atoi` используется для вызова подпрограмм из внешнего файла, который преобразует символы `ascii`-кода в имена чисел и записывает результат в регистр `eax`.
4. За вычисление варианта отвечают следующие предложения: `xor edx,edx` ; обнуление `edx` для корректной работы `div` `mov ebx,20` ; `ebx = 20` `div ebx` ; `eax = eax/20`, `edx` - остаток от деления `inc edx` ; `edx = edx + 1`
5. При выполнении инструкции остаток деления `div ebx` записывается в регистр `edx`.
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1
7. За вывод результатов на экране в листинге 6.4 отвечают строки: `mov eax,edx` `call iprintLF`

5 Задания для самостоятельной работы

1. Создаю файл lab6-4.asm и ввожу в него текст программы для вычисления значения выражения $(8 \cdot x - 6) / 2$. Выражение было в варианте 12. Вычисляю значение выражения для заданных x .

```

;Программа вычисления выражения

%include 'in_out.asm'          ; подключение внешнего файла
SECTION .data
msg: DB 'Введите значение переменной x: ',0
res: DB 'Результат: ',0

SECTION .bss                  ; секция неинициализированных данных
x: RESB 80                    ; Переменная, заполнится с клавиатурой, выделенный размер - 80 байт

SECTION .text
GLOBAL _start
_start:

    ; Вычисление выражения
    mov eax,msg               ; EAX=выводимое сообщение
    call sprint
    mov ecx, x                 ; запись адреса переменной x в ecx
    mov edx, 80                ; запись длины вводимого значения в edx
    call sread
    mov eax,x                  ; вывод подпрограммы преобразования
    call atoi                  ; ASCII код в число, eax = x
    mov ebx,8                  ; ebx=8
    mul ebx                    ; eax = 8*x
    add eax, -6                ; eax= eax-6 = 8*x-6
    mov ebx, 2                 ; запись значения 2 в регистр ebx
    div ebx                    ; eax=eax/ebx = (8*x - 6)/2
    mov edi, eax               ; запись результата вычисления в edi

    ; Вывод результата на экран
    mov eax,res
    call sprint
    mov eax,edi
    call iprintf               ; из 'edi' в виде символов
    call quit                  ; вывод подпрограммы завершения

```

Рис. 5.1: Текст программы lab6-4.asm

```

tvbondar@fedora:~/work/arch-pc/lab06$ nasm -f elf lab6-4.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ld -m elf_i386 lab6-4.o -o lab6-4
tvbondar@fedora:~/work/arch-pc/lab06$ ls
in_out.asm  lab6-1.o  lab6-2.o  lab6-3.o  lab6-4.o  variant.o
lab6-1      lab6-2    lab6-3    lab6-4    variant
lab6-1.asm  lab6-2.asm lab6-3.asm lab6-4.asm variant.asm
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите значение переменной x: 1
Результат: 1
tvbondar@fedora:~/work/arch-pc/lab06$ ./lab6-4
Введите значение переменной x: 5
Результат: 17
tvbondar@fedora:~/work/arch-pc/lab06$

```

Рис. 5.2: Запуск программы

6 Выводы

В результате выполнения лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.