

Лабораторная работа №6

Архитектура вычислительных систем

Дмитрий Владимирович Орлюк

Содержание

1	Цель работы	5
2	Задание	6
3	Выполнение лабораторной работы	7
4	Выводы	15

Список иллюстраций

3.1	Создание каталога для программ	7
3.2	Ввод текст программы	7
3.3	Создание и запуск исполняемого файла	7
3.4	Изменения в тексте программы	8
3.5	Создание и запуск исполняемого файла	8
3.6	Таблица ASCII	9
3.7	Создание файла lab6-2	9
3.8	Ввод текста программы из листинга	9
3.9	Создание и запуск исполняемого файла	10
3.10	Изменение программы	10
3.11	Проверка работоспособности	10
3.12	Создание файла 6-3	11
3.13	Программа для вычисления выражения	11
3.14	Проверка работоспособности	12
3.15	Программа для вычисления измененного выражения	12
3.16	Проверка работоспособности	13
3.17	Создание файла variant	13
3.18	Программа для вычисления варианта по номеру студ. билета . . .	13
3.19	Проверка работоспособности	14

Список таблиц

1 Цель работы

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

3 Выполнение лабораторной работы

1. Создадим каталог для программ л.р №6, перейдем в него и создадим файл

```
dvorlyuk@dk2n22 ~ $ mkdir ~/work/arch-pc/lab06
dvorlyuk@dk2n22 ~ $ cd ~/work/arch-pc/lab06
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ touch lab6-1.asm
```

Рис. 3.1: Создание каталога для программ

2. Введем в файл текст программы из листинга, с помощью МС

```
lab6-1.asm      [-M--]  9 L:[ 1+12 13/ 13] *(172 / 172b) <EOF>
%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 sprintLF
call quit
```

Рис. 3.2: Ввод текст программы

3. Создаем исполняемый файл и запускаем

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-1
j
```

Рис. 3.3: Создание и запуск исполняемого файла

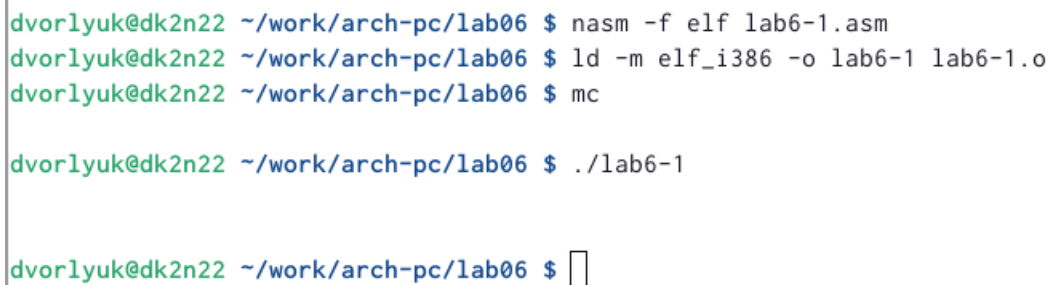
4. Изменим текст программы (уберем кавычки)

A screenshot of a text editor showing two lines of assembly code. The first line is 'mov eax,6' and the second line is 'mov ebx,4'. The code is displayed in a monospaced font. A vertical line is on the left, and a small red horizontal bar is at the top left.

```
mov eax,6
mov ebx,4
```

Рис. 3.4: Изменения в тексте программы

5. Создаем исполняемый файл и запускаем его.

A terminal window screenshot showing the compilation and execution of an assembly program. The user is at a prompt in the directory ~/work/arch-pc/lab06. They run 'nasm -f elf lab6-1.asm', then 'ld -m elf_i386 -o lab6-1 lab6-1.o', then 'mc', then './lab6-1', and finally a prompt with a cursor.

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ mc

dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-1

dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ █
```

Рис. 3.5: Создание и запуск исполняемого файла

6. Просмотрим по таблице ASCII определим какому символу соответствует код 10. Символ не отображается.

ASCII (American Standard Code for Information Interchange)

В таблице приведены ASCII-символы (Char) и их коды в десятичной (Dec) и шестнадцатеричной (Hex) системах счисления. Некоторые коды (00-32h, 7Fh) могут использоваться и в качестве команд (Cmd).

Проверить соответствие графического символа коду достаточно легко, например, при помощи стандартной программы "Блокнот" (Notepad) из состава Windows. Для этого необходимо при нажатой клавише **Alt** набрать нужный код (в десятичном виде) и на экране появится изображение соответствующего символа. Например, комбинация *Alt+31* даст символ ▼.

Dec	Hex	Char	Cmd	Dec	Hex	Char	Cmd	Dec	Hex	Char	Cmd	Dec	Hex	Char	Cmd
0	00		NUL	32	20		(sp)	64	40	@		96	60	`	
1	01	☉	SOH	33	21	!		65	41	A		97	61	a	
2	02	●	STX	34	22	"		66	42	B		98	62	b	
3	03	♥	ETX	35	23	#		67	43	C		99	63	c	
4	04	♦	EOT	36	24	\$		68	44	D		100	64	d	
5	05	♣	ENQ	37	25	%		69	45	E		101	65	e	
6	06	♠	ACK	38	26	&		70	46	F		102	66	f	
7	07	•	BEL	39	27	'		71	47	G		103	67	g	
8	08	▣	BS	40	28	(72	48	H		104	68	h	
9	09	○	TAB	41	29)		73	49	I		105	69	i	
10	0A	▣	LF	42	2A	*		74	4A	J		106	6A	j	

Рис. 3.6: Таблица ASCII

7. Создадим файл ассемблера с помощью команды touch

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ touch lab6-2.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ mc
```

Рис. 3.7: Создание файла lab6-2

8. Введем команду из листинга 6.2

```
lab6-2.asm [-M--] 9 L:[ 1+ 8 9/ 9] *(117 / 117b) <EOF>
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprintLF
call quit
```

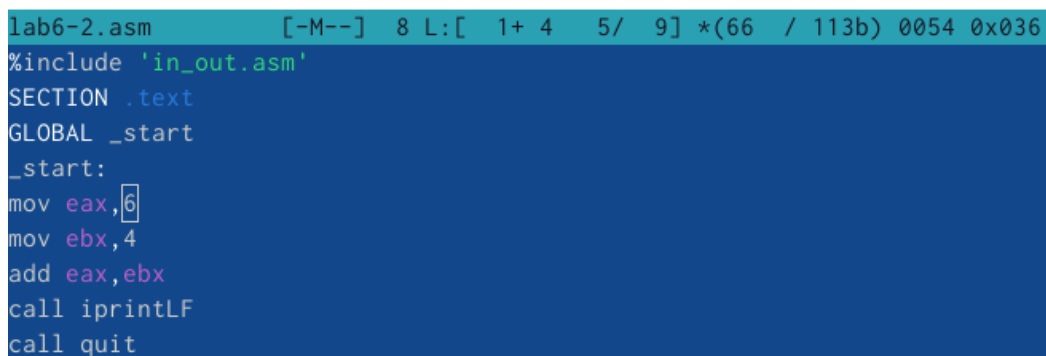
Рис. 3.8: Ввод текста программы из листинга

9. Создаем исполняемый файл и запускаем его.

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-2
106
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $
```

Рис. 3.9: Создание и запуск исполняемого файла

10. Вводим нужные изменения в текст программы



```
lab6-2.asm [-M--] 8 L:[ 1+ 4 5/ 9] *(66 / 113b) 0054 0x036
#include 'in_out.asm'
SECTION .text
GLOBAL _start
_start:
mov eax,6
mov ebx,4
add eax,ebx
call iprintLF
call quit
```

Рис. 3.10: Изменение программы

11. Создадим исполняемый файл и проверим его работоспособность (Команда выводит просто число 10. Команды iprint и iprintLF отличаются тем, что LF - это перенос на новую строку)

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-2
10
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $
```

Рис. 3.11: Проверка работоспособности

12. Создадим файл 6-3 с помощью команды touch

```
dvor1yuk@dk2n22 ~/work/arch-pc/lab06 $ touch lab6-3.asm
```

Рис. 3.12: Создание файла 6-3

13. Вставим текст из листинга.

```
lab6-3.asm      [-M--] 41 L:[ 1+25 26/ 26] *(1236/1236b)
%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 ; EAX=5
mov ebx,2 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,3 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,3 ; EBX=3
div ebx ; EAX=EAX/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. 3.13: Программа для вычисления выражения

14. Создадим исполняемый файл и запустим.

```

dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ 

```

Рис. 3.14: Проверка работоспособности

15. Меняем текст программы для вычисления другого выражения.

```

lab6-3.asm      [-M--]  9 L:[ 1+13 14/ 26] *(423 /1236b) 0032 0x020
#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=5
mov ebx,6 ; EBX=2
mul ebx ; EAX=EAX*EBX
add eax,2 ; EAX=EAX+3
xor edx,edx ; обнуляем EDX для корректной работы div
mov ebx,5 ; EBX=3
div ebx ; EAX=EAX/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. 3.15: Программа для вычисления измененного выражения

16. Проверяем работоспособность

```

dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1

```

Рис. 3.16: Проверка работоспособности

17. Создадим файл под названием variant

```

dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/variant.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ 

```

Рис. 3.17: Создание файла variant

18. Вставка нужной программы

```

variant.asm [B---] 39 L:[ 1+15 16/ 25] *(391 / 490b) 0010 0x00A
#include 'in_out.asm'
SECTION .data
msg: DB 'Введите No 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 sprintf
mov eax, edx
call iprintLF
call quit

```

Рис. 3.18: Программа для вычисления варианта по номеру студ. билета

19. Проверка работоспособности

```
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ nasm -f elf variant.asm
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o variant variant.o
dvorlyuk@dk2n22 ~/work/arch-pc/lab06 $ ./variant
Введите No студенческого билета:
1132220823
Ваш вариант: 4
```

Рис. 3.19: Проверка работоспособности

#Ответы на вопросы

1. Какие строки листинга 7.4 отвечают за вывод на экран сообщения ‘Ваш вариант:’ Строчка `rem: DB “Ваш вариант: ”` отвечает за вывод сообщения.
2. Для чего используются следующие инструкции? `nasm mov ecx, x mov edx, 80 call sread`
3. Для чего используется инструкция “`call atoi`”? Функция `call atoi` преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`.
4. Какие строки листинга 7.4 отвечают за вычисления варианта? `mov eax, msg call sprintLF`
5. В какой регистр записывается остаток от деления при выполнении инструкции “`div ebx`”? Остаток записывается в регистр `ah`.
6. Для чего используется инструкция “`inc edx`”? Команда `inc edx` увеличивает значение регистра `edx` на
7. Какие строки листинга 7.4 отвечают за вывод на экран результата вычислений? `mov eax,rem call sprint mov eax,edx call iprintLF call quit`

4 Выводы

Я освоил арифметические инструкции языка ассемблера NASM, а также узнал как записывать в МС все действия с числами