

Лаборатору Отчет No6

ДЭВИД МАЙКЛ ФРАНСИС

Цель работы

Цель данной лабораторной работы - освоение арифметических инструкций языка ассемблера NASM.

Выполнение лабораторной работы

Символьные и численные данные в NASM

С помощью утилиты mkdir создаю директорию, в которой буду создавать файлы с программами для лабораторной работы №6. Перехожу в созданный каталог с помощью утилиты cd.

```
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06 $ cd "Архитектура компьютера/arch-pc/lab6"
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6 $ mkdir lab06
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6 $ cd lab06
```

Screenshot1

С помощью утилиты touch создаю файл lab6-1.asm

```
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6/lab06 $ touch lab6-1.asm
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6/lab06 $ godit lab6-1.asm
MESA: error: ZINK: failed to choose pdev
glx: failed to create drisw screen
```

Screenshot2

Открываю созданный файл lab7-1.asm, вставляю в него программу вывода значения регистра eax

```
GNU nano 7.2 lab6-1.asm *
#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
```

Screenshot3

Создаю исполняемый файл программы и запускаю его. Вывод программы: символ j, потому что программа вывела символ, соответствующий по системе ASCII сумме двоичных кодов символов 4 и 6

```
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6/lab06 $ nasm -f elf lab6-1.asm
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
(base) m1ke@DESKTOP-1HEKTC0: ~/lab06/arch-pc/lab6/lab06 $ ./lab6-1
j
```

Screenshot4

Изменяю в тексте программы символы “6” и “4” на цифры 6 и 4

```
GNU nano 7.2 lab6-1.asm
#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
```

Screenshot5

Создаю новый исполняемый файл программы и запускаю его. Теперь вывелся символ с кодом 10, это символ перевода строки, этот символ не отображается при выводе на экран

```
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ nano lab6-1.asm
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ nasm -f elf lab6-1.asm
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ ld -m elf_i386 -o lab6-1 lab6-1.o
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ ./lab6-1
```

Screenshot6

Создаю новый файл lab6-2.asm с помощью утилиты touch. Ввожу в файл текст другой программы для вывода значения регистра eax

```
GNU nano 7.2 lab6-2.asm *
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax,'6'
mov ebx,'4'
add eax,ebx
call iprint
call quit
```

Screenshot7

Создаю и запускаю исполняемый файл lab6-2.. Теперь вывод число 106, потому что программа позволяет вывести именно число, а не символ, хотя все еще происходит именно сложение кодов символов “6” и “4”

```
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ nasm -f elf lab6-2.asm
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ ld -m elf_i386 -o lab6-2 lab6-2.o
(base) mik@DESKTOP-1HEKTCO: /laboratory-work/Архитектура компьютера/arch-pc/labs/lab06$ ./lab6-2
106
```

Screenshot8

Заменяю в тексте программы в файле lab6-2.asm символы “6” и “4” на числа 6 и 4

```

GNU nano 7.2                               lab6-2.asm *
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

mov eax, 0
mov ebx, 4
add eax, ebx
call iprint
call quit

```

Screenshot9

Создаю и запускаю новый исполняемый файл. Теперь программа складывает не соответствующие символы коды в системе ASCII, а сами числа, поэтому вывод 10

```

(base) mike@DESKTOP-1BENTC01:~/Laboratory-work/Архитектура компьютера/arch-pc/Labs/Lab6$ ./lab6-2
10
(base)

```

Screenshot10

Выполнение арифметических операций в NASM

Создаю файл lab6-3.asm с помощью утилиты touch. Ввожу в созданный файл текст программы для вычисления значения выражения $f(x) = (5 * 2 + 3) / 3$

```

GNU nano 7.2                               lab6-3.asm
#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 EDI для корректной работы div
mov ebx, 3 ; EBX=3
div ebx ; EAX=EAX/3, EDI=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 iprintf ; 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 iprintf ; из 'edx' (Остаток) в виде символов
call quit ; вызов подпрограммы завершения

```

Screenshot11

Создаю исполняемый файл и запускаю его

```

(base) mike@DESKTOP-1BENTC01:~/Laboratory-work/Архитектура компьютера/arch-pc/Labs/Lab6$ nasm -f elf lab6-3.asm
(base) mike@DESKTOP-1BENTC01:~/Laboratory-work/Архитектура компьютера/arch-pc/Labs/Lab6$ ld -m elf_i386 -o lab6-3 lab6-3.o
(base) mike@DESKTOP-1BENTC01:~/Laboratory-work/Архитектура компьютера/arch-pc/Labs/Lab6$ ./lab6-3
Результат: 4
Остаток от деления: 1

```

Screenshot12

Создаю файл variant.asm с помощью утилиты touch. Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета

```

GNU nano 7.2 variant.asm
#include "in_out.asm"

SECTION .data
msg: DB "Введите % студенческого билета: ",0
rem: DB "Ваш вариант: ",0

SECTION .bss
r: 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 sprintf
call quit

```

Screenshot13

Создаю и запускаю исполняемый файл.. Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 4

```

(base) ~/lab01:~/lab01$ nasm -f elf variant.asm
(base) ~/lab01:~/lab01$ ld -o elf-1386 -o variant variant.o
(base) ~/lab01:~/lab01$ ./variant
Введите % студенческого билета:
183238983
Ваш вариант: 4

```

Screenshot14

Ответы на вопросы по программе

1. За вывод сообщения “Ваш вариант” отвечают строки кода:

```

mov eax, rem
call sprint

```

2. Инструкция `mov ecx, x` используется, чтобы положить адрес вводимой строки `x` в регистр `ecx` `mov edx, 80` - запись в регистр `edx` длины вводимой строки `call sread` - вызов подпрограммы из внешнего файла, обеспечивающей ввод сообщения с клавиатуры
3. `call atoi` используется для вызова подпрограммы из внешнего файла, которая преобразует `ascii`-код символа в целое число и записывает результат в регистр `eax`
4. За вычисления варианта отвечают строки:

```

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 `div ebx` остаток от деления записывается в регистр `edx`
6. Инструкция `inc edx` увеличивает значение регистра `edx` на 1
7. За вывод на экран результатов вычислений отвечают строки:

```

mov eax, edx
call iprintLF

```

Выполнение заданий для самостоятельной работы

Создаю файл lab6-4.asm с помощью утилиты touch. Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения $(11 + x) * 2 - 6$

```
GNU nano 7.2 lab6-4.asm
#include "in_out.asm" ; подключение внешнего файла
SECTION .text
msg DB 'Введите значение переменной x: ',0
res DB 'Результат: ',0
SECTION .bss
x RESB 80 ; Переменная, значение которой будет вводиться с клавиатуры
SECTION .text
GLOBAL _start
_start:

; ----- Вычисление выражения
mov eax, msg
call sprint
mov ecx, x
mov edx, 80
call sread
mov ebx, x
call atoi ; вызов подпрограммы преобразования
; REXX: код в число, 'eax=x'
add eax, 11 ; eax = eax + 11 = x + 11
mov ebx, 2
mul ebx ; EAX=EAX*EBX = (x+11)*2
add eax, -6 ; eax = eax - 6 = (x+11)*2 - 6
mov edi, eax ; запись результата вычисления в 'edi'

; ----- Вывод результата на экран
mov eax, res
call sprint ; вызов подпрограммы печати
; сообщение 'Результат: '
mov eax, edi ; вызов подпрограммы печати значения
call sprint ; из 'edi' в виде символов
call quit ; вызов подпрограммы завершения
```

Screenshot15

Создаю и запускаю исполняемый файл. При вводе значения 3, вывод - 22.

```
(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ nasm -f elf lab6-4.asm
(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ ld -m elf_i386 -o lab6-4.o lab6-4.o
(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-4
Введите значение переменной x: 3
Результат: 22(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-4
```

Screenshot16

Провожу еще один запуск исполняемого файла для проверки работы программы с другим значением на входе. Программа отработала верно

```
Результат: 22(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ ./lab6-4
Введите значение переменной x: 1
Результат: 18(base) mike@DESKTOP-1HEKTC01:~/laboratory-work/Архитектура компьютера/arch-pc/labs/lab6$ nano lab6-1.asm
```

Screenshot17

Выводы

При выполнении данной лабораторной работы я освоила арифметические инструкции языка ассемблера NASM.

Ссылка на официальный сайт [Github](#)