

Лабораторная работа №6

Арифметические операции в NASM.

Жукова Арина Александровна

Содержание

| | | |
|----------|---|-----------|
| 1 | Цель работы | 5 |
| 2 | Выполнение лабораторной работы | 6 |
| 2.1 | Символьные и численные данные в NASM | 6 |
| 2.2 | Выполнение арифметических операций в NASM | 11 |
| 2.2.1 | Ответы на вопросы | 16 |
| 2.3 | Выполнение заданий для самостоятельной работы | 17 |
| 3 | Выводы | 20 |
| | Список литературы | 21 |

Список иллюстраций

| | | |
|------|---|----|
| 2.1 | Создание файла | 6 |
| 2.2 | Ввод текста программы | 7 |
| 2.3 | Создание исполняемого файла и проверка его работы | 7 |
| 2.4 | Изменение текста программы | 8 |
| 2.5 | Создание и запуск исполняемого файла | 8 |
| 2.6 | Создание файла lab6-2.asm | 9 |
| 2.7 | Редактирование файла | 9 |
| 2.8 | Создание и проверка работы файла | 9 |
| 2.9 | Редактирование файла | 10 |
| 2.10 | Редактирование файла | 10 |
| 2.11 | Создание и проверка работы файла | 11 |
| 2.12 | Ввод текста программы | 12 |
| 2.13 | Создание и проверка работы файла | 13 |
| 2.14 | Редактирование файла | 14 |
| 2.15 | Создание и проверка работы файла | 15 |
| 2.16 | Ввод текста программы | 16 |
| 2.17 | Создание и проверка работы файла | 16 |
| 2.18 | Ввод текста программы | 18 |
| 2.19 | Создание и проверка работы файла | 19 |

Список таблиц

1 Цель работы

Целью данной лабораторной работы является освоение арифметических инструкций языка ассемблера NASM.

2 Выполнение лабораторной работы

2.1 Символьные и численные данные в NASM

1. Создаём файл lab6-1.asm в новом каталоге для программ лабораторной работы №6 (рис. 2.1).



Рис. 2.1: Создание файла

2. Вводим в файл lab6-1.asm текст программы из данного листинга 6.1 (рис. 2.2).

```

lab6-1.asm      [-M--] 10 L:[ 1+16 17/ 17] *(191 / 191b) <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. 2.2: Ввод текста программы

В программе в регистр `eax` записывается символ 6 ('6'), в `ebx` - 4 ('4'). Далее значения складываются, результат сложения записывается в регистр `eax`. Для вывода результата при помощи команды `sprintLF` необходимо, чтобы в регистре `eax` был записан адрес, для этого используем дополнительную переменную `buf1`.

Создаём исполняем файл и проверяем его работу (рис. 2.3).

```

aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-1

aazhukova1@dk5n52 ~/work/arch-pc/lab06 $

```

Рис. 2.3: Создание исполняемого файла и проверка его работы

При выводе программы отображается символ `j`, так как программа выводит символ, соответствующий сумме двоичных кодов символа 4 и 6 по системе ASCII.

3. Изменяем текст программы и вместо символов ('4' и '6'), запишем в регистры числа (4 и 6) (рис. 2.4)

```

lab6-1.asm      [-M--] 10 L:[ 1+16
%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. 2.4: Изменение текста программы

Создаём исполняемый файл и запускаем его (рис. 2.5).

```

aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-1.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-1 lab6-1.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-1

aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ 

```

Рис. 2.5: Создание и запуск исполняемого файла

Теперь выводится символ с кодом 10 - перевод строки. Этот символ выводится на экран пустой строкой.

4. Создаём новый файл lab6-2.asm при помощи команды touch (рис. 2.6).


```
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ touch ~/work/arch-pc/lab06/lab6-2.asm
```

Рис. 2.6: Создание файла lab6-2.asm

Вводим в него текст программы для вывода значения регистра eax (рис. 2.7).

```
lab6-2.asm [-M--] 13 L:[ 1+11 12/ 12] *(137
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

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

call quitLF
```

Рис. 2.7: Редактирование файла

Создаём и запускаем исполняемый файл (рис. 2.8).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-2
106
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $
```

Рис. 2.8: Создание и проверка работы файла

Программа выводит число 106, так как программа выводит сумму кодов символов '4' и '6'. Однако функция `iprintLF` позволяет вывести на экран число, а не символ, кодом которого является это число.

5. Заменяем символы на числа в тексте программы файла lab6-2.asm (рис. 2.9).

```
lab6-2.asm      [-M--] 13 L:[ 1+11 12/ 12] *(133
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

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

call quitLF
```

Рис. 2.9: Редактирование файла

Теперь программа складывает не коды, соответствующие символам, а сами числа, поэтому мы получаем вывод 10.

Заменяем в тексте программы функцию `iprintLF` на `iprint` (рис. 2.10).

```
/afs/.dk.sci.pfu.edu.ru/home/a/a/aazhukova1/work/arch
%include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:

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

call quit
```

Рис. 2.10: Редактирование файла

Далее создаём и запускаем исполняемый файл (рис. 2.11).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-2.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-2 lab6-2.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-2
10
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $
```

Рис. 2.11: Создание и проверка работы файла

2.2 Выполнение арифметических операций в NASM

6. Создадим файл lab6-3.asm в каталоге ~/work/arch-pc/lab06 и введем в него текст листинга 6.3 (рис. 2.12).

```

lab6-3.asm      [----]  0 L:[ 1+31 32/ 32]
#include      'in_out.asm'

SECTION .data

div:  DB 'Результат: ',0
rem:  DB 'Остаток от деления: ',0

SECTION .text
GLOBAL _start
_start:

    mov eax,5
    mov ebx,2
    mul ebx
    add eax,3
    xor edx,edx
    mov ebx,3
    div ebx

    mov edi,eax

    mov eax,div
    call sprint
    mov eax,edi
    call iprintLF

    mov eax,rem
    call sprint
    mov eax,edx
    call iprintLF

    call quit

```

Рис. 2.12: Ввод текста программы

Создаём исполняемый файл и запускаем его (рис. 2.13).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 4
Остаток от деления: 1
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $
```

Рис. 2.13: Создание и проверка работы файла

Изменим текст программы для вычисления выражения

$$f(x) = (4 * 6 + 2) / 5$$

(рис. 2.14).

```
lab6-3.asm      [----]  0 L:[ 1+30
%include      'in_out.asm'

SECTION .data

div:  DB 'Результат: ',0
rem:  DB 'Остаток от деления: ',0

SECTION .text
GLOBAL _start
_start:

    mov eax,4
    mov ebx,6
    mul ebx
    add eax,2
    xor edx,edx
    mov ebx,5
    div ebx

    mov edi,eax

    mov eax,div
    call sprint
    mov eax,edi
    call iprintLF

    mov eax,rem
    call sprint
    mov eax,edx
    call iprintLF

    call quit
```

Рис. 2.14: Редактирование файла

Создаём исполняемый файл и проверяем его работу (рис. 2.15).

```
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ nasm -f elf lab6-3.asm
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o lab6-3 lab6-3.o
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $ ./lab6-3
Результат: 5
Остаток от деления: 1
aazhukova1@dk5n52 ~/work/arch-pc/lab06 $
```

Рис. 2.15: Создание и проверка работы файла

7. Рассмотрим программу вычисления варианта задания по номеру студенческого билета, работающую по следующему алгоритму:

- вывести запрос на введение № студенческого билета
- вычислить номер варианта по формуле:

$$(Sn \bmod 20) + 1$$

, где S_n – номер студенческого билета (В данном случае $a \bmod b$ – это остаток от деления a на b).

- вывести на экран номер варианта.

Создаём файл `variant.asm` в каталоге `~/work/arch-pc/lab06`, вводим текст листинга 6.4 в файл `variant.asm` (рис. 2.16).

```

variant.asm      [----] 12 L:[ 1+32 33/ 33] *(462 / 462b) <EOF>
#include "in_out.asm"

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

SECTION .bss
x:   RESB 80

SECTION .bss
GLOBAL _start
_start:

mov eax, msg
call sprintf

mov ecx, x
mov edx, 80
call sread

mov eax, x
call atoi
xor edx, edx
mov ebx, 20
div ebx
inc edx

mov eax, rem
call sprintf
mov eax, edx
call iprintLF

call quit

```

Рис. 2.16: Ввод текста программы

Создаём исполняемый файл и запустим его (рис. 2.17).

```

aazhukova1@dk2n26 ~/work/arch-pc/lab06 $ ./variant
Введите № студенческого билета:
1132239120
Ваш вариант: 1

```

Рис. 2.17: Создание и проверка работы файла

2.2.1 Ответы на вопросы

1. За вывод на экран сообщения 'Ваш вариант:' отвечают строки:


```
mov eax, rem  
call sprint
```

2. Данные инструкции используются для: `mov ecx, x` - адрес вводимой строки `x` вкладывается в регистр `ecx` `mov edx, 80` - запись в регистр `call sread` вызов подпрограммы из внешнего файла, обеспечивающий ввод данных с клавиатуры.
3. `call atoi` используется для вызова подпрограммы из внешнего файла, который преобразует `ascii`-код символа в целое число и записывает результат в регистр.
4. За вычисление варианта отвечают строки:

```
xor edx, edx  
mov ebx, 20  
div ebx  
inc edx
```

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 результата вычислений отвечает строка:

```
mov eax, edx  
call iprintLF
```

2.3 Выполнение заданий для самостоятельной работы

Создаём файл `zadanie.asm` и вводим в него текст программы для вычисления выражения

$$y = (10 + 2x)/3$$

. Программа должна выводить выражение для вычисления, выводить запрос на ввод значения x , вычислять заданное выражение в зависимости от введенного x , выводить результат вычислений (рис. 2.18).

```
/afs/.dk.sci.pfu.edu.ru/home/a/a/aazhukova1/work/arch-pc/lab06
#include 'in_out.asm'

SECTION .data
div: DB 'Результат: ',0
msg: DB 'Введите значение x: ',0

SECTION .bss
x: RESB 80

SECTION .text
GLOBAL _start
_start:

mov eax, msg
call sprintLF

mov ecx, x
mov edx, 80

call sread

mov eax, x           ;EAX=x
call atoi
mov [x], eax
mov ebx, 2           ;EBX=x
mul ebx              ;EAX=EAX*EBX
add eax, 10          ;EAX=EAX+10
xor edx, edx         ;Обнуление EDX
mov ebx, 3           ;EBX=3
div ebx              ;EAX=EAX/3, EDX-остаток от деления

mov edi, eax

mov eax, div
call sprint
mov eax, edi
call iprintLF

call quit
```

Рис. 2.18: Ввод текста программы

Создаём исполняемый файл и запустим его (рис. 2.19).

```
aazhukova1@dk8n56 ~/work/arch-pc/lab06 $ nasm -f elf zadanie.asm
aazhukova1@dk8n56 ~/work/arch-pc/lab06 $ ld -m elf_i386 -o zadanie zadanie.o
aazhukova1@dk8n56 ~/work/arch-pc/lab06 $ ./zadanie
Введите значение x:
1
Результат: 4
aazhukova1@dk8n56 ~/work/arch-pc/lab06 $ ./zadanie
Введите значение x:
10
Результат: 10
```

Рис. 2.19: Создание и проверка работы файла

3 Выводы

При выполнении данной лабораторной работы нами были освоены арифметические инструкции языка ассемблера NASM.

Список литературы