

Отчет по лабораторной работе №6

Дисциплина: архитектура компьютера

Царёв Максим Александрович

Содержание

1	Цель работы	1
2	Задание	1
3	Теоретическое введение.....	1
4	Выполнение лабораторной работы.....	2
4.1	Символьные и численные данные в NASM.....	2
4.2	Выполнение арифметических операций в NASM.....	7
4.2.1	Ответы на вопросы по программе.....	11
4.3	Выполнение заданий для самостоятельной работы	11
5	Выводы.....	12

1 Цель работы

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

2 Задание

1. Символьные и численные данные в NASM
2. Выполнение арифметических операций в NASM
3. Выполнение заданий для самостоятельной работы

3 Теоретическое введение

Большинство инструкций на языке ассемблера требуют обработки операндов. Адрес операнда предоставляет место, где хранятся данные, подлежащие обработке. Это могут быть данные хранящиеся в регистре или в ячейке памяти. - Регистровая адресация – операнды хранятся в регистрах и в команде используются имена этих регистров, например: `mov ax,bx`. - Непосредственная адресация – значение операнда задается непосредственно в команде, Например: `mov ax,2`. - Адресация памяти –

операнд задает адрес в памяти. В команде указывается символическое обозначение ячейки памяти, над содержимым которой требуется выполнить операцию.

Ввод информации с клавиатуры и вывод её на экран осуществляется в символьном виде. Кодирование этой информации производится согласно кодовой таблице символов ASCII. ASCII – сокращение от American Standard Code for Information Interchange (Американский стандартный код для обмена информацией). Согласно стандарту ASCII каждый символ кодируется одним байтом. Среди инструкций NASM нет такой, которая выводит числа (не в символьном виде). Поэтому, например, чтобы вывести число, надо предварительно преобразовать его цифры в ASCII-коды этих цифр и выводить на экран эти коды, а не само число. Если же выводить число на экран непосредственно, то экран воспримет его не как число, а как последовательность ASCII-символов – каждый байт числа будет воспринят как один ASCII-символ – и выведет на экран эти символы. Аналогичная ситуация происходит и при вводе данных с клавиатуры. Введенные данные будут представлять собой символы, что сделает невозможным получение корректного результата при выполнении над ними арифметических операций. Для решения этой проблемы необходимо проводить преобразование ASCII символов в числа и обратно

4 Выполнение лабораторной работы

4.1 Символьные и численные данные в NASM

С помощью утилиты `mkdir` создаю директорию, в которой буду создавать файлы с программами, перехожу в директорию.

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/lab06
[fedora@fedora ~]$ cd work/study/2024-2025/Архитектура\ компьютеров/
[fedora@fedora Архитектура компьютеров]$ mkdir lab06
[fedora@fedora Архитектура компьютеров]$ cd lab06/
[fedora@fedora lab06]$
```

С помощью утилиты `touch` создаю файл `lab6-1.asm`

```
[fedora@fedora lab06]$ touch lab6-1.asm
[fedora@fedora lab06]$ ls
lab6-1.asm
[fedora@fedora lab06]$
```

Открываю созданный файл `lab6-1.asm`, вставляю в него программу вывода значения регистра `eax`

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/lab06 — nano lab6-1.asm
GNU nano 7.2 lab6-1.asm Modified
#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

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

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.

```
[fedora@fedora lab06]$ nasm -f elf lab6-1.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-1 lab6-1.o
mold: fatal: unknown -m argument: elf_i386
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-1 lab6-1.o
[fedora@fedora lab06]$ ./lab6-1
j
[fedora@fedora lab06]$
```

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

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano...
GNU nano 7.2 lab6-1.asm Modified
#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

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

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.

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_
[fedora@fedora lab06]$ nano lab6-1.asm
[fedora@fedora lab06]$ nasm -f elf lab6-1.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-1 lab6-1.o
[fedora@fedora lab06]$ ./lab6-1

[fedora@fedora lab06]$
```

Создаю новый файл lab6-2.asm с помощью утилиты touch

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2
[fedora@fedora lab06]$ touch lab6-2.asm
[fedora@fedora lab06]$ ls
in_out.asm lab6-1 lab6-1.asm lab6-1.o lab6-2.asm
[fedora@fedora lab06]$
```

Ввожу в файл текст другой программы для вывода значения регистра eax

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano...
GNU nano 7.2 lab6-2.asm Modified
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
    mov eax, '6'
    mov ebx, '4'
    add eax, ebx
    call iprintLF
    call quit

File Name to Write: lab6-2.asm
^G Help      M-D DOS Format  M-A Append     M-B Backup File
^C Cancel    M-M Mac Format  M-P Prepend    ^T Browse
```

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

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06$ nasm -f elf lab6-2.asm
fedora@fedora lab06$ mold -m elf_i386 -o lab6-2 lab6-2.o
fedora@fedora lab06$ ./lab6-2
106
fedora@fedora lab06$
```

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

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano...
GNU nano 7.2 lab6-2.asm Modified
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
    mov eax, 6
    mov ebx, 4
    add eax, ebx
    call iprintLF
    call quit

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

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

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025
[fedora@fedora lab06]$ nasm -f elf lab6-2.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-2 lab6-2.o
[fedora@fedora lab06]$ ./lab6-2
10
[fedora@fedora lab06]$
```

Заменяю в тексте программы функцию iprintLF на iprint

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano...
GNU nano 7.2 lab6-2.asm Modified
#include 'in_out.asm'

SECTION .text
GLOBAL _start
_start:
    mov eax, 6
    mov ebx, 4
    add eax, ebx
    call iprint
    call quit

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^_ Go To Line
```

Создаю и запускаю новый исполняемый файл. Вывод не изменился, потому что символ переноса строки не отображался, когда программа исполнялась с функцией `iprintLF`, а `iprint` не добавляет к выводу символ переноса строки, в отличие от `iprintLF`.

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06$ nasm -f elf lab6-2.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-2 lab6-2.o
[fedora@fedora lab06]$ ./lab6-2
10[fedora@fedora lab06]$
```

4.2 Выполнение арифметических операций в NASM

Создаю файл `lab6-3.asm` с помощью утилиты `touch`

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06$ touch lab6-3.asm
[fedora@fedora lab06]$ ls
in_out.asm  lab6-1.asm  lab6-2      lab6-2.o
lab6-1      lab6-1.o    lab6-2.asm  lab6-3.asm
[fedora@fedora lab06]$
```


Ввожу в созданный файл текст программы для вычисления значения выражения $f(x) = (5 * 2 + 3)/3$

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano lab6-3.asm
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
    mov ebx, 2
    mul ebx
    add eax, 3
    xor edx, edx
    mov ebx, 3
    div ebx
    mov edi, eax ; запись результата вычисления в 'edi'

    ; Вывод результата на экран
    mov eax, div ; вывод подпрограммы печати
    call sprint ; сообщения 'Результат: '
    mov eax, edi ; вызов подпрограммы печати значения
    call iprintLF ; из 'edi' в виде символов

    mov eax, rem ; вывод 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
```

Создаю исполняемый файл и запускаю его

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025
[fedora@fedora lab06]$ nasm -f elf lab6-3.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-3 lab6-3.o
[fedora@fedora lab06]$ ./lab6-3
Результат: 4
Остаток от деления: 1
[fedora@fedora lab06]$
```

Изменяю программу так, чтобы она вычисляла значение выражения $f(x) = (4 * 6 + 2)/5$


```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06 — nano lab6-3.asm
GNU nano 7.2 lab6-3.asm Modified
#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

^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location  M-U Undo
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line M-E Redo
```

Создаю и запускаю новый исполняемый файл, программа выполняется верно

```
[fedora@fedora lab06]$ nasm -f elf lab6-3.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o lab6-3 lab6-3.o
[fedora@fedora lab06]$ ./lab6-3
Результат: 5
Остаток от деления: 1
[fedora@fedora lab06]$
```

Создаю файл variant.asm с помощью утилиты touch.

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06
[fedora@fedora lab06]$ touch variant.asm
[fedora@fedora lab06]$ ls
in_out.asm  lab6-1.asm  lab6-2      lab6-2.o  lab6-3.asm  variant.asm
lab6-1      lab6-1.o   lab6-2.asm  lab6-3    lab6-3.o
[fedora@fedora lab06]$
```

Ввожу в файл текст программы для вычисления варианта задания по номеру студенческого билета

```
%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 sprintf
mov eax, edx
call iprintLF
```

Создаю и запускаю исполняемый файл. Ввожу номер своего студ. билета с клавиатуры, программа вывела, что мой вариант - 6

```
[fedora@fedora lab06]$ nasm -f elf variant.asm
[fedora@fedora lab06]$ mold -m elf_i386 -o variant variant.o
[fedora@fedora lab06]$ ./variant
Введите № студенческого билета:
1132241585
Ваш вариант: 6
```

4.2.1 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
```

4.3 Выполнение заданий для самостоятельной работы

Создаю файл lab6-4.asm с помощью утилиты touch

```
fedora@fedora:~/work/study/2024-2025/Архитектура компьютеров/study_2024-2025_ar-h-pc/lab06
[fedora@fedora lab06]$ touch lab6-4.asm
[fedora@fedora lab06]$ ls
in_out.asm  lab6-2  lab6-3.asm  variant
lab6-1  lab6-2.asm  lab6-3.o  variant.asm
lab6-1.asm  lab6-2.o  lab6-4.asm  variant.o
lab6-1.o  lab6-3  qemu_variant_20241111-133057_11198.core
[fedora@fedora lab06]$
```

Открываю созданный файл для редактирования, ввожу в него текст программы для вычисления значения выражения $(8 - 6) / 2$

```
GNU nano 7.2 lab6-4.asm Modified
%include 'in_out.asm' ; подключение внешнего файла

SECTION .data ; секция инициализированных данных
msg: DB 'Введите значение переменной x: ',0
res: DB 'Результат: ',0

SECTION .bss ; секция неинициализированных данных
x: RESB 80 ; Переменная для хранения значения, вводимого с клавиатуры

SECTION .text ; Код программы
GLOBAL _start ; Начало программы
_start: ; Точка входа в программу

; ---- Ввод значения x ----
mov eax, msg ; Адрес выводимого сообщения в eax
call sprint ; Вызов подпрограммы для печати сообщения
mov ecx, x ; Запись адреса переменной x в ecx
mov edx, 80 ; Указание длины вводимого значения в edx
call sread ; Вызов подпрограммы для ввода значения

mov eax, x ; Преобразование ASCII в число
call atoi ; Преобразование ASCII кода в число, теперь `eax = x`

; ---- Вычисление выражения (8x - 6) / 2 ----
mov ebx, 8 ; Установка значения 8 в ebx
mul ebx ; Умножение: EAX = EAX * EBX = x * 8
sub eax, 6 ; Вычитание 6: EAX = EAX - 6 = 8x - 6

mov ebx, 2 ; Установка значения 2 в ebx для деления
cdq ; Расширение EAX в EDX:EAX для корректной работы div
div ebx ; Деление: EAX = (8x - 6) / 2

; ---- Вывод результата ----
mov eax, res ; Адрес строки 'Результат: ' в eax
call sprint ; Вызов подпрограммы для печати сообщения

^G Help ^O Write Out ^W Where Is ^K Cut ^T Execute ^C Location M-U Undo
^X Exit ^R Read File ^\ Replace ^U Paste ^J Justify ^_ Go To Line M-E Redo
```

Создаю и запускаю исполняемый файл, при вводе значения 1, вывод 1. При вводе значения 5, вывод 17. Программа отработала верно.

5 Выводы

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