



Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ
КАФЕДРА КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)
НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и Вычислительная техника

О т ч е т
по лабораторной работе № 2

Дисциплина: Машинно-зависимые языки и основы компиляции

Название лабораторной работы: Программирование целочисленных
вычислений

Студент гр. ИУ6-416 _____ И.С. Марчук
(Подпись, дата) (И.О. Фамилия)

Преподаватель _____ Кузнецов Н.О.
(Подпись, дата) (И.О. Фамилия)

Москва, 2020

Цель работы:

Изучение форматов машинных команд, команд целочисленной арифметики ассемблера и программирование целочисленных вычислений.

Задание:

1) Разработать программу, вычисляющую заданное выражение. Просмотреть в отладчике и зафиксировать в отчете ход выполнения вычислений (покомандно). Убедиться в правильности программы.

2) Посмотреть в отладчике форматы 3-4 команд mov и расшифровать двоичные коды этих команд, используя материалы теоретической части.

Вариант 16: $n = (q^2)/3 - a*d + 5$.

Текст программы с комментариями:

; Lab 02 Marchuk

.586

.MODEL flat, stdcall

OPTION CASEMAP:NONE

Include kernel32.inc

Include masm32.inc

IncludeLib kernel32.lib

IncludeLib masm32.lib

.CONST

MsgExit DB 0AH, 0DH, 0AH, 0DH, "Press Enter to Exit", 0AH, 0DH, 0

helloText DB "Please, enter a, d, q", 0AH, 0DH, 0

endText DB "(q^2)/3 - a*d + 5 = ", 0

strEnd DB 0AH, 0DH, 0

.DATA

.DATA?

myInBuffer DB 10 DUP (?)

vA DWORD ?

vD DWORD ?

vQ DWORD ?

ans DWORD ?

.CODE

Start:

Invoke StdOut, ADDR helloText

; ввод числа A

Invoke StdIn, ADDR myInBuffer, LengthOf myInBuffer

Invoke StripLF, ADDR myInBuffer ; Преобразование в SDWORD

Invoke atoi, ADDR myInBuffer ; результат в EAX

mov vA, EAX

; ввод числа D

Invoke StdIn, ADDR myInBuffer, LengthOf myInBuffer

Invoke StripLF, ADDR myInBuffer ; Преобразование в SDWORD

Invoke atoi, ADDR myInBuffer ; результат в EAX

mov vD, EAX

; ввод числа Q

Invoke StdIn, ADDR myInBuffer, LengthOf myInBuffer

Invoke StripLF, ADDR myInBuffer ; Преобразование в SDWORD

Invoke atoi, ADDR myInBuffer ; результат в EAX

mov vQ, EAX

```

; расчеты
mul EAX      ;(q^2) EDX:EAX = EAX*EAX

mov ECX, 3
CDQ
div ECX      ;EAX = (EDX:EAX)/ECX
mov ECX, EAX ; запомнили результат

mov EAX, vA
mul vD       ; EDX:EAX = EAX*vD

sub ECX, EAX ; ECX = ECX - EAX
add ECX, 5   ; ECX = ECX + 5
mov ans, ECX

; ВЫВОД ОТВЕТА
Invoke StdOut, ADDR endText      ; вывод описания ответа
    Invoke dwtoa, ans, ADDR myInBuffer ; преобразование ответа-числа в
    строку
Invoke StdOut, ADDR myInBuffer    ; вывод
Invoke StdOut, ADDR strEnd

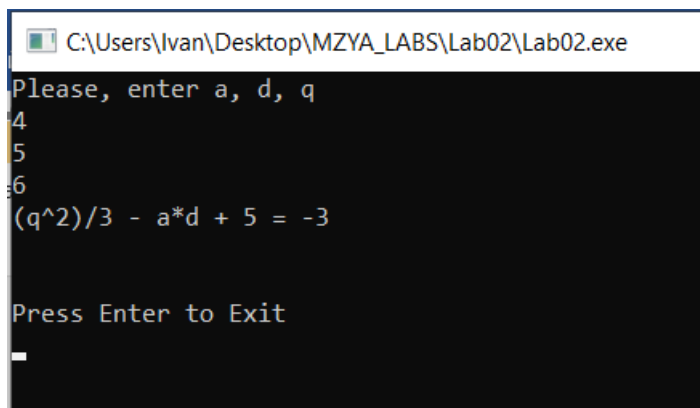
XOR  EAX, EAX
Invoke StdOut, ADDR MsgExit
Invoke StdIn, ADDR myInBuffer, LengthOf myInBuffer

Invoke ExitProcess, 0
End Start

```

Запуск программы на выполнение:

Результат работы программы представлен на рисунке 1, в начале производится ввод чисел, а затем вывод ответа в полной форме.



```
C:\Users\Ivan\Desktop\MZYA_LABS\Lab02\Lab02.exe
Please, enter a, d, q
4
5
6
(q^2)/3 - a*d + 5 = -3
Press Enter to Exit
```

Рисунок 1 – Выполнение программы

Работа программы с тестовыми данными приведена в таблице 1:

Исходные данные	Ожидаемый результат	Полученный результат
a=4 d=5 q=6	-3	$(q^2)/3 - a*d + 5 = -3$
a=6 d=5 q=4	-19.66667	$(q^2)/3 - a*d + 5 = -20$
a=12 d=2 q=10	14.333333	$(q^2)/3 - a*d + 5 = 14$
a=100 d=10 q=5	-986.66667	$(q^2)/3 - a*d + 5 = -987$
a=-100 d=-10 q=-5	-986.66667	$(q^2)/3 - a*d + 5 = -987$
a=0 d=0 q=0	5	$(q^2)/3 - a*d + 5 = 5$
a=3 d=1000 q=-2	-2994.33333	$(q^2)/3 - a*d + 5 = -2994$
a=9 d=7 q=200	13275.33333	$(q^2)/3 - a*d + 5 = 13275$
a=42 d=33 q=43	-764.66667	$(q^2)/3 - a*d + 5 = -765$
a=0 d=0 q=10	38.33333	$(q^2)/3 - a*d + 5 = 38$

Таблица 1 – Отладка программы

Ход выполнения вычислений:

Ниже представлен код вычислений на языке ассемблера, а на рисунках 2 и 3 скриншоты выполнения этого кода в отладчике.

mul EAX ;(q^2) EDX:EAX = EAX*EAX

mov ECX, 3

CDQ

div ECX ;EAX = (EDX:EAX)/ECX

mov ECX, EAX ; запомнили результат

mov EAX, vA

mul vD ; EDX:EAX = EAX*vD

sub ECX, EAX ; ECX = ECX - EAX

add ECX, 5 ; ECX = ECX + 5

mov ans, ECX

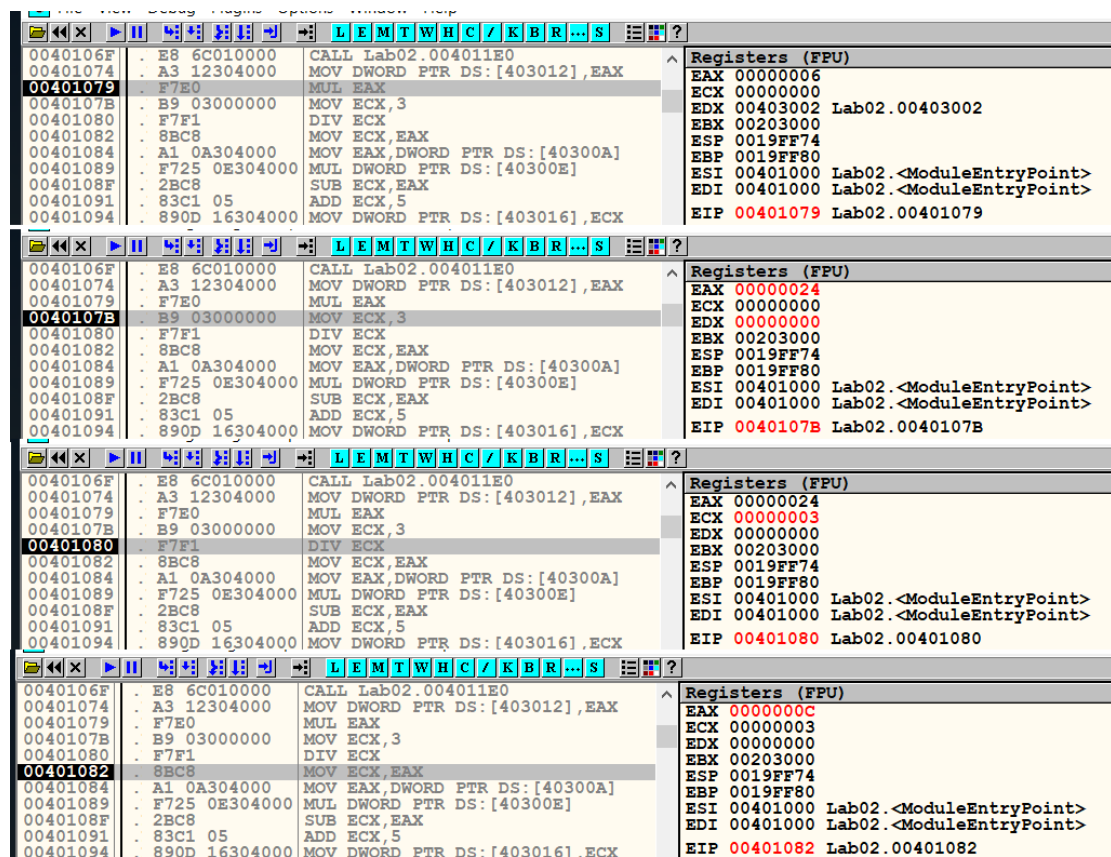


Рисунок 2 – Выполнение первых трех команд вычислений

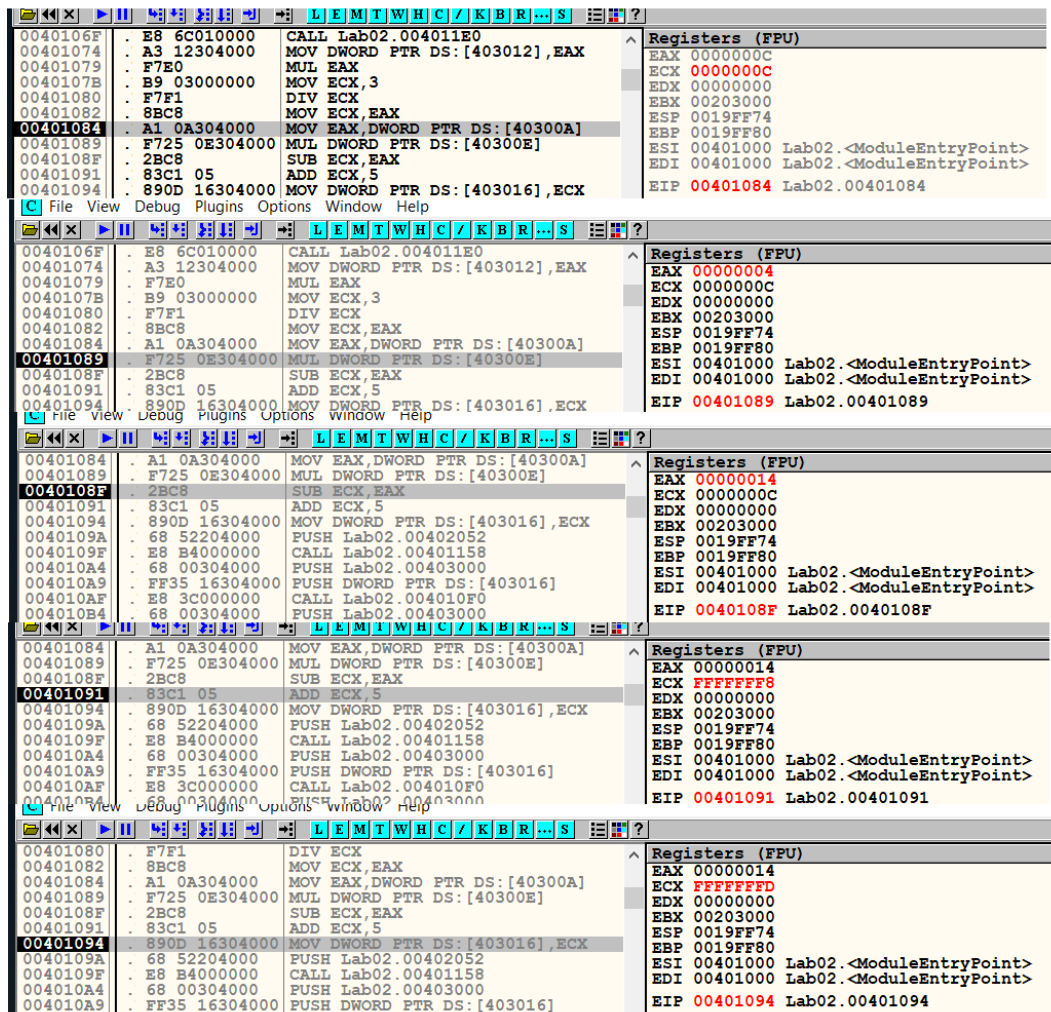


Рисунок 3 – Выполнение остальных четырех команд вычислений

В итоге в регистре ECX получаем ответ -3 (что видно на третьем рисунке, в самом последнем случае: 0xFFFFFFFFD), который затем выводится.

Расшифровка нескольких команд mov:

- **mov ECX, EAX**

dw mod reg reg

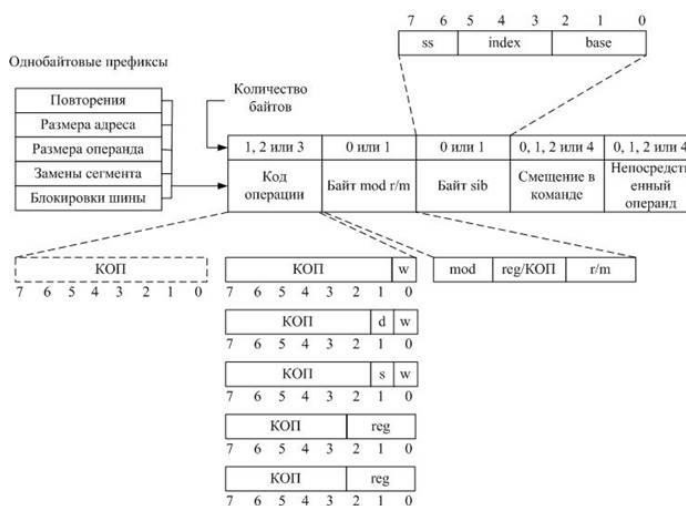
1000 1011 11 001 000

8 b c 8

- **mov EAX, vA**

8 9 0 D 1 6 3 0 4 0 0 0

Это элементарная инструкция компьютеру. Машинная команда состоит из двух частей: операционной и адресной. Операционная часть команды – это группа разрядов в команде, предназначенная для указания кода операции. Структура команды mov представлена на рисунке 4



```
mov    reg, reg
```


mov mem, reg

mov reg, mem

- Работа со стеком

PUSH imm16 / imm32 / r16 / r32 / m16 / m32

POP r16 / r32 / m16 / m32

- инкремент/декремент

INC reg/mem

DEC reg/mem

Сложение и вычитание производится с двумя операндами, а вот команды деления и умножения принимают только один операнд, который в последствии умножается на значение, помещенное заранее в EAX/AX/AL после чего результат помещается в EAX.

3) Сформулируйте основные правила построения линейной программы вычисления заданного выражения.

В программе все операторы выполняются последовательно, один за другим, при вычислении результат записывается на место одного из операндов.

4) Почему ввод-вывод на языке ассемблера не программируют с использованием соответствующих машинных команд? Какая библиотека используется для организации ввода-вывода в данной лабораторной?

Машинные команды сложны для чтения и отладки человеком, по этому используются их удобно читаемые аналоги – команды ассемблера.

В данной лабораторной используются команды ввода вывода стандартной библиотеки среды RADASM32.

5) Расскажите, какие процедуры используют для организации ввода вывода. Какие операции выполняет каждая процедура?

- Процедура ввода:

StdIn PROC lpszBuffer:DWORD, bLen:DWORD

- Процедура замены символов конца строки нулем:

StripLF PROC lpszBuffer:DWORD

- Функция преобразования завершающейся нулем строки в число:

atoi proc lpszBuffer:DWORD

- Процедура вывода завершающейся нулем строки в окно консоли:

StdOut PROC lpszBuffer:DWORD ; буфер вывода, зав. Нулем

- Процедура преобразования числа в строку:

dwtoa PROC public dwValue:DWORD, lpBuffer:PTR BYTE

Вывод:

Я изучил форматы машинных команд, команд целочисленной арифметики ассемблера и программирование целочисленных вычислений. А также выполнил практическое задание по этим темам.

