Отчёт по лабораторной работе №7

Арифметические операции в NASM

Мулин Иван Владимирович

Содержание

# 1 Цель работы

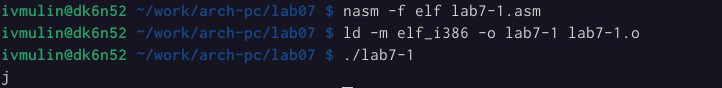
Цель выполнения лабораторной работы № 7 - изучить арифметические операции в языке ассемблера NASM. Репозиторий github можно найти по адресу <https://github.com/ivmulin/study_2022-2023_arch-pc>.

# 2 Ход работы

## 2.1 Выполнение лабораторной работы

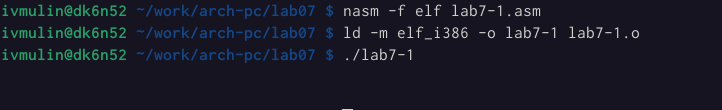
Вычислим значение выражения

В рабочей директории напишем программу, складывающую коды символов в двоичном представлении и проверяем её работу:



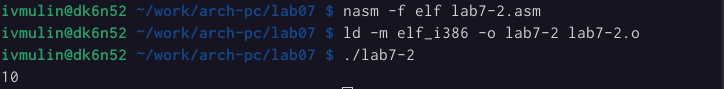
Запуск программы 1, дубль 1

В изменённой версии в консоль выводится символ с кодом - символ перевода строки в таблице ASCII:



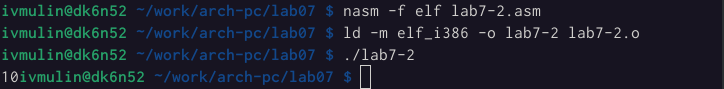
Запуск программы 1, дубль 2

Программа lab7-2 выводит ожидаемый нами результат:



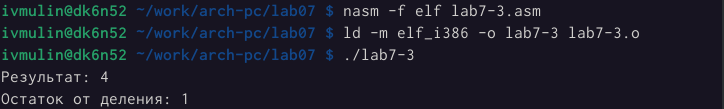
Запуск программы 2, дубль 1

Заменим в программе функцию iprintLF на iprint, которая выводит число без перевода строки:



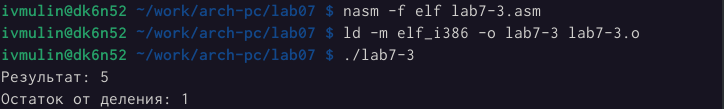
Запуск программы 2, дубль 2

Программа lab7-3 вычисляет значение выражения



Значение выражения (5 \* 2 + 3)/3

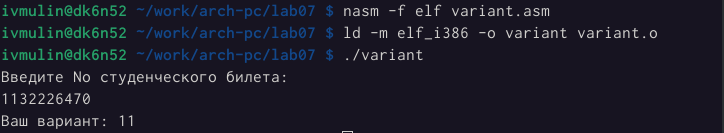
Изменённая версия этой же программы должна вычислять значение числа :



Значение выражения (4 \* 6 + 2)/5

Программа variant вычисляет номер варианта самостоятельной работы по формуле

Очевидно, программа работает корректно:



Вычисление номера варианта

### 2.1.1 Анализ программы variant

В программе variant строки

mov eax, rem  
 call sprint

отвечают за вывод в консоль надписи “Ваш вариант:”.

Строки

mov ecx, x  
 mov edx, 80  
 call sread

отвечают за ввод значения переменной x.

Инструкция call atoi преобразует значение регестра eax из кода ASCII в число.

Непосредственно номер варианта вычисляют следующие строки:

mov ebx, 20  
 div ebx  
 inc edx

Остаток от целочисленного денления помещается в регистр edx.

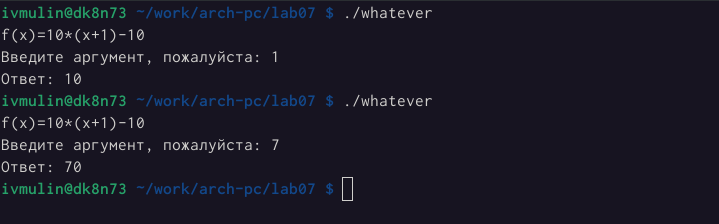
Команда inc edx увеличивает на единицу значение в соответствующем регистре.

Результат вычислений осуществляется посредством строк

mov eax, edx  
 call iprintLF

## 2.2 Выполнение заданий для самостоятельной работы

В ходе самостоятельной работы необходимо написать программу, вычисляющую значение многочлена , причём значение аргумента нужно получить от пользователя. В качестве входных значений используем сначала 1, затем 7:



Запуск программы для самостоятельной работы

Как видно, программа работает корректно.

# 3 Листинги написанных программ

1. lab7-1.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 sprintLF  
  
 call quit

1. lab7-2.asm

%include 'in\_out.asm'  
  
section .text  
 global \_start  
  
\_start:  
 mov eax, 6  
 mov ebx, 4  
 add eax, ebx  
 call iprint  
  
 call quit

1. lab7-3.asm

%include 'in\_out.asm'  
  
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  
  
 call quit ; вызов подпрограммы завершения

1. variant.asm

%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 sprintLF  
 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 iprintLF  
  
 call quit

1. whatever.asm

; f(x)=10\*(x+1)-10, 1, 7  
  
%include 'in\_out.asm'  
  
section .data  
 function: db 'f(x)=10\*(x+1)-10', 0  
 msg: db 'Введите аргумент, пожалуйста: ', 0  
 ans: db 'Ответ: ', 0  
  
section .bss  
 x: resb 80  
  
section .text  
 global \_start  
  
\_start:  
 mov eax, function  
 call sprintLF  
  
 mov eax, msg  
 call sprint  
  
 mov ecx, x  
 mov edx, 80  
 call sread  
 mov eax, x  
 call atoi ; eax=x  
  
 inc eax  
 mov ebx, 10  
 mul ebx  
 sub eax, 10  
 mov edx, eax  
  
 mov eax, ans  
 call sprint  
  
 mov eax, edx  
 call iprintLF  
  
 call quit

# 4 Заключение

Цель даннной лабораторной работы была достигнута, потому как были изучены арифметические операции (сложение, вычитание, умножение, деление, домножение на -1) на языке NASM.