**Федеральное государственное образовательное бюджетное учреждение  
высшего образования**

**«Финансовый университет при Правительстве Российской Федерации»**

**КОЛЛЕДЖ ИНФОРМАТИКИ И ПРОГРАММИРОВАНИЯ**

|  |  |
| --- | --- |
|  | |
|  |  |

**ОТЧЕТ**

**по практическому занятию**

**Машинно-ориентированное программирование для решения задач защиты информации**

**Студент:** Неделько Владислава Евгеньевна

**Специальность: 10.02.05 Обеспечение информационной безопасности автоматизированных систем**

**Группа: *3ОИБАС-718***

Оценка:\_\_\_\_\_\_\_\_\_\_\_\_

**Москва**

**2020 г.**

**Содержание**

[Задание 1 3](#_Toc54097784)

[Задание 2 5](#_Toc54097785)

# Задание 1

Реализовать сложение двух 64-разрядных чисел.

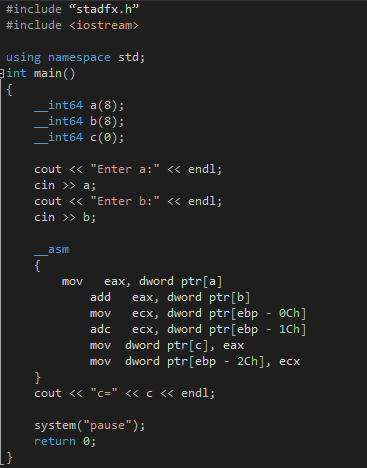


Рис. 1

Команды “add” и “adc” служат для реализации операций сложения чисел в различных вариациях. В качестве операндов для команд сложения могут выступать как регистры, так и ячейки памяти и конкретные значения. Команда “add” производит простое суммирование операндов. Команда “adc” производит суммирование операндов и значения флага “cf”, что позволяет производить суммирование больших чисел, которые не помещающихся в указанные размеры операндов.

Код:

#include “stdafx.h”

#include <iostream>

using namespace std;

int main()

{

\_\_int64 a(8);

\_\_int64 b(8);

\_\_int64 c(0);

cout << "Enter a:" << endl;

cin >> a;

cout << "Enter b:" << endl;

cin >> b;

\_\_asm

{

mov eax, dword ptr[a]

add eax, dword ptr[b]

mov ecx, dword ptr[ebp - 0Ch]

adc ecx, dword ptr[ebp - 1Ch]

mov dword ptr[c], eax

mov dword ptr[ebp - 2Ch], ecx

}

cout << "c=" << c << endl;

system("pause");

return 0;

}

Дизассемблерный код:

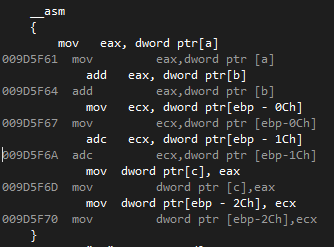


Рис.2

# Задание 2

Показать на примере реализацию команд умножения и деления командами 32-разрядного процессора (формат посмотреть в окне дизассемблера).

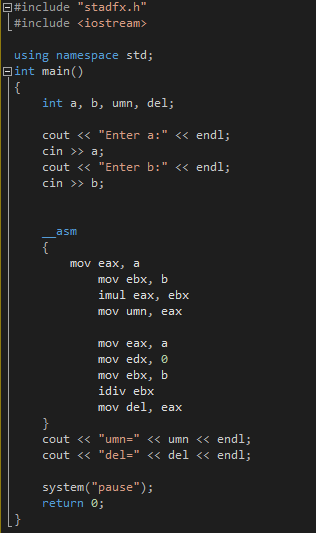


Рис. 3

Код:

#include "stdafx.h"

#include <iostream>

using namespace std;

int main()

{

int a, b, umn, del;

cout << "Enter a:" << endl;

cin >> a;

cout << "Enter b:" << endl;

cin >> b;

\_\_asm

{

mov eax, a

mov ebx, b

imul eax, ebx

mov umn, eax

mov eax, a

mov edx, 0

mov ebx, b

idiv ebx

mov del, eax

}

cout << "umn=" << umn << endl;

cout << "del=" << del << endl;

system("pause");

return 0;

}

Дизассемблерный код:

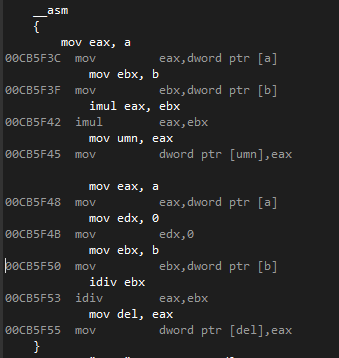


Рис.4