Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра информатики

Дисциплина «Архитектура вычислительных систем»

**ОТЧЕТ**

к практической работе №3

на тему:

**«ТЕХНОЛОГИЯ MMX»**

БГУИР 1-40 04 01

|  |
| --- |
| Выполнил студент группы 253502  Альховик Данила Игоревич |
| (дата, подпись студента) |
| Проверила ассистент кафедры информатики  Калиновская Анастасия Александровна |
| (дата, подпись преподавателя) |

Минск 2024

**Цель работы:** Задание 1. Вариант 2. Обработать массивы из 8 элементов по следующему выражению.

𝐹[𝑖] = (𝐴[𝑖] + 𝐵[𝑖]) \* 𝐶[𝑖] - 𝐷[𝑖], 𝑖 = 1.. 8

Используются следующие массивы: A, B и С – 8 разрядные целые знаковые числа (\_\_int8); D – 16 разрядные целые знаковые числа (\_\_int16). Полученный результат отобразить на форме с использованием соответствующих элементов

**Ход работы:** На рисунке 1 представлено значение в регистрах перед выполнением программы. На рисунке 2 представлено значение в регистрах после выполнения программы.

Листинг 1 – Исходный код программы задания

#include <iostream>

#include <mmintrin.h>

#pragma GCC target("mmx")

int main()

{

\_\_m64 a = \_mm\_set\_pi8(8, 7, 6, 5, 4, 3, 2, 1);

\_\_m64 b = \_mm\_set\_pi8(-1, -2, -3, -4, -5, -6, -7, -8);

\_\_m64 cHigh = \_mm\_set\_pi16(-8, -7, -6, -5);

\_\_m64 cLow = \_mm\_set\_pi16(-4, -3, -2, -1);

\_\_m64 dHighHigh = \_mm\_set\_pi32(1, 2);

\_\_m64 dHighLow = \_mm\_set\_pi32(3, 4);

\_\_m64 dLowHigh = \_mm\_set\_pi32(5, 6);

\_\_m64 dLowLow = \_mm\_set\_pi32(7, 8);

// F[i] = (A[i] + B[i]) \* C[i] - D[i]

// 1. A[i] + B[i]

\_\_m64 res = \_mm\_adds\_pi8(a, b);

int8\_t\* resArray = (int8\_t\*)&res;

\_\_m64 resLow = \_mm\_set\_pi16(resArray[3], resArray[2], resArray[1], resArray[0]);

\_\_m64 resHigh = \_mm\_set\_pi16(resArray[7], resArray[6], resArray[5], resArray[4]);

// 2. (A[i] + B[i]) \* C[i]

// Low

\_\_m64 resMulLowLowHigh = \_mm\_mulhi\_pi16(resLow, cLow);

\_\_m64 resMulLowLowLow = \_mm\_mullo\_pi16(resLow, cLow);

\_\_m64 resLowLow = \_mm\_unpacklo\_pi16(resMulLowLowLow, resMulLowLowHigh);

\_\_m64 resLowHigh = \_mm\_unpackhi\_pi16(resMulLowLowLow, resMulLowLowHigh);

// High

\_\_m64 resMulHighLowHigh = \_mm\_mulhi\_pi16(resHigh, cHigh);

\_\_m64 resMulHighLowLow = \_mm\_mullo\_pi16(resHigh, cHigh);

\_\_m64 resHighLow = \_mm\_unpacklo\_pi16(resMulHighLowLow, resMulHighLowHigh);

\_\_m64 resHighHigh = \_mm\_unpackhi\_pi16(resMulHighLowLow, resMulHighLowHigh);

// 3. (A[i] + B[i]) \* C[i] - D[i]

\_\_m64 resFinalLowLow = \_mm\_sub\_pi32(resLowLow, dLowLow);

\_\_m64 resFinalLowHigh = \_mm\_sub\_pi32(resLowHigh, dLowHigh);

\_\_m64 resFinalHighLow = \_mm\_sub\_pi32(resHighLow, dHighLow);

\_\_m64 resFinalHighHigh = \_mm\_sub\_pi32(resHighHigh, dHighHigh);

// Output

int32\_t\* arrLowLow = (int32\_t\*)&resFinalLowLow;

int32\_t\* arrLowHigh = (int32\_t\*)&resFinalLowHigh;

int32\_t\* arrHighLow = (int32\_t\*)&resFinalHighLow;

int32\_t\* arrHighHigh = (int32\_t\*)&resFinalHighHigh;

std::cout << "Result:\n";

for (size\_t i = 0; i < 2; i++)

{

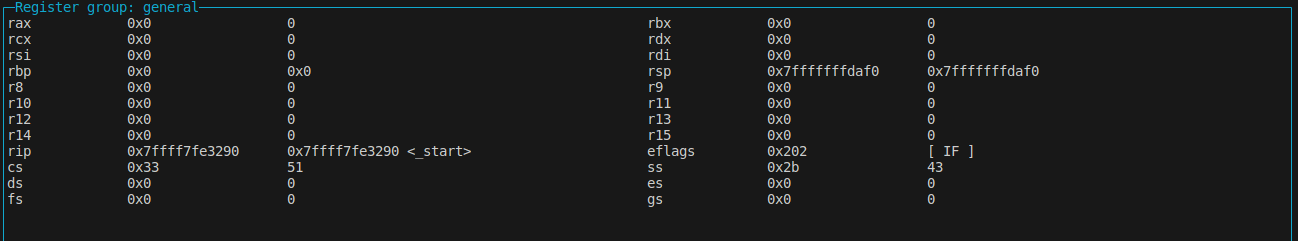
std::cout << arrLowLow[i] << " " << arrLowHigh[i] << " " << arrHighLow[i] << " " << arrHighHigh[i] << " ";

}

std::cout << std::endl;

return 0;

}

Рисунок 1 – Значения регистров программы перед выполнением

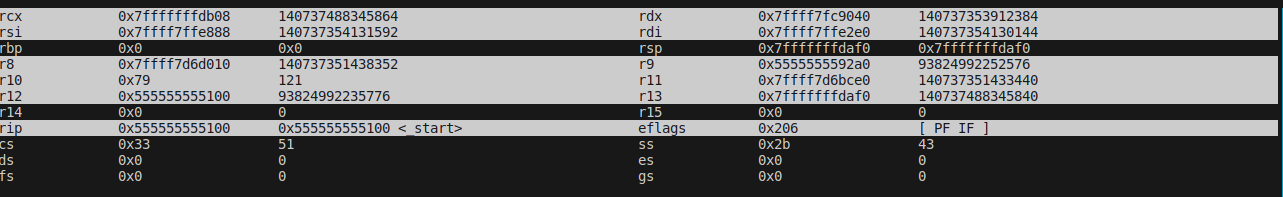


Рисунок 2 – Значения регистров программы после выполнения

Результат выполнения программы

Result:

-1 3 -9 -37 3 -1 -21 -57