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 лабораторной работе №4

на тему:

«ПРОГРАММИРОВАНИЕ С ИСПОЛЬЗОВАНИЕМ РАСШИРЕНИЙ SSE/SSE2»

БГУИР 1-40 04 01

Выполнил студент группы 253505
Снежко Максим Андреевич

(дата, подпись студента)

Проверил ассистент кафедры информатики
Калиновская Анастасия Александровна

(дата, подпись преподавателя)

Минск 2024

Цель работы: Вариант 20. Изучить программную модель MMX, изучить систему команд MMX, обработать массивы из 8 элементов по следующему выражению: F[i]=A[i] + (B[i] * C[i]) -D[i], i=1...8.

Теоретические сведения: SSE включает в архитектуру процессора восемь 128-битных регистров xmm0…xmm7, каждый из которых трактуется как 4 последовательных значения с плавающей точкой одинарной точности. Расширение позволяет выполнять векторные (пакетные) и скалярные инструкции. *Векторные инструкции* реализуют операции сразу над четырьмя комплектами операндов. *Скалярные инструкции* работают только с одним комплектом операндов – младшим 32-битным словом.

Реализация блоков SIMD осуществляется распараллеливанием вычислительного процесса между данными. То есть когда через один блок проходит поочередно множество потоков данных.

SSE2 (англ. Streaming SIMD Extensions 2, потоковое SIMD-расширение процессора) – это SIMD (англ. Single Instruction, Multiple Data, Одна инструкция – множество данных) набор инструкций, разработанный Intel , и впервые представленный в процессорах серии Pentium 4.

SSE2 использует те же восемь 128-битных регистров xmm0…xmm7 что и расширение SSE, каждый из которых трактуется как 2 последовательных значения с плавающей точкой двойной точности. SSE2 включает в себя набор инструкций, которые производят операции со скалярными и упакованными типами данных. Также SSE2 содержит инструкции для потоковой обработки целочисленных данных в тех же 128-битных xmm регистрах, что делает это расширение более предпочтительным для целочисленных вычислений, нежели использование набора инструкций MMX.

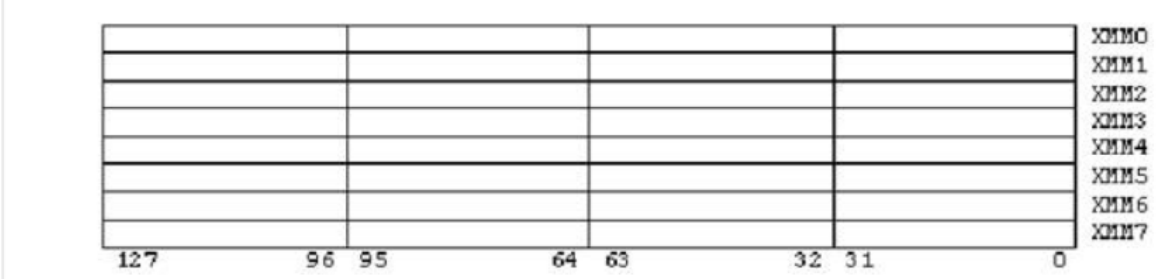

Рисунок 1 – Регистры SSE/SSE2

Типы данных SSE/SSE2

Новые расширения микропроцессора дополняют уже имеющиеся типы данных новыми упакованными типами:

- 4 упакованных вещественных числа одинарной точности;
- 2 упакованных вещественных числа двойной точности;
- 16 упакованных целых байтов;
- 8 упакованных целых слов;
- 4 упакованных целых двойных слова;
- 2 упакованных целых учетверенных слова.

Команды SSE

Команды SSE делятся на 4 категории:

- SIMD-команды для данных одинарной точности с плавающей запятой (SPFP-команды);
- дополнительные SIMD-команды для целочисленных данных;
- команды управления кэшированием;
- команды сохранения и восстановления компонент состояния процессора.

Одна SIMD-команда с плавающей запятой может обрабатывать одновременно четыре 32-разрядных числа одинарной точности с плавающей запятой (называемых SPFP-элементами данных). Каждое 32-разрядное число с плавающей запятой имеет 1 знаковый бит, 8 битов порядка и 23 бита мантиссы, что соответствует стандарту IEEE-754 на формат представления чисел одинарной точности с плавающей запятой. SIMD-команды поддерживают два типа операций над упакованными данными с плавающей запятой - параллельные и скалярные. Параллельные операции, как правило, действуют одновременно на все четыре 32-разрядных элемента данных в каждом из 128-разрядных операндов. В именах команд, выполняющих параллельные операции, присутствует суффикс PS. Скалярные операции действуют на младшие (занимающие разряды 0-31) элементы данных двух операндов. Остальные три элемента данных в выходном операнде не изменяются (исключение составляет команда скалярного копирования MOVSS). В имени команд, выполняющих скалярные операции, присутствует суффикс SS.

Ход работы: на рисунке 1 представлены изначальные значения регистров, на рисунку 2 - значения регистров и результат работы программного продукта после выполнения.

Листинг 1 – Исходный код программы задания

```
#include <iostream>

int main()
{
    __int8 A[8] = { 5, 9, 3, 14, 6, 10, 3, 8 };
    __int8 B[8] = { 3, 14, 11, 16, 12, 18, 5, 11 };
    __int8 C[8] = { 15, 11, 16, 21, 22, 25, 19, 24 };
    __int16 D[8] = { 10, 4, 7, 8, 10, 4, 11, 12 };
    __int16 F[8];

    __asm {
            // a + b * c - d
            movsd xmm0, [A] // запись значений в регистры
            movsd xmm1, [B]
            movsd xmm2, [C]
            movlpd xmm3, [D]
```

```
            movhpd xmm3, [D + 8] // Расширение значений int8 до int16 (до 2
байт)
            pmovsxbw xmm0, xmm0 // Расширение значений int8 до int16 (до 2 байт)
            pmovsxbw xmm1, xmm1
            pmovsxbw xmm2, xmm2

            pmullw xmm1, xmm2
            paddd xmm0, xmm1
            psubd xmm0, xmm3

            movdqu[F], xmm0
    }

    std::cout << "Results of calculation F[i]: \n";
    for (int i = 0; i < 8; i++) {
        std::cout << "F[" << i << "] = " << F[i] << '\n';
    }
    return 0;
}
```

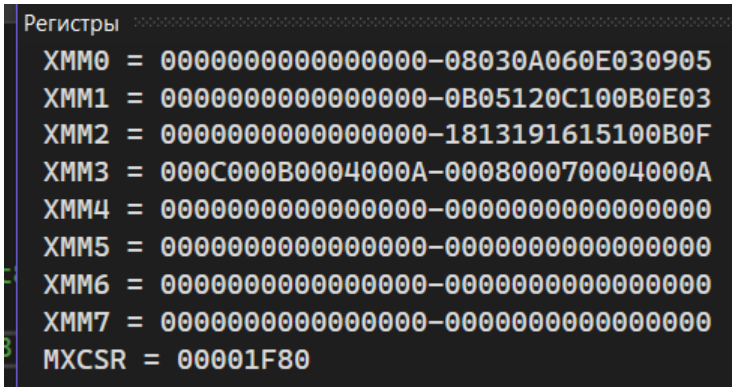

Рисунок 1 – Значения регистров программы перед выполнением

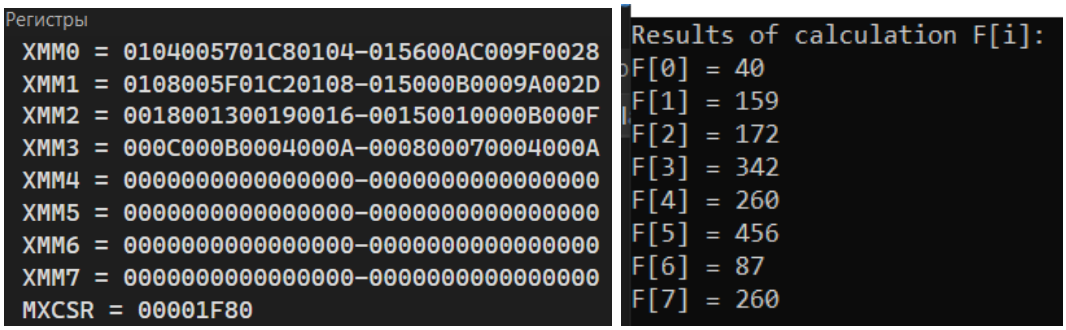

Рисунок 2 – Значения регистров и результат работы программы

Вывод: В ходе работы было выполнено задание: обработать массивы из 8 элементов по следующему выражению: F[i]=A[i] + (B[i] * C[i]) -D[i], i=1...8. А также изучены программная модель SSE и система команд SSE.