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 253504
Дмитрук Богдан Ярославович

(дата, подпись студента)

Проверил ассистент кафедры информатики
Калиновская Анастасия Александровна

(дата, подпись преподавателя)

Минск 2024

Теоретические сведения:

Расширение SSE

SSE (англ. Streaming SIMD Extensions, потоковое SIMD-расширение процессора) — это набор SIMD инструкций, разработанный Intel , и впервые представленный в процессорах серии Pentium III.

Технология SSE позволяет преодолеть основную проблему MMX — при использовании MMX невозможно одновременно использовать инструкции сопроцессора, так как его регистры используются и для MMX и для работы FPU.

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.

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.

Расширение SSE2

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.

Программная модель SSE/SSE2

Регистры SSE/SSE2

Все три расширения работают с одним набором 128-битных регистров, обозначаемых XMM0…XMM7, как показано на рисунке 1.

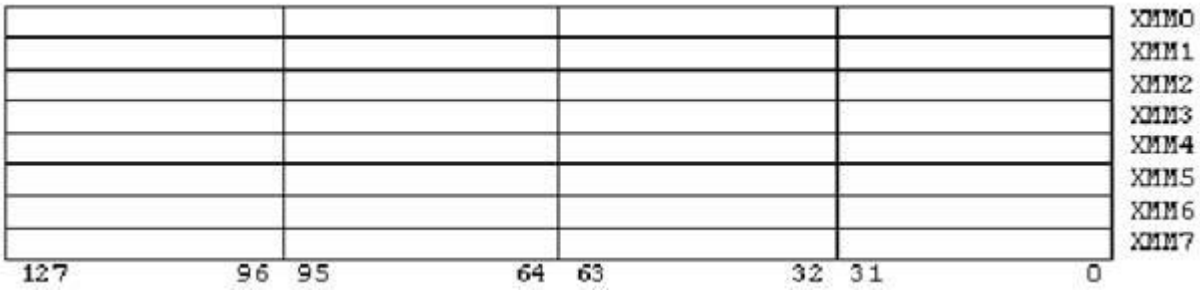

Рисунок 1 – Регистры SSE/SSE2

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.

Цель работы: Вариант 7. 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 C – 8 разрядные целые знаковые числа (_int8);

D – 16 разрядные целые знаковые числа (_int16).

Полученный результат отобразить на форме с использованием соответствующих элементов. При распаковке знаковых чисел совместно с командами распаковки использовать команды сравнения (сравнивать с нулём перед распаковкой).

Ход работы: на рисунке 1 представлены регистры MMX, на рисунке 2 представлены входные данные, на рисунке 3 представлены результаты программы.

Листинг 1 – Исходный код ассемблерной вставки реализующей вычисления в соответствии с поставленным условием

```
__asm {
        xorpd xmm0, xmm0
        xorpd xmm1, xmm1
        xorpd xmm2, xmm2
        xorpd xmm3, xmm3
        xorpd xmm4, xmm4
        xorpd xmm5, xmm5
        xorpd xmm6, xmm6


        movupd xmm0, [A]
        movupd xmm1, [B]

        punpcklbw xmm0, xmm4
        punpcklbw xmm1, xmm4

        psubw xmm0, xmm1

        movupd xmm1, [C]
        movupd xmm2, [C]
        movupd xmm3, [C]
        movupd xmm4, [D]

        punpcklbw xmm1, xmm5
        punpcklbw xmm2, xmm5
        punpcklbw xmm3, xmm5
```

```
        pmullw xmm1, xmm4
        pmullw xmm2, xmm4
        pmulhw xmm3, xmm4

        paddw xmm1, xmm0
        paddw xmm2, xmm0

        punpcklwd xmm1, xmm3
        punpckhwd xmm2, xmm3


        paddq xmm2, xmm3

        pmullw xmm0, xmm2

        movupd[F + 16], xmm2
        movupd[F], xmm1
}
```

```
XMM0 = 0000000000000000-0000000000000000
XMM1 = 0000006500780065-002E003400440057
XMM2 = 006E0055005C0033-0065006D0075006C
XMM3 = 004C0041005C0053-00430041005C0069
XMM4 = 0075006200650044-005C003400440057
XMM5 = 0000000000000000-0000000000000000
XMM6 = 0000000000000000-0000000000000000
XMM7 = 0000000000000000-0000000000000000
MXCSR = 00001F80
```

Рисунок 1 – Регистры SSE

```
__int8 A[8] = { 1, 2, 3, 4, 5, 6, 7, 8};
__int8 B[8] = { 1, 1, 1, 1, 1, 1, 1, 1 };
__int8 C[8] = { 1, 2, 3, 4, 5, 6, 7, 8};
__int16 D[8] = { 1, 2, 3, 4, 5, 6, 7, 8};
__int32 F[8] = {};
```

Рисунок 2 – Входные данные

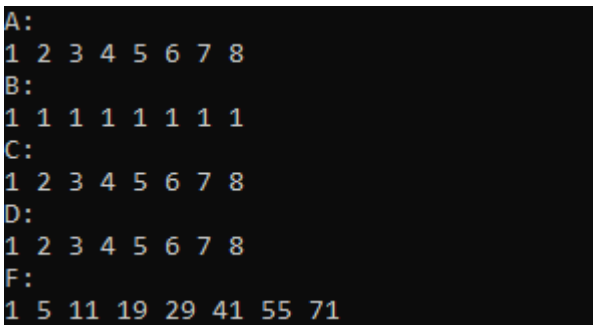

Рисунок 3 – Результаты работы программы

Выводы: в результате лабораторной работы была выполнена одна задача, где с помощью программной модели SSE и системы команд SSE было посчитано выражение.