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 №3

на тему:

«ТЕХНОЛОГИЯ ММХ»

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:

Технология Intel MMX представляет собой набор расширений к архитектуре Intel, которые были разработаны для того, чтобы увеличить производительность средств мультимедиа и коммуникаций.

Расширение MMX предназначено для ускорения выполнения приложений типа «подвижное видео», комбинированной графики с видеообработкой изображений, звуковым синтезом, синтезом и сжатием речи, телефонией, видео, конференц-связью, и 2D и 3D графикой, которые обычно используют алгоритмы с интенсивными вычислениями, чтобы выполнять повторяющиеся действия на больших множествах простых элементов данных.

MMX-технология использует методику «одиночная команда, множественные данные» (Single Instruction Multiple Data – SIMD) для выполнения арифметических и логических операций над байтами, словами или двойными словами, упакованными в 64-разрядные регистры MMX. Например, команда paddsb складывает 8 знаковых байт источника с 8 знаковыми байтами в приемнике и сохраняет их в операнде-приемнике.

Программная модель расширения MMX.

Технология MMX обеспечивает следующие новые расширения к окружающей среде программирования архитектуры IA-32:
восемь 64-разрядных MMX-регистров MM0-MM7;
четыре типа данных MMX (упакованные байты, слова, двойные слова и учетверенное слово);
систему команд MMX.

Регистры MMX.

Набор регистров MMX состоит из восьми 64-разрядных регистров MM0-MM7. Эти регистры могут использоваться только для выполнения вычислений над MMX типами данных и не могут использоваться для адресации памяти.

Типы данных MMX:

упакованные байты – восемь байт, упакованные в одно 64-разрядное поле;
упакованные слова – четыре слова, упакованные в одно 64-разрядное поле;
упакованные двойные слова – два двойных слова, упакованные в одно 64-разрядное поле;
учетверенное слово – одно 64-разрядное поле.

Арифметика с насыщением и арифметика цикличности.

Технология MMX поддерживает новую арифметическую возможность, известную как арифметика с насыщением (Saturated Arithmetics). Арифметику с насыщением лучше всего определить, противопоставляя ее арифметике цикличности (Wraparound Arithmetic). В арифметике цикличности результаты, которые переполняются или антипереполняются, усекаются и возвращаются только самые младшие биты результата (только те, которые входят в разрядную сетку соответствующего типа), т.е. перенос игнорируется. В режиме насыщения результаты операции, которые

переполняются или антипереполняются, приводятся к соответствующим значениям границ диапазона для данного типа данных (таблица 1).

Система команд

Система команд MMX состоит из 57 команд, сгруппированных в следующие категории:
команды пересылки данных;
арифметические команды;
команды сравнения;
логические команды;
команды сдвига;
команды упаковки и распаковки;
дополнительные команды;
команда инициализации.

Команды пересылки данных

movd dst, src – пересылает 32-разрядные данные из памяти в регистры MMX и обратно или из целочисленных регистров процессора в регистры MMX и обратно.
movq dst,src – пересылает 64-разрядные упакованные данные из памяти в регистры MMX и обратно или между регистрами MMX.

Арифметические команды

Упакованное сложение и вычитание выполняют следующие команды:
paddsb, paddsw, paddwd – выполняют сложение знаковых или беззнаковых упакованных байтов, слов, двойных слов.
psubb, psubw и psubd – выполняют вычитание знаковых или беззнаковых упакованных байтов, слов, двойных слов.

Упакованное умножение. Команды pmulhw и pmullw умножают знаковые слова операндов источника и адресата и записывают старшую или младшую часть результата в операнд адресата.

Упакованное умножение/сложение. Команда pmaddwd вычисляет произведение знаковых слов операндов адресата и источника. Четыре промежуточных 32-разрядных произведения суммируются в парах, чтобы произвести два 32-разрядных результата.

Команды сравнения.

Команды pcmpeqb, pcmpeqw, pcmpeqd (упакованное сравнение на равенство) и pcmpgtb, pcmpgtw, pcmpgtd (упакованное сравнение на «больше») сравнивают соответствующие элементы данных в операндах источника и адресата на равенство или оценивают, кто из них больше.

Команды преобразования.

Команды packsswb и packssdw (упакованный со знаковой насыщенностью) преобразовывают знаковые слова в знаковые байты или знаковые двойные слова в знаковые слова в режиме знаковой насыщенности.

Команда packuswb (упакованный насыщенностью без знака) преобразовывает знаковые слова в байты без знака в режиме насыщенности без знака.

Команды punpckhbw, punpckhwd и punpckhdq (распаковать старшие упакованные данные) и punpcklbw, punpcklwd и punpckldq (распаковать младшие упакованные данные) преобразовывают байты в слова, слова в двойные слова или двойные слова в четверное слово.

Логические команды.

Команды pand (поразрядное логическое И), pandn (поразрядное логическое И-НЕ), por (поразрядное логическое ИЛИ) и pxor (поразрядное логическое исключающее ИЛИ) выполняют поразрядные логические операции над 64-разрядных данными.

Команды сдвига.

Команды psllw, pslld (упакованный логический сдвиг влево) и psrlw, psrld (упакованный логический сдвиг вправо) выполняют логический левый или правый сдвиг и заполняют пустые старшие или младшие битовые позиции нулями.

Команды psraw и psrad (упакованный арифметический сдвиг вправо) выполняют арифметический сдвиг вправо, копируя знаковый разряд в пустые разрядные позиции на старшем конце операнда.

Команда EMMS.

Команда emms освобождает состояние MMX. Эта команда должна использоваться, чтобы очистить состояние MMX (чтобы освободить tag-слово регистров FPU) в конце MMX подпрограммы перед вызовом других подпрограмм, которые могут выполнять операции с плавающей точкой.

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 программы задания

```
#include <iostream>

int main()
{
    __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] = {};
    __int32 AMinB[8] = {};
    __int64 temp = 0;


    __asm {
        movq mm1, C
        movq mm0, C
        pxor mm7, mm7

        movq mm2, D + 8


        punpckhbw mm1, mm7
        movq temp, mm1
        movq mm3, temp

        pmullw mm1, mm2

        movq temp, mm1
        movq mm4, temp
```

```
pmulhw mm3, mm2



punpcklwd mm1, mm3
punpckhwd mm4, mm3

movq F + 16, mm1
movq F + 24, mm4

movq mm2, D

punpcklbw mm0, mm7
movq temp, mm0
movq mm3, temp

pmullw mm0, mm2
movq temp, mm0
movq mm4, temp

pmulhw mm3, mm2

punpcklwd mm0, mm3
punpckhwd mm4, mm3

movq F, mm0
movq F + 8, mm4

movq mm1, B
movq mm0, A

psubsb mm0, mm1

movq temp, mm0
movq mm1, temp

movq mm3, F
movq mm4, F + 8

packssdw mm3, mm4

movq mm4, F + 16
movq mm5, F + 24
packssdw mm4, mm5
```

```
    punpcklbw mm0, mm7
    punpckhbw mm1, mm7

    paddw mm0, mm3
    paddw mm1, mm4

    movq temp, mm0
    movq mm2, temp

    movq temp, mm1
    movq mm3, temp

    punpcklwd mm0, mm7
    punpcklwd mm1, mm7
    punpckhwd mm2, mm7
    punpckhwd mm3, mm7


    movq F, mm0
    movq F + 16, mm1
    movq F + 8, mm2
    movq F + 24, mm3
    emms
}
std::cout << "A:\n";
for (size_t i = 0; i < 8; i++)
{
    std::cout << (int)A[i] << " ";
}
std::cout << "\nB:\n";
for (size_t i = 0; i < 8; i++)
{
    std::cout << (int)B[i] << " ";
}
std::cout << "\nC:\n";
for (size_t i = 0; i < 8; i++)
{
    std::cout << (int)C[i] << " ";
}
std::cout << "\nD:\n";
for (size_t i = 0; i < 8; i++)
{
    std::cout << D[i] << " ";
}
```

```
    std::cout << "\nF:\n";
    for (size_t i = 0; i < 8; i++)
    {
        std::cout << F[i] << " ";
    }
}
```

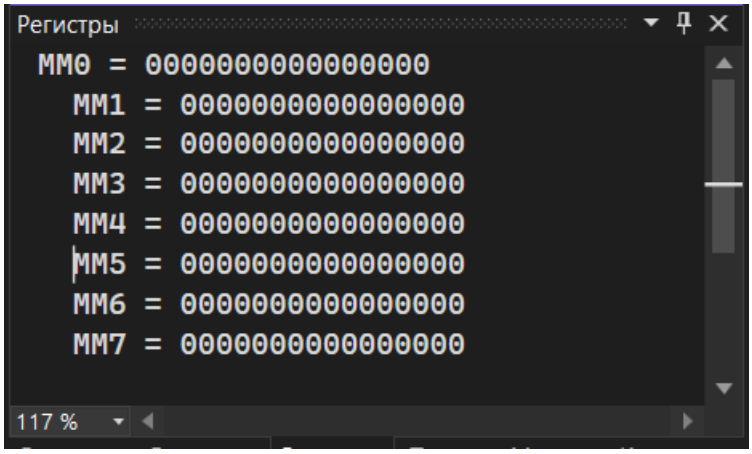

Рисунок 1 – Регистры MMX

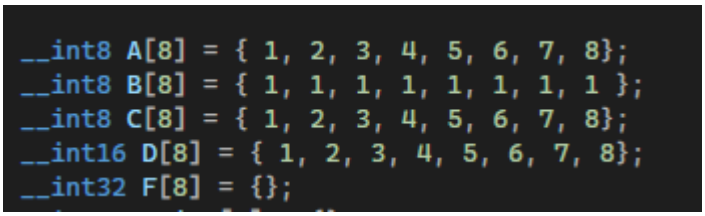

Рисунок 2 – Входные данные

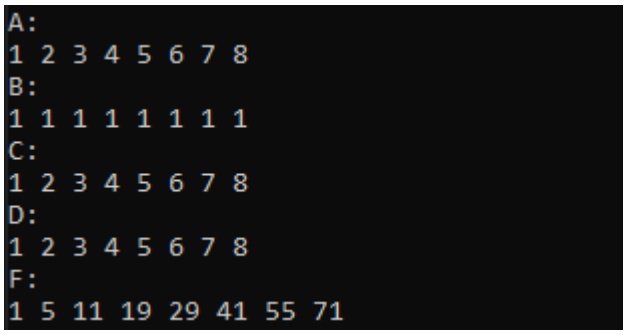

Рисунок 3 – Результаты работы программы

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