###### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

###### ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

###### НОВОСИБИРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

###### Факультет информационных технологий

**Кафедра параллельных вычислений**

ОТЧЕТ

О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ

«ИЗМЕРЕНИЕ СТЕПЕНИ АССОЦИАТИВНОСТИ КЭШ-ПАМЯТИ»

студента 2 курса, группы 21209

Усольцева Антона Андреевича

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

М.А.Мичуров

В.А.Перепёлкин

Новосибирск 2022

**СОДЕРЖАНИЕ**

Цель 3

Задачи 3

Описание работы 4

Заключение 7

# ЦЕЛЬ

1. Экспериментальное определение степени ассоциативности кэш-памяти.

# ЗАДАЧИ

1. Написать программу, выполняющую обход памяти в соответствии с заданием.

2. Измерить среднее время доступа к одному элементу массива (в тактах процессора) для разного числа фрагментов: от 1 до 32. Построить график зависимости времени от числа фрагментов

3. По полученному графику определить степень ассоциативности кэш-памяти, сравнить с реальными характеристиками исследуемого процессора

.

# ОПИСАНИЕ РАБОТЫ

Листинг 1

#include <iostream>  
#include <fstream>  
  
//size in bytes:  
constexpr int cache\_size\_lvl3 = 8 \* 1024 \* 1024;  
constexpr int cache\_size\_int = cache\_size\_lvl3 / 4;  
constexpr int offset = cache\_size\_int;  
constexpr int repeats = 100;  
  
int \*fill(size\_t size, int fragm\_count) {  
 auto arr = new int[size];  
 int fragm\_size = cache\_size\_int / fragm\_count;  
 int ind;  
 for (int i = 0; i < fragm\_size; i++) {  
 ind = i;  
 for (int j = 0; j < fragm\_count - 1; j++) {  
 arr[ind] = ind + offset;  
 ind += offset;  
 }  
 arr[ind] = i + 1;  
 }  
 arr[ind] = 0;  
 return arr;  
}  
  
void bypass(const int \*arr, int fragm\_count, std::ofstream &fout) {  
 union ticks {  
 unsigned long long t64;  
 struct s32 {  
 long th, tl;  
 } t32;  
 } start{}, end{};  
 int fragm\_size = cache\_size\_int / fragm\_count;  
 long long min = LLONG\_MAX;  
 for (int j = 0; j < repeats; j++) {  
 asm volatile ("rdtsc\n":"=a"(start.t32.th), "=d"(start.t32.tl)::"memory");  
 for (int k = 0, i = 0; i < fragm\_count \* fragm\_size; i++) {  
 k = arr[k];  
 if (k == -1)  
 fout << "--\n";  
 }  
 asm volatile ("rdtsc\n":"=a"(end.t32.th), "=d"(end.t32.tl)::"memory");  
 long long ticks = end.t64 - start.t64;  
 min = min < ticks ? min : ticks;  
 }  
 fout << "Count of fragments: " << fragm\_count << " ";  
 fout << "Time taken: " << (min) / (fragm\_count \* fragm\_size) << " ticks\n";  
 fout.flush();  
}  
  
int main() {  
 try {  
 int \*test\_arr;  
 std::ofstream res("res.txt");  
 if (!res.is\_open()) {  
 std::cerr << "Failed to open file" << std::endl;  
 return 1;  
 }

for (int frag\_cnt = 1; frag\_cnt <= 32; frag\_cnt++) {  
 int arr\_size = offset \* frag\_cnt;  
 test\_arr = fill(arr\_size, frag\_cnt);  
 bypass(test\_arr, frag\_cnt, res);  
 }  
 res.close();  
 }  
 catch (std::bad\_alloc &e) {  
 std::cerr << e.what();  
 }  
 return 0;  
}

Рассмотрим выводимые данные и построим график зависимости среднего времени обращения от количества фрагментов.



Построим график:

Зависимость среднего времени чтения элемента массива от количества фрагментов для процессора Intel Core i5 8300H

Из конфигураций процессора определим фактическую степень ассоциативности кэш-памяти

Кэш 1-ого уровня (L1): 8-way

Кэш 2-ого уровня (L2): 4-way

Кэш 3-ого уровня (L3): 16-way

Из графика видно, что первый скачок кривой происходит при количестве фрагментов, равным 4: либо происходит буксование кэша второго уровня, либо это соответствует степени ассоциативности буфера трансляции адресов. Следующий скачок на 8 фрагментах – буксование первого уровня, и затем скачок на 16 фрагментах – буксование третьего уровня.

# ЗАКЛЮЧЕНИЕ

В ходе работы были исследованы зависимость времени доступа к данным в памяти от количества фрагментов данных в памяти, отстоящих друг от друга на смещение, кратное размеру банка. Изучен эффект кэш-буксования и экспериментально определена степень ассоциативности кэш-памяти. Был построен график зависимости и полученные данные были сравнены с реальными значениями степени ассоциативности кэш-памяти.

# Контрольные вопросы:

1. Что такое степень ассоциативности кэш-памяти?

Степень ассоциативности кэш-памяти - это количество банков кэш

памяти (в рассматриваемом множественно-ассоциативном типе кэш-памяти)

2) Назовите достоинства и недостатки множественно-ассоциативной кэш-памяти?

Множество-ассоциативная кэш-память является промежуточным вариантом между полностью ассоциативным кэшем и кэшем с прямым отображением. Достоинства множество-ассоциативной кэш-памяти состоят в том, что для определения наличия нужного элемента в кэше достаточно проверить соответствующие ячейки каждого банка, а количество банков сильно меньше суммарного количества кэш-строк, где может лежать элемент (в этом плюс относительно полностью ассоциативного кэша). Также плюсом является то, что данные, лежащие на одинаковом смещение, кратном размеру банка, могут записаться в соответствующие ячейки разных банков (в этом плюс относительно кэша с прямым отображением). Но так как количество банков конечно, то в какой-то момент новые данные, находящиеся на одинаковом смещении, кратном размеру банка, от предыдущих, будут записаны поверх уже записанных, то есть произойдет эффект кэш буксования. В этом минус множество-ассоциативной кэш-памяти.

3) Что такое эффект «буксования» кэш-памяти? Как его вызвать? Как его избежать?

При обращении к данным, расположенным в памяти на одинаковом смещении друг от друга и кратном размеру банка, эти данные будут записываться в идентичные ячейки каждого банка, и когда количество обращений в память превысит количество банков, то старые данные будут сначала вытесняться из кэша и потом записываться новые, т.е. всегда будет обращение в медленную память. Избежать этого можно, обращаясь к элементам в более «хаотичном» порядке, т.е. с различным смещением в памяти относительно друг друга.

4) Какое минимальное число элементов требуется, чтобы организовать эффект «буксования» кэш-памяти (при «наихудшем» размещении этих элементов в оперативной памяти)?

По принципу Дирихле минимальное число элементов = степень ассоциативности кэша + 1

5) Какой график получится в результате тестирования, если величину смещения между фрагментами взять равной половине размера банка?

График растянется в два раза в ширину (по оси числа фрагментов)

Причину можно показать на рисунке устройства кэша

То есть буксование начнется только с 2\*(степень ассоциативности) +1