###### 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 19201

Хаецкой Дарьи Владимировны

Направление 09.03.01 – «Информатика и вычислительная техника»

Преподаватель:

Власенко Андрей Юрьевич

Новосибирск 2020

1. **Цель**

Экспериментально определить степени ассоциативности кэш-памяти разных уровней.

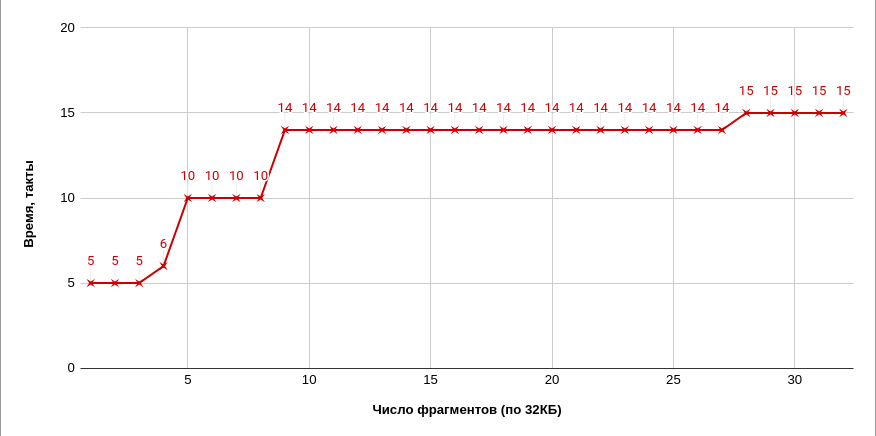
1. **Задание**

На основе обхода циклического массива, что вызывает кэш-буксование, сделать среднюю оценку количества тактов для обращения к одному элементу массива. На основе полученных данных выяснить степени ассоциативности кэш-памяти процессора.

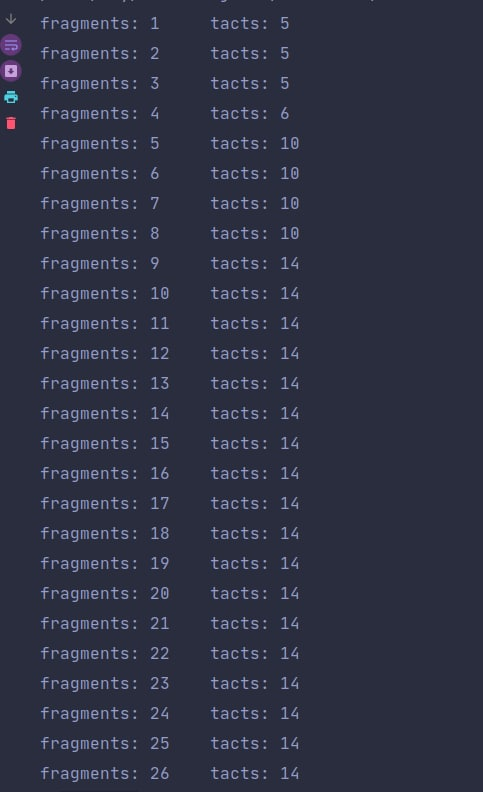
1. **График роста тактов обращения**

В лабораторной работе создавался массив с фрагментами со смещением в 1МБ. Количество элементов в фрагменте 32КБ, равняется размеру кэш памяти первого уровня.

Следующий график показывает зависимость времени обращения к одному элементу в тактах от количества фрагментов в массиве:



Вывод программы, на основе которого заполнялась таблица:



1. **Листинг программы**

#include <iostream>

#include <climits>

#include <x86intrin.h>

const unsigned int N = 200000000;

const unsigned int RUN\_TIMES = 4;

void initArray(unsigned int \*array, unsigned int fragments, size\_t offset, size\_t size){

size\_t j = 1;

size\_t i = 0;

for(; i < size; i++)

{

for(j = 1; j < fragments; j++)

array[i + (j - 1) \* offset] = i + j \* offset;

array[i + (j - 1) \* offset] = i + 1;

}

array[i - 1 + (j - 1) \* offset] = 0;

}

unsigned long long runArray(unsigned int const \*array){

unsigned long long startTime, endTime;

unsigned long long minTime = ULLONG\_MAX;

for(size\_t j = 0; j < RUN\_TIMES; j++){

startTime = \_\_rdtsc();

for(volatile size\_t k = 0, i = 0; i < N; i++){

k = array[k];

}

endTime = \_\_rdtsc();

if (minTime > endTime - startTime)

minTime = endTime - startTime;

}

return minTime;

}

void countTime(unsigned int \*array, unsigned int fragments, int offset, int size){

initArray(array, fragments, offset, size);

std::cout << fragments << " fragments " << runArray(array) / N << " tacts" << std::endl;

}

int main(){

auto \*array = (unsigned int \*) malloc(N \* sizeof(unsigned int));

if (array == nullptr) {

return 1;

}

unsigned int offset = 8388608; // 1MB

unsigned int BlockSize = 262144; // 32KB == L1 cache size

for(int fragments = 1; fragments <= 32; fragments++) {

countTime(array, fragments, offset / sizeof(int), BlockSize / sizeof(int));

}

free(array);

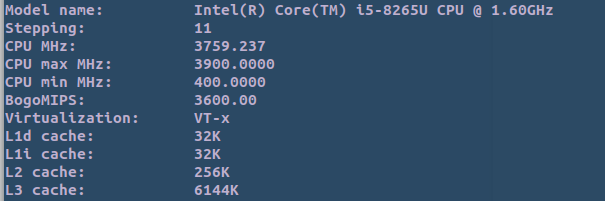
return 0;

}

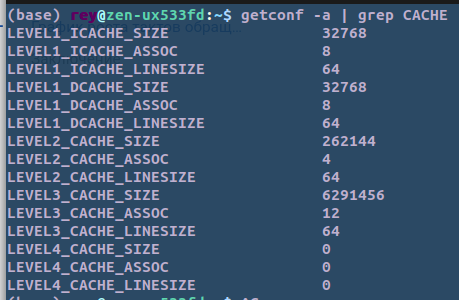
1. **Оценка ассоциативности кэша**

На графике заметно два скачка. Первый скачок происходит после 4 фрагментов, что, скорее всего, говорит о степени ассоциативности TLB-кэша (буфера трансляции адресов) либо степени ассоциативности кэша L2. Следующий скачок роста времени обращения произошел на 9-ом фрагменте, что соответствует ассоциативности L1.

Данные о процессоре, на котором производился эксперемент:



Информация об ассоциативности, полученная из конфигураций:



1. **Заключение**

В ходе данной работы было изучено строение множественно-ассоциативного кэша и эксперементально получены степени ассоциативности кэш памяти первого и второго уровня.