

## КОММЕНТАРИЙ К ЛАБОРАТОРНОЙ РАБОТЕ №8 «ВЛИЯНИЕ КЭШ-ПАМЯТИ НА ВРЕМЯ ОБРАБОТКИ МАССИВОВ»

Лабораторная направлена на практическое исследование размеров разных уровней кэш-памяти у используемого процессора. Главное задание – построить графики зависимости среднего времени доступа к элементу массива от размера массива при трех вариантах обхода этого массива (прямом, обратном и случайном).

Единственное, что должно подвергаться замеру времени – следующий цикл:

```
for (k=0, i=0; i<N*K; i++) k = x[k];
```

Очень важно то, что кроме доступа к очередному элементу массива этот цикл не должен делать ничего. Мы ведь измеряем именно среднее время доступа к определенной ячейке памяти, которое составляет несколько процессорных тактов. Таким образом, практически любая работа будет на каждой итерации занимать еще, как минимум, несколько тактов и измерения уже будет нельзя назвать правдивыми. С ростом массива он сначала не будет помещаться в кэш-память первого уровня, потом второго, потом третьего (если есть). Время доступа к кэшу следующего уровня или в итоге к памяти, на порядок больше, чем к кэшу предыдущего уровня. Соответственно на графике случайного обхода (а почему только случайного?) у Вас должны наблюдаться точки, после которых идет резкий рост.

Поскольку Вы будете замерять достаточно большие промежутки времени (много итераций цикла), то функцию-таймер, конечно, можно использовать любую. Но поскольку нам нужно именно время в тактах, то удобнее сразу применить таймер `rdtsc`, возвращающий именно число тактов.

Еще важный момент. В тексте описания лабораторной вам предлагается увеличивать на каждой итерации массив в 2 раза. На самом деле полученный график будет очень грубым – на нем будет невозможно понять где точки перелома, потому как на каждый уровень кэша будет приходиться всего 2-3 точки. Советую шагать хотя бы умножая предыдущее значение длины в 1,2 раза. Если еще больше промежуточных точек сделаете – будет еще лучше.

Полученные точки перелома следует сравнить с реальными размерами кэшей разных уровней вашего процессора. Получить информацию о процессоре в Linux можно, например, по файлу `/proc/cpuinfo`, утилите `lscpu`. В windows, например, по бесплатной утилите CPU-Z. В отчет вставьте скриншот по одному из таких средств, демонстрирующих размеры кэшей Вашего процессора.

При формировании массива для случайного обхода необходимо учитывать следующий фактор. Если, например, проинициализировать массив из 7 элементов  $A[i]=i$ , а потом перемешать случайным образом, то можно получить такую картину:

3 2 1 6 7 4 0 5

При обходе мы получим цикл: 3 6 0, потому как

$k=A[0]; (3)$

$k=A[3]; (6)$

$k=A[6]; (0)$

Соответственно, сколько бы итераций не делали, а будем крутиться по этим элементам, которые успешно сядут в кэш 1-го уровня или вообще в регистры и длина массива может быть уже любой – среднее время расти не будет.

Поэтому для случайного обхода необходимо формировать массив таким образом, чтобы цикл был один и его длина равнялась длине массива.

В начале программы перед началом счёта и измерения времени нужно около 1 секунды погонять пустой цикл с каким-нибудь расчётом (например, умножение матриц), чтобы процессор с динамически изменяемой частотой установил её на фиксированном уровне. Иначе в начале замеряемого промежутка времени процессор будет работать на пониженной частоте, и значения на графике будут чуть выше.

Еще стоит сделать 1 прогон цикла «for (k=0, i=0; i<N\*K; i++) k = x[k];» «вхолостую», то есть без замера времени. Эта операция называется «прогрев кэша», когда из него вытесняются все ненужные «мусорные» данные.

Компилировать важно с ключом оптимизации O1, а не O0, потому как в противном случае переменные, фигурирующие в цикле, компилятор может разместить в стеке и, стало быть, будут дополнительные обращения к памяти, резко увеличивающие время. С другой стороны, поскольку будут применены оптимизации, компилятор может выкинуть весь цикл как ненужный. Чтобы он этого не делал, нужно после завершения замера времени изобразить “использование” полученного в цикле значения, например, так:

```
if (k==12345) printf("Wow!");
```

Также следует поступить после «холостых» вычислений для выхода процессора на максимальную частоту и после прогрева кэша.

Успешной работы!