Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

«Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет)» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»(ИУ7)

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.04 «Программная инженерия»

## О Т Ч Е Т

по лабораторной работе № 3

Название Организация памяти конвейерных суперскалярных ЭВМ

Дисциплина Архитектура элекронно-вычислительных машин

|  |  |  |
| --- | --- | --- |
| Студент: |  | Золотухин А. В. |
| Преподаватель: | подпись, дата | Фамилия, И.О.  Попов А. Ю. |
|  | подпись, дата | Фамилия, И. О. |

Москва — 2022 г.

# Цель работы

Основной целью работы является освоение принципов эффективного исполь- зования подсистемы памяти современных универсальных ЭВМ, обеспечивающей хранение и своевременную выдачу команд и данных в центральное процессорное устройство. Работа проводится с использованием программы для сбора и анализа производительности PCLAB.

В ходе работы необходимо ознакомиться с теоретическим материалом, каса- ющимся особенностей функционирования подсистемы памяти современных кон- вейерных суперскалярных ЭВМ, изучить возможности программы PCLAB, изу- чить средства идентификации микропроцессоров, провести исследования времени выполнения тестовых программ, сделать выводы о архитектурных особенностях используемых ЭВМ.

# Исследования расслоения динамической па-

мяти

Исходные данные: размер линейки кэш-памяти верхнего уровня (128), объем физической памяти.

Настраиваемые параметры: максимальное расстояния между читаемыми бло- ками (128К), шаг увеличения расстояния между читаемыми 4-х байтовыми ячей- ками (64Б), размер массиваи (2М).

Графики полученных характеристик представлены на рисунке [1.](#_bookmark0)

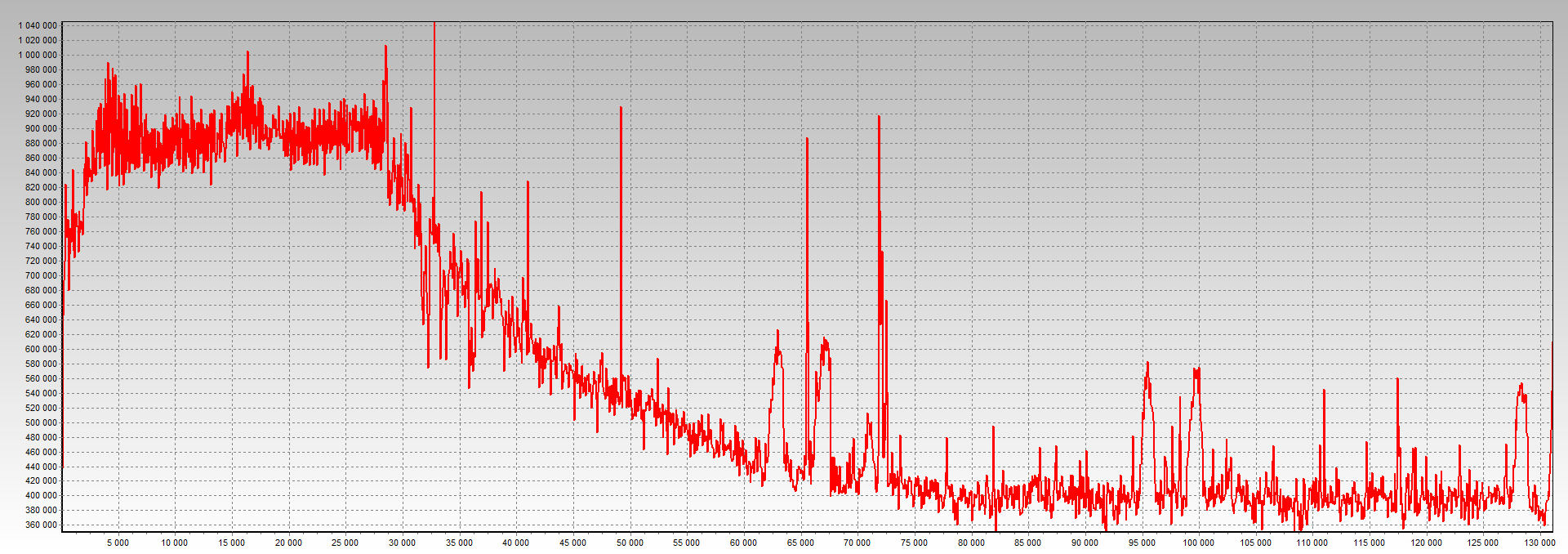


Рисунок 1 – Результат исследования расслоения динамической памяти График показывает количество тактов работы алгоритма. Ось абсцисс отража-

ет шаг приращения адреса читаемых данных. Ось ординат отображает количество тактов.

# Вывод

Оперативная память расслоена и неоднородна, поэтому обращение к после- довательно расположенным данным требует различного времени из-за наличия открытыя и закрытыя страниц динамической памяти. При этом чем больше ад- ресное расстояние, тем больше время доступа. В связи с этим для создания эффек- тивных программ необходимо учитывать расслоение памяти и размещать рядом данные для непосредственной обработки.

# Сравнение эффективности ссылочных и век-

торных структур

Настраиваемые параметры: количество элементов в списке (4М), максималь- ная фрагментации списка (100К), шаг увеличения фрагментации (2К).

Графики полученных характеристик представлены на рисунке [2.](#_bookmark2)

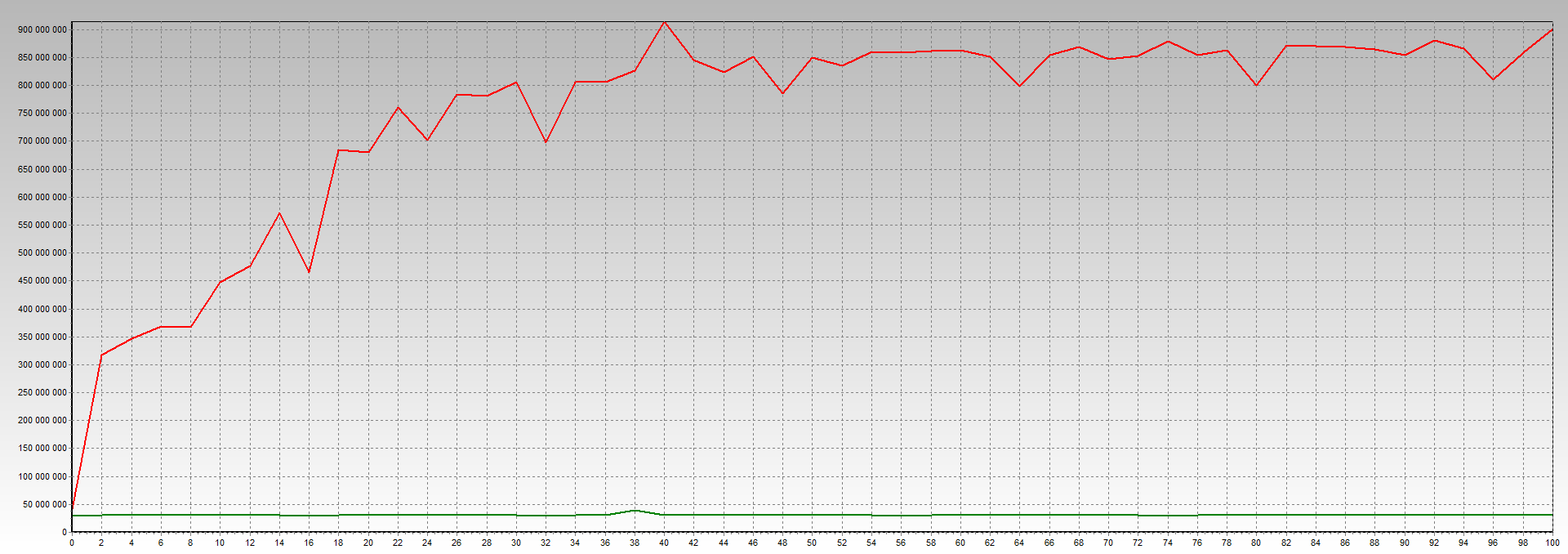


Рисунок 2 – Результат исследования эффективности ссылочных и векторных структур

Красный график (верхний) показывает количество тактов работы алгоритма, использующего список. Зеленый график (нижний) показывает количество тактов работы алгоритма, использующего массив. Ось абсцисс отражает фрагментацию списка.

Результаты эксперимента:отношение времени работы алгоритма, использую- щего зависимые данные, ко времени обработки аналогичного алгоритма обработ- ки независимых данных = 24.7

## Вывод

Видна проблема семантического разрыва: машина не присоблена к работе со ссылочными структурами. Использовать структуры данных надо с учётом скры- тых технологических констант. Если алгоритм предполагает возможность исполь- зования массива, а списки не дают существенной разницы, то использование мас- сива вполне оправдано.

# Исследование эффективности программной

предвыборки

Исходные данные: степень ассоциативности и размер TLB данных. Настраиваемые параметры: шаг увеличения расстояния между читаемыми-

данными (1024Б), размер массива и (8192К).

Графики полученных характеристик представлены на рисунке [3.](#_bookmark3)

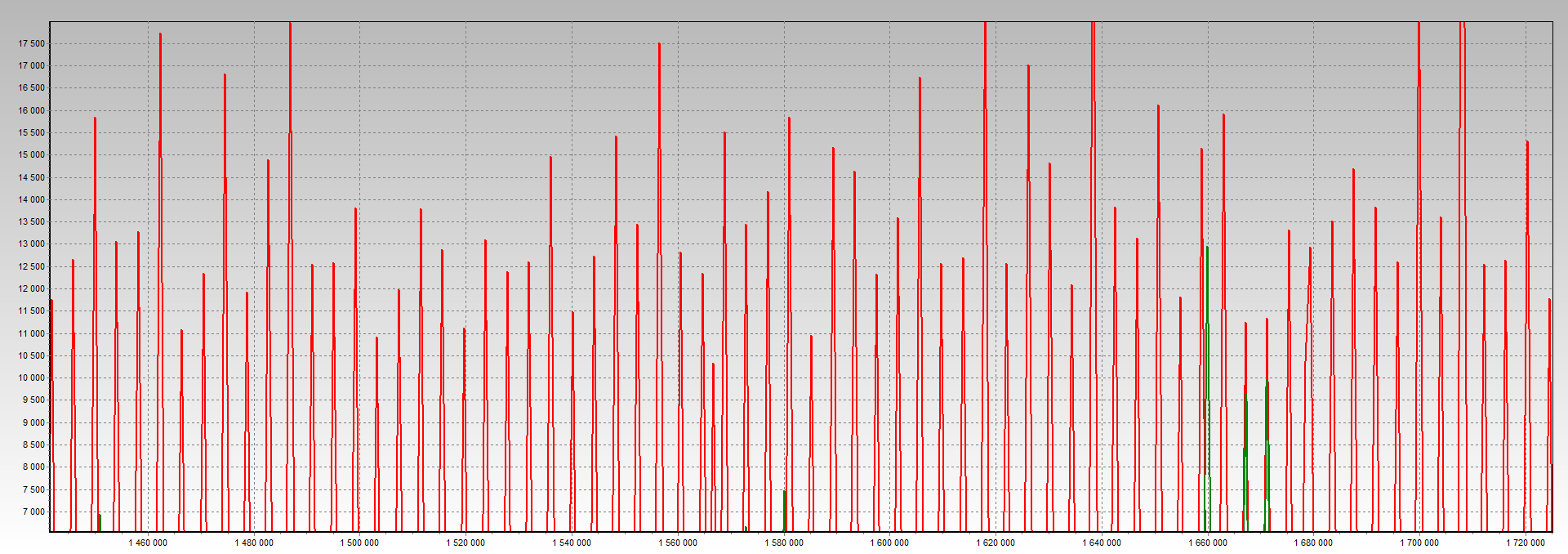


Рисунок 3 – Результат исследования эффективности предвыборки

Красный график (верхний с острыми пиками) показывает количество тактов работы алгоритма без предвыборки. Зеленый график (нижний без значимых пи- ков) показывает количество тактов работы алгоритма с использованием предвы- борки. Ось абсцисс отражает смещение читаемых данных от начала блока.

Результаты эксперимента: отношение времени последовательной обработки бло- ка данных ко времени обработки блока с применением предвыборки = 4.04; ко- личество тактов первого обращения к странице данных = 15000

## Вывод

Обработка больших массивов информации сопряжена с открытием большого количества физических страниц памяти. При первом обращении к странице па- мяти наблюдается увеличенное время доступа к данным в 20 раз, так как оно при отсутствии информации в TLB вызывает двойное обращение к оперативной па- мяти: сначала за информацией из таблицы страниц, а далее за востребованными данными. Поэтому для ускорения работы программы можно использовать пред- выборку. Например, пока процессор занят некоторыми расчетами и не обращается к памяти, можно заблаговременно провести все указанные действия благодаря до- полнительному запросу небольшого количества данных из оперативной памяти.

Также стоит стараться не использовать в программе массивы, к которым об- ращение выполняется только один раз.

# Исследование способов эффективного чте-

ния оперативной памяти

Исходные данные: адресное расстояние между банками памяти, размер буфера чтения.

Настраиваемые параметры: размер массива (4М), количество потоков данных (64).

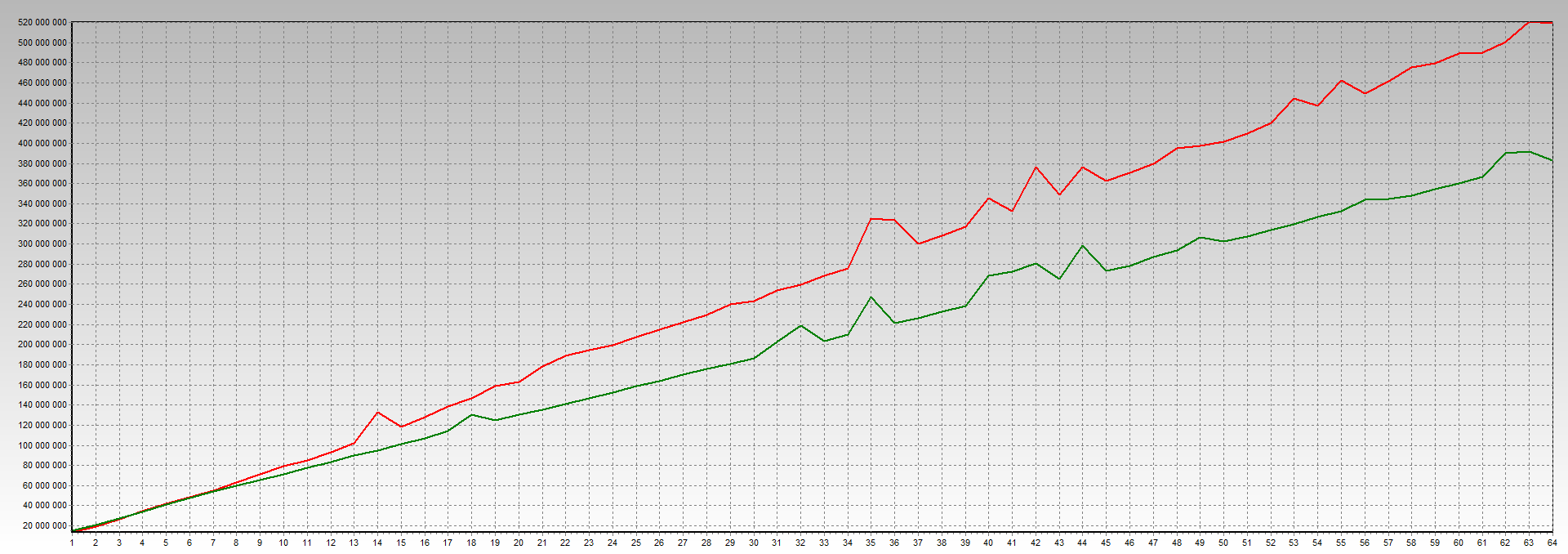
Графики полученных характеристик представлены на рисунке [4.](#_bookmark4)

Рисунок 4 – Результат исследования оптимизирующих структур данных Красный график (верхний) показывает количество тактов работы алгоритма,

использующего неоптимизированную структуру. Зеленый график (нижний) пока- зывает количество тактов работы алгоритма с использованием оптимизированной структуры. Ось абсцисс отражает количество одновременно обрабатываемых мас- сивов.

Результаты эксперимента: отношение времени обработки блока памяти неопти- мизированной структуры ко времени обработки блока структуры, обеспечиваю- щей эффективную загрузку и параллельную обработку данных = 1.3.

## Вывод

Эффективная обработка нескольких векторных структур данных без их до- полнительной оптимизации не использует в должной степени возможности аппа- ратных ресурсов.

Для создания структур данных, оптимизирующих их обработку, необходимо передавать в каждом пакете только востребованную для вычислений информа- цию.То есть для ускорения алгоритмов необходимо правильно упорядочивать дан- ные.

# Исследование конфликтов в кэш-памяти

Исходные данные: размер банка кэш-памяти данных первого и второго уровня, степень ассоциативности кэш-памяти первого и второго уровня, размер линейки кэшпамяти первого и второго уровня.

Настраиваемые параметры: размер банка кэш-памяти (16К), размер линейки кэш-памяти (64б), количество читаемых линеек (128).

Графики полученных характеристик представлены на рисунке [5.](#_bookmark5)

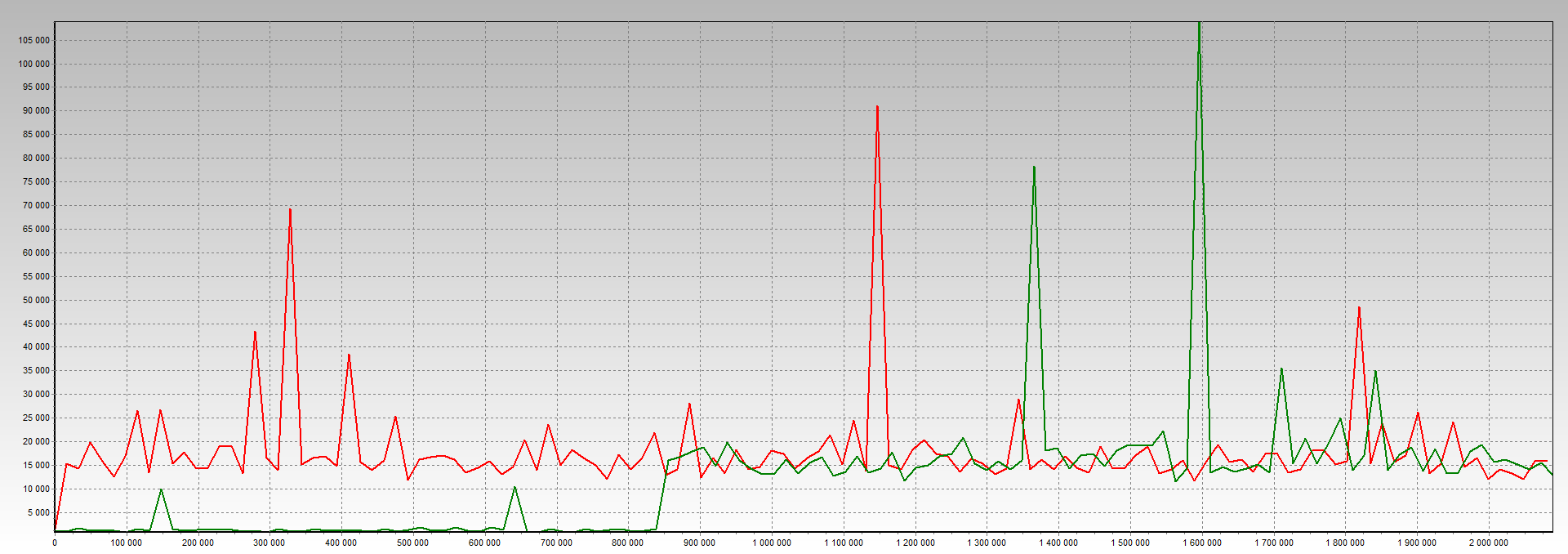


Рисунок 5 – Результат исследования конфликтов в кэш-памяти

Красный график (верхний) показывает количество тактов работы процедуры, читающей данные с конфликтами в кэш-памяти. Зеленый график (нижний) по- казывает количество тактов работы процедуры, не вызывающей конфликтов в кэш-памяти. Ось абсцисс отражает смещение читаемой ячейки от начала блока данных

Результаты эксперимента: отношение времени обработки массива с конфлик- тами в кэш-памяти ко времени обработки массива без конфликтов = 1.5.

## Вывод

Попытка читать данные из оперативной памяти с шагом, кратным размеру банка, приводит к их помещению в один и тот же набор. Если же количество за- просов превосходит степень ассоциативности кэш-памяти, т.е. количество банков или количество линеек в наборе, то наблюдается постоянное вытеснение данных из кэш-памяти, причем больший ее объем остается незадействованным. Кэш па- мять ускоряет работу процессора в 1.5 раз

# Сравнение алгоритмов сортировки

Исходные данные: количество процессоров вычислительной системы, размер пакета, количество элементов в массиве, разрядность элементов массива

Настраиваемые параметры: количество 64-х разрядных элементов массивов (2М), шаг увеличения размера массива (64К).

Графики полученных характеристик представлены на рисунке [6.](#_bookmark6)

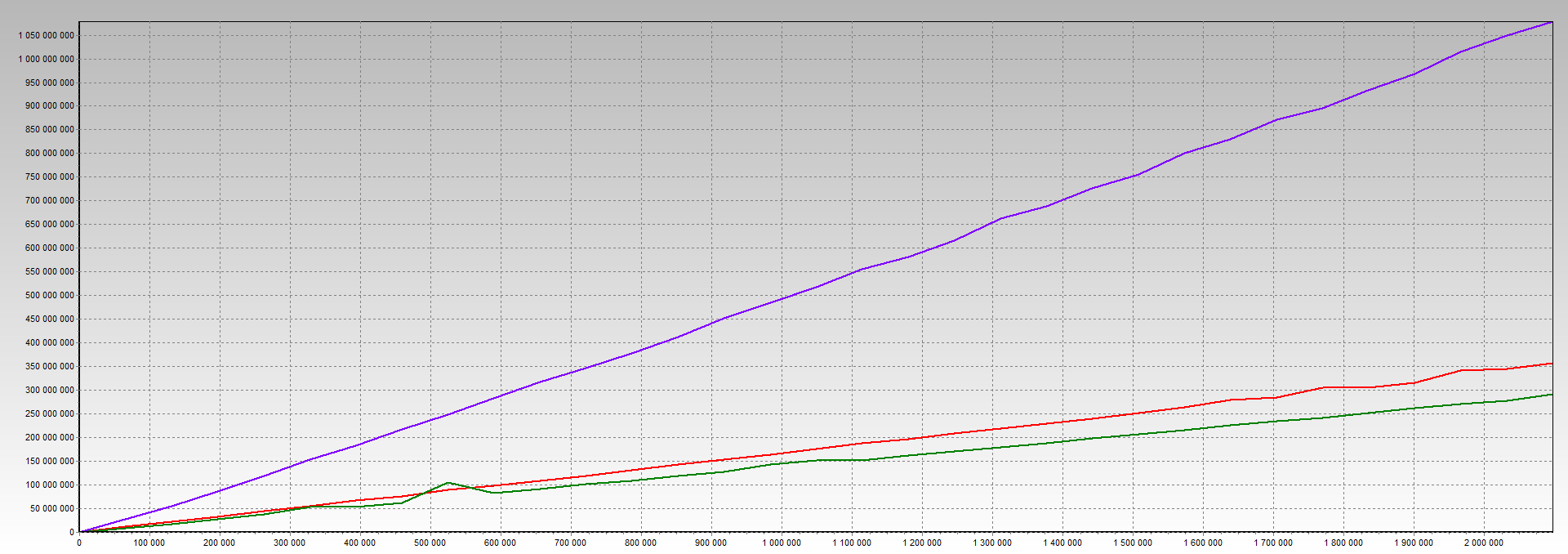


Рисунок 7 – Результат исследования алгоритмов сортировки Фиолетовый график (верхний) показывает количество тактов работы алгорит-

ма QuickSort. Красный график (средний) показывает количество тактов работы неоптимизированного алгоритма Radix-Counting. Зеленый график (нижний) по- казывает количество тактов работы оптимизированного под 8-процессорную вы- числительную систему алгоритма Radix-Counting. Ось абсцисс отражает количе- ство 64-разрядных элементов сортируемых массивов.

Результаты эксперимента: отношение времени сортировки массива алгорит- мом QuickSort ко времени сортировки алгоритмом Radix-Counting Sort (= 2.9) и ко времени сортировки Radix-Counting Sort, оптимизированной под 8-процессорную вычислительную систему (=3.5).

## Вывод

Существует алгоритм сортировки менее чем линейной вычислительной слож- ности.

# Общий вывод

В результате выполнения лабораторной работы были изучены принципы эф- фективного использования подсистемы памяти современных универсальных ЭВМ. В ходе работы проработан теоретический материал, касающийся особенностей функционирования подсистемы памяти современных конвейерных суперскаляр- ных ЭВМ, изучены возможности программы PCLAB, изучены средства иденти- фикации микропроцессоров, проведены исследования времени выполнения тесто- вых программ, сделаны выводы об архитектурных особенностях используемых

ЭВМ.