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»

Отчет по лабораторной работе № 3 по дисциплине "Архитектура ЭВМ"

Тема Организация памяти конвейерных суперскалярных электронных вычислительных машин

Студент Хамзина Р. Р.

Группа ИУ7-53Б

Оценка (баллы) _____

Преподаватель Дубровин Е. Н.

Москва — 2021 г.

Содержание

Введение

Целью данной лабораторной работы является освоение принципов эффективного использования подсистемы памяти современных универсальных ЭВМ, обеспечивающей хранение и своевременную выдачу команд и данных в центральное процессорное устройство.

Для достижения поставленной цели необходимо решить следующие задачи:

- ознакомиться с теоретическим материалом, касающимся особенностей функционирования подсистемы памяти современных конвейерных суперскалярных ЭВМ;
- изучить возможности программы PCLAB;
- изучить средства идентификации микропроцессоров;
- провести исследования времени выполнения тестовых программ;
- сделать выводы об архитектурных особенностях используемых ЭВМ.

1 Применение программы PCLAB при исследовании производительности ЭВМ

Программа PCLAB предназначена для исследования производительности x86 совместимых ЭВМ с IA32 архитектурой, работающих под управлением операционной системы Windows (версий 95 и старше).

Процесс сбора и анализа экспериментальных данных в PCLAB основан на процедуре профилировки критического кода, т.е. в измерении времени его обработки центральным процессорным устройством. Для измерения времени работы циклов в PCLAB используется следующая методика:

- длительность обработки участка профилируемой программы характеризуется изменением величины счетчика тактов процессора, произошедшим за время его работы;
- для предотвращения влияния соседних участков кода на результаты измерений, перед началом замера и после его окончания необходимо выдать команду упорядоченного выполнения CPUID, препятствующую переупорядочиванию потока команд на конвейере процессора;
- замеры количества тактов процессора необходимо повторить несколько раз;
- взаимное влияние последовательных повторов экспериментального участка программы исключается благодаря очищению кэш-памяти и буферов процессора;
- часть граничных результатов отбрасывается (как наибольших, так и наименьших). По оставшимся результатам замеров определяется средняя величина.

2 Идентификация процессора

Из выполнения команды идентификации процессора CPUID получены характеристики процессора, показанные на рисунке 2.1.

```
_____________________________________________
Vendor ID: "GenuineIntel"; CPUID level 13

Дополнительные функции Intel:
Верисия 0001067a:
Type 0 - Original OEM
Family 6 - Pentium Pro
Model 7 - Pentium III/Pentium III Xeon - external L2 cache
Stepping 10
Reserved 4

Extended brand string: "Intel(R) Celeron(R) CPU        E3300  @ 2.50GHz"
CLFLUSH instruction cache line size: 8
Initial APIC ID: 1
Hyper threading siblings: 2

Feature flags bfebfbff:
0       FPU             Присутствует Математический сопроцессор
1       VME             Поддержка расширенных возможностей обработки прерываний в режиме виртуального i8086
2       DE              Поддержка отладки
3       PSE             Поддержка страниц размером 4 MB
4       TSC             Счетчик меток реального времени
5       MSR             Поддержка команд rdmsr и wrmsr
6       PAE             Поддержка физического адреса более 32 бит
7       MCE             Поддержка исключений 18 - об аппаратных ошибках
8       CX8             Поддержка инструкции cmpxchg8b
9       APIC            Микропроцессор содержит программно доступный контроллер прерываний
11      SEP             Поддержка инструкций быстрых системных вызовов sysenter и sysexit
12      MTRR            Поддержка регистра mtrr_cap (относится к MSR-регистрам)
13      PGE             Поддержка глобальных страниц
14      MCA             Поддержка архитектуры машинного контроля
15      CMOV            Поддержка инструкций условной пересылки cmov, fcmovcc, fcomi
16      PAT             Процессор поддерживает таблицу атрибутов страницы
17      PSE-36          Процессор поддерживает 4 MB страницы, которые способны адресовать физическую память до 64 GB
19      CLFLSH          Поддержка инструкции CLFLUSH
21      DS              Поддержка записи отладочной информации
22      ACPI            Управление охлаждением процессора с помощью пустых циклов в зависимости от температуры
23      MMX             Поддержка MMX
24      FXSR            Поддержка инструкций FXSAVE и FXRSTOR
25      SSE             Поддержка SSE
26      SSE2            Поддержка SSE2
27      SS              Управление конфликтующими типами памяти
28      HTT             Поддержка Hyper-Threading
29      TM              Поддержка автоматического мониторинга температуры
31      SBF             Сигнал Останова при FERR

TLB and cache info:
b1: unknown TLB/cache descriptor
b0: дескриптор TLB-команд, 4K страницы, acc. 4-направ., 128 элементов
05: unknown TLB/cache descriptor
f0: unknown TLB/cache descriptor
57: unknown TLB/cache descriptor
56: unknown TLB/cache descriptor
78: unknown TLB/cache descriptor
30: L1 кэш-команд, 32 KB, acc. 8-направ., длина строки 64 байта
b4: unknown TLB/cache descriptor
2c: L1 кэш-данных, 32 KB, acc. 8-направ., длина строки 64 байта
Processor serial: 0001-067A-BFEB-FBFF-0400-E3BD
```

Рисунок 2.1 – Характеристики процессора

Таким образом, размер линейки кэш-памяти верхнего уровня микропроцессора равен 8, объем физической памяти микропроцессора равен 2 ГБ.

3 Эксперимент «Исследование расслоения динамической памяти»

Цель эксперимента заключается в определении способа трансляции физического адреса, используемого при обращении к динамической памяти.

3.1 Исходные данные

- размер линейки кэш-памяти верхнего уровня;
- объем физической памяти.

3.2 Настраиваемые параметры

- максимальное расстояние между читаемыми блоками, равное 6;
- шаг увеличения расстояния между читаемыми 4-х байтовыми ячейками, равный 8;
- размер массива, равный 2.

3.3 Графики полученных характеристик

На рисунке 3.1 график показывает количество тактов работы алгоритма. Ось абсцисс отражает шаг приращения адреса читаемых данных. Ось ординат отображает количество тактов.

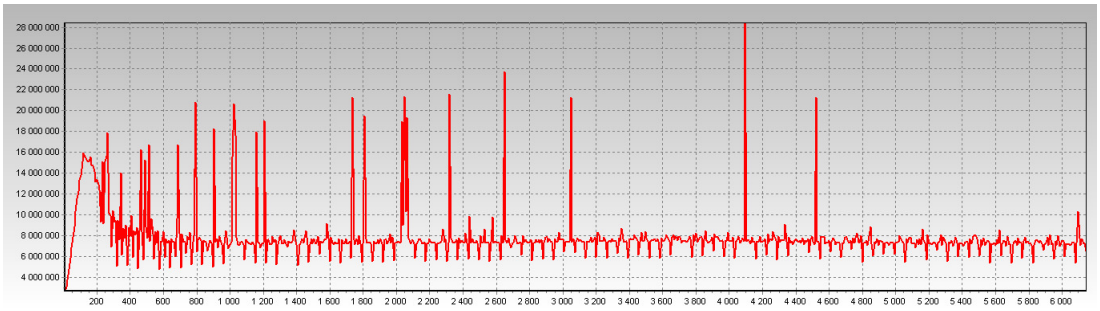

Рисунок 3.1 – Исследование расслоения динамической памяти

3.4 Результаты эксперимента

В результате исследования расслоения динамической памяти были получены следующие параметры:

- минимальный шаг чтения динамической памяти, при котором происходит постоянное обращение к одному и тому же банку, равный $T1 = 120$ Б.
- количество банков динамической памяти, равное $= 120(128)/64 = 2(1.875)$;
- расстояние между началом двух последовательных страниц одного банка, равное $T2 = 4096$ Б;
- размер одной страницы динамической памяти, равный $PC = 4096/2 = 2048$ Б;
- количество страниц в динамической памяти, равный $= 4096 * 64 * 8/(2048 * 2 * 64) = 8$.

3.5 Вывод

Оперативная память неоднородна, и для обращения к последовательно расположенным данным может потребоваться различное количество времени.

4 Эксперимент «Сравнение эффективности ссылочных и векторных структур данных»

Цель эксперимента заключается в оценке влияния зависимости команд по данным на эффективность вычислений.

4.1 Настраиваемые параметры

- количество элементов в списке, равное 1;
- максимальная фрагментация списка, равная 256;
- шаг увеличения фрагментации, равный 4.

4.2 Графики полученных характеристик

На рисунке 4.1 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. Ось ординат отображает количество тактов.

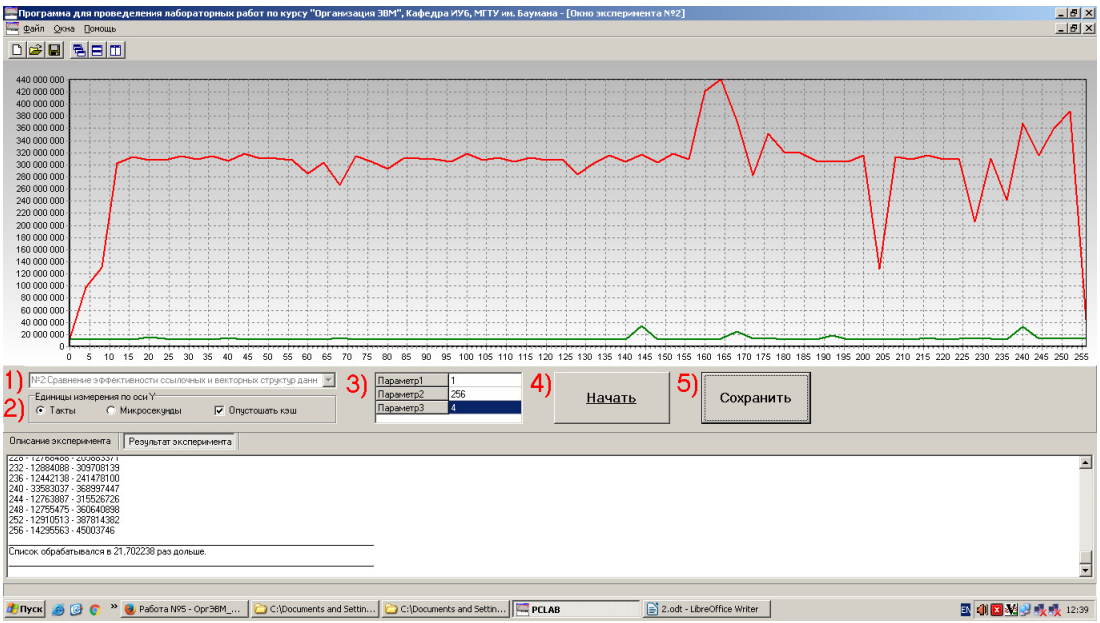

Рисунок 4.1 – Исследование эффективности ссылочных и векторных структур

4.3 Результаты эксперимента

В результате исследования эффективности ссылочных и векторных структур, показанном на рисунке 4.1, получено, что список обрабатывался в 21,702238 раз дольше.

4.4 Вывод

Использовать структуры данных необходимо с учетом технологического фактора определенной задачи.

5 Эксперимент «Исследование эффективности программной предвыборки»

Цель эксперимента заключается в выявлении способов ускорения вычислений благодаря применению предвыборки данных.

5.1 Исходные данные

- степень ассоциативности;
- размер TLB данных.

5.2 Настраиваемые параметры

- шаг увеличения расстояния между читаемыми данными, равный 64;
- размер массива, равный 128.

5.3 Графики полученных характеристик

На рисунке 5.1 красный график (верхний с острыми пиками) показывает количество тактов работы алгоритма без предвыборки. Зеленый график (нижний без значимых пиков) показывает количество тактов работы алгоритма с использованием предвыборки. Ось абсцисс отражает смещение читаемых данных от начала блока. Ось ординат отображает количество тактов.

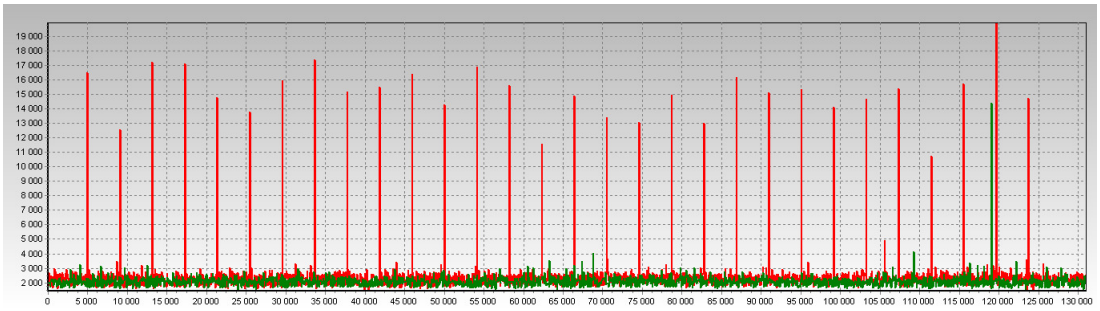

Рисунок 5.1 – Исследование эффективности предвыборки

5.4 Результаты эксперимента

В результате исследования эффективности предвыборки, показанном на рисунке 5.2, получено, что обработка без загрузки таблицы страниц в TLB производилась в 1,1726314 раз дольше.

```
130688 - 1863
130752 - 1550
130816 - 2450
130880 - 2050
130944 - 2450
131008 - 1912
_____________________________________________________________
Обработка без загрузки таблицы страниц в TLB производилась в 1,1726314 раз дольше.
_____________________________________________________________
```

Рисунок 5.2 – Результат исследования эффективности предвыборки

5.5 Вывод

За счет заблаговременной загрузки страниц в память можно ускорить время работы программы.

6 Эксперимент «Исследование способов эффективного чтения оперативной памяти»

Цель эксперимента заключается в исследовании возможности ускорения вычислений благодаря использованию структур данных, оптимизирующих механизм чтения оперативной памяти.

6.1 Исходные данные

- адресное расстояние между банками памяти;
- размер буфера чтения.

6.2 Настраиваемые параметры

- размер массива, равный 2;
- количество потоков данных, равное 64.

6.3 Графики полученных характеристик

На рисунке 6.1 красный график (верхний) показывает количество тактов работы алгоритма, использующего неоптимизированную структуру. Зеленый график (нижний) показывает количество тактов работы алгоритма с использованием оптимизированной структуры. Ось абсцисс отражает сколичество одновременно обрабатываемых массивов. Ось ординат отображает количество тактов.

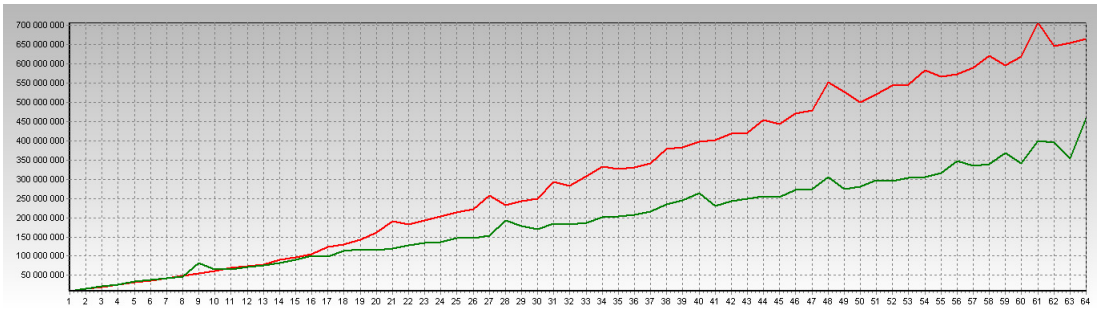

Рисунок 6.1 – Исследование оптимизирующих структур данных

6.4 Результаты эксперимента

В результате исследования оптимизирующих структур данных, показанном на рисунке 6.2, получено, что неоптимизированная структура обрабатывалась в 1,6132398 раз дольше.

```
61 - 399536925
62 - 396011838
63 - 354453450
64 - 456887213
_____________________________________________________________________
Неоптимизированная структура обрабатывалась в 1,6132398 раз дольше.
_____________________________________________________________________
```

Рисунок 6.2 – Результат исследования оптимизирующих структур данных

6.5 Вывод

Для ускорения работы алгоритмов, необходимо правильно упорядочить данные.

7 Эксперимент «Исследование конфликтов в кэш-памяти»

Цель эксперимента заключается в исследовании влияния конфликтов кэш-памяти на эффективность вычислений.

7.1 Исходные данные

- размер банка кэш-памяти данных первого и второго уровня;
- степень ассоциативности кэш-памяти первого и второго уровня;
- размер линейки кэш-памяти первого и второго уровня.

7.2 Настраиваемые параметры

- размер банка кэш-памяти, равный 256;
- количество читаемых линеек, равное 32;
- размер линейки кэш-памяти, равный 64.

7.3 Графики полученных характеристик

На рисунке 7.1 красный график (верхний) показывает количество тактов работы процедуры, читающей данные с конфликтами в кэш-памяти. Зеленый график (нижний) показывает количество тактов работы процедуры, не вызывающей конфликтов в кэш-памяти. Ось абсцисс отражает смещение читаемой ячейки от начала блока данных. Ось ординат отображает количество тактов.

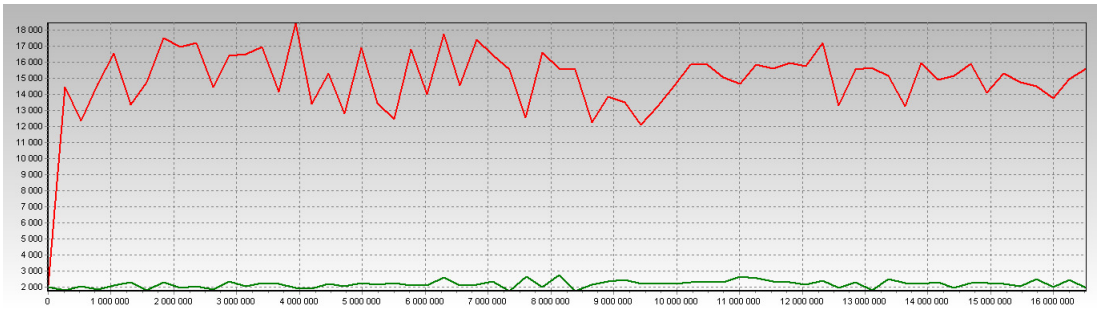

Рисунок 7.1 – Исследование конфликтов в кэш-памяти

7.4 Результаты эксперимента

В результате исследования конфликтов в кэш-памяти, показанном на рисунке 7.2, получено, что чтение с конфликтами банков производилось в 6,7962867 раз дольше.

```
15468384 - 2050
15730560 - 2525
15992736 - 2013
16254912 - 2450
16517088 - 1975

Чтение с конфликтами банков производилось в 6,7962867 раз дольше.
```

Рисунок 7.2 – Результат исследования конфликтов в кэш-памяти

7.5 Вывод

Использование кэш-памяти ускоряет работу процессора почти в 7 раз.

8 Эксперимент «Исследование алгоритмов сортировки»

Цель эксперимента заключается в исследовании способов эффективного использования памяти и выявлении наиболее эффективных алгоритмов сортировки, применимых в вычислительных системах.

8.1 Исходные данные

- количество процессоров вычислительной системы;
- размер пакета;
- количество элементов в массиве;
- разрядность элементов массива.

8.2 Настраиваемые параметры

- количество 64-х разрядных элементов массивов, равное 10;
- шаг увеличения размера массива, равный 1024.

8.3 Графики полученных характеристик

На рисунке 8.1 синий график (верхний) показывает количество тактов работы алгоритма QuickSort. Красный график (средний) показывает количество тактов работы неоптимизированного алгоритма Radix-Counting. Зеленый график (нижний) показывает количество тактов работы оптимизированного под 8-процессорную вычислительную систему алгоритма Radix-Counting. Ось абсцисс отражает количество 64-разрядных элементов сортируемых массивов. Ось ординат отображает количество тактов.

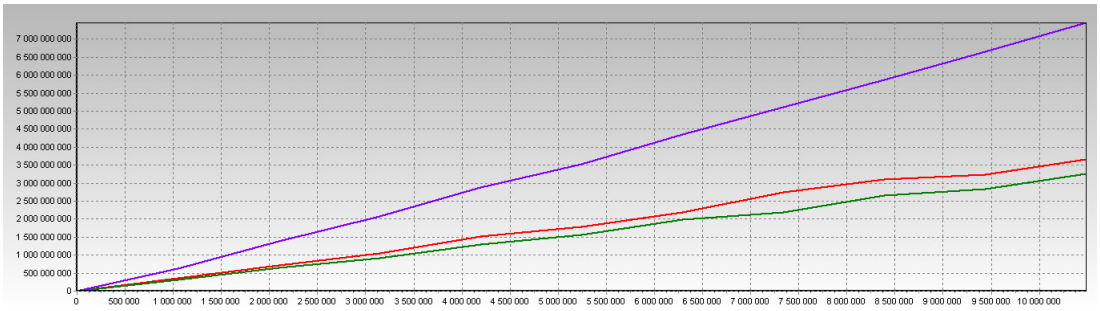

Рисунок 8.1 – Исследование алгоритмов сортировки

8.4 Результаты эксперимента

В результате сравнения времени работы алгоритмов сортировки, показанном на рисунке 8.2, получено, что QuickSort работал в 1,963879 раз дольше Radix-Counting Sort, и QuickSort работал в 2,2629607 раз дольше Radix-Counting Sort, оптимизированного под 8-процессорную ЭВМ.

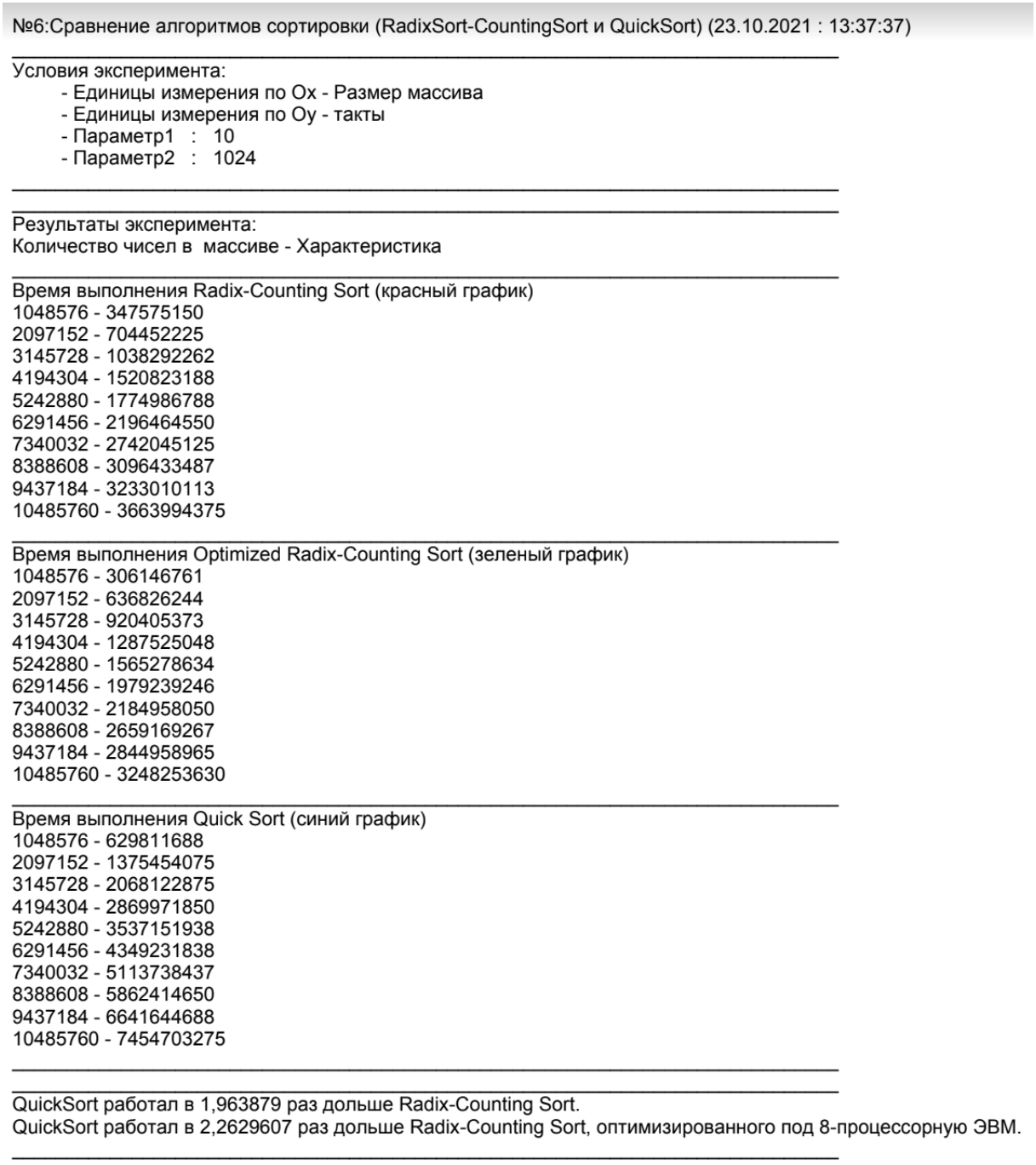

Рисунок 8.2 – Результат сравнения времени работы

8.5 Вывод

Radix-Counting Sort является более быстрой сортировкой, чем QuickSort, при этом Radix-Counting Sort можно улучшить, оптимизировав алгоритм под 8-процессорную ЭВМ.

9 Контрольные вопросы

1. Назовите причины расслоения оперативной памяти.

Причиной расслоения оперативной памяти является дисбаланс в скорости доступа к данным, размещенным в оперативной памяти, и производительностью процессора.

2. Как в современных процессорах реализована аппаратная предвыборка?

Аппаратная предвыборка происходит без участия программиста или компилятора. Кэш-контроллер анализирует, по каким адресам и в каком порядке программа обращается к оперативной памяти, и пытается предугадать, какие данные вскоре могут понадобиться программе, и осуществляет их автоматическую предвыборку в кэш-память.

3. Какая информация храниться в TLB?

TLB (Translation-Lookaside Buffer) представляет собой память с ассоциативной выборкой, которая содержит 20-тиразрядные базовые адреса 32-х страниц, то есть, старшие 20 разрядов физического адреса страницы.

Каждый из базовых адресов имеет свой признак - тег. В качестве тега используются старшие 20 разрядов линейного адреса, то есть поля TABLE и PAGE.

Помимо тега для каждого базового адреса страницы в TLB хранится дополнительная информация, позволяющая определить, какую страницу можно заменить на вновь вводимую.

4. Какой тип ассоциативной памяти используется в кэш-памяти второго уровня современных ЭВМ и почему?

В современных компьютерах применяют кэш-память второго уровня, которая находится между процессором и оперативной памятью и еще больше повышает производительность ЭВМ, потому что кэш-память 2-го уровня, может содержать как команды, так и данные.

5. Приведите пример программной предвыборки.

Примерами программной предвыборки являются предвыборка данных в цикле для независимости от информации находящейся вне цикла и предвыборка в основных блоках, которые часто исполняются, но которые редко используют ее независимо от целей предвыборки.

Заключение

В данной лабораторной работе были освоены принципы эффективного использования подсистемы памяти современных универсальных ЭВМ, обеспечивающей хранение и своевременную выдачу команд и данных в центральное процессорное устройство. Цель, поставленная перед началом работы, была достигнута. В ходе лабораторной работы были решены все поставленные задачи.