Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Сибирский государственный университет телекоммуникаций и информатики»

(СибГУТИ)

Кафедра вычислительных систем

**ОТЧЕТ**

по лабораторной работе

по дисциплине «Параллельные вычислительные технологии»

Выполнил:

студент группы

ИС-241 Жур А. А.

Проверил:

ассистент кафедры ВС

Челканова Т. В.

Новосибирск 2024

**Оглавление**

[ПОСТАНОВКА ЗАДАЧИ 3](#_TOC_250003)

[ОПИСАНИЕ ТЕОРИИ МЕТОДА РЕШЕНИЯ ЗАДАЧИ 5](#_TOC_250002)

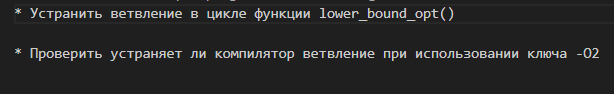
[СХЕМА АЛГОРИТМА РЕШЕНИЯ 6](#_TOC_250001)

ЛИСТИНГ ПРОГРАММЫ С КОММЕНТАРИЯМИ 7

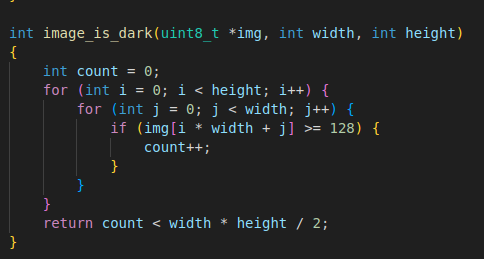
[РЕЗУЛЬТАТ РАБОТЫ ПРОГРАММЫ 9](#_TOC_250000)

# ПОСТАНОВКА ЗАДАЧИ

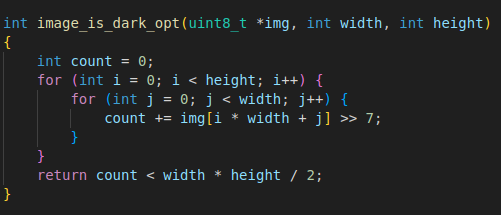
Практическая работа 1.1 (cache)



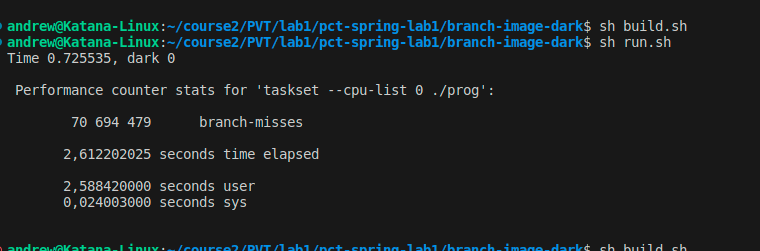
Рассмотрим код исходной функции:

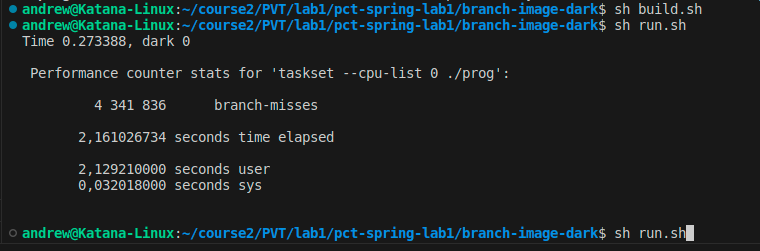


Эта функция проверяет тёмное ли это изображение. Тёмным считается изображение, в котором меньше половины пикселей имеют значение интенсивности >=128. У нас есть 2 цикла, проверяющих это условие. В них мы проходим по всем пикселям и считаем только яркие.

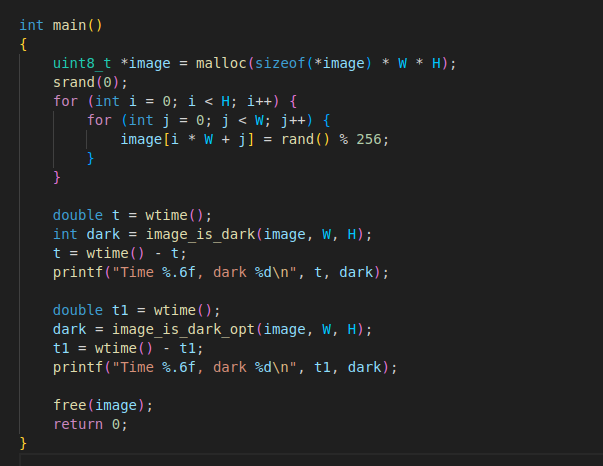


Это та же функция, но с устранённым ветвлением в циклах. Тут используется побитовый сдвиг в качестве деления, и к счётчику count прибавляется 0 или 1(сдвигаем на 7 т. к. 128 это 2^7 и в двоичном виде 1 стоит в 7 разряде).

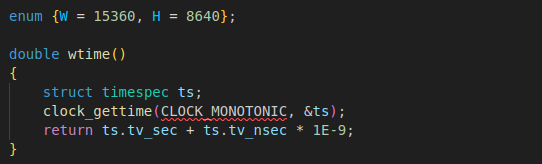




На первом скриншоте продемонстрирована работа программы с ветвлением, а на втором с его устранением. С помощью perf выводится информация о branch-misses -это количество промахов модуля предсказания переходов.

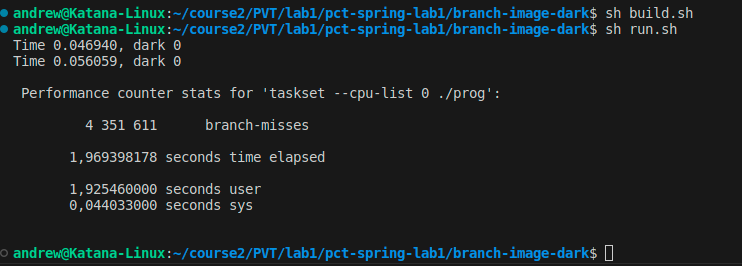


В функции main мы генерируем наше изображение а затем считаем и выводим время для версии с ветвлением и оптимизированной версии.



Тут представлены размер нашей картинки в пикселях и вспомогательная функция для подсчёта времени.

Дальше проверим, устраняет ли ключ -О2 ветвление:

Получается, что при ключе -О2 компилятор устраняет ветвление. Программа выполнилась ещё быстрее и с меньшем количеством промахов.

Ответы на контрольные вопросы по блоку 1.1 (cache):

1. Неэффективно используется кеш-память при обращении к массиву с большим шагом, превышающим размер строки кеша. Например, если шаг равен размеру строки кеша или более. Тогда каждая следующая выборка данных вызывает cache miss, что значительно замедляет выполнение программы.

2. Физический адрес разбивается на три части: индекс набора, тэг и смещение. 1. По полю Set index выбирается одно из S множеств 2. Среди E записей множества отыскивается строка с требуемым полем Tag и установленным битом Valid. Найдена – cache hit, не найдена – cache miss 3. Данные из блока считываются с заданным смещением Block offset.

3. Если все каналы множества заняты, кеш-память использует политику замещения для выбора линии кеша, которая будет заменена новыми данными. Алгоритм замещения строк кеш-памяти (replacement policy) — выбирает строку и удаляет из кеш-памяти для размещения новой записи.

Алгоритмы требуют хранения вместе с каждой строкой кеш-памяти специализированного поля флагов/истории (age bits).

LRU (Least Recently Used) – вытесняется наименее востребованную строку (2Q, LRU/K).

RR (Random Replacement) – вытесняет случайную строку.

Алгоритм L. Belady – вытесняет запись, которая с большой вероятностью не понадобиться в будущем (IBM Research, 1966).

4. Политика write-through (сквозная запись) – запись в кеш-память влечет за собой немедленное обновление данных в кеш-памяти и оперативной памяти (кеш “отключается”)

Политика write-back (отложенная запись, copy-back) – первоначально данные записываются только в кеш-память

Строки нет в кеш-памяти (write miss):

 В кеш-памяти выделяется строка

 Из оперативной памяти загружается строка, соответствующая адресу

 Необходимые байты изменяются в загруженной строке кеш-памяти

 Строка помечается как модифицированная (dirty)

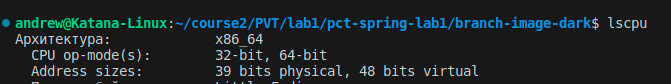
Строка присутствует в кеш-памяти (write hit):

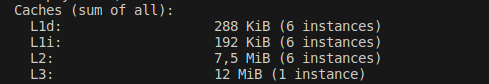
 Необходимые байты изменяются в строке кеш-памяти

 Строка помечается как модифицированная (грязная, dirty)

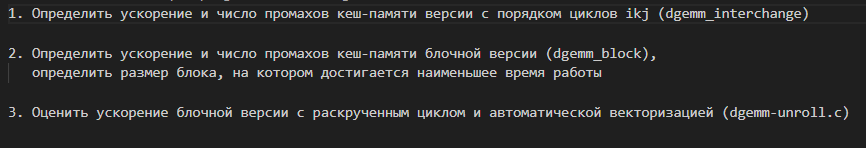
Запись в память модифицированных строк осуществляется при их замещении

5.

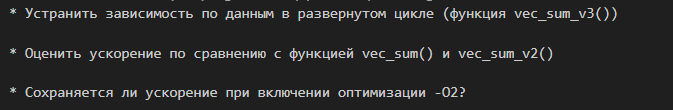




Практическая работа 1.2 (branch)



Практическая работа 1.3 (loop-unrolling)



# Задание 1.1

# Задание 1.2

# Задание 1.3