Министерство образования Республики Беларусь

Учреждение образования «Белорусский государственный университет

информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

Кафедра Информатики

Дисциплина «Архитектура вычислительных систем»

**ОТЧЕТ**

к практической работе №1

на тему:

**«РАБОТАТЬ С ПРОЦЕССОРАМИ РАЗЛИЧНЫХ СЕМЕЙСТВ И ПОКОЛЕНИЙ В РЕЖИМЕ ОДНОГО ЯДРА»**

БГУИР 6-05-0612-02 01

|  |
| --- |
| Выполнила студентка группы 353504  АНТОНОВА Лидия Сергеевна |
|  |
| (дата, подпись студента) |
| Проверила  КАЛИНОВСКАЯ Анастасия Александровна |
|  |
| (дата, подпись преподавателя) |

Минск 2025

# Цель работы

Осуществить методом математического выполнения функции согласно варианту задания.

# Индивидуальное задание

Выбрав вариант задания, а именно, заданную функцию осуществить написание программы на любом удобном языке программирования с вставками кода ассемблера для выполнения задачи на i количестве итерации (i>5000) для получения достоверных результатов эксперимента. Выполнение задачи должно осуществляться в операционной системе без графической оболочки (Linux Ubuntu, Debian и др. – можно в LiveCD режиме, можно посредством установки второй системы на ноутбук). Построить графики нагрузки на одно ядро процессора. По оси Х – время выполнения, по оси Y – количество итераций (i). Для Intel – два графика на одно ядро, с функцией Hyper-Threading и без. Для AMD – с функцией SMT и без. Для ARM – одно низкопроизводительное ядро и одно высокопроизводительное ядро.

# Выполнение работы

Для выполнения задачи была разработана программа, реализующая вычисление значения функции, суммы и количества итераций до достижения заданной точности. Программа была написана на языке C++ с использованием вставок кода на ассемблере для оптимизации вычислений. Также проведен эксперимент по анализу производительности на одном ядре процессора с включенным и отключенным режимом Hyper-Threading. Время выполнения каждого цикла программы замерялось с использованием библиотеки <chrono>. Программа автоматически ограничивала выполнение на одно ядро, используя параметр sched\_setaffinity, чтобы исключить влияние многопоточности. Функция Hyper-Threading была отключена путем изменения параметров загрузчика GRUB, добавив nosmt в параметры загрузки ядра Linux. Основной листинг программы:

#include <iostream>

#include <chrono>

#include <fstream>

#include <cmath>

std::ofstream file("data.txt", std::ios::out);

std::pair<int, double> SolveX(double x, double Y, double ep) {

double iter = 1;

double kx, term;

double result = 0.0;

while (fabs(result - Y) >= ep) {

kx = iter \* x;

term = 0.0;

\_\_asm\_\_ volatile (

"fldl %2;" // Загружаем iter

"fldl %1;" // Загружаем kx в FPU стек

"fcos;" // Вычисляем cos(kx)

"fdivp;" // Деление cos(kx) / iter

"fstpl %0;" // Сохраняем результат в term

: "=m"(term) // Выходной операнд

: "m"(kx), "m"(iter) // Входные операнды

);

\_\_asm\_\_ volatile (

"fldl %1;" // Загружаем term в FPU стек

"faddl %0;" // Складываем term и result

"fstpl %0;" // Сохраняем результат обратно в result

: "=m"(result) // Выходной операнд

: "m"(term) // Входной операнд

);

iter++;

}

return std::pair((int)iter, result);

}

double SolveY(double x){

return -(log(fabs(2 \* sin(x / 2))));

}

int main() {

double a, b, h, epsilon;

// std::cout << "Введите a: ";

// std::cin >> a;

a = 0.01;

// std::cout << "Введите b: ";

// std::cin >> b;

b = 1;

// std::cout << "Введите шаг h: ";

// std::cin >> h;

h = 0.01;

// std::cout << "Введите epsilon: ";

// std::cin >> epsilon;

epsilon = 0.0005;

std::cout << " x | Y(x) | S(x) | Итерации | Время (сек) " << std::endl;

std::cout << "-------------------------------------------------" << std::endl;

auto start = std::chrono::high\_resolution\_clock::now();

for(double x = a; x <= b; x += h) {

double Y = SolveY(x);

std::pair<int, double> resultS = SolveX(x, Y, epsilon);

auto end = std::chrono::high\_resolution\_clock::now();

std::chrono::duration<double> elapsed = end - start;

std::cout << x << " | " << Y << " | " << resultS.second << " | " << resultS.first << " | " << elapsed.count() << "\n";

file << resultS.first << " " << elapsed.count() << "\n";

}

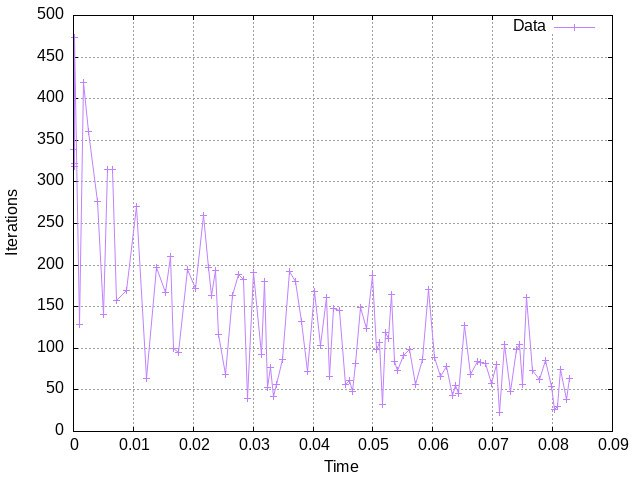
file.close();

return 0;

}

# АНАлиз полученных результатов

На рисунке 1 показан график нагрузки на одно ядро процессора с включенным и отключенным режимом Hyper-Threading.



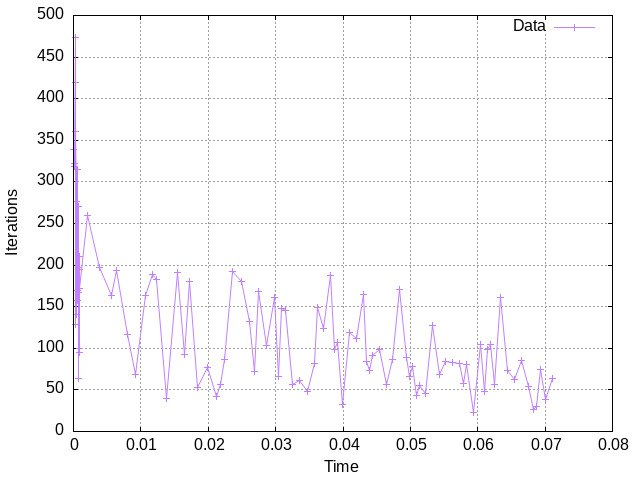
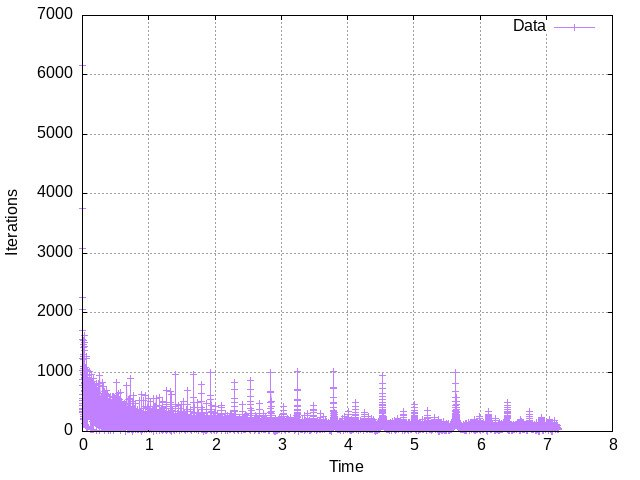


Рисунок 1

На небольшом количестве итераций (до 5000) программа показывает идентичное время выполнения. После 5000 итераций отключенный режим SMT начинает показывать более высокую производительность (рисунок 2).



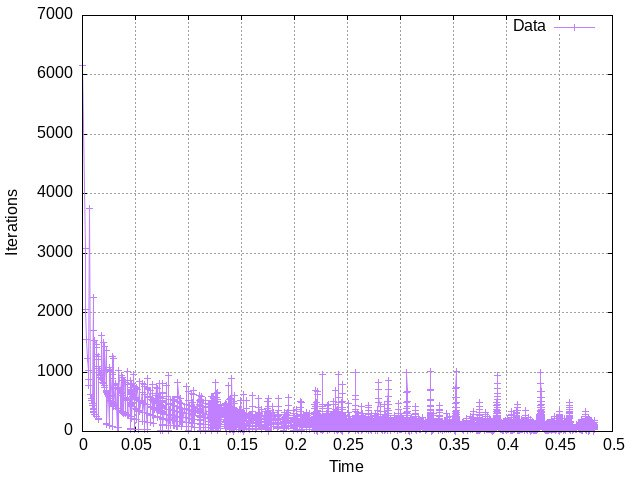


Рисунок 2

# выводы

С отключенной функцией Hyper-Threading программа демонстрирует более стабильное время выполнения на большом количестве итераций. Отключение Hyper-Threading эффективно снижает нагрузку на одно ядро процессора, что подтверждено графиками.

В результате проделанной работы подтверждено, что использование одного ядра без SMT предпочтительнее для долгих вычислений при большом количестве итераций.