МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

**Лабораторная работа №1**
**по курсу «Программирование графических процессоров»**

# Изучение технологии CUDA

Выполнил: А.С. Федоров
Группа: 8О-407Б
Преподаватели: К.Г. Крашенинников,
А.Ю. Морозов

Москва, 2022

## Условие

**Цель работы.** Ознакомление и установка программного обеспечения для работы с программно-аппаратной архитектурой параллельных вычислений(CUDA). Реализация одной из примитивных операций над векторами.
**Вариант задания.** 8

## Программное и аппаратное обеспечение

**Графический процессор**
Название: NVIDIA GeForce GTX 1050
Compute capability: 6.1
Объем графической памяти: 2147352576 байтов
Объем разделяемой памяти на блок: 49152 байтов
Объем регистров на блок: 65536
Размер варпа: 32
Максимальное количество потоков на блок: (1024, 1024, 64)
Максимальное число блоков: (2147483647, 65535, 65535)
Объем постоянной памяти: 65536 байтов
Число мультипроцессоров: 5

**Процессор**
Название: i5-8250U
Базовая тактовая частота процессора: 1,60 ГГц
Количество ядер: 4
Количество потоков: 8
Кеш L1: 256 Кб
Кеш L2: 1 Мб
Кеш L3: 6 Мб

**Оперативная память**
Тип: DDR4
Объем: 11.9 Гб
Частота: 2400 МГц

**Програмное обеспечение**
ОС: WSL2 (Windows 11)
IDE: Microsoft Visual Studio 2022 (аддон NVIDIA Nsight)
Компилятор: nvcc

## Метод решения

Сложение элементов векторов происходит параллельно, если размерность оказалась больше, чем количество потоков, то каждый поток складывает элементы по индексу своего номера и далее до конца векторов с шагом $n$, где $n$ – число потоков.

## Описание программы

Разделение по файлам, описание основных типов данных и функций. Обязательно описать реализованные ядра.

## Результаты

| | **Размер теста** | | | | |
|---|---|---|---|---|---|
| | $2^5$ | $2^{10}$ | $2^{15}$ | $2^{20}$ | $2^{25}$ |
| **GPU(CUDA) <1, 32>** | 2.076 мс | 1.438 мс | 1.788 мс | 14.743 мс | 323.914 мс |
| **GPU(CUDA) <32, 32>** | 1.204 мс | 1.372 мс | 1.577 мс | 2.062 мс | 17.102 мс |
| **GPU(CUDA) <64, 64>** | 1.456 мс | 1.401 мс | 1.400 мс | 2.449 мс | 9.962 мс |
| **GPU(CUDA) <128, 128>** | 2.495 мс | 1.165 мс | 1.400 мс | 2.381 мс | 10.794 мс |
| **GPU(CUDA) <256, 256>** | 1.726 мс | 1.359 мс | 1.407 мс | 2.017 мс | 12.124 мс |
| **GPU(CUDA) <512, 512>** | 1.431 мс | 1.441 мс | 1.408 мс | 2.554 мс | 10.246 мс |
| **GPU(CUDA) <1024, 1024>** | 1.420 мс | 1.432 мс | 1.493 мс | 1.786 мс | 10.476 мс |
| **CPU** | >0.000 мс | 0.009 мс | 0.0212 мс | 5.220 мс | 212.225 мс |

## Выводы

Сложение векторов может быть использовано везде, где используется линейная алгебра, начиная от компьютерной графики и заканчивая экономикой. Одной из типовых задач, которую эффективно способен решить данный алгоритм – это вычисление функции ошибки для обучения нейросетевых моделей. Например, для вычисления MSE необходимо складывать вектора признаков объектов.

При программировании данной лабораторной работы существенных проблем не возникло.

Исходя из результатов тестирования алгоритма можно сделать выводы, что параллельное вычисление результата на GPU существенно эффективнее вычисления на CPU, когда размерность векторов очень велика. Похоже, что на инициализацию работы графического процессора требуется время (около 1.4 мс), поэтому тесты на сравнительно небольших данных дают примерно один и тот же результат. С ростом числа блоков и потоков, время на вычисление снижается.