МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ

(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Кафедра вычислительной математики и программирования

**спецкурс «Параллельные и распределенные вычисления»**

**ОТЧЕТ**

**Лабораторная работа № 1**

**«Программирование CUDA»**

Выполнил: Савельев А.С.

Группа: М80-107-22

Преподаватель: Семенов С. А.

Москва, 2022

Содержание

[1. Постановка задачи 2](#_Toc52570380)

[2. Описание решения 2](#_Toc52570381)

[3. Аппаратное обеспечение и ПО 2](#_Toc52570382)

[4. Основные моменты кода 2](#_Toc52570383)

[5. Результат работы программы 2](#_Toc52570384)

[6. Сравнение скорости выполнения на CPU и GPU 2](#_Toc52570385)

[7. Выводы 3](#_Toc52570386)

[8. Приложения 3](#_Toc52570387)

# 1. Постановка задачи

Вариант 12: задать массив чисел и вычислить их квадраты

# Описание решения

Результатом выполнения программы является выходной массив *out* квадратов значений входного массива *in*

# Основные моменты кода

Реализация алгоритма на GPU и CPU выглядит следующим образом:

#define N 65536

\_\_global\_\_ void square(int \* in, int \* out)

{

out[blockIdx.x] = in[blockIdx.x] \* in[blockIdx.x];

}

void squareCPU(int\* in, int\* out)

{

for (int i = 0; i < N; i++)

{

out[i] = in[i] \* in[i];

}

}

В виду простоты алгоритма видимость каких-либо временных интервалов возможна лишь при многоразовым исполнении программы на CPU и GPU. для этих целей вызов устройства и вызов функции выполняются многократно с помощью цикла for:

double executionTime;

int iterations = 1000;

// Исполнение на устройстве

clock\_t startGPU = clock();

for (int i = 0; i < iterations; i++)

{

square <<<N, 1 >>> (dev\_in, dev\_out);

}

executionTime = ((double)clock() - startGPU) / CLOCKS\_PER\_SEC;

printf("\nGPU Execution time is %.10lf", executionTime);

// Исполнение на хосте

clock\_t startСPU = clock();

for (int i = 0; i < iterations; i++)

{

squareCPU(in, out);

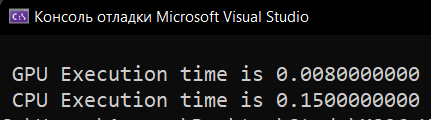
}

executionTime = ((double)clock() - startСPU) / CLOCKS\_PER\_SEC;

printf("\nCPU Execution time is %.10lf", executionTime);

# 5. Результат работы программы

# 

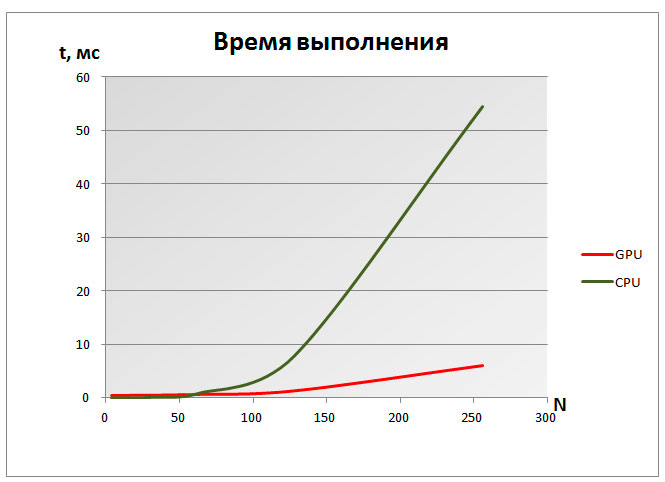


# 6. Сравнение скорости выполнения на CPU и GPU

При запуске программы с различными значениями *N* видно, что вычисления на видеокарте произвелись быстрее, чем на процессоре компьютера, при *N*> 32 (Количество итераций равно 100)

Время выполнения программы при различных значениях *N*:

|  |  |  |  |
| --- | --- | --- | --- |
| **N** | **GPU**  **время выполнения, мс** | **CPU**  **время выполнения, мс** | **tCPU/tGPU** |
| 1024 |  |  |  |
| 2048 |  |  |  |
| 4096 |  |  |  |
| 8192 |  |  |  |
| 16384 |  |  |  |
| 32768 |  |  |  |
| 65536 | 0.001000 | 0.016300 |  |



**Рис.** График зависимости времени выполнения программы от порядка матрицы *N*.

# 7. Выводы

В Лабораторной работе №1 проведен анализ работы различных программ по решению задачи возведения в квадрат значений массива. Из графика видно, что рост производительности GPU относительно CPU коррелирует с ростом размера обрабатываемого массива значений.

# 8. Приложения