МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Информационные технологии и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №1
по курсу «Программирование графических процессоров»

Освоение программного обеспечения для работы с технологией CUDA. Примитивные операции над векторами.

Выполнил: К.О.Вахрамян
Группа: 8О-406Б
Преподаватели: К.Г. Крашенинников,
А.Ю. Морозов

Москва, 2021

Условие

1. Цель работы, общая постановка задачи (один абзац).

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

2. Вариант задания.

Входные данные. На первой строке задано число n -- размер векторов. В следующих 2-х строках, записано по n вещественных чисел -- элементы векторов.
Выходные данные. Необходимо вывести n чисел -- результат сложения исходных векторов.

Программное и аппаратное обеспечение

GPU:

--- General Information for device ---
Name: NVIDIA GeForce GTX 1650
Compute capability: 7.5
Clock rate: 1560000
Device copy overlap: Enabled
Kernel execution timeout : Enabled
--- Memory Information for device ---
Total global mem: 4100521984
Total constant Mem: 65536
Max mem pitch: 2147483647
Texture Alignment: 512
--- MP Information for device ---
Multiprocessor count: 16
Shared mem per mp: 49152
Registers per mp: 65536
Threads in warp: 32
Max threads per block: 1024
Max thread dimensions: (1024, 1024, 64)
Max grid dimensions: (2147483647, 65535, 65535)

CPU:

Architecture: x86_64
CPU op-mode(s): 32-bit, 64-bit
Byte Order: Little Endian
Address sizes: 39 bits physical, 48 bits virtual
CPU(s): 8
On-line CPU(s) list: 0-7
Thread(s) per core: 2
Core(s) per socket: 4

Socket(s): 1
NUMA node(s): 1
Vendor ID: GenuineIntel
CPU family: 6
Model: 158
Model name: Intel(R) Core(TM) i5-9300HF CPU @ 2.40GHz
Stepping: 13
CPU MHz: 1274.759
CPU max MHz: 2400.0000
CPU min MHz: 800.0000
BogoMIPS: 4800.00
Virtualization: VT-x
L1d cache: 128 KiB
L1i cache: 128 KiB
L2 cache: 1 MiB
L3 cache: 8 MiB

OS:
Linux Mint 20

Compiler:
nvcc

Code Editor:
VS Code

Метод решения

Сложение векторов происходит на девайсе. Число блоков и количество потоков в них можно представить в виде сетки. В каждом потоке индекс суммируемых элементов рассчитывается как произведение номера блока на размерность блока + номер потока. (idx = threadIdx.x + blockIdx.x * blockDim.x). Индекс инкрементируется произведением размеронсти блока на число блоков.

Описание программы

Т.к. это первая л.р., вся программа реализована в одном фале. Код ядра принимает в качестве аргументов векторы в виде указателей на тип double и размерность векторов. Сумма записывается в 3-й указатель.

Результаты

Все время представлено в миллисекундах.

	n=2**8	n=2**15	n=2**24
<<<1,32>>>	0.01597	0.23776	205.43732
<<<32,32>>>	0.01405	0.01936	7.66051
<<<1,128>>>	0.01466	0.07078	52.83277
<<<128,128>>>	0.01462	0.01421	3.67539
<<<1,1024>>>	0.01424	0.02474	8.76506
<<<1024,1024>>>	0.02867	0.03034	3.60547
<<<65535,1024>>>	0.91270	0.91482	4.33424
CPU	0.00810	0.21299	111.85571

Выводы

Из приведенных результатов мы видим, что с увеличением размерности векторов вычисления на устройстве происходят намного быстрее вычислений на CPU. Кроме того слишком большое число блоков и потоков не дает самый быстрый результат. Такое значительное ускорение арифметических действий позволяет решать очень сложные задачи вычислительной математики. А простота CUDA C делает это очень доступным.