Vienna University of Technology

360.252 Computational Science on Many Core Architectures

Institute for Microelectronics

Exercise 8

Authors:
Camilo Tello Fachin
12127084

Supervisor:
Dipl.-Ing. Dr.techn. Karl Rupp

December 13, 2022

TECHNISCHE
UNIVERSITÄT
WIEN
Vienna University of Technology

Abstract

Here documented the results of Exercise 8.

Contents

1 Algortihm in CUDA with double precision arithmetic and data types (3/4 Points)

1.1 OpenCL Kernel to Compute Dot Product of two vectors

The given code skelleton `vector_add.cpp` was adapted in a very rudimentary way without shared memory or any other bamboozles! The Sum of the entries later computed on the CPU.

C++ Code Changes in `vector_add.cpp`

```
1  const char *my_opencl_program = R"(
2  #pragma OPENCL EXTENSION cl_khr_fp64 : enable // required to enable 'double' inside OpenCL programs
3
4  __kernel void vec_add( __global double *x,
5                         __global double *y,
6                         unsigned int N)
7  {
8    for (unsigned int i  = get_global_id(0);
9                      i  < N;
10                     i += get_global_size(0))
11     x[i] = x[i] * y[i];
12 })";
13
14
15 double dot_product = 0;
16 for(int i = 0; i<vector_size; i++){
17   dot_product += x[i];
18 }
```

One "advantage" of this blatantly simple kernel for the dot product, is that without introducing shared memory, partial exercise 3 works without problems if the device is set to CPU.

1.2 Compare Performance of OpenCL Kernels to Cuda Kernels

I used the CUDA Kernel for the dot product from exercise sheet 2 for comparison. For this rudimentary Kernel, the CUDA kernels outperform the OpenCL Kernels on both GPU's.

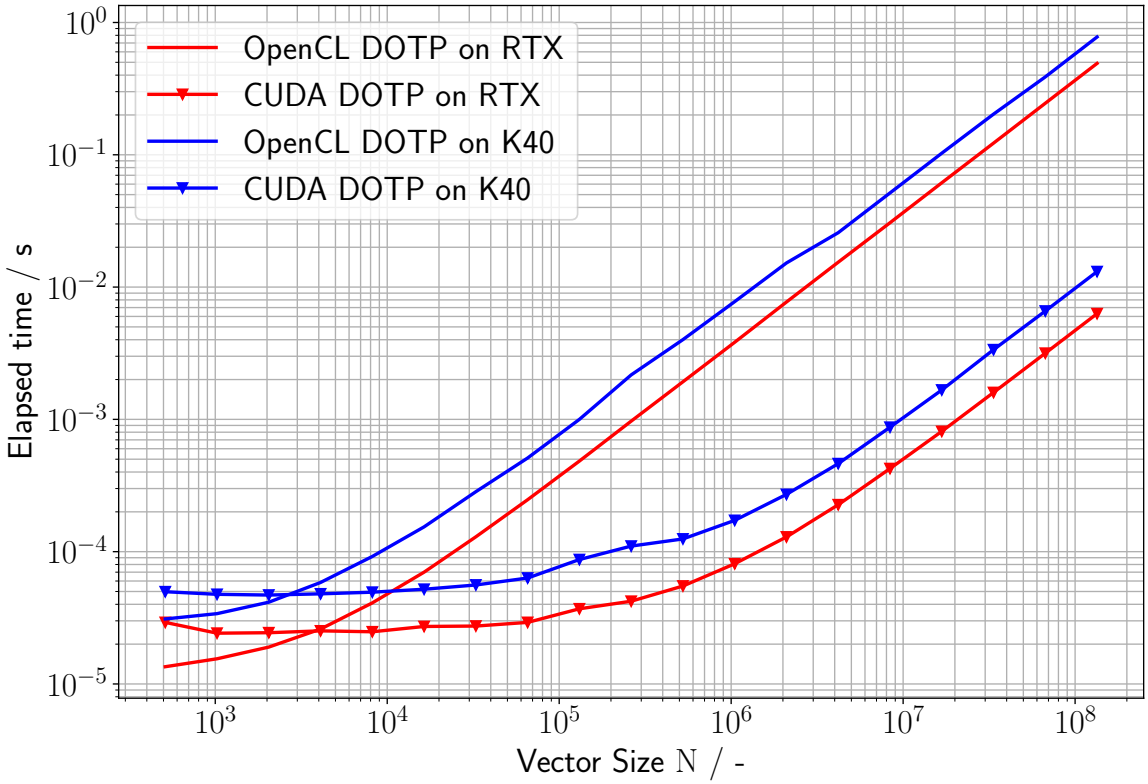

Figure 1: Kernel Performance Comparison

1.3 Compare Performance of OpenCL Kernels on K40 CPU

Like mentioned in part 1, here one simply has to change a 0 to a 1 and one can confirm in the output that the CPU is actually used. CUDA obviously still the fastest. OpenCL Kernel run on K40 is faster than on the CPU.

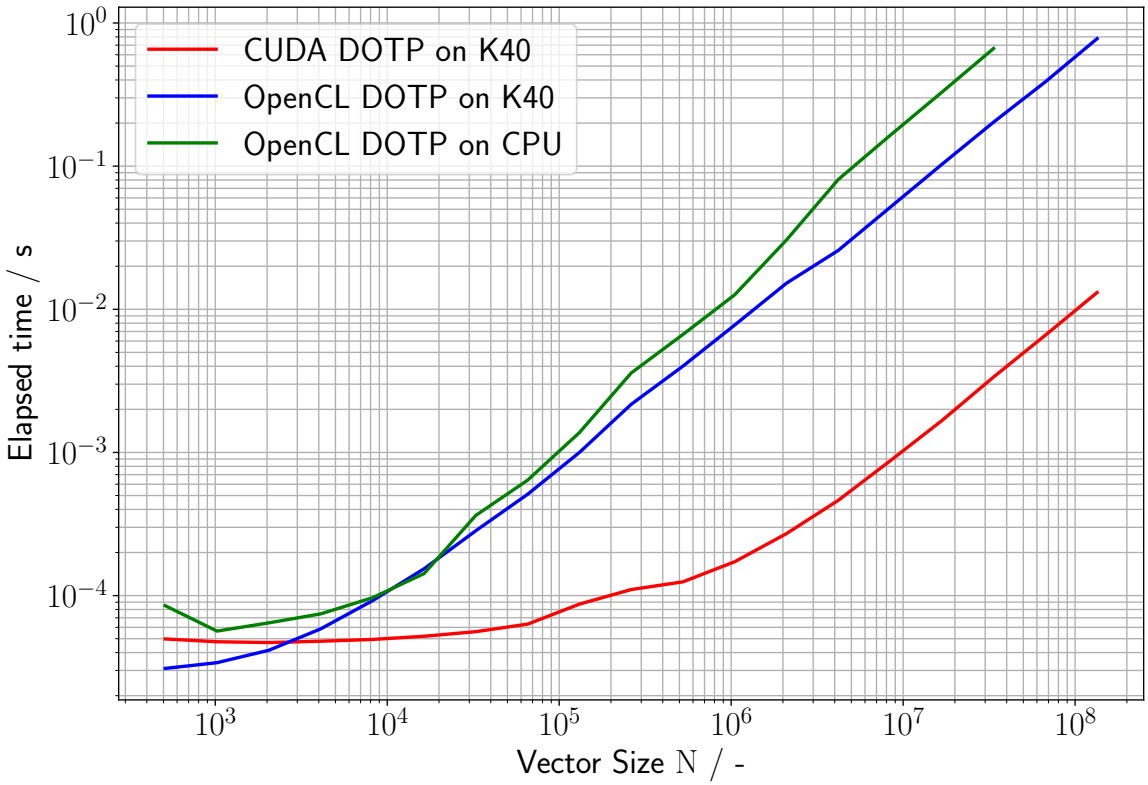

Figure 2: CPU GPU performance Comparison

1.4 Measure Time it takes to build M OpenCL Programs

not done :-(.

2 Bonus: Implement CUDA+OpenCL (CUCL) Approach (0/1 Points)

not done :-(.