GPU Programming
Hands-on Lab 3

Profiling sumMatrix.cu (follows Lecture 10)

Instructions **Marks: 10**

Due date: 12 noon on Tuesday 30th April 2024

Please submit the results tabulated in the provided worksheet.
By submitting your assignment via the RUConnected link, you declare that the assignment submitted is entirely your own work, unless noted otherwise.

Aim

Using the visual profiler (nvvp) to profile Cuda programs

Tasks

For this practical, you will use the same sumMatrix.cu code and thread block dimensions that produced the best and the worst timings for each of the three different execution exercises completed in Hands-on Lab 2 (you should have 6 different entries in the spreadsheet for this prac). You may use the solution code uploaded on RUConnected.

1. Rerun the application with each configuration entry and note the kernel execution time for each.
2. Now, profile each of these executions. To collect the metrics easily, you will need to select the appropriate metrics explicitly once *nvvp* has started. (Use the Magifier glass icon on the top of the screen to "Collect Metrics and Events" or find them in the information window (bottom extreme right side) for the relevant guided/unguided analysis.)
3. First metric to consider is the *achieved occupancy*: ratio of average active warps per cycle to the maximum number of warps on SM.
4. Next consider the memory operations: there are two global memory loads and one global memory store.
    a. First, check the memory read efficiency of the kernel by considering the *global load throughput*.
    b. Next, check the *global load efficiency*, that is, the ratio of requested global load throughput to required global load throughput, which measures how well the application's load operations use device memory bandwidth.
    c. Do the same for the *global store throughput* and *global store efficiency.*
5. Considering the 3 entries in the table for the best kernel execution times obtained, do they all represent optimal values, i.e., best achieved occupancy, best global load throughput, and best global memory load efficiency? Or is it the case that no single metric can predict improved performance?

GPU Programming
Hands-on Lab 3 worksheet

Name: **Marks: 10**

Block size x * y (Insert your own values)	Kernel execution time (ms)	Achieved occupancy (%)	Global load throughput (GB/s)	Global memory load efficiency (%)	Global store throughput (GB/s)	Global store efficiency (%)
x * y (orig 2D grid best)						
x * y (orig 2D grid worst)						
x * y (1D grid best)						
x * y (1D grid worst)						
x * y (16 elem best)						
x * y (16 elem worst)						

Discussion of task 5: