M2-BIG DATA
GPGPU - Chapter 12

Exercice 1

Réalisé par:
Gaby Maroun

Encadré par:
Dr. Etancelin JM

February 15, 2021

Objectives

The purpose of this lab is to get you familiar with using the CUDA streaming API by reimplementing a the vector addition lab to use CUDA streams.

Instructions

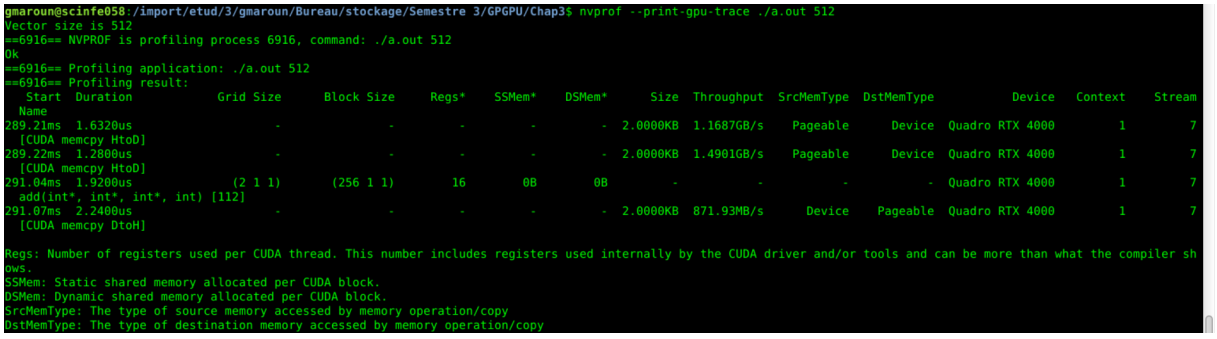
From the 1-vectorAdd code (chapter 3) adapt the code with the following modifications :

- Replace host calloc by pinned memory host allocation.
- Create arrays of pointer for device memory allocations and an array of cudaStream_t for streams. Use STREAM_NB=4 streams to begin.
- Create the streams using cudaStreamCreate function
- Allocate device memory for each buffer in each stream.
- Split the computations in a loop over STREAM_NB*STREAM_SIZE elements. Each sequence of TransferA, TransferB, add kernel and TransferC is processing STREAM_NB*STREAM_SIZE elements.

Indication : you should use two variable for the starting index and the length of the current bloc of elements.

Questions

1. **What is the identifier of the default stream when profiling the initial version of vectorAdd from previous lab ?**
   The identifier of the default stream from the old version of the code is, as I've understood, 7 while the ones for the current code are composed between 14,15,16 & 17.



2. **Compare the profiling informations from the Chapter 3 code and your current code, using Nvidia Visual profiler (nvvp) :**

   The profiler of the code from this exercise is,

```
gmaroun@scinfe058:/import/etud/3/gmaroun/Bureau/stockage/Semestre 3/GPGPU/Chap12/Exo1$ nvprof --print-gpu-trace ./a.out 512
Vector size is 512
==2937== NVPROF is profiling process 2937, command: ./a.out 512
Ok
==2937== Profiling application: ./a.out 512
==2937== Profiling result:
   Start  Duration           Grid Size      Block Size     Regs*    SSMem*    DSMem*      Size  Throughput  SrcMemType  DstMemType           Device   Context    Stream
  Name
301.68ms  1.6640us                    -               -         -         -         -  2.0000KB  1.1462GB/s      Pinned      Device  Quadro RTX 4000         1        14
  [CUDA memcpy HtoD]
301.69ms  2.8480us                    -               -         -         -         -  2.0000KB  685.79MB/s      Pinned      Device  Quadro RTX 4000         1        14
  [CUDA memcpy HtoD]
303.77ms  1.9200us              (2 1 1)       (256 1 1)        16        0B        0B         -           -           -           -  Quadro RTX 4000         1        14
  add(int*, int*, int*, int) [128]
303.77ms  2.0160us                    -               -         -         -         -  2.0000KB  968.81MB/s      Device      Pinned  Quadro RTX 4000         1        14
  [CUDA memcpy DtoH]
303.78ms  2.4320us                    -               -         -         -         -  2.0000KB  803.09MB/s      Pinned      Device  Quadro RTX 4000         1        15
  [CUDA memcpy HtoD]
303.79ms  2.4000us                    -               -         -         -         -  2.0000KB  813.80MB/s      Pinned      Device  Quadro RTX 4000         1        15
  [CUDA memcpy HtoD]
303.80ms  1.6320us              (2 1 1)       (256 1 1)        16        0B        0B         -           -           -           -  Quadro RTX 4000         1        15
  add(int*, int*, int*, int) [132]
303.81ms  2.2720us                    -               -         -         -         -  2.0000KB  859.65MB/s      Device    Pageable  Quadro RTX 4000         1        15
  [CUDA memcpy DtoH]
303.82ms  1.2800us                    -               -         -         -         -  2.0000KB  1.4901GB/s      Pinned      Device  Quadro RTX 4000         1        16
  [CUDA memcpy HtoD]
303.83ms  1.3440us                    -               -         -         -         -  2.0000KB  1.4192GB/s    Pageable      Device  Quadro RTX 4000         1        16
  [CUDA memcpy HtoD]
303.83ms  1.6320us              (2 1 1)       (256 1 1)        16        0B        0B         -           -           -           -  Quadro RTX 4000         1        16
  add(int*, int*, int*, int) [136]
303.84ms  2.0800us                    -               -         -         -         -  2.0000KB  939.00MB/s      Device    Pageable  Quadro RTX 4000         1        16
  [CUDA memcpy DtoH]
303.85ms  1.2800us                    -               -         -         -         -  2.0000KB  1.4901GB/s    Pageable      Device  Quadro RTX 4000         1        17
  [CUDA memcpy HtoD]
303.85ms  1.3760us                    -               -         -         -         -  2.0000KB  1.3862GB/s    Pageable      Device  Quadro RTX 4000         1        17
  [CUDA memcpy HtoD]
303.86ms  1.6000us              (2 1 1)       (256 1 1)        16        0B        0B         -           -           -           -  Quadro RTX 4000         1        17
  add(int*, int*, int*, int) [140]
303.86ms  1.8880us                    -               -         -         -         -  2.0000KB  1.0102GB/s      Device    Pageable  Quadro RTX 4000         1        17
  [CUDA memcpy DtoH]
```

- **What is the speedup of Host-Device transfer speed when using pinned memory.**

  The speedup of Host-Device transfer speed when using pinned memory is of 10,624µs. Noting that between the start (301,68ms) of the first host-device pinned memory transfer and the start of the last one(303,82ms) is of 2,14ms.

  If we go on and add all the rest(pageable) of the host-device transfer, the time consumed goes up to 14,624µs.

  Which in comparison to the old version of the code with 2,912µs, we can see that it's way more memory time consuming

- **Measure the entire execution time between start of the first copy to device and the end of the last copy from device.**

  The execution time between start of the first copy to device and the end of the last copy from device is equal to 29,664µs. If we exclude the time of the kernels execution, we'll have a time for memory transfer of 22,88µs.
  Compared to the previous code, which used to take up to 7,072µs, this code is more time consuming as they're treating a small calculation. But I believe, on a bigger problem, the old version will struggle compared to the new one.

La fin.