Assignment 1

Exercise 1 - Reflection on GPU-accelerated Computing

In the first lecture, we covered the main architectural differences between CPU and GPU. Please refresh or read further and answer the following questions:

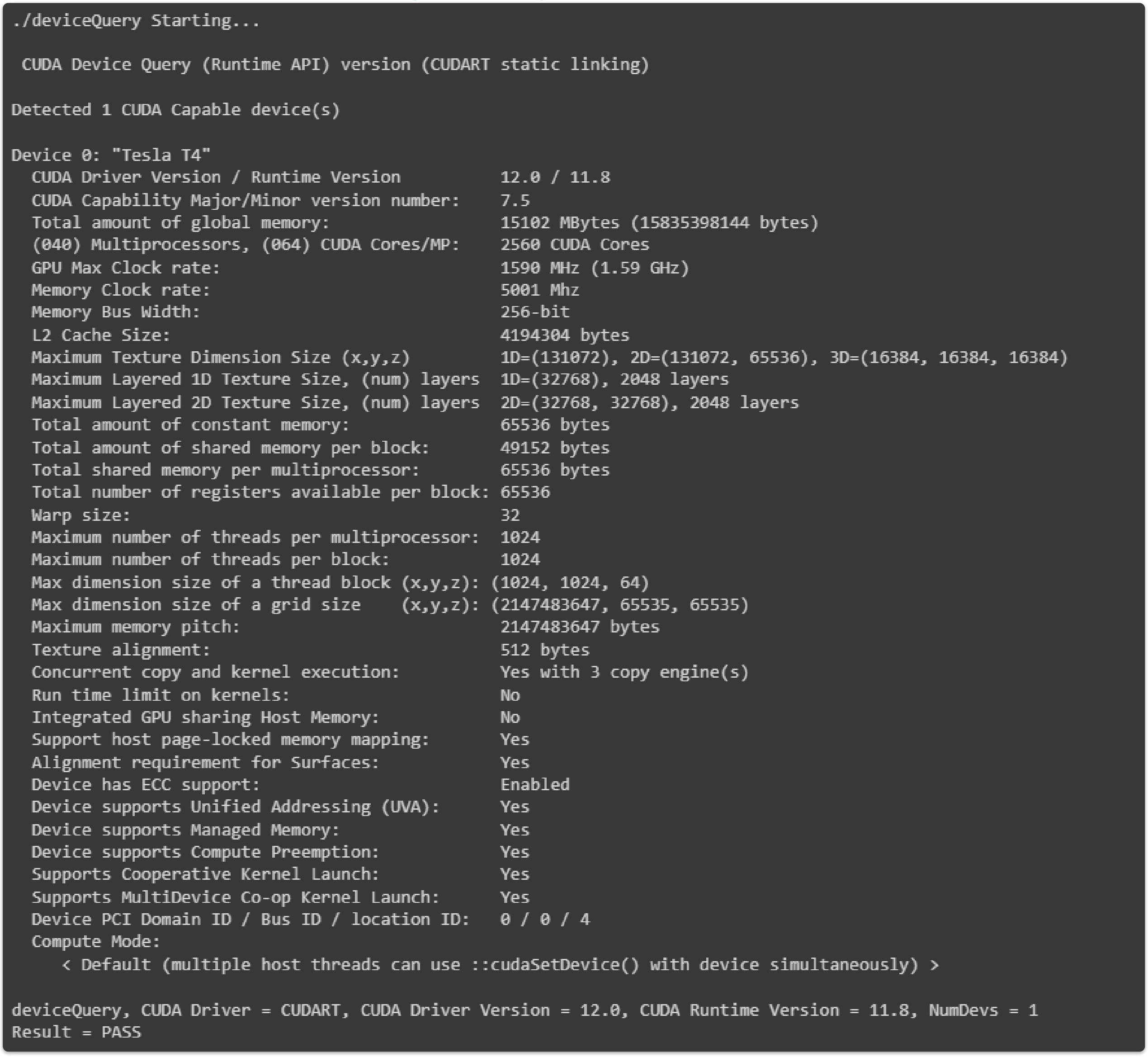
1. List the main differences between GPUs and CPUs in terms of architecture.
   - GPU: have a lot of ALU for parallel calculation
   - CPU: have more cores and excel at control works
2. Check the latest Top500 list that ranks the top 500 most powerful supercomputers in the world. In the top 10, how many supercomputers use GPUs? Report the name of the supercomputers and their GPU vendor (Nvidia, AMD, ...) and model.
   - 7 of which use GPUs

Rank	Name	GPU vendor	Model
1	Frontier	AMD	HPE Cray EX235a
3	LUMI	AMD	HPE Cray EX235a
4	Leonardo	Nvidia	BullSequana XH2000
5	Summit	Nvidia	IBM Power System AC922
6	Sierra	Nvidia	IBM Power System S922LC
8	Perlmutter	Nvidia	HPE Cray EX235n
9	Selene	Nvidia	NVIDIA DGX A100

3. One main advantage of GPU is its **power efficiency**, which can be quantified by *Performance/Power*, e.g., throughput as in FLOPS per watt power consumption. Calculate the power efficiency for the top 10 supercomputers. (Hint: use the table in the first lecture)
   Here, we use the formula $\frac{RmaxPFlop/s}{kW} = GFlops/watt$ to calculate the results with the data provide from the net "Top500"
   1. Frontier ($\frac{1{,}194.00}{22{,}703} = 52.592$)
   2. Supercomputer Fugaku ($\frac{442.01}{29{,}899} = 14.783$)
   3. LUMI ($\frac{309.10}{6{,}016} = 51.382$)
   4. Leonardo ($\frac{238.70}{7{,}404} = 32.239$)
   5. Summit ($\frac{148.60}{10{,}096} = 14.719$)
   6. Sierra ($\frac{94.64}{7{,}438} = 12.724$)
   7. Sunway TaihuLight ($\frac{93.01}{15{,}371} = 6.051$)
   8. Perlmutter ($\frac{70.87}{2{,}589} = 27.374$)
   9. Selene ($\frac{63.46}{2{,}646} = 23.983$)
   10. Tianhe-2A ($\frac{61.44}{18{,}482} = 3.324$)

Exercise 2 - Query Nvidia GPU Compute Capability

1. The screenshot of the output from running deviceQuery test in /1_Utilities.

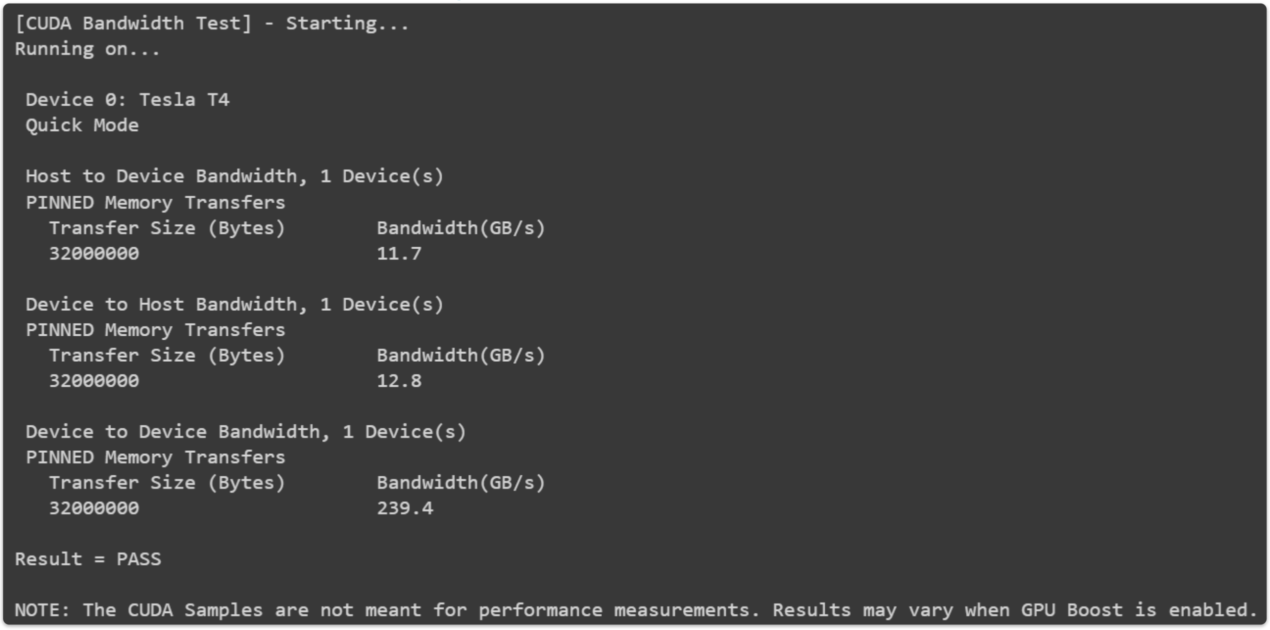


```
./deviceQuery Starting...

 CUDA Device Query (Runtime API) version (CUDART static linking)

Detected 1 CUDA Capable device(s)

Device 0: "Tesla T4"
  CUDA Driver Version / Runtime Version          12.0 / 11.8
  CUDA Capability Major/Minor version number:    7.5
  Total amount of global memory:                 15102 MBytes (15835398144 bytes)
  (040) Multiprocessors, (064) CUDA Cores/MP:    2560 CUDA Cores
  GPU Max Clock rate:                            1590 MHz (1.59 GHz)
  Memory Clock rate:                             5001 Mhz
  Memory Bus Width:                              256-bit
  L2 Cache Size:                                 4194304 bytes
  Maximum Texture Dimension Size (x,y,z)         1D=(131072), 2D=(131072, 65536), 3D=(16384, 16384, 16384)
  Maximum Layered 1D Texture Size, (num) layers  1D=(32768), 2048 layers
  Maximum Layered 2D Texture Size, (num) layers  2D=(32768, 32768), 2048 layers
  Total amount of constant memory:               65536 bytes
  Total amount of shared memory per block:       49152 bytes
  Total shared memory per multiprocessor:        65536 bytes
  Total number of registers available per block: 65536
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  1024
  Maximum number of threads per block:           1024
  Max dimension size of a thread block (x,y,z): (1024, 1024, 64)
  Max dimension size of a grid size    (x,y,z): (2147483647, 65535, 65535)
  Maximum memory pitch:                          2147483647 bytes
  Texture alignment:                             512 bytes
  Concurrent copy and kernel execution:          Yes with 3 copy engine(s)
  Run time limit on kernels:                     No
  Integrated GPU sharing Host Memory:            No
  Support host page-locked memory mapping:       Yes
  Alignment requirement for Surfaces:            Yes
  Device has ECC support:                        Enabled
  Device supports Unified Addressing (UVA):      Yes
  Device supports Managed Memory:                Yes
  Device supports Compute Preemption:            Yes
  Supports Cooperative Kernel Launch:            Yes
  Supports MultiDevice Co-op Kernel Launch:      Yes
  Device PCI Domain ID / Bus ID / location ID:   0 / 0 / 4
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simultaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 12.0, CUDA Runtime Version = 11.8, NumDevs = 1
Result = PASS
```

2. What is the Compute Capability of your GPU device?

   The GPU device I used here is Nvidia Tesla T4, and its compute capability is 7.5

3. The screenshot of the output from running bandwidthTest test in /1_Utilities.



```
[CUDA Bandwidth Test] - Starting...
Running on...

 Device 0: Tesla T4
 Quick Mode

 Host to Device Bandwidth, 1 Device(s)
 PINNED Memory Transfers
   Transfer Size (Bytes)        Bandwidth(GB/s)
   32000000                     11.7

 Device to Host Bandwidth, 1 Device(s)
 PINNED Memory Transfers
   Transfer Size (Bytes)        Bandwidth(GB/s)
   32000000                     12.8

 Device to Device Bandwidth, 1 Device(s)
 PINNED Memory Transfers
   Transfer Size (Bytes)        Bandwidth(GB/s)
   32000000                     239.4

Result = PASS

NOTE: The CUDA Samples are not meant for performance measurements. Results may vary when GPU Boost is enabled.
```

4. How will you calculate the GPU memory bandwidth (in GB/s) using the output from deviceQuery? (Hint: memory bandwidth is typically determined by clock rate and bus width, and check what double date rate (DDR) may impact the bandwidth). Are they consistent with your results from bandwidthTest?

Memory bandwidth is calculated as $\frac{clock.rate \times bus.width}{8} \times data.rate$
So for T4 which use GDDR6, the memory bandwidth is $\frac{5001 \times 256}{8} \times 2 = 320064 = 320$ GB/s
Compare to the results from bandwidthTest which is reported as 239.4 GB/s (device to device), it's is roughly 75% of the theoretical value calculated by us.

Exercise 3 - Rodinia CUDA benchmarks and Comparison with CPU

1. Compile both OMP and CUDA versions of your selected benchmarks. Do you need to make any changes in Makefile?
   I only need to make sure the path of cuda and sdk are correct in the make.config file. Since the information in Makefile for both benchmarks are already right, I don't need to change anything.
2. Ensure the same input problem is used for OMP and CUDA versions. Report and compare their execution time.
   Since the reported execution time of OMP is excluded the time of reading data to memory (i.e. only the time of execution for the benchmark is counted), here I also report the time of CUDA version followed this fashion.

benckmarks	OMP	CUDA
nn	0.039179 s	0.000006 s
bfs	0.284244 s	0.003540 s

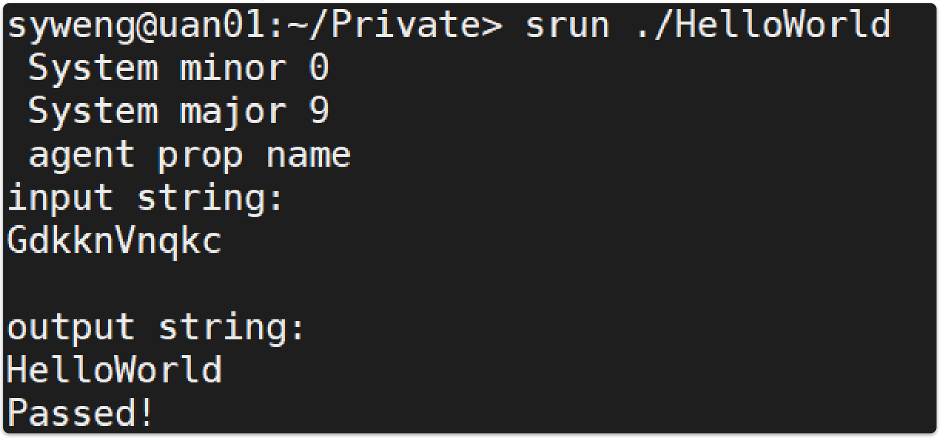
3. Do you observe expected speedup on GPU compared to CPU? Why or Why not?
   Yes, there are significant speedup for both benchmarks on GPU. I think it is because the input data is quiet large for both case. For instance, nn (which is k nearest neighbor) has to compute distance between two points, and the computation burden grows faster with the data points. But GPU can parallel computing which can mitigate the slow computing time.

Exercise 4 - Run a HelloWorld on AMD GPU

1. How do you launch the code on GPU on Dardel supercomputer?
   Firstly, I download the provided cpp and Makefile.txt, and then I delete the file extension of Makefile. Later, I use `make` to produce the execution file
   Below are the output messages showed after `make`

```
syweng@uan01:~/Private> make
hipcc --offload-arch=gfx90a   -c -o HelloWorld.o HelloWorld.cpp
hipcc HelloWorld.o -o HelloWorld
```

2. Include a screenshot of your output from Dardel



```
syweng@uan01:~/Private> srun ./HelloWorld
 System minor 0
 System major 9
 agent prop name
input string:
GdkknVnqkc

output string:
HelloWorld
Passed!
```

I finish this assignment by myself, since I haven't get in touch with my groupmate before the assignment deadline.