Applied GPU Programming - Assignment 1

DD2360 HT20

Pontus Asp

November 6, 2020

1 Exercise 1

1. GPUs emerged as suitable hardware for computer graphics because computer graphics needs to calculate a lot of pixels. Most of the time the pixels can be independently calculated from each other and therefore are also susceptible to parallelization. The nature of GPUs are that they work great with calculating highly parallelized single instructions on multiple data (SIMD), like pixels in an image and make a perfect fit for this application.

2. When talking about GPUs we talk about throughput-oriented architecture because that is the type of architecture that the GPUs use. It is thanks to this architecture that GPUs are highly efficient at handling SIMD operations.

3. The main differences in the architecture between CPUs and GPUs are that the GPUs have many more computational units than a CPU. The GPUs architecture is made to maximize the amount (thoughput) of calculations per unit of time when the calculations are independent and has the same instructions but on different data. While the architecture of the CPU is better at handling instructions with less parallelization.

4. During this course I plan to use my own GPU, the NVIDIA GeForce GTX 1070. It has 15 SMs and 128 CUDA Cores per SM. The clock frequency is 1860 MHz and it has a memory of 8 GB. The specific name of the model is ROG-STRIX-GTX1070-O8G-GAMING.

5. 1. The Tensor core supports doing one multiply-and-accumulate on a 4x4 dimensional matrix per clock cycle.

   2. The Tensor Cores supports both taking input and matrix multiplication calculations in half floating-point precision but the accumulation uses single floating-point precision. How precise these precisions are, is decided in the IEEE 754 standard.

   3. I think tensor cores were introduced in new GPUs mainly for the increasing demand of machine learning and their heavy use of matrix multiplication operations.

   4. The biggest difference between the two other than the structure is response-time. TPUs have a better response-time while GPUs (also GPUs with Tensor cores) have better throughput.

   5. I have not been able to find any information on what operation the NVIDIA RT Core supports in a single clock cycle. However what

I did find is that the NVIDIA RT Core accelerates calculating ray tracing. The RT Core has two specialized units, one which does bounding box tests and the other does ray-trinagle intersection tests.

6. Six out of the ten most powerful supercomputers use GPUs, all of them use NVIDIA GPUs. The supercomputers names along with the GPU model is specified below.

    1. Summit, NVIDIA Quadro GV100 Volta
    2. Sierra, NVIDIA Quadro GV100 Volta
    3. HPC5, NVIDIA Tesla V100
    4. Selene, NVIDIA A100
    5. Marconi-100, NVIDIA Tesla V100
    6. Piz Daint, NVIDIA Tesla P100

7. I found a paper related to machine learning.
   Title: SONG: Approximate Nearest Neighbor Search on GPU
   Authors: Weijie Zhao, Shulong Tan, Ping Li
   Converence: 2020 IEEE 36th International Conference on Data Engineering (ICDE)
   GPU used: NVIDIA TESLA V100
   Approach: They approached their problem with using CUDA to perform graph based searches on ANNs using GPUs to speed up the searches.

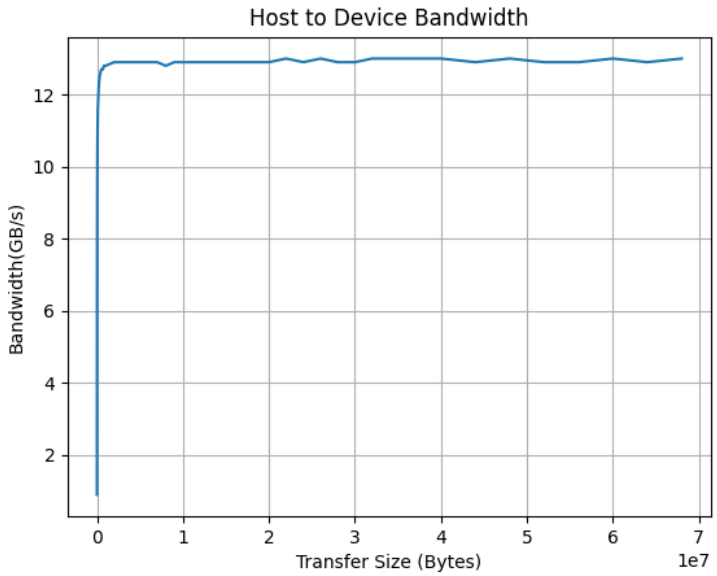

Figure 1: Host to Device Bandwidth.

2 Exercise 2

When running the bandwidthTest utility there was three (3) different tests, Host to Device Bandwidth (Figure 1), Device to Host Bandwidth (Figure 2) and Device to Device Bandwidth (Figure 3) test. My first observation was that the Device to Device test had a massively larger bandwidth than the other two tests. I realized this must be because the internal memory transfer rate is much higher than the transfer rate between the GPU and the rest of the system. My second observation was that the bandwidth in the transfer between the GPU and system seemed to have a roof at approximately 13 GB/s that it reached fast. I think it does not have the same speed when transferring very small amounts of bytes because of the latency in the transfer. Since the less bytes you transfer the bigger impact the latency has, which would explain the shape of the curve and why it flattens out so quickly. In Figure 3 there is a big drop in the bandwidth after 10^6 in the X-axis. I am not sure why this happens but I would assume it has something to do with that since the transfer is between two locations in the GPU memory, the memory simply runs out while transferring when doing too big transfers so that it can not transfer with the same efficiency as before.

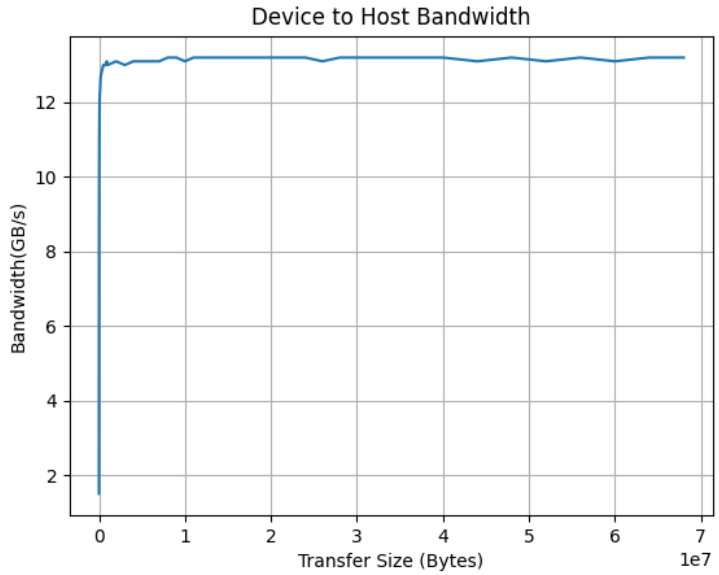

Figure 2: Device to Host Bandwidth.

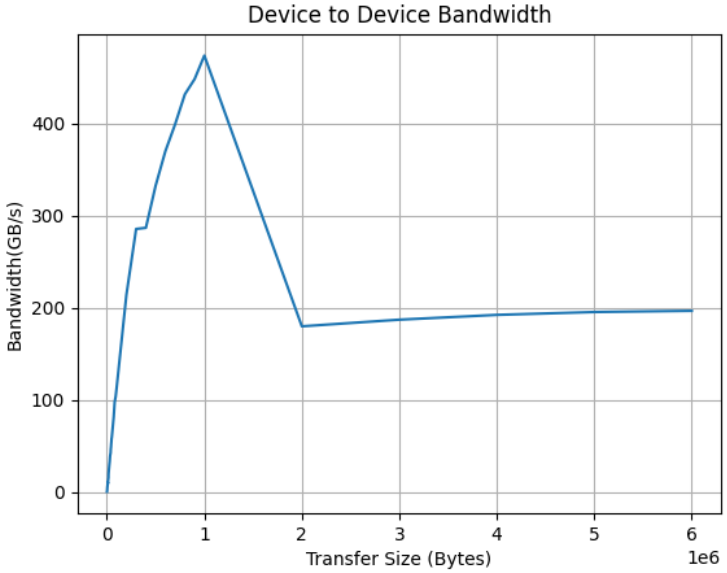

Figure 3: Device to Device Bandwidth.