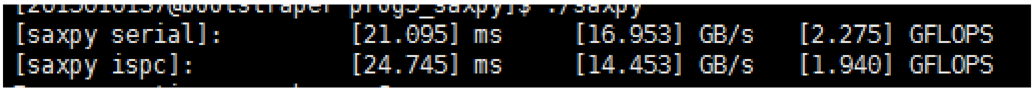
Assignment 2

2015010137 Lu Cheng

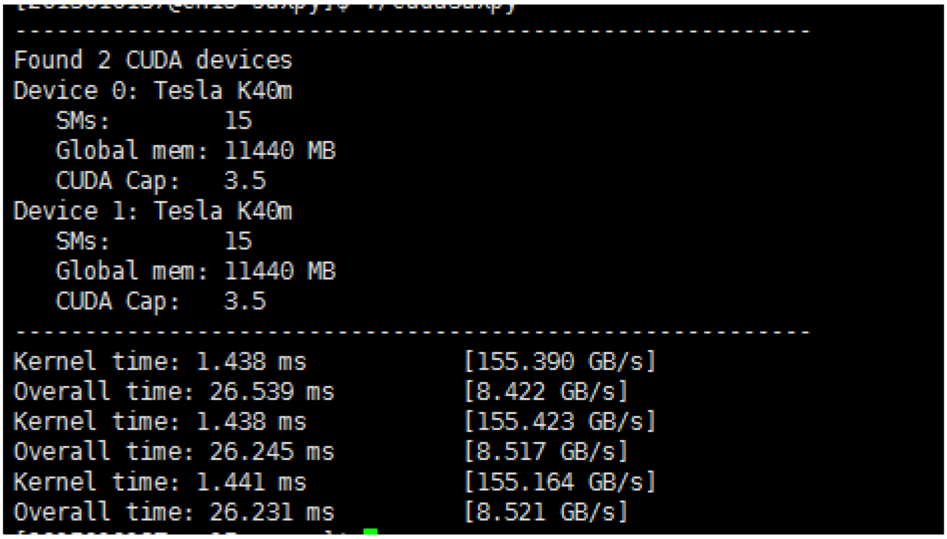
Part 1:

sequential:



```
[saxpy serial]:         [21.095] ms     [16.953] GB/s   [2.275] GFLOPS
[saxpy ispc]:           [24.745] ms     [14.453] GB/s   [1.940] GFLOPS
```

Cuda:



```
---------------------------------------------------------
Found 2 CUDA devices
Device 0: Tesla K40m
   SMs:        15
   Global mem: 11440 MB
   CUDA Cap:   3.5
Device 1: Tesla K40m
   SMs:        15
   Global mem: 11440 MB
   CUDA Cap:   3.5
---------------------------------------------------------
Kernel time: 1.438 ms           [155.390 GB/s]
Overall time: 26.539 ms         [8.422 GB/s]
Kernel time: 1.438 ms           [155.423 GB/s]
Overall time: 26.245 ms         [8.517 GB/s]
Kernel time: 1.441 ms           [155.164 GB/s]
Overall time: 26.231 ms         [8.521 GB/s]
```

The bandwidth of PCIe 3.0 -x16 bus is 15.8GB/s and the memory bandwidth of GTX1080 is 320GB/s.

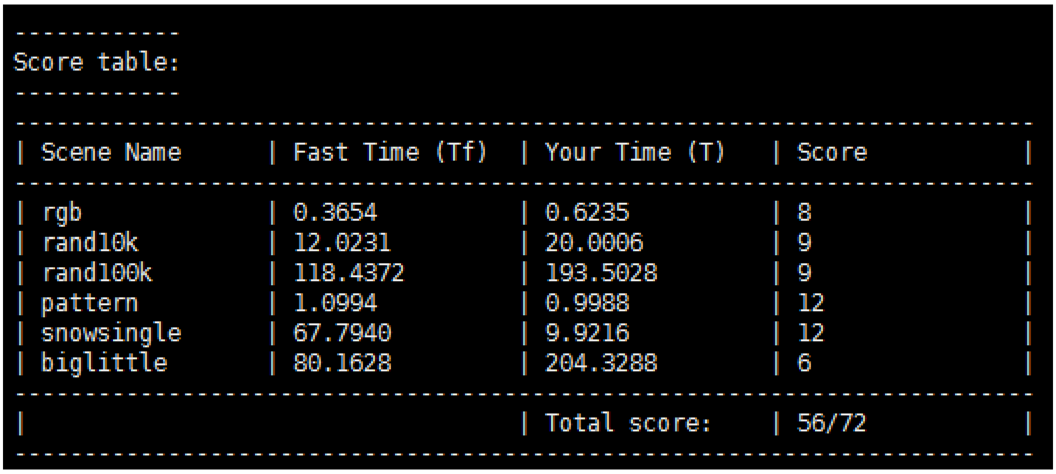
As we can see, the bandwidth of the overall time is nearly half of that of PCIe, and the bandwidth of the kernel time is nearly half of that of the memory of GPU.

The memory in GPU can read/write fast, but data through GPU and CPU is much slower because of the bandwidth limitation.

Part 3:

1. I finished this work with the help of TA Yong.

2.



```
------------
Score table:
------------
--------------------------------------------------------------------------
| Scene Name      | Fast Time (Tf)  | Your Time (T)   | Score            |
--------------------------------------------------------------------------
| rgb             | 0.3654          | 0.6235          | 8                |
| rand10k         | 12.0231         | 20.0006         | 9                |
| rand100k        | 118.4372        | 193.5028        | 9                |
| pattern         | 1.0994          | 0.9988          | 12               |
| snowsingle      | 67.7940         | 9.9216          | 12               |
| biglittle       | 80.1628         | 204.3288        | 6                |
--------------------------------------------------------------------------
|                                   | Total score:    | 56/72            |
--------------------------------------------------------------------------
```

3. The solution is:

    a) Divide the picture into 32*32 cells. Every cell is 32*32.

    b) Parallel for each circle. Compute the number of cells that is intersected with the bonding box of each circle, than compute the total number. (total number of all pairs.) The sum is competed in parallel, using shared memory and atomicAdd.

    c) Malloc the memory of all pairs, using the total number.

    d) Parallel for each circle to get all the pairs.

    e) Sort the pairs in parallel with the cell-index.

    f) Compute the pair length of each cell in parallel (According to Part 2).

    g) Compute each pixel of each cell in parallel.

4. The (b), (d), (e), (f), (g) is synchronization.

5. Using shared memory in (b); reduce array number with (b) and (c); using precise number of each circle’s pair (computed by (b)) to avoid list-appending lock in (d);

6. I tried to use bitmap and scan, but it takes long time since every cell need to scan.