EE217: GPU Architecture and Parallel Programming
Lab#4

Akber Raza
araza008@ucr.edu

March 6, 2019

```
==14843== Profiling application: ./vector_add
==14843== Profiling result:
   Start  Duration            Grid Size      Block Size     Regs*    SSMem*    DSMem*      Size  Throughput  SrcMemType  DstMemType          Device   Context    Stream  Name
426.65ms  729.38us                    -               -         -         -         -  1.2716MB  1.7025GB/s      Pinned      Device    Tesla M60 (0)         1        17  [CUDA memcpy HtoD]
427.39ms  745.31us                    -               -         -         -         -  1.2716MB  1.6661GB/s      Pinned      Device    Tesla M60 (0)         1        17  [CUDA memcpy HtoD]
428.15ms  703.43us                    -               -         -         -         -  1.2716MB  1.7653GB/s      Pinned      Device    Tesla M60 (0)         1        19  [CUDA memcpy HtoD]
428.15ms  30.048us           (652 1 1)       (512 1 1)        8        0B        0B         -           -           -           -    Tesla M60 (0)         1        17  VecAdd(int, float const *, float c
onst *, float*) [435]
428.23ms  666.63us                    -               -         -         -         -  1.2716MB  1.8628GB/s      Device      Pinned    Tesla M60 (0)         1        17  [CUDA memcpy DtoH]
428.92ms  798.88us                    -               -         -         -         -  1.2716MB  1.5544GB/s      Pinned      Device    Tesla M60 (0)         1        19  [CUDA memcpy HtoD]
429.72ms  662.24us                    -               -         -         -         -  1.2716MB  1.8751GB/s      Pinned      Device    Tesla M60 (0)         1        18  [CUDA memcpy HtoD]
429.73ms  28.832us           (652 1 1)       (512 1 1)        8        0B        0B         -           -           -           -    Tesla M60 (0)         1        19  VecAdd(int, float const *, float c
onst *, float*) [437]
429.79ms  625.51us                    -               -         -         -         -  1.2716MB  1.9852GB/s      Device      Pinned    Tesla M60 (0)         1        19  [CUDA memcpy DtoH]
430.43ms  503.23us                    -               -         -         -         -  1.2716MB  2.4676GB/s      Pinned      Device    Tesla M60 (0)         1        18  [CUDA memcpy HtoD]
430.94ms  28.608us           (652 1 1)       (512 1 1)        8        0B        0B         -           -           -           -    Tesla M60 (0)         1        18  VecAdd(int, float const *, float c
onst *, float*) [436]
430.97ms  507.65us                    -               -         -         -         -  1.2716MB  2.4461GB/s      Device      Pinned    Tesla M60 (0)         1        18  [CUDA memcpy DtoH]
```

Figure 1: Stream Visualization

```
==18096== Profiling application: ./vector_add
==18096== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   63.83%  1.2020ms         6  200.33us  186.63us  236.96us  [CUDA memcpy HtoD]
                   31.33%  589.96us         3  196.65us  189.31us  203.94us  [CUDA memcpy DtoH]
                    4.85%  91.264us         3  30.421us  28.480us  33.152us  VecAdd(int, float const *, float const *, float*)
      API calls:   93.18%  169.95ms         3  56.651ms  2.3098ms  165.22ms  cudaHostAlloc
                    1.96%  3.5684ms         9  396.49us  259.02us  1.4781ms  cudaFree
                    1.78%  3.2517ms         3  1.0839ms  1.0191ms  1.1941ms  cudaFreeHost
                    1.47%  2.6890ms       384  7.0020us     165ns  325.00us  cuDeviceGetAttribute
                    1.01%  1.8455ms         9  205.05us  188.64us  270.37us  cudaMalloc
                    0.31%  565.10us         4  141.28us  137.66us  143.31us  cuDeviceTotalMem
                    0.09%  171.29us         4  42.822us  38.865us  53.427us  cuDeviceGetName
                    0.07%  127.98us         3  42.659us  11.342us  101.22us  cudaLaunchKernel
                    0.05%  96.960us         9  10.773us  4.3140us  38.647us  cudaMemcpyAsync
                    0.03%  63.554us         3  21.184us  14.792us  32.024us  cudaStreamCreateWithFlags
                    0.02%  35.746us         4  8.9360us  2.6260us  25.206us  cudaDeviceSynchronize
                    0.01%  11.174us         4  2.7930us  1.6570us  4.5930us  cuDeviceGetPCIBusId
                    0.00%  2.3580us         8     294ns     186ns     826ns  cuDeviceGet
                    0.00%  1.5780us         3     526ns     236ns     899ns  cuDeviceGetCount
                    0.00%     997ns         4     249ns     215ns     308ns  cuDeviceGetUuid
```

Figure 2: Percentage breakdown

Questions

What is the execution time of your vector add using no streams and 3 streams?

4.947 ms (without streams)
1.884 ms (with streams)

Were you able to observe full overlapping of computation and memory transfer? (Please use profiler output to justify your answer). Why or why not?

No, because kernel evaluation time ($30\mu s$) was much smaller than HtoD data transfer time ($700\mu s$)

How does your profiler timeline look now? Are you able to observe full overlapping of computation and memory transfer?

```
==27939== Profiling application: ./vector_add 1000000
==27939== Profiling result:
   Start  Duration            Grid Size      Block Size     Regs*    SSMem*    DSMem*      Size  Throughput  SrcMemType  DstMemType          Device   Context    Stream  Name
412.78ms  233.47us                    -               -         -         -         -  1.2716MB  5.3187GB/s      Pinned      Device    Tesla M60 (0)         1        17  [CUDA memcpy HtoD]
413.02ms  259.59us                    -               -         -         -         -  1.2716MB  4.7836GB/s      Pinned      Device    Tesla M60 (0)         1        17  [CUDA memcpy HtoD]
413.29ms  233.35us                    -               -         -         -         -  1.2716MB  5.3216GB/s      Pinned      Device    Tesla M60 (0)         1        19  [CUDA memcpy HtoD]
413.31ms  582.47us           (652 1 1)       (512 1 1)         8        0B        0B         -           -           -           -    Tesla M60 (0)         1        17  VecAdd(int, float const *, float c
onst *, float*) [435]
413.53ms  214.08us                    -               -         -         -         -  1.2716MB  5.8004GB/s      Pinned      Device    Tesla M60 (0)         1        19  [CUDA memcpy HtoD]
413.76ms  311.84us                    -               -         -         -         -  1.2716MB  3.9820GB/s      Pinned      Device    Tesla M60 (0)         1        18  [CUDA memcpy HtoD]
413.82ms  627.88us           (652 1 1)       (512 1 1)         8        0B        0B         -           -           -           -    Tesla M60 (0)         1        19  VecAdd(int, float const *, float c
onst *, float*) [437]
413.91ms  482.02us                    -               -         -         -         -  1.2716MB  2.5762GB/s      Device      Pinned    Tesla M60 (0)         1        17  [CUDA memcpy DtoH]
414.12ms  525.96us                    -               -         -         -         -  1.2716MB  2.3610GB/s      Pinned      Device    Tesla M60 (0)         1        18  [CUDA memcpy HtoD]
414.46ms  362.21us                    -               -         -         -         -  1.2716MB  3.4283GB/s      Device      Pinned    Tesla M60 (0)         1        19  [CUDA memcpy DtoH]
414.71ms  583.40us           (652 1 1)       (512 1 1)         8        0B        0B         -           -           -           -    Tesla M60 (0)         1        18  VecAdd(int, float const *, float c
onst *, float*) [436]
415.30ms  337.44us                    -               -         -         -         -  1.2716MB  3.6799GB/s      Device      Pinned    Tesla M60 (0)         1        18  [CUDA memcpy DtoH]

Regs: Number of registers used per CUDA thread. This number includes registers used internally by the CUDA driver and/or tools and can be more than what the compiler shows.
SSMem: Static shared memory allocated per CUDA block.
DSMem: Dynamic shared memory allocated per CUDA block.
SrcMemType: The type of source memory accessed by memory operation/copy
DstMemType: The type of destination memory accessed by memory operation/copy
```

Figure 3: Stream Visualization

As shown in the figure above, kernel computation takes $580\mu s$ and two HtoD transfers take $450\mu s$. There is almost a full overlapping of computation and memory transfers.

Would a Vector Add implementation with 4 streams help improve performance?

Adding a 4th stream will not improve performance because at any given time instant, not more than 3 streams can be operational. One stream performs HtoD transfers, another evaluates the kernel while the third stream does DtoH transfers.