**Name: SHANMUKHA SAI RAYAPATI**

**CSU ID: 2788976**
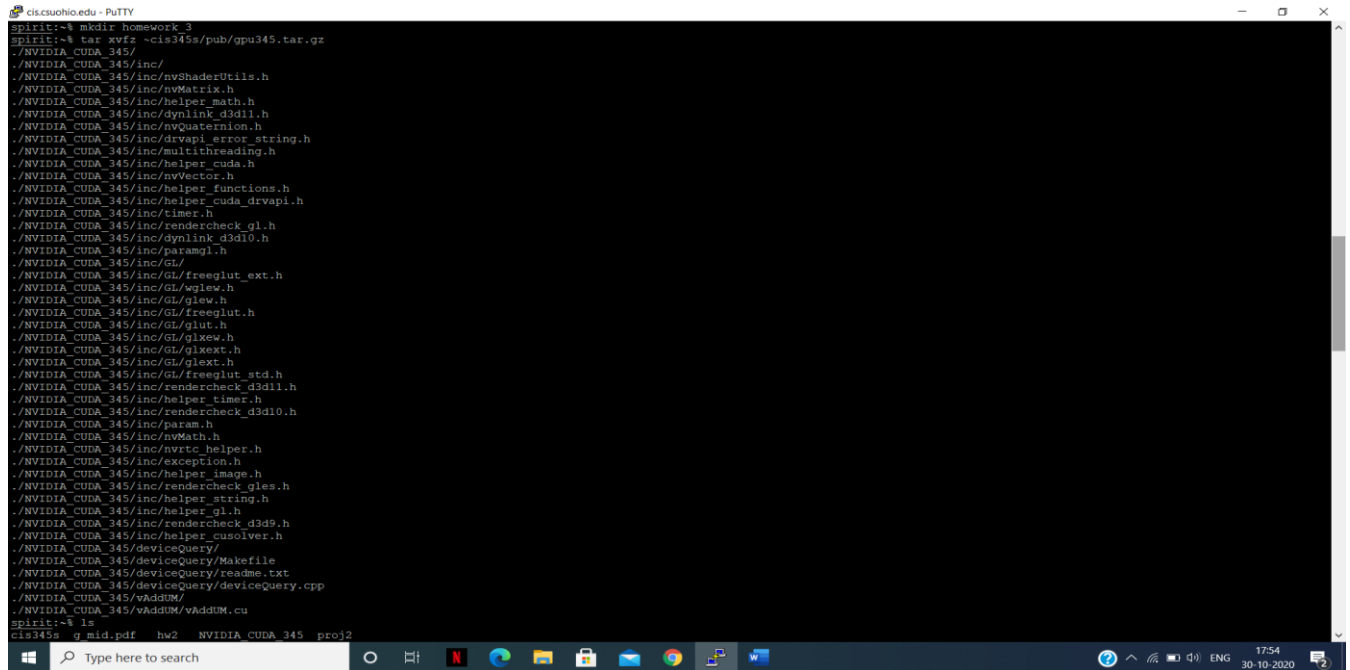


**Name: Anil Pavuluru**
**CSU ID:2782551**

**LOGIN ID: anpavulu**

## Task:1

Step-1: I make a directory called hw3 and under this directory 1 have extracted the NVIDIA_CUDA_345 using tar xvfz ~cis345s/pub/gpu345.tar.gz



-For the first part we navigate to ~/hw3/NVIDIA_CUDA_345/deviceQuery% and execute the make command to compile and build the kernel

```
./NVIDIA_CUDA_345/inc/helper_gl.h
./NVIDIA_CUDA_345/inc/rendercheck_d3d9.h
./NVIDIA_CUDA_345/inc/helper_cusolver.h
./NVIDIA_CUDA_345/deviceQuery/
./NVIDIA_CUDA_345/deviceQuery/Makefile
./NVIDIA_CUDA_345/deviceQuery/readme.txt
./NVIDIA_CUDA_345/deviceQuery/deviceQuery.cpp
./NVIDIA_CUDA_345/vAddUM/
./NVIDIA_CUDA_345/vAddUM/vAddUM.cu
spirit:~/homework_3% ls
NVIDIA_CUDA_345
spirit:~/homework_3% cd NVIDIA_CUDA_345
spirit:~/homework_3/NVIDIA_CUDA_345% ls
deviceQuery  inc  vAddUM
spirit:~/homework_3/NVIDIA_CUDA_345% cd deviceQuery
spirit:~/homework_3/NVIDIA_CUDA_345/deviceQuery% ls
deviceQuery.cpp  Makefile  readme.txt
spirit:~/homework_3/NVIDIA_CUDA_345/deviceQuery% make
"/usr/local/cuda"/bin/nvcc -ccbin g++ -I../../inc  -m64      -gencode arch=compute_30
,code=sm_30 -gencode arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm
_37 -gencode arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52 -gen
code arch=compute_60,code=sm_60 -gencode arch=compute_61,code=sm_61 -gencode arc
h=compute_70,code=sm_70 -gencode arch=compute_75,code=sm_75 -gencode arch=comput
e_75,code=compute_75 -o deviceQuery.o -c deviceQuery.cpp
"/usr/local/cuda"/bin/nvcc -ccbin g++   -m64      -gencode arch=compute_30,code=
sm_30 -gencode arch=compute_35,code=sm_35 -gencode arch=compute_37,code=sm_37 -g
encode arch=compute_50,code=sm_50 -gencode arch=compute_52,code=sm_52 -gencode a
rch=compute_60,code=sm_60 -gencode arch=compute_61,code=sm_61 -gencode arch=comp
ute_70,code=sm_70 -gencode arch=compute_75,code=sm_75 -gencode arch=compute_75,c
ode=compute_75 -o deviceQuery deviceQuery.o
spirit:~/homework_3/NVIDIA_CUDA_345/deviceQuery% ./deviceQuery
./deviceQuery Starting...

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

Detected 1 CUDA Capable device(s)

Device 0: "Quadro P620"
  CUDA Driver Version / Runtime Version          10.0 / 10.0
  CUDA Capability Major/Minor version number:    6.1
  Total amount of global memory:                 1992 MBytes (2089091072 bytes)
  ( 4) Multiprocessors, (128) CUDA Cores/MP:     512 CUDA Cores
  GPU Max Clock rate:                            1354 MHz (1.35 GHz)
  Memory Clock rate:                             2505 Mhz
  Memory Bus Width:                              128-bit
  L2 Cache Size:                                 524288 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
```

-After compiling we are running it using

./deviceQuery



```
ode=compute_75 -o deviceQuery deviceQuery.o
spirit:~/homework_3/NVIDIA_CUDA_345/deviceQuery% ./deviceQuery
./deviceQuery Starting...

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

Detected 1 CUDA Capable device(s)

Device 0: "Quadro P620"
  CUDA Driver Version / Runtime Version          10.0 / 10.0
  CUDA Capability Major/Minor version number:    6.1
  Total amount of global memory:                 1992 MBytes (2089091072 bytes)
  ( 4) Multiprocessors, (128) CUDA Cores/MP:     512 CUDA Cores
  GPU Max Clock rate:                            1354 MHz (1.35 GHz)
  Memory Clock rate:                             2505 Mhz
  Memory Bus Width:                              128-bit
  L2 Cache Size:                                 524288 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 number of registers available per block: 65536
  Warp size:                                     32
  Maximum number of threads per multiprocessor:  2048
  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 2 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:                        Disabled
  Device supports Unified Addressing (UVA):      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 / 1 / 0
  Compute Mode:
     < Default (multiple host threads can use ::cudaSetDevice() with device simu
ltaneously) >

deviceQuery, CUDA Driver = CUDART, CUDA Driver Version = 10.0, CUDA Runtime Vers
ion = 10.0, NumDevs = 1
Result = PASS
```

Streaming Multiprocessor (SM):4

CUDA core [(4) Multiprocessors, (128) CUDA Cores/MP]:    512 CUDA Cores

(Off chip) Global memory: 1992 MBytes (2089091072 bytes)

GPU clock rate: 1354 MHz (1.35 GHz)

TASK-2(page default):

In this task i reach the program vAddUM.cu by using following command

~/hw3/NVIDIA_CUDA_345/vAddUM

Next compile by using nvcc vAddUM.cu -o vAddUM

Next I run by using nvcc vAddUM.cu -o vAddUM then again I run by using nvprof ./vAddUM and taken screenshot. Because,it is in c++ that's why double cc



| Count | Avg Size | Min Size | Max Size | Total Size | Total Time | Name |
|-------|----------|----------|----------|------------|------------|------|
| 60 | 136.53KB | 4.0000KB | 0.9922MB | 8.000000MB | 1.169728ms | Host To |

| | | | | | | Device |
|---|---|---|---|---|---|---|
| 24 | 170.67KB | 4.0000KB | 0.9961MB | 4.000000MB | 343.6800us | Device To Host |
| 25 | - | - | - | - | 4.097280ms | Gpu page fault groups |

Total CPU Page faults: 36

Huge array A and Array b and add them store in array c (a[0]+b[0]=c[0] like that)

Int N=1<<20; $2^{20}$=1million array size

Then after allocating 1million through cudaMallocManaged by float 4bytes,4bytes,upto 1million

Then after simple logic written for the array

Timing purpose cude event start and stop

Int blocksize=256;

12*256

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|

Block 0      ---      --           -- --                     Block11

Int blocknum=12;

Vector add

The result c[0] array stores in block 0,thread 0,c[1] block 0,thread1

We come to know the absolute value of the index x is for the starting index block and then dimension of x is 256

Block 0

| |
|---|

  0,1.2,-------255

Block 1                                                      stride

| 0,1,2,-----255 |
256,257,
    ↑
    Index

-----

 Block 11

Page migration is caused that's why page fault occurs

Task-3(vAddUM_Init.cu):

In this task we copy the vAddUM and named That file in the name of vAddUM_Init.cu and changes are done based on handout. Removing the initialization of x and y we initialized values so that we reduce the page faults

Next step we run the program nvcc vAddUM_Init.cu  -o vAddUM_Init.cu and execute the output nvprof ./vAddUM_Init and taken screen shot from bach.

I move initialization from the CPU to the GPU by __global__, the addkernel launch by init<<<num Blocks, blockSize >>>won't page fault. Here's a simple CUDA C++ kernel to initialize the data. We can just replace the host code(that's why not getting answers for host to device) that initializes **x** and **y** with a launch of this kernel. So we can reduce the page fault.

```
% nvcc vAddUM_Init.cu -o vAddUM_Init
% nvprof ./vAddUM_Init
==19899== NVPROF is profiling process 19899, command: ./vAddUM_Init
Kernel time: 1.374080 ms
Max error: 0
==19899== Profiling application: ./vAddUM_Init
==19899== Profiling result:
            Type  Time(%)      Time     Calls       Avg       Min       Max  Name
 GPU activities:   61.19%  2.1613ms         1  2.1613ms  2.1613ms  2.1613ms  init(int, float*, float*)
                   38.81%  1.3710ms         1  1.3710ms  1.3710ms  1.3710ms  vecAdd(int, float*, float*, float*)
      API calls:   97.99%  274.53ms         3  91.509ms  40.825us  274.38ms  cudaMallocManaged
                    1.23%  3.4574ms         1  3.4574ms  3.4574ms  3.4574ms  cudaEventSynchronize
                    0.34%  944.62us         3  314.87us  179.27us  537.72us  cudaFree
                    0.25%  696.08us        96  7.2500us     748ns  302.75us  cuDeviceGetAttribute
                    0.12%  332.07us         2  166.03us  41.030us  291.04us  cudaLaunchKernel
                    0.02%  64.799us         1  64.799us  64.799us  64.799us  cuDeviceTotalMem
                    0.02%  55.361us         1  55.361us  55.361us  55.361us  cuDeviceGetName
                    0.01%  32.398us         2  16.199us  11.070us  21.328us  cudaEventRecord
                    0.01%  17.390us         2  8.6950us  6.3420us  11.048us  cudaEventCreate
                    0.00%  9.5230us         1  9.5230us  9.5230us  9.5230us  cudaEventElapsedTime
                    0.00%  4.8560us         3  1.6180us     741ns  2.4280us  cuDeviceGetCount
                    0.00%  2.6440us         1  2.6440us  2.6440us  2.6440us  cuDeviceGetPCIBusId
                    0.00%  2.0750us         2  1.0370us     857ns  1.2180us  cuDeviceGet
                    0.00%     849ns         1     849ns     849ns     849ns  cuDeviceGetUuid

==19899== Unified Memory profiling result:
Device "Quadro P620 (0)"
   Count  Avg Size  Min Size  Max Size  Total Size  Total Time  Name
      24  170.67KB  4.0000KB  0.9961MB  4.000000MB  344.4160us  Device To Host
      34         -         -         -           -  3.287744ms  Gpu page fault groups
Total CPU Page faults: 12
%
```

| Count | Avg Size | Min Size | Max Size | Total Size | Total Time | Name |
|---|---|---|---|---|---|---|
| 24 | 170.67KB | 4.0000KB | 0.9961MB | 4.000000MB | 344.4160us | Device To Host |
| 34 | - | - | - | - | 3.287744Ms | Gpu page fault groups |

Total CPU Page faults: 12

Task-4 (vAddUM_Pfetch.cu):

Copy vAddUM.cu to another file vAddUM_Pfetch.cu and change the code add the code before the kernel launch or before the z variable use the Pfetch so that page migration is reduced.

Next compile for the command using nvcc vAddUM_Pfetch.cu -o vAddUM_Pfetch and then run by using nvprof vAddUM_Pfetch after that taken the screenshot of the result

```
% nvcc vAddUM_Pfetch.cu -o VAddUM_Pfetch
% nvprof ./vAddUM_Pfetch
==17899== NVPROF is profiling process 17899, command: ./vAddUM_Pfetch
Kernel time: 4.147488 ms
Max error: 0
==17899== Profiling application: ./vAddUM_Pfetch
==17899== Profiling result:
            Type  Time(%)      Time    Calls       Avg       Min       Max  Nam
e
 GPU activities:  100.00%  1.4592ms        1  1.4592ms  1.4592ms  1.4592ms  vec
Add(int, float*, float*, float*)
      API calls:   96.15%  163.20ms        3  54.400ms  32.361us  163.10ms  cud
aMallocManaged
                    1.44%  2.4369ms        3  812.29us  244.06us  1.3803ms  cud
aMemPrefetchAsync
                    1.39%  2.3638ms        1  2.3638ms  2.3638ms  2.3638ms  cud
aEventSynchronize
                    0.43%  721.88us       96  7.5190us     696ns  325.81us  cuD
eviceGetAttribute
                    0.40%  677.45us        3  225.82us  186.77us  260.43us  cud
aFree
                    0.08%  133.74us        1  133.74us  133.74us  133.74us  cud
aLaunchKernel
                    0.04%  63.065us        1  63.065us  63.065us  63.065us  cuD
eviceGetName
                    0.04%  61.606us        1  61.606us  61.606us  61.606us  cuD
eviceTotalMem
                    0.02%  34.442us        2  17.221us  13.326us  21.116us  cud
aEventRecord
                    0.01%  21.526us        2  10.763us  5.1440us  16.382us  cud
aEventCreate
                    0.00%  5.1730us        1  5.1730us  5.1730us  5.1730us  cud
aEventElapsedTime
                    0.00%  4.7180us        1  4.7180us  4.7180us  4.7180us  cud
aGetDevice
                    0.00%  3.4500us        3  1.1500us     696ns  1.9500us  cuD
eviceGetCount
                    0.00%  2.5440us        1  2.5440us  2.5440us  2.5440us  cuD
eviceGetPCIBusId
                    0.00%  1.8390us        2     919ns     700ns  1.1390us  cuD
eviceGet
                    0.00%     822ns        1     822ns     822ns     822ns  cuD
eviceGetUuid

==17899== Unified Memory profiling result:
Device "Quadro P620 (0)"
```

```
==17899== Unified Memory profiling result:
Device "Quadro P620 (0)"
   Count  Avg Size  Min Size  Max Size  Total Size  Total Time  Name
       4  2.0000MB  2.0000MB  2.0000MB  8.000000MB  967.5200us  Host To Device
       2  2.0000MB  2.0000MB  2.0000MB  4.000000MB  338.3360us  Device To Host
      12         -         -         -           -  1.252768ms  Gpu page fault g
roups
Total CPU Page faults: 24
%
```

| Count | Avg Size | Min Size | Max Size | Total Size | Total Time | Name |
|---|---|---|---|---|---|---|
| 4 | 2.0000MB | 2.0000MB | 2.0000MB | 8.000000MB | 967.5200us | Host To Device |
| 2 | 2.0000MB | 2.0000MB | 2.0000MB | 4.000000MB | 338.3360us | Device To Host |
| 12 | - | - | - | - | 1.252768ms | Gpu page fault groups |

Total CPU Page faults: 24

Task-5(vAddUM_Over.cu)

Copy the code form the vAddUM to vAddUM_Over.cu and modfity the number form $2^{20}$ to $2^{28}$ and calculate the value $2^{28}*4=2^{30}$bytes

Then compile  nvcc vAddUM_Over.cu -o vAddUM_Over and run by using the command nvprof vAddUM_Over.cu and taken the screen shot of the results.

| Count | Avg Size | Min Size | Max Size | Total Size | Total Time | Name |
|-------|----------|----------|----------|------------|------------|------|
| 17616 | 119.05KB | 4.0000KB | 0.9961MB | 2.000000GB | 178.4112ms | Host To Device |
| 4336 | 432.18KB | 4.0000KB | 2.0000MB | 1.787109GB | 156.1746ms | Device To Host |
| 6185 | - | - | - | - | 556.7007ms | Gpu page fault groups |

Total CPU Page faults: 8211

Here cpu has more space so its not problem of exceeding memory we increase from 2^20 to 2^28 .