CS336 : Parallel & Distributed processing

Project 8 Report

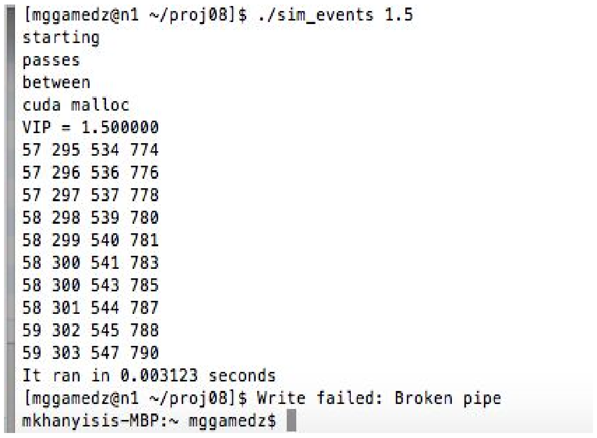
Summary of tasks

For this project the main aim was to implement a CUDA C version of the SCN oscillator simulations which uses CUDA C GPU parallel threads to make it run quickly. This projects emphasizes on creating a working version of sim_events. The host code generates the periods and the VIP strength. One kernel function performs the simulation and records the events. One network of oscillators is in one block of threads and each thread controls one oscillator. The GPU generates the list of CT6-crossing times, which are then copied back to the host.

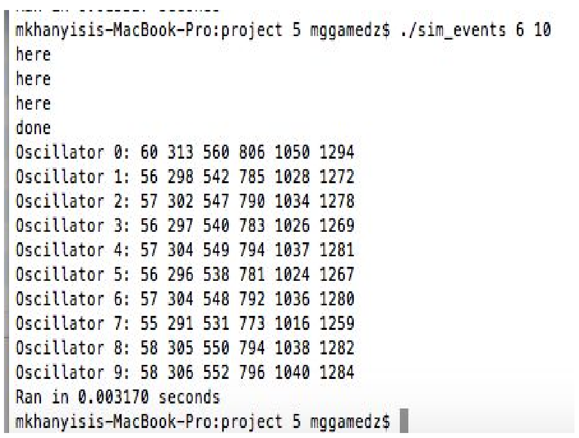
Tasks

1. I made my_math functions to be both host and device callable functions
2. I copied and pasted the code for sim_sizes.h
3. Phase_support. I copied the supporting code and updated the required code sections using class suggested methods. I also downloaded all the other supporting code.

7. For the same VIP strength, number of oscillators and number of events reported, the oscillation results are around the same ballpark. Below is a comparison with my project 5 code:



```
[mggamedz@n1 ~/proj08]$ ./sim_events 1.5
starting
passes
between
cuda malloc
VIP = 1.500000
57 295 534 774
57 296 536 776
57 297 537 778
58 298 539 780
58 299 540 781
58 300 541 783
58 300 543 785
58 301 544 787
59 302 545 788
59 303 547 790
It ran in 0.003123 seconds
[mggamedz@n1 ~/proj08]$ Write failed: Broken pipe
mkhanyisis-MBP:~ mggamedz$
```

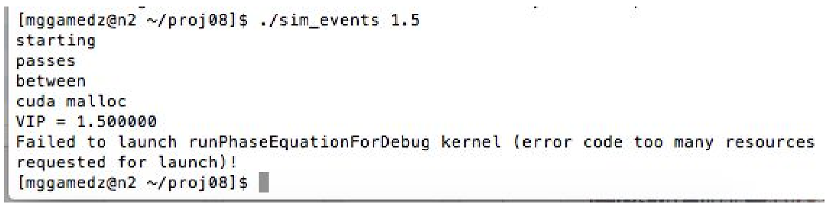


```
mkhanyisis-MacBook-Pro:project 5 mggamedz$ ./sim_events 6 10
here
here
here
done
Oscillator 0: 60 313 560 806 1050 1294
Oscillator 1: 56 298 542 785 1028 1272
Oscillator 2: 57 302 547 790 1034 1278
Oscillator 3: 56 297 540 783 1026 1269
Oscillator 4: 57 304 549 794 1037 1281
Oscillator 5: 56 296 538 781 1024 1267
Oscillator 6: 57 304 548 792 1036 1280
Oscillator 7: 55 291 531 773 1016 1259
Oscillator 8: 58 305 550 794 1038 1282
Oscillator 9: 58 306 552 796 1040 1284
Ran in 0.003170 seconds
mkhanyisis-MacBook-Pro:project 5 mggamedz$
```

This is a good sign this code works.

8. From project 7, I know that the thread limit per block is 1024 and since we use one block of threads for the SCN, I predicted this to be the limit. One cannot run multiple blocks & threads for the same SCN, as there are shared variables and the syncthread conditions would not hold between blocks and thus this would make our code break.

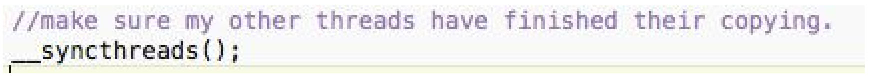
Since NX in sim_sizes.h controls the number of parallel threads, I assumed the limit for this would be 1024 as expected theoretically. However, that value fails to compile, as I get this error, with the threads failing to launch.



```
[mggamedz@n2 ~/proj08]$ ./sim_events 1.5
starting
passes
between
cuda malloc
VIP = 1.500000
Failed to launch runPhaseEquationForDebug kernel (error code too many resources
requested for launch)!
[mggamedz@n2 ~/proj08]$
```

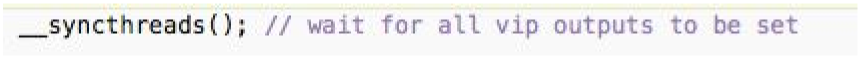
This different for my project 7. The limit for my code was NX=960, as for any higher value, there was a crash.

9. The purpose for my syncthreads__



```
//make sure my other threads have finished their copying.
__syncthreads();
```

This ensures the program starting its simulation after every thread having obtained its correct parameters that it will be using.



```
 __syncthreads(); // wait for all vip outputs to be set
```

For the phase velocity calculated right below this sync, the mean for the vi[] array is needed, and thus it makes sense that all the threads are in sync at that point not to omit any points. An error here could make the simulation inaccurate.

Extensions

None

Collaborators

I worked alone. Thanks to stephanie for help debugging.

.