15418: Final Project Milestone Report

Owen Lu (olu), Yutai Long (yutailon)

April 15, 2026

Webpage URL: https://toaster06.github.io/418-project-website/

1 Work Completed

We have completed most of our original Week 1–2 deliverables but are slightly behind on the optimizations. In particular, we have completed the following:

- **Serial CPU reference** (AOS layout, push streaming, double-buffered): this serves as our baseline for correctness and implements the core algorithm. Boundary handling (bounce-back, body forces, etc) and initial conditions are implemented here, and our GPU code is based upon this.
- **Naive CUDA baseline:** directly implements the naive implementation (like our CPU reference). This includes AOS layout, push streaming, one thread per cell, and two full grids.
- **Poiseuille flow validator:** we compare our algorithm's simulated $u_x(y)$ against the analytical solution $u(y) = \frac{f_x}{2\nu}(y - 0.5)(H - 1 - y)$. Our naive GPU implementation passes with relative L_2 error below the 5×10^{-2} threshold on a 64×64 grid, 20000 steps.
- **Benchmarking:** for benchmarking, we set up a testing suite that runs multiple initial conditions (Poiseuille, lid-driven cavity) at varying grid sizes and records MLUPS (Million Lattice Updates per Second, standard benchmark for LBM). We then take the mean, min/max, and coefficient of variation across multiple runs. We also started profiling with Nsight Compute and collected metrics like memory workload, cache hit rate, etc.

2 Preliminary Results

2.1 Naive GPU Baseline Throughput

Table 1 shows the performance of our naive GPU kernel, averaged over five runs per configuration after discarding an initial warmup run.

Note that we're keeping the total work ($W \times H \times$ steps) constant across all scenarios, so each run has a similar amount of work. We see that throughput rises with grid size: each timestep issues one kernel launch, so the 256×256 case pays 8000 launch overheads while the 1024×1024 case

Scenario	Grid	Steps	Time (s)	MLUPS	CV (%)
Poiseuille	256×256	8000	0.602	871.0	0.01
Poiseuille	512×512	2000	0.562	933.4	0.00
Poiseuille	1024×1024	500	0.552	949.9	0.04
Cavity	512×512	2000	0.560	936.7	0.00

Table 1: MLUPS for naive CUDA baseline

pays only 500. This overhead likely accounts for the MLUPS gap we observe. Cavity and Poiseuille deliver effectively the same throughput, confirming that boundary handling is not the bottleneck at these grid sizes.

2.2 Nsight Compute Profile of the Naive Kernel

Table 2 summarizes the key Nsight metrics collected for the naive kernel on a 512×512 grid.

Metric	Value	Interpretation
Global load sectors / request	16.00	Reflects the uncoalesced accesses of the AOS data layout
DRAM throughput (% peak)	24.80%	This translates to around 80 GB/s of 320 GB/s
SM throughput (% peak)	19.43%	This indicates that our workload is memory bound, as expected
Achieved occupancy	74.27%	Reasonable

Table 2: Nsight profiling for the naive GPU kernel at 512×512.

This data reveals that our naive kernel is *memory-bound* with coalescing issues. With 9 floats per cell stored contiguously (AOS), consecutive threads in a warp read addresses separated by 36 bytes. Moreover, the SM throughput of 19% vs. DRAM throughput of 25% confirms the kernel spends most of its time waiting on memory, not on computation.

This is the bottleneck our planned SOA + pull-streaming optimization is designed to eliminate. We expect DRAM throughput to rise toward 60 to 80% of peak. If the measured improvement falls short of this, the Nsight metrics (especially L2 hit rate and DRAM % peak) will guide our next optimizations. For instance, if cache misses remain reasonable but the redundant two-grid traffic is eating up memory bandwidth, we will proceed with our original plan of implementing the AA-pattern.

3 Status vs. Proposal Goals

Our original goals were: serial CPU baseline, naive GPU baseline, SOA + coalesced layout, AA-pattern in-place streaming, warp-level optimizations, and a profiling-driven analysis comparing them. We still believe all of these are achievable as we've completed the baselines and have set up our benchmarking suite. For the remaining 2 weeks, we plan on focusing our attention on optimizations.

However, since we're a bit behind, we decided that we will de-prioritize our stretch goals (the main one was extending to 3D) for now and commit the remaining time to thorough profiling and analysis of the core optimization sequence. We will revisit them only if Week 4 finishes ahead of schedule.

4 Poster Plan

We plan to show key Nsight metrics after each of our optimizations coupled with speedup graphs and MLUPS data for grids of different sizes. Additionally, we hope to include a visualization of Poiseuille or cavity flow.

5 Revised Schedule

- **Apr. 15–18:** SOA layout with pull streaming and row padding.
  - Implement and validate SOA kernel (Owen)
  - Run padding experiment and profile with Nsight (Yutai)
- **Apr. 19–23:** AA-pattern in-place streaming.
  - Implement even-step and odd-step kernels (Owen)
  - Build unit-test harness and handle boundary logic (Yutai)
- **Apr. 24–27:** Warp-level and occupancy experiments; direction depends on what profiling reveals after AA (both).
- **Apr. 28–29:** Aggregate measurements, generate figures, write final report (both).
- **Apr. 30:** Finalize report and prepare poster (both).

To keep track of the optimizations we make, we plan on maintaining a shared log with the git commit hash and exact `ncu` command for every measurement so numbers are reproducible.

6 Concerns

Our main concern is the AA-pattern boundary logic. This will likely be the most involved part of our project and will likely be prone to subtle indexing errors. We plan to trace the index math on paper for a single boundary cell before implementing, and to validate against both Poiseuille (analytical) and mass conservation.

A secondary concern is that some of our optimizations may not have as much of an effect as we were hoping for. We consider this an acceptable outcome as long as we can explain *why* with profiling data and if it reveals any interesting insights that guide further optimizations.