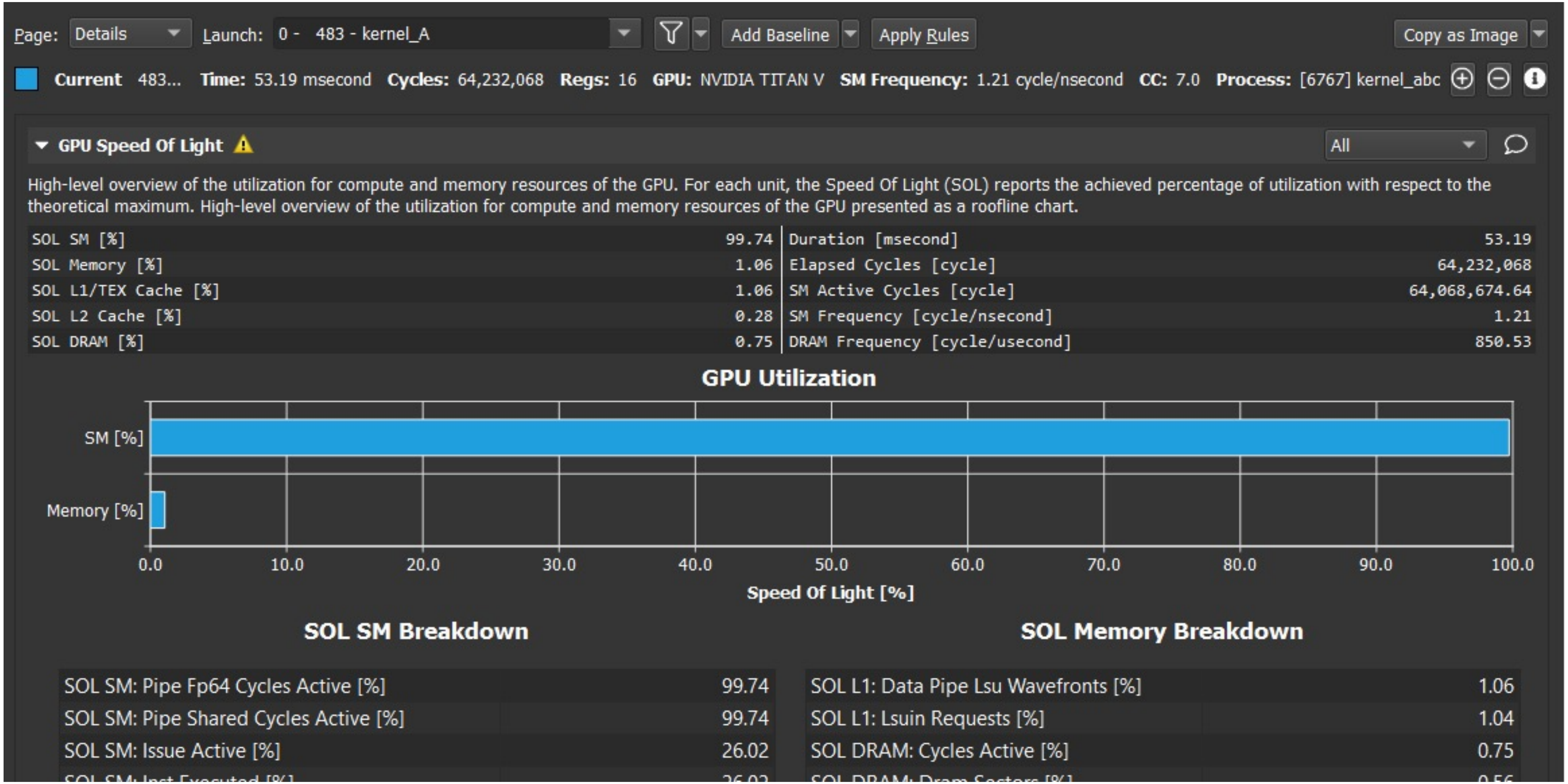
Roofline Model Using Nsight-Compute

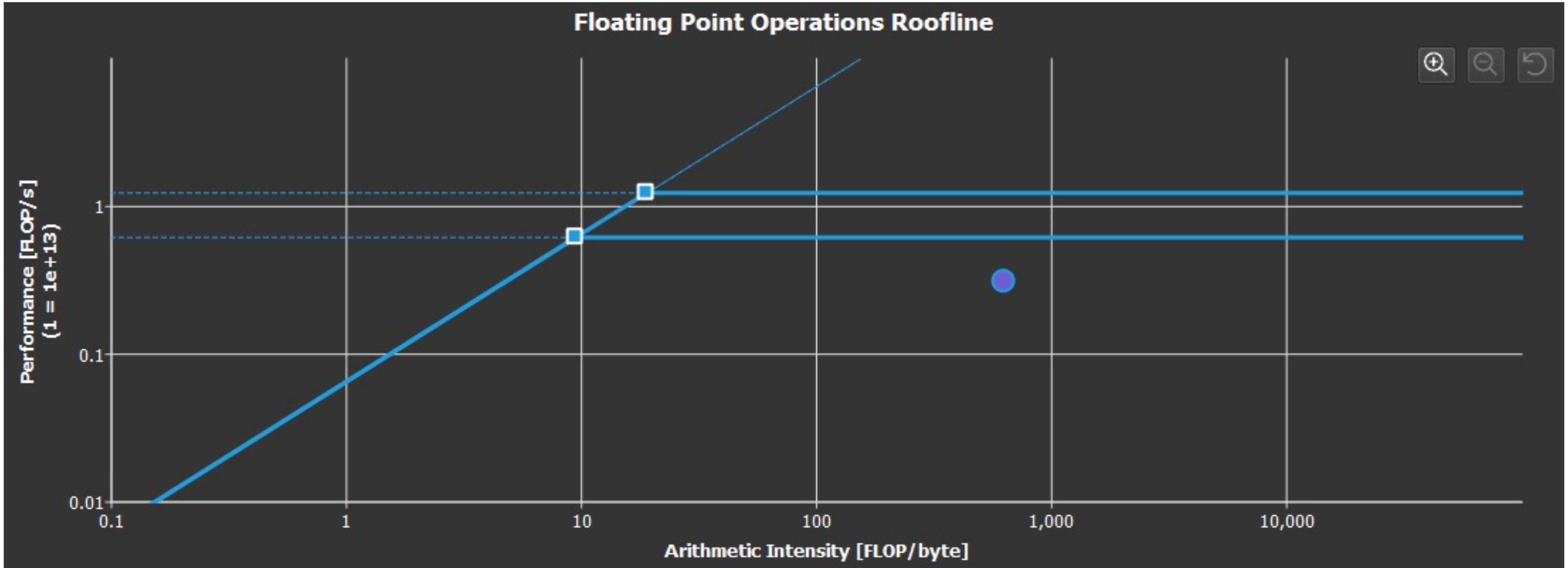
Jonathan Madsen
NERSC
Lawrence Berkeley National Lab
JRMadsen@lbl.gov

GPU Speed-of-Light



BERKELEY LAB

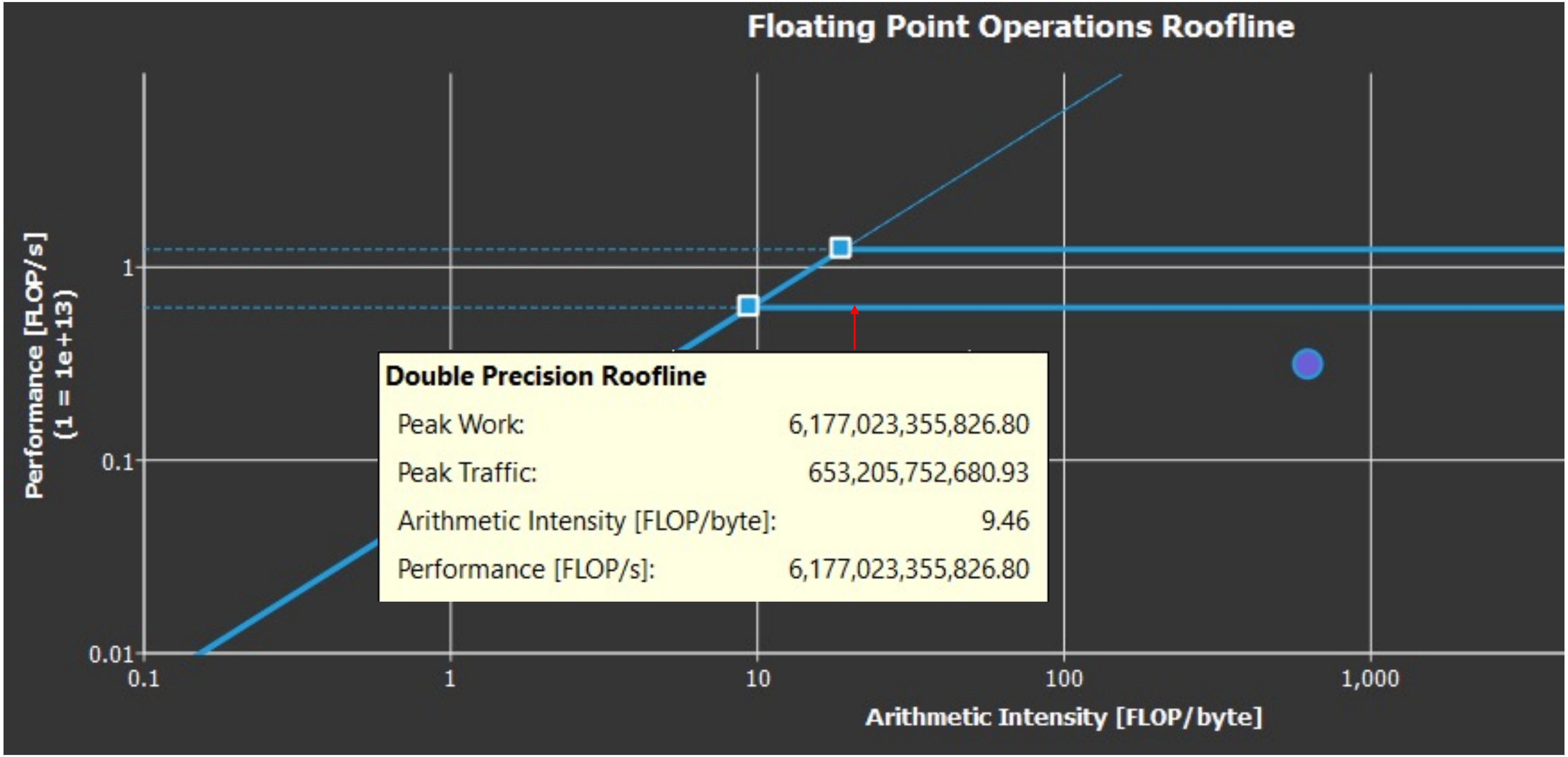
Roofline Chart



Recommendations

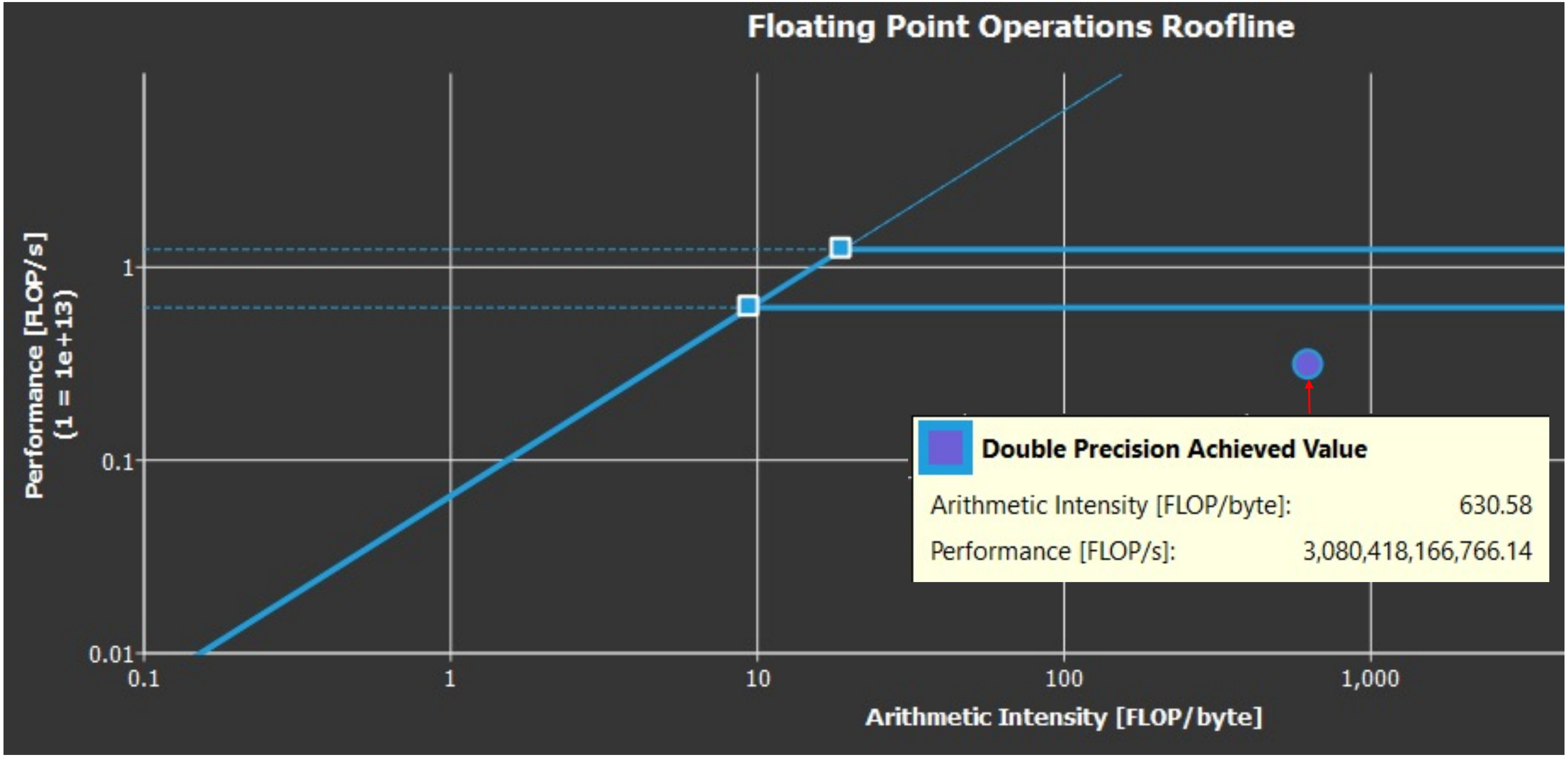
ⓘ	**Bottleneck**	The kernel is utilizing greater than 80.0% of the available compute or memory performance of the device. To further improve performance, work will likely need to be shifted from the most utilized to another unit. Start by analyzing workloads in the Compute Workload Analysis section.
⚠	**Roofline Analysis**	[Warning] The ratio of peak float (fp32) to double (fp64) performance on this device is 2:1. The kernel achieved 0% of this device's fp32 peak performance and 50% of its fp64 peak performance. If Compute Workload Analysis determines that this kernel is fp64 bound, consider using 32-bit precision floating point operations to improve its performance.

Roofline Chart: Peak Values

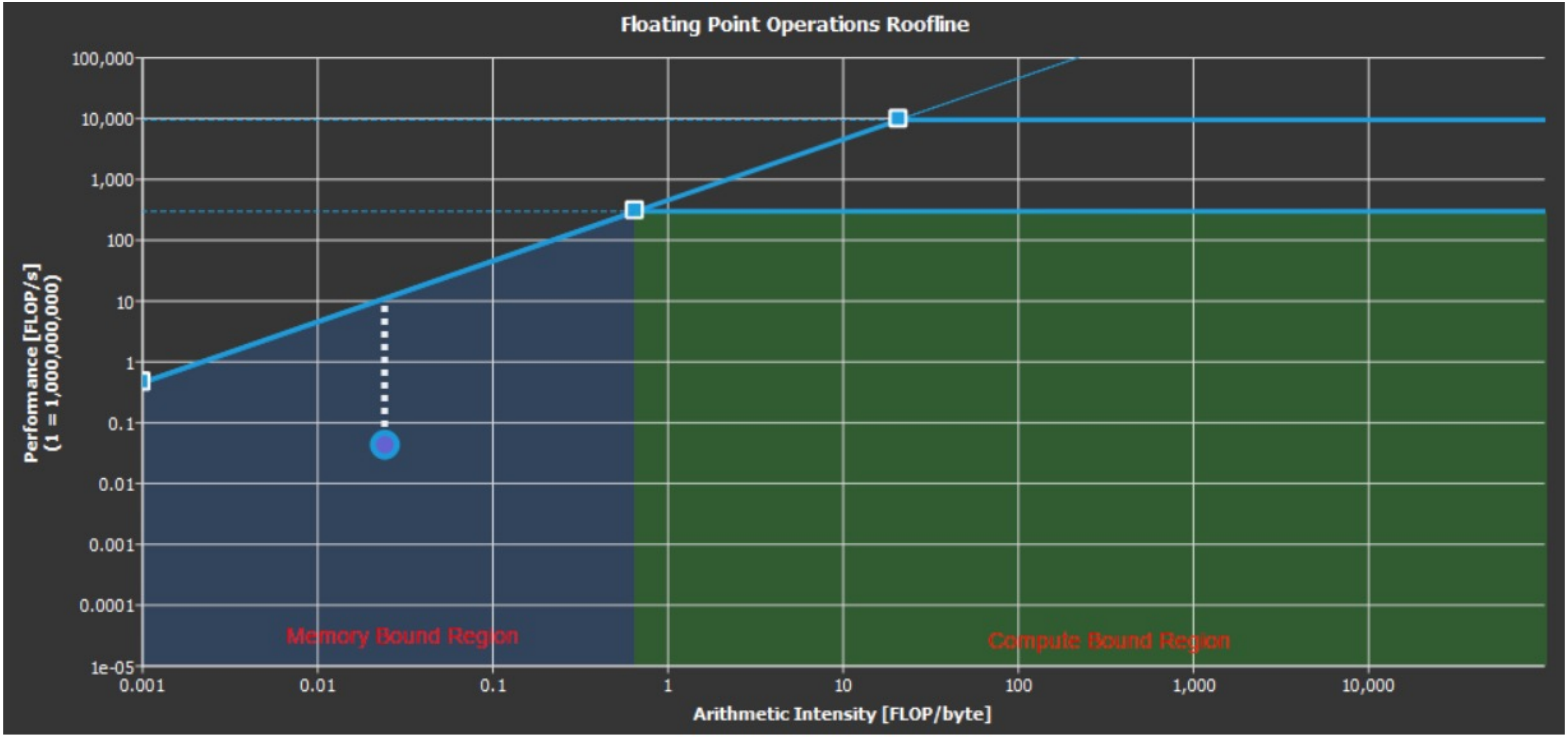


BERKELEY LAB

Roofline Chart: Achieved Values

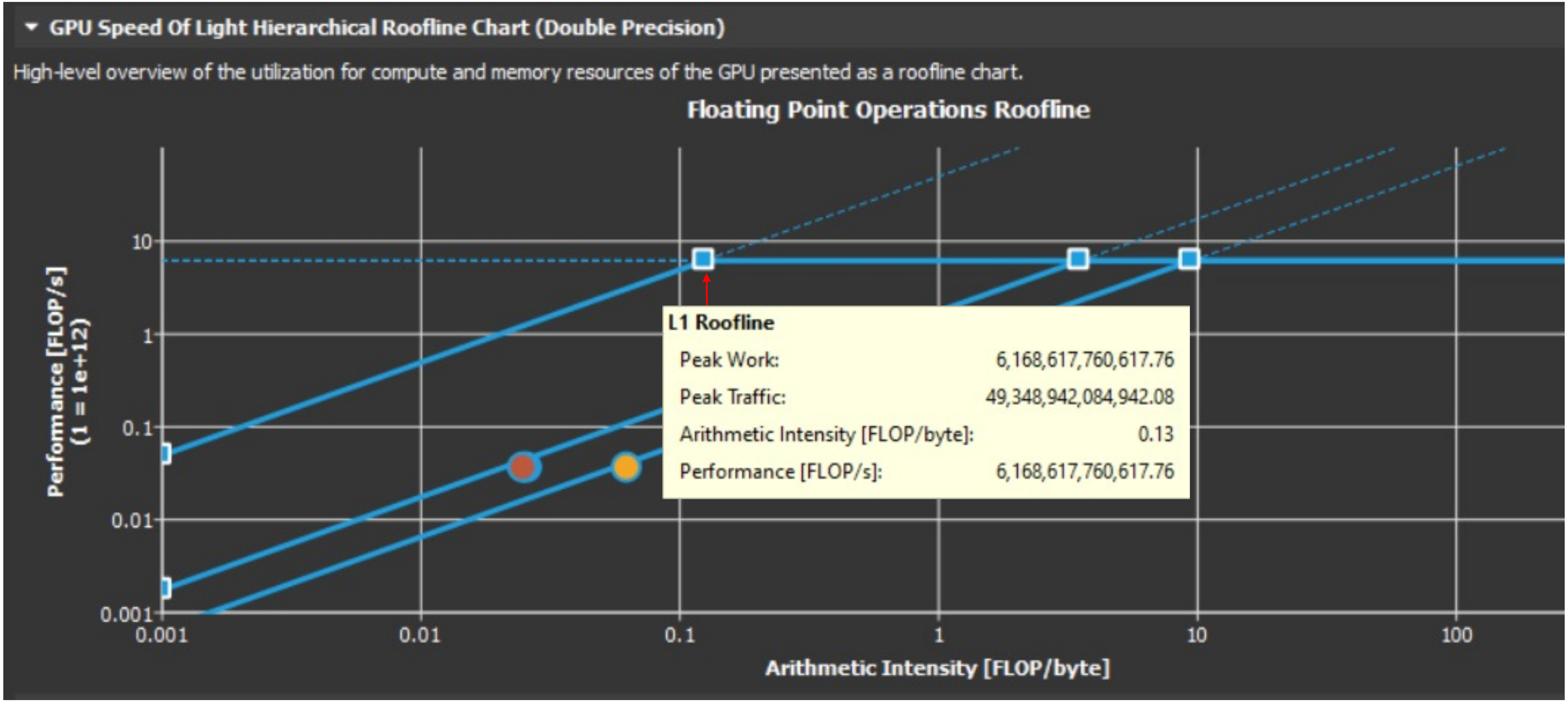


Roofline Chart: Regions

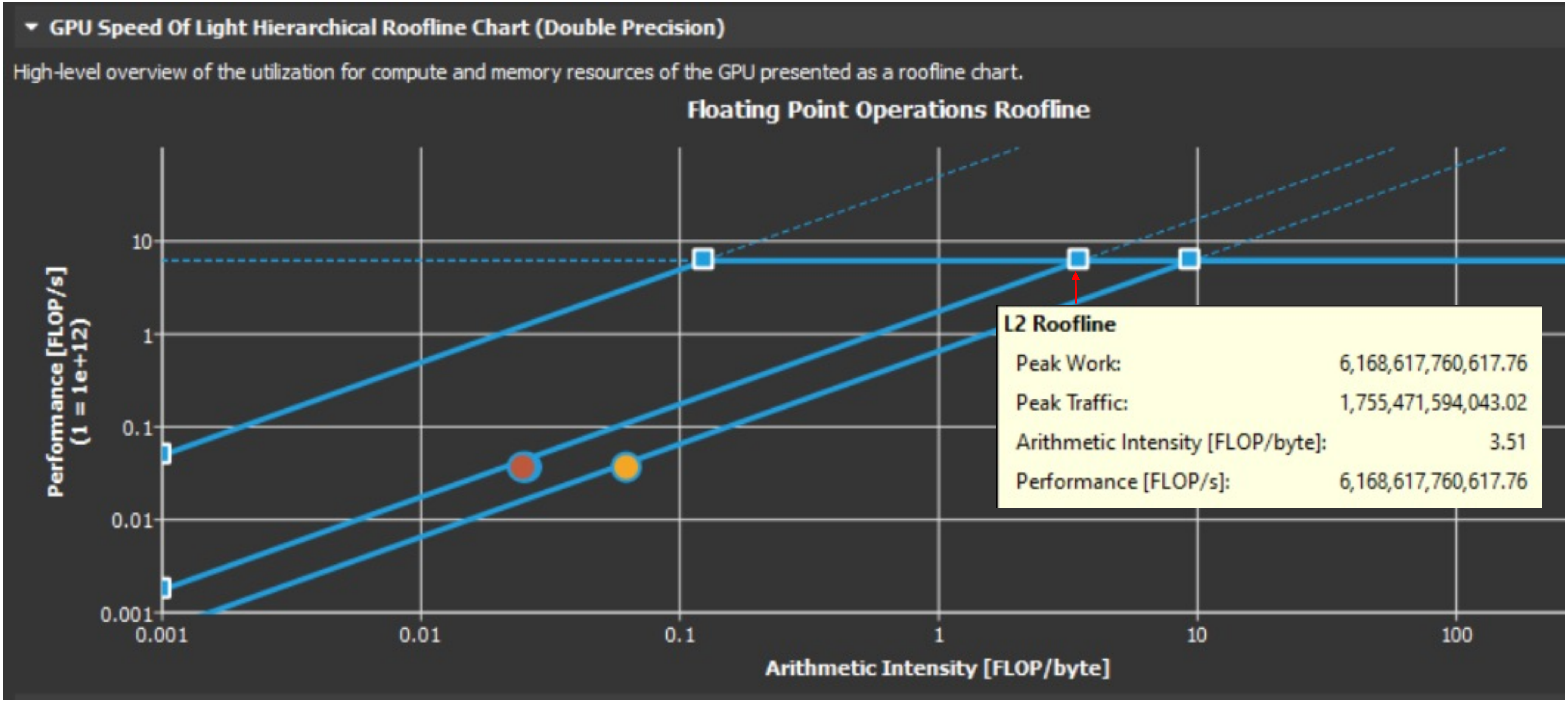


BERKELEY LAB

Hierarchical Roofline: L1 Peak Values

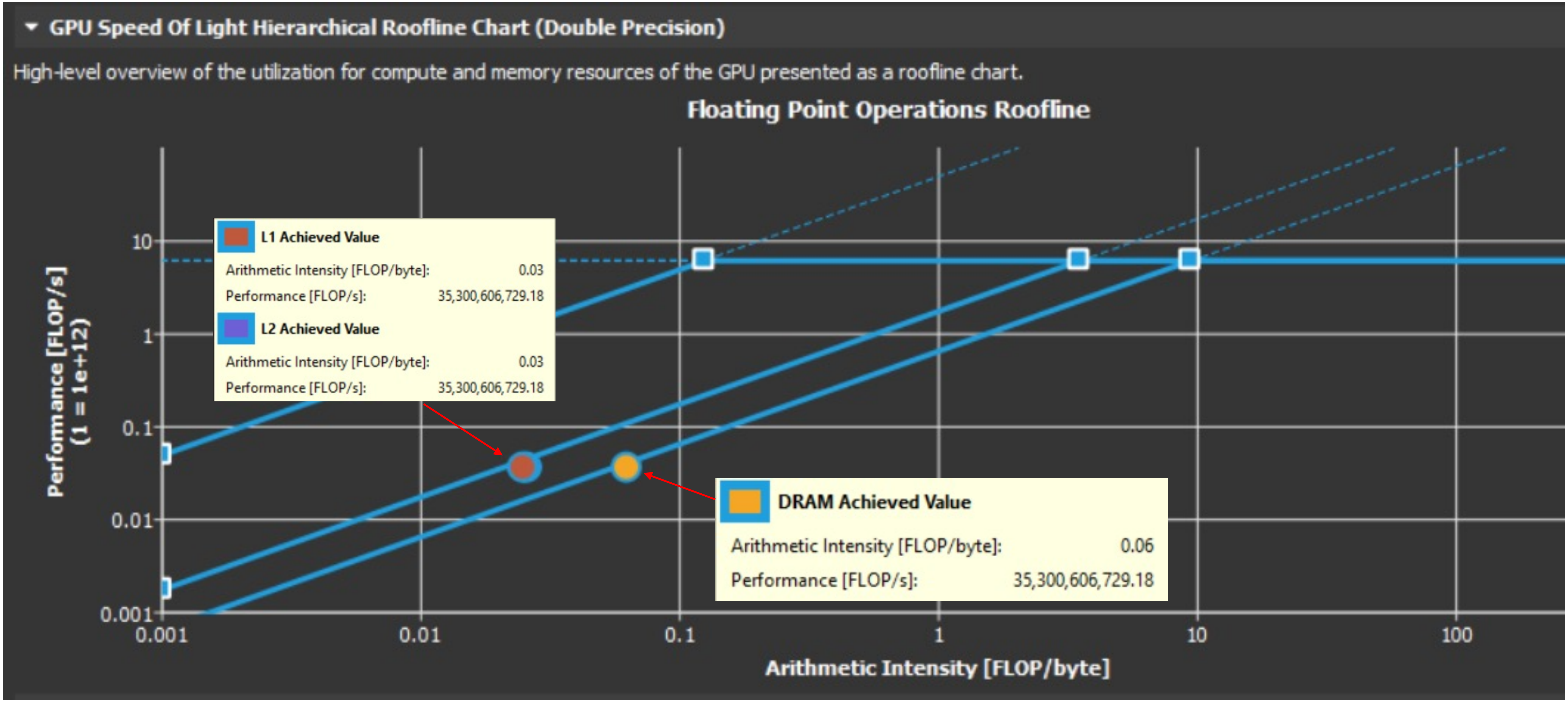


Hierarchical Roofline: L2 Peak Values



BERKELEY LAB

Hierarchical Roofline: Achieved Values

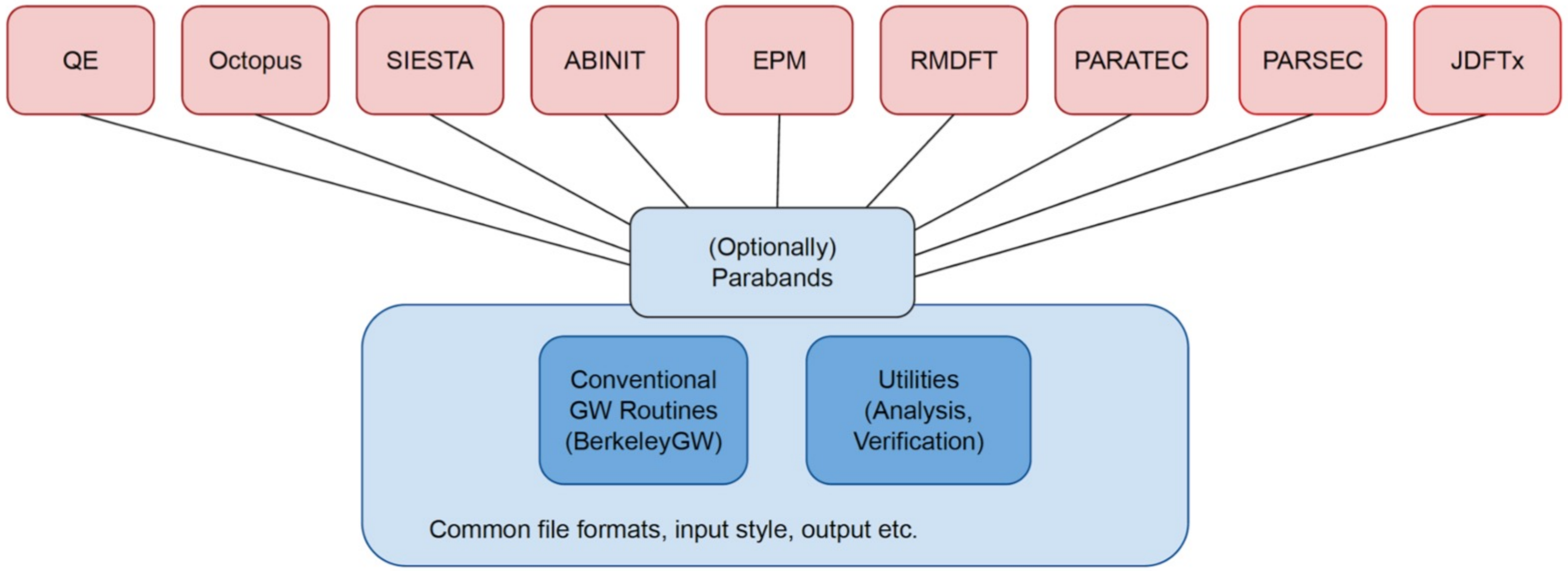


Roofline Analysis Example

BerkeleyGW

- Massively parallel package for GW calculations
- Sits on top of DFT codes

- Computational Motifs
  - FFTs
  - Dense linear algebra
  - Large reductions



BerkeleyGW

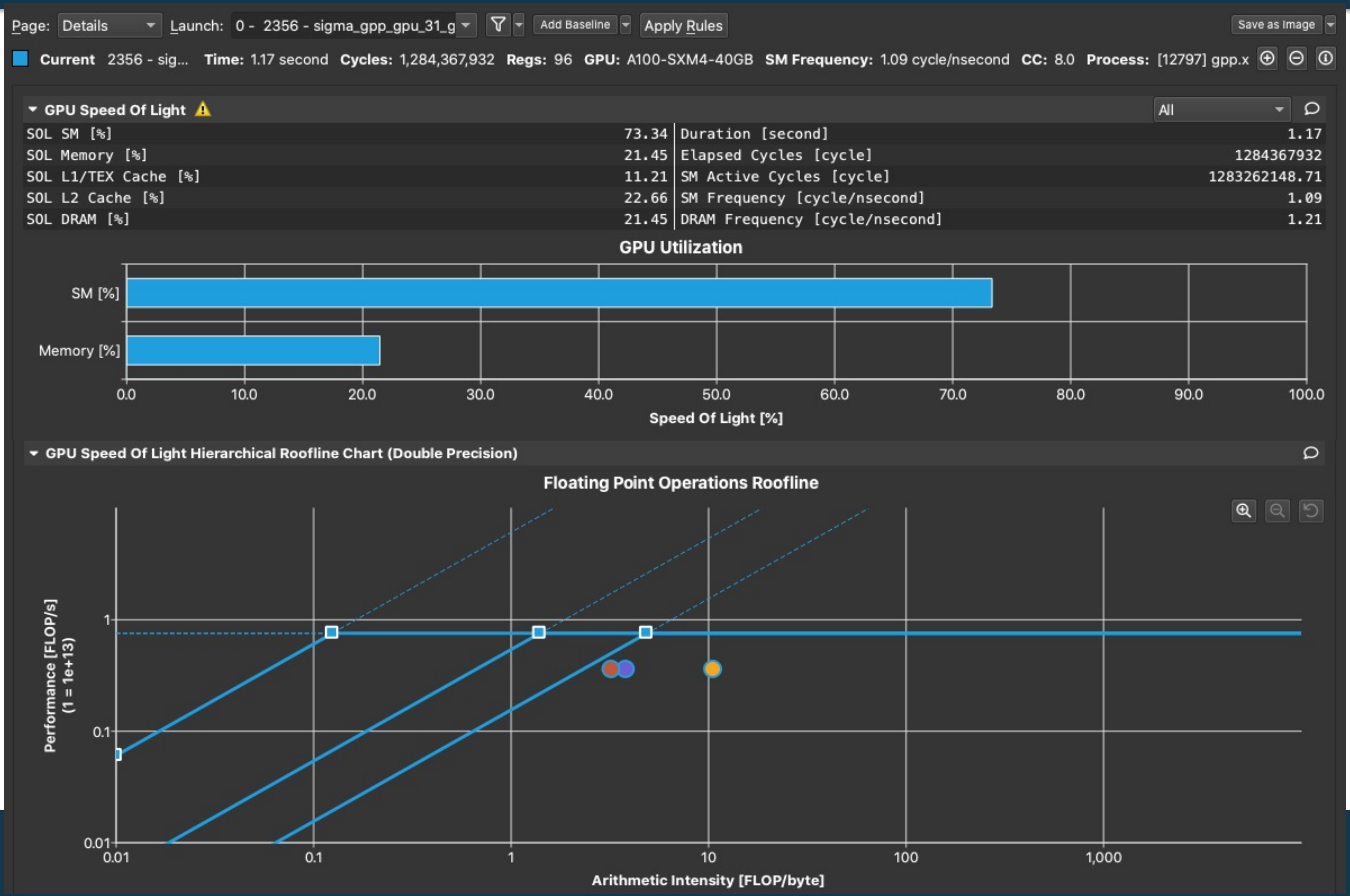
C2SEPEM

GPP Pseudocode

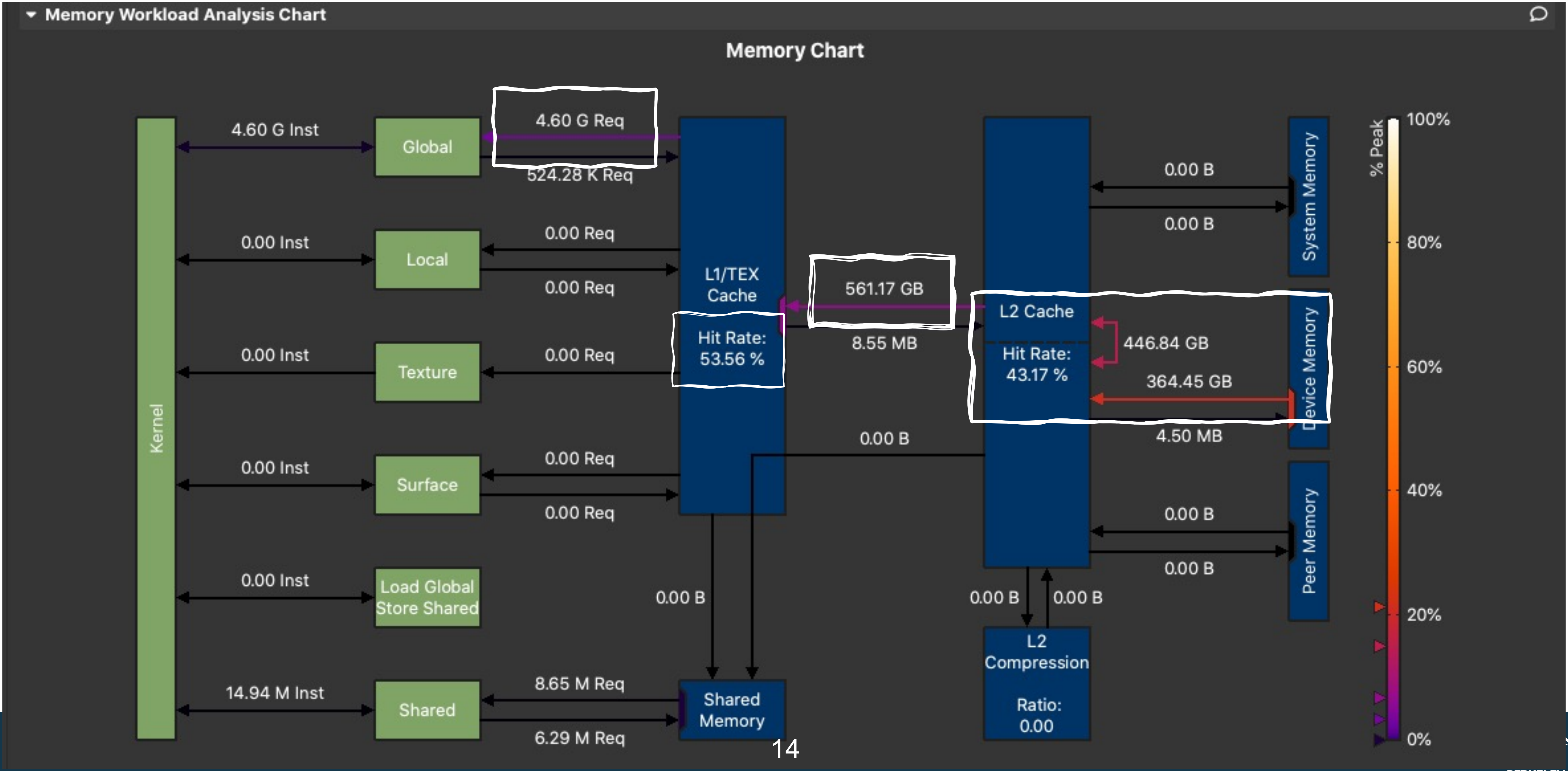
```
do i in 1, nbands:      ! n' ~= 2763
  do j in 1, ngpown:    ! G' ~= 6633
    do k in 1, ncouls:  ! G  ~= 26529
      do h in 1, nw:    ! E  ~= 3
        compute()     ! Mixed data types:
                      !   complex double, double, int
                      ! Various memory access patterns
                      ! Complex number divisions
        reduction()   ! Complex numbers
                      ! Billions of iterations
```


BERKELEY LAB

Initial GPU Port – SOL and Roofline



Initial GPU Port – Memory Analysis



Optimization Step #1: Loop Reordering

```
< !$ACC LOOP GANG VECTOR reduction(+:...) collapse(3)
< do i = 1, nbands ! O(1000)
<   do j = 1, ngpown ! O(1000)
<     do k = 1, ncouls ! O(10000)
---
> !$ACC LOOP GANG VECTOR reduction(+:...) collapse(2)
> do j = 1, ngpown ! O(1000)
>   do k = 1, ncouls ! O(10000)
>     !$ACC LOOP SEQ
>     do i = 1, nbands ! O(1000)
```

Runtime: 1.17 sec → 1.10 sec

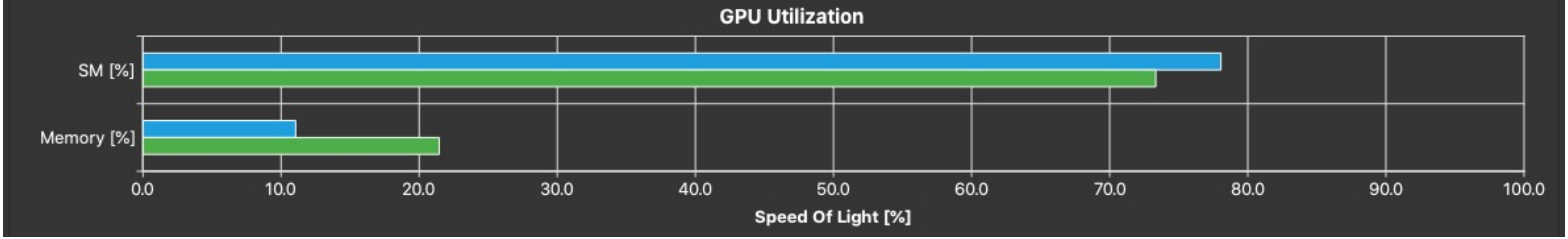
Speed-up: ~6%

Optimization Step #1: SOL

Current	2356 - si...	**Time:** 1.10 second	**Cycles:** 1,206,675,468	**Regs:** 118	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.09 cycle/nsecond	**CC:** 8.0	**Process:** [14360] gpp.x
v0	2356 - si...	**Time:** 1.17 second	**Cycles:** 1,284,367,932	**Regs:** 96	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.09 cycle/nsecond	**CC:** 8.0	**Process:** [12797] gpp.x

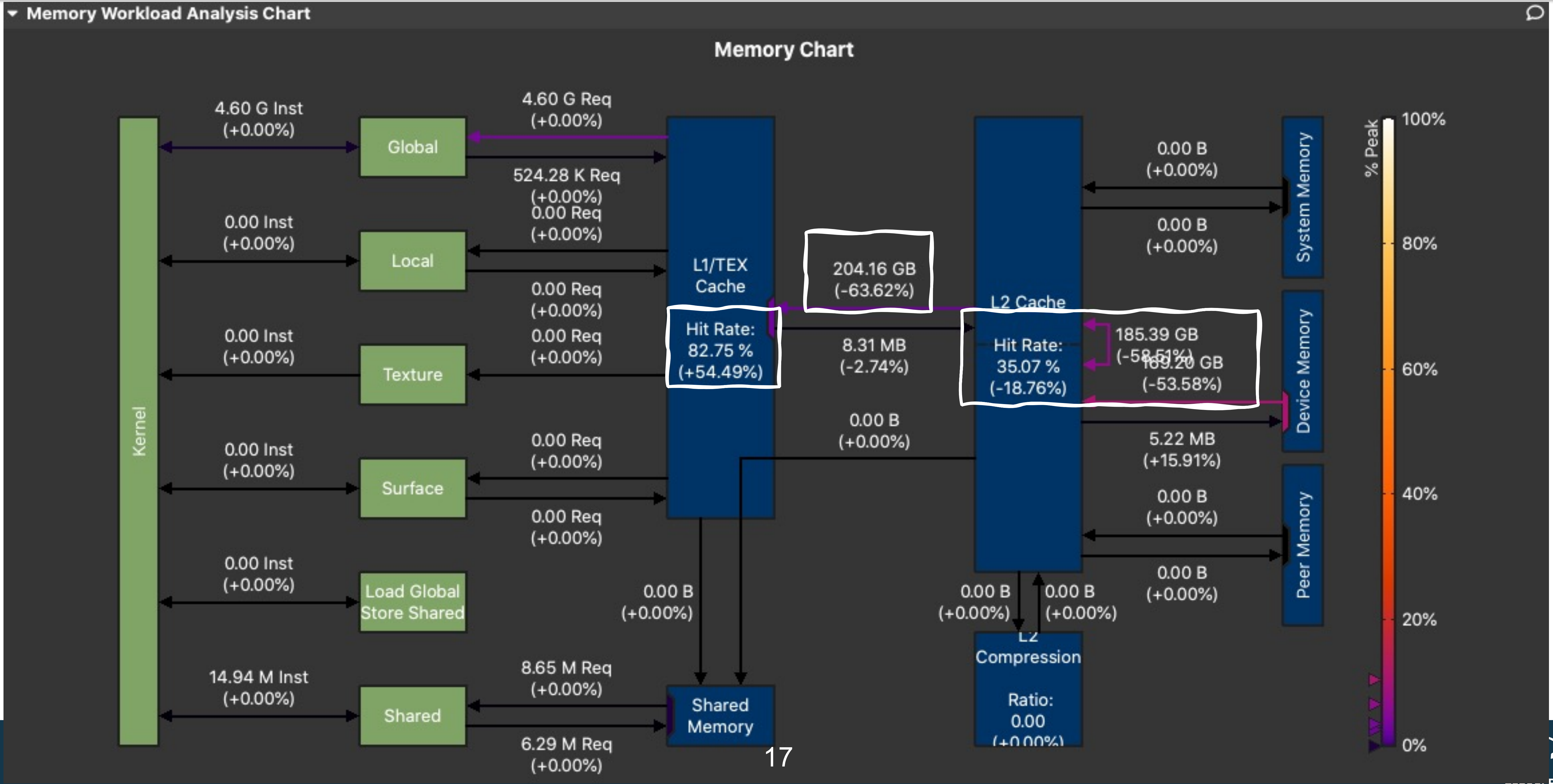
GPU Speed Of Light SOL Chart

SOL SM [%]	78.06	(+6.44%)	Duration [second]	1.10	(-6.07%)
SOL Memory [%]	11.06	(-48.47%)	Elapsed Cycles [cycle]	1206675468	(-6.05%)
SOL L1/TEX Cache [%]	11.06	(-1.35%)	SM Active Cycles [cycle]	1206014587.82	(-6.02%)
SOL L2 Cache [%]	9.99	(-55.90%)	SM Frequency [cycle/nsecond]	1.09	(+0.02%)
SOL DRAM [%]	10.60	(-50.59%)	DRAM Frequency [cycle/nsecond]	1.21	(+0.02%)

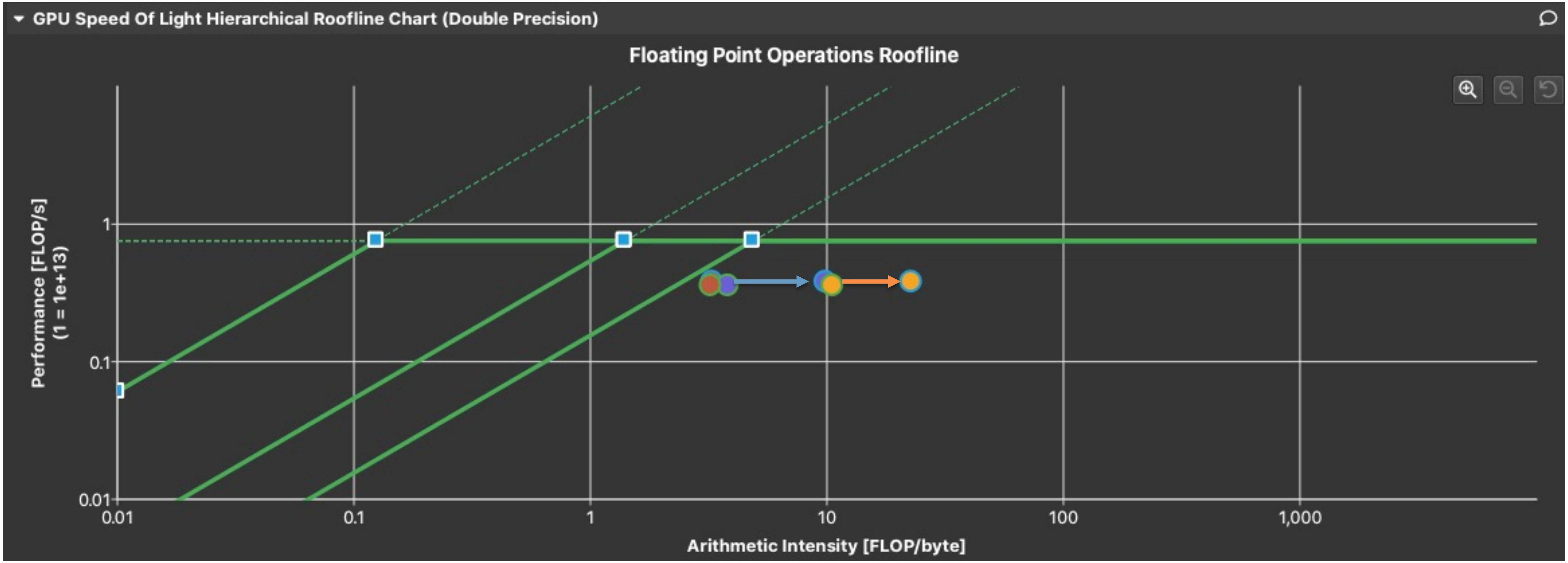


Bottom value (green) represents baseline

Optimization Step #1: Memory Analysis



Optimization Step #1: Roofline



BERKELEY LAB

Optimization Step #2: Data Reuse

```
< complex(DPC) :: ssx_array_2, ssx_array_3,
sch_array_2, sch_array_3
---
> complex(DPC) :: ssx_array, sch_array
```

- More changes to accommodate data restructuring
- Split kernel into two iterations

Runtime: 1.10 sec → 0.98 sec

Speed-up: ~11%

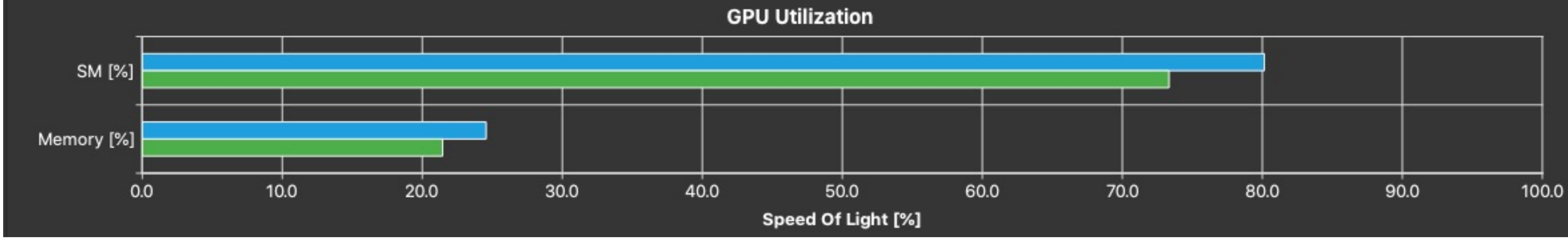
Total Speed-up: ~17%

Optimization Step #2: SOL

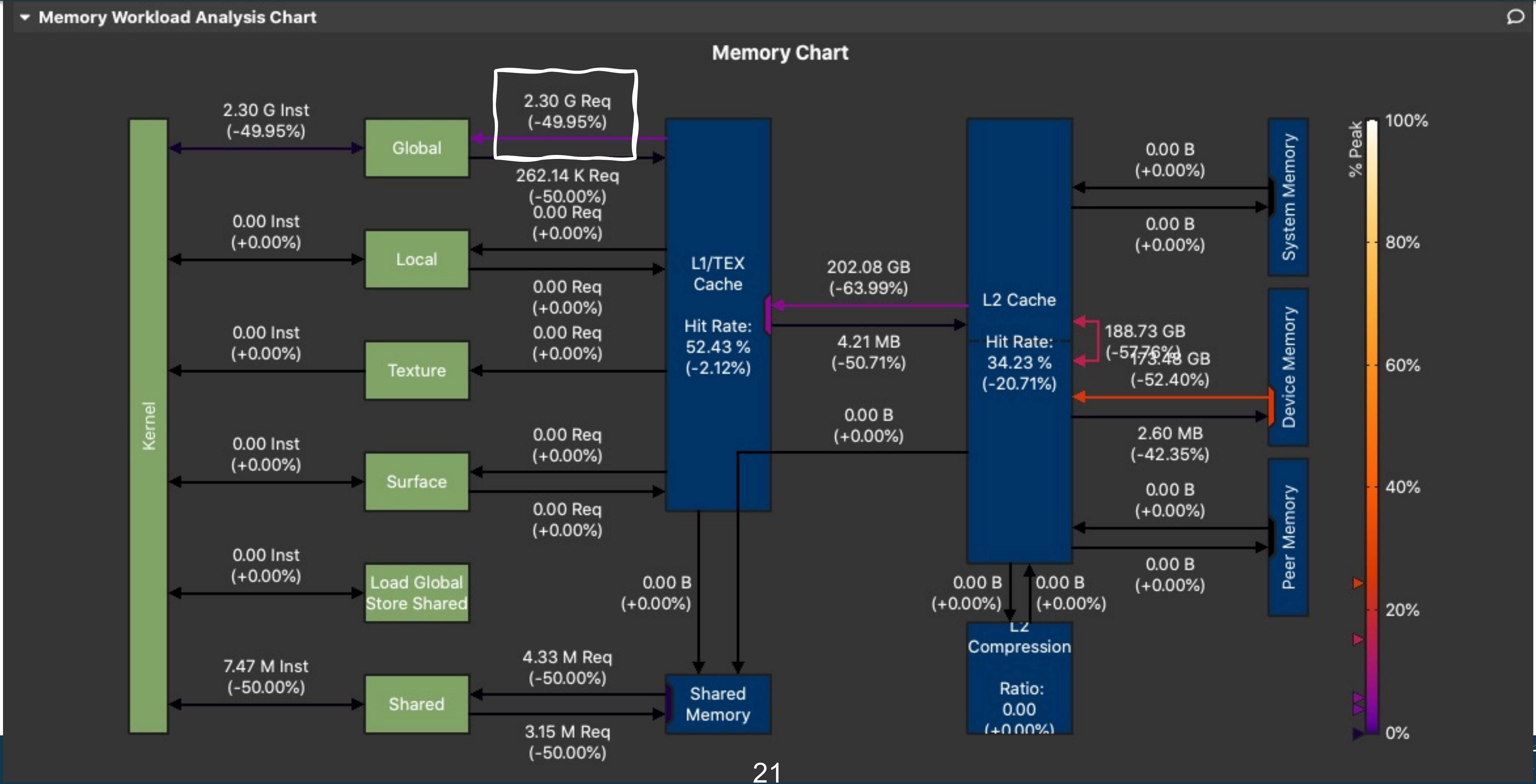
Current	2350...	Time: 487.39 msecond	Cycles: 534,036,779	Regs: 96	GPU: A100-SXM4-40GB	SM Frequency: 1.10 cycle/nsecond	CC: 8.0	Process: [15206] gpp.x
v0	2356...	Time: 1.17 second	Cycles: 1,284,367,932	Regs: 96	GPU: A100-SXM4-40GB	SM Frequency: 1.09 cycle/nsecond	CC: 8.0	Process: [12797] gpp.x

GPU Speed Of Light — SOL Chart

SOL SM [%]	80.15	(+9.28%)	Duration [msecond]	487.39	(-58.46%)
SOL Memory [%]	24.56	(+14.48%)	Elapsed Cycles [cycle]	534036779	(-58.42%)
SOL L1/TEX Cache [%]	11.66	(+3.95%)	SM Active Cycles [cycle]	533728027.36	(-58.41%)
SOL L2 Cache [%]	22.60	(-0.26%)	SM Frequency [cycle/nsecond]	1.10	(+0.09%)
SOL DRAM [%]	24.56	(+14.48%)	DRAM Frequency [cycle/nsecond]	1.22	(+0.09%)

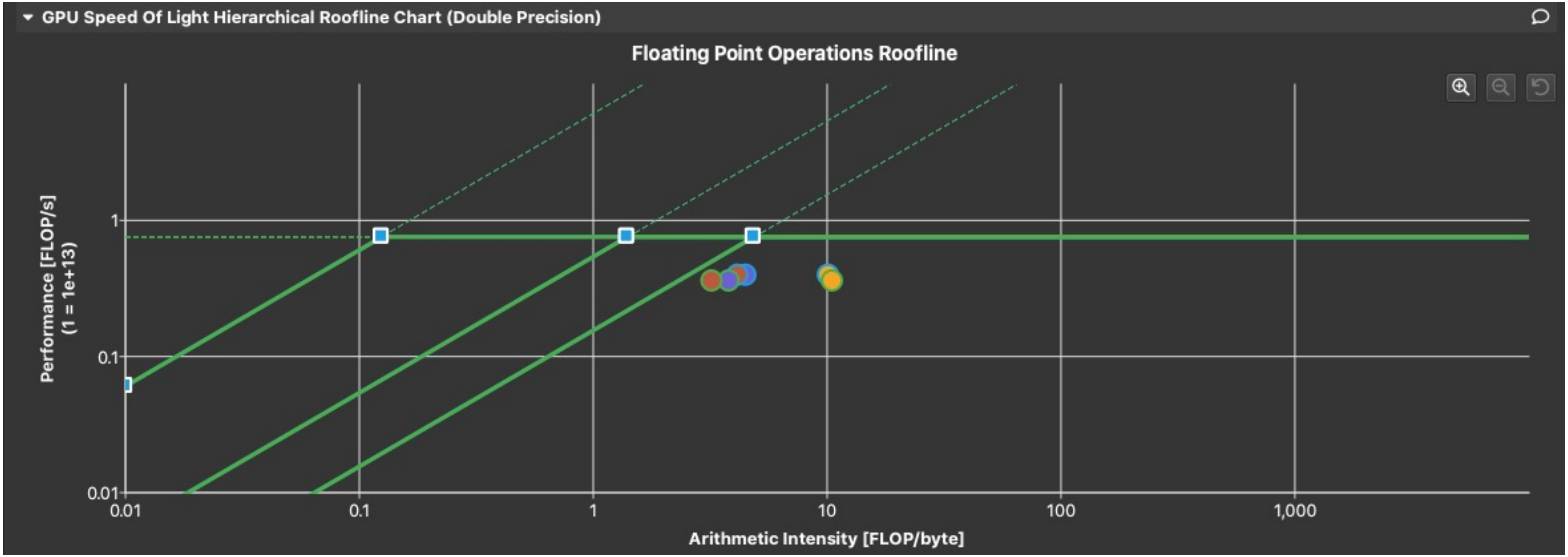


Optimization Step #2: Memory Analysis



BERKELEY LAB

Optimization Step #2: Roofline



BERKELEY LAB

Optimization Step #3: Arithmetic Optimization

```
< delw = wtilde / wdiff
<
< if (abs(ssx) .gt. ssxcutoff .and. ...) ssx=0.0d0
---
> wdiffr = wdiff * CONJG(wdiff) ! reciprocal math
> rden = 1.0d0 / wdiffr
> delw = wtilde * CONJG(wdiff) * rden
>
> rden = ssx * CONJG(ssx) ! replace abs with squares
> ssxcutoff = sexcut**2 * ...)
> if (rden .gt. ssxcutoff .and. ...) ssx=0.0d0
```

- division → reciprocal math & abs → squares

Runtime: 0.98 sec → 0.53 sec

Speed-up: ~45%

Total Speed-up: ~54%

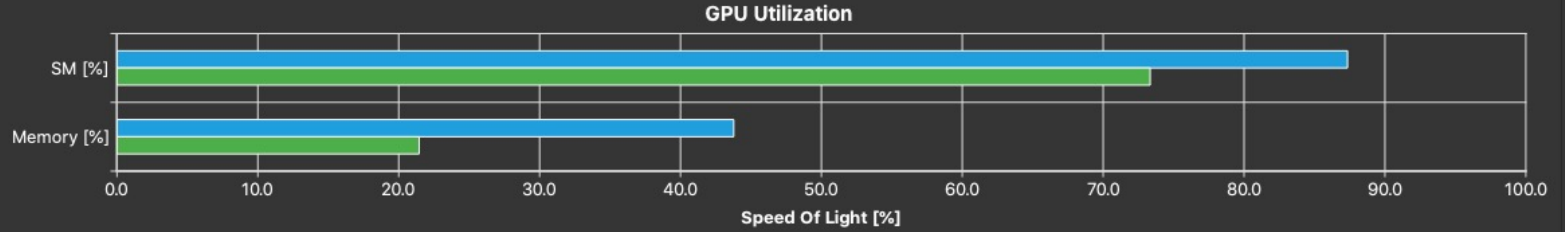
BERKELEY LAB

Optimization Step #3: SOL

Current	2350...	**Time:** 265.85 msecond	**Cycles:** 291,136,399	**Regs:** 94	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.10 cycle/nsecond	**CC:** 8.0	**Process:** [16767] gpp.x
v0	2356...	**Time:** 1.17 second	**Cycles:** 1,284,367,932	**Regs:** 96	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.09 cycle/nsecond	**CC:** 8.0	**Process:** [12797] gpp.x

GPU Speed Of Light — SOL Chart

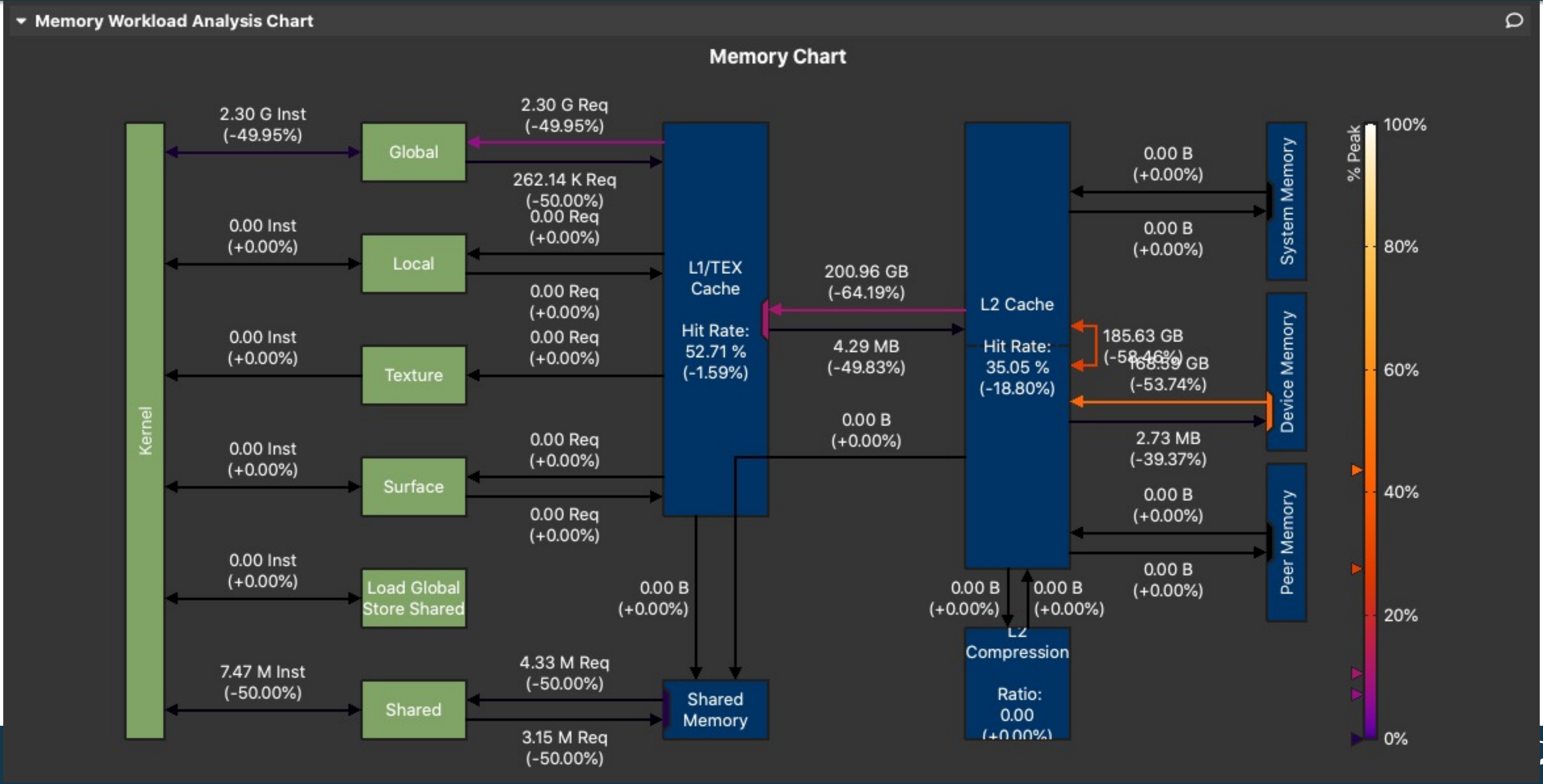
SOL SM [%]	87.35	(+19.10%)	Duration [msecond]	265.85	(-77.34%)
SOL Memory [%]	43.78	(+104.08%)	Elapsed Cycles [cycle]	291136399	(-77.33%)
SOL L1/TEX Cache [%]	21.38	(+90.70%)	SM Active Cycles [cycle]	290958596.19	(-77.33%)
SOL L2 Cache [%]	39.96	(+76.34%)	SM Frequency [cycle/nsecond]	1.10	(+0.03%)
SOL DRAM [%]	43.78	(+104.08%)	DRAM Frequency [cycle/nsecond]	1.22	(+0.03%)



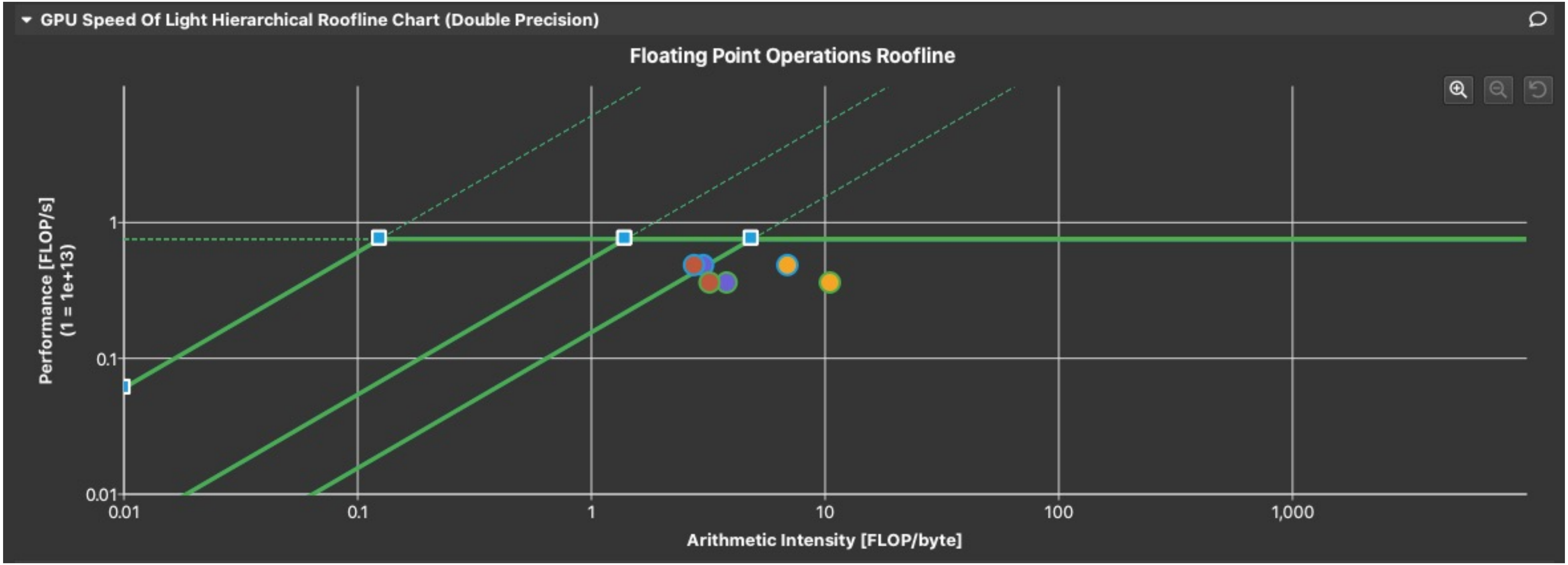
Recommendations

BERKELEY LAB

Optimization Step #3: Memory Analysis



Optimization Step #3: Roofline



Deoptimization Step #4: Fixed vector length

```
< !$ACC PARALLEL PRESENT(I_eps_array, aqsntemp)
---
> !$ACC PARALLEL PRESENT(I_eps_array, aqsntemp) vector_length(512)
```

- Set the vector length to a non-optimal value

Runtime: 0.53 sec → 0.58 sec

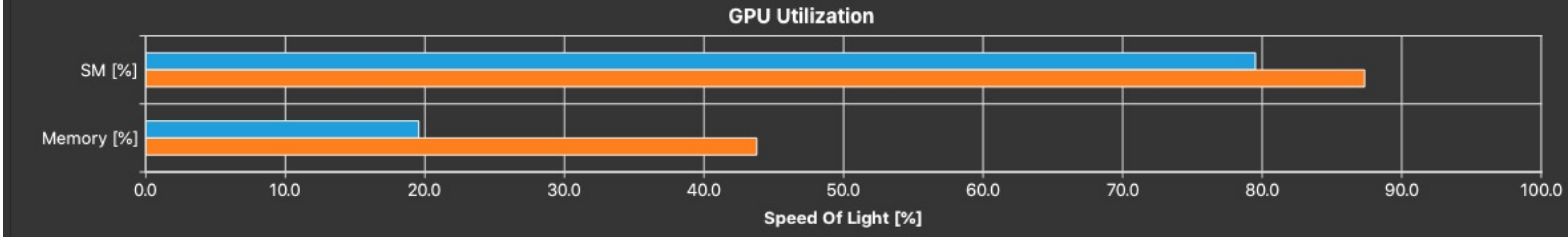
Speed-up: -10%

Step #4: SOL of v4 vs. v3

Current	2350 - ...	**Time:** 292.02 msecond	**Cycles:** 319,798,580	**Regs:** 95	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.10 cycle/nsecond	**CC:** 8.0	**Process:** [17738] gpp.x
v3	2350 - ...	**Time:** 265.85 msecond	**Cycles:** 291,136,399	**Regs:** 94	**GPU:** A100-SXM4-40GB	**SM Frequency:** 1.10 cycle/nsecond	**CC:** 8.0	**Process:** [16767] gpp.x

GPU Speed Of Light SOL Chart

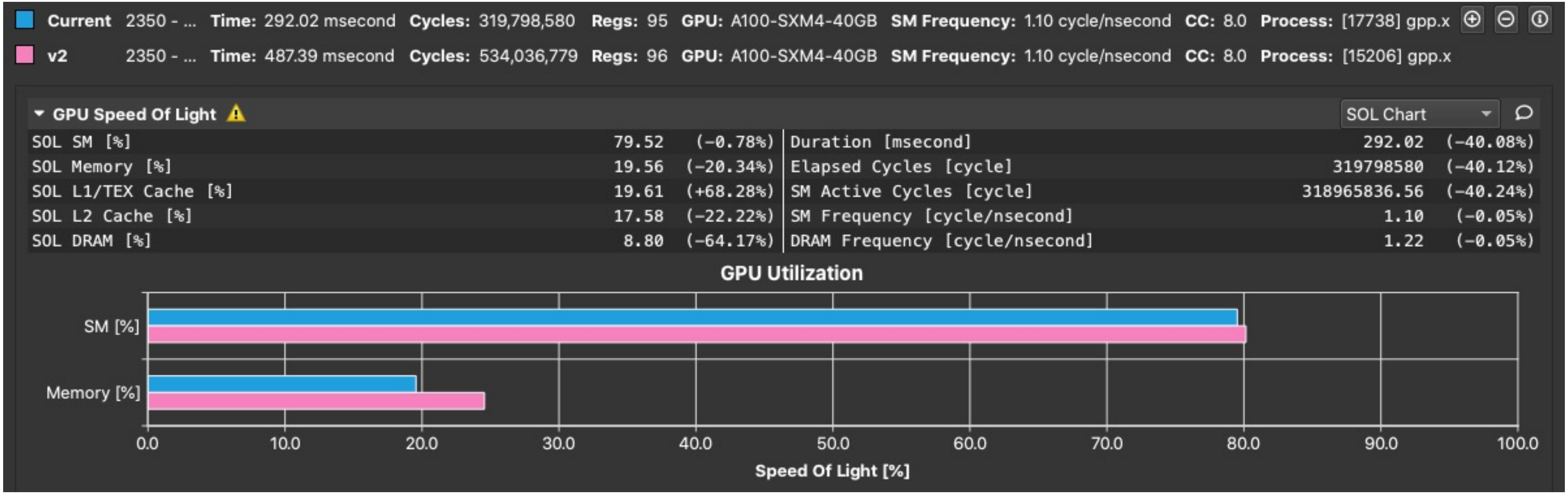
SOL SM [%]	79.52	(-8.96%)	Duration [msecond]	292.02	(+9.84%)
SOL Memory [%]	19.56	(-55.31%)	Elapsed Cycles [cycle]	319798580	(+9.84%)
SOL L1/TEX Cache [%]	19.61	(-8.27%)	SM Active Cycles [cycle]	318965836.56	(+9.63%)
SOL L2 Cache [%]	17.58	(-56.01%)	SM Frequency [cycle/nsecond]	1.10	(+0.00%)
SOL DRAM [%]	8.80	(-79.90%)	DRAM Frequency [cycle/nsecond]	1.22	(+0.00%)



Bottom value (orange) represents optimization step #3

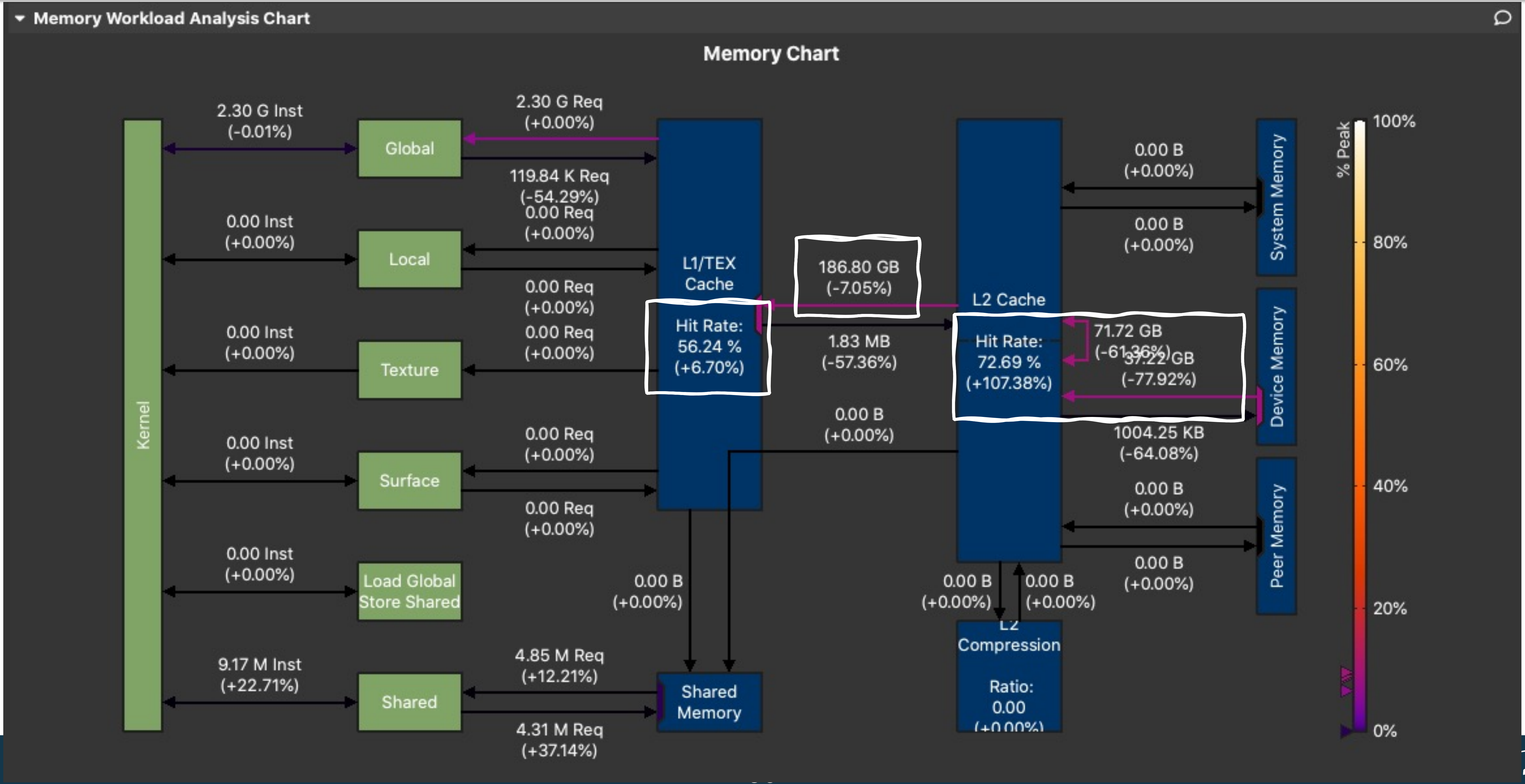
BERKELEY LAB

Step #4: SOL of v4 vs. v2

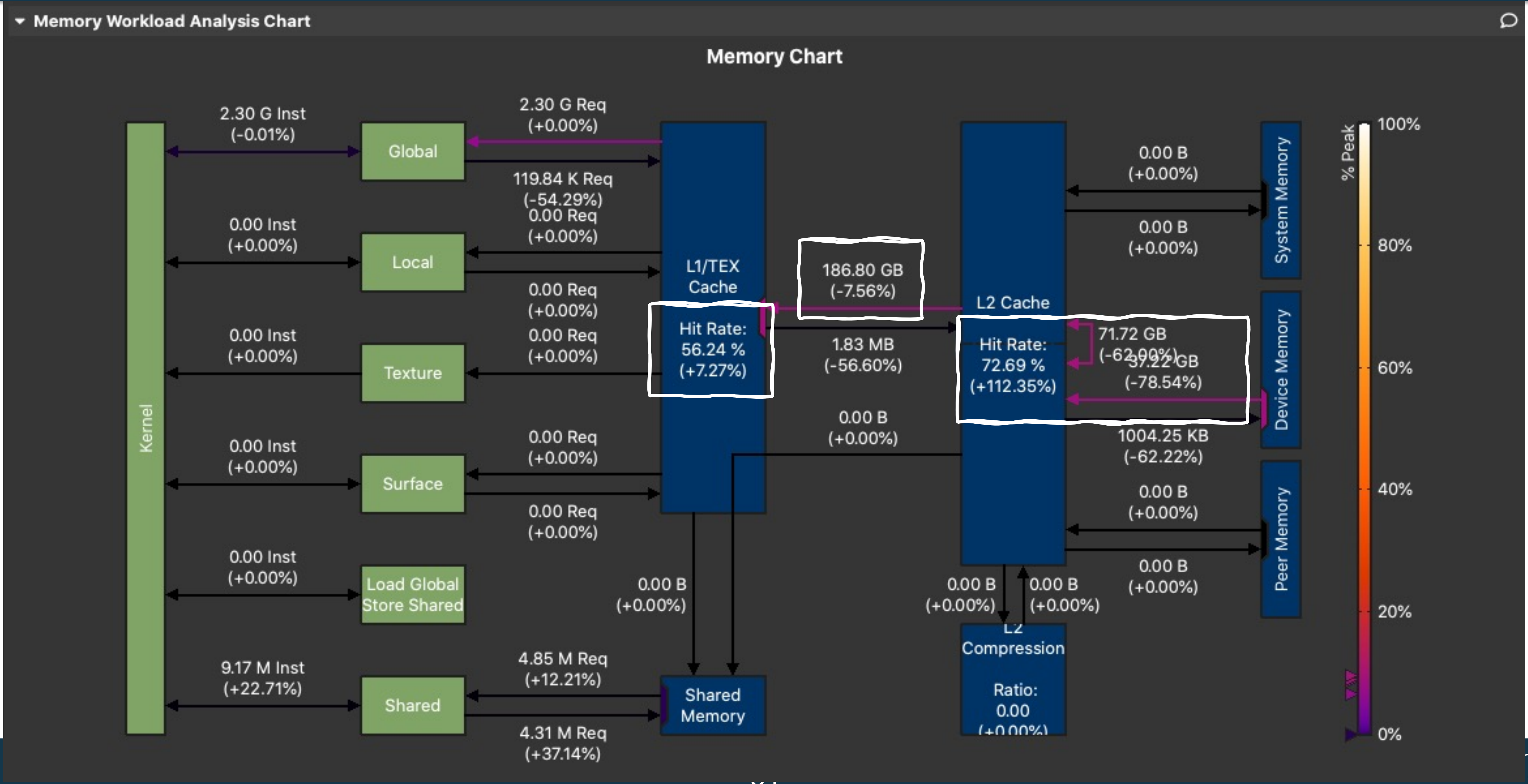


Bottom value (pink) represents optimization step #2

Step #4: Memory Analysis of v4 vs. v3



Step #4: Memory Analysis of v4 vs. v2



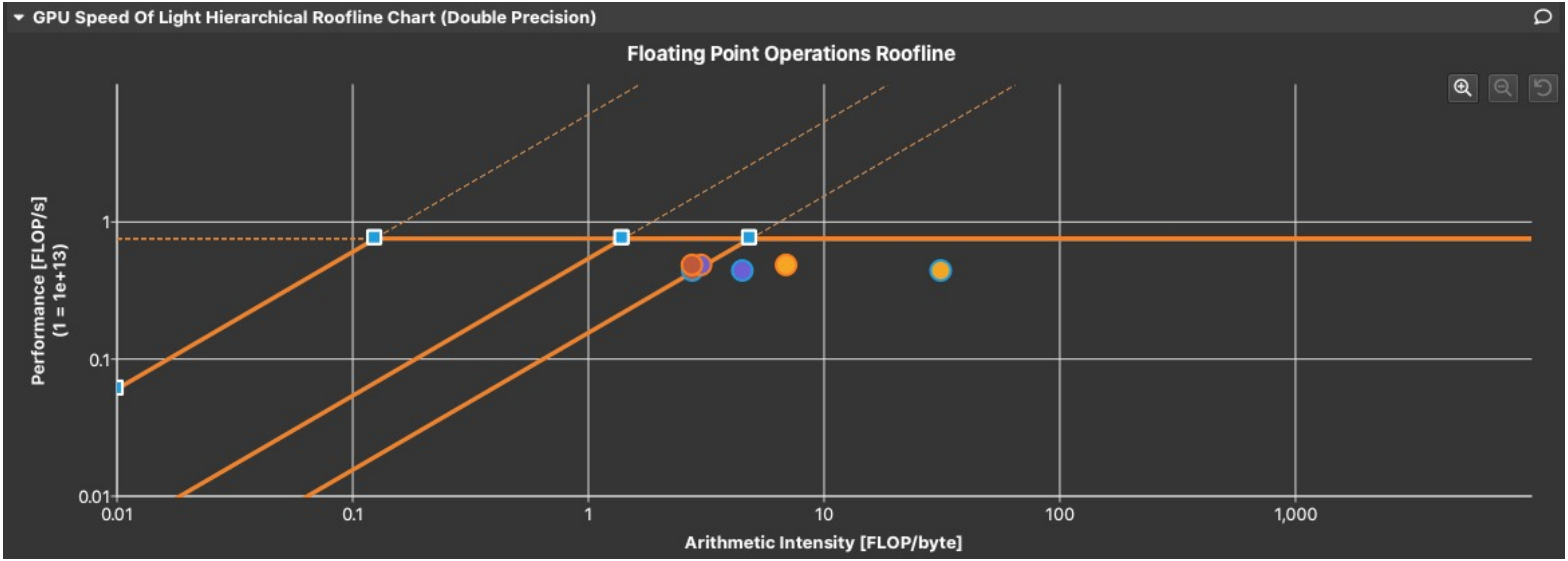
BERKELEY LAB

Recap of Step 4 vs. Steps 2 and 3

	Runtime (msec)	SM SOL %	Memory SOL %	L1 Cache Hit %	L2 Cache Hit %	Memory Traffic
Step #4	292	79.5	19.6	54.2	72.7	186 GB A* 71 GB B* 37 GB C*
Step #2	487	+0.8%	+25.54%	-6.8	-52.9	+8.2 % +163 % +366 %
Step #3	265	+9.8%	+123%	-6.2	-51.8	+7.6 % +158 % +352 %

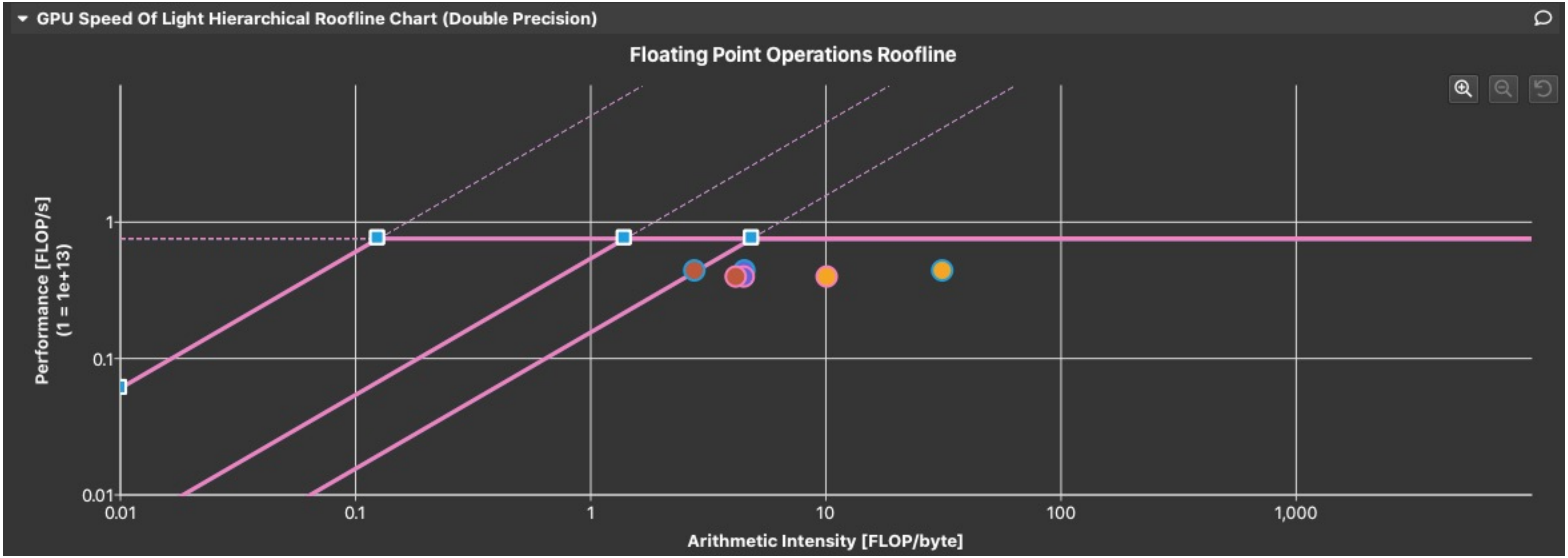
A* = L1 to L2 traffic, B* = L2 partition traffic, C* = Device to L2 traffic

Step #4: Roofline of v4 vs. v3



BERKELEY LAB

Step #4: Roofline of v4 vs. v2



BERKELEY LAB

References

- S. Williams, A. Waterman, and D. Patterson, “Roofline: An Insightful Visual Performance Model for Multicore Architectures,” *Commun. ACM*, vol. 52, no. 4, 2009.
- C. Yang, T. Kurth, and S. Williams, "Hierarchical Roofline Analysis for GPUs: Accelerating Performance Optimization for the NERSC-9 Perlmutter System", *Concurrency and Computation: Practice and Experience, DOI: 10.1002/cpe.5547*

Acknowledgement

- This research used resources at the National Energy Research Scientific Computing Center (NERSC), which is supported by the U.S. Department of Energy Office of Science under contract DE-AC02-05CH11231.
- This research used resources at the Oak Ridge Leadership Computing Facility (OLCF) through the Innovative and Novel Computational Impact on Theory and Experiment (INCITE) program, which is supported by the U.S. Department of Energy Office of Science under Contract No. DE-AC05-00OR22725.
- This work was supported by the Center for Computational Study of Excited-State Phenomena in Energy Materials (C2SEPEM), funded by the U.S. Department of Energy Office of Science under Contract No. DEAC02-05CH11231.