Comparing NVSim and Destiny Simulators on Memory Technologies with Varying Densities

Morgan Rockett
Dept. of Electrical and Computer Engineering
Tufts University
Medford, MA
morgan.rockett@tufts.edu

***Abstract*—This paper explores two simulators: NVSim and Destiny on evaluating RAM cells in 2D mode and also higher densities. Specifically, SRAM, STTRAM, RRAM, and PCRAM are evaluated in 2D by both NVSim and Destiny.**

Destiny also supports high-density cells, so simulations were ran on 3D RRAM and MLC RRAM to explore the additional capabilities of Destiny.

The result of this these experiments is a database of simulation results including: Total Area, Read Latency, Write Latency, Read Bandwidth, Write Bandwidth, Read Dynamic Energy, Write Dynamic Energy, and Leakage Power as outputs.

Each of the following outputs was ran with eight optimization targets: Total Area, Read Latency, Write Latency, Read Dynamic Energy, Write Dynamic Energy, Read Energy Delay Product, Write Energy Delay Product, Leakage Power.

And these simulations were modeled on each cell type: SRAM, STTRAM, RRAM, PCRAM, 3D RRAM, and MLC RRAM. In addition, charts are provided to visualize results.

***Index Terms*—SRAM, STT-RAM, STTRAM, ReRAM, RRAM, PCRAM, MLC, 2D, 3D, NVSim, Destiny, modeling tool, validation.**

I. Introduction

The purpose of this experiment is twofold. First, it would be helpful to determine if one of the two state-of-the-art simulators, NVSim or Destiny, is better overall – or perhaps in certain domains – given the task of RAM cell modeling. Also, there is interest in exploring the 3D and high-density capabilities of Destiny to evaluate the tool's features. With a data-driven approach and hundreds of simulations, this paper identifies advantages of each tool and provides suggestions for optimal use cases for each.

II. Background and Related Work

In this class, Special Topics: Emerging Memory Technologies, several homework assignments allowed us to explore NVSim on caches and ram cells. Destiny came up in name only but was recommended during final project proposal feedback as a worthy tool to compare NVSim with. Additionally, CACTI, the simulator was another tool that could be tested, but was not nearly as general of a tool as NVSim or Destiny [5]. In the paper "DESTINY: A Comprehensive Tool with 3D and Multi-Level Cell Memory Modeling Capability", the authors write that CACTI does not work with all the optimization targets that NVSim does.

Two papers, "NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory" [1] and "DESTINY: A Tool for Modeling Emerging 3D NVM and eDRAM caches" [2] serve as the core material for benchmarking. The former paper, from 2012, explains how the tool was developed for simulation in the industry and was expected to work well with non-volatile memory technologies, as well as the RAM cells I chose to test. The latter paper, from 2015, discusses the abilities of Destiny and the additional cell types that can be simulated such as 3D RRAM and MLC cells.

After reviewing a lot of the source code between NVSim and Destiny, it appeared that there was much overlap, but Destiny had more features, as claimed, so comparing 2D results would be of interest to see what actually differs in standard mode. High-density MLC and 3D exploration of RRAM cells could prove valuable and it is important to see if Destiny is overall a better tool to be using for researchers and for those in industry.

III. Methodology

A python script was developed to generate configuration files, create a directory tree, run all simulations, output simulation results to logs, parse the logs to create csv files, and create graphs based on those csv sheets [4]. Latency was converted to ns, Area to um squared, Leakage Power to uW, Energy Delay Products to pJ, and Dynamic Energy to pJ.

All eight supported optimization targets: Total Area, Read Latency, Write Latency, Read Dynamic Energy, Write Dynamic Energy, Read Energy Delay Product, Write Energy Delay Product, Leakage Power were iterated through in each simulation. In the 2D realm, the RAM cell types: STTRAM, SRAM, RRAM, and PCRAM were tested on NVSim and Destiny. For Destiny high-density testing, 3D RRAM cells and MLC RRAM cells were used for simulations.

As for output results, the following fields were parsed from output files: Total Area, Read Latency, Write Latency, Read Bandwidth, Write Bandwidth, Read Dynamic Energy, Write Dynamic Energy, and Leakage Power. Generally all eight output fields were used in graphing; however a few combinations did not work with the simulators, which is discussed in the 'Limitations' section.

The script generated configuration files for each unique simulation, and used the 22 nanometer technology node for the smallest available process technology supported, for consistency and to be as close to state-of-the-art results as possible. A comprehensive sweep of capacities from 16 - 4096KB, incrementing by each power of two, was used as a consistent x-variable despite other variations.

IV. Resources

A MacBook Pro with 64GB of unified memory ran the simulations without issue. For equivalent simulations, Destiny took approximately four times longer than NVSim to run.

V. Results

A. 2D NVSim vs Destiny [STTRAM SRAM RRAM]

Since NVSim did not generate results for PCRAM, the comparison here is limited to 2D, and STTRAM, SRAM, and RRAM between NVSim and Destiny.

In Fig. 1, with Total Area being plotted vs Capacity, the optimization target of Area allowed for the best distinction between the categories. Note that a log scale is used. Looking at NVSim in isolation for all three cell types, they rise in linear fashion. On the other hand, Destiny shows that STTRAM doesn't trend higher in area until capacity grows beyond 1MB. Similar results occur with Destiny on RRAM, but Destiny with SRAM is most similar to NVSim results. Neither simulator produces consistently lower Total Area results, so it is inconclusive as to which is better overall here.

Moving on to Fig. 2, choosing Read Latency as an optimization target for that same measure highlights SRAM in both simulators. While STTRAM and RRAM Read Latency stay virtually unchanged as capacity increases, SRAM grows linearly for Destiny and NVSim. The gap between the two stays roughly fixed with Destiny yielding lower latency times. It appears that in NVSim STTRAM there is a slight uptick in latency once capacity reaches the megabyte sizes, unlike Destiny.

Fig. 3 with Write Latency plotted and optimized is similar to Fig. 2 with Read Latency results, with one difference. STTRAM and RRAM for both simulators are nearly identical; however, RRAM shows slightly higher write latency results. Looking at the prior chart, the read latency appeared nearly the same between STTRAM and RRAM cells. The y-axis for write latency shows that the RRAM and STTRAM times are close to an order of magnitude higher than read latency, lest we get too distracted by trends alone.

Read Latency in Fig. 4 had the most significant impact on Read Bandwidth. Although on a logarithmic scale STTRAM and RRAM bandwidth dropped moderately as capacity increased - which makes sense due to data access causing slowdowns - the SRAM again took the spotlight showing the most dramatic decreases. With by far the highest initial Read Bandwidths at over 100 GB/s at 16KB capacity, both tapered down and Destiny stayed at higher throughput in all the simulations. For all three cell types Destiny yielded roughly the same or higher Read Bandwidth values vs NVSim.

Fig. 5 does not vary much from Fig. 4, with Write Bandwidth being the y-variable this time. Oddly enough, using an optimization target of Read Latency gave the biggest gap between SRAM and the other cell types. I did not anticipate Read Latency impacting Write Bandwidth, so that is why I included this graph as an unexpected trend. In the Read Bandwidth chart, there are small variances between STTRAM and RRAM of the two simulators; whereas here they are the same with only SRAM showing gaps between simulators.

Fig. 6, which shows a dual Read Dynamic Energy (RDE) result and optimization target, shows NVSim consistently delivering lower energy results. SRAM on NVSim really shines especially at lower capacities, while the Destiny counterpart shows much more energy in every capacity. STTRAM and RRAM have similar correlations between the two tools but they are not as far off as the SRAM discrepancy. When aiming for low RDE values, NVSim looks to be the tool of choice.

A dual Write Dynamic Energy (WDE) result and optimization target, in Fig. 7, again shows superior WDE values with NVSim and SRAM. Looking at STTRAM, the gap between Destiny and NVSim is much closer than RDE. On the other hand, RRAM does not vary much from the RDE deltas with both simulators.

Fig. 8, which finishes off this results section, had the most interesting chart when plotting Leakage Power with the same as an optimization target. NVSim with SRAM showed the highest increase as scaled with capacity, with Leakage Power leading all others until 256KB was tested. With STTRAM, Destiny stayed nearly an order of magnitude above NVSim, with Leakage Power curving upwards as capacity reached 2MB. With RRAM, Destiny barely changed Leakage Power as capacity grew; however, NVSim gapped up once 1MB capacity was reached, nearly closing the deficit of Leakage Power as seen in smaller capacities.

In summary, a few general trends became apparent from these simulations. Destiny tends to give higher bandwidth values and lower latency. NVSim shows lower energy consumption. So each tool could be selected if certain goals were priority. SRAM generally came out superior to STTRAM and RRAM, showing its competitiveness in 2D memory cell technologies.

B. 2D Destiny [STTRAM SRAM RRAM PCRAM]

Since NVSim did not yield PCRAM results, this section shows PCRAM in comparison to the other three cell types with Destiny 2D modeling.

The Total Area chart in Fig. 9 with Area as an optimization chart made the least sense in terms of capacity scaling. There was low correlation with RRAM, and several values appeared out of place with PCRAM and STTRAM. The most normal cell was SRAM as area relates to capacity gains. PCRAM looked the best here given the lowest area starting at 128KB.

Fig 10. shows a dual Read Latency result and optimization. STTRAM and RRAM show almost no change despite capacity growth; however, SRAM and PCRAM show tight correlation and linear increases with capacity getting larger. SRAM is better with capacities of 256KB and below, but PCRAM takes over for lower latency most noticeably at 1MB and above.

A Write Latency result and optimization target, as featured in Fig. 11, show PCRAM to be the slowest cell technology. PCRAM, STTRAM, and RRAM showed no change as capacity grew, but SRAM had a linear increase proportionate to capacity. The latter cell type had by far the lowest latency by over an order of magnitude.

Fig. 12, showing Read Bandwidth optimized by Read Latency reveals SRAM and PCRAM are contenders for highest throughput. STTRAM and RRAM and much slower by over an order of magnitude and all cell types decrease bandwidth as capacity scales up.

In Write Bandwidth, Fig. 13, optimized by write bandwidth, PCRAM trails SRAM and even STTRAM and RRAM. This chart appears strange for this deviation. SRAM again shows very high bandwidth performance.

This optimization target of Read Dynamic Energy has the largest difference between cell types when looking at the Read Dynamic Energy vs Capacity results, Fig. 14. It is interesting to note that RRAM actually uses less energy as capacity increases. STTRAM appears unaffected with capacity, and SRAM energy increases as it scales up. PCRAM is the most dramatic compared to the other cell types but does follow a high correlation with capacity - uses lots of energy at higher capacities.

Looking at Fig. 15, it is the most dramatic of all the graphs with an Optimization Target of Write Latency. PCRAM consumes the most energy here by a wide margin; however none of the memory cell types besides SRAM appear to use more energy as capacity grows.

This graph, Fig. 16, shows that RRAM has no correlation with capacity when it comes to Leakage Power. STTRAM doesn't show linear correlation with capacity until the larger capacities such as 512KB are simulated. The closest log correlation with leakage power and capacity occurs at SRAM, showing that on smaller capacities, this is the best cell type to use if leakage power is the primary concern. The PCRAM has the strangest correlation to the capacity as it jumps in series every few capacity increases.

Capacity appears to have little influence on Write Bandwidth, if anything is decreases slightly as capacity increases, which could have to do with data access taking up minor amounts of time. This graph stands out the most of the batch, but it makes sense that reducing latency would be among the strongest of contributors to more bandwidth.

C. 2D NVSim vs 2D Destiny vs 3D Destiny vs MLC Destiny [RRAM]

The section compares RRAM across all the simulators in 2D and 3D.

Fig. 17 shows Total Area optimized by Area with NVSim 2D coming out with the lowest results. Although that configuration progresses in proportion to capacity, Destiny 2D and 3D RRAM don't show any correlation with capacity. Destiny MLC RRAM appears to have correlation with the capacity but has the least favorable area results.

With a dual Read Latency result and Optimization Target, Fig. 18 shows Destiny MLC has by far the lowest latency and best results with linear correlation to capacity. The other three configurations are slower but don't appear to get worse as capacity increases.

Fig. 19 shows even better results for MLC RRAM with dual Write Latency settings. Again, the other three setups do not change despite capacity increases.

In Fig. 20 a dual Read Dynamic Energy result and Optimization Target show linear correlation with all of the tool and RRAM settings. Destiny 2D and 3D RRAM are the least competitive - consuming the most energy. Destiny MLC RRAM fares better, being beat out by NVSim 2D RRAM with only a small gap.

Fig. 21 provides Write Dynamic Energy as the y-axis with Leakage Power as the Optimization Target. MLC RRAM dominates the results here, but it is worth noting the dip at 256KB since the trend is not linear from 16KB to 4MB. Destiny 2D and 3D consume the most write energy and are relatively constant despite capacity; whereas NVSim 2D RRAM beats them out by a much smaller margin than MLC RRAM does.

In Fig. 22, Leakage Power is plotted vs Capacity with Area as the Optimization Target due to bigger gaps compared to other targets. NVSim 2D RRAM wins in terms of least Leakage Power here, while all configurations show higher leakage power with larger capacity. There is not a large difference between the three Destiny setups.

In summary here, MLC RRAM is very promising with Read and Write Dynamic Energy as well as Read and Write Latency. NVSim 2D RRAM secures wins in Leakage Power and Area metrics, which makes sense having seen its advantage in prior results.

D. 2D NVSim vs 2D Destiny vs 3D Destiny [RRAM]

Since Destiny MLC RRAM did not yield results for Read and Write Bandwidth, this section shows the other three configurations as compared to throughput.

Fig. 23 shows Read Bandwidth optimized by Read Latency performing best and evenly on Destiny 2D and 3D RRAM. NVSim trails the bandwidth and as capacity grows, Read Bandwidth decreases across the board.

In Fig. 24, Write Bandwidth optimized by Write Latency shows nearly identical results among NVsim 2D RRAM, Destiny 2D RRAM, and Destiny 3D RRAM. And as capacity increases, there is not a visible change of bandwidth - which I would expect to decrease based on similar graphs.

Limitations

Destiny 2D simulations on STTRAM, SRAM, RRAM, and PCRAM worked flawlessly. NVSim on the other hand, failed to generate data for PCRAM simulations. Looking into the source code of this tool, PCRAM had a note that it was currently unsupported in simulations. So in comparisons, Destiny and NVSim were able to go head to head with three RAM cell types: STTRAM, SRAM, and RRAM.

In the high-density category, the 3D RRAM simulations worked without issue. The MLC RRAM simulations did not work with Read Bandwidth and Write Bandwidth as negative and infinity values were displayed. That is possibly attributable to the cell configuration file being referenced by [3]. The file could have been outdated for Destiny version 2 even if it worked for the initial version, for example.

VI. Conclusions

A. 2D NVSim vs Destiny [STTRAM SRAM RRAM]

When comparing both tools in 2D, for cell types: STTRAM SRAM RRAM, here are some highlights. We learned that NVSim will generally have a leading result in Area and Leakage Power with the best of the three performing cells. With Latency and Bandwidth, the combination of Destiny and SRAM consistently beats out NVSim. NVSim and SRAM have the lowest Read and Write Dynamic Energy by a long shot. Although tool choice would probably come down to goal output, the advantage of SRAM over STTRAM and RRAM is very apparent.

B. 2D Destiny [STTRAM SRAM RRAM PCRAM]

Now taking PCRAM into account with Destiny, the categories of Area, Read Latency, and Read Bandwidth and won or nearly tied by PCRAM vs SRAM. Write Latency and Bandwidth and Dynamic Energy were not competitive for PCRAM, but SRAM won out by a large margin there. Depending on use case, either PCRAM and SRAM could be a solid choice - with an example of frequent reads but few writes selecting PCRAM as cell technology and vice versa for SRAM.

C. 2D NVSim vs 2D Destiny vs 3D Destiny vs MLC Destiny [RRAM]

With a four way RRAM comparison, NVSim 2D wins in Area, Leakage Power, and Read Dynamic Energy. The tool seems very area-optimized. Destiny MLC RRAM proves excellent in Read and Write Latency, and Write Dynamic Energy. Destiny 2D and 3D RRAM do not have best-in-class results but it was good to find that out since 3D RRAM does have high density like MLC RRAM - just didn't stack up with compelling results in these categories.

D. 2D NVSim vs 2D Destiny vs 3D Destiny [RRAM]

Excluding MLC RRAM due to lack of data, Destiny 2D and 3D both tie and lead for Read Bandwidth, while all three RRAM configurations are nearly equivalent for Write Bandwidth.

Taking a macro view on the results, neither tool was perfect, as NVSim PCRAM failed due to lack of code functionality, and Destiny MLC RRAM failed in Read and Write Bandwidth simulations. The added capabilities of Destiny such as including MLC and 3D cells show that it is useful and powerful for going beyond 2D simulations. NVSim still has a place though, as it optimizes Area and Leakage Power better than Destiny did with these datapoints. Ultimately, tool choice is simple if one is going to be using high-density cells - pick Destiny. But especially if optimal results in specific categories such as Latency or Area, the choice of tool is not obvious.

Future work could explore more cell types on Destiny, or write code for NVSim to enable PCRAM. Debugging could take place to dive into the root cause of why MLC RRAM didn't produce good data on Read and Write Bandwidth (on process technology ranging from 22nm to 90nm). More charts could be generated for comparisons as well given the spreadsheets [4] if users are curious about more head-to-head visualizations. And any tools on the horizon that appear could be ran through similar simulations to understand advantages and shortcomings.

Acknowledgment

I would like to thank Professor Donato, of the Tufts University Electrical and Computer Engineering Department, for advising me on this project and providing iterative feedback throughout each checkpoint. Professor Donato guided my initial proposal to compare NVSim and Destiny 2D technologies as well as explore high density capabilities of Destiny such as 3D and MLC RRAM.

References

[1] X. Dong, C. Xu, Y. Xie and N. P. Jouppi, ”NVSim: A Circuit-Level Performance, Energy, and Area Model for Emerging Nonvolatile Memory,” in IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 31, no. 7, pp. 994-1007, July 2012, doi: 10.1109/TCAD.2012.2185930.

[2] M. Poremba, S. Mittal, D. Li, J. S. Vetter and Y. Xie, ”DESTINY: A tool for modeling emerging 3D NVM and eDRAM caches,” 2015 Design, Automation Test in Europe Conference Exhibition (DATE), 2015, pp. 1543-1546, doi: 10.7873/DATE.2015.0733.

[3] Seal-Ucsb, “Seal-UCSB/nvsim: Nvsim - a performance, energy and area estimation tool for non-volatile memory (NVM),” GitHub. [Online]. Available: https://github.com/SEAL-UCSB/NVSim. [Accessed: 31-Dec-2021].

[4] Rockett-M, “EECE0193/FINAL_PROJECT/experiments at main · Rockett-M/EECE0193,” GitHub. [Online]. Available: https://github.com/rockett-m/EECE0193/tree/main/Final_Project/EXPERIMENTS. [Accessed: 31-Dec-2021].

[5] “JLPEA — Free Full-text — destiny: A comprehensive tool ...” [Online]. Available: https://www.mdpi.com/2079-9268/7/3/23. [Accessed: 31-Dec-2021].

VII. Appendix

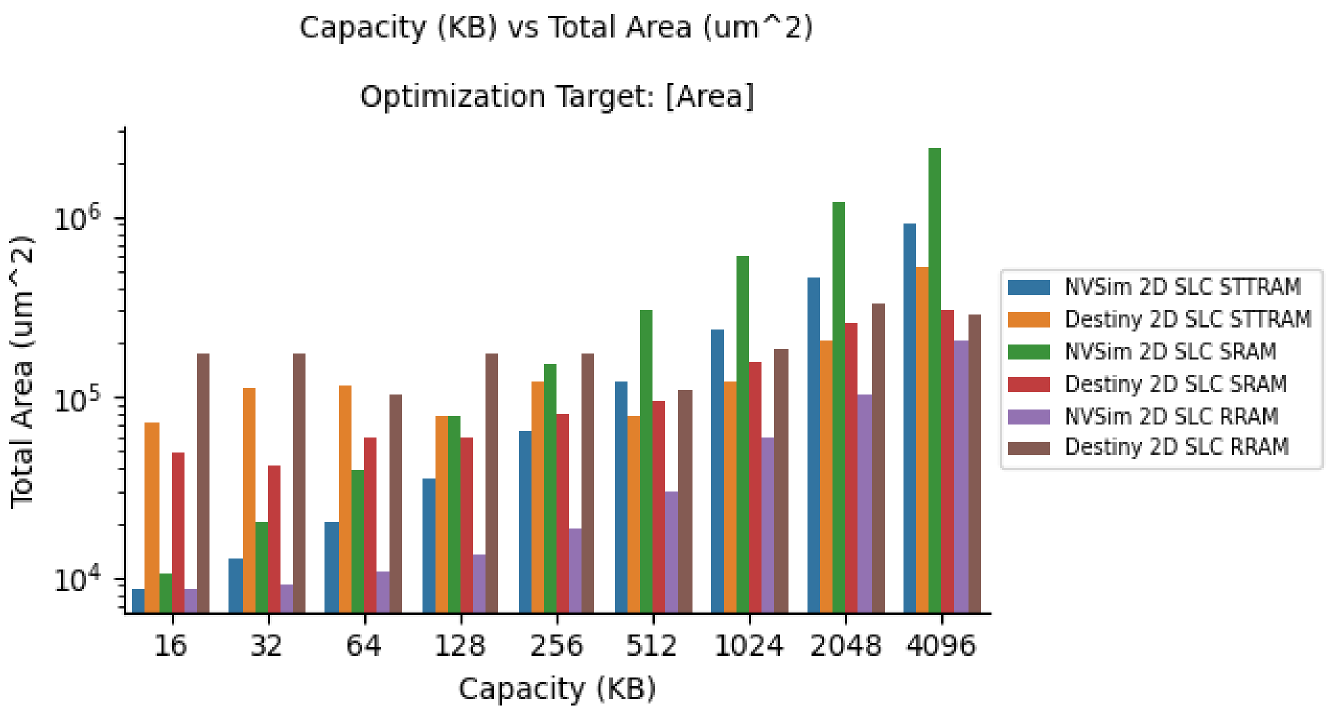

Capacity (KB) vs Total Area (um^2)
Optimization Target: [Area]
Total Area (um^2)
10^6
10^5
10^4
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 1.

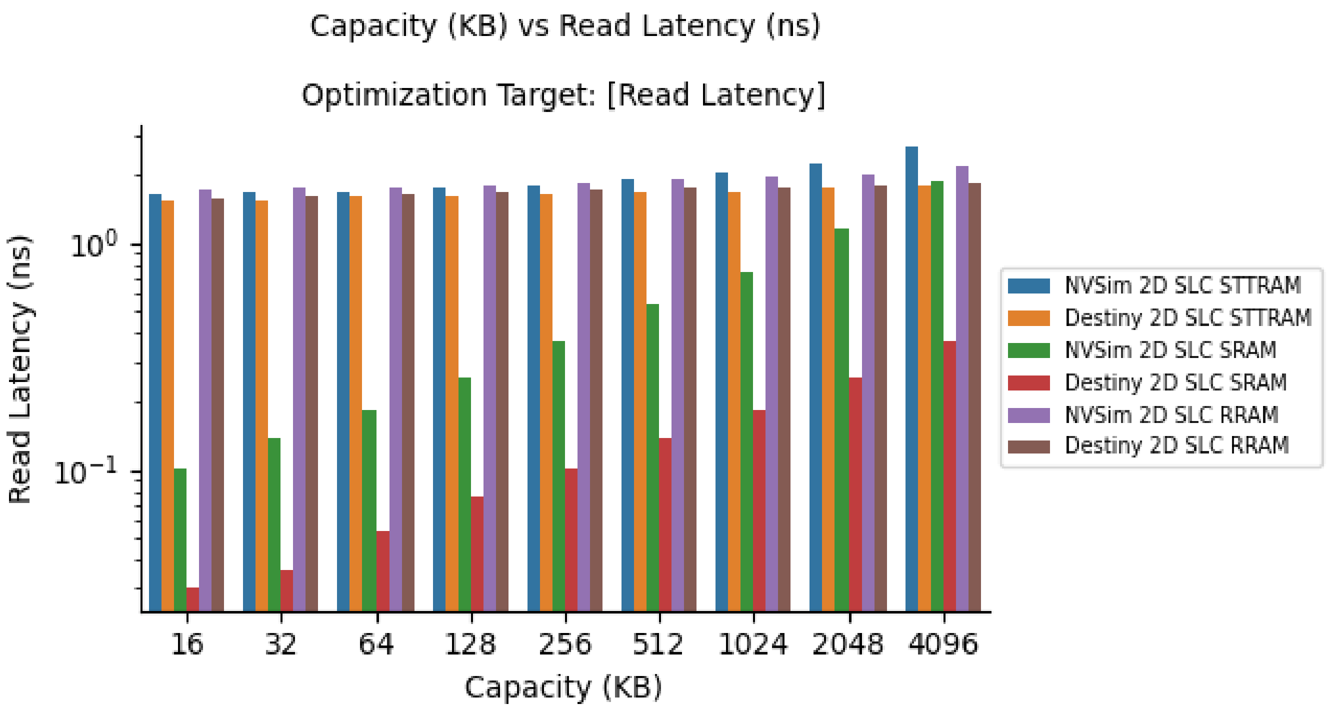

Capacity (KB) vs Read Latency (ns)
Optimization Target: [Read Latency]
Read Latency (ns)
10^0
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 2.

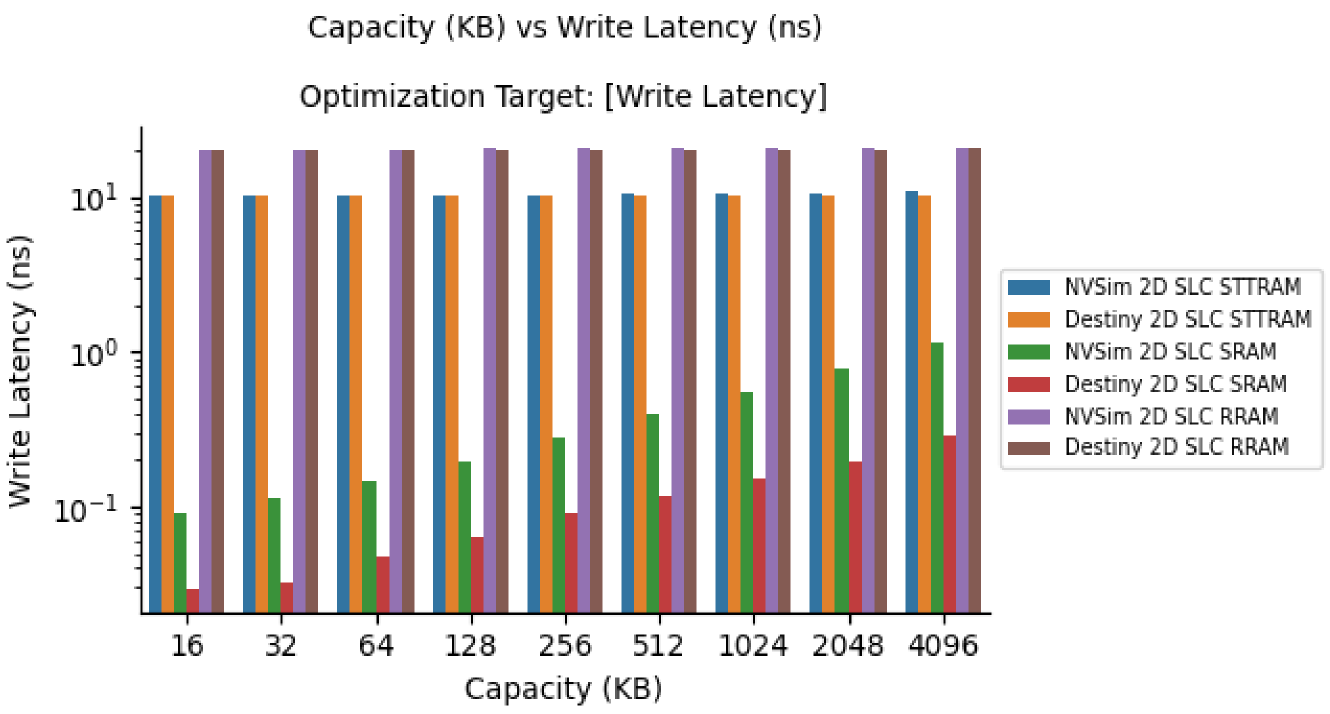

Capacity (KB) vs Write Latency (ns)
Optimization Target: [Write Latency]
Write Latency (ns)
10^{1}
10^{0}
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 3.

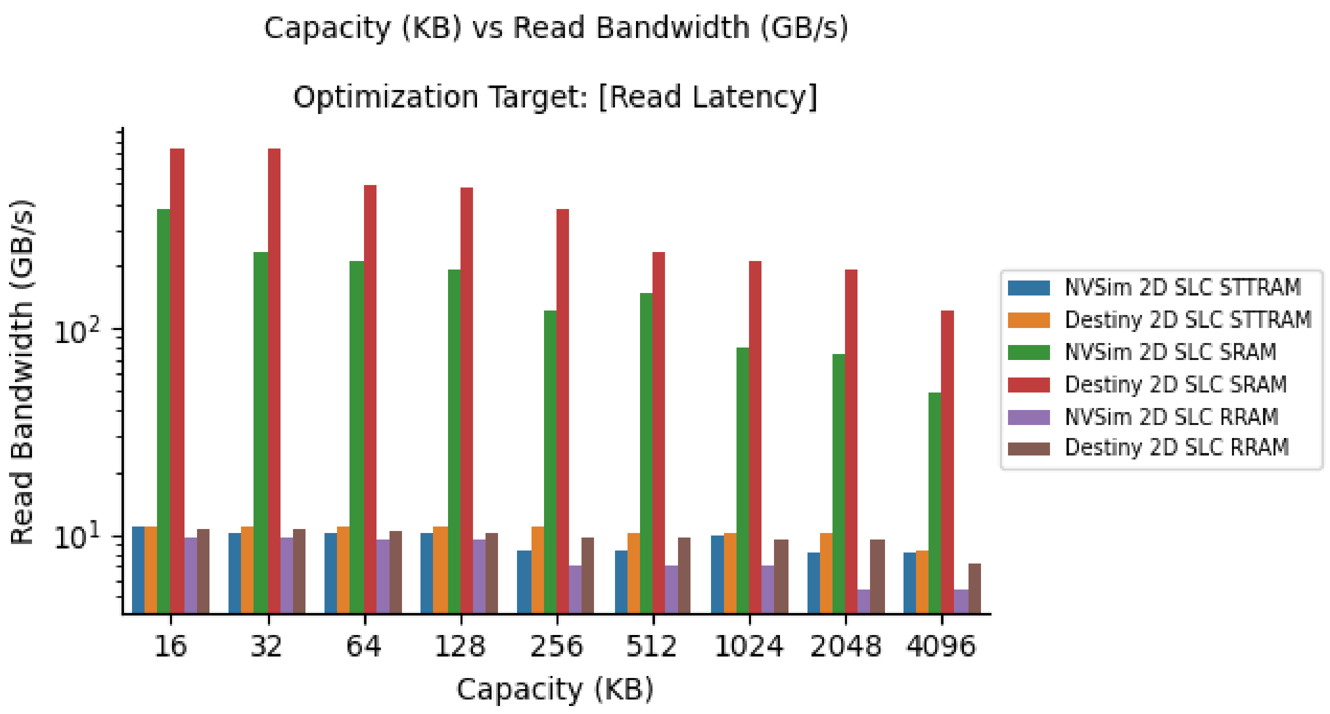

Capacity (KB) vs Read Bandwidth (GB/s)
Optimization Target: [Read Latency]
Read Bandwidth (GB/s)
10^{2}
10^{1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 4.

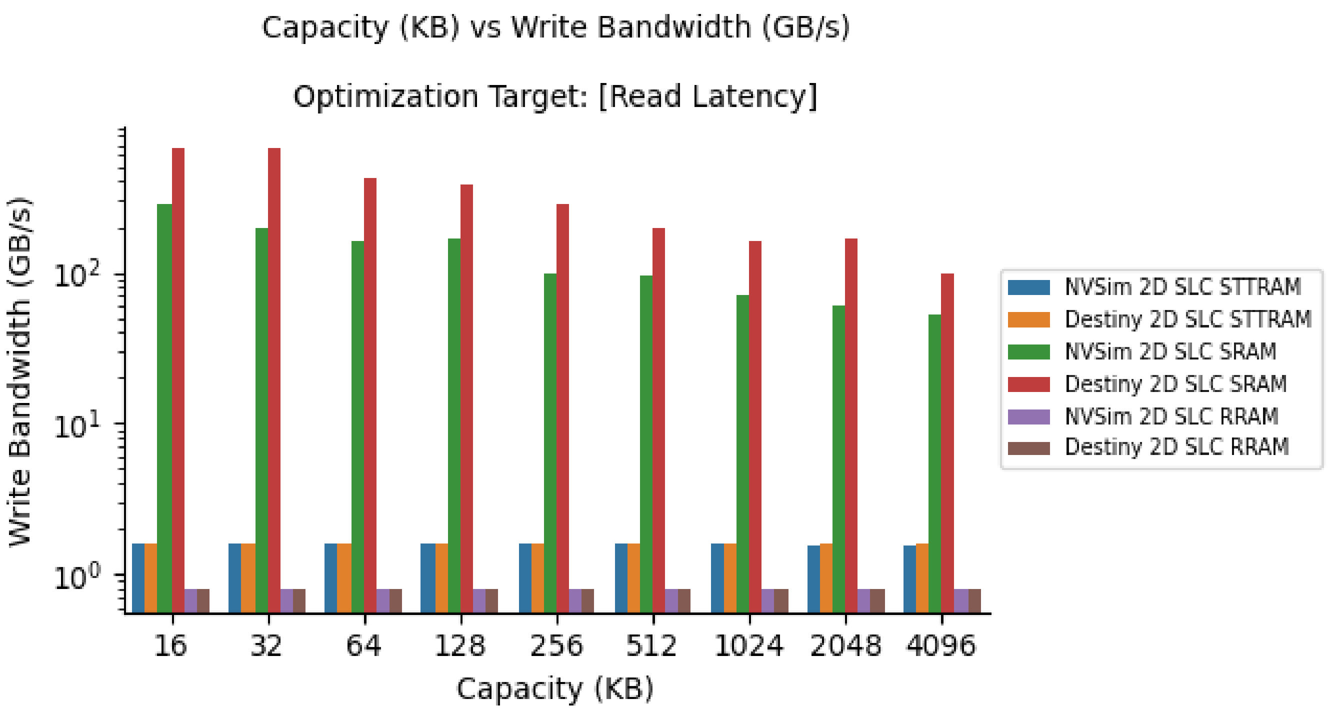

Capacity (KB) vs Write Bandwidth (GB/s)
Optimization Target: [Read Latency]
Write Bandwidth (GB/s)
10^0
10^1
10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 5.

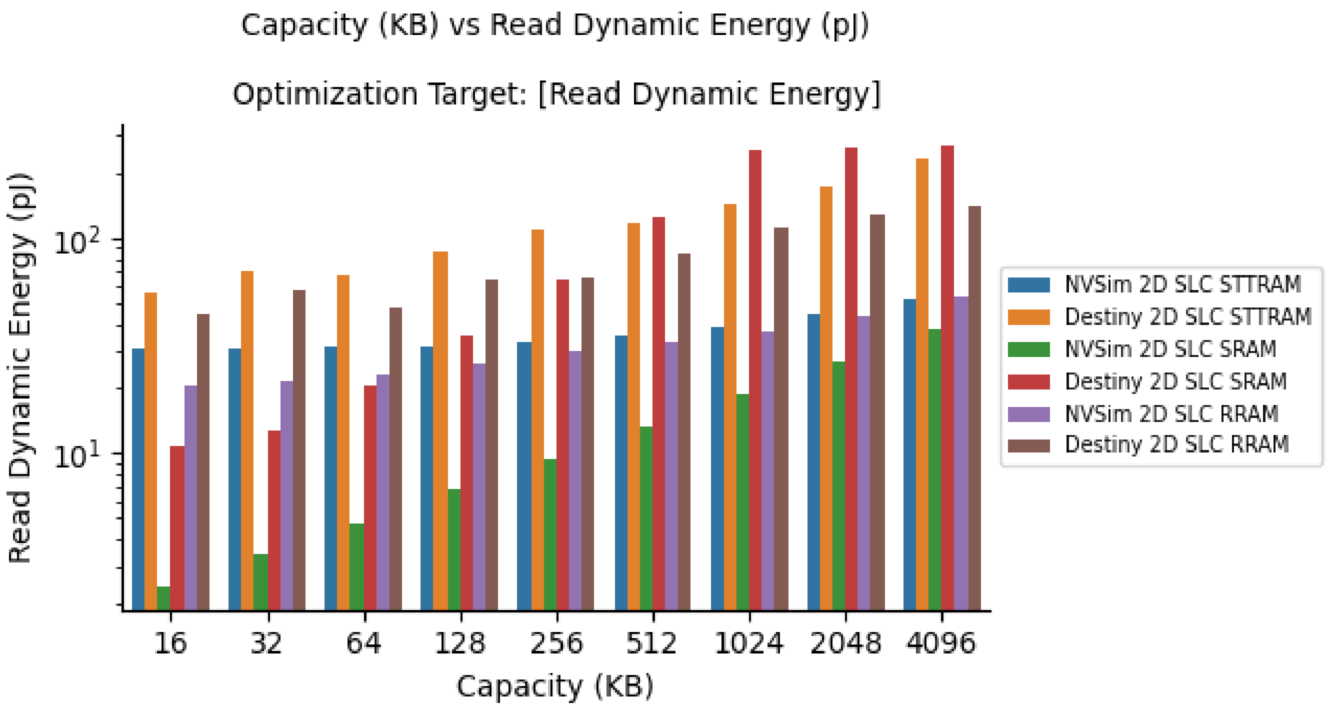

Capacity (KB) vs Read Dynamic Energy (pJ)
Optimization Target: [Read Dynamic Energy]
Read Dynamic Energy (pJ)
10^1
10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 6.

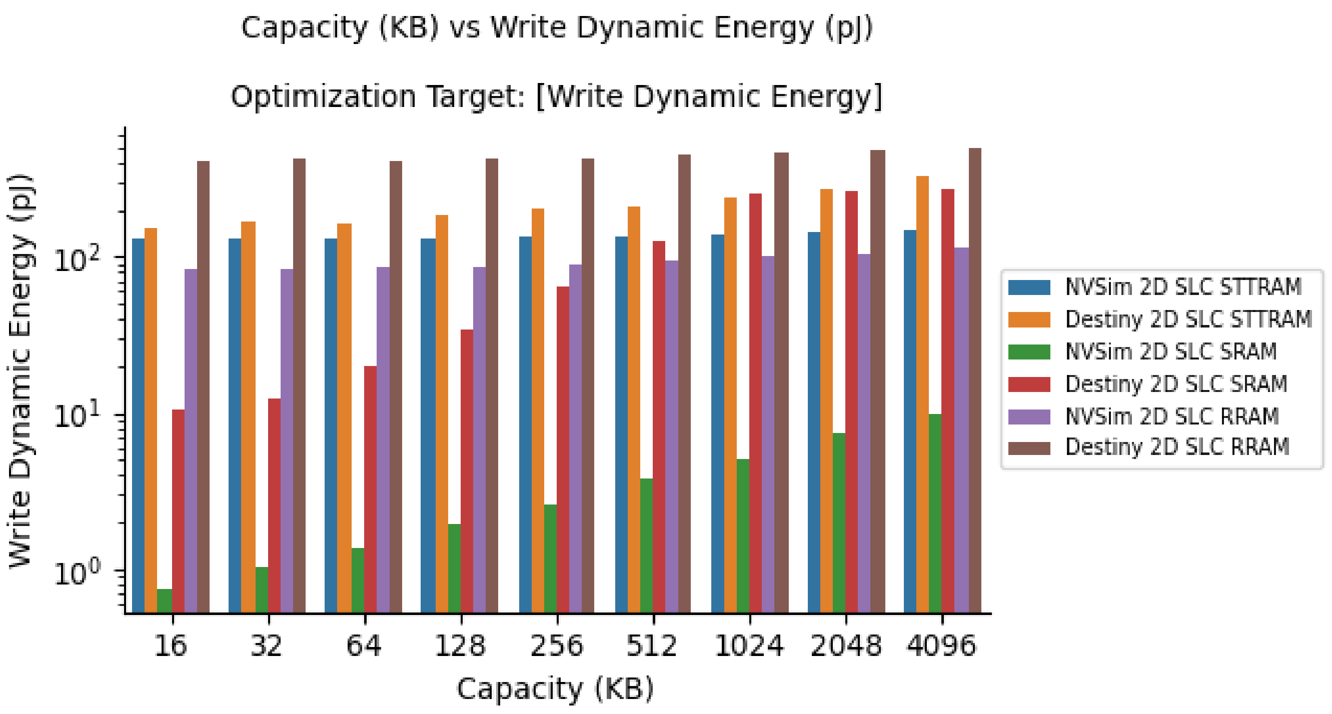

Capacity (KB) vs Write Dynamic Energy (pJ)
Optimization Target: [Write Dynamic Energy]
Write Dynamic Energy (pJ)
10^0
10^1
10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 7.

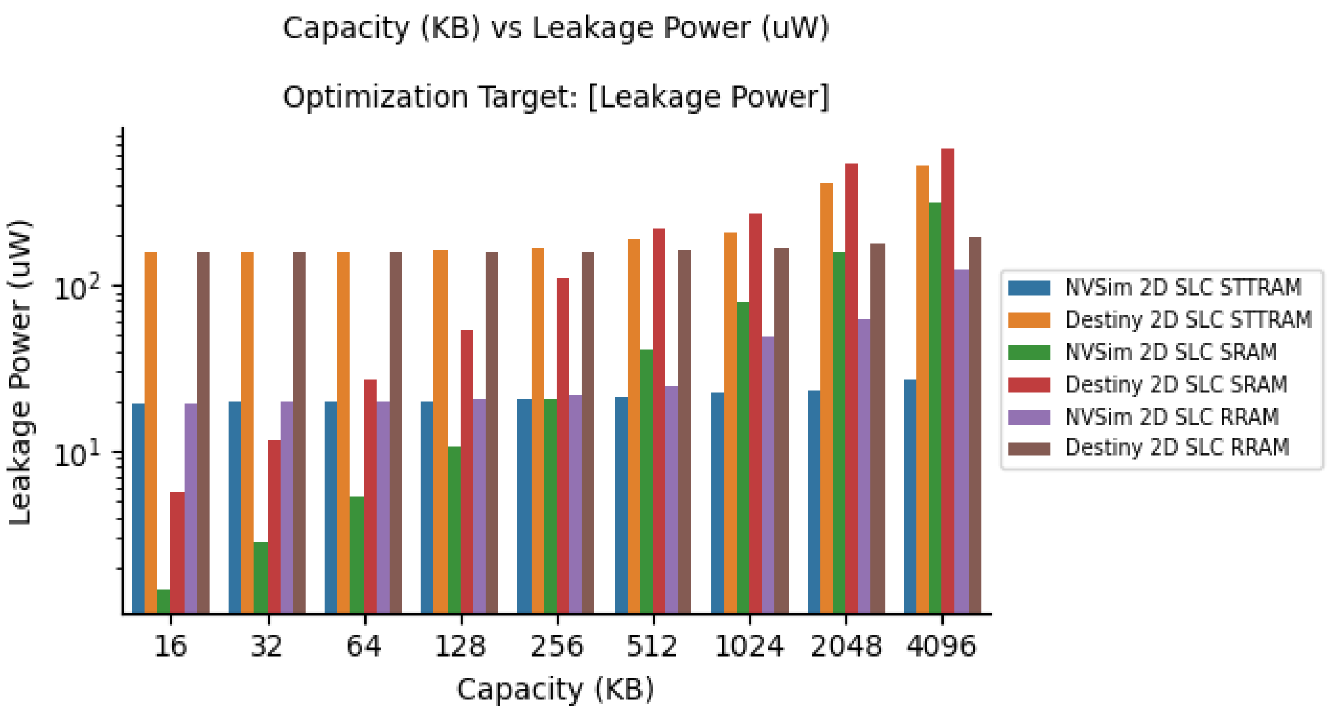

Capacity (KB) vs Leakage Power (uW)
Optimization Target: [Leakage Power]
Leakage Power (uW)
10^1
10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC STTRAM
Destiny 2D SLC STTRAM
NVSim 2D SLC SRAM
Destiny 2D SLC SRAM
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM

Fig. 8.

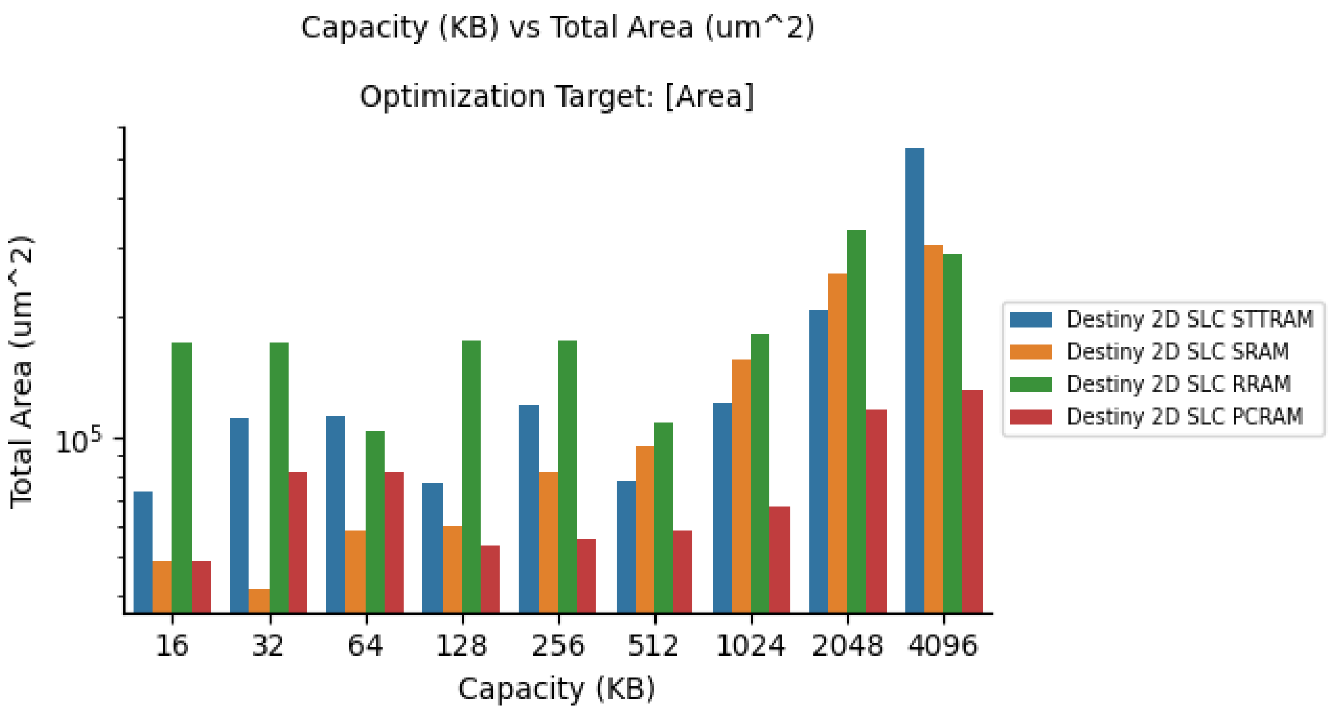

Capacity (KB) vs Total Area (um^2)
Optimization Target: [Area]
Total Area (um^2)
10^5
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 9.

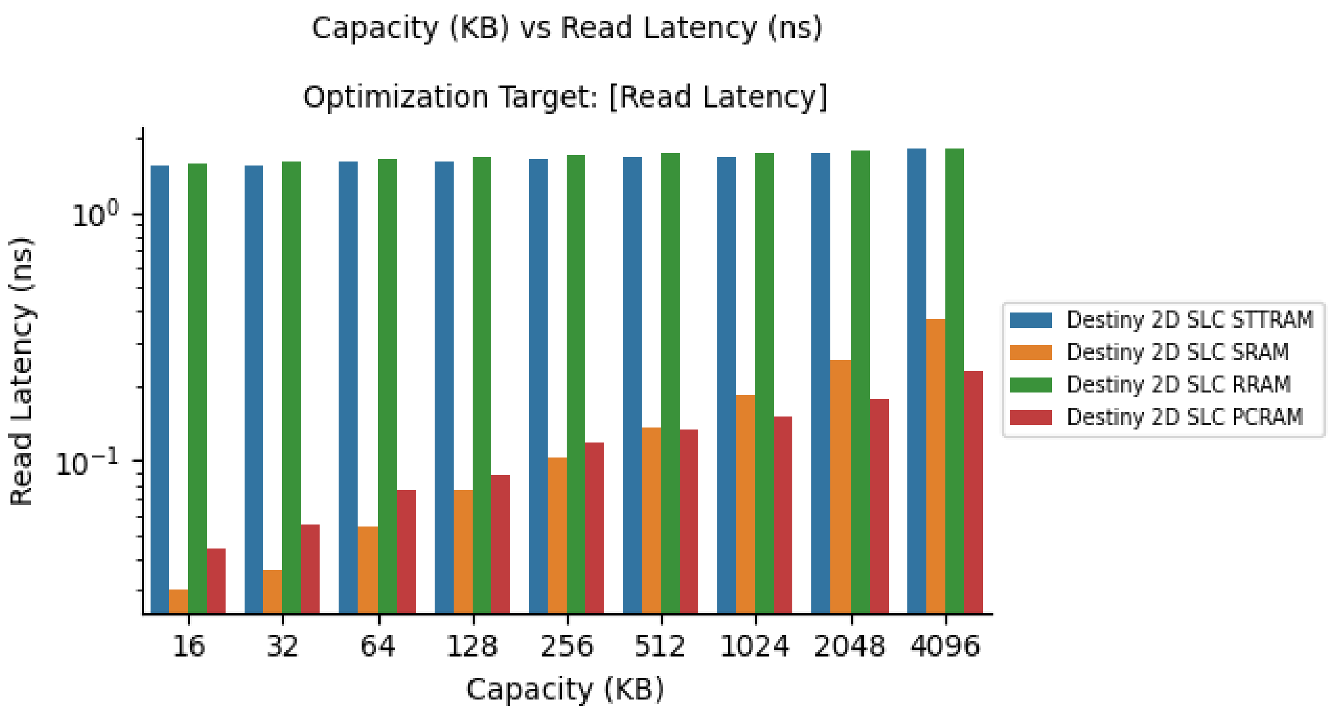

Capacity (KB) vs Read Latency (ns)
Optimization Target: [Read Latency]
Read Latency (ns)
10^0
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 10.

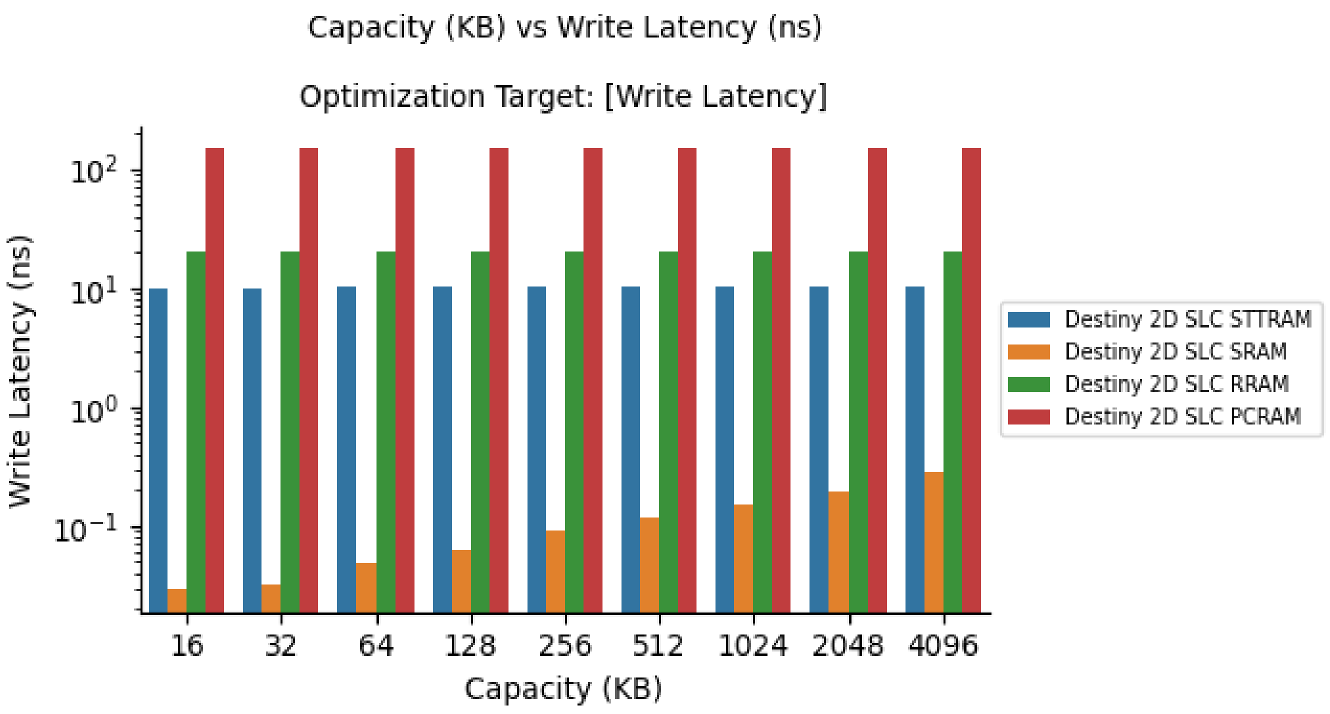

Capacity (KB) vs Write Latency (ns)
Optimization Target: [Write Latency]
Write Latency (ns)
10^2
10^1
10^0
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 11.

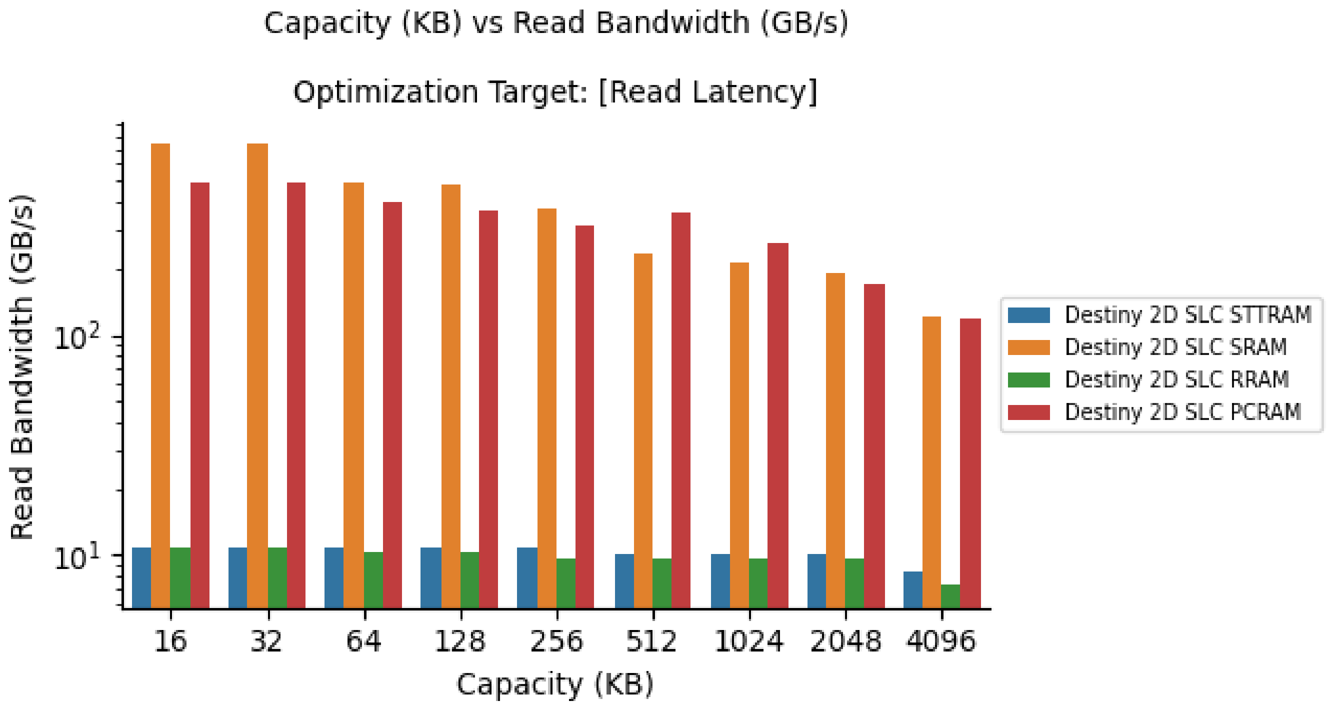

Capacity (KB) vs Read Bandwidth (GB/s)
Optimization Target: [Read Latency]
Read Bandwidth (GB/s)
10^2
10^1
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 12.

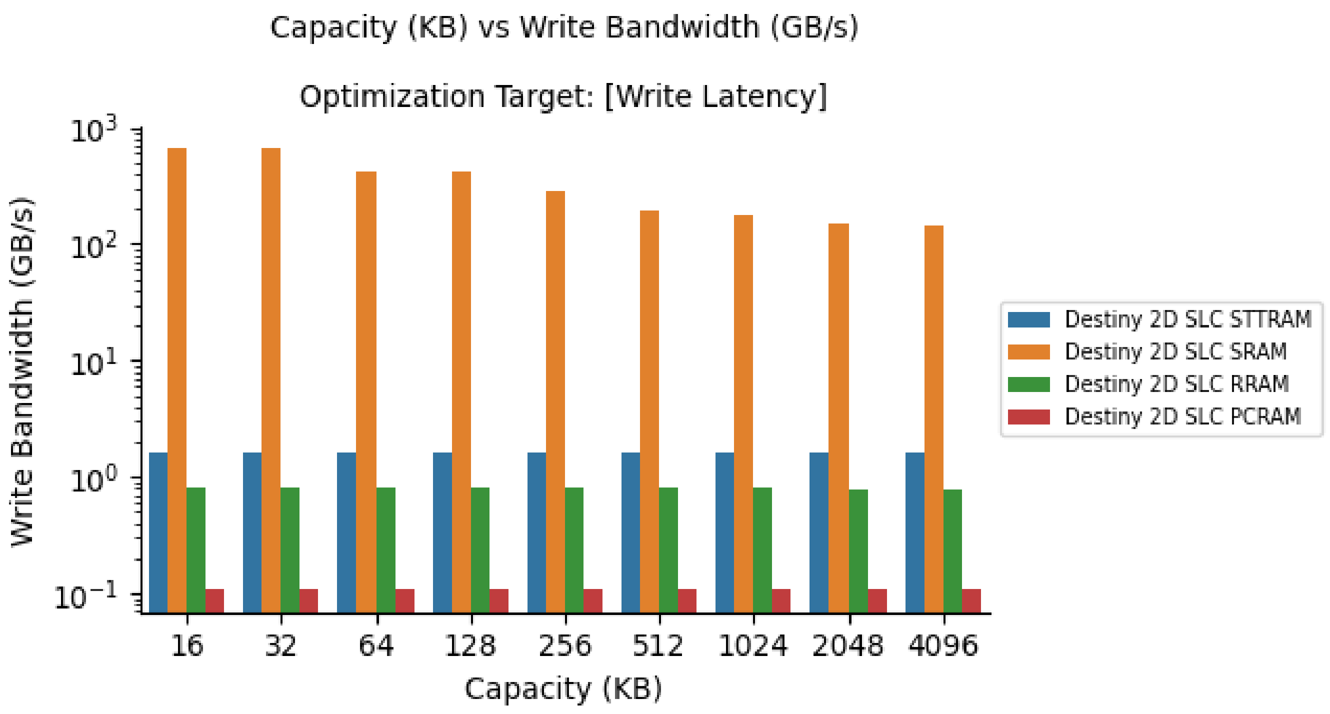

Capacity (KB) vs Write Bandwidth (GB/s)
Optimization Target: [Write Latency]
Write Bandwidth (GB/s)
10^3
10^2
10^1
10^0
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 13.

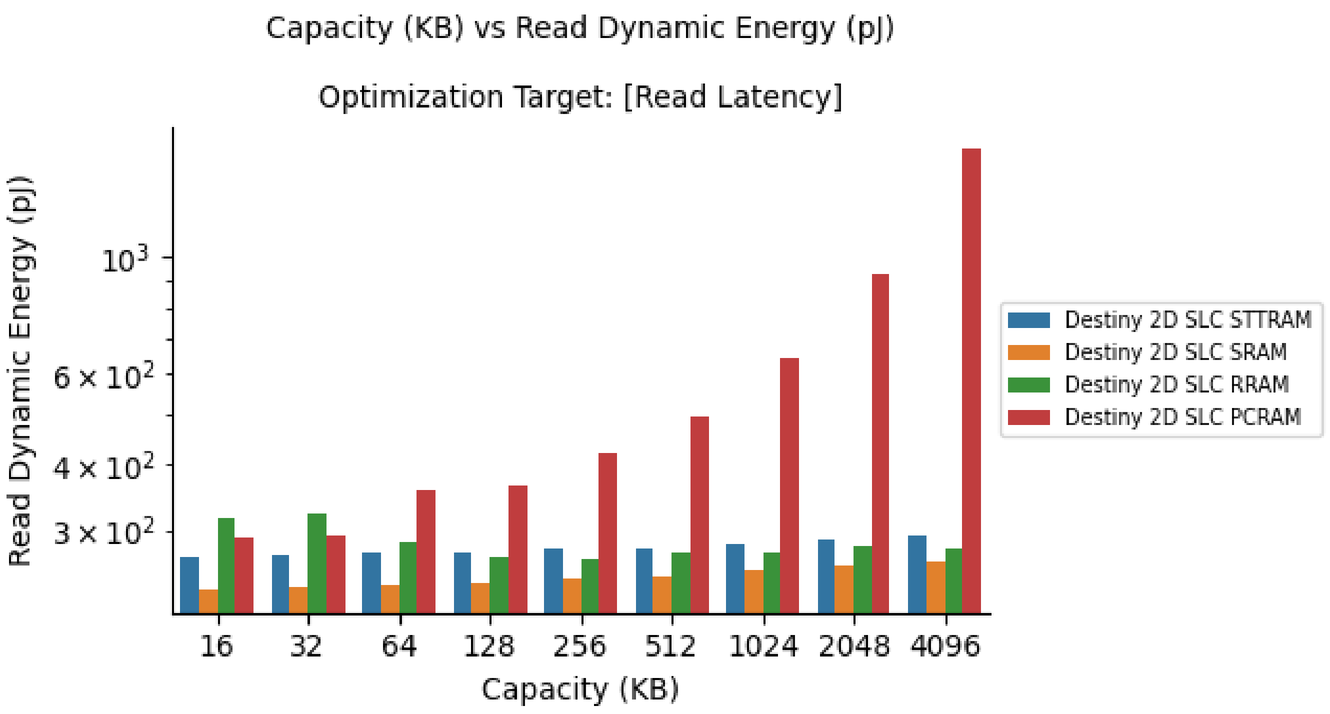

Capacity (KB) vs Read Dynamic Energy (pJ)
Optimization Target: [Read Latency]
Read Dynamic Energy (pJ)
10^3
6×10^2
4×10^2
3×10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 14.

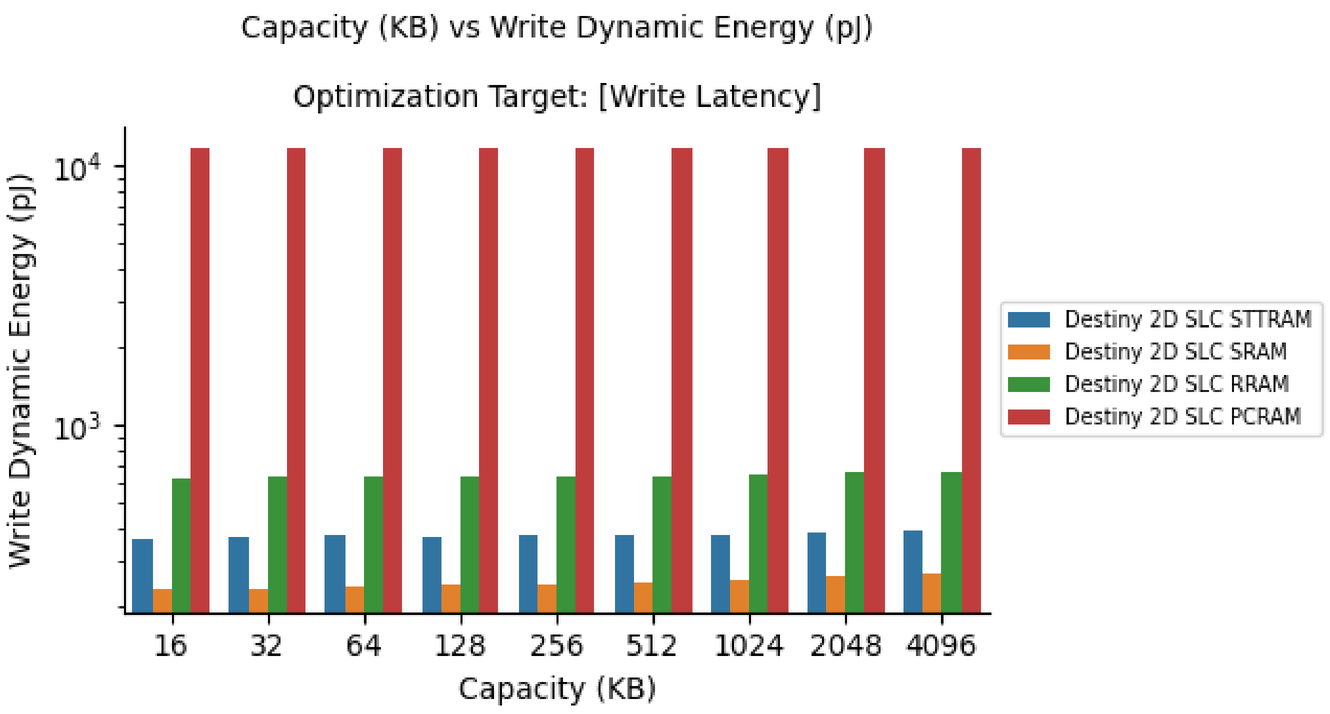

Capacity (KB) vs Write Dynamic Energy (pJ)
Optimization Target: [Write Latency]
Write Dynamic Energy (pJ)
10^4
10^3
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 15.

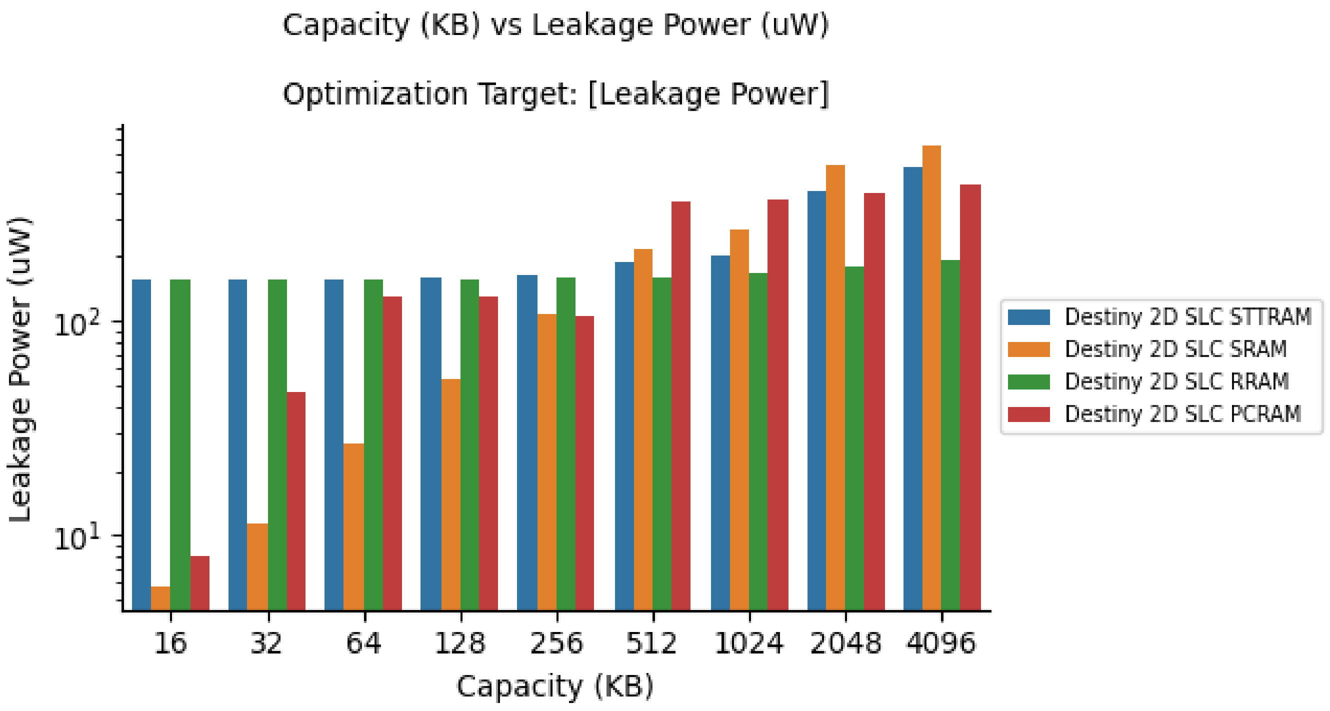
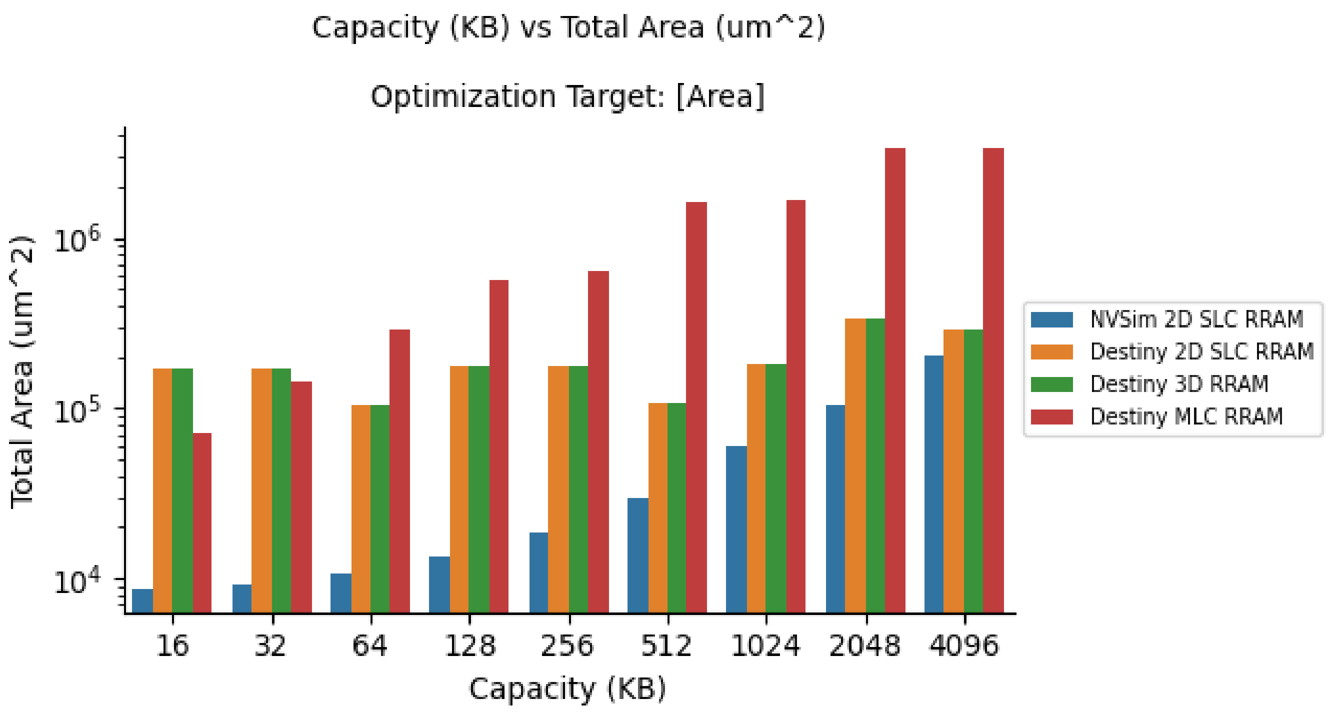
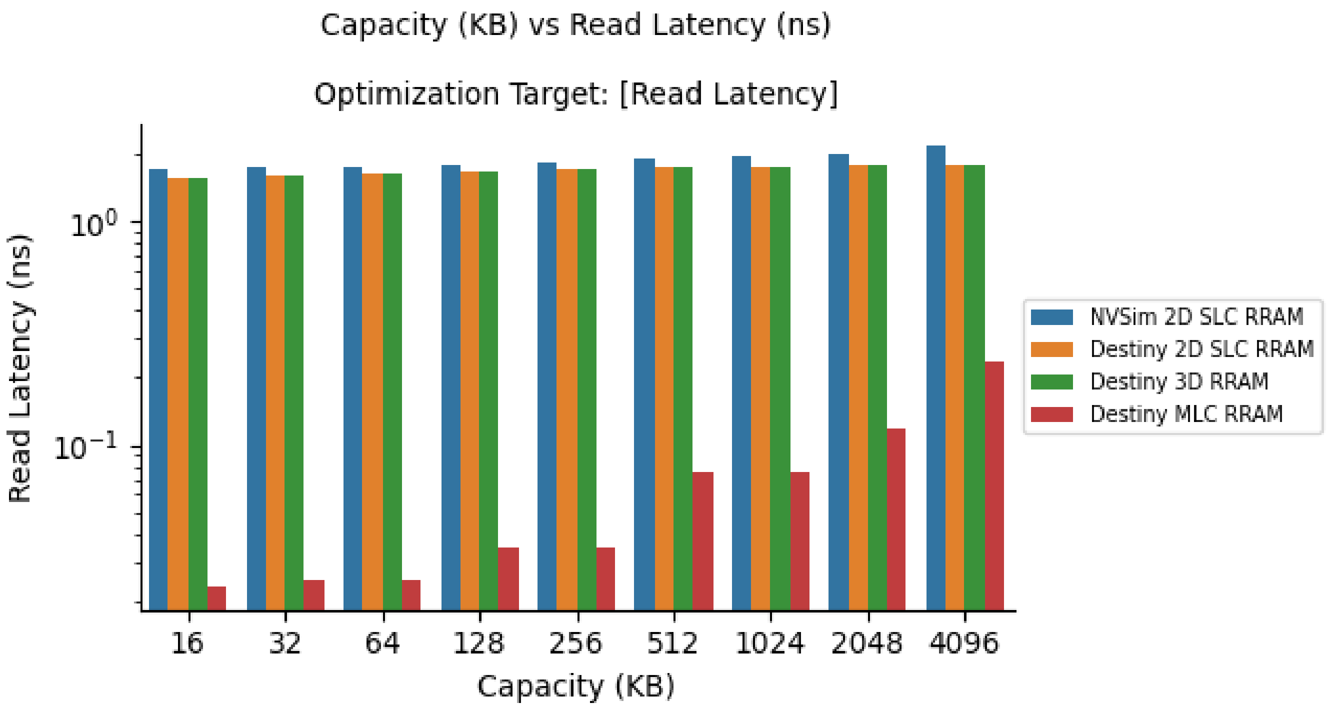

Capacity (KB) vs Leakage Power (uW)
Optimization Target: [Leakage Power]
Leakage Power (uW)
10^2
10^1
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
Destiny 2D SLC STTRAM
Destiny 2D SLC SRAM
Destiny 2D SLC RRAM
Destiny 2D SLC PCRAM

Fig. 16.



Capacity (KB) vs Total Area (um^2)
Optimization Target: [Area]
Total Area (um^2)
10^4
10^5
10^6
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 17.



Capacity (KB) vs Read Latency (ns)
Optimization Target: [Read Latency]
Read Latency (ns)
10^{-1}
10^0
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 18.

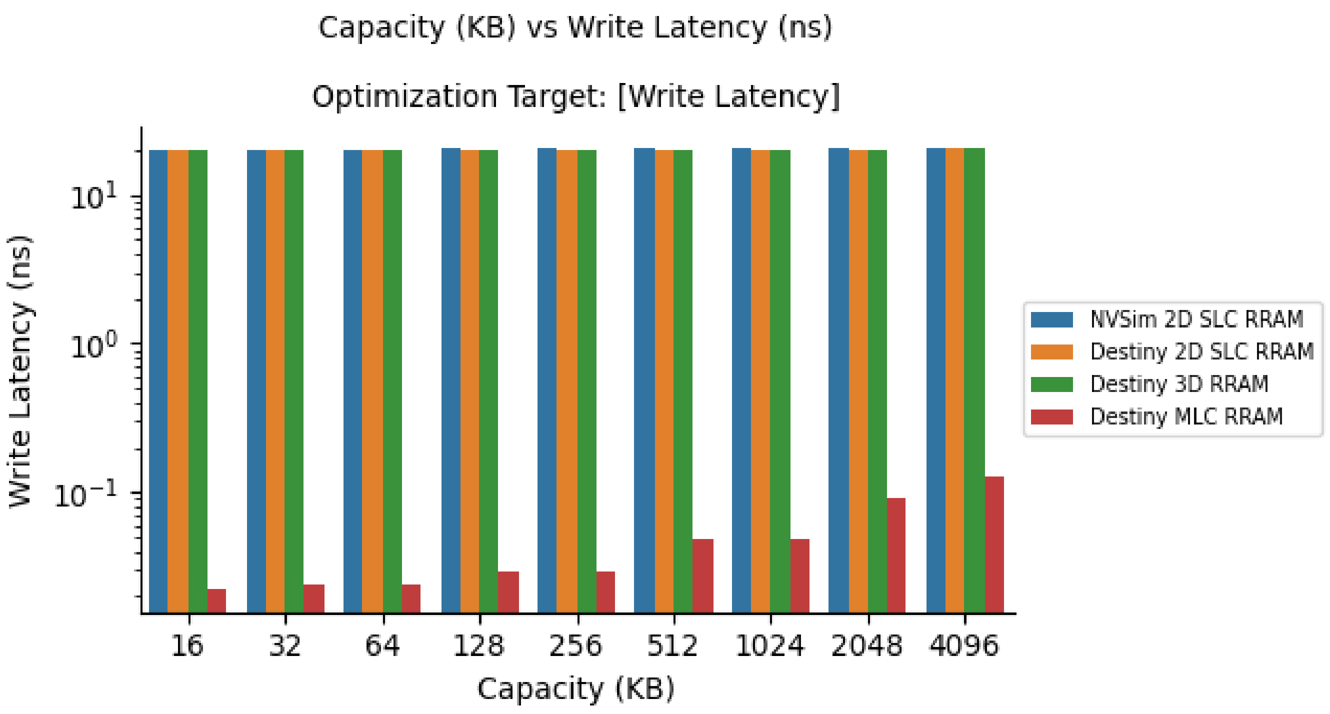

Capacity (KB) vs Write Latency (ns)
Optimization Target: [Write Latency]
Write Latency (ns)
10^1
10^0
10^{-1}
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 19.

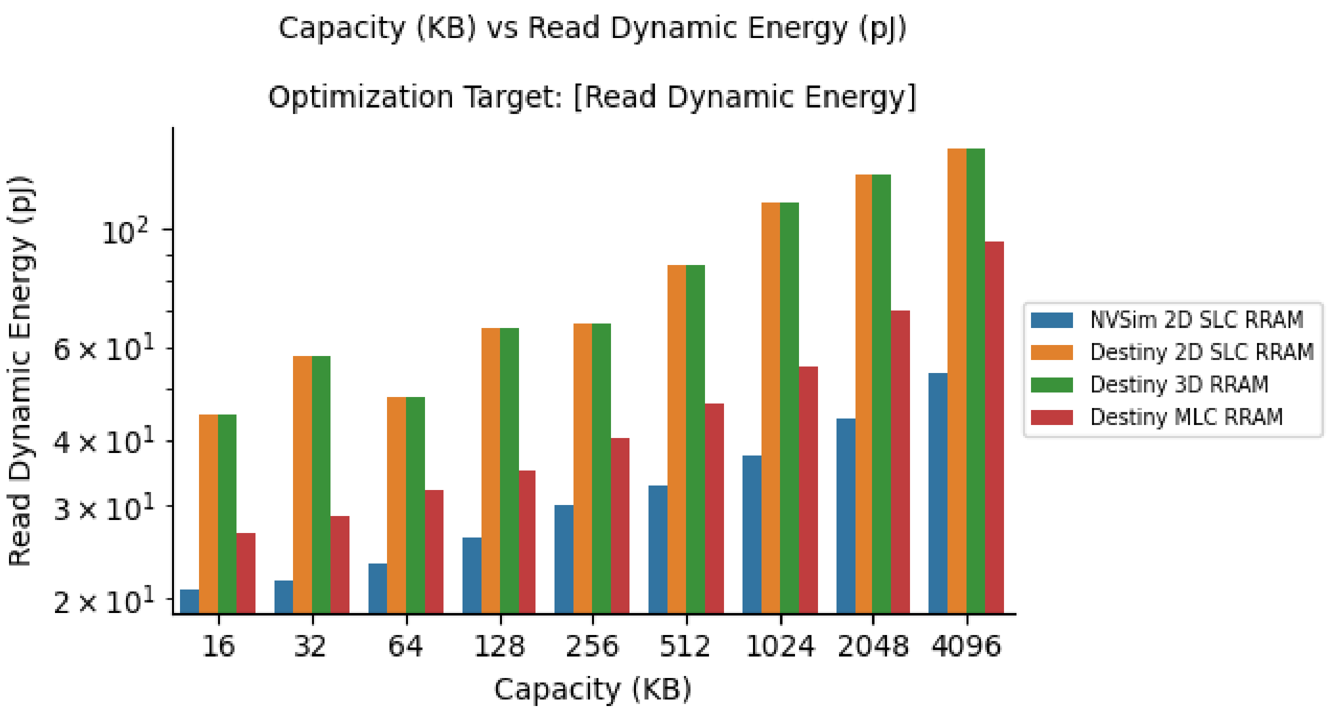

Capacity (KB) vs Read Dynamic Energy (pJ)
Optimization Target: [Read Dynamic Energy]
Read Dynamic Energy (pJ)
10^2
6×10^1
4×10^1
3×10^1
2×10^1
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 20.

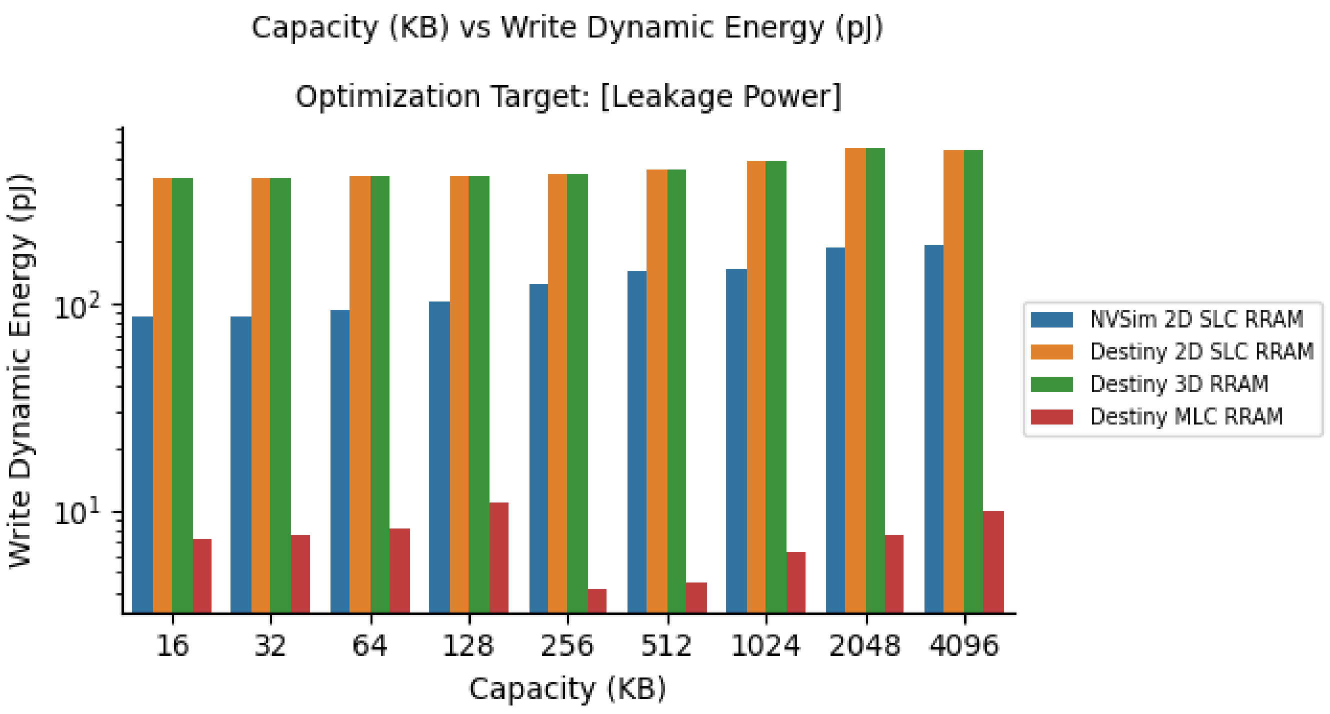

Capacity (KB) vs Write Dynamic Energy (pJ)
Optimization Target: [Leakage Power]
Write Dynamic Energy (pJ)
10^2
10^1
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 21.

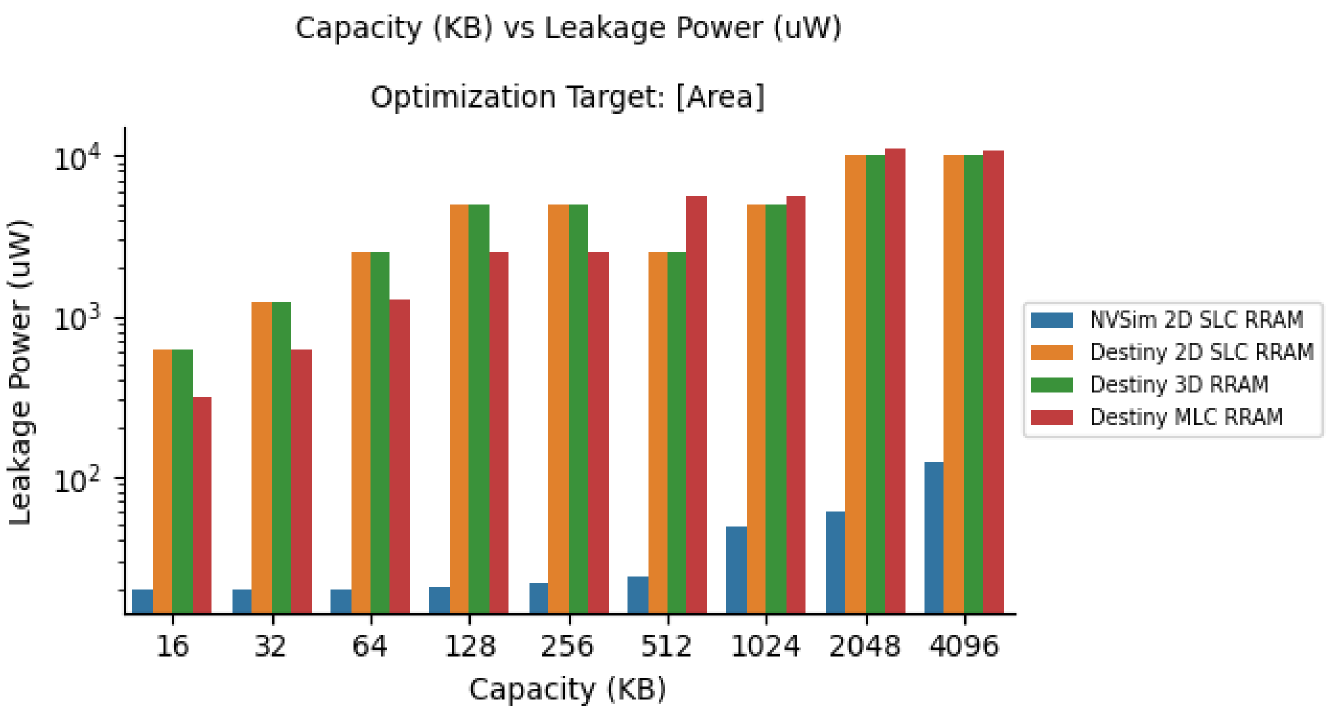

Capacity (KB) vs Leakage Power (uW)
Optimization Target: [Area]
Leakage Power (uW)
10^4
10^3
10^2
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM
Destiny MLC RRAM

Fig. 22.

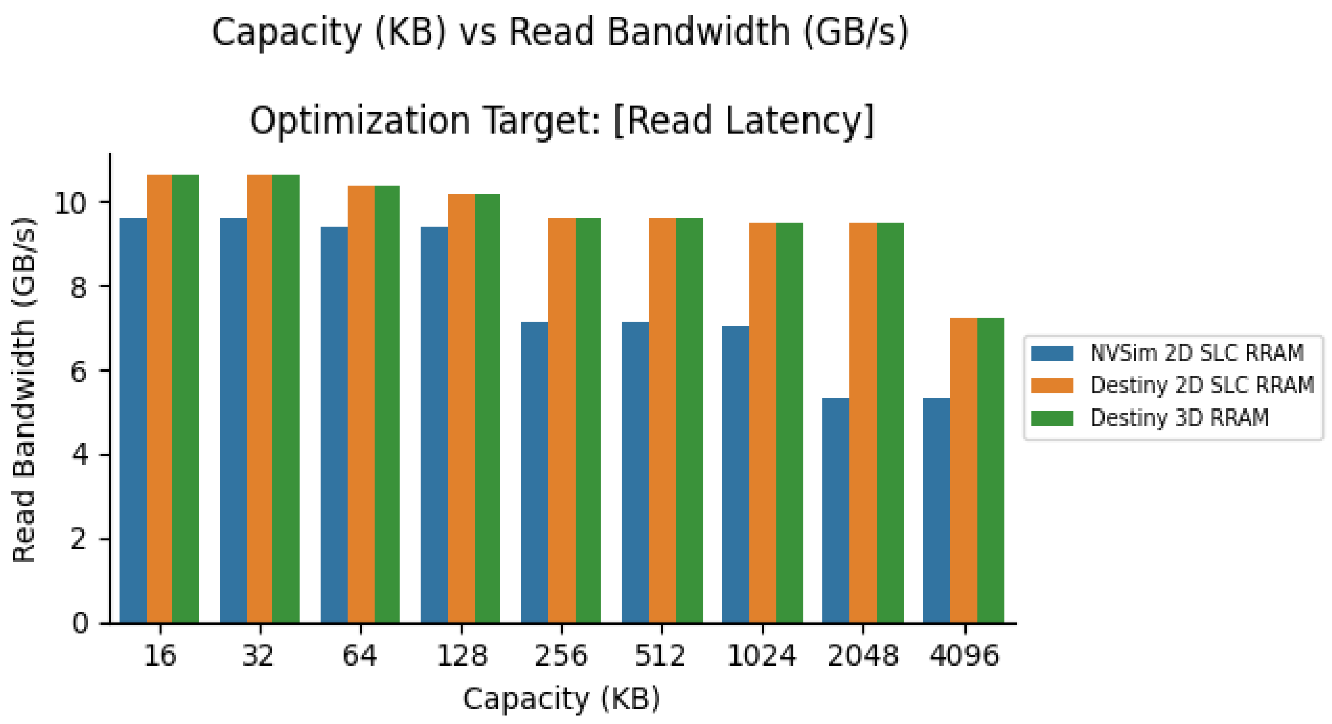

Capacity (KB) vs Read Bandwidth (GB/s)
Optimization Target: [Read Latency]
Read Bandwidth (GB/s)
0
2
4
6
8
10
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM

Fig. 23.

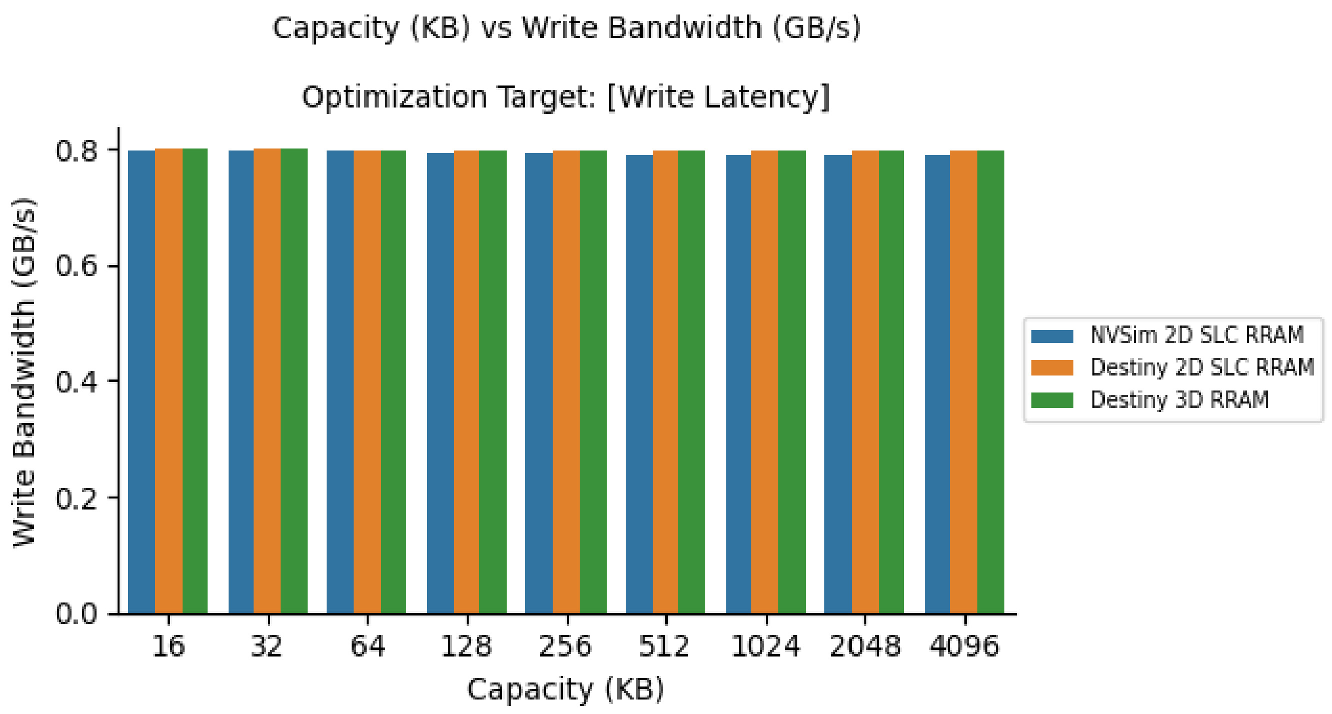

Capacity (KB) vs Write Bandwidth (GB/s)
Optimization Target: [Write Latency]
Write Bandwidth (GB/s)
0.0
0.2
0.4
0.6
0.8
16
32
64
128
256
512
1024
2048
4096
Capacity (KB)
NVSim 2D SLC RRAM
Destiny 2D SLC RRAM
Destiny 3D RRAM

Fig. 24.