



## CS561 Spring 2026 - Research Project

**Title:** *Flexible Zoned Namespace Interface*

**Background:** Zoned Namespace (ZNS) SSDs offer a promising interface for stable throughout and low-latency storage by eliminating device-side garbage collection [3,4]. ZNS exposes storage as a collection of fixed-size zones that have sequential write requirement. Due to hardware resource constraints the number of open zones is limited (between 8-32) [2]. Once the limit is reached, the host application can run the **zone FINISH** command to lock the zone for the writes and indicate to the controller to release the write resources for that zone. From the application perspective the write pointer simply moves to the end of the zone [1]. However, on the controller side the leftover empty blocks in the zone may be filled with dummy data or with more flexible mapping approach the empty blocks may be released and allocated to the next logical zone [2]. If the host frequently finishes the partially filled zones, the system may reach a state where all zones are finished, and no new zone can be allocated. However, in flexible mapping approaches there may still be empty physical space remaining on the device. Since the host assumes a static number of zones and the address space is fixed, after the **zone FINISH** command, the host cannot write to the remaining blocks in the zone, preventing the application from using the leftover physical space. This limitation highlights the need for a more flexible ZNS standard where the number of zones can vary, where new logical zones can be constructed from the remaining empty space, or the size of the logical zones can be flexible [2].

**Objective:** The objective of the project is to design and evaluate a flexible ZNS interface where the number of logical zones can vary or the size of logical zones is not fixed to make full use of the available hardware resources.

### Steps:

- Explore the ZNS SSD architecture and configuration options through the ConfZNS++ emulator
- Run initial experiments with Rocksdb showing how much empty space is inaccessible due to the fixed zoned interface
- Design and implement in the emulator flexible interface with varying number of zones or varying zone sizes
- Evaluate the new flexible interface with raw device benchmarks with FIO and application like Rocksdb

**Responsible mentor:** *Teona Bagashvili*

### References:

- [1] Doekemeijer, Krijn, et al. "Exploring I/O Management Performance in ZNS with ConfZNS++." *Proceedings of the 17th ACM International Systems and Storage Conference*. 2024.
- [2] Bagashvili, Teona, et al. "Eliminating the Hidden Cost of Zone Management in ZNS SSDs." *arXiv preprint arXiv:2511.04687* (2025).



[3] Song, Inho, et al. "Confzns: A novel emulator for exploring design space of zns ssds." *Proceedings of the 16th ACM International Conference on Systems and Storage*. 2023.

[4] Bjørling, Matias, et al. "{ZNS}: Avoiding the block interface tax for flash-based {SSDs}." *2021 USENIX annual technical conference (USENIX ATC 21)*. 2021.