Software-Defined Emulation Infrastructure for High Speed Storage

Krishna T. Malladi Manu Awasthi Hongzhong Zheng

Samsung Semiconductor, Inc

{k.tej, manu.awasthi, hz.zheng}@ssi.samsung.com

Categories and Subject Descriptors B.3 [*Memory Structures*]: Performance Analysis and Design Aids; C.4 [*Performance of Systems*]: Measurement techniques

Keywords Storage systems, Emulation, NVMe, SSD

Abstract

NVMe, being a new I/O communication protocol, suffers from a lack of tools to evaluate storage solutions built on the standard. In this paper, we provide the design and analysis of a comprehensive, fully customizable emulation infrastructure that builds on the NVMe protocol. It provides a number of knobs that allow system architects to quickly evaluate performance implications of a wide variety of storage solutions while natively executing workloads.

1. Introduction and Motivation

The arena of high performance storage solutions is rapidly evolving with innovations across the stack, including communication interfaces and new device technologies. Contemporary SSDs are Flash based, while memory technologies like PCM, STTRAM are poised to provide even higher performance. Non volatile memory express (NVMe) protocol was invented to provide a technology-agnostic I/O communication mechanism. This protocol has become the industry standard, and will be the *de facto* interface for future SSDs.

However, being a relatively new protocol, NVMe suffers from a lack of tools to evaluate various storage configurations. To classify the benefits of a variety of current and future NVMe SSDs while running applications natively, architects need reliable, scalable infrastructure that can *emulate* and analyze different storage options [1, 2].

2. Infrastructure Design

In order to design an NVMe based emulation platform, we first identify the five main parameters necessary to model storage systems. These are : random read/write latencies, sequential read/write bandwidths, random IOPS and interface bandwidth. Providing explicit latency knobs helps account for asymmetric read/write latencies in storage technologies. Also, knobs for bandwidths and IOPS help emulate devices with varied characteristics. Finally, interface bandwidth tunability helps study a wide range of interface considerations.

To build these characteristics into an emulator, we implement the knobs in the NVMe device driver. The driver resides directly above the hardware, thereby minimizing request reordering in the upper layers. Device drivers also provide ease of use across platforms with different stacks. Moreover, based on an actual SSD, it also captures the effects of FTL algorithms and other SSD functionalities like garbage collection and wear leveling. These knobs that specify drive characteristics are controlled through the `/sys/fs` interface, allowing software-defined runtime configurability.

Next, we extend our infrastructure to evaluate the projected benefits of faster, future SSDs. These devices have smaller latencies that cannot be emulated with existing SSDs. So, we apply our methodology to a RAMdisk to obtain optimistic latencies and then use analytical interpolation to project their performance. We validate our emulation infrastructure with FIO and obtain near perfect correlation.

3. Conclusion

We present a high fidelity, software-defined emulation infrastructure for NVMe devices. It allows system designers to quickly iterate over storage configurations, while executing workloads natively. Furthermore, an analytical model projects the performance of future storage systems.

References

[1] K. T. Malladi et al. FAME: A Fast and Accurate Memory Emulator for New Memory System Architecture Exploration. In *MASCOTS*, 2015.

[2] Q. Xu et al. Performance Analysis of NVMe SSDs and their Implication on Real World Databases. In *SYSTOR*, 2015.

SYSTOR '16, June 06-08, 2016, Haifa, Israel.
Copyright is held by the owner/author(s).
ACM 978-1-4503-4381-7/16/06.
http://dx.doi.org/10.1145/2928275.2933277