2025년도 한국통신학회 추계종합학술발표회

NVMe 에뮬레이터 NVMeVirt의 성능 재현성 및 한계 분석

김정현, 최수빈, 전경구*
인천대학교 임베디드시스템공학과
holssy12@inu.ac.kr, csbcms22@inu.ac.kr, kjun@inu.ac.kr*

Analysis of Performance Reproducibility and Limitations of the NVMe Emulator NVMeVirt

Jeonghyeon Kim, Subeen Choi, Kyungkoo Jun*

Dept. of Embedded Systems Engineering, Incheon National University

요 약

초고속 저장장치의 성능 병목 관련 연구에 활용되는 에뮬레이터의 실제 장치에 대한 성능 모사 정확성을 검증할 필요가 있다. 본 논문은 Non-Volatile Memory express (NVMe) 에뮬레이터 NVMeVirt와 실제 Solid State Drive (SSD) 간 성능 유사도를 비교하고, 차이의 원인을 분석한다. 실제 SSD와 이를 모사한 NVMeVirt에 다양한 워크로드를 이용하여 처리량과 지연 시간을 측정한다. 실험은 측정 편차를 최소화하기 위해 시스템 상황을 일관되게 통제하고, 벤치마크 툴 FIO를 이용한다. 실험 결과에서 NVMeVirt는 실제 SSD보다 낮은 처리량과 높은 지연 시간을 보였다. 이러한 NVMeVirt의 상대적 저성능은 읽기 워크로드보다 쓰기 위주의 워크로드에서 두드러졌다. 원인은 NVMeVirt의 제한적 병렬구조로 추정된다. 실제 SSD는 다중 플레인 구조에 기반한 병렬성으로 성능을 향상시킨다. 반면, 단일 플레인 구조로 구현된 NVMeVirt는 성능에 제한을 가진다. 이러한 한계에도 불구하고, NVMeVirt는 워크로드 변화에 따른 성능 추세 변화에서는 실제 SSD를 모사하였다.

Ⅰ. 서 론

NVMe SSD 도입으로 하드웨어 성능은 향상됐으나, 기존 운영체제 시스템 한계로 I/O 병목이 주요 문제로 대두되었다 [1]. 이를 보완하기 위한 연구들에서 실제 SSD를 활용한 실험이 필요하지만, 다양한 장치를 확보하여 제안 방법들의 일반성을 검증하는 데에는 어려움이 많아, NVMeVirt[3] 등과 같은 에뮬레이션 기반 실험을 진행한다 [2].

NVMeVirt는 운영체제에 물리 SSD와 동일하게 인식되는 에뮬레이터다. 하지만, 이러한 에뮬레이터가 실제 SSD를 모사했을 때의 성능 재현성 연구는 드물다[3]. 본 논문에서는 NVMeVirt와 실제 SSD (KIOXIA XG6)의 성능을 비교하고, 차이의 원인을 분석한다. 본 논문의 기여는 다음과 같다.

- **성능 비교의 일관성 보장을 위한 절차 마련**: 측정 결과의 신뢰도를 확보하는 벤치마크 방법론을 제시한다.
- **NVMe SSD와 NVMeVirt 간 성능 비교**: NVMeVirt의 성능 모사 능력을 정량적으로 파악한다.
- **NVMeVirt의 내부 병렬성 한계 파악**: 성능 차이의 원인 중 하나인 구조적 제약을 분석한다.

Ⅱ. 본 론

2.1. 실험 환경 및 방법

KIOXIA XG6(이하 XG6)와 NVMeVirt의 성능 비교를 위해 블록 저장장치 벤치마크 툴인 FIO[10]를 사용한다. 실험은 표 1과 같은 동일 사양 서버 2대에서 진행한다. NVMeVirt는 32GB RAM을 스토리지 백엔드로 사용하도록 구성한다. 실험 일관성 및 결과 재현성을 위해 TRIM 명령어로 주소 매핑 테이블의 유효상태 갱신, Preconditioning 및 Pre-fill로 장치가 안정상태에서 동작하도록 유도, Cool-down 및 Ramp-up으로 측정 간 차이를 최소화한다 [5-9]. 또한, 시스템 변수 통제로 실험 결과에 영향을 끼칠 수 있는 간섭을 최소화한다. 먼저, 일정한 처리 속도를 위해 CPU의 동적 주파수 조절 기능인 governor를 고정하고 부스트 기능을 해제한다. 또한, 메모리 접근 지연의 변동성을 줄이고자 운영체제의 동적 메모리 관리 기능인 Transparent Huge Pages와 Address Space Layout Randomization을 비활성화한다. 마지막으로 스토리지 I/O 경로의 잠재적 지연 방지를 위해 NVMe의 자동 전력 관리와 운영체제의 I/O 스케줄러를 해제하고, 비필수 데몬들을 모두 중지한다.

표 1. 서버 하드웨어 및 소프트웨어 사양.

항목	사양
CPU	Intel Xeon W-2255
메모리	128GB
메인보드	Dell 06JWJY
칩셋	Intel C422
운영체제	Ubuntu 24.04.2 LTS
커널 버전	5.19.17

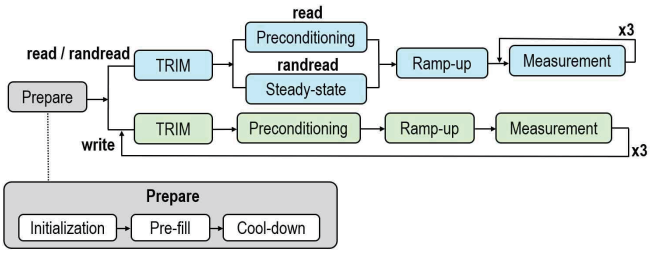
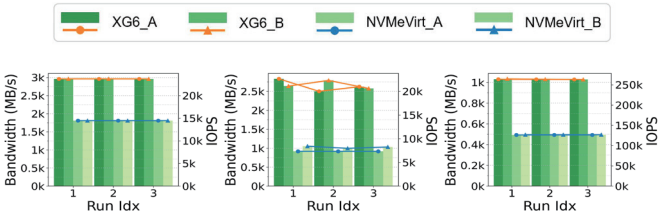


그림 1. read(순차읽기), write(순차쓰기), randread(랜덤읽기) 워크로드의 성능을 일관된 조건에서 측정하는 과정.

벤치마크 워크로드는 순차와 랜덤으로 구성한다. 순차는 처리량 중심 측정을 위해 128 KiB 블록을, 랜덤은 Flash Translation Layer 오버헤드 분



(a) (b) (c)

그림 2. 동일 사양 서버 2대(A, B)의 XG6와 NVMeVirt 벤치마크 대역폭과 IOPS. (a) 순차읽기, (b) 순차쓰기, (c) 랜덤읽기.

표2. 동일 사양 서버 2대(A, B)의 XG6와 NVMeVirt 벤치마크 결과. 각 워크로드의 3-run 지연(p50, p99) 평균값.

Device	p50 mean(ms)			p99 mean(ms)		
	순차 읽기	순차 쓰기	랜덤 읽기	순차 읽기	순차 쓰기	랜덤 읽기
XG6_A	1.335	1.379	0.108	1.592	5.691	0.257
XG6_B	1.335	1.319	0.108	1.489	5.953	0.256
NVMeVirt_A	2.212	1.270	0.196	3.075	21.365	0.930
NVMeVirt_B	2.212	3.927	0.196	4.047	6.477	0.930

석을 위해 4 KiB 블록을 사용한다. 순차는 읽기와 쓰기를 모두 포함하나, 랜덤은 쓰기의 Garbage Collection (GC), wear leveling 등으로 인한 성능 편차 때문에 읽기만 사용한다 [4]. 성능 비교 지표로 Input/Output Operations Per Second (IOPS)와 지연 시간을 측정한다. 모든 워크로드의 병렬 처리를 위해 Queue Depth (QD)=32로 설정한다.

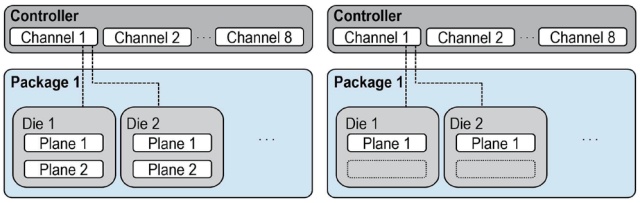
워크로드별 실행 절차는 그림 1과 같다. 한 차례의 전체 실행을 세션으로, 한 세션 안에 각 반복 측정인 런으로 구성한다. 모든 세션은 TRIM 단계까지 진행하여 장치를 초기화한다. 이후, 읽기는 세션당 한 번 Preconditioning을 수행하고 Ramp-up 및 Measurement를 런마다 반복한다. 반면 쓰기 워크로드는 장치 상태를 변경하므로, 런마다 TRIM부터 Measurement까지 진행하여 이전 런의 영향을 최소화한다. 각 런은 처리량을 IOPS와 MB/s로, 지연을 p50과 p99로 산출한다. 여기서 p50은 중앙값, p99는 전체 측정값 중 99%가 해당 값 이하임을 의미한다.

2.2. 결과 및 분석

그림 2와 표 2는 NVMeVirt가 모든 워크로드에서 XG6 성능과 차이가 있음을 보여준다. XG6보다 처리량은 낮고 지연시간이 길다. 이러한 차이는 쓰기에서 더 두드러졌다. XG6는 높은 처리량을 유지한 반면, NVMeVirt는 더 낮은 부하에서 성능이 포화되었고 p99 지연 시간이 급격히 증가하였는데, 이는 GC 개입의 영향으로 보인다.

성능 차이의 주요 원인은 NVMeVirt의 병렬 처리 에뮬레이션 한계로 추정된다. 그림 3은 XG6의 경우 다중 플레인 구조로 병렬 처리를 지원하지만, NVMeVirt는 단일 플레인 구조로 한계가 있음을 보여준다. 이러한 차이가 NVMeVirt의 상대적으로 낮은 처리량, 높은 지연 시간, 그리고 쓰기 부하에서 조기 성능 포화를 유발하는 것으로 보인다.

절대적 성능 차이에도 불구하고, NVMeVirt가 워크로드간 성능 변화 추이는 실제 장치와 유사성을 보인다. 그림 2의 추세와 표 2를 보면 워크로드별 변화, 지연시간 변화 등에서 비슷한 경향을 보인다. 이는 NVMeVirt가 새로운 알고리즘 연구 등에 있어 효과성 검증 등에 활용될 수 있음을 의미한다.



(a) (b)

그림 3. NAND 플레인 병렬성 비교. (a) KIOXIA XG6: 채널 - 다이 - 복수 플레인 구조로 플레인 병렬화 가능 (b) NVMeVirt: 다이당 플레인=1로 단순화되어 내부 병렬성 부재.

Ⅲ. 결 론

본 연구에서 NVMe SSD와 NVMeVirt의 성능을 비교한 결과, NVMeVirt는 단일 플레인 구조 때문에 실제 장치 성능 모사에는 한계가 있다. 하지만, 워크로드 간 성능 변화 추이는 재현 가능하여, 비교 연구의 기초 자료 등으로 제한적 활용은 가능하다.

ACKNOWLEDGMENT

"이 논문은 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원 - 학·석사연계ICT핵심인재양성 지원을 받아 수행된 연구임"
(IITP-2025-RS-2023-00260175)

참 고 문 헌

[1] Haas, G., et al., "What Modern NVMe Storage Can Do, And How To Exploit It," PVLDB, 2023.

[2] A survey of SSD simulators and emulators, The Journal of Supercomputing, 2025.

[3] Kim, S.-H., Shim, J., Lee, E., Jeong, S., Kang, I., and Kim, J.-S., "NVMeVirt: A Versatile Software-defined Virtual NVMe Device," Proc. USENIX FAST '23, pp. 379-394, 2023.

[4] Ren, T., Du, Y., Cui, J., Lv, Y., Li, Q., and Xue, C. J., "Device-Level Optimization Techniques for Solid-State Drives: A Survey," arXiv:2507.10573, Jul. 2025.

[5] Frankie, T., Hughes, G., and Kreutz-Delgado, K., "Analysis of Trim Commands on Overprovisioning and Write Amplification in Solid State Drives," arXiv:1208.1794, Aug. 2012.

[6] Maruf, A., Brahmakshatriya, S., Li, B., Tiwari, D., Quan, G., and Bhimani, J., "Do temperature and humidity exposures hurt or benefit your SSDs?," Proc. DATE '22, pp. 352 - 357, 2022.

[7] Zhan, Y., et al., "Rethinking the Request-to-IO Transformation Process of File Systems for Full Utilization of High-Bandwidth SSDs," Proc. USENIX FAST '25, pp. 69 - 86, 2025.

[8] Jiang, X., et al., "Thermal Modeling of Hybrid Storage Clusters," J. Signal Process. Syst., vol. 72, no. 3, pp. 181 - 196, Sep. 2013.

[9] M. Beseda, V. Cortellessa, D. D. Pompeo, L. Traini, and M. Tucci, "A Kernel-Based Approach for Accurate Steady-State Detection in Performance Time Series," Jun. 04, 2025, arXiv:2506.04204.

[10] Jens Axboe. fio: flexible I/O tester. https://github.com/axboe/fio