ISSN 2383-630X(Print) / ISSN 2383-6296(Online)
Journal of KIISE, Vol. 49, No. 4, pp. 271-275, 2022. 4
https://doi.org/10.5626/JOK.2022.49.4.271

MQSim-E: 엔터프라이즈향 SSD에 특화된 NVMe SSD 시뮬레이터

(MQSim-E: Design and Implementation of an NVMe SSD Simulator for Enterprise SSDs)

홍 두 원 † 이 두 솔 †† 김 지 홍 †††
(Duwon Hong) (Dusol Lee) (Jihong Kim)

요 약 SSD와 같은 저장장치 시스템의 연구에서는 시스템 내부의 SW/HW의 동작 방식을 정확히 모사하는 시뮬레이터가 중요한 역할을 한다. 본 논문에서는 NVMe SSD의 연구에 광범위하게 사용되는 MQSim이 엔터프라이즈향 SSD의 개발에 부적합함을 보이고 엔터프라이즈향 SSD에서 채택되는 최적화 기법들을 지원하는 MQSim-E 시뮬레이터를 제안한다. MQSim-E는 기존의 MQSim에 비해 플래시 메모리의 병렬성을 적극적으로 활용하며 가비지컬렉션의 성능 오버헤드를 최소화하여 엔터프라이즈향 SSD에서 중요한 설계 목표인 IOPS는 최대 210% 개선하며 꼬리 응답시간은 최대 16,000% 감소시켜 상용 엔터프라이즈향 SSD의 특성이 정확히 반영되도록 개선하였다.

키워드: 시뮬레이터, 엔터프라이즈향 SSD, 낸드 플래시, 엔터프라이즈 서버, 데이터 센터

Abstract In the study of storage systems such as SSD, a simulator that accurately mimic the operation of SW/HW inside the system plays an important role. In this paper, MQSim, which is widely used in research on NVMe SSDs, was shown to be inappropriate for the development of enterprise-SSD, and we propose an MQSim-E simulator that supports optimized techniques adopted in enterprise-SSD. MQSim-E fully utilizes the parallelism of flash memory and minimizes the performance overhead of garbage collection, improving IOPS, which is an important design goal for enterprise-SSDs, by up to 210% and reducing tail latency by up to 16,000% compared to the existing simulator (MQSim) to accurately reflect the characteristics of commercial enterprise SSDs.

Keywords: simulator, enterprise SSD, NAND flash, enterprise server, data center

· 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임(2021R1H1A2093240)
· 이 논문은 삼성전자 미래기술육성센터의 지원을 받아 수행된 연구임(SRFC-IT2002-06)
· 이 논문은 2021 한국컴퓨터종합학술대회에서 'MQSim-E: 엔터프라이즈향 SSD에 특화된 NVMe SSD 시뮬레이터'의 제목으로 발표된 논문을 확장한 것임

† 비 회 원 : 서울대학교 컴퓨터공학부 박사
duwon.hong@davinci.snu.ac.kr
†† 학생회원 : 서울대학교 컴퓨터공학부 학생
dslee@davinci.snu.ac.kr
††† 종신회원 : 서울대학교 컴퓨터공학부 교수(Seoul Nat'l Univ.)
jihong@davinci.snu.ac.kr
(Corresponding author임)

논문접수 : 2021년 10월 7일
(Received 7 October 2021)
논문수정 : 2021년 12월 24일
(Revised 24 December 2021)
심사완료 : 2022년 1월 3일
(Accepted 3 January 2022)

정보과학회논문지 제49권 제4호(2022. 4)

1. 서 론

SSD(Solid State Drive)가 저장장치 매체로 모바일 시스템에서부터 엔터프라이즈 규모의 컴퓨터 시스템에 이르기까지 여러 분야에서 활발히 사용됨에 따라 SSD 관련 연구에 사용되는 시뮬레이터들이 개발되어왔다. SSD 시뮬레이터의 개발은 기존 시뮬레이터들의 한계점들을 보완하는 방향으로 개발되어왔고 가장 최근에는 NVMe 프로토콜, SSD 내부 각 구성요소 동작들의 정확한 지연 모델, 최신 FTL(Flash Translation Layer) 기법들을 모두 지원하여 정확히 SSD의 입출력 성능을 평가하는 MQSim 시뮬레이터가 제안되었다[1].

하지만 MQSim 내에서 I/O 집약적인 엔터프라이즈 워크로드를 수행 시 정확한 결과를 예측하지 못하는 상황이 발생하며, 이는 MQSim이 엔터프라이즈향 SSD 연구에 부적합함을 의미한다. 클라이언트 SSD와 달리 엔터프라이즈향 SSD는 (i) 높은 수준의 I/O operations per second (IOPS) (ii) QoS 유지를 보장해야 한다. 하지만 현재 MQSim은 이 조건들을 충족하고 있지 못하다.

따라서 우리는 MQSim이 엔터프라이즈향 SSD의 필수조건들을 충족하지 못하는 이유들을 식별하였다. 첫째, 낮은 다중 플레인(multi plane) 병렬 연산 활용으로 인해 플래시 메모리의 병렬성을 충분히 활용하지 못해 실제보다 쓰기 성능이 저평가되는 문제가 있다. 최근 스토리지 서버에서는 높은 쓰기 대역폭을 달성하기 위해 동적 주소 흩뿌리기 방식(dynamic physical address stripping) 방식을 통해 데이터를 저장하고 있으며 병렬 쓰기를 최대한 활용하기 위해 쓰기 캐시에서 버퍼링(buffering) 기능을 활용하고 있다[2]. 하지만 MQSim에서는 논리주소(LBA)에 따라 물리주소(PBA)를 정적으로 할당하여 데이터를 저장하는 정적 주소 흩뿌리기(static physical address stripping) 방식을 사용하고 있으며 랜덤 쓰기 요청들을 버퍼링하지 않는 캐시의 랜덤 쓰기 추방 정책으로 인해 병렬 쓰기를 충분히 활용하고 있지 못하다. 둘째, OP(Over-provisioning) 영역과 가비지컬렉션이 만들어 낸 빈 공간(free space)의 비효율적 활용으로 인해 과도한 쓰기 증폭(write amplification) 및 긴 꼬리 응답 시간의 문제를 발생시킨다. SSD는 블록 내 무효화된(invalidated) 페이지들을 회수하여 새로운 쓰기를 위한 빈 공간(free space)을 만들기 위해 가비지컬렉션을 수행하는데, 가비지컬렉션의 대상 블록(victim block)의 유효한 페이지들이 많을수록 가비지컬렉션 오버헤드로 인한 쓰기 증폭이 증가하는 특징이 있다. 일반 SSD들은 실제 SSD의 물리적인 주소 공간 크기보다 논리 주소 공간을 작게 허용하여 OP 영역을 확보하고 이를 통해서 가비지컬렉션 발생을 지연시켜 쓰기 증폭을 최소화하고 있다. 하지만 MQSim에서는 OP 영역의 이러한 기능을 고려하지 않은 가비지컬렉션 정책으로 쓰기 증폭이 높게 측정되고 있다. 또한 가비지컬렉션이 만들어 낸 빈 공간의 양을 고려하지 않은 쓰기 정책으로 SSD 내의 빈 공간이 모두 소진되어 발생하는 포그라운드-가비지컬렉션에 의해 사용자 I/O요청의 꼬리 응답시간이 악화되는 현상이 발생한다.

본 논문에서는 식별된 MQSim의 문제점을 다음과 같이 개선한다. 첫째, 동적 주소 흩뿌리기 방식과 랜덤 쓰기 버퍼링 방식의 도입을 통해 시뮬레이터 내에서 정확한 쓰기 성능을 측정할 수 있도록 한다. 둘째, OP 영역을 활용한 지연-가비지컬렉션을 통해 과대 측정되었던 쓰기 증폭의 문제를 최소화 하고, 가비지컬렉션의 수행으로 인해서 생성된 빈 공간을 고려하여 사용자 쓰기를 허용하는 사용자/가비지컬렉션 수행 조절을 통해 포그라운드-가비지컬렉션을 사전 방지하여 과대평가 되었던 긴 꼬리 응답시간을 정상화한다.

논문의 구성은 다음과 같다. 2장에서는 대상이 되는 MQSim의 구조 및 특성과 문제점에 대해서 서술하고 3장에서는 MQSim-E의 전체 구조와 정확한 쓰기 대역폭/쓰기 증폭/긴 꼬리 응답 성능 평가들을 가능하게 하는 기법들을 소개하고 4장에서는 MQSim-E로 엔터프라이즈 워크로드 수행 시 개선된 평가들을 성능 평가 비교를 통해 설명하고 5장에서 결론을 맺는다.

2. 대상 시스템 구조 및 특성

2.1 엔터프라이즈향 SSD 특성과 필요 기능들

빅데이터를 다루는 데이터 센터나 엔터프라이즈 서버에서 사용자에게 높은 성능을 보장하기 위해서는 높은 수준의 SSD 내부 병렬 쓰기 지원이 필요하다. 따라서 최신 고성능 엔터프라이즈향 SSD에서는 동적 주소 흩뿌리기 방식인 슈퍼블록(superblock) 단위(SSD 내 채널, 칩, 다이, 플레인 내 블록 묶음 단위)의 쓰기 지원[5,6]을 통해 SSD 내부 병렬 쓰기를 최대한 활용한다. 또한, 서버에서의 높은 수준의 성능 안정성을 보장하기 위해 QoS 유지를 보장하는 가비지컬렉션 수행 기능을 지원해야 한다[4]. SSD 제조 업체에 따라 그 기능의 디자인과 구현 방식에는 차이가 있으나 기본적으로 가비지컬렉션의 동작들(유효 데이터 복사, 블록 지우기 연산)의 분산 처리를 통해 긴 꼬리 응답 시간을 방지하고 안정적인 IOPS 성능을 보장한다[7,8].

2.2 MQSim 시스템 구조

MQSim은 엔터프라이즈 환경을 위한 고 성능 SSD 시뮬레이터로서, 최신 인터페이스 기술인 NVMe를 지원하여 높은 IOPS 성능을 제공하며 SSD내에서 사용자 요청이 겪는 모든 지연 시간들을 모델링하여 정확한

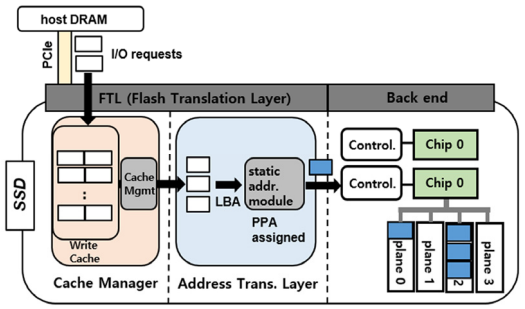


그림 1 MQSim 시스템구조와 각 요소의 동작 방식

Fig. 1 MQSim system structure and operation of each component

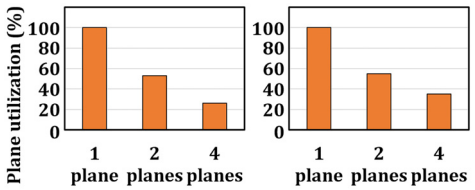
입출력 성능을 평가할 수 있다. 그림 1은 대상 SSD 시스템 구조를 보여준다. MQSim에서는 저장되는 데이터의 논리 주소의 값에 따라 채널, 칩, 다이, 플레인 위치 순서를 달리하는 총 24가지의 정적 주소 흩뿌리기 방식들을 지원하며 GC 도중에 사용자 I/O가 도착할 시, GC를 선점하는 스케쥴링 기법[9]을 따라 GC가 수행된다.

2.3 엔터프라이즈향 SSD 연구로 MQSim 활용의 부적합성

현재 MQSim에서는 비효율적인 SSD 내 자원 활용으로 인하여 잘못된 결과를 예측하는 상황들이 발생한다.

첫째, 비효율적인 쓰기 캐시 관리 정책과 주소 변환 정책으로 인해 다중 플레인 병렬 연산을 충분히 활용하지 못하고 있다. 쓰기 요청 시, 정적 주소 할당 방식은 요청의 논리주소 값에 따라 물리적인 플레인(plane)을 할당한다. 하지만 이 방식은 그림 1에서와 같이 플레인별 쓰기요청의 편차로 인해서 특정 플레인에만 쓰기요청이 집중되는 문제가 있다. 또한 쓰기 캐시에서 병렬성을 고려하지 않은 추방정책으로 인해서 산발적으로 쓰기요청이 발생하면서 효율적인 다중 플레인 쓰기 명령[10]의 활용 가능성이 현저하게 저하된다. 그림 2는 낮은 다중 플레인 병렬 연산 활용으로 쓰기 병렬성을 충분히 활용하지 못하는 상황을 보여준다. 그림 2(a)는 쓰기 집약적인 워크로드에서 실험을 진행한 모습이며 plane 개수가 늘어남에 따라 plane 활용률이 낮아져 4 planes 상황에서는 28% 수준으로, 다중 plane 연산을 충분히 활용하지 못함을 확인할 수 있다. 그림 2(b)는 읽기와 쓰기가 균등하게 발생하는 워크로드에서 실험한 결과이며 쓰기에 비해 읽기 요청의 경우 캐시에서의 재활용률이 높기 때문에 그림 2(a)에 비해 plane 활용률이 전체적으로 높으나, 여전히 4 planes 상황에서 34%로 낮은 plane 활용률을 보여준다.

둘째, 가비지컬렉션 수행 간 OP 영역의 낮은 활용과 빈 공간의 비효율적 활용으로 인한 성능 오버헤드가 있다.



(a) R:W = 0: 100% (b) R:W = 50%: 50%

그림 2 성격이 다른 두 워크로드 상황에서 plane 변화에 따른 MQSim의 plane 활용률

Fig. 2 Plane utilization of MQSim varying plane numbers under two workloads

SSD는 블록 내 무효화된 페이지들을 회수하여 빈 공간을 생성하기 위해 가비지컬렉션을 수행한다. 가비지컬렉션은 페이지들의 집합인 블록 단위로 발생하며 블록 내 유효화된 페이지들이 많을수록 쓰기 증폭이 증가하여 가비지컬렉션 성능 오버헤드가 증가한다. 쓰기 요구량이 많은 엔터프라이즈 SSD의 경우 빈 공간의 소진을 방지하기 위해 가비지컬렉션 성능 오버헤드 최소화로 빠른 빈 공간의 생성이 필요하다. 하지만 MQSim에서는 GC로 인한 긴 꼬리 응답 문제를 회피하기 위해서 일찍부터 GC를 수행하기 때문에 OP공간을 효율적으로 활용하지 못하여 쓰기 증폭이 증가하는 문제가 있으며, GC의 빈 공간의 생성 속도를 고려하지 않은 쓰기정책으로 인해서 포그라운드-가비지컬렉션의 발생으로 인한 긴 꼬리 응답의 문제도 완벽하게 회피하지 못하는 한계를 가지고 있다. 그림 3은 높은 가비지컬렉션 성능 오버헤드와 비효율적인 쓰기 정책으로 인해 빈 공간이 모두 소진되어 꼬리 응답시간이 악화되는 상황을 보여준다. 빈 공간은 GC가 발생하는 순간부터 포그라운드-가비지컬렉션 발생 임계지점(빈 공간: 100)까지 감소하여 포그라운드-가비지컬렉션을 발생시키며, 이로 인한 긴 꼬리 응답시간이 6.5초까지 증가하는 것을 관찰할 수 있다.

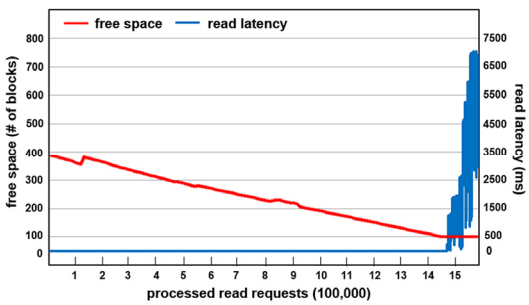


그림 3 포그라운드-가비지컬렉션 발생으로 인한 읽기 꼬리 응답 시간 악화

Fig. 3 Deteriorated read tail latency due to foreground-garbage collection

3. MQSim-E

3.1 동적 흩뿌리기 방식과 랜덤쓰기 버퍼링 방식

쓰기의 정적 물리 주소 할당 방식과 캐시의 랜덤 쓰기 추방정책은 플래시 메모리 내부 병렬성을 충분히 활용하지 못하여 쓰기 성능이 낮다는 문제점이 있다. 이 문제를 해결하기 위해 쓰기의 경우 요청들을 다중 플레인 단위(채널, 칩, 다이, 플레인 내 블록 묶음 단위)로 캐시에 버퍼링하여 플래시 메모리 내부 모든 플레인에 동시에 흩뿌려주는 동적 주소 흩뿌리기 방식을 채택하였다. 그림 4는 캐시에 버퍼링된 다중 쓰기 요청들이 모든 플레인에 흩뿌려져 저장되는 모습을 나타낸다.

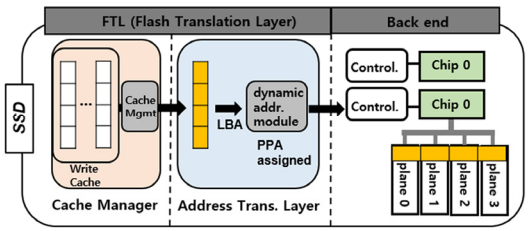


그림 4 버퍼링된 다중 쓰기 요청들이 모든 플레인에 흩뿌려져 저장되는 모습

Fig. 4 Dynamic stripping through allocating write requests over all planes

3.2 지연 가비지컬렉션과 사용자/가비지컬렉션 수행 조절 방식

빈 공간을 생성하는 속도보다 소모 속도가 빠르면 빈 공간이 모두 소진되어 포그라운드-가비지컬렉션 발생으로 꼬리 응답 시간이 악화된다. 이를 해결하기 위해 OP 영역을 활용한 가비지컬렉션 발생 지연으로 가비지컬렉션 성능 오버헤드를 최소화하여 빈 공간 생성 속도를 높였으며, 사용자 쓰기의 빈 공간 사용을 조절하여 빈 공간이 모두 소진되는 현상을 방지하였다. 그림 5는 SSD 내에서 OP 영역의 활용을 통해 지연-가비지컬렉션이 수행되는 모습과, 생성된 빈 공간의 양을 고려해 사용자 쓰기가 조절/수행되는 모습을 보여준다.

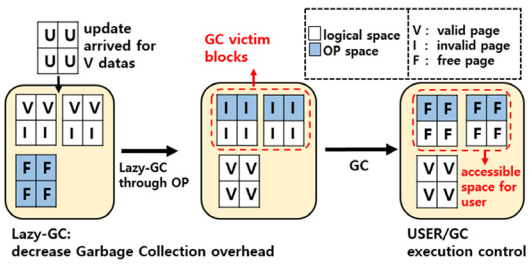


그림 5 지연-가비지컬렉션 (Lazy-GC)와 사용자/가비지컬렉션 수행 컨트롤

Fig. 5 Lazy-GC and USER/GC execution control

4. 실험 결과

개선된 시뮬레이터의 성능의 검증을 위해서 SSD 시뮬레이터인 MQSim 환경에서 FTL에 제안된 방식들을 적용하여 MQSim-E를 구현하였다. 표 1은 시뮬레이터 환경을 나타낸 것이다. 실험을 위해서 빅데이터 클라우드 벤치마크 YCSB[3]를 이용하여 특성이 다른 세 가지 워크로드들에 대해 I/O trace들을 생성했으며, 각 I/O trace에 대하여 실험을 진행하였다.

표 2는 제안된 MQSim-E 환경에서 IOPS (초당 입/출력 처리 횟수)와 꼬리 응답 시간 개선의 정도를 측정한 결과를 보여준다. MQSim-E에서는 플래시 메모리의 내부 병렬성을 최대한 활용할 수 있기 때문에 높은 I/O 처리량을 보여준다. 따라서 IOPS 성능이 기존 환경에서 보다 평균적으로 153% 개선되었고 최대의 경우 210% 개선 된 것을 확인할 수 있다(workload C의 경우). 또한 MQSim-E의 경우, 지연-가비지컬렉션의 효과로 쓰기 증폭이 대폭 감소하여 가비지컬렉션 성능 오버헤드 최소화로 빈 공간을 효과적으로 만들어내며, 사용자 쓰기의 조절/수행으로 포그라운드-가비지컬렉션 발생 없이 안정적인 꼬리 응답 시간을 보여준다. 표 2에서 99.9% 꼬리 응답 시간(read tail latency)이 workload A의 경우 최대 16,000% 개선된 것을 확인할 수 있다.

5. 관련 연구

다양한 SSD 시뮬레이터들이 공개되어왔다[11-13]. 공개

표 1 SSD 시뮬레이터 환경

Table 1 Simulated device configuration

Capacity	Channel	Chip	Die	plane	blocks per plane	pages per plane
2TB	4	4	1	4	4096	512

표 2 개선된 시뮬레이터(MQSim-E)의 성능 개선

Table 2 Performance evaluation of MQSim-E

results	workload A (W 50%)		workload B (W 30%)		workload C (W 20%)	
	MQSim	MQSim-E	MQSim	MQSim-E	MQSim	MQSim-E
plane util. (%)	47%	100%	43%	100%	40%	100%
IOPS	1,865	3,184	1,869	5,291	1,715	5,329
WAF	3.4	0.74	2.8	0.71	2.1	0.64
99% read tail latency (ms)	2201	0.114	1806	0.110	1862	0.110
99.9% read tail latency(ms)	6558	0.359	4346	0.288	3326	0.284

된 SSD 시뮬레이터들은 기존의 시뮬레이터를 보완하는 방향으로 발전해 왔지만 엔터프라이즈향 SSD 연구에 필수적인 기능들이 고려되지 않았다. 예를 들어, FlashSim과 MQSim의 경우 동적 주소 흩뿌리기 방식과 낮은 효율의 가비지 컬렉션 방식으로 인해 낮은 수준의 IOPS 성능을 보이며 QoS 유지를 보장하지 못하였다. 또한, 동적 주소 흩뿌리기 방식을 적용 가능한 FEMU와 SimpleSSD에서는 지연 가비지컬렉션과 사용자/가비지 컬렉션 수행 조절의 부재로 전체적인 IOPS 성능과 긴 꼬리 응답 시간이 MQSim과 비슷한 수준을 보임을 확인하였다.

6. 결 론

본 연구는 엔터프라이즈향 SSD에 필요한 기능들을 탐색하고 기존 SSD 시뮬레이터가 엔터프라이즈향 SSD 연구에서 보이는 문제점들을 식별하여 개선하였다. 기존 MQSim은 플래시 메모리의 병렬성을 충분히 활용하지 못하여 낮은 IOPS 성능을 보였고, 비효율적인 자원 활용으로 꼬리 응답 시간이 악화되는 문제를 보였다. 본 연구에서는 플래시 메모리의 병렬성을 최대한 활용하는 캐시 정책과 주소 할당 정책으로 IOP성능을 대폭 개선하였으며, 가비지컬렉션 효율 증대와 사용자 쓰기 조절 정책을 통해 꼬리 응답 시간을 정상화하였다.

References

[1] A. Tavakkol et al., "MQSim: A Framework for Enabling Realistic Studies of Moder Multi-Queue SSD Devices," *Proc. of the FAST*, 2018.

[2] C. Gao et al., "Boosting the Performance of SSDs via Fully Exploiting the Plane Level Parallelism," *IEEE TPDS*, Vol. 31, pp. 2185-2200, 2020.

[3] Yahoo Cloud Servicing Benchmark [Online]. Available: https://github.com/brianfrankcooper/YCSB (Accessed 12 May 2021).

[4] kingston, 2020, The Right Solid-State Drive (SSD) Matters, [Online]. Available: https://www.kingston.com/en/blog/servers-and-data-centers/ssd-matters

[5] D. Hong et al., "Reparo: A Fast RAID Recovery Scheme for Ultra-large SSDs," *TOS*, Vol. 17, No. 3, pp. 1-24, 2021.

[6] Micron Technology, "NAND Flash Media Management Through RAIN," 2011.

[7] S. Choudhuri et al., "Deterministic service guarantees for NAND flash using partial block cleaning," *Proc. of the CODES/ISSS*, 2008.

[8] M. Jung et al., "HIOS: A host interface I/O scheduler for solid state disks," *Proc of the ISCA*, 2014.

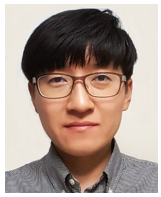
[9] J. Lee et al., "Preemptible I/O scheduling of garbage collection for solid state drives," *IEEE TACO*, Vol. 32, No. 2, pp. 247-260.

[10] C. Gao et al., "Parallel all the time: Plane level parallelism explo-ration for high performance ssds," *Proc of the MSST*, 2019.

[11] Y. Kim et al., "Flashsim: A simulator for nand flash-based solid-state drives," *Proc. of the SIMUL*, 2009.

[12] H. Li et al., "The CASE of FEMU: Cheap, accurate, scalable andextensible flash emulator," *Proc of the FAST*, 2018.

[13] D. Gouk et al., "Amber*: Enabling precise full-system simulationwith detailed modeling of all ssd resources," *Proc of the MICRO*, 2018.



홍 두 원

2003년 서울대학교 전기공학부(학사). 2006년 서울대학교 전기컴퓨터(석사). 2017년~2021년 서울대학교 컴퓨터공학부(박사). 2006년~현재 삼성전자 메모리사업부 연구원. 관심분야는 플래시 저장장치

이 두 솔

2019년 숭실대학교 스마트시스템소프트웨어학과(학사). 2019년~2020년 서울대학교 컴퓨터공학부(석사). 2020년~현재 서울대학교 컴퓨터공학부 박사과정. 관심분야는 플래시 저장장치, 임베디드 소프트웨어

김 지 홍

정보과학회논문지

제 49 권 제 2 호 참조