creative commons

COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:

저작자표시. 귀하는 원저작자를 표시하여야 합니다.

비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.

변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

공학석사학위논문

PowerSim: 하드웨어 가속 모듈을 활용한 HW/SW 통합 고성능 FTL 설계 환경 구축

2023년 8월

서울대학교 대학원

컴퓨터공학부

최 순 원

PowerSim: 하드웨어 가속 모듈을 활용한 HW/SW 통합 고성능 FTL 설계 환경 구축

지도 교수 김 지 홍

이 논문을 공학석사 학위논문으로 제출함
2023 년 5 월

서울대학교 대학원
컴퓨터공학부
최 순 원

최순원의 공학석사 학위논문을 인준함
2023 년 6월

위 원 장 _____ 이 창 건 _____ (인)

부위원장 _____ 김 지 홍 _____ (인)

위 원 _____ 김 태 현 _____ (인)

초 록

다양한 컴퓨터 시스템에서 저장 장치로 사용되는 SSD(Solid-State Drive)의 성능은 호스트 인터페이스의 개선, 채널/칩 대역폭 증가와 같은 다양한 최적화 연구를 통해 꾸준히 증가해오고 있다. 최신 SSD의 고성능 FTL의 기능을 수행하기 위해서 전통적인 소프트웨어 중심 FTL 설계에서는 보다 강력하고 많은 코어를 사용해야만 성능 요구조건에 부합할 수 있다. 하지만 이러한 방식은 과도한 전력소모를 발생시키는 문제점이 존재한다.

성능과 전력 요구조건을 모두 만족시키기 위해 임계 경로(Critical Path)에 대한 하드웨어 가속이 많이 이루어지고 있는 추세이다. 성능과 전력에 이점이 존재하지만 높은 개발비용과 구현 복잡도를 가지는 하드웨어 설계와 유연성을 통해 다양한 알고리즘 구현이 가능한 소프트웨어 설계의 장단점을 적절히 결합한 통합 설계가 필요하다.

본 논문에서는 하드웨어 가속 모듈을 이용하여 고성능 FTL을 위한 하드웨어/소프트웨어 통합 설계와 전력 모델을 고려한 PowerSim 시뮬레이터를 제안한다. SSD 컨트롤러를 설계하는데 있어 소프트웨어 설계의 경우 코어의 수와 코어 스펙, 하드웨어 설계의 경우 하드웨어 가속 유무와 하드웨어 가속 모듈의 수를

설정하여 다양한 하드웨어/소프트웨어 통합 설계 및 평가가 가능하다.

제안하는 하드웨어/소프트웨어 통합 설계를 위해 멀티큐 SSD 시뮬레이터인 MQSim을 확장하여 시뮬레이션 도구를 구현하였다. PowerSim의 효용성을 검증하기 위해 고성능 FTL을 가지는 최신 삼성 PM9A3 960GB SSD와 동일하게 낸드 플래시를 구성하고 전력 요구조건을 설정하였다. PowerSim을 통해 다양한 하드웨어/소프트웨어 통합 SSD 컨트롤러를 구성하고 실험하였으며, 설계한 컨트롤러에 대한 성능 및 전력을 평가하였다. 요구조건을 만족하는 다양한 컨트롤러를 제시함으로써 제조사는 클라이언트의 요구조건과 제조사의 상황에 맞는 SSD 컨트롤러 탐색이 가능하다.

주요어 : 낸드 플래시 메모리 기반 저장장치, SSD 시뮬레이터, 전력 모델, HW/SW 통합 설계, 하드웨어 가속
학 번 : 2021-29273

목 차

초 록 i

목 차 iii

표 목차 v

그림 목차 vi

제 1 장 서 론 1
제 1 절 연구 배경 1
제 2 절 연구 목적 및 기여 4
제 3 절 논문 구성 5

제 2 장 배경 지식 6
제 1 절 SSD 설계 도구: MQSim 6
제 2 절 그 외의 SSD 설계 도구1: FEMU 9
제 3 절 그 외의 SSD 설계 도구2: OpenSSD 10

제 3 장 HW/SW 통합 설계 환경의 필요성 11
제 1 절 소프트웨어 중심 SSD 설계의 한계 11
제 2 절 HW/SW 통합 설계 환경의 부재 12
제 3 절 전력 모델의 부재 14

제 4 장 SSD 내부 모듈화 지점 탐색 15
제 1 절 하드웨어 가속화 가능 작업 15
제 2 절 하드웨어 가속화 불가능 작업 17

제 5 장 HW/SW 통합 설계 환경 19
제 1 절 PowerSim 개요 19
제 2 절 SSD 컨트롤러 모델링 20
제 3 절 HW/SW 통합 설계 환경 구조 25
제 4 절 모듈 성능 및 전력 모델 27

제 6 장 실험 31
제 1 절 실험 환경 31

제 2 절 요구 전력 소모량 32
제 3 절 SSD 컨트롤러 구성.....33
제 4 절 실험 결과 35

제 7 장 결론.....38
제 1 절 결론 및 향후 연구.....38

참고 문헌 39

Abstract 42

표 목차

[표 1] SSD 컨트롤러 구성 매개변수 20

[표 2] PowerSim 이벤트 종류 26

[표 3] gem5 시스템 구성 매개변수 29

[표 4] PowerSim 매개변수 31

[표 5] SSD 컨트롤러 SW 설계 모델 34

[표 6] SSD 컨트롤러 HW/SW 통합 설계 모델 34

그림 목차

[그림 1] MQSIM 멀티 큐 구조 6

[그림 2] SSD 임계 경로..... 16

[그림 3] SSD 컨트롤러 구성 1 22

[그림 4] SSD 컨트롤러 구성 2 22

[그림 5] SSD 컨트롤러 구성 3 23

[그림 6] HW/SW 통합 SSD 컨트롤러 구성 1 23

[그림 7] HW/SW 통합 SSD 컨트롤러 구성 2 24

[그림 8] HW/SW 통합 SSD 컨트롤러 구성 3 24

[그림 9] HW/SW 통합 설계 환경 구조..... 25

[그림 10] 모듈 별 수행시간 및 전력 측정 과정 27

[그림 11] GEM5를 이용한 수행시간/전력 평가 환경 구축..... 28

[그림 12] SSD 전력 소모 비율 32

[그림 13] SSD 컨트롤러 구성 별 성능 및 전력 37

[그림 14] SSD 컨트롤러 구성 별 총 에너지 소모량 37

제 1 장 서 론

제 1 절 연구 배경

SSD는 현대 컴퓨팅에서 중요한 역할을 하는 저장장치로, SSD 설계 요구조건은 다양한 측면을 고려하여 충족되어야 한다. 꼬리응답시간, 장치수명, 용량, 신뢰성, 호환성, 보안, 비용, 성능, 전력소모 등과 같은 다양한 요구조건이 존재하지만, 제한적인 자원을 가지는 임베디드 환경에서 동작하기 때문에 SSD 요구조건을 맞추는 일이란 여간 쉬운 일이 아니다.

SSD 성능은 매년 증가하고 있으며, 클라이언트용 최신 SSD 1TB의 경우 초당 1,200,000개의 I/O 작업까지 처리할 수 있게 되었다 [1]. 이렇게 SSD 성능이 증가하고 있는 상황 속에서 전통적인 소프트웨어 중심 FTL 설계로 성능 요구조건을 만족시키기 위해서는 보다 강력한 ARM 코어를 사용해야 한다. 하지만 ARM 코어의 성능을 계속하여 높인다면, 전력 소모량 또한 높이지기에 전력소모 요구조건을 만족할 수 없게 된다.

성능적인 요구조건을 만족하면서 전력소모 요구조건을 만족시키기 위해 임계 경로(Critical Path)에 대해 하드웨어 가속화가 많이 되고

있는 추세이다. 복잡하고 변동성이 큰 작업에 대해 하드웨어로 구성한다는 것은 높은 개발비용과 구현 복잡도가 존재하지만, 임계 경로 상의 단순하면서도 빈번히 호출되는 단순한 작업에 대해 하드웨어 가속화를 통해 성능과 전력소모 요구조건을 모두 만족시킬 수 있다. 즉, 고객의 높은 요구조건을 만족시키기 위해 하드웨어/소프트웨어 통합 설계를 하지 않을 수 없다.

SSD 시뮬레이터는 낸드 플래시 기반의 스토리지에서 발생할 수 있는 다양한 상황을 시뮬레이션하고, 이를 통해 SSD의 동작을 분석하고 최적화할 수 있는 도구이다. 낸드 플래시는 쓰기 전 지우기(erase before write) 특성으로 인하여 다양한 기술적인 도전과제들이 존재한다. 예를 들어, 가비지 컬렉션(Garbage Collection), 쓰기 증폭(Write Amplification), 주소 변환(Address Mapping) 등의 현상이 발생할 수 있다.

이러한 복잡한 작업들과 고객의 SSD 설계 요구조건을 맞추기 위해 성능평가 및 기능검증 목적으로 실제 하드웨어로 구성하는 것은 시간이 많이 소요된다. 또한, 수정사항이 존재할 때마다 하드웨어를 매번 새롭게 구성한다면 무시할 수 없는 비용이 발생할 수밖에 없을 것이다. 그렇기 때문에 SSD 시뮬레이터를 통해 낸드 플래시 스토리지에서 발생할 수 있는 다양한 상황을 시뮬레이션하고,

이를 통해 설계한 SSD의 성능평가 및 기능검증을 보다 쉽게 가능하게 한다.

학술연구 목적으로 SSD를 시뮬레이션 할 수 있는 다양한 도구 [2, 3, 4]들이 존재한다. 하지만 대부분의 도구들이 성능(IOPS) 측면에서만 고려되었으며, 하드웨어/소프트웨어 통합 설계와 전력적인 부분은 전혀 고려되고 있지 않다. 임베디드 환경에서 엄격하고 높은 성능 및 전력소모 요구조건을 만족시키기 위해 하드웨어/소프트웨어 통합 설계가 불가피한 시점에 이를 지원하는 SSD 시뮬레이터가 요구된다.

제 2 절 연구 목적 및 기여

본 논문에서는 기존 SSD 시뮬레이터에서 고려되지 않았던 하드웨어/소프트웨어 통합 설계와 전력소모 모델링을 지원하는 시뮬레이터인 PowerSim을 제안한다. 임계 경로에 위치한 SSD 내부 작업에 대해 하드웨어 가속화 여부에 따라 모듈화를 진행하여, SSD 컨트롤러 구성 매개변수를 통해 다양한 하드웨어/소프트웨어 통합 SSD 컨트롤러 구성이 가능하다. 또한, SSD 컨트롤러에서 사용되는 코어와 DRAM의 전력모델을 추가하여 전력소모에 대한 평가도 가능하다. 구현한 시뮬레이션을 통해 고객 요구조건에 맞는 최적의 하드웨어/소프트웨어 통합 설계 환경을 제공한다.

멀티큐 NVMe SSD 시뮬레이터인 MQSim을 기반으로 하드웨어/소프트웨어 통합 시뮬레이터인 PowerSim을 개발하고 이를 바탕으로 실험을 진행하였다. 실험에서는 하드웨어/소프트웨어 통합 설계를 통해 다양한 SSD 컨트롤러를 구성하고, 구성한 SSD 컨트롤러에 대한 성능 및 전력을 확인하기 위해 다양한 실험을 진행하였다. 실험을 통해 고객 요구조건에 부합하는 다양한 SSD 컨트롤러 구성을 확인하고 최적의 하드웨어 가속화 지점에 대한 다양한 선택지를 제시할 수 있음을 확인하였다.

제 3 절 논문 구성

본 논문은 다음과 같이 구성되어 있다. 2장에서는 학술 연구 목적으로 흔히 사용되는 SSD 설계 및 평가 도구들에 대해 살펴본다. 3장에서는 소프트웨어 중심 SSD 컨트롤러 설계에 대한 한계점과 전력 모델 지원에 대한 어려움을 분석하고 하드웨어/소프트웨어 통합 설계 환경의 필요성을 제시한다. 4장에서는 하드웨어/소프트웨어 통합 설계를 위하여 SSD 내부 작업에 대한 모듈화 지점 탐색에 대하여 이야기한다. 5장에서는 논문에서 제안하는 하드웨어 가속 모듈을 활용한 고성능 FTL HW/SW 통합 설계 환경인 PowerSim의 전체적인 구조와 HW/SW 통합 SSD 구성 방법, 전력 모델에 대하여 다룬다. 6장에서는 구성한 다양한 SSD 컨트롤러에 대해 성능 및 전력 실험을 진행하고 요구조건에 부합한 SSD 컨트롤러를 탐색한다. 마지막으로 7장에서는 결론을 통해 논문의 기여를 다시 정리하고, 향후 확장될 수 있는 연구에 대해 다룬다.

제 2 장 배경 지식

제 1 절 SSD 설계 도구: MQSim

MQSim [2]은 Multi-Queue SSD Simulator의 약자로, SSD의 설계 및 성능 평가를 위해 개발된 도구이다. MQSim은 컨트롤러의 동작, 낸드 플래시 메모리의 특성, 요청 스케줄링 등 다양한 측면에서 SSD의 동작을 모사하고 분석하는데 사용되며, 다음과 같은 주요 특징을 가진다.

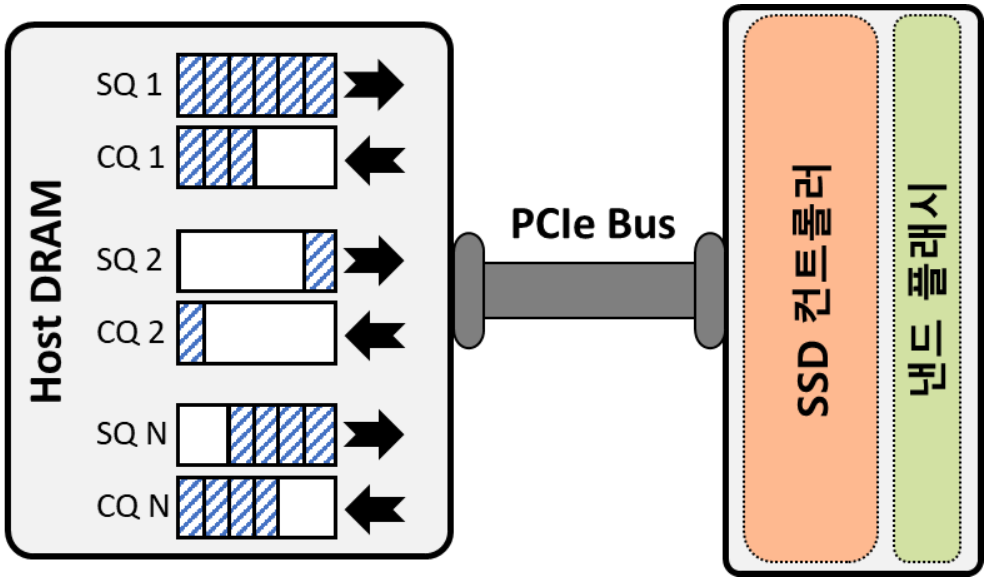

그림 1. MQSim 멀티 큐 구조

첫째, 그림 1과 같이 멀티 큐 구조를 지원하여 병렬 처리 및 더

나은 스케줄링을 가능케 한다. 기존 방식은 운영체제 I/O 스케줄러가 하나의 큐에 스케줄링 정책에 따라 I/O 요청을 삽입하고 이를 디바이스 스토리지에서 가져가게 된다. 이를 단일 큐에서 멀티 큐를 지원하도록 함으로써 병렬 처리가 가능하도록 하였으며, 운영체제 I/O 스케줄러를 제거하고 SSD 내부 자원 사용 실태를 잘 알고 있는 SSD 컨트롤러에게 스케줄링을 맡김으로써 보다 효율적인 스케줄링이 가능하다. 운영체제 I/O 스케줄러보다 더 나은 스케줄링이 가능하다.

둘째, 정상 상태(Steady State)에서 SSD 시뮬레이션 성능을 평가한다. 정상 상태란 SSD의 동작이 초기화 단계 이후에 안정화된 상태를 의미한다. 이 상태에서는 SSD의 동작 특성이 일정하며, 특정한 워크로드 또는 접근 패턴에 대해서 반복적인 동작 수행을 하기에 정상 상태에서의 성능 측정이 필요하다. 대부분의 기존 시뮬레이션은 정상 상태가 아닌 FOB(Fresh Out of the Box) 상태에서 평가가 이루어지기 때문에, 1) GC를 고려하지 않음, 2) FOB 장치에서는 쓰기 요청에 대해 무조건 쓰기 캐시(Write Cache)에서 처리되기 때문에 비현실적인 성능을 가짐, 3) FOB 장치에서는 항상 빈 페이지(Free Page)에 대한 요청을 처리하기 때문에 채널, 칩 수준의 병렬성을 완전히 활용할 수 있어

비현실적인 성능을 가짐 과 같은 문제가 발생한다. MQSim은 프리컨디셔닝(Pre-conditioning)을 통해 빠르게 정상 상태에 도달할 수 있도록 하여 보다 정확한 SSD 성능 평가가 가능하다.

셋째, I/O 요청에 대한 디바이스 응답시간에 대하여 보다 자세한 동작 시간을 측정한다. 사용자 I/O 요청 하나에 대한 처리 과정을 살펴보면 1) 제출 대기열(Submission Queue)에 I/O 요청 삽입, 2) PCIe 버스를 통해 I/O 요청을 디바이스로 전송, 3)주소 변환 후 칩 수준 대기열 삽입 후 스케줄링, 4) 플래시 칩으로 플래시 명령 전송, 5) 플래시 칩 읽기 수행, 6) ONFI(Open NAND Flash Interface) 버스를 통해 읽기 데이터 전송, 7) PCIe 버스를 통해 호스트에게 읽기 데이터 전송으로 세분화할 수 있다. 기존 시뮬레이터의 경우 디바이스 응답시간 중 지배적이라고 판단되는 5, 6번 작업에 대해서만 모델링 되어 있다. 하지만 위 과정에서 데이터 캐시 탐색 과정은 포함되어 있지 않으며, 주소변환 테이블 미스와 같은 상황에서는 3번 작업의 수행시간이 무시할 수 있는 수준이 아니게 된다. 이러한 점을 고려하여 MQSim은 데이터 캐시에서 읽기/쓰기 요청을 수행하는 시간과 필요로 하는 주소변환 정보가 DRAM에 위치하지 않을 경우 낸드 플래시에서 주소변환 정보를 가져오는 시간을 반영한 성능을 평가하고 있다.

제 2 절 그 외의 SSD 설계 도구1: FEMU

FEMU(Flash Emulator) [3] SSD 에뮬레이터는 낸드 플래시 동작을 에뮬레이션 하기 위해 개발된 도구로 실제 하드웨어 장치를 완전히 대체하여 동작을 한다. 이를 위해 소프트웨어 구성 요소를 사용하여 하드웨어 기능을 에뮬레이션하고, 소프트웨어에서 이를 호출하여 기능을 테스트 및 설계, 평가, 디버깅 등이 가능하다.

하지만 에뮬레이션 되는 호스트 스펙과 동일한 환경에서 SSD가 동작하게 된다. 그렇기 때문에 연구용 PC에서 실험하게 될 경우 임베디드 환경의 SSD 컨트롤러와 괴리가 존재하며 FEMU에서 측정되는 SSD 컨트롤러 수행시간은 고성능 자원에서 동작하기에 미미할 수밖에 없다. 이를 해결하기 위해 임베디드 환경의 호스트를 구성하기에는 비용적인 제약사항이 존재한다.

제 3 절 그 외의 SSD 설계 도구2: OpenSSD

OpenSSD [4]는 오픈 소스 기반의 SSD 프로젝트로, SSD의 하드웨어와 소프트웨어를 개발하고 연구하기 위한 플랫폼이다. SSD의 하드웨어와 그 제어를 담당하는 소프트웨어를 모두 다루기 때문에 전체 시스템을 포괄적으로 개발하고 평가할 수 있으며, 보다 정확한 SSD 컨트롤러 수행시간을 가진다. 또한, 사용자 정의된 플래시 관리 기술이나 새로운 기능을 추가하여 SSD의 성능 평가 및 검증이 가능하다.

하지만 학술연구 목적으로 사용하기에는 하드웨어 구매에 대한 비용적인 부담이 많이 존재하며, 업체에서 지원하는 규격에 대해서만 하드웨어 구성이 가능하다. 그렇기 때문에 하드웨어/소프트웨어 통합 설계에는 많은 어려움이 존재하기에 최신 SSD의 고성능 FTL을 시뮬레이션 하기에는 한계가 존재한다.

제 3 장 HW/SW 통합 설계 환경의 필요성

제 1 절 소프트웨어 중심 SSD 설계의 한계

SSD는 우수한 성능과 높은 저장 밀도 덕분에 다양한 컴퓨팅 영역에서 주요 저장매체가 되었다 [5, 6]. SSD는 플래시 계층을 쌓거나 비트당 더 많은 비트 표현을 넣어 밀도를 높이는 식으로 발전해왔다. 그로 인해 낸드 플래시의 저장 용량이 증가함에 따라 사용자 I/O 요청의 대기 시간이 악화되는 문제가 발생하였다.

펌웨어 대기 시간을 완화하는 방법 중 하나는 SSD 내부에 더 많은 프로세서를 배치하거나 높은 클럭 주파수를 가지는 프로세서를 사용하는 것이다 [7, 8]. 하지만 계속해서 프로세서를 강화하는 방식은 그만큼 더 많은 전력소모가 발생하고 높은 클럭 주파수의 프로세서를 사용한 펌웨어 수행은 높은 작동 온도 문제를 발생시킬 수 있다.

제 2 절 HW/SW 통합 설계 환경의 부재

SSD 컨트롤러의 성능을 끌어올림과 동시에 낮은 전력 소모를 유지할 수 있는 최적화 방법 중 하나가 하드웨어 가속화이다. SSD 주요 공급업체들은 임계 경로에 대한 하드웨어 가속화를 통해 클라이언트들이 성능과 전력소모 모두 만족할 수 있는 SSD를 제공하고 있다.

SSD 컨트롤러 설계에 대한 산업계의 추세가 바뀌고 있는 반면, 학계의 SSD 시뮬레이터는 여전히 전통적인 소프트웨어 중심으로 설계되어 있다. MQSim은 SSD 컨트롤러의 수행시간을 DRAM 접근을 제외하고 전혀 고려하고 있지 않으며, PCIe 전송 시간과 낸드 플래시 처리 시간으로 사용자 I/O 요청에 대한 디바이스 응답 시간이 결정된다. 즉, SSD의 용량이 증가하거나 SSD를 완전히 활용할 수 있는 워크로드에 대해서도 SSD 컨트롤러의 수행 시간이 고려되고 있지 않으므로, 높은 클럭 주파수를 가지는 프로세서를 사용하여 성능을 만족하고 있다고 볼 수 있다.

FEMU는 MQSim과는 달리 SSD 컨트롤러의 수행시간을 어느 정도 반영을 하고 있는 SSD 에뮬레이터이다. 하지만, 호스트 스펙 기반으로 동작하는 에뮬레이터이기 때문에 임베디드 환경에서의 자원과는 괴리가 있는 환경이며, 호스트 스펙에 따라 SSD

컨트롤러의 수행시간이 달라진다. FEMU 또한 호스트 스펙에 기반한 프로세서에서 소프트웨어 중심 SSD 컨트롤러가 동작한다고 볼 수 있다.

OpenSSD는 SSD의 하드웨어와 소프트웨어를 개발하고 연구하기 위한 플랫폼이다. SSD의 하드웨어와 그 제어를 담당하는 소프트웨어를 모두 다룰 수 있어, 전체 시스템을 포괄적으로 개발하여 성능평가 및 기능검증이 가능하다. 하지만 현재 지원하고 있는 제품에 대해서는 하드웨어/소프트웨어 통합 설계를 지원하지 않으며, 추후 지원한다 하더라도 다양한 SSD 컨트롤러 구성에 대한 탐색은 비용적인 부담이 존재한다.

제 3 절 전력 모델의 부재

SSD는 HDD와는 달리 기계적으로 움직이는 부품이 필요하지 않기 때문에, 임의 접근 성능과 전력소모 측면에서 HDD보다 우수하다. 높은 전력 효율성으로 인해 HDD의 전력 소모 분석 및 최적화에 대한 여러 연구들이 있지만[9], SSD의 전력 효율성은 많은 관심을 받지 못하였다.

하지만 최근 SSD는 I/O 성능을 높이기 위해 보다 집약적인 병렬 방식을 사용하고 DRAM 버퍼의 양이 많아지면서 SSD의 에너지 소비량은 HDD와 비슷하다. 초기 SSD와는 달리 SSD의 전력소모가 증가함에 따라 전력소모에 대한 최적화가 매우 중요해졌다.

그럼에도 불구하고 학술연구 목적의 기존 SSD 시뮬레이터들은 전력 모델을 지원하지 않고 있다. 먼저, 각 제조사마다 공정이 다를 뿐만 아니라 SSD의 내부 동작 및 전력 특성에 대한 세부 정보들은 외부에서 접근할 수 없다. 또한, SSD는 동적인 전력 소비 패턴을 가지고 있으며, 작업에 따라 전력 소비가 달라진다. 예를 들어, 읽기 작업과 쓰기 작업은 전력 소비 패턴이 다르며, 다양한 작업 부하에서의 동작 패턴에 따라 전력 소비가 달라질 수 있다. 전력 모델은 SSD의 동작과 관련된 이와 같은 요소들을 고려해야 하기 때문에 모델링하기에 많은 어려움이 존재한다.

제 4 장 SSD 내부 모듈화 지점 탐색

제 1 절 하드웨어 가속화 가능 작업

SSD에서 임계 경로란 데이터를 읽거나 쓰는 과정에서 가장 긴 시간이 소요되는 경로를 말하며, 성능에 직접적인 영향을 줄 수 있다. 일반적인 상황에서 가장 빈번히 발생하는 경로로 고객의 전력 소모, 성능 등의 높은 요구조건을 만족시키기 위해 경로 상의 작업에 대해 하드웨어 가속화를 많이 하고 있는 추세이다. 이를 위해서는 임계 경로에서 어떠한 작업이 하드웨어 가속화의 후보군이 될 수 있는지, 어떠한 작업이 소프트웨어로만 구성되어 동작할 수 있는지에 대해 분별하는 과정이 매우 중요하다.

그림 2는 SSD 내부 기능들 중에서 하드웨어 가속화가 가능한 작업들을 나타내며, 임계 경로는 다음과 같다. 1) 호스트 DRAM의 제출 대기열에서 PCIe 버스를 통한 I/O 요청 패치, 2) 호스트 인터페이스 컨트롤러에서 I/O 요청을 하나 이상의 트랜잭션으로 분할, 3) 트랜잭션의 논리 주소가 데이터 캐시에 존재 유무 탐색, 4) 트랜잭션의 논리 주소를 물리 주소로 변환, 5) 낸드 플래시 명령 전달, 6) 낸드 플래시 명령 수행, 7) 처리 완료된 트랜잭션에 대하여

I/O 요청 반환.

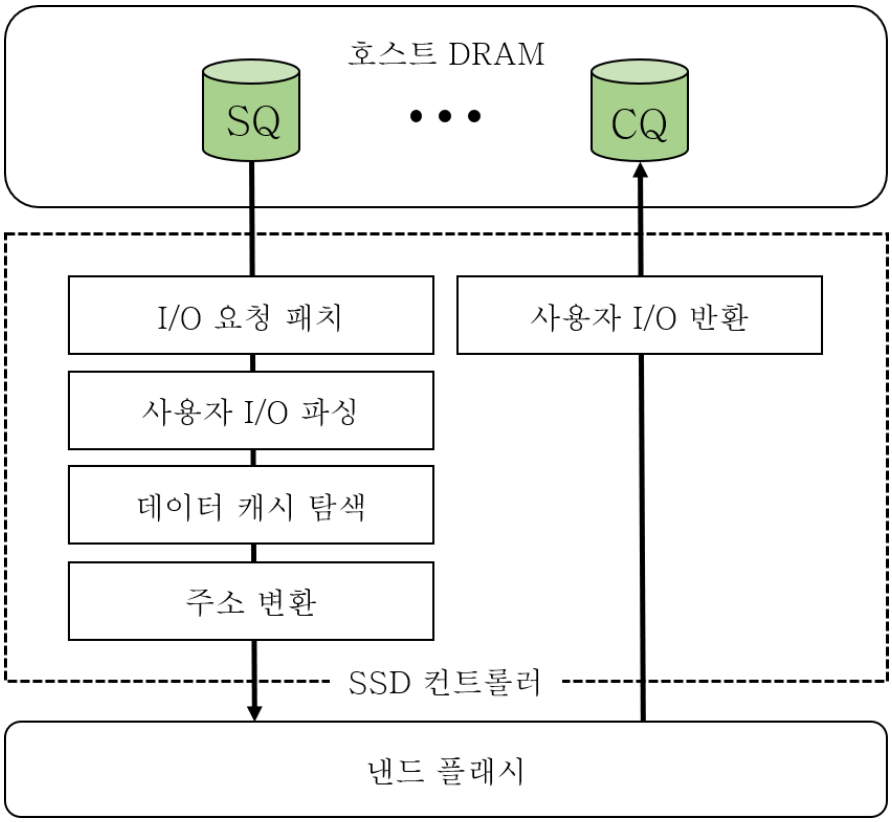

그림 2. SSD 임계 경로

그림 2의 임계 경로에 위치한 SSD 컨트롤러의 작업들은 모두 하드웨어 가속화가 가능하다. 단순하지만 빈번한 호출의 작업을 하드웨어 가속화를 통해 성능과 전력소모에 대해 긍정적인 효과를 이끌어낼 수 있을 뿐만 아니라, 변동성이 존재하지 않는 작업이기에 하드웨어 수정이 필요하지 않아 제조사에서 비용적인 측면에서 위험 부담이 적다.

제 2 절 하드웨어 가속화 불가능 작업

하드웨어 가속화는 일부 작업을 하드웨어 수준에서 직접 수행하므로 성능과 전력소모 측면에서 이점을 가질 수 있다. 특히 실시간 시스템이나 대량의 데이터 처리와 같은 작업에서는 더 유리하게 작용하게 된다. 이는 임베디드 시스템 환경에서 동작하여 제한적인 자원을 가지는 SSD에게는 매우 매력적인 최적화가 아닐 수 없다.

하지만 하드웨어 가속화는 하드웨어 제어, 신호 처리 등과 같은 여러 상호작용을 고려해야 하기 때문에 복잡도가 매우 높은 작업이다. 또한, 하드웨어 가속화를 위한 추가적인 하드웨어 부품이 필요하며, 하드웨어에 직접적으로 의존하기 때문에 변경사항이나 추가 요구사항에 대한 대응이 소프트웨어에 비해 더 복잡하고 많은 비용이 발생할 수 있다.

SSD 내부 작업에서는 임계 경로가 원활히 동작하기 위한 부가적인 작업들이 대부분 하드웨어 가속화가 불가능한 작업이라 할 수 있고, 대게 복잡한 알고리즘 또는 유연한 정책이 필요한 작업들로 다음과 같다. 그림 2의 사용자 I/O 파싱 과정에서 사용자 I/O 요청에 대해 하나 이상의 트랜잭션으로 분할하는 과정을 거치게 된다. 여기서 더 이상 할당 가능한 트랜잭션의 수가 부족하다면,

요청의 우선순위 및 다양한 상황을 고려하여 예외 처리를 수행하게 된다. 데이터 캐시 탐색 과정에서는 쓰기 캐시를 지원하고 캐시가 가득 찬 상황에서 캐시에 존재하지 않는 논리 주소에 대한 쓰기 요청이 들어올 경우, 어떠한 캐시 슬랏을 낸드 플래시 쓰기를 수행할지에 대해 결정을 하여야 한다. FTL 계층에서의 주소 변환 과정에서는 쓰기 요청에 대해 활성 블록(Active Block)의 페이지를 모두 사용하였을 경우, 어떠한 활성 블록을 새로 할당 받을지에 대한 결정이 필요하다. 마지막으로 SSD 내부의 사용 가능한 블록(Free Block)이 일정 수준 이하로 떨어진다면, 가비지 컬렉션(Garbage Collection)을 통해 사용 가능한 블록을 확보하는 작업이 필요하다. 이 과정에서 어떠한 블록을 희생 블록(Victim Block)으로 선정하여 가비지 컬렉션을 수행할지에 대한 결정이 필요하다. 이 외에도 오류 정정 코드(Error Correction Code), 읽기 간섭(Read Disturb), 읽기 재시도(Read Retry) 등과 같이 하드웨어로 구현될 수 없는 작업들이 존재한다.

제 5 장 HW/SW 통합 설계 환경

제 1 절 PowerSim 개요

본 논문에서는 새로운 접근 방법으로 앞서 언급한 한계점들을 해결하는 PowerSim 시뮬레이터를 제안한다. PowerSim 시뮬레이터는 HW/SW 통합 설계 환경을 제공해줄 뿐만 아니라, 전력 모델도 지원한다.

PowerSim 시뮬레이터는 SSD 내부 동작을 모듈화 하여 각 모듈에 대해 하드웨어 가속 여부를 결정할 수 있다. 이를 통해 HW/SW 통합 설계 환경을 제공하여 다양한 SSD 컨트롤러를 구성하여 시뮬레이터에서 성능평가 및 기능검증이 가능하다. 또한, 각 모듈 별 수행시간 및 전력소모를 모델링하여 구성한 SSD 컨트롤러에 대한 종합적인 전력소모 및 수행시간을 확인할 수 있다. 낸드 플래시 제공업체는 비용, 성능, 전력소모를 고려하여 클라이언트의 요구사항에 가장 적합한 SSD 컨트롤러 탐색이 가능하다.

제 2 절 SSD 컨트롤러 모델링

매개변수	값
총 코어 수	정수
코어 스펙	1GHz/100MHz
각 코어 별 담당 모듈	1 ~ 5
각 모듈 별 HW 가속 여부	True/False
HW 가속화 모듈 수	정수

표 1. SSD 컨트롤러 구성 매개변수

제안된 시뮬레이터를 통해 낸드 플래시를 설계하는 도구 사용자는 표 1의 SSD 컨트롤러 구성 매개변수를 통해 다양한 SSD 컨트롤러 구축이 가능하다. 총 코어 수는 SSD 컨트롤러에서 사용할 코어 수를 결정한다. 코어 스펙을 통해 사용하는 코어에 대한 스펙을 설정할 수 있으며, 현재 1GHz 또는 100MHz 코어를 지원하고 있다. 이는 필요에 따라 다양한 코어를 지원이 가능하도록 확장이 가능하다. 코어에 따라 상이한 수행시간과 전력 소비를 가진다. 각 코어 별 담당 모듈 또한 코어 별로 설정이 가능하다.

이를 통해 소프트웨어 모듈에 대한 다양한 SSD 컨트롤러 구성이 가능하다. 다양한 정책과 같이 유연하고 복잡한 작업이 필요한 FTL 메인 코어의 경우 1GHz의 코어로 구성하고 나머지 모듈들에

대해서 100MHz의 간단한 코어로 구성을 할 수 있다 (그림 4). 타 모듈에 비해 많은 연산으로 보다 긴 수행시간을 차지하는 사용자 I/O 파싱과 캐시 탐색 모듈에 대해서는 코어 스펙을 1GHz로 업그레이드한 구성도 가능하다 (그림 3). 또한, 연산 수행 시간이 매우 짧은 사용자 I/O 반환 모듈에 대해 하나의 코어가 전담하여 사용하는 것이 낭비가 될 경우 100MHz 코어 두 개를 주소 변환과 사용자 I/O 반환 모듈을 함께 담당하도록 하여 코어 사용률을 높일 수 있다 (그림 5).

HW/SW 통합 설계를 위해서는 표 1의 각 모듈 별 HW 가속 여부와 HW 가속화 모듈 수 매개 변수를 통해 SSD 컨트롤러 구성이 가능하다. 기존의 소프트웨어 중심 SSD 컨트롤러에서 병목이라고 판단되는 모듈에 대하여 하드웨어 가속 구성이 가능하다 (그림 6). 또한, 임계 경로 전체에 대하여 완전히 하드웨어 가속 구성이 가능하며 (그림 7), 일부 계층에 대해서는 하드웨어 가속 모듈을 여러 개 배치할 수도 있다 (그림 8).

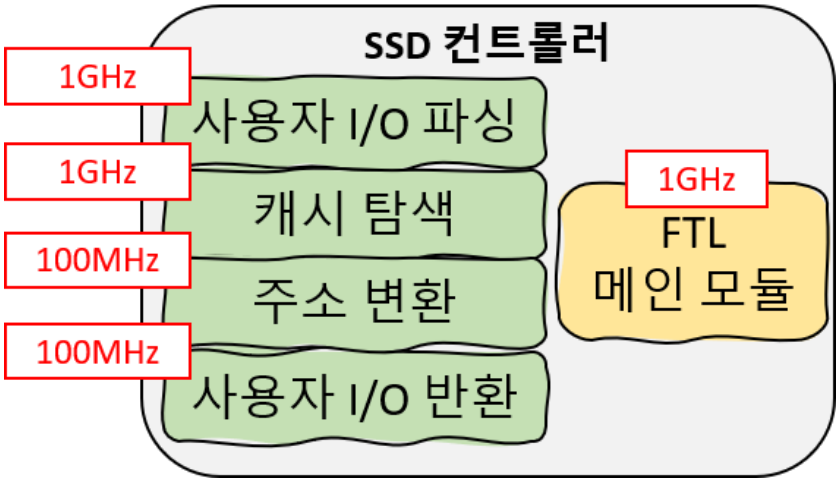

그림 3. SSD 컨트롤러 구성 1

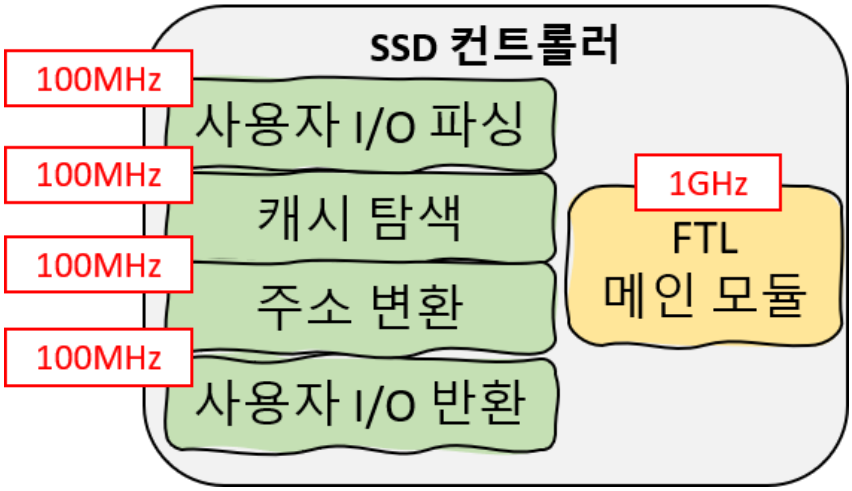

그림 4. SSD 컨트롤러 구성 2

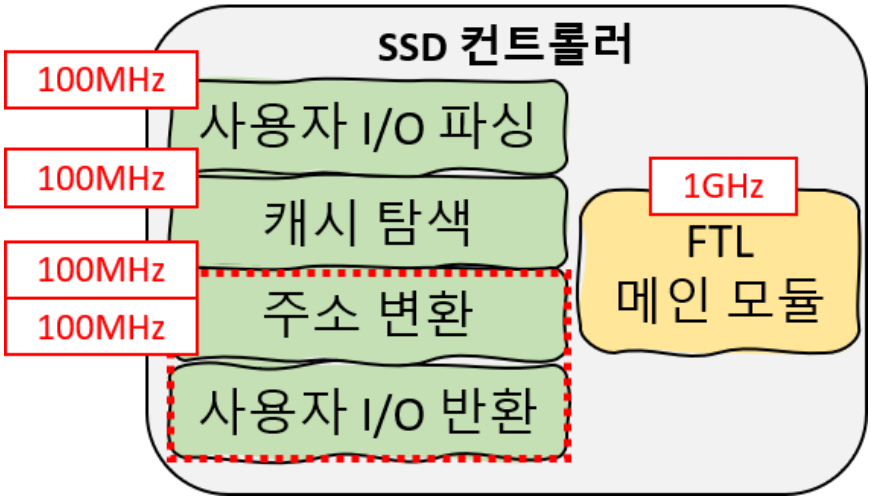

그림 5. SSD 컨트롤러 구성 3

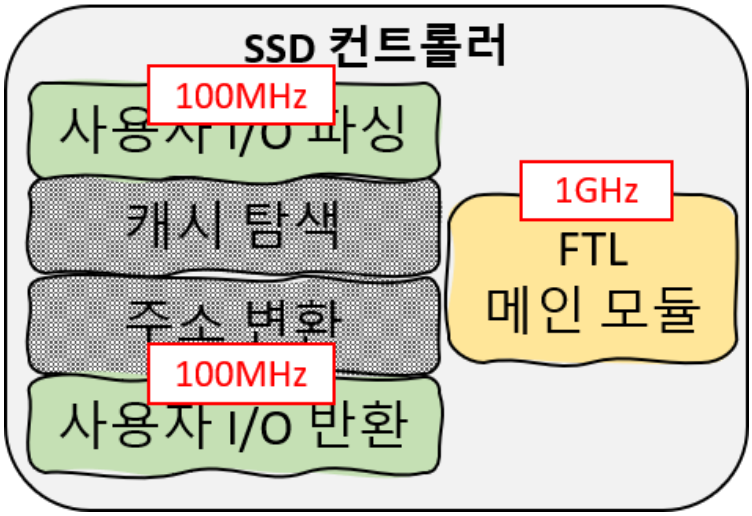

그림 6. HW/SW 통합 SSD 컨트롤러 구성 1

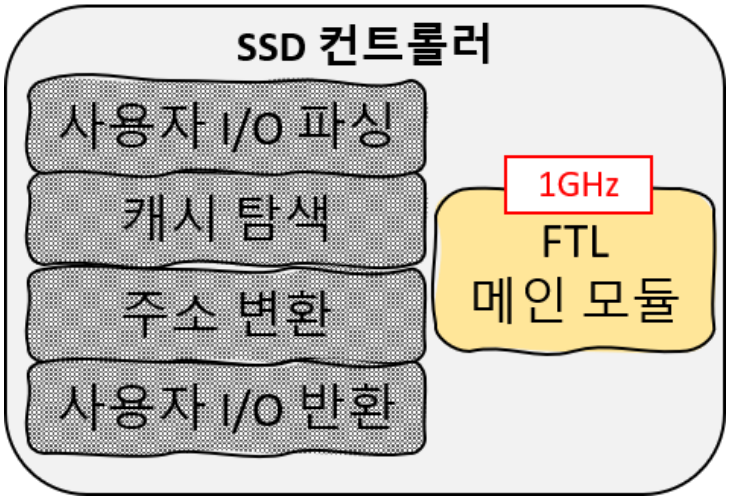

그림 7. HW/SW 통합 SSD 컨트롤러 구성 2

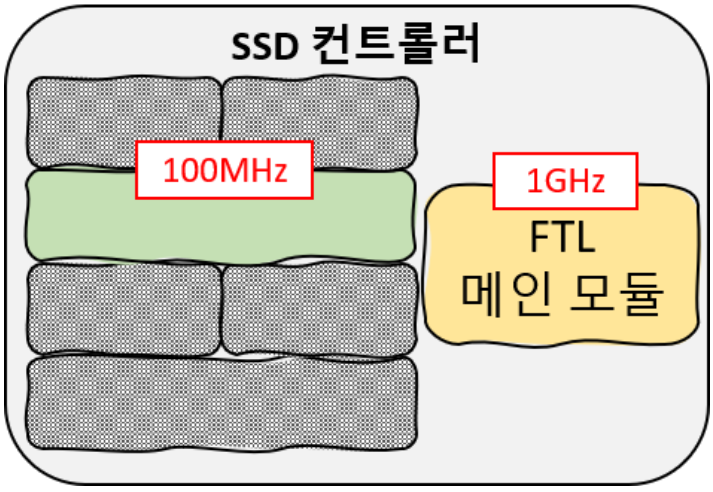

그림 8. HW/SW 통합 SSD 컨트롤러 구성 3

제 3 절 HW/SW 통합 설계 환경 구조

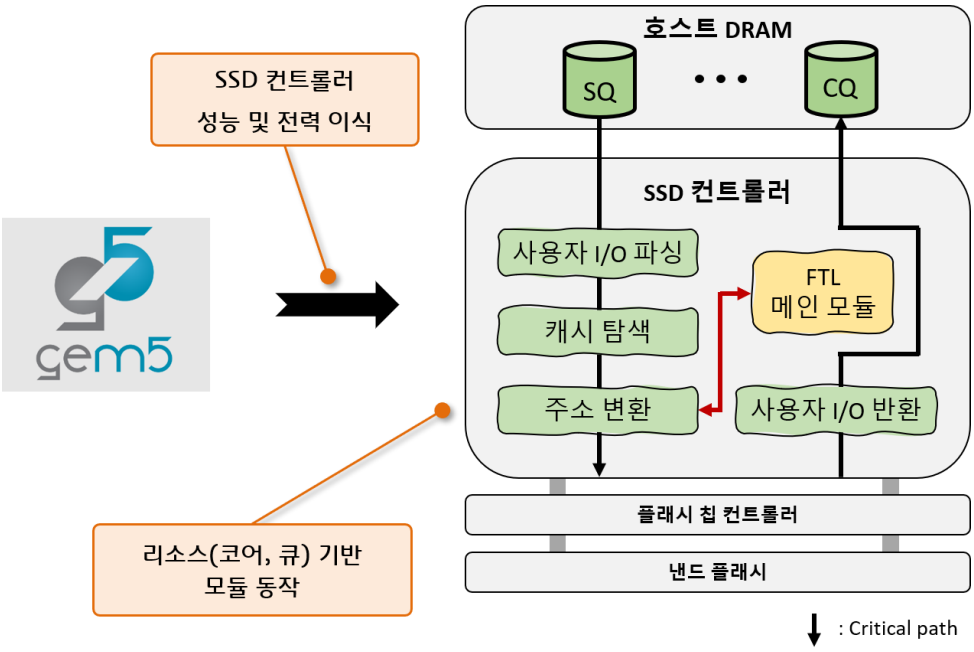

그림 9. HW/SW 통합 설계 환경 구조

제안한 HW/SW 통합 설계 환경을 위해 본 논문은 멀티 큐를 지원하는 SSD 시뮬레이터인 MQSim을 기반으로 구현하였다. HW/SW 통합 설계 환경의 전체 구조는 그림 9와 같다. 컴퓨터 시스템 시뮬레이터 플랫폼인 gem5 [10]를 이용하여 임베디드 환경을 구축하여 모듈화 한 작업들의 수행시간 및 전력을 측정한 후 PowerSim에 이식하게 된다.

각 모듈들은 gem5 측정 값을 기반으로 수행시간 및 전력이

고려되며, SSD 내부 자원(가용 가능 코어 수, 큐에 쌓인 요청 수) 상황에 맞게 동작하도록 MQSim의 이벤트 기반 시뮬레이션 동작을 확장하였다.

PowerSim 계층	이벤트 종류
PCIe	DELIEVER
호스트 인터페이스	IO_PARSING
데이터 캐시	MEMORY_READ_FOR_CACHE_EVICTION_FINISHED
	MEMORY_WRITE_FOR_CACHE_FINISHED
	MEMORY_READ_FOR_USERIO_FINISHED
	MEMORY_WRITE_FOR_USERIO_FINISHED
	CACHE_SLOT_SEARCH
주소 변환	ADDRESS_TRANSLATION
플래시 칩 컨트롤러	READ_DATA_TRANSFERRED
	READ_CMD_ADDR_TRANSFERRED
	PROGRAM_CMD_ADDR_DATA_TRANSFEREED
	PROGRAM_COPYBACK_CMD_ADDR_TRANSFERRED
	ERASE_SETUP_COMPLETED
	RETURN_USERIO
낸드 플래시	COMMAND_FINISHED
FTL 메인	ACTIVE_BLOCK_SELECTION
	GC_VICTIM_BLOCK_SELECTION

표 2. PowerSim 이벤트 종류

제 4 절 모듈 성능 및 전력 모델

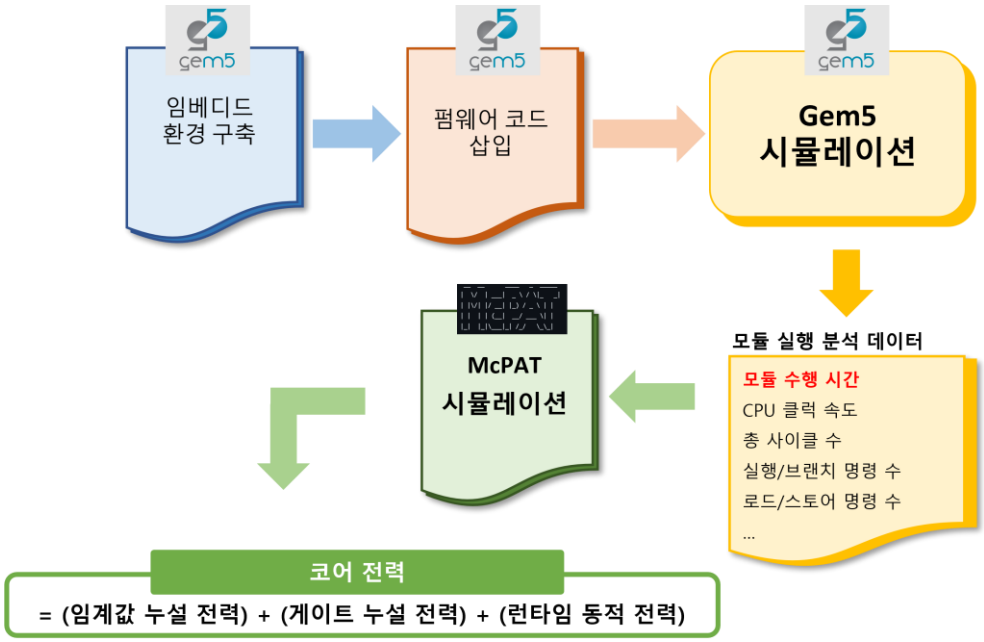

그림 10. 모듈 별 수행시간 및 전력 측정 과정

PowerSim의 SSD 컨트롤러 수행시간 및 전력 모델을 지원하기 위해서 컴퓨터 시스템 시뮬레이터 플랫폼인 gem5와 프로세서 전력 측정 도구인 McPAT [11]을 활용하였다. 각 모듈 별 수행시간 및 전력을 측정하기 위한 전체 과정은 그림 10과 같다.

임베디드 환경 구축 단계에서는 gem5를 이용하여 실제 임베디드 환경과 유사한 컴퓨터 시스템을 구축한다 (그림 11). ARM 코어는 L1 데이터 캐시와 L1 명령 캐시, L2 캐시를 가지며, 메모리 버스를 통해 DRAM과 통신하게 된다. 구성한 시스템은 시스템 구성

매개변수를 통해 필요한 임베디드 환경으로 구성이 용이하다. 표 3의 시스템 구성은 PowerSim에서 지원하는 1GHz ARM 코어의 수행시간 및 전력소모를 측정하였을 때의 매개변수이다.

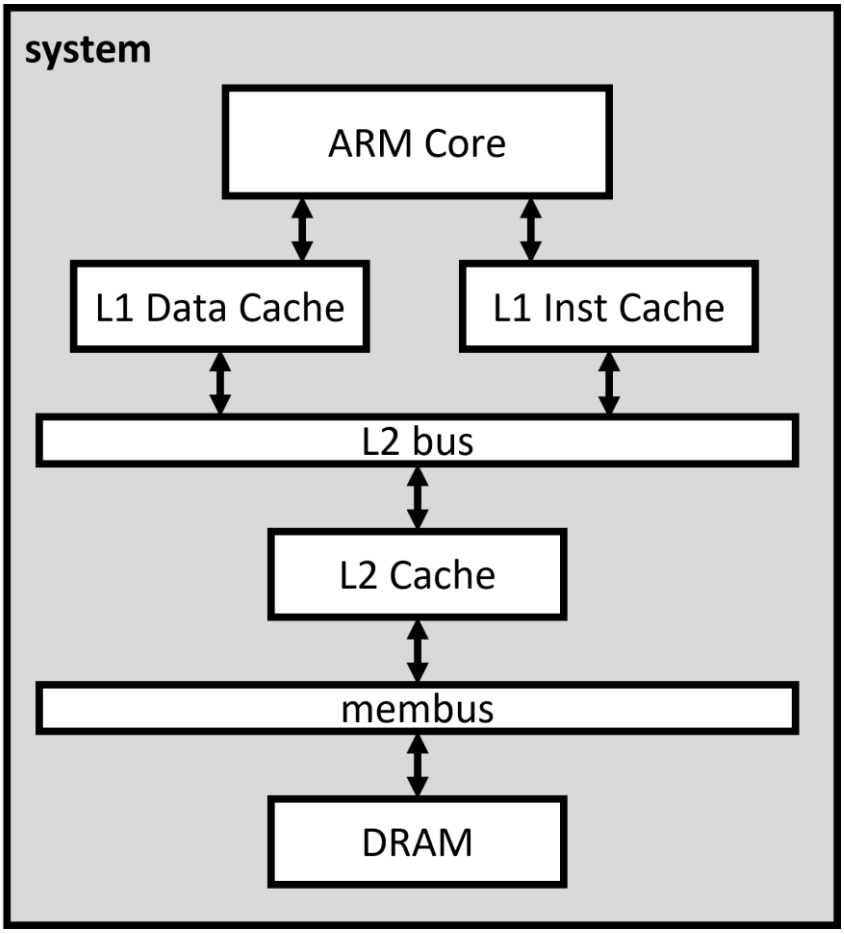

그림 11. Gem5를 이용한 수행시간/전력 평가 환경 구축

매개변수	값
Arm Core	1GHz
L1 Data Cache	32KB
L1 Inst Cache	32KB
L2 Cache	256KB

표 3. gem5 시스템 구성 매개변수

펌웨어 코드 삽입 단계에서는 MQSim의 모듈화 한 작업들에 대한 수행 코드를 구성한 시스템에 이식하게 된다. MQSim 시뮬레이터의 구현 코드는 단순히 시뮬레이션 동작을 위한 코드이기에 펌웨어 최적화가 전혀 고려되어 있지 않다. 단순히 MQSim 코드를 이식할 경우 비정상적으로 큰 수행 시간과 전력이 측정될 수 있다. 펌웨어 코드를 SSD 제조사에서 공개하지 않기 때문에 확인은 할 수 없지만, 코드 최적화 후의 코드를 이식하여 측정하였다. 또한, 구성한 시스템의 캐시 효과를 충분히 활용하기 위해 100번의 반복 수행 결과를 측정하였다.

gem5 시뮬레이션 이후 프로그램 수행 시간, CPU 클럭 속도, 총 사이클 수, 실행/브랜치 명령 수, 로드/스토어 명령 수, 캐시 미스 횟수, DRAM 접근 소모 전력 등이 포함되어 있는 데이터를 얻을 수 있다. 여기서 측정된 수행 시간이 PowerSim에 이식되어 사용자 I/O 요청 또는 트랜잭션 수에 기반한 모듈 수행 시간을 도출하게

된다.

gem5 시뮬레이션을 통해 얻은 데이터는 전력 소모를 측정하는데 활용된다. gem5 결과물을 McPAT 프로세서 전력 소모 측정 도구를 통해 시뮬레이션 하면 해당 코어의 임계값 누설 전력(Subthreshold Leakage), 게이트 누설 전력(Gate Leakage), 런타임 동적 전력(Runtime Dynamic)을 얻을 수 있다. 임계값 누설 전력과 게이트 누설 전력의 합이 유휴 전력(Idle Power)를 의미하며, 런타임 동적 전력과의 총 합이 해당 코어의 모듈 수행 전력이다.

제 6 장 실험

제 1 절 실험 환경

매개변수	값
ARM core 1GHz idle power	50 mW
ARM core 100MHz idle power	20 mW
DRAM idle power	30 mW
tR	45 us
tProg	390 us
tBERS	3.5 ms

표 4. PowerSim 매개변수

실험은 논문에서 구현한 PowerSim을 사용하였다. 고성능 FTL에 대한 HW/SW 통합 설계 환경을 평가하기 위해 삼성 PM9A3 960GB SSD [12]와 낸드 플래시 구성 및 페이지 읽기, 쓰기 그리고 블록 삭제 대기 시간을 동일하게 구성하였다 (표 4). PowerSim에서 지원하는 1GHz, 100MHz ARM 코어의 유휴 전력에 대해서는 스토리지 시스템에서 주로 사용되는 ARM Cortex-R 시리즈의 유휴 전력을 참고하였다 [13, 14]. 평가 워크로드는 임의 읽기(4KB, QD32) 합성 워크로드를 사용하였다.

제 2 절 요구 전력 소모량

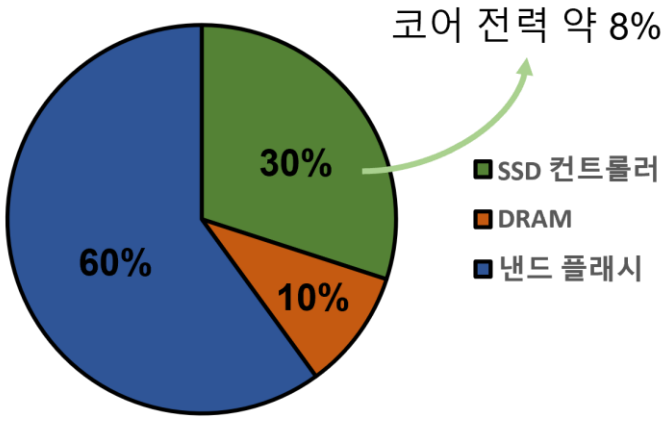

그림 12. SSD 전력 소모 비율

SSD는 구성 요소 별 전력 소모 비율은 용량과 워크로드에 따라 달라지지만 대략적인 비율은 그림 12와 같다. SSD 컨트롤러는 약 30%의 전력을 차지하며 그 중에서도 8% 정도 만이 SSD 내부 코어 전력 소모 비율이다. 삼성 PM9A3 960GB SSD는 8W 전력을 소모하며 그림 12의 비율을 바탕으로 전력 요구조건을 계산해보면, DRAM 요구 전력은 800mW 이하, 코어 요구 전력은 192mW 이하의 수치를 가져야만 한다.

제 3 절 SSD 컨트롤러 구성

본 연구에서 제안하는 PowerSim을 이용하여 다양한 SSD 컨트롤러 구성이 가능하며, 표 5는 SW 설계 모델, 표 6은 HW/SW 통합 설계 모델이다.

컨트롤러 1은 MQSim과 가장 유사한 성능을 나타내는 SSD 컨트롤러 구성이다. MQSim은 SSD 컨트롤러 수행시간을 고려하지 않기 때문에, PowerSim에서 사용하는 전력 모델을 이식하여 측정이 불가능하다. 그러므로 각 모듈 별로 ARM 코어 1GHz를 배치하여, 대기열에 요청이 쌓이지 않고 즉시 처리되도록 하였다.

컨트롤러 2는 전력 소모 완화를 위해 단순하고 반복적인 임계 경로 상의 작업에 대하여 ARM 코어 100MHz로 변경한 구성이다.

컨트롤러 3 ~ 6의 경우 임계 경로 상의 모듈에 대해 각각 하드웨어 가속기 하나로 대체한 SSD 컨트롤러 구성이다.

컨트롤러 7과 8은 임계 경로 상의 병목 지점이라 할 수 있는 높은 수행 시간을 차지하는 2개 또는 3개의 모듈에 대하여 하드웨어 가속기 하나로 대체한 구성이다.

컨트롤러 9와 10은 임계 경로에 대하여 완전 하드웨어 가속으로 대체하였으며, 컨트롤러 10은 비교적 높은 수행 시간을 가지는 모듈에 대하여 하드웨어 가속 모듈을 하나씩 더 추가하였다.

모델 이름	SSD 컨트롤러 구성
컨트롤러1	ARM core 1GHz * 5
컨트롤러2	ARM core 1GHz * 1 + ARM core 100MHz * 4

표 5. SSD 컨트롤러 SW 설계 모델

모델 이름	HW 가속화 여부 (HW 모듈 수)			
	사용자 I/O 파싱	캐시 탐색	주소 변환	트랜잭션 반환
컨트롤러3	O (1)	X	X	X
컨트롤러4	X	O (1)	X	X
컨트롤러5	X	X	O (1)	X
컨트롤러6	X	X	X	O (1)
컨트롤러7	O (1)	O (1)	X	X
컨트롤러8	O (1)	O (1)	O (1)	X
컨트롤러9	O (1)	O (1)	O (1)	O (1)
컨트롤러10	O (2)	O (2)	O (2)	O (1)

HW 가속화가 되어 있지 않은 모듈은 ARM core 100MHz 기반 동작

표 6. SSD 컨트롤러 HW/SW 통합 설계 모델

제 4 절 실험 결과

그림 13은 PowerSim을 이용하여 구성한 SSD 컨트롤러 별 성능(IOPS) 및 전력 측정 결과를 보여준다. MQSim과 유사한 환경인 컨트롤러1은 실제 삼성 PM9A3 960GB SSD에 비해 조금 낮은 IOPS를 달성하지만 공격적으로 최적화가 되지 않은 기본적인 형태의 SSD 시뮬레이터이기에 이를 고려한다면, MQSim의 성능 모델은 적절히 잘 동작한다고 볼 수 있다. 하지만 SSD의 전력 요구조건은 전혀 고려하고 있지 않기 때문에, 비정상적인 코어 전력 소모를 가진다.

컨트롤러2는 의도한 바와 같이 코어 전력이 41% 감소하여 전력 소모를 완화할 수 있었으나, IOPS 또한 19% 낮아진 수치를 기록하였다. 최신 SSD의 고성능 FTL을 지원하기 위해 소프트웨어 중심 SSD 컨트롤러 설계는 성능과 전력적인 측면 모두 만족하기에는 많은 어려움이 존재한다.

컨트롤러 3-6의 구성에서는 임계 경로 중 가장 긴 수행시간을 차지하는 데이터 캐시 탐색 모듈을 하드웨어 가속하였을 때 가장 높은 개선율을 보였으나, 아직 코어 전력 요구조건에는 상회하는 수치이다.

컨트롤러 7-10은 모두 삼성 PM9A3 960GB SSD의 전력

요구조건을 만족하고 있는 구성이다. 임계 경로에 대하여 완전 하드웨어 가속을 한 컨트롤러 9, 10은 가장 높은 성능과 가장 낮은 전력 소모를 보여주고 있으며 에너지 소모량 또한 그림 14에서 보는 것 같이 가장 낮은 측정치를 가진다.

그림 13의 실험 결과를 미루어 보면 임계 경로에 대하여 완전 하드웨어 가속을 하는 것이 가장 이상적인 것처럼 보인다. 하지만 SSD를 설계하는데 있어 성능과 전력 외에도 SSD 제조사는 개발 비용, 구현 복잡도, 모듈의 유연성 등 다양한 요구조건이 존재하기에 컨트롤러 9, 10이 가장 이상적인 SSD 컨트롤러 구성이라고 단정지을 수 없다. 성능/전력 요구조건을 만족하는 SSD 컨트롤러에 대하여 클라이언트의 요구조건과 제조사의 상황에 맞는 다양한 SSD 컨트롤러를 탐색할 수 있다.

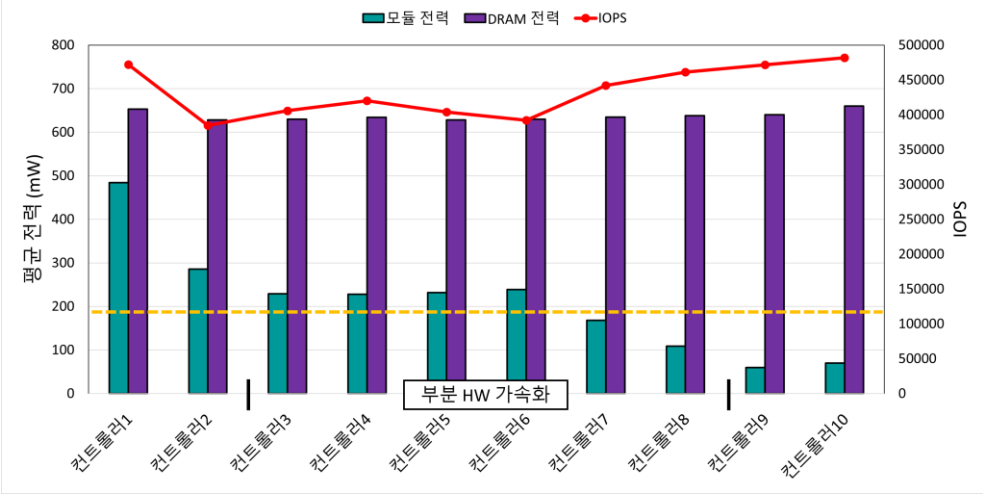

그림 13. SSD 컨트롤러 구성 별 성능 및 전력

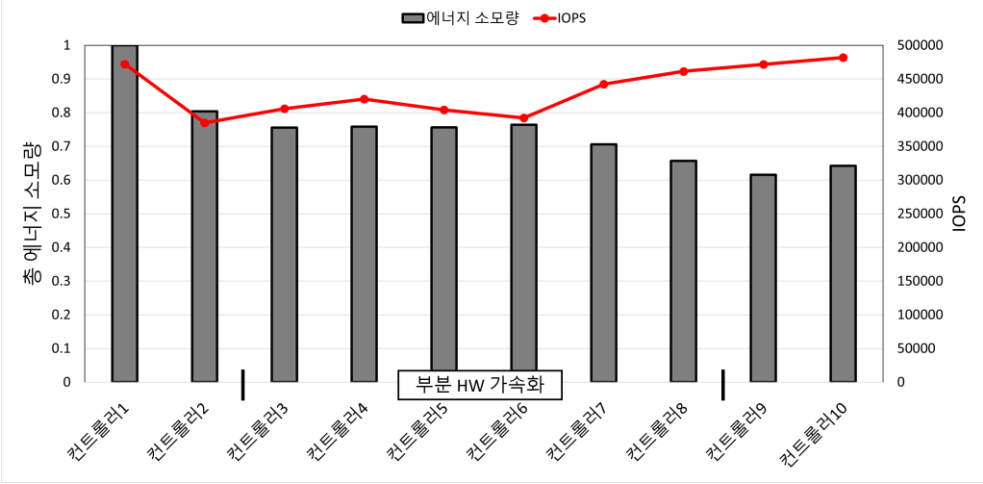

그림 14. SSD 컨트롤러 구성 별 총 에너지 소모량

제 7 장 결론

제 1 절 결론 및 향후 연구

본 논문에서는 하드웨어 가속 모듈을 활용한 고성능 FTL 하드웨어/소프트웨어 통합 설계 환경인 PowerSim을 제안한다. PowerSim을 이용하여 소프트웨어 설계에 있어서는 코어 수, 코어 스펙 등을 설정할 수 있으며 하드웨어 설계에 있어서는 하드웨어 가속 여부, 하드웨어 가속 모듈 수를 설정하여 다양한 하드웨어/소프트웨어 통합 SSD 컨트롤러 구성이 가능하다. 또한, 구성한 SSD 컨트롤러에 대한 성능 및 전력 평가를 통해 클라이언트 요구조건에 부합한 SSD 컨트롤러 탐색이 가능하다.

향후 연구로는 SSD 컨트롤러 전력에서 코어 외에 ECC, PCIe 등과 같은 작업에 대한 전력 모델을 추가할 것이며, 더 나아가 낸드 플래시의 전력 모델까지 확장하여 SSD 전체 전력에 대한 모델을 지원하는 시뮬레이터를 제안할 계획이다.

참고 문헌

[1] Samsung Corporation. 990 PRO NVMe 1TB. https://www.samsung.com/sec/memory-storage/990-pro-nvme-ssd/MZ-V9P1T0BW/, 2022.

[2] Arash Tavakkol, Juan Gomez-Luna, Mohammad Sadrosadati, Saugata Ghose, and Onur Mutlu. MQSim: A framework for enabling realistic studies of modern multi-queue SSD devices. In *16th USENIX Conference on File and Storage Technologies (FAST)*, pages 49-66. 2018.

[3] Huaicheng Li, Mingzhe Hao, Michael Hao Tong, Swaminatahan Sundararaman, Matias Bjorling, and Haryadi S. Gunawi. The CASE of FEMU: Cheap, accurate, scalable and extensible flash emulator. In *16th USENIX Conference on File and Storage Technologies (FAST)*, pages 83-90. 2018.

[4] Jaewook Kwak, Sangjin Lee, Kibin Park, Jinwoo Jeong, and Yong Ho Song. Cosmos+ OpenSSD: Rapid Prototype for Flash Storage Systems. *ACM Transactions on Storage*, pages 35. 2020.

[5] Dirik, Cagdas, and Bruce Jacob. The performance of PC solid

state disks (SSDs) as a function of bandwidth, concurrency, device architecture, and system organization. *ACM SIGARCH Computer Architecture News*, pages 279–289. 2009.

[6] Wei Liu, Kai Wu, Jialin Liu, Feng Chen, and Dong Li. Performance Evaluation and Modeling of HPC I/O on Non-Volatile Memory. In *2017 International Conference on Networking, Architecture, and Storage (NAS)*, pages 1–10. 2017.

[7] Marvell Corporation. Marvell 88SS1093 Flash Memory Controller. https://www.marvell.com/storage/assets/Marvell-88SS1093-0307-2017.pdf, 2017.

[8] Jie Zhang, Miryeong Kwon, Michael Swift, and Myoungsoo Jung. Scalable Parallel Flash Firmware for Many-core Architectures. In *18th USENIX Conference on File and Storage Technologies (FAST)*, pages 121–136. 2020.

[9] Benini, Luca, Alessandro Bogliolo, and Giovanni De Micheli. A survey of design techniques for system-level dynamic power management. *IEEE transactions on very large scale integration (VLSI) systems*, pages 299–316. 2000.

[10] Jason Lowe-Power, Abdul Mutaal Ahmad, Ayaz Akram,

Mohammad Alian, Rico Amslinger, Matteo Andreozzi, ..., and Eder F. Zulian. The gem5 simulator: Version 20.0+. https://arxiv.org/pdf/2007.03152.pdf, 2020.

[11] Sheng Li, Jung Ho Ahn, Richard D. Strong, Jay B. Brockman, Dean M. Tullsen, and Norman P. Jouppi. The McPAT Framework for Multicore and Manycore Architectures: Simultaneously Modeling Power, Area, and Timing. *ACM Transactions on Architecture and Code Optimization (TACO)*, pages 1-29. 2013

[12] Samsung Corporation. PM9A3 960GB, https://www.techpowerup.com/ssd-specs/samsung-pm9a3-960-gb.d1259, 2021.

[13] Arm Corporation. Cortex-R4, https://developer.arm.com/Processors/Cortex-R4, 2004.

[14] Arm Corporation. Cortex-R5, https://developer.arm.com/Processors/Cortex-R5, 2011.

Abstract

PowerSim: A HW/SW Codesign Environment for Modern High-Performance FTLs with HW Accelerated Modules

Soonwon Choi
Department of Computer Science & Engineering
College of Engineering
Seoul National University

The performance of Solid-State Drives (SSDs), which are widely used as storage devices in various computer systems, has been steadily increasing through various optimization studies such as improvements in host interfaces and increasing channel/chip bandwidth. To achieve the high-performance functionality of modern SSDs, traditional software-centric Flash Translation Layer (FTL) designs require more powerful

and numerous cores to meet performance requirements. However, this approach presents the problem of excessive power consumption.

To satisfy both performance and power requirements, there is a growing trend of hardware acceleration for critical paths. While hardware designs offer advantages in performance and power, they come with higher development costs and implementation complexities. Therefore, an integrated design that combines the strengths and weaknesses of hardware designs with high flexibility for implementing various algorithms in software designs is necessary.

This thesis proposes a hardware/software co-design for high-performance FTL, utilizing hardware acceleration modules and introducing the PowerSim simulator, which considers power models. For software design of the SSD controller, it allows the configuration of the number of cores and core specifications, while for hardware design, it enables the setting of hardware acceleration availability and the number of hardware acceleration modules, facilitating various hardware/software

co-designs and evaluations.

To implement the proposed hardware/software co-design, we extend the MQSim simulator, a multi-queue SSD simulator, and develop a simulation tool. We configure the NAND flash similarly to the latest Samsung PM9A3 960GB SSD with high-performance FTL and set power requirements to validate the effectiveness of PowerSim. Through PowerSim, we construct and experiment with various hardware/software integrated SSD controllers, evaluating their performance and power consumption. By presenting diverse controllers that meet the requirements, manufacturers can explore SSD controller designs that align with client demands and their own circumstances.

Keywords : NAND flash memory-based storage devices, SSD simulator, Power model, HW/SW co-design, Hardware acceleration
Student Number : 2021-29273