# 인텔 비휘발성 메모리 기술 동향

## Trend of Intel Nonvolatile Memory Technology

**이용섭** (Y.S. Lee, yongseob@etri.re.kr) 클라우드기반SW연구실 박사후연수연구원
**우영주** (Y.J. Woo, youngjoo@etri.re.kr) 클라우드기반SW연구실 박사후연수연구원
**정성인** (S.I. Jung, sijung@etri.re.kr) 클라우드기반SW연구실 책임연구원

### ABSTRACT

With the development of nonvolatile memory technology, Intel has released the Optane datacenter persistent memory module (DCPMM) that can be deployed in the dual in-line memory module. The results of research and experiments on Optane DCPMMs are significantly different from the anticipated results in previous studies through emulation. The DCPMM can be used in two different modes, namely, memory mode (similar to volatile DRAM: Dynamic Random Access Memory) and app direct mode (similar to file storage). It has buffers in 256-byte granularity; this is four times the CPU (Central Processing Unit) cache line (i.e., 64 bytes). However, these properties are not easy to use correctly, and the incorrect use of these properties may result in performance degradation. Optane has the same characteristics of DRAM and storage devices. To take advantage of the performance characteristics of this device, operating systems and applications require new approaches. However, this change in computing environments will require a significant number of researches in the future.

**KEYWORDS** 옵테인, 저장장치, 비휘발성 메모리

## Ⅰ. 서론

컴퓨터에서 기억장치(Memory)는 연산장치(Processing unit)에서 사용하는 데이터를 보관한다. 이러한 기억장치는 주기억장치(Primary memory 또는 Main memory)와 보조기억장치(Secondary memory)로 구분한다. 주기억장치는 보조기억장치보다 높은 대역폭과 빠른 성능 특성이 있다. 연산장치는 주기억장치에 직접 접근하여 원하는 데이터를 사용하지만, 보편적으로 사용하고 있는 보조기억장치에 저장된 데이터는 연산장치가 직접 접근할 수 없어 주기억장치에 옮긴 후 사용한다. 전원 공급이

* DOI: https://doi.org/10.22648/ETRI.2020.J.350306
* 이 논문은 2020년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임[No. 2014-3-00035, 매니코어 기반 초고성능 스케일러블 OS 기초연구(차세대 OS 기초연구센터)].

차단되었을 때 주기억장치의 데이터는 사라지기 때문에 보조기억장치는 전원 공급 중단에도 데이터를 영구적으로 보관할 수 있는 특성이 있는 장치를 사용하게 된다.

데이터를 공유하는 데 효과적인 형태인 파일 사용이 활성화되고 파일 시스템이 발전함에 따라 보조기억장치는 파일을 보관하는 저장장치(Storage)로 진화하였다. 주기억장치와 비교해 여전히 느린 보조기억장치(또는 파일 저장장치, 이하 저장장치로 표현)의 사용은 컴퓨터 시스템의 성능을 떨어뜨리는 주요 요소로 남아 있다. 이러한 문제를 해결하기 위하여 자주 사용되는 데이터를 주기억장치에 잠시 보관하여 연산장치가 직접 접근하여 사용할 수 있게 하는 캐싱 기법을 널리 사용하고 있다. 컴퓨팅시스템의 성능을 개선하기 위해 빠른 저장장치를 사용하는 것은 필수 조건이 되었다.

마그네틱을 사용하는 저장장치보다 빠른 처리가 가능한 플래시 기반의 저장장치 사용이 늘어나면서 더욱 빠르게 데이터에 접근할 수 있는 인터페이스를 사용하는 기술이 적용됐다. 최근에는 메모리 버스를 통해 사용할 수 있는 저장장치를 필요로 하는 분야가 늘어나고 있다. 이러한 요구에 따라 NVDIMM(Non-volatile Dual-inline Memory Module)[1]과 같은 제품이 만들어지기도 하였다.

최근 인텔사에서 옵테인 데이터센터 영구 메모리 모듈(Optane Datacenter Persistent Memory Module, DCPMM)을 출시함에 따라, 연산장치가 직접 대용량의 영구 저장장치에 접근하여 사용할 수 있는 환경이 제공되기 시작하였다.

본 고에서는 가장 최신의 바이트(Byte) 단위로 접근 가능한 영구 메모리 저장장치인 인텔의 옵테인 DCPMM의 구조, 성능 특성과 활용 분야에 대해 알아보고자 한다.

## Ⅱ. 메모리 저장장치의 발전

### 1. 휘발성 메모리

휘발성 메모리(Volatile memory)는 전원 공급이 유지되는 동안 데이터를 보관할 수 있는 기억장치를 의미한다. 휘발성 메모리에는 레지스터, SRAM(Static Random Access Memory), DRAM(Dynamic Random Access Memory) 등이 있으며, 레지스터와 SRAM은 연산장치와 함께 단일 칩 형태로 존재하고 DRAM은 메모리 버스를 통해 CPU와 연결하여 사용된다.

레지스터는 가장 빠른 휘발성 메모리로 메모리 계층의 최상위층에 위치한다. SRAM은 레지스터 다음으로 빠른 성능을 갖고 있어 주로 연산장치의 캐쉬로 사용한다. DRAM과 다르게 정적으로 상태를 유지하므로 전력 소모량이 적다. 그러나 DRAM보다 면적당 저장용량이 적어 대용량으로 사용하기에 적합하지 않다. DRAM은 연산장치 외부에 위치하며, 커패시터(Capacitor)의 전하 방전에 따른 데이터 손실 방지를 위해 주기적으로 전하를 충전하는 과정이 필요하므로 전력 소모량이 많다. 그러나 단위면적당 높은 저장용량을 구현할 수 있어 휘발성 메모리 중 저렴한 비용으로 높은 저장 공간을 사용할 수 있는 장점이 있다. DRAM은 저장장치를 사용할 때 발생하는 느린 접근시간을 가려주는 캐쉬 역할도 한다. 초기의 DRAM은 연산장치와 동기화 기능이 없어 메모리 제어기는 원하는 데이터 처리 상태를 알 수 없으므로 기다려야 했다. 연산장치와 메모리 간의 동기화가 가능해진 SDRAM(Synchronous Dynamic Random Access Memory)의 개발로 이러한 문제가 해결되기 시작하였다. 내부와 외부에 동작하는 동기화 클럭이 같아 SDR SDRAM(Single Data Rate SDRAM)으로도

불렀으며, 한 클럭 사이클에 하나의 명령만 수행하고 종료될 때까지 다음 명령을 수행할 수 없는 특징이 있다. 이후 한 클럭 사이클에 2번 데이터를 전송할 수 있는 DDR SDRAM(Double Data Rate SDRAM)으로 발전하였고, 최근에는 전력소비량을 낮추고 데이터 전송량은 높은 DDR4 SDRAM을 주 기억장치로 많이 사용하고 있다.

### 2. 비휘발성 메모리

비휘발성 메모리(Non-volatile memory)는 전원 공급 중단에도 이미 기억하고 있는 데이터를 보존한다. 가장 널리 사용하고 있는 비휘발성 메모리인 플래시 메모리(Flash memory)는 1987년 일본의 도시바에서 상품으로 출시하였다. 플래시 메모리에 쓰기를 할 때 해당 영역에 이미 쓰인 것이 있으면 삭제 과정을 선행하기 때문에 상대적으로 수명이 짧은 단점이 있다. 이후 FeRAM(Ferroelectric Random Access Memory), MRAM(Magnetic RAM), PCM(Phase Change Memory), Memrister, STT-RAM(Spin Transfer Torque Magnetoresistive RAM), ReRAM(Resistive RAM), 3D XPoint 등과 같은 다양한 비휘발성 메모리를 연구하였다.

MRAM은 DRAM처럼 전하 충전으로 저장하지 않고 자기장의 특성을 이용하여 데이터를 저장한다. STT-RAM은 MRAM의 일종으로 DRAM과 비슷하거나 더 좋은 지연 시간을 갖고 있어 CPU 캐쉬로 사용할 수 있지만, 단위 면적당 저장용량이 DRAM보다 작은 단점이 있다. PCM이나 ReRAM의 경우 집적도가 높아 대용량 저장장치로 사용하기 유리하지만, DRAM보다는 접근 지연 성능이 떨어지는 단점이 있다. 3D XPoint는 플래시 메모리보다 높은 성능을 갖고 있으며 집적도가 높아 대용량 메모리 저장장치로 활용할 수 있는 장점이 있다.

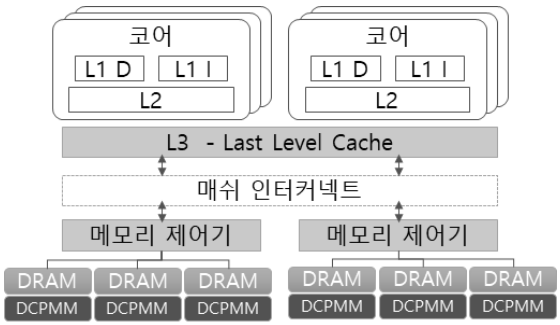
인텔의 비휘발성 메모리 저장장치인 옵테인 DCPMM은 PCM 기술을 기반으로 하여 만들어졌다. DRAM보다 낮은 성능 특성을 갖지만, 고밀도이고 비휘발성이며 바이트 단위로 접근 가능한 특성이 있다.

## III. 비휘발성 메모리 기술 현황

### 1. 옵테인 DCPMM 구조 및 성능

옵테인 DCPMM 지원하는 인텔 Cascade Lake CPU는 그림 1과 같은 메모리 계층 구조로 되어 있다. 2개의 메모리 제어기(Memory controller)가 있고, 각 메모리 제어기마다 3개의 채널을 제공한다. 각 채널에는 2개의 메모리 슬롯이 있지만, DCPMM은 각 채널마다 최대 1개씩 설치할 수 있다. 즉, 연산장치 소켓 별로 최대 6개의 DCPMM을 설치할 수 있다. DCPMM은 모듈당 128, 256, 512GB의 총 3가지 용량이 있으며, 하나의 연산장치로 구성 가능한 최대 용량은 3TB이다. 최대 8개의 연산장치로 구성 가능한 다중처리기 시스템은 총 24TB 메모리 영역을 제공할 수 있다.

DCPMM은 메모리 모드(Memory mode)와 앱 다




출처 Reproduced from S. Scargall, "Persistent Memory Architecture. In: Programming Persistent Memory," 2020, Apress, Berkeley, CA, CC BY 4.0.

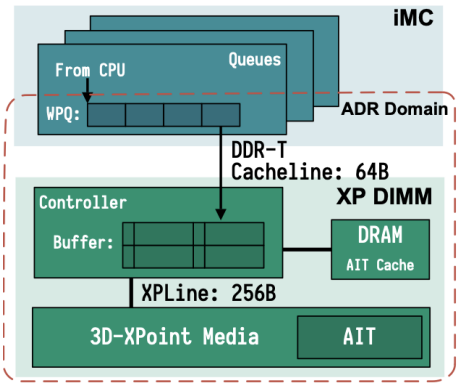
그림 1 CPU 캐쉬와 메모리 계층 구조

이렉트 모드(App direct mode) 2가지를 제공한다. 메모리 모드는 응용의 변경 없이 거대한 메모리 용량을 저렴하게 사용하려는 목적에 적합하다. 이 모드에서는 DCPMM을 주기억장치로 사용하고, 기존의 DRAM은 CPU의 직접 사상(Direct mapped) 방식의 근접 메모리(Near memory)로 동작한다[2]. 이 모드는 전원을 인가할 때마다 새로운 키로 데이터를 암호화한다. 따라서 이전에 저장한 데이터를 복호화할 수 없으며, 장치의 비휘발성은 무의미하다.

앱 다이렉트 모드에서는 운영체제가 DCPMM을 SSD나 HDD와 같은 저장장치로 인식한다. 따라서 사용자는 블록 저장장치 기반의 API를 통해 DCPMM을 매우 빠른 스토리지로 사용할 수 있다. 그리고 새로운 API를 사용하면 기존의 I/O 스택을 거치지 않고 load, store 명령어로 바이트 단위로 직접 접근할 수 있다. 앱 다이렉트 모드에서는 메모리 제어기가 메모리 접근 명령을 DRAM을 거치지 않고 DCPMM에 직접 수행한다. 일부 영역을 앱 다이렉트 모드로, 나머지는 메모리 모드로 활용하는 혼합 모드(Mixed mode)도 제공한다. 혼합 모드는 메모리의 데이터를 스토리지에 저장할 때 I/O 버스로 인한 지연 시간(Latency) 증가가 발생하지 않는다.

앱 다이렉트 모드에서 각 채널에 연결된 DCPMM들은 각각 인터리빙(Interleaving)을 통해 하나의 장치처럼 사용하거나, 별개의 장치로 사용할 수 있다. 하나 이상의 DCPMM을 묶어 관리하는 물리적인 연속 영역을 리전(Region)이라고 한다. 인터리빙은 인접한 가상주소의 페이지들을 해당 리전에 속한 장치들로 분산시키는 하드웨어 수준의 병렬화로 메모리 대역폭을 증가시켜준다. 이는 RAID 0와 유사하다. 인텔은 동작 모드 설정 및 리전의 생성과 관리 도구로 ipmctl 유틸리티를 제공하고 있다. 리전에서 할당받은 영역을 이름공간(Namespace)이라고 하며 ndctl 유틸리티를 통해 관리할 수 있다. 운영체제는 이름공간을 I/O 디바이스로 인식하기 때문에 파일 시스템을 설치하여 사용 가능하다.

응용이 기존 I/O 스택을 사용하지 않고 직접 접근하고자 하는 경우, DAX(Direct Access) 기반 파일 시스템을 설치하고 mmap 시스템호출 기능을 사용하여 DCPMM 공간을 할당받아 사용한다. 이때, 응용은 실행 중 갑자기 전원이 차단된 이후에도 DCPMM에 쓰인 데이터의 무결성을 보장받기를 원한다. 이를 위해 영구 지역(Persistent domain)을 지원한다. 영구 지역에 데이터가 도착하면, 이후 발생하는 전원 차단에도 데이터의 영구적 저장을 보장한다. 인텔은 그림 2와 같이 ADR(Asynchronous DRAM Refresh) 기능을 통해 CPU의 메모리 제어기에 있는 WPQ(Write Pending Queue)와 DCPMM을 포함하는 영역을 영구 지역으로 보장하고 있다. 따라서 캐쉬 플러쉬 명령어(clflush, clflushopt, clwb)와 메모리 배리어 명령어(sfence,




출처 Reprinted from J. Yang et el., "An Empirical Guide to the Behavior and Use of Scalable Persistent Memory," 2019, arXiv.org (Vol. cs.DC), CC BY 4.0.

그림 2 옵테인 DCPMM의 내부 구조 및 영구 지역 (그림의 점선 부분)

mfence)를 통해 데이터가 WPQ에 도달하기만 하면, 해당 데이터는 DCPMM에 영구적으로 저장되는 것이 보장된다.

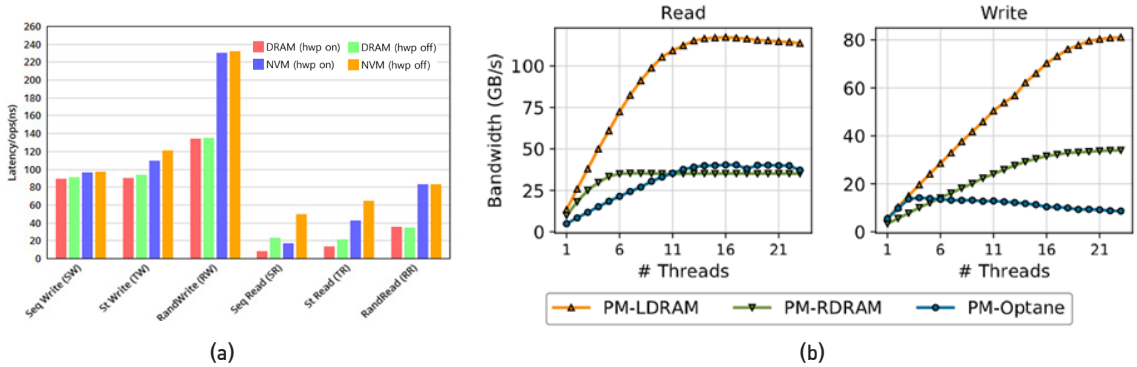
메모리 제어기의 WPQ는 CAM(Content Addressable Memory)으로 구현되어 있으므로, 아직 메모리에 쓰이지 않은 데이터를 바로 읽어 가거나 같은 메모리 영역에 반복한 쓰기 동작을 병합할 수 있다. WPQ에 저장된 데이터는 64바이트 캐쉬 라인 크기에 맞춰 DCPMM 제어기로 보내진다. DCPMM 제어기는 내부 버퍼를 활용하여 64바이트 쓰기 4개를 256바이트 쓰기 1개로 묶어서 실제 미디어로 내려 보낸다. 또한 AIT(Address Indirection Table)을 보유하고 있어, 실제 데이터가 저장되는 위치를 조정하여 장치의 수명을 관리하고 있다. 이 과정을 웨어 레벨링(Wear leveling)이라고 한다.

DCPMM은 제어기 내부에서 수행되는 쓰기 병합이나 주소 변환 등의 동작으로 인해 DRAM과 매우 다른 성능 특성 양상을 보인다. 또한 메모리 제어기의 WPQ나 DCPMM 제어기의 버퍼가 병목 지점이 되어 성능 저하를 유발하기도 한다. 이에 관한 자세한 성능 분석을 다음 절에서 소개하고자 한다.

## 2. 기본 성능

### 가. 지연 시간(Latency)

그림 3은 다양한 조건의 지연 시간 실험이다. DCPMM의 읽기 지연 시간은 DRAM에 비해 2~3배 높지만, 쓰기 지연 시간은 비슷하다[3]. 읽기에서 DCPMM의 지연 시간은 순차, 무작위 순으로 높다. 쓰기에서는 무작위 패턴이 높고 순차는 거의 비슷하다. 시스템의 응답 시간에 영향을 미치는 꼬리응답시간(Tail latency)은 안정적이다. clflush, clflushopt, clwb, non-temporal store 등의 메모리 명령어에 대한 지연 시간은 clflushopt, clwb가 우수하다. 그림 3(a)와 같이 HW prefetcher가 지연 시간에 미치는 영향 분석에 따르면, DRAM과 비교하여 prefetcher를 끄면 DCPMM의 읽기 성능은 3배 더 나빠진다. 쓰기 경우도 읽기와 유사한 성능 차이가 있지만, 쓰기 flush 오버헤드로 prefetcher를 끄면서 발생하는 성능의 영향이 보이지 않는다. 이는 DCPMM 성능을 향상시키기 위해 prefertcher 친화적인 코딩이 필요함을 의미한다.




출처 Reprinted from J. Izraelevitz et al., "Basic Performance Measurements of the Intel Optane DC Persistent Memory Module," 2019, arXiv:1903.05714, CC BY 4.0.

그림 3 옵테인 DCPMM 성능: (a) 지연 시간 성능 비교 (b) 대역폭 성능 비교

### 나. 대역폭(Bandwidth)

인터리빙 유무에 따라 성능 차이가 난다[3]. 인터리빙을 하면 읽기는 최대 5.8배(최대 39.4GB/sec), 쓰기는 최대 5.6배의 성능(최대 13.9GB/sec) 차이가 있다. 인터리빙 시, 읽기에 대해서 쓰레드 수 증가에 따라 성능도 증가하지만 쓰기는 그렇지 않다. 그림 3(b)에서 읽기는 최대 17개 쓰레드까지 스케일러블하고, 쓰기는 4개 쓰레드에서 성능이 정체된다. 또한 읽기, 쓰기 패턴에서 순차적인 패턴은 인터리빙에 영향이 없으며, 무작위 패턴은 영향을 받는 것으로 확인된다. DRAM보다 패턴의 영향을 많이 받으며, 특히 읽기와 쓰기가 혼합하는 경우 더욱 성능에 영향을 받는다. DCPMM의 대역폭 성능을 충분히 활용하기 위해서는 256바이트 또는 큰 크기의 접근을 권장한다. 특히 256바이트 이하의 무작위 접근은 권장하지 않는다.

### 다. 쓰레드 동시 지원수

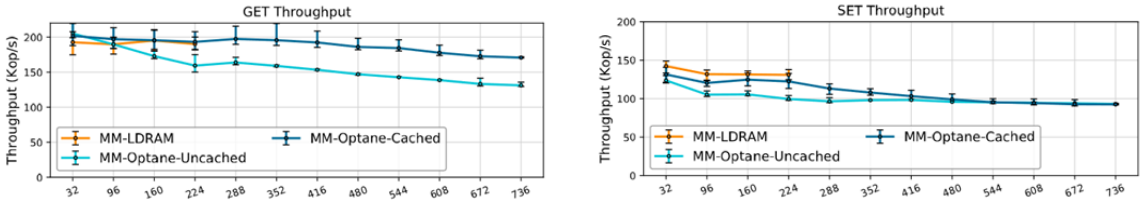
단일 DCPMM을 동시에 사용하는 쓰레드 수를 최소화해야 한다. 인텔 DCPMM의 경우, 메모리 제어기 및 제한된 버퍼는 여러 쓰레드의 접근을 동시에 처리하는 데 한계가 있다. Non-temporal store하는 쓰레드가 한 개일 때 EWR(Effective Write Ratio, 메모리 제어기가 쓰기 요청한 바이트 수 대비 실제 쓰인 바이트 수의 비율)가 0.98이지만 8개가 되면서 0.62로 떨어진다. 많은 코어가 한쪽의 DCPMM를 사용하면 메모리 제어기의 병목으로 성능이 저하되는데 읽기, 쓰기 경우 쓰레드 증가에 따라 성능 대역폭이 떨어짐을 알 수 있다.

### 라. NUMA(Non-uniform memory access) 영향

읽기의 경우, 지연 시간이 로컬 대비 원격 접근 때 순차 읽기 시 1.79배, 무작위 읽기 시 1.2배 증가한다. 쓰기의 경우, 최대 2.53배까지 차이가 나는 것으로 실험되었다. DRAM에도 NUMA 영향(NUMA effect) 문제가 있지만, DCPMM이 더 심하다. DCPMM의 NUMA 영향 문제는 메모리 제어기 부분의 병목 개선이 우선적이다. 그 이유는 NUMA 접근 시 발생하는 통신 지연은 DRAM과 다르지 않지만, 메모리 제어기 내 WPQ 등 추가적인 큐 지연이 더 문제이기 때문이다. 읽기 및 쓰기를 혼합한 워크로드에서 쓰레드 수가 증가하면, DCPMM의 대역폭 성능은 급속도로 떨어지는 현상이 있다.

## 3. 메모리 모드 성능

DRAM 크기에 맞는 메모리 풋프린트(Memory




(a) (b)

출처 Reprinted from J. Izraelevitz et al., "Basic Performance Measurements of the Intel Optane DC Persistent Memory Module," 2019, arXiv:1903.05714, CC BY 4.0.

그림 4 옵테인 DCPMM의 메모리 모드에서 성능 실험: (a) 메모리 모드에서 Redis GET 성능 실험 (b) 메모리 모드에서 Redis SET 성능 실험

footprint)를 가지는 워크로드는 DRAM를 캐쉬로 사용하는 메모리 모드가 효과적이다. SPEC 2006, SPEC 2017 등 CPU 집약적인 벤치마크 실험에서 메모리 모드에서 성능이 대부분 우수하였다. 하지만 몇 개의 워크로드에서 메모리 모드에서 성능이 떨어지는 예도 있다. 멀티 쓰레드 환경인 PARSEC 벤치마크 실험도 유사한 결과를 보인다. 그림 4와 같이 Redis 등 인메모리 데이터베이스도 메모리 모드가 효과적이고, DRAM 캐쉬 크기보다 데이터베이스 워크로드 크기가 크더라도 DCPMM 접근 횟수는 증가하지만, 메모리 모드의 성능이 효과적이라는 실험 결과들이 있다.

## 4. 앱 다이렉트 모드 성능

앱 다이렉트 모드는 시스템 전원이 꺼져도 데이터가 저장된 상태로 유지되는 영구 메모리가 된다. 이 영구 메모리에 데이터를 쓰기 위한 몇 가지 방법이 있다. store 명령어를 사용하면 데이터가 DCPMM에 저장되지 않으며, CPU 캐쉬에 데이터가 남아 있다. 완전하게 데이터를 DCPMM에 저장하기 위해서 캐쉬 플러쉬 명령어(clflush, clflushopt, clwb)를 사용해야 한다. 다른 방법은 캐쉬를 거치지 않고 바로 DCPMM에 저장하는 명령어(Non-temporal store)이다. 이 모든 명령어를 사용할 때 반드시 메모리 배리어 명령어(sfence, mfence)를 사용해야 한다. 데이터의 일관성을 유지하기 위해서 데이터 쓰기 순서를 제어해야 하기 때문이다. 다음은 이들 명령어들이 DCPMM 성능에 어떤 영향을 미치는지에 대한 분석이다.

Non-temporal store 명령어는 캐쉬 플러쉬 방법보다 쓰기 지연 시간이 크며(2~3배), 순차쓰기, 무작위 쓰기 순으로 지연 시간이 길다. 하지만 256바이트 이상 쓰기에서 non-temporal store 명령어가 캐쉬 플러쉬 방법보다 지연 시간과 대역폭 성능에서 우수하다. 반대로 256바이트 이하의 경우 캐쉬 플러쉬 방법을 사용하는 것이 좋다.

다음으로 3가지 캐쉬 플러쉬 방법의 성능을 분석한다. clflushopt와 clwb 명령어는 여러 명령어가 인터리빙 방식으로 동시에 수행되지만, clflush 명령어는 그렇지 않다. 따라서 동시에 수행되는 명령어를 사용하는 경우, sfence 등 메모리 배리어 명령어를 사용하여 쓰기 순서를 조정해야 한다. 그림 5의 여러 쓰기의 성능 실험 결과, clflush 명령

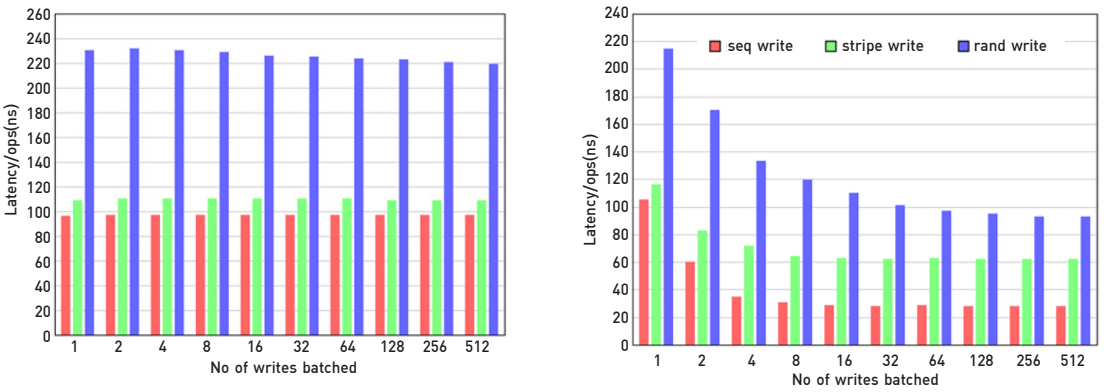



그림 5 캐쉬 플러쉬 명령어의 성능 비교: (a) 옵테인 DCPMM의 clflush 실험 결과 (b) 옵테인 DCPMM에서 clflushopt 실험 결과

어를 사용하는 경우 성능에 아무런 이득이 없고, clflushopt와 clwb는 쓰기 패턴에 따라 차이는 있지만 4~16개 동시 쓰기를 하면 순차 쓰기보다 성능이 우수하다.

파일 시스템의 성능 관점에서 분석해 보면, 기존의 디스크, SSD보다 DCPMM이 빠른 디바이스이지만 모든 파일 시스템 성능이 DCPMM 디바이스에서 우수한 것은 아니다. 나름대로 각 파일 시스템이 목표로 하는 디바이스에 맞추어 최적 기능들을 구현하고 있기 때문이다. 기존 블록 계층을 사용하지 않고 직접 접근을 하는 DAX(Direct Access) 기반 파일 시스템도 non-DAX 기반 파일 시스템보다 DCPMM 환경에서 항상 성능이 우수하지 않다. 그 이유는 페이지 캐쉬 기법 때문이다. 페이지 캐쉬를 회피하여 실험한 결과, DAX 기반 파일 시스템이 읽기, 쓰기에서 모두 우수한 성능을 발휘하며, SSD 디바이스보다도 DCPMM에서 파일 시스템의 성능이 우수했다. 여러 개의 응용 벤치마크에서 웹 서비스와 같은 읽기 위주의 워크로드는 DCPMM 환경에서 실험한 모든 파일 시스템의 성능이 비슷했지만, 쓰기 동작이 포함된 경우에 NOVA 파일 시스템이 DCPMM 환경에서 가장 성능이 우수하다.

## Ⅳ. 활용 사례

DCPMM은 DRAM보다 저비용으로 대용량의 저장 공간을 제공할 수 있어, 메모리 사용량이 많거나 빠른 파일 처리를 해야 하는 다양한 분야에서 사용이 기대된다. 사고 방지와 예방을 위해 수백만 건의 거래 정보를 기반으로 빠른 분석 결과를 필요로 하는 금융기관이나 보험회사, 동시에 수백만 고객의 방문기록을 실시간으로 저장하고 방문고객의 성향을 빠르게 분석하여 제품을 추천하고자 하는 온라인 쇼핑몰, 사이버 위협에 실시간으로 대비해야 하는 기관 등에서 운영하는 시스템에 바이트 단위로 접근 가능한 고성능 영구 메모리 사용이 증가할 것이다. 인메모리 데이터베이스, 빅데이터의 실시간 분석, 고성능 컴퓨팅시스템, 가상화 시스템, 인공지능 시스템 등도 대용량의 메모리 또는 고성능 저장장치를 사용하는 대표적인 분야이다.

인메모리 데이터베이스는 디스크나 SSD 등에 데이터를 저장하는 전통적인 데이터베이스와 다르게 데이터를 DRAM에 저장한다. 데이터를 DRAM에 저장하면 디스크나 SSD 접근에 필요한 접근시간을 제거할 수 있어서 보다 빠른 응답을 기대할 수 있다. 그러나 휘발성 메모리에 저장되는 데이터는 장애 발생에 취약하다. 고성능의 영구 저장장치인 DCPMM을 인메모리 데이터베이스에 사용하면 데이터를 안전하게 저장할 수 있다. 대표적인 인메모리 데이터베이스는 Redis, Memcached, Aerospike, SAP사의 HANA DB 등이 있다. Redis Enterprise platform의 경우 192GB DRAM과 1.5TB DCPMM을 함께 사용하면, 1.5TB DRAM만 사용 때와 동등한 성능이거나 다소 느리지만 비용은 43% 절감할 수 있다[4]. 앱 다이렉트 모드에서 SAP HANA 데이터베이스로 실험[5]한 경우, 순차 질의 성능은 DRAM만 사용할 때보다 약간 높은 성능을 얻을 수 있고 병렬 질의 성능은 다소 낮지만, 재부팅에 걸리는 시간은 4배 이상 단축할 수 있음을 보여준다.

고성능 컴퓨팅 응용에 DCPMM의 사용이 주는 영향 분석[6]에 따르면 DCPMM의 높은 지연 시간과 낮은 대역폭으로 메모리를 많이 사용하는 고성능 컴퓨팅 응용의 성능은 DRAM만으로 필요한 메모리를 사용할 때보다 저하되지만, DRAM을 캐쉬로 사용함으로써 성능 저하를 보완할 수 있고 에너지 사용을 줄일 수 있다. 많은 메모리를 사용하는

대부분의 과학 문제 응용들에 대한 벤치마크를 실행한 연구[7]는 DCPMM의 사용이 SSD를 사용하는 경우보다 높은 성능을 얻을 수 있지만, 응용의 특성에 따라서는 SSD와 IMDT(Intel Memory Drive Technology)를 함께 사용할 때 더 높은 성능을 얻을 수 있는 예를 보여주고 있다.

VMware는 메모리 모드에서 여러 가상머신을 사용할 때 DRAM 캐쉬의 충돌로 성능 문제가 발생할 수 있음을 참고문헌 [8]에서 보여주고 있다. 가상머신 환경에서 DCPMM을 가상머신의 저장장치로 사용할 때 NVMe 같은 저장장치를 사용할 때보다 2배 이상 높은 성능을 얻기도 한다.

참고문헌 [9]에서는 응용의 수정을 최소화하고 DCPMM의 사용으로 성능 이익을 높일 수 있는 FLEX(FiLe Emulation with DAX)의 사용을 제안하고 있다.

기계학습은 주어진 데이터를 기반으로 더 나은 결정을 하기 위해 반복적으로 데이터를 사용한다. 학습한 결과를 저장하고, 추가 데이터를 이용하여 학습을 반복하는 일련의 과정을 빠른 시간에 수행하기 위해서는 많은 메모리 사용이 필요하다. 아파치 스파크(Apache Spark)는 데이터 처리 엔진으로 대규모 데이터 분석에 많이 사용되고 있으며 기계학습, 클라우드 기반 서비스, 사물 인터넷 등 다양한 분야에서 활용되고 있다. 아파치 스파크의 성능 저하 요인 중 하나는 데이터 접근 과정에서 발생하며 성능 문제를 완화하기 위하여 캐싱 기법을 사용하고 있다. 그러나 데이터의 크기가 증가할수록 효과적으로 대응하기 어려운 문제가 있다. 이런 문제를 해결하기 위하여 DCPMM과 OAP(Optimized Analytics Package)를 사용하면 높은 성능을 얻을 수 있다. 데이터셋이 아주 큰 경우 DRAM만 사용할 때보다 1.66배 높은 성능을 얻을 수 있다. K-means 알고리즘처럼, 캐싱이 많이 필요한 경우는 2.85배 높은 성능을 보이기도 한다[10,11].

## V. 연구개발 동향

DCPMM이 출시되면서 기기의 성능을 분석하려는 연구들이 다수 수행되었고, 이전에 오랫동안 에뮬레이터 환경에서 실험한 결과와 다른 결과들이 나왔다.

DCPMM의 바이트 단위 접근과 비휘발성을 활용하는 노력은 인메모리 데이터베이스, Key-value store와 같은 응용에서 주로 이루어지고 있다. DRAM 기반의 인메모리 데이터베이스에서 사용하는 기존의 인덱스 자료구조 알고리즘은 DCPMM 환경에서 크래시(Crash)로부터 복구를 보장할 수 없으므로 crash consistency를 보장하는 인덱스 알고리즘에 관한 연구가 진행되고 있다. 한편으로는 DRAM과 DCPMM을 동시에 활용하기 위한 데이터베이스나 Key-value store 구조를 제안하는 연구가 진행되고 있다.

DCPMM을 매우 빠른 저장소로 활용하면서, 바이트 단위 접근이 가능하다는 장점을 그대로 활용하기 위해 DCPMM에 최적화된 파일 시스템 연구도 진행하고 있다. DCPMM에 최적화된 성능 기능뿐만 아니라 많은 코어 환경에서 확장성 연구도 진행하고 있다.

PCM 기반의 DCPMM의 저장 매체는 NAND flash보다 압도적으로 높은 내구성을 가지는 것으로 알려졌다. 하지만 DCPMM을 활용하는 응용은 더 자주 메모리 셀에 접근할 수 있으며, 빠른 속도로 내구성을 소모할 수 있다. 따라서 장치의 수명을 극대화하는 기술에 관한 연구가 필요하다. 일례로, 메모리 제어기의 WPQ의 쓰기 적중률(Write hit rate)을 높일 수 있다면 저장 매체의 쓰기 동작을 줄여 수명을 늘릴 수 있다.

DCPMM은 기존에는 존재하지 않던 새로운 보안 이슈를 발생시킬 수 있다. DCPMM에 저장된

데이터는 영구적으로 유지되기 때문에 데이터가 완전히 삭제되지 않았을 경우 해당 영역을 할당받아 접근할 수 있는 문제가 있다. 따라서 접근 권한을 엄격하게 관리하고 데이터를 암호화해야 하는데, 이러한 해결 방식은 성능을 떨어뜨릴 수 있다. 따라서 DCPMM의 보안 문제를 발견하고, 효율적인 해결책을 찾으려는 연구도 진행되고 있다.

한편 HPC, AI, 보안 등과 같이 빅데이터를 활용하여 분석을 수행하는 응용의 경우 DCPMM을 저렴하고 큰 용량의 메모리로 활용하고자 한다. 이러한 응용의 경우 워킹 데이터셋을 모두 메모리에 적재할 수 있다면, 분석 알고리즘에서 I/O를 위한 루틴을 제거하여 성능을 높일 수 있다. 또한, 입출력 동작을 제거하면서 알고리즘 구현을 단순화할 수 있으므로 개발 기간도 단축할 수 있다. 하지만 이를 위해서 운영체제가 대용량 메모리를 효율적으로 관리하고 있는지 검증되어야 한다.

## VI. 결론

바이트 단위로 데이터를 영구적으로 저장할 수 있으면서 메모리 버스에 직접 설치하여 사용할 수 있는 장치가 최근 인텔에서 옵테인이라는 상품으로 출시되었다. 새로운 제품의 구조와 성능 특성을 분석하기 위한 연구와 실험이 다양하게 수행되었으며, 그 결과는 지난 10여 년 동안 진행했던 에뮬레이션 기반 연구 결과와 다른 것이 많았다.

인텔의 옵테인 DCPMM은 DRAM처럼 사용할 수 있고, 전통적인 저장장치처럼 사용할 수도 있다. 옵테인 DCPMM은 CPU의 캐쉬 라인 크기인 64바이트의 4배 크기인 256바이트 단위로 버퍼링과 데이터를 결합하는 작업을 한다. 이러한 특성을 적절하게 사용하기는 쉽지 않으며 잘못 사용할 경우 오히려 성능이 떨어지기도 한다. 새롭게 등장한 옵테인은 전통적으로 사용하던 DRAM과 저장장치의 특성을 공유하고 있으며, 이러한 장치의 성능 특성을 최대한 활용하려면 운영체제와 응용에서 새로운 접근이 필요하다. 연산장치에서 직접 접근할 수 있는 대용량 메모리 사용환경이 실현됨에 따라 분산된 자원을 하나의 시스템에서 운영할 수도 있게 되었다. 이러한 컴퓨팅 환경의 변화에 대응하기 위하여 앞으로 많은 연구가 필요하다.

**용어해설**

**인터리빙(Interleaving)** 하드웨어 수준의 병렬화를 통해 장치의 대역폭을 증가시키는 기법

**캐쉬 플러쉬(Cache flush)** 캐쉬에 저장되어 있는 데이터를 다음 단계의 메모리(또는 저장장치)로 내보내는 절차

**메모리 배리어(Memory barrier)** CPU 또는 컴파일러에게 메모리 접근 처리 순서를 강제하기 위한 기법

**웨어 레벨링(Wear leveling)** 메모리 저장장치의 서비스 수명을 연장하는 기법

**메모리 풋프린트(Memory footprint)** 프로그램이 실행하는 동안 사용하거나 참조하는 기본 메모리의 양

**약어 정리**

| | |
|---|---|
| ADR | Asynchronous DRAM Refresh |
| CPU | Central Processing Unit |
| DAX | Direct Access |
| DCPMM | DataCenter Persistent Memory Module |
| DRAM | Dynamic Random Access Memory |
| EWR | Effective Write Ratio |
| HDD | Hard Disk Drive |
| HPC | High Performance Computing |
| LLC | Last Level Cache |
| ReRAM | Resistive RAM |
| SSD | Solid State Drive |
| WPQ | Write Pending Queue |

## 참고문헌


[1] Wikipedia, “NVDIMM.” https://en.wikipedia.org/wiki/NVDIMM

[2] J. Yang, J. Kim, M. Hoseinzadeh, J. Izraelevitz, and S. Swanson, “An Empirical Guide to the Behavior and Use of Scalable Persistent Memory, arXiv:1908.03583, 2019.

[3] J. Izraelevitz et al., “Basic Performance Measurements of the Intel Optane DC Persistent Memory Module,” arXiv:1903.05714, 2019.

[4] Intel, “Break the Cost and Capacity Barrier with intel® OptaneTM DC Persistent Memory,” https://www.intel.com/content/dam/www/public/us/en/documents/solution-briefs/redis-enterprise-brief.pdf

[5] Fujitsu, “White paper: Performance Test Report of Intel Optane DC Persistent Memory on PRIMEFLEX for SAP HANA,” Apr. 2019. https://sp.ts.fujitsu.com/dmsp/Publications/public/wp-performance-report-primergy-inteloptane-saphane.pdf

[6] O. Patil et al., “Performance Characterization of a DRAM-NVM Hybrid Memory Architecture for HPC Applications Using Intel Optane DC Persistent Memory Modules,” in Proc. Int. Symp. Memory Syst., Washington, D.C., USA, Sept. 2019. pp. 288-303.

[7] V. Mironov et al., “Performance Evaluation of the Intel Optane DC Memory With Scientific Benchmarks.” In Proc. IEEE/ACM Workshop Memory Centric High Performance Computer., Denver, CO, USA, Nov. 2019. pp. 1-6.

[8] vmware, “Intel Optane DC Persistent Memory “Memory Mode” Virtualized Performance Study,” July 28, 2019, https://www.vmware.com/content/dam/digitalmarketing/vmware/en/pdf/techpaper/performance/IntelOptaneDC-PMEM-memory-mode-perf.pdf

[9] J. Xu et al., “Finding and Fixing Performance Pathologies in Persistent Memory Software Stacks,” in Proc. Int. Conf. Architectural Support for Programming Languages Operat. Syst., Providence, RI, USA, Apr. 2019. pp. 427-439.

[10] Intel Developer zone, “Speeding Up Apache Spark Workloads on Intel® Optane™ DC Persistent Memory,” https://software.intel.com/en-us/articles/speeding-up-apache-spark-workloads-on-intel-optane-dc-persistent-memory

[11] C, Xu and P. Balver, “Accelerate Your Apache Spark with Intel Optane DC Persistent Memory,” https://databricks.com/session/accelerate-your-apache-spark-with-intel-optane-dc-persistent-memory