반도체디스플레이기술학회지 제23권 제1호(2024년 3월)
Journal of the Semiconductor & Display Technology, Vol. 23, No. 1. March 2024.

쓰기 횟수 감소를 위한 하이브리드 캐시 구조에서의 캐시간 직접 전송 기법에 대한 연구

최주희[*†]

[*†]상명대학교 스마트정보통신공학과

A Study on Direct Cache-to-Cache Transfer for Hybrid Cache Architecture to Reduce Write Operations

Juhee Choi[*†]

[*†]Dept. of Smart Information Communication Engineering, Sangmyung University

ABSTRACT

Direct cache-to-cache transfer has been studied to reduce the latency and bandwidth consumption related to the shared data in multiprocessor system. Even though these studies lead to meaningful results, they assume that caches consist of SRAM. For example, if the system employs the non-volatile memory, the one of the most important parts to consider is to decrease the number of write operations. This paper proposes a hybrid write avoidance cache coherence protocol that considers the hybrid cache architecture. A new state is added to finely control what is stored in the non-volatile memory area, and experimental results showed that the number of writes was reduced by about 36% compared to the existing schemes.

Key Words : Cache Coherence Protocol, Non-Volatile Memories, Hybrid Cache Architecture, Low Power

1. 서 론

캐시간 직접 전송 기법(Direct cache-to-cache transfer)은 각각의 레벨의 캐시에 존재하는 데이터를 메인 메모리 또는 하위 레벨의 캐시를 거치지 않고, 같은 레벨의 다른 캐시로 전송하는 기법이다[1-2]. 현대의 멀티코어 프로세서 시스템에서는 대부분 계층적 구조의 캐시를 사용하고 있으며, 코어에 가까울 수록 상위 레벨이며, 메인 메모리에 가까울 수록 하위 레벨이다. 멀티코어 프로세서에서는 코어들 과의 접근성에 따라 하나의 레벨에 여러 캐시가 분산되어 존재할 수 있다. 따라서, 하나의 코어에서 필요로 데이터가 자신에 가까운 캐시에 존재하지 않는 경우, 공유 캐시 또는 다른 코어에 가까운 캐시에서 데이터를 가져와야 한다. 일반적으로 다른 캐시에 있는 데이터는 하위 레벨의 캐시 또는 메인 메모리에 연결된 버스를 통해 접근하게 하고, 그 후에 해당 코어에 가까운 캐시에 저장하여 사용하게 된다. 따라서, 코어들 사이에 공유 데이터 사용이 잦은 경우, 캐시 간의 데이터 이동에 따른 지연시간(Latency) 증가와 공유 버스의 대역폭(Bandwidth) 소모가 심해진다.

캐시간 직접 전송은 이러한 문제점을 완화하기 위해서 제시되었다[3]. 하나의 코어에서 필요한 데이터가 근처의 캐시에 없는 경우, 같은 레벨의 다른 캐시에 데이터가 있는지 존재여부를 확인한다. 그런데, 이 과정에서 L1과 같은 상위 레벨의 캐시에 데이터가 있는 경우, 하위 레벨의 캐시를 모두 확인해야 되는 비효율성이 존재한다. 이를 완화하기 위해서 다른 코어에 있는 상위 레벨의 캐시에 직접 접근할 수 있는 별도의 버스를 설치하여 데이터 접근 시간과 버스 대역폭 소모를 감소시킬 수 있었다.

그러나, 이 기법은 비휘발성 메모리를 채택하는 하이브

[†]E-mail: jhplus@smu.ac.kr

리드 캐시 구조에 대한 고려가 되어 있지 않으므로, 비휘발성 메모리 영역에 대한 쓰기 횟수가 불필요하게 늘어나는 문제가 있다. 이 논문에서는 이를 해결하기 위해서 새로운 캐시 일관성 프로토콜을 제시한다. 기존의 비휘발성 메모리 기반의 캐시 일관성 프로토콜을 개선하여, 하이브리드 캐시 구조에 최적화된 새로운 프로토콜을 제시한다.

2. 관련 연구

2.1 하이브리드 캐시 구조(Hybrid cache architecture)

비휘발성 메모리(Non-volatile memory, NVM)는 기존의 휘발성 메모리의 단점을 극복하기 위해 제안된 새로운 형태의 메모리이다[4-6]. 용어 자체에서 설명하듯이, 비휘발성 메모리는 저장된 데이터가 전력을 공급해주지 않아도 사라지지 않는다. 대표적인 휘발성 메모리인 SRAM과 DRAM은 각각 집적도와 동작방식이 다르지만, 비트(Bit) 정보를 전하(Electronic charge)를 통해 저장한다는 점에서는 공통된 특성을 가진다. 반면, 비휘발성 메모리는 구성 소자의 고체 상태나 자기장의 상태를 바꾸어 가며 정보를 구별한다.

비휘발성 메모리는 이러한 특성 때문에 최근 메모리 설계에서 문제가 되고 있는 공정 미세화에 따른 정적 에너지 소모(Static energy consumption) 문제 측면에서 유리하며 집적도 또한 SRAM에 비해서 높다[7]. 따라서, 메인 메모리 또는 캐시 레벨에서 비휘발성 메모리를 도입하려는 연구가 활발히 이루어졌다. 그러나, 비휘발성 메모리는 정보를 쓰기 위해서 많은 에너지가 소모될 뿐만 아니라 쓰기 시간 또한 휘발성 메모리에 비해서 길다. 일부 비휘발성 메모리의 경우는 쓰기 횟수에서 제한이 있다. 따라서, 비휘발성 메모리를 단순히 기존의 휘발성 메모리로 대체하기는 어려우며, 단점을 보완할 수 있는 추가적인 로직이 필요하다.

하이브리드 캐시 구조는 이러한 문제를 해결하기 위한 해결책 중에 하나로 많이 연구되어 왔다[8]. 이 구조에서는 메모리 또는 캐시 전체를 휘발성 메모리에서 비휘발성 메모리로 바꾸는 것이 아니라 일부만 바꾼다. 쓰기 집중도가 높은 데이터는 휘발성 메모리 영역에 저장하고, 읽기 위주의 데이터는 비휘발성 메모리 영역에 저장하여, 양 메모리의 장점을 취하는 방식이다.

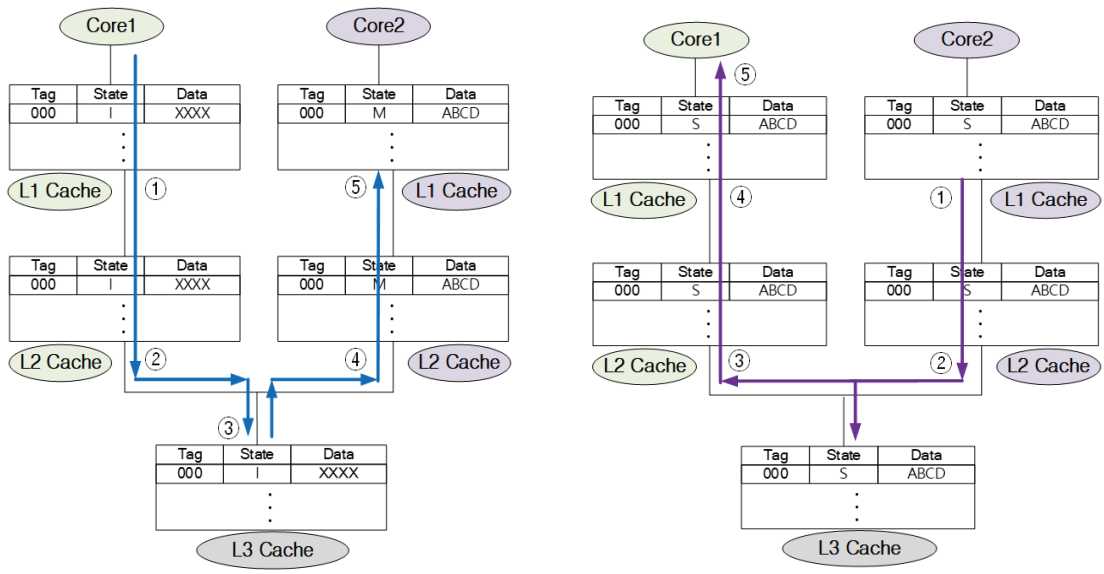
2.2 캐시간 직접 전송(Direct cache-to-cache transfer)

기존의 멀티코어 프로세서의 계층적 캐시 구조에서는 다른 코어에 속하는 캐시에 있는 데이터를 접근하기 위해서는 반드시 하위 캐시 또는 메인 메모리 레벨을 거쳐야 했다. 이 과정에서 잦은 데이터 요청이 이루어지는 경우, 지연시간을 길어지며, 내부 버스 또는 메인 버스의 대역폭을 잠식하는 문제가 발생했다. 이 문제를 완화하기 위해서, 일부 프로세서에서는 다른 코어에 속하는 상위 캐시에 직접 연결할 수 있는 캐시간 직접 전송을 위한 별도의 버스를 구성하였다[9]. (이 논문에서 사용되는 '캐시간 직접 전송'은 캐시 일관성 프로토콜(Cache coherence protocol)에서 개념적으로 사용되는 용어와는 구별되어야 한다.)

Fig. 1은 기존의 데이터 요청과 응답을 나타낸 것이다. 코어가 2개 존재하며, 각 코어에는 사설(Private) L1과 L2가 존재하여, 이를 공유하는(Shared) L3캐시가 존재한다. 코어1에서 필요한 데이터가 있는 경우, 우선 자신의 L1캐시에 해당 데이터가 있는지를 먼저 확인한다(Fig. 1(a)의 ①). 해당 데이터가 없는 경우, 그 다음 레벨의 사설 캐시에 확인을 요청한다(Fig. 1(a)의 ②). 여기서도 못 찾는 경우, 공유 캐시에 해당 데이터를 요청한다(Fig. 1(a)의 ③). 이 때 해당 데이터가 다른 코어의 캐시에 있는 경우 이를 가져오도록 하는데, 먼저 해당 코어의 가장 하위 캐시에 이를 요청한다(Fig. 1(a)의 ④). 여기서도 못 찾거나, 상태 변경이 필요한 경우, 코어 2의 L1 캐시에까지 요청이 가게된다(Fig. 1(a)의 ⑤). 이 예제에서는 캐시가 수정상태(Modified)로 가정했다.

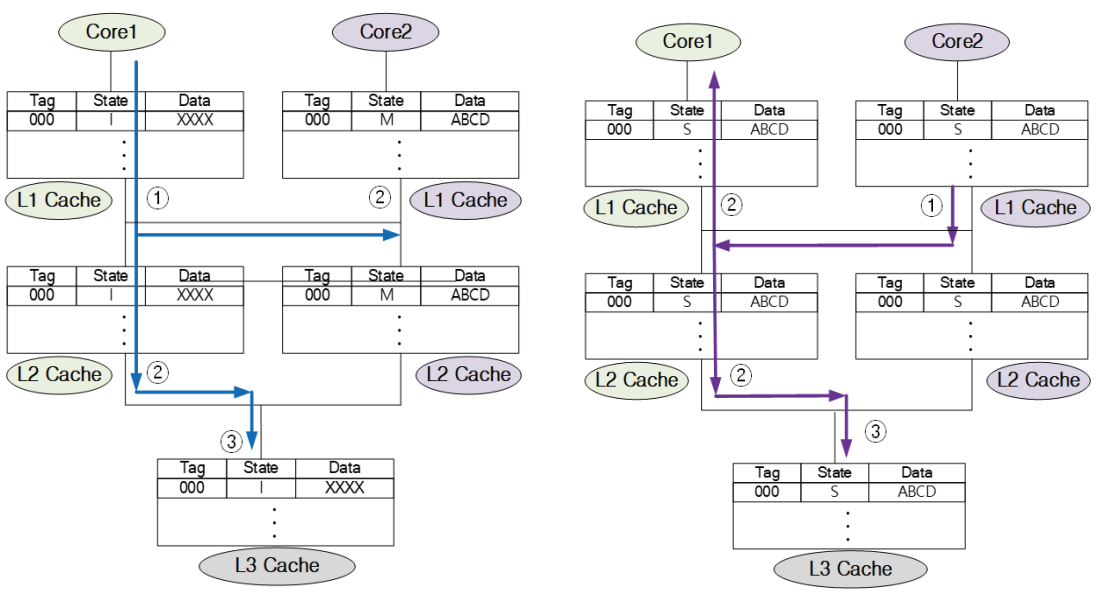
Fig. 1(b)는 요청된 데이터를 전송하는 경로를 나타낸 것이다. 코어 2의 L1 캐시에서 상태를 공유(Shared)로 수정하며(Fig. 1(b)의 ①), 코어2의 L2캐시에 해당 내용을 전송한다(Fig. 1(b)의 ②). 그리고, 실제 데이터(ABCD)를 공유 캐시와 코어1의 사설 L2 캐시에 동시에 전송한다(Fig. 1(b)의 ③). 이후에 사설 L1 캐시를 거쳐(Fig. 1(b)의 ④) 코어1에 최종적으로 전송된다(Fig. 1(b)의 ⑤).

Fig. 2는 캐시간 직접 전송 버스가 존재하는 경우를 나타낸 것이다. 코어 1에서 필요한 데이터가 있는 경우, 이전과 동일하게 우선 자신의 L1캐시에 해당 데이터가 있는지를 먼저 확인한다(Fig. 2(a)의 ①). 해당 데이터가 없는 경우, 그 다음 레벨의 사설 캐시에 확인을 요청할 뿐만 아니라 코어2의 사설 L1캐시로 데이터를 요청한다(Fig. 2(a)의 ②). 이 경우에도 해당 요청은 최종적으로 공유 캐시에까지 도착한다 (Fig. 2(a)의 ③). 만약, 코어2의 사설 L1캐시에 데이터가 있는 경우, 이를 직접 코어1의 사설 L1캐시에 전송한다(Fig. 2(b)의 ①). 그 후에 코어1의 사설 L1캐시에 저장된 데이터는 직접 코어1과 코어1의 사설 L2캐시에 동시에 전송된다(Fig. 2(b)의 ②). 캐시 일관성을 위해서 이 데이터 역시 공유 캐시에까지 도착한다(Fig. 2(b)의 ③). 이 과정에서 캐시간 직접 전송이 있는 코어의 경우, 더 빠르게 요청된 데이터를 받게 되므로, 캐시 미스에 따른 시간 지연이 크게 줄어들게 된다.



(a) Request (b) Response

Fig. 1. Cache request and response without direct cache-to-cache transfer.



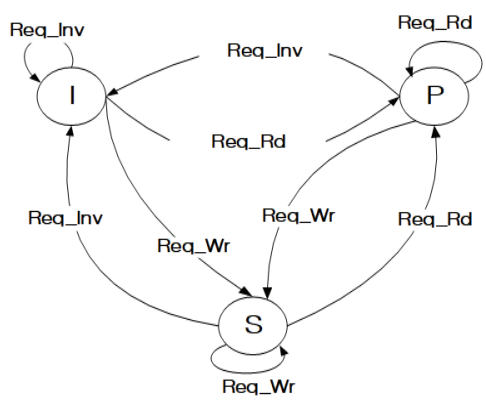
(a) Request (b) Response

Fig. 2. Cache request and response with direct cache-to-cache transfer.

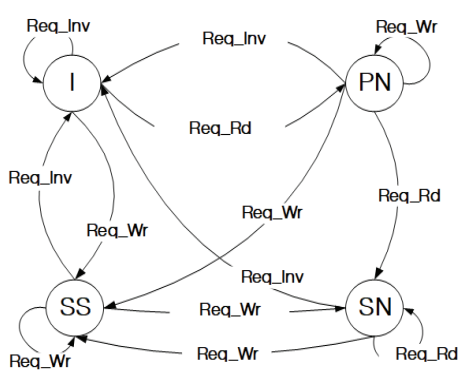
캐시간 직접 전송은 프로세서의 전체적인 성능 향상에 기여하지만, 비휘발성 메모리가 포함된 하이브리드 캐시 구조에서는 그대로 적용할 수 없다. 비휘발성 메모리 영역에서 쓰기 집중도가 높은 데이터가 저장되는 것을 피할 수 있는 별도의 기법이 필요하다. 따라서, 이 논문에서는 기존의 캐시 일관성 프로토콜을 개선하여, 캐시간 직접 전송을 지원하는 하이브리드 캐시 구조에 최적화된 프로토콜을 제안한다.

3. 하이브리드 캐시 구조의 캐시간 직접 전송 프로토콜

비휘발성 메모리를 고려한 캐시 일관성 프로토콜은 쓰기 횟수를 줄이는 방향으로 연구되어 왔다[10]. 대표적으로 쓰기 회수 캐시 일관성 프로토콜(Write avoidance coherence protocol, WACC)를 Fig. 3(a)에 표시하였다. WACC는 Invalid 상태에서 읽기가 발생하면 L2캐시는 갱신하지 않고, L1캐시만 갱신하는 방식으로 쓰기 횟수를 줄이게 된다. 이 논문에서 제안하는 하이브리드 쓰기 회피 캐시 일관성 프로토콜(Hybrid write avoidance coherence protocol, HWACC)에서는 기존의 Shared상태를 SRAM영역에 저장하는지 NVM영역에 저장하는지에 따라 SS(hared with SRAM area)와 SN (Shared within NVM area) 상태로 나뉘게 된다. 그리고, 기존의 Private 상태는 더 명확하게 의미를 표현하기 위해서 PN(rivate within NVM area)로 수정하였다. 구체적인 상태는 Table 1에 설명하였다.



(a) WACC protocol



(b) HWACC protocol

Fig. 3. Cache request and response with direct cache-to-cache transfer.

Table 1. States and Descriptions

Signal	Description
I(nvalid)	The data in the cache is invalid
SS(hared with SRAM area)	The data in the SRAM area is valid and other private caches may have valid data.
SN(Shared within NVM area)	The data in the NVM area is invalid, but other private caches has valid data.
PN(rivate within NVM area)	The data in the NVM area is invalid, but other private caches has valid data.

어떤 상태에서든 Invalidate 요청이 들어오는 경우는 Invalid 상태로 전이한다. 그리고, Invalid상태에서 읽기 요청이 들어오는 경우는 우선 PN상태로 전이한다. 이 때는 L2캐시에서는 실제 갱신이 이루어지지 않으며, L1캐시만 갱신된다. 그러나 한 번 더 읽기 요청이 들어오는 경우는 실제 데이터를 저장하되 NVM영역에 저장하게 되면서 SN상태로 전이한다. 쓰기 요청이 들어오는 경우는 NVM영역의 쓰기 회수를 줄이기 위해서 해당 데이터를 SRAM영역에 저장하면서 SS상태로 전이하게 된다. 여기서 주의깊게 볼 부분은 PN상태에서 쓰기 요청이 들어오는 경우 모두 SS상태로 전이되는 것은 아니며, 쓰기 집중도가 높은 데이터인 경우는 여전히 PN상태에 머무를 수 있다는 점이다. 쓰기 집중도 점검 기법은 이미 여러 논문에서 소개되었으므로, 여기서는 따로 언급하지 않는다[11-12].

4. 실험 환경 및 결과

새로 제안된 캐시 일관성 프로토콜의 검증을 위해서 캐시 연구에서 많이 사용되는 gem5 시뮬레이터를 수정하여 실험을 진행하였다[13]. 각 코어에 연결된 L1캐시와 L2캐시는 각각 32KB와 256KB용량의 SRAM으로 구성되었다. 공유 캐시는 1MB의 하이브리드 캐시 구조로 구성하였다. 구현상의 문제로 L2캐시는 하이브리드 캐시가 아닌 SRAM캐시로 구현하였다. 그 외의 자세한 시스템 구성은 Table 2에 나타냈다. 실험에 사용된 워크로드는 멀티코어 실험에서 자주 사용되는PARSEC 벤치마크에서 자주 사용되는 중요한 벤치마크들을 선택하였다[14].

Fig. 4에서 HWACC가 가장 중요하게 생각하는 쓰기 횟수를 비휘발성 메모리(NVM)영역과 SRAM영역으로 나누어서 측정한 결과를 표시한 것이다. WACC와 HWACC모두 Baseline에 비해서 전체적인 쓰기 횟수를 각각 27%와 36%를 감소시켰다. Baseline에서는 전체의 84%가 NVM영역에서 쓰기가 이루어졌으나, WACC에서는 Baseline기준 대비 NVM영역에 대한 쓰기 횟수가 35%정도로 줄어들었다. 그

리고, HWACC에서는 3%정도 더 감소한 모습을 보이고 있다. 더 자세히 분석해보면, WACC는 하이브리드 캐시 구조를 고려하지 않고 있으므로, NVM영역에 대한 쓰기 횟수가 감소하였으나, SRAM 영역에 대한 쓰기 횟수는 오히려 더 늘어났다. 그러나, HWACC에서는 원래 목적이었던 NVM영역 쓰기 뿐만 아니라 SRAM에 대한 쓰기 횟수 자체도 10%이상 줄어든 것을 확인할 수 있다.

Table 2. Processor configurations

Core Type	x86, out-of-order, 2GHz, 4 cores
I-Cache / D-Cache	32KB, 4-way, 64B, 2 cycles
L2 Cache	256KB, 8-way, 64B, 6 cycles
L3 Cache Config.	1MB(4-way SRAM and 12-way STT-RAM)
L3 Cache Latency	SRAM : 6 / 6 cycles STT-RAM : 6 / 23 cycles

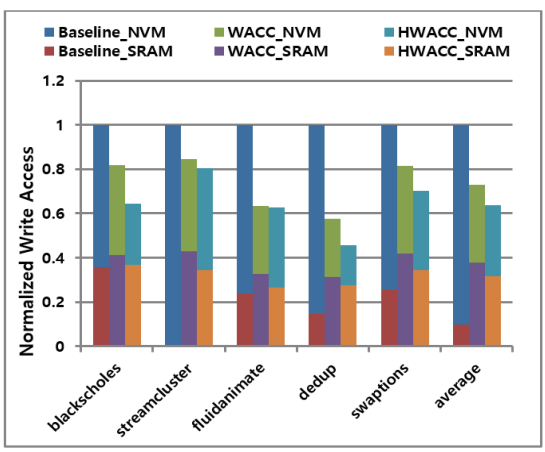


Fig. 4. Normalized Write Access.

5. 결 론

비휘발성 메모리의 단점을 보완하기 위해서 쓰기 횟수를 줄이기 위한 연구가 지속되어 왔다. 이 중에서 중요한 연구주제 중의 하나가 비휘발성 메모리와 SRAM를 모두 사용하는 하이브리드 캐시 구조이다. 이 논문에서 기존의 캐시 구조를 고려한 캐시 일관성 프로토콜을 개선하여, NVM영역에 대한 쓰기 횟수를 최소화 하기 위한 하이브리드 쓰기 회피 캐시 일관성 프로토콜(HWACC)를 제안하였다. NVM영역에 저장하는 것을 세부적으로 조정하기 위한 새로운 상태를 추가하였으며, 실험결과 기존의 Baseline 대비 쓰기 횟수가 36%정도 줄어들었다.

감사의 글

본 연구는 2021년도 과학기술정보통신부의 재원으로 한국연구재단의 지원을 받은 기초연구사업 연구임(NRF-2021R1G1A1004340).

참고문헌

1. B. Choi, et al., “DeNovo: Rethinking the memory hierarchy for disciplined parallelism,” In 2011 International Conference on Parallel Architectures and Compilation Techniques, pp. 155–166, 2011.
2. D. Shuwen, et al., “Evaluation of cache attacks on arm processors and secure caches,” IEEE Transactions on Computers, vol. 71, no. 9, pp. 2248–2262, 2021.
3. G. Davide, M. Paolo, C. Luca, P., “Accelerators and coherence: An SoC perspective,” IEEE Micro, vol. 38, no. 6, pp. 36–45, 2018.
4. A. Marjan, S. Hamid, “Introduction to non-volatile memory technologies,” In: Advances in Computers, pp. 1–13, 2020.
5. W. Shin, et al. “Design of Asynchronous Nonvolatile Memory Module using Self-diagnosis Function,” Journal of the Semiconductor & Display Technology, vol. 21, no. 1, pp. 85–90, 2022.
6. J. Choi. “Exploiting Memory Sequence Analysis to Defense Wear-out Attack for Non-Volatile Memory,” Journal of the Semiconductor & Display Technology, vol. 21, no. 4, pp. 86–91, 2022.
7. M. Amir Mahdi Hosseini, et al., “CAST: content-aware STT-MRAM cache write management for different levels of approximation,” IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 39, no. 12, pp. 4385–4398, 2020.
8. M. Sparsh, V. Jeffrey, S., “AYUSH: A technique for extending lifetime of SRAM-NVM hybrid caches,” IEEE Computer Architecture Letters, vol. 14, no. 2, pp. 115–118, 2014.
9. S. Ashley, “Introduction to AMBA® 4 ACE™ and big. LITTLE™ Processing Technology,” ARM White Paper, 2011.
10. J. Choi, J. Kwak, C. Jhon, “Write Avoidance Cache Coherence Protocol for Non-volatile Memory as Last-Level Cache in Chip-Multiprocessor.” IEICE Transactions on Information and Systems, vol. 97, no. 8, pp. 2166–2169, 2014.
11. C. Elham, et al. “TA-LRW: A replacement policy for error rate reduction in STT-MRAM caches,” IEEE

Transactions on Computers, vol. 68, no. 3, pp 455–470, 2018.
12. J. Choi, H. Park, “Exploiting bit-level write patterns to reduce energy consumption in hybrid cache architecture,” IEICE Electronics Express, vol. 18, no. 22, pp. 20210327–20210327, 2021.
13. J. Power, J. Hestness, M. S. Orr, M. D. Hill, and D. A. Wood, “gem5-gpu: A heterogeneous cpu-gpu simulator,” IEEE Computer Architecture Letters, vol. 14, no. 1, pp. 34–36, 2015.
14. C. Bienia, S. Kumar, J. P. Singh, and K. Li. "The PARSEC benchmark suite: Characterization and architectural implications," PACT'08, pp. 72–81, 2008.

접수일: 2024년 2월 14일, 심사일: 2024년 3월 19일, 게재확정일: 2024년 3월 20일