Journal of The Korea Society of Computer and Information
Vol. 23 No. 4, pp. 1-8, April 2018

www.ksci.re.kr
https://doi.org/10.9708/jksci.2018.23.04.001

Performance Analysis of Adaptive Partition Cache Replacement using Various Monitoring Ratios for Non-volatile Memory Systems

Sang-Ho Hwang*, Jong Wook Kwak**

Abstract

In this paper, we propose an adaptive partition cache replacement policy and evaluate the performance of our scheme using various monitoring ratios to help lifetime extension of non-volatile main memory systems without performance degradation. The proposal combines conventional LRU (Least Recently Used) replacement policy and Early Eviction Zone (E2Z), which considers a dirty bit as well as LRU bits to select a candidate block. In particular, this paper shows the performance of non-volatile memory using various monitoring ratios and determines optimized monitoring ratio and partition size of E2Z for reducing the number of writebacks using cache hit counter logic and hit predictor. In the experiment evaluation, we showed that 1:128 combination provided the best results of writebacks and runtime, in terms of performance and complexity trade-off relation, and our proposal yielded up to 42% reduction of writebacks, compared with others.

▸Keyword: Non-volatile memory, cache replacement policy, performance analysis, writeback, LRU

I. Introduction

현재 컴퓨터 시스템의 메인 메모리로 사용되고 있는 DRAM은 데이터 유지를 위한 지속적인 리프레시(refresh)가 필요하며, 이는 전체 시스템 전력의 40%를 차지할 정도로 전력소모가 많다[1]. 이를 해결하기 위해 최근에는 PCM(Phase Change Memory)과 같은 비 휘발성 메모리를 메인 메모리로 활용하는 시스템에 대한 연구가 활발히 이루어지고 있다.

비휘발성 메모리인 PCM은 지속적인 전원 공급이 없어도 데이터 유지가 가능하여 DRAM에 비해 전력 소모를 크게 줄일 수 있다. 하지만, PCM을 메인 메모리로 사용하기 위해서는 크게 두가지 단점을 해결해야 한다. 첫 번째로, PCM은 쓰기 연산과 읽기 연산이 속도면에서 비대칭이다. PCM의 쓰기 연산과 읽기 연산의 속도는 각각 150ns, 22ns이다. 이러한 PCM의 쓰기 연산 및 읽기 연산의 비대칭 문제는 가교 역할을 수행하는 DRAM 버퍼를 이용하여 해결이 가능하다[2]. 두 번째로, PCM은 쓰기 횟수가 기존 DRAM의 10^{15} 보다 적은 10^{8} ~ 10^{9}번으로 제한되어 있다. 이러한 PCM의 수명 문제를 해결하기 위해 PCM 메모리에서 이루어지는 쓰기 연산을 분산시키는 Start-Gap[3], Security Refresh[4], Multi-Way Wear Leveling[1]과 같은 마모도 평준화 기법들이 제안되었고, 쓰기 연산을 줄이기 위해 현재 데이터와 이전 데이터를 비교하여 비트단위로 변경이 발생한 셀만 쓰기 연산을 수행하는 DCW (Data-Comparison Write)[5]와 같은 기법과 DCW를 개선하여 변경이 발생한 비트가 절반이상일 때 데이터를 반전시켜서 저장시키는 Flip-N-Write[6] 기법 등이 제안되었다.

하지만 이러한 연구들은 쓰기 패턴에 따라 쓰기 연산이 자주 발생하는 데이터에 의해 메인 메모리로 사용되는 PCM의 수명이 감소되는 것을 방지할 수 없다. 이를 해결하기 위해 최근에는 캐시를 활용하여 PCM에 이루어지는 쓰기 연산의 수를 줄이는 N-Chance,

• First Author: Sang-Ho Hwang, Corresponding Author: Jong Wook Kwak
*Sang-Ho Hwang (shhwang@dgist.ac.kr), Wellness Convergence Research Center, DGIST.
**Jong Wook Kwak (kwak@yu.ac.kr), Dept. of Computer Engineering, Yeungnam University
• Received: 2017. 12. 04, Revised: 2017. 12. 20, Accepted: 2018. 01. 15.
• This research was supported by Basic Science Research Program through the National Research Foundation of Korea (NRF) funded by the Ministry of Education (2017R1D1A1A09000654).

Table 1. Memory Technologies[11],[12],[13],[20],[21]

	SRAM	DRAM	FeRAM	RRAM	NAND Flash	PCM
Cell Size(F^2)	150	10	22	30	4	4
Write Endurance	-	10^{15}	4×10^{12}	10^{11}	10^4	10^9
Read / Write time(ns)	2 / 2	20 / 20	40 / 65	100 / 100	25×10^3 / 250×10^3	12 / 100
Volatile	Volatile	Volatile	Non-Volatile	Non-Volatile	Non-Volatile	Non-Volatile

WCP (Writeback-aware Cache Partitioning), WADE (Writeback-Aware Dynamic Cache Management), AC-WAR (Adaptive and Combined Wear-out-Aware Replacement algorithm)와 같은 기법들이 제안되었다[7-10].

본 논문에서는 캐시를 활용하여 PCM 등의 비휘발성 메모리리를 메인 메모리로 활용하는 시스템에서 쓰기 연산을 줄여 시스템의 수명을 연장시키는 캐시 교체 기법을 소개하며, 해당 기법에서의 다양한 모니터링 비율에 따른 성능의 차이를 분석하여 최적의 모니터링 비율과 캐시의 분할 크기를 결정한다. 제안하는 기법은 기존의 LRU (Least Recently Used) 교체 기법과 교체 블록 선택 시 LRU 비트뿐만 아니라 더티 (dirty) 비트도 고려하는 E2Z (Early Eviction Zone)를 동시에 적용하고 있다. 제안하는 기법은 더티가 아닌 클린 (clean) 블록들의 재사용 빈도가 떨어지는 구간에서 클린 블록을 강제로 추출시키는 방법을 통해 더티 블록들이 더 많은 캐시라인을 사용할 수 있도록 유도하며, 이를 통해 더티 블록들의 히트율 (hit ration)을 높이는 방법으로 WB (Write Back) 횟수를 줄이고 있다. 더티 블록은 쓰기 연산에 의해 값이 변경되어 메인 메모리로 WB를 유발시키는 블록을 말하며, 그렇지 않은 블록은 클린 블록이다.

이하 본 논문의 구성은 다음과 같다. 2장에서는 PCM 및 PCM을 메인 메모리로 활용하는 시스템에 대한 기술과 이러한 시스템에서의 기존 연구들에 대하여 기술한다. 3장에서는 본 논문에서 제안하는 기법을 소개하고, 4장에서는 실험을 통해 기존의 기법들과 성능을 비교하고, 5장에서는 결론 및 향후 연구 과제에 대하여 기술한다.

II. Background and Related Works

1. Background

1.1 Non-volatile memory and PCM

DRAM은 지속적인 리프레시에 의한 상시 전원 소비가 많고, 좁아진 셀간 간격에 의해 Row Hammer 문제가 발생하는 등 물리적으로 집적도를 높이는 것에 대하여 한계에 달했다. 이러한 문제를 해결하기 위해 많은 기법이 제안되었는데, 최근에는 메인 메모리를 DRAM에서 비휘발성 메모리로 대체하는 논의가 활발히 이루어지고 있다. 표 1은 현재 비휘발성 메모리들을 집적도, 쓰기 횟수, 읽기 및 쓰기 연산 속도의 관점에서 비교하고 있다[11][12][13][20][21].

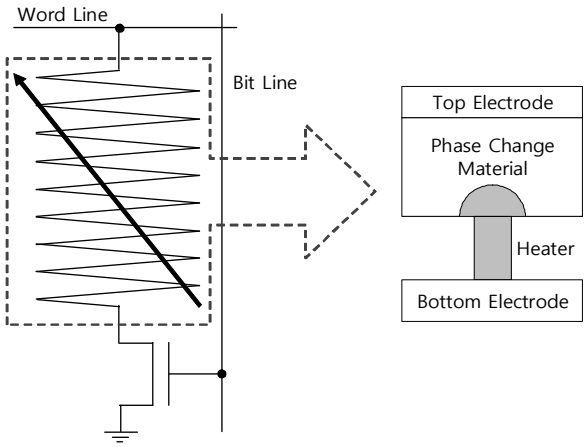


Fig. 1. Cell Structure of Phase Change Memory

PCM은 상변화를 일으키는 물질의 비정질상태에서 저항이 크고 결정질시 저항이 작은 특징을 이용하여 데이터를 저장하는 비휘발성 메모리이다. 그림 1은 PCM의 셀 구조를 보여주고 있다[14]. 상변환 물질에 1의 값을 저장하는 것을 RESET이라 하며, 이는 상변환 물질에 전류를 가하여 열(약 600 ℃)을 발생시켜 녹인 후 급속 냉각시켜 비정질상태로 만드는 것이다. 반대로 0의 값을 저장하는 것을 SET이라 하며, 이는 비정질상태를 만들 때보다 낮은 열(약 300 ℃)을 일정시간동안 가하여 결정질로 만드는 것이다. 이러한 PCM의 셀에 저장된 데이터를 읽는 것은 결정질시에는 저항 값이 작고 비정질상태에서는 저항 값이 큰 특징을 이용하는 것이다. 그림 2는 PCM의 상변환 물질의 데이터 쓰기를 위해 필요한 전류펄스(current pulse)를 보여주고 있다[6].

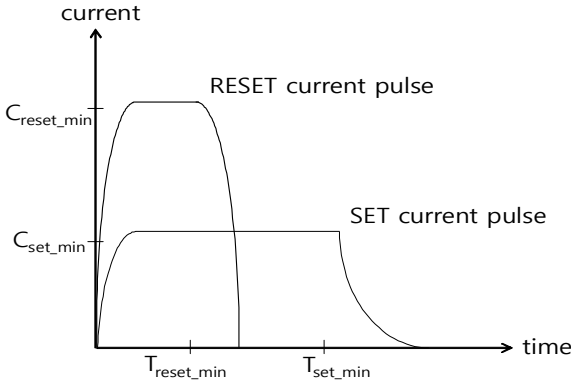
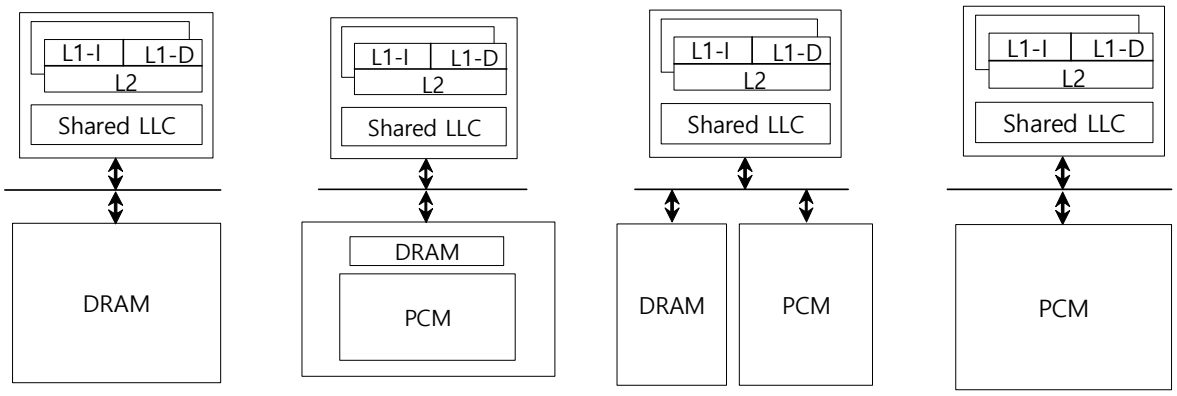


Fig. 2. Current pulses of the SET and RESET in PCM

이러한 PCM을 포함한 비휘발성 메모리들은 기존의 DRAM



(a) Conventional memory system (b) Hybrid memory system (Hierarchical organization) (c) Hybrid memory system (Parallel organization) (d) Phase change main memory system

Fig. 3. Types of main memory system organization

에 비해 전력소모가 적은 장점이 있지만, 수명에 한계가 있고 대부분 쓰기 연산 및 읽기 연산의 속도에 있어서 비대칭을 보이는 단점이 있다. 따라서 비휘발성 메모리들을 기존의 DRAM을 대체하는 메모리로 활용하기 위해서는 수명, 비대칭적인 연산 속도, 그리고 에너지 소비량에 대한 보완이 필요하다.

1.2 PCM main memory Architecture

비휘발성 메모리에는 STT-RAM, FeRAM, RRAM, PCM 등이 있으나, 대용량화에 필요한 집적도에서 가장 우수한 PCM은 DRAM을 대체하기에 가장 적합하다. 하지만 다른 모든 비휘발성 메모리들과 같이, PCM 역시 쓰기 횟수가 10^9 정도로 제한되어 있으며, 쓰기 연산과 읽기 연산이 비대칭적이다. 이러한 문제를 해결하기 위해, Qureshi et. al은 PCM을 메인 메모리로 사용하면서 DRAM을 버퍼로 활용하는 하이브리드 메인 메모리 방식을 제안하였다[2]. 기존의 메모리 시스템인 그림 3의 (a)와 비교하여, 그림 3의 (b)와 같은 PCM과 DRAM의 하이브리드 메인 메모리 방식은 DRAM 버퍼가 PCM에서 발생될 수 있는 쓰기 연산 부하를 줄여 수명과 성능을 향상시키고 있다. PCM과 DRAM의 하이브리드는 그림 3의 (c)와 같은 구조도 가능하다. 이는 PCM과 DRAM이 동등한 계층으로 구성되어 하나의 메모리 시스템을 구성하는 형태이며, 쓰기 연산을 많이 유발하는 데이터는 DRAM에 유지시키고 그렇지 않은 데이터는 PCM에 저장시키는 방법으로 PCM의 수명을 연장시키고 성능을 향상시킨다. 이러한 구조에서는 PCM에 저장될 데이터와 DRAM에 저장될 데이터를 구별하는 것이 중요하며, 이를 위해 CLOCK-DWF (Clock with Dirty bits and Write Frequency)[15]와 같이 메인 메모리에서의 페이지 교체정책에 대한 연구가 주로 이루어진다. 최근에는 그림 3의 (d)와 같이 PCM만 이용하여 메인 메모리를 구성하는 구조에 대한 연구도 활발히 진행되고 있다. 이러한 구조에서는 상위의 캐시에서 N-Chance, WCP, WADE, AC-WAR와 같은 기법을 활용하여 쓰기 연산 부하를 줄이고 있다. 이러한 기법들은 PCM에 쓰기 연산을 유발하는 WB 횟수를 줄이는 기법으로 하이브리드 형태의 구조에서도 활용될 수 있다.

2. Related Works

캐시의 효율을 높이기 위한 교체 정책은 DRAM을 사용하는 전통적인 메인 메모리 시스템에서도 많이 연구되어왔다[16]. 이 연구들은 주로 데이터 패턴을 이용하여 캐시 히트율을 높이는 기법들이며, 최근에는 비휘발성 메인 메모리 시스템을 위한 캐시 정책들이 연구되고 있다. 비휘발성 메모리들은 수명이 DRAM에 비해 상당히 짧고 쓰기 연산의 속도 또한 느리기 때문에, 이러한 비휘발성 메모리 시스템을 위한 캐시 정책들은 주로 WB을 유발시키는 더티 블록을 중점 관리하고 있다.

N-Chance는 LRU 방향의 N개의 블록들 중에 클린 블록들이 존재한다면 LRU 방향에 가장 가까운 클린 블록을 희생 블록으로 선택하는 교체 정책이다. 만약 클린 블록이 존재하지 않는다면 LRU 위치의 블록이 교체 블록으로 선택된다. 비록 N-Chance는 클린 블록을 N개의 구간 내에서 먼저 추출시키는 방법으로 더티 블록의 재사용 가능성을 높여 메인 메모리 방향으로의 WB 횟수를 줄이고 있지만, 고정적인 N값을 사용하기 때문에, 응용 프로그램에 따라 효율이 급격히 나빠지는 단점이 존재한다[7].

WADE는 메인 메모리로 WB을 자주 유발시키는 블록을 예측하는 기법이다. 캐시는 WB을 자주 유발시키는 블록과 그렇지 않은 블록으로 동적으로 분할되어 2개의 논리적인 리스트로 관리된다. 그리고 미스 발생 시에 최적의 리스트 크기와 비교하여 2개의 리스트 중에 1개를 선택하여 교체 블록을 선택하는 기법이다[9].

AC-WAR는 기존의 LRU 정책과 함께 수정된 서브 블록의 개수가 적은 블록을 먼저 교체대상으로 선택하는 LMF (Least Modified block First) 정책을 사용하여, 캐시의 블록 내에 수정된 서브 블록의 수가 많은 블록을 최대한 캐시에 유지시키고 있다. 이를 통하여 AC-WAR는 WB당 메인 메모리에 발생될 수 있는 비트 쓰기 횟수를 줄인다. 하지만 수정된 서브 블록의 수가 많은 블록이 자주 참조되지 않는 응용 프로그램에서는 효율이 떨어지는 단점이 있다[10].

AWC (Adaptive Writeback-aware Cache management)는 클린 블록과 더티 블록의 히트율을 모니터링하여 클린 블록의 히트율이 떨어지는 구간에서 클린 블록을 추출시키는 방법으로 더티

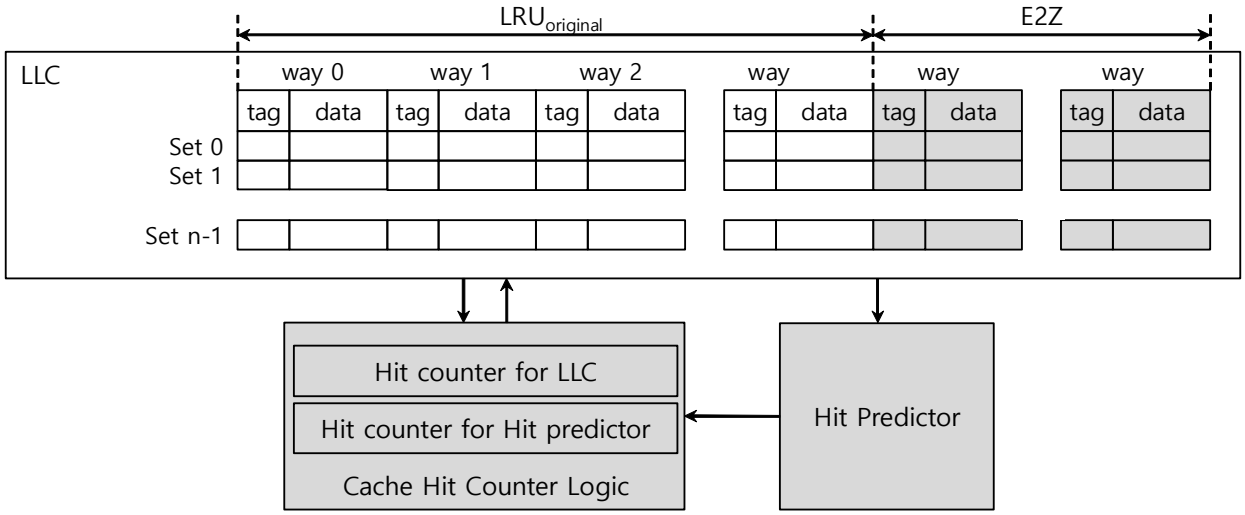


Fig. 4. The proposed writeback-aware cache replacement policy

블록의 히트율을 높인다[17]. 이를 통하여, AWC는 비휘발성 메인 메모리에서 발생될 수 있는 쓰기 연산 횟수를 줄이고 있지만, 모니터링 비율이 정적으로 고정되어 모니터링에서 최적의 샘플링 비율을 제시하지 못하였다. 또한, 성능과 모니터링 비율뿐만 아니라, 하드웨어 구현 복잡도까지 고려하였을 때 최적의 결과를 도출하는 최선의 조합을 추가로 탐색할 필요가 있다.

III. Adaptive Partition Cache Replacement Policy

본 논문에서 제안하는 기법은 캐시의 way를 기존의 LRU 정책을 사용하는 구간과 E2Z 구간으로 나누고 있다. E2Z는 교체 블록을 선택할 때 기존의 LRU 비트뿐만 아니라 더티 비트도 같이 고려하는 영역[17]으로, 이를 통하여 E2Z 내에 클린 블록이 존재하는 경우 우선적으로 교체블록으로 선택시킨다. E2Z의 크기는 모니터링을 통해 동적으로 변경된다.

1. LRU and E2Z

응용 프로그램에 따라 차이는 있지만, 각 way에서 히트율(hit ratio)이 차지하는 비율은 대체적으로 MRU 위치에 가까울수록 높고 LRU 위치에 가까울수록 떨어진다. 이러한 비율은 WB을 유발하는 더티 블록과 그렇지 않은 클린 블록에서도 차이가 있다. 제안하는 기법은 클린 블록의 히트율이 떨어지는 way 구간을 E2Z로 정하여 더티 블록의 재사용 거리를 늘려서 PCM에 발생될 수 있는 WB 횟수를 줄인다. 재사용 거리란 캐시에 저장된 블록이 다음 재사용 시점까지 캐시에 존재하기 위한 해당 associativity 내의 최소한의 way 개수를 의미한다. 그림 4는 w-way 캐시에서의 제안하는 구조를 보여주고 있다.

LLC에 삽입되는 데이터는 LRU 정책 구간의 MRU 위치에 저장된다. 이 구간에 존재하는 데이터들은 모두 LRU 정책에 따라 동작 하게 되며 (MRU 위치로 이동), LRU 위치에 도달한 블록은 2가지 환경변수인 1) 미스 또는 히트, 2) 클린 블록 또는 더티 블록인 경우에 따라 E2Z에 삽입되거나 혹은 그대로 추출된다.

첫 번째로 캐시에서 미스가 발생되면, 데이터를 저장할 블록을 확보하기 위해 교체 블록을 선택해야 하는데, 기존의 LRU 정책 구간에서 LRU 위치에 해당하는 블록이 클린 블록인 경우에는 이 블록을 교체 블록으로 선택한다. 이 경우에는 E2Z로 이동되는 블록이 없다. 만약 LRU 위치에 해당하는 블록이 더티 블록인 경우에는 E2Z에서 교체 블록을 선택하고, LRU 정책 구간의 LRU 위치에 있는 더티 블록은 E2Z로 이동된다. E2Z의 교체 블록 선택은 더티 비트와 LRU 비트를 동시에 고려하여 결정된다. 즉, 클린 블록이 존재하는 경우에는 클린 블록 중에 LRU 위치에 가장 가까운 블록이 선택되며, 클린 블록이 존재하지 않는 경우에는 LRU 위치에 해당하는 블록이 교체 블록으로 선택된다.

두 번째로 E2Z 내에 존재하는 블록이 참조가 될 경우, 해당 블록은 LRU 정책 구간의 MRU 위치로 이동하게 되며, 이에 따라 LRU 정책 구간의 LRU 위치의 블록은 자동적으로 E2Z의 MRU 위치로 이동된다. 이때 이동되는 블록은 더티 블록일 수도 있고, 클린 블록일 수도 있다.

그림 5는 6-way의 LLC 구조에서 LRU를 적용하는 구간과 E2Z가 각각 3-way 크기임을 가정한 상황에서 제안하는 기법의 동작을 보여주고 있다. 최초 상태에서 전체 LRU 위치의 블록 f는 기존의 LRU 정책만을 사용하는 구조에서는 미스가 발생하고 그에 따라 WB을 더 유발하게 되지만, 제안하는 구조에서는 참조되어 WB가 발생되는 상황을 방지시키는 것을 확인할 수 있다.

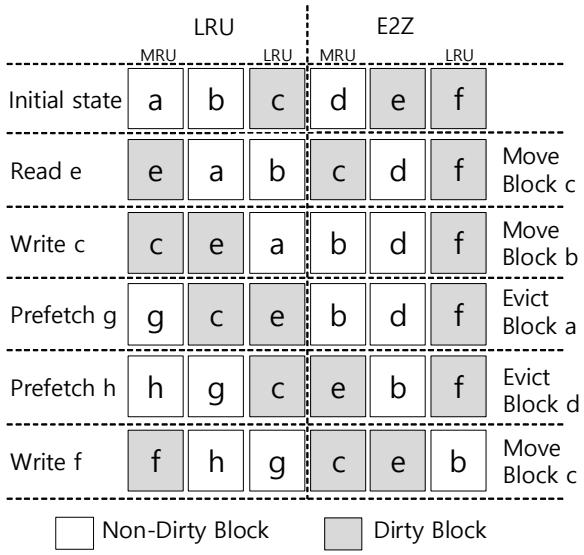


Fig. 5. Example replacement of the proposed scheme

2. Adaptive Partition Cache

제안하는 기법은 동적으로 E2Z의 크기를 결정한다. 효율적인 캐시 분할지점을 찾기 위해, 제안하는 구조는 전체 set을 모니터링 하는 대신에 일부 set을 일정한 사이클 (cycle) 동안 모니터링 하여 재사용 거리에 따른 히트율을 계산한다. 제안하는 기법의 E2Z의 크기 결정을 위한 모니터링은 AWC[17]에서 제시한 모니터 기법을 활용하고 있으며, 본 논문에서는 전체 LLC의 set 중에서 모니터링하는 set의 비율을 다양하게 변화시켜 하드웨어 복잡도 대비 최적의 E2Z의 크기를 결정한다.

적응적으로 캐시 분할 지점을 찾기 위한 모니터링 기법은 히트 카운터(Hit Counter)와 히트 예측기(Hit Predictor)로 이루어져 있다. 히트 카운터는 재사용 거리에 따른 LLC 블록들의 히트율을 확인하기 위해서 LRU 위치에서 블록이 참조가 되었을 때 1씩 증가시킨다. 즉, 히트 카운터는 해당 거리에서 얼마나 많은 블록들이 재참조 되는지를 보여준다. 이러한 히트 카운터는 더티 블록과 클린 블록에 대하여 각각 별도로 집계 한다.

히트 예측기는 E2Z의 크기를 늘리는 경우 히트율이 어떻게 변화되는지를 관찰하는 모듈이다. 이를 위해서 히트 예측기는 모니터링 되는 set 당 LLC way 길이만큼의 더티 블록의 태그만 저장하는 큐 (Queue) 구조와 1개의 클린 블록의 태그를 저장하는 버퍼를 사용한다. 큐 구조에는 LLC에서 추출되는 더티 블록의 태그만 저장하며, 큐에서 빠져나가지 전까지 참조가 된다면 어느 위치에서 참조가 되는지를 관찰한다. 즉, 이는 E2Z를 늘리는 경우 더티 블록의 참조 비율이 얼마나 변화 되는지를 확인하는 것이다. 반대로 LLC에서 추출되는 클린 블록은 버퍼에 저장된다. 이 버퍼구조는 LRU 정책을 적용하는 구간을 늘리는 경우에 클린 블록의 참조가 얼마나 증가될 수 있는지를 관찰하기 위한 용도로 사용된다.

히트 예측기는 LLC 전체 set이 아닌 일부를 모니터링한다. 전체를 모니터링하는 것은 최적의 분할지점을 찾을 수 있으나 모니터링을 위해 추가되는 메모리 공간의 오버헤드가 큰 단점이 있다. 이러한 모니터링에 따른 추가 메모리 오버헤드를 줄이기 위해 본 논문에서는 모니터링 되는 set을 결정하고 이 set의 태그만 히트 예측기에 저장된다. 모니터링 set에 대한 최적의 샘플링 비율은 실험에서 밝힌다.

일정한 사이클 동안 집계되는 각 구조에서의 재사용 거리에 따른 히트 횟수는 분할 지점을 결정하기 위해 사용된다. 만약 히트 예측기상에서 더티 블록의 히트횟수가 LRU 정책 구간 내에서 클린 블록의 히트횟수보다 많은 구간이 존재한다면 E2Z의 크기를 증가시킨다. 이는 E2Z 크기를 증가시켰을 때 더티 블록의 히트율 향상에 도움이 된다는 의미이다. E2Z의 크기를 증가시켰을 때 이득을 볼 수가 없는 경우에는 반대로 E2Z의 LRU 위치의 참조횟수와 히트 예측기내의 클린 블록 버퍼의 참조횟수를 비교한다. 만약 클린 블록 버퍼의 참조횟수가 많은 경우에는 LRU 정책 구간의 크기를 늘린다. 즉, w-way 캐시에서 현재 E2Z의 시작 위치가 s이고 히트 예측기 내의 더티 블록 태그 참조횟수가 LRU 정책 구간에서 클린 블록의 참조횟수보다 높은 구간의 크기가 n일 때, 다음의 캐시 분할지점은 “s – n”이 된다. 만약 n이 0이고 E2Z내에 LRU 위치의 더티 블록의 참조횟수가 히트 예측기 내에 있는 클린 블록 태그 참조횟수보다 적다면 다음의 캐시 분할지점은 “s + 1”이 된다.

IV. Performance Evaluation

1. Experiment Setup

본 논문에서 제안하는 기법의 성능을 평가하기 위해 gem5[18] 시뮬레이터를 사용하였다. 표 2는 실험에서 사용한 환경을 보여주고 있다.

Table 2. Experimental environment

Parameter	Value
CPU	3GHz
L1 instruction	32KB, 4-way, 64B line2 cycle latency
L1 data	32KB, 4-way, 64B line2 cycle latency
L2	2MB, 16-way, 64B line 20 cycle latency
Memory	4GB

실험에서는 SPEC 2006[19] 벤치마크 프로그램을 사용하였다. 우선 실험의 정확성을 위해 fast forwarding을 10억으로 설정하여 준비 단계의 결과를 제외하였고, 그 뒤 10억 개의 명령어에 대한 결과를 활용하여 성능을 비교하였다. 제안하는 기법의 성능 평가를 위해 N-Chance와 AC-WAR를 실행 시간 및 메모리에 쓰기를 유발시키는 WB 관점에서 비교하였다. N-Chance의 N 값은 8로 설정하였고, AC-WAR의 d 값은 1, 3, 5를 사용하였다. 제안하는 기법의 모니터링 사이클은 10만 사이클이다.

2. Performance Evaluation

본 논문에서 제안하는 기법의 성능비교는 WB 횟수 및 수행시간 관점에서 수행한다. 수행시간의 변화는 캐시의 히트율과 관련되어 있어 히트율이 개선되면 수행시간은 줄어든다.

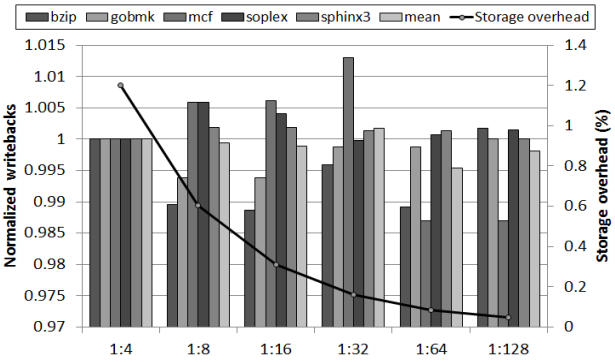


Fig. 6. Normalized writebacks according to monitoring ratio

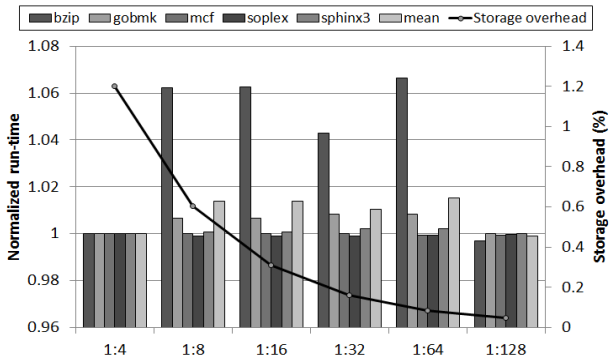


Fig. 7. Normalized runtime according to monitoring ratio

그림 6과 그림 7은 제안하는 기법의 샘플링 비율에 따른 WB 변화율 및 수행시간을 보여주고 있다. WB 횟수 관점에서 1:4 ~ 1:128의 변화율은 1:4를 기준으로 -0.5%에서 0.2%의 변화율로 대체적으로 변화율이 크지 않다. 수행시간 관점에서 1:4 ~ 1:128의 변화율은 1:4를 기준으로 -0.08%에서 1.3%를 보인다. 이러한 결과를 바탕으로 수행시간 관점에서는 1:128이 모니터링 샘플링 비율로 적절하다고 판단된다. 샘플링 비율에 따른 WB 횟수 및 수행시간에서는 WB 횟수가 감소하면 그에 따라 수행시간은 증가하는 패턴을 보인다.

Table 3. Storage overhead

Sampling Ratio	Size of Hit Predictor (byte)	Size of Counter (byte)	Storage overhead(%)
1:1	100096	200	4.78
1:2	50048	200	2.39
1:4	25024	200	1.20
1:8	12512	200	0.60
1:16	6256	200	0.30
1:32	3128	200	0.15
1:64	1564	200	0.08
1:128	782	200	0.04

표 3은 제안하는 기법의 모니터링 비율에 따른 공간 복잡도를 보여주고 있다. E2Z의 크기를 결정하기 위한 히트 예측기는 모니터링 set마다 17개의 태그를 저장하는 큐와 버퍼가 있다. 2MB LLC에서 태그의 크기는 23bit이며 1개의 모니터링 set은 391bit(23 bit × (1 buffer + 16 queue))의 공간이 필요하다. 또한 모니터링 set의 모든 참조횟수를 가산하기 위한 카운터가 히트 예측기에 17개, LLC의 클린 및 더티 블록을 위해 각각 16개, 모니터링 사이클 주기를 위해 1개로 총 50개(17 + (16 × 2) + 1)가 존재한다. 즉, 카운터를 4byte 크기로 했을 때 200byte의 공간이 추가로 필요하다. 표 3에서는 샘플링 비율에 따른 공간 복잡도를 보여주고 있으며, 1:4에서 1:128의 비율에서 공간 복잡도가 LLC 크기를 기준으로 4.78%에서 0.04%이다. 그림 6과 그림 7의 결과에 따라 고려한 1:128 비율로 샘플링 했을 때에는 제안하는 기법이 약 0.04%의 공간 복잡도를 보이고 있다. 이는 AC-WAR에서의 공간 복잡도인 3%에 비해 매우 적으며, 추가 메모리 복잡도 측면에서 사실상 무시할만한 수준이다.

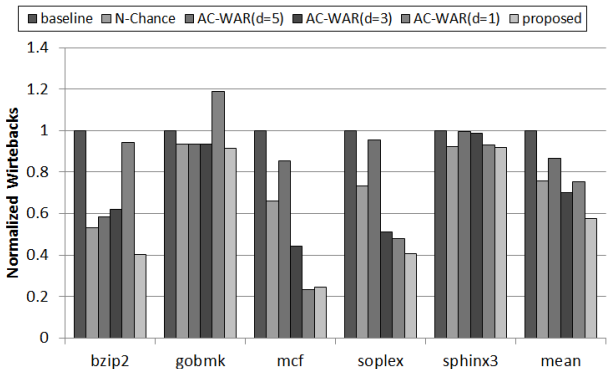


Fig. 8. Normalized writebacks

추가 메모리 공간 대비 WB 감소 효율은 1:4 ~ 1:128에서 각각 0.1, 0.34, -1.1, 5.5, 4.1로 1:64와 1:128에서 감소효율이 높다. 추가 메모리 공간 대비 실행시간 증가 오버헤드에서는 1:4 ~ 1:128에서 각각, 2.2, 4.7, 6,6, 17.9, -1.8로 1:128에서만 실행시간이 감소하였다. 따라서, 추가 메모리 공간 복잡도, WB 및 수행시간을 모두 고려했을 때 샘플링 비율은 1:128로 설정하는 것이 효율적이며, 이에 따라 그림 8과 그림 9에서의 비교에서 제안하는 기법의 샘플링 비율은 1:128로 한다.

그림 8은 WB 횟수 관점에서 제안하는 기법을 다른 기법들과 비교하여 보여주고 있다. AC-WAR는 d 값이 작을수록 WB 횟수가 적다. 이는 d 값이 작을수록 히트율을 희생시켜 WB 횟수를 줄이고 있기 때문이다. AC-WAR는 더티 블록의 재사용 거리가 길거나 WB을 많이 유발시키는 응용 프로그램에서 대체적으로 N-Chance에 비해 좋은 효율을 보이고 있으나, 더티 블록의 재사용 거리가 짧고 WB을 적게 유발하는 응용 프로그램에서는 성능이 나빠지고 있다. 이는 AC-WAR의 LMF에 저장된 더티 블록 가운데 수정된 서브 블록을 많이 포함하는 더티 블록이 자

주 참조되지 않아 캐시의 효율이 급격하게 나빠지기 때문이다. 제안하는 기법은 응용 프로그램의 특성과 관련된 구별 없이 대체적으로 비교 기법들에 비해 WB 횟수를 현저히 줄이고 있다. 평균적으로, 제안하는 기법은 LRU, N-Chance와의 비교에서 약 42%, 23%의 WB 횟수를 줄이고 있고, AC-WAR와의 비교에서는 d=5, d=3, d=1에서 각각 33%, 17%, 23%의 WB을 줄이고 있다. 이를 통해 줄어드는 WB 횟수는 메인 메모리로 사용되는 PCM의 수명을 연장시키는데 직접적으로 기여하게 된다.

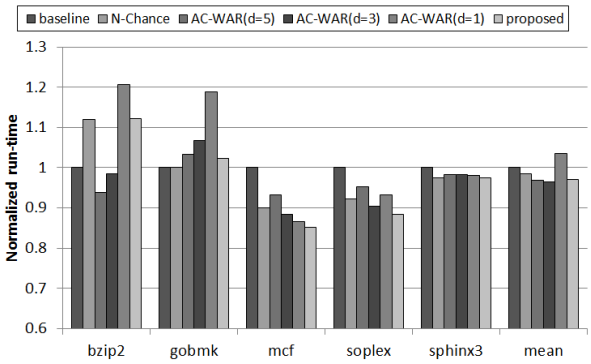


Fig. 9. Normalized runtime

그림 9는 제안하는 기법의 수행시간을 N-Chance 및 AC-WAR와 비교하여 보여주고 있으며, 응용 프로그램별로 기존의 LRU 정책을 적용한 결과를 기준으로 정규화하고 있다. AC-WAR는 d 값에 따라 수행 시간에 차이를 보여주고 있으며, d 값이 작을수록 수행 시간이 증가하고 있다. 이는 d 값이 작을수록 캐시 적중 실패율이 높아지기 때문이다. 평균적으로, 제안하는 기법은 AC-WAR(d=5) 및 AC-WAR(d=3)에 비해 각각 0.2%, 0.5% 실행 시간의 증가가 발생하였지만, 이는 실행 구동형 시뮬레이터의 특성 상 무시할 만한 수준이다. 오히려, 기존의 LRU, N-Chance, AC-WAR(d=1)과의 비교에서는 약 3%, 1.4%, 6.6%의 수행시간을 줄이고 있다.

V. Conclusions

본 논문에서는 캐시의 교체정책을 활용하여 PCM 등의 비휘발성 메모리를 메인 메모리로 활용하는 시스템에서 쓰기 연산의 횟수를 줄이고, 이를 통해 시스템의 수명을 연장시키는 환경을 소개하고, 해당 기법에 대한 최적의 모니터링 비율과 이와 연관된 파티션 사이즈를 결정하였다. 제안하는 기법은 기존의 LRU 교체 기법과 교체 블록 선택 시 LRU 비트뿐만 아니라 더티 비트도 고려하는 E2Z를 혼합 적용하여 더티 블록들의 히트율을 높이는 방법으로 WB 횟수를 줄이고 있다. 또한 제안하는 기법은 히트 예측기와 캐시 히트 카운터 로직을 활용하여 성능이 최적이 되는 블록의 재사용 거리를 동적으로 예측하여 대부분의 워크로드에서 성능을 개선시키고 있다. 실험을 통해, 주어진 기법은 모니터링 비율 측면에서 1:128의 비율이 WB 횟수와 실행 시간 측면에서 최적이라고 판단하였다. 해당 비율을 활용하였을 경우, 기존의 LRU, N-Chance, AC-WAR(d=5)와 비교하여 해당 비율 구성은 수행 시간의 감소가 거의 없이 WB 횟수를 최대 42%, 23%, 33% 줄이고 있다. 또한 모니터링 모듈은 40bit 메모리 주소를 활용할 경우 약 0.04%의 추가 메모리 오버헤드가 발생한다. 하지만 이는 추가 메모리 복잡도 측면에서 무시할만한 수준이라고 판단되며, 추후 모니터링 모듈의 동작 예측 정확도 향상을 위해 해당 모듈의 동작 알고리즘을 보다 더 정교화 해 나갈 예정이다.

REFERENCES

[1] Yu, Hongliang, and Yuyang Du. "Increasing Endurance and Security of Phase-Change Memory with Multi-Way Wear-Leveling." IEEE Transactions on Computers, Vol. 63, No. 5, pp. 1157-1168, May 2014.

[2] Qureshi, Moinuddin K., Vijayalakshmi Srinivasan, and Jude A. Rivers. "Scalable high performance main memory system using phase-change memory technology." ACM SIGARCH Computer Architecture News, Vol. 37, No. 3, pp. 24-33, June 2009.

[3] Qureshi, Moinuddin K., et al. "Enhancing lifetime and security of PCM-based main memory with start-gap wear leveling." Proceedings of the 42nd Annual IEEE/ACM International Symposium on Microarchitecture, pp. 14-23, 2009.

[4] Seong, Nak Hee, Dong Hyuk Woo, and Hsien-Hsin S. Lee. "Security refresh: prevent malicious wear-out and increase durability for phase-change memory with dynamically randomized address mapping." ACM SIGARCH computer architecture news, Vol. 38, No. 3, pp. 383-394, June 2010.

[5] Yang, Byung-Do, et al. "A low power phase-change random access memory using a data-comparison write scheme." 2007 IEEE International Symposium on Circuits and Systems, pp. 3014-3017, May 2007.

[6] Cho, Sangyeun, and Hyunjin Lee. "Flip-N-Write: a simple deterministic technique to improve PRAM write performance, energy and endurance." 2009 42nd Annual IEEE/ACM International Symposium on Microarchitecture (MICRO), pp. 347-357, Dec 2009.

[7] Ferreira, Alexandre P. et al. "Increasing PCM main memory lifetime." Proceedings of the conference on design, automation and test in Europe. European Design and Automation Association, pp. 914-919, 2010.

[8] Zhou, Miao, et al. "Writeback-aware partitioning and replacement for last-level caches in phase change main memory systems." ACM Transactions on Architecture and Code Optimization (TACO), Vol. 8, No. 4, Article No. 53, January 2012.
[9] Wang, Zhe, et al. "WADE: Writeback-aware dynamic cache management for NVM-based main memory system." ACM Transactions on Architecture and Code Optimization (TACO), Vol. 10, No. 4, Article No. 51, December 2013.
[10] Abad, Pablo, et al. "AC-WAR: Architecting the Cache Hierarchy to Improve the Lifetime of a Non-Volatile Endurance-Limited Main Memory." IEEE Transactions on Parallel and Distributed Systems, Vol. 27, No. 1, pp.66-77, January 2016.
[11] Torres, Lionel, et al. "Trends on the application of emerging nonvolatile memory to processors and programmable devices." 2013 IEEE International Symposium on Circuits and Systems (ISCAS2013), pp. 101-104, May 2013.
[12] Mittal, Sparsh, and Jeffrey S. Vetter. "A survey of software techniques for using non-volatile memories for storage and main memory systems." IEEE Transactions on Parallel and Distributed Systems, Vol. 27, No. 5, pp. 1537-1550, May 2016.
[13] Mittal, Sparsh, Jeffrey S. Vetter, and Dong Li. "A survey of architectural approaches for managing embedded DRAM and non-volatile on-chip caches." IEEE Transactions on Parallel and Distributed Systems, Vol. 26, No. 6, pp. 1524-1537, June 2015.
[14] Xia, Fei, et al. "A survey of phase change memory systems." Journal of Computer Science and Technology, Vol. 30, No. 1, pp. 121-144, January 2015.
[15] Lee, Soyoon, Hyokyung Bahn, and Sam H. Noh. "Clock-dwf: A write-history-aware page replacement algorithm for hybrid pcm and dram memory architectures." IEEE Transactions on Computers, Vol. 63, No. 9, pp. 2187-2200, September 2014.
[16] Jaleel, Aamer, et al. "High performance cache replacement using re-reference interval prediction (RRIP)." ACM SIGARCH Computer Architecture News, Vol. 38. No. 3, pp. 60-71, June 2010.
[17] S. Hwang et al. "Adaptive Writeback-aware Cache Management Policy for Lifetime Extension of Non-volatile Memory" Journal of Semiconductor Technology and Sciecne, Vol 17, No. 4, August 2017.
[18] Martin, Milo MK, et al. "Multifacet's general execution-driven multiprocessor simulator (GEMS) toolset." ACM SIGARCH Computer Architecture News, Vol. 33, No. 4, pp. 92-99, November 2005.
[19] J. L. Henning, “Spec cpu2006 benchmark descriptions,” ACM SIGARCH Computer Architecture News, Vol. 34, No. 4, pp. 1-17, September 2006.
[20] Hwang, Sang-Ho, et al. "CL-Tree: B+ tree for NAND Flash Memory using Cache Index List." Journal of The Korea Society of Computer and Information, Vol. 20, No. 4, pp. 1-10, April, 2015.
[21] Kim, Seon Hwan, et al. "Sampling-based Block Erase Table in Wear Leveling Technique for Flash Memory." Journal of The Korea Society of Computer and Information, Vol. 22, No. 5, pp. 1-9, May, 2017.

Authors

Sang-Ho Hwang received the B.S., M.S. and Ph.D. degrees in Computer Engineering from Yeungnam University, Korea, in 2009, 2013 and 2017, respectively. He is currently a researcher in the Wellness Convergence Research Center, Deagu Gyeongbuk Institute of Science & Technology. His current research interests include non-volatile memory systems and machine learning.

Jong Wook Kwak received the B.S. degree in Computer Engineering from Kyungpook National University, Taegu, Korea in 1998, the M.S. degree in Computer Engineering from Seoul National University, Seoul, Korea in 2001, and the Ph.D. degree in Electrical Engineering and Computer Science from Seoul National University, Seoul, Korea in 2006, respectively. From 2006 to 2007, he was a Senior Engineer with the system-on-chip (SoC) Research and Development Center, Samsung Electronics Company, Ltd., Suwon, South Korea. During 2011-2012, he was a Guest Researcher at the Research Institute of Advanced Computer Technology, Seoul National University. During 2012-2013, he was a Visiting Scholar at the Georgia Institute of Technology, Atlanta, GA, USA. He is currently an Associate Professor with the Department of Computer Engineering, Yeungnam University, Gyeongsan, South Korea. His current research interests include SoC design, advanced processor architecture, low-power mobile embedded system design, and high-performance parallel and distributed computing.