새도우 블록: 스토리지 클래스 메모리의 블록 입출력 원자성 보장 및 캐시 이슈

최정헌*, 정재민*, 원유집*
*한양대학교 전자컴퓨터통신공학과
e-mail : rhythmical@ece.hanyang.ac.kr

Shadow Block: Guaranteeing Atomicity of Block I/O in Storage Class Memory and Cache issue

Jeongheon Choi*, Jaemin Jung*, Youjip Won*
*Division of Electrical and Computer Engineering, Hanyang University

요 약

비휘발성 나노 저장 소자는 고속의 바이트 단위 접근성과 함께 비휘발성을 동시에 갖고 있다. 이와 같은 특징은 차세대 장치로 주목 받을 만큼 오늘날의 컴퓨터 구조에 큰 변화를 줄 수 있는 잠재력을 갖고 있으며 이를 접목한 시스템적인 개발 역시 활발하게 진행되고 있다. 본 논문에서는 기존의 메인 메모리와 나노 저장 소자가 융합된 스토리지 메모리 클래스(SCM) 환경 하에서 입출력 시에 원자성(Atomicity)이 보장되도록 설계, 구현된 새도우 블록 기법을 소개하고, 더불어 캐시를 사용하며 발생할 수 있는 데이터 일관성 처리의 보장을 다루었다. 또한 실제 FRAM 이 장착된 하드웨어 환경 하에서 개선된 새도우 블록을 동작하여 측정한 성능 결과를 함께 제공한다.

1. 서론

최근 들어 MRAM (Magnetic RAM), FRAM (Ferro-electric RAM), PRAM (Phase-change RAM) 등과 같은 나노소자 기반의 차세대 비휘발성 저장 소자에 대한 개발이 활발히 진행되고 있다. 위 메모리들은 고속의 바이트 단위 읽고 쓰기가 가능하며 또한 비휘발성의 특징도 동시에 가지고 있다.

이러한 특성의 비휘발성 나노 저장 소자는 주 기억장치의 역할에 저장장치의 역할을 함께 가질 수 있어 시스템 구성 면에서 이를 접합한 많은 개선된 사항이 필요할 것이라 판단된다.

본 논문은 다음의 세 항목으로 구성되어 있다. 첫 번째는 나노 저장 소자가 결합된 시스템 환경 하에서 블록 단위의 입출력 요구 시에 원자성이 보장되도록 설계, 구현된 새도우 블록 방식[1]을 소개하는 것이고, 이어서 새도우 블록 기법의 원자성 보장을 위한 처리 과정에서 캐시를 사용하며 발생할 수 있는 캐시 이슈의 소개 및 제어과정을 다루며, 마지막으로 실제로 FRAM 을 이용하여 구성한 하드웨어 환경에서 이를 평가하는 것이다.

2. 스토리지 클래스 메모리 기반의 새도우 블록 방식

2.1 스토리지 클래스 메모리 (SCM)

오늘날의 컴퓨터 시스템은 입출력 속도의 차이로 인해 메모리와 스토리지가 분리된 계층적 구조로 구성된다. 이 분리된 두 계층간의 간격을 좁히고 나아가 메모리-스토리지 융합까지 잠재할 수 있는 시스템 구성을 위해 메인 스토리지를 비휘발성 나노 저장 소자로 구성하였으며, 이를 스토리지 클래스 메모리(SCM)라고 명명한다.

시스템의 계층적 구조로 인해 개별 레이어 마다의 작업 수행에 대한 기본 단위(unit) 설정은 필수적이며, 또한 이는 기본 단위의 확립을 통해 입출력 연산의 원자성을 보장할 수 있게 됨을 의미한다.

SCM 의 경우 메모리적 특징으로 인해 입출력의 기본 단위는 word 이지만, 스토리지적 측면에서 block 단위의 입출력이 이루어진다. 이러한 SCM 환경을 고려하여 block 단위에서의 원자성이 보장된 입출력 인터페이스인 새도우 블록을 구현하였다.

2.2 새도우 블록(Shadow Block)

SCM 환경하에서 블록 단위의 입출력 작업에 대한 원자성을 보장해주기 위하여 개발한 새도우 블록 기법의 기본 아이디어는 데이터베이스에서 사용되는 새도우 페이징 기법에서 착안하였다

새도우 블록 기법을 통하여 특정 블록의 데이터 갱신 요청을 하게 되면 원본 데이터에 직접 갱신을 하는 대신, 별도의 블록(*shadow_block*)에 이를 쓰고 원본 데이터에 대한 접근 경로를 현재 갱신 데이터가 쓰여진 블록으로 대체한다. 이는 외부에서 SCM 에 접근하는데 사용되는 논리적 블록주소와 실제 SCM 에 접근하는데 필요한 물리적 블록주소 간의 맵핑 테이블(*M*)이 존재하여 이를 통해 SCM 의

관리가 이루어짐을 의미한다. 또한 새도우 블록 기법은 별도의 관리 데이터를 사용하여 원자성 보장이 가능하도록 구현되었다. 맵핑 테이블과 관리 데이터들은 SCM 의 특정 영역에 위치한다.

```
Procedure SHADOW_BLOCK_WRITE ( block_no, buf )
    shadow_block = get_address( nshadow ); // S0
Ⓐ   l = block_no;                          // S1
Ⓑ   tmp = M[ l ];                          // S2
    MEMCPY( shadow_block, buf );
Ⓒ   M[ l ] = nshadow;                      // S3
Ⓓ   nshadow = tmp;                         // S4
Ⓔ   l = NULL;                              // S5
Ⓕ   tmp = NULL;                            // S0
End procedure
```

(그림 1) 새도우 블록 기법의 쓰기 연산

그림 1 의 알고리즘과 같이 새도우 블록 기법의 쓰기 연산 과정에서 관리 데이터들(l, tmp, nshadow)의 갱신이 순차적으로 이루어짐을 확인할 수 있다. 이는 쓰기 연산 도중 예기치 않은 사태로 인해 연산이 완료되지 못한 채 멈추어 부분(Partial) 쓰기가 되는 문제의 해결 방안으로서, 복구 시점의 관리 데이터들의 상태는 원자성 체크의 판단 기준이 된다. 새도우 블록 쓰기 연산에 대한 원자성 체크 정책 및 복구 처리 정책은 표 1 과 같다.

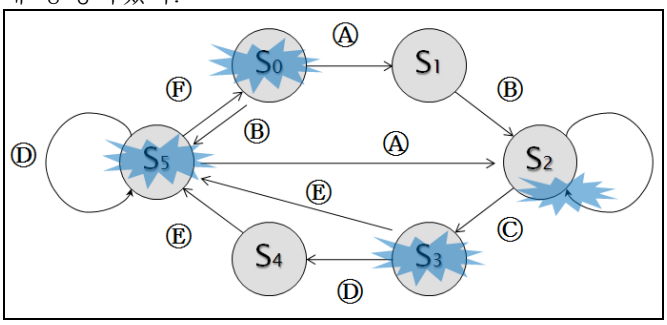
state	l	tmp	n_{shadow}	$M[l]$	action
S_0	*null*	*null*	B	C	*none*
S_1	A	*null*	B	C	*undo*
S_2	A	C	B	C	*undo*
S_3	A	C	B	B	*redo*
S_4	A	C	C	B	*redo*
S_5	*null*	C	C	B	*redo*

<표 1> 새도우 블록 진행 단계별 상태 및 복구작업

3. 캐시 이슈

새도우 블록의 원자성은 진행단계에 따라 순차적인 관리 데이터의 갱신을 통해 보장된다. 하지만 DMA 와 같은 별도의 하드웨어적인 지원이 없는 상태에서의 블록 입출력 요청은 수많은 load-store 연산으로 수행되며, 이 과정에서 CPU 캐시가 함께 동작하면 런타임 시점에 데이터 갱신 처리를 실행하였어도 실제 SCM 에서의 데이터가 갱신되었음을 보장할 수 없는 문제가 발생한다.

이러한 문제를 방지하기 위하여 그림 1 의 상태 값이 변경되는 a~f 의 연산 처리 시에 즉각적인 SCM 의 반영이 이루어져야 한다. 이는 캐시 클린(Cache Clean) 요청을 명시적으로 처리함을 의미하며, 적절한 캐시 클린 타이밍의 선정은 그림 2 와 같이 오토마타를 이용해 증명하였다.



(그림 2) 캐시 클린 시점

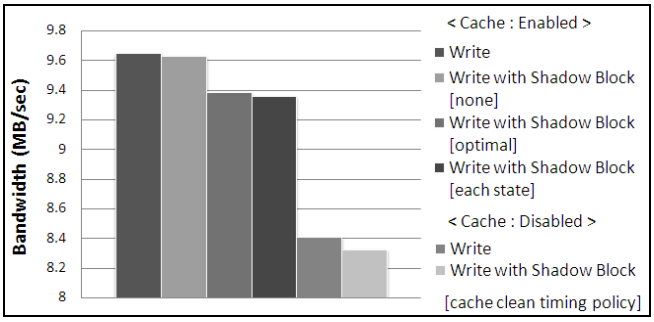
매 연산 처리마다 캐시 클린 요청을 하는 대신, 그림 2 와 같이 네번의 캐시 클린만으로 새도우 블록 기법의 원자성이 보장 가능하다.

4. 성능측정

SCM 환경은 8MB FRAM 을 SMDK2440 임베디드 보드에 장착하여 구성하였고, ARM920T 마이크로 프로세서와 16KB D 캐시를 사용한다.

캐시 이슈의 방지를 위해 개선된 새도우 블록 기법을 구현하여 보드에서의 성능 측정을 하였으며, 이와 함께 캐시 클린이 동반되지 않는 기존 새도우 블록 기법과 새도우 블록 기법이 적용되지 않은 순수한 블록 쓰기도 함께 실험하여 동일한 조건하에서의 개별 오버헤드를 측정하였다.

그림 3 의 결과를 보면 캐시 클린으로 인해 성능 저하가 발생하지만 캐시를 사용함으로 인한 성능 향상폭이 더 큼을 알 수 있으며, 매 시점마다의 캐시 클린 요청에 비교하여서도 적절한 시점만의 캐시 클린 요청이 효율적임을 알 수 있다.



(그림 1) 새도우 블록 방식의 오버헤드

참고문헌

[1] Jaemin Jung, Jeongheon Choi, Youjip Won. and Sooyong Kang. Shadow Block: Imposing Block Device Abstraction on Storage Class Memory, International Workshop on Software Support for Portable Storage(IWSSPS2009). 15 October 2009. Grenoble, France.

[2] R. Freitas and W. Wilcke. Storage-class memory: The next storage system technology. IBM Journal of Research and Development, 52(4-5):439–448, 2008.