http://dx.doi.org/10.7236/JIWIT.2012.12.2.35
JIWIT 2012-2-5

낮은 에너지 소모와 공간 오버헤드의 Last Level Cache 신뢰성 향상 기법

Improving Reliability of the Last Level Cache with Low Energy and Low Area Overhead

김영웅*

Young-Ung Kim

요 약 반도체 집적 기술의 발전은 단위 면적당 더 많은 캐쉬 메모리를 프로세서 내에 적재할 수 있도록 하였으나, 이로 인하여 프로세서는 소프트 에러에 대해 더 취약해지는 추세이며, 이는 설계 고려사항 중 신뢰성의 비중이 점점 더 커짐을 의미한다. 본 연구에서는 캐쉬 메모리 계층 중 소프트 에러에 가장 취약한 Last Level Cache에 대하여 낮은 에너지 소모와 공간 오버헤드를 갖는 저비용의 신뢰성 향상 기법에 대하여 제안하고 실험하였다. 실험 결과 소프트 에러에 대해 95.4%의 높은 에러 보호율을 보였으며, 성능은 단지 0.26%이하로 저하되었다. 또한 추가적인 에너지는 2.96%만 요구되었다.

Abstract Due to the technology scaling, more transistors can be placed on a cache memories of a processor. However, processors become more vulnerable to the soft error because of the highly integrated transistors, and consequently, the reliability of the cache memory must consider seriously at the design space level. In this paper, we propose the reliability improving technique which can be achieved with low energy and low area overheads. The simulation experiments of the proposed scheme shows over 95.4% of protection rate against the soft error with only 0.26% of performance degradations. Also, It requires only 2.96% of extra energy consumption.

Key Words : Last Level Cache, Reliability, Soft Error, Narrow Value Duplication

Ⅰ. 서 론

반도체 공정 기술은 수 조 개에서 수십 조 개 이상의 트랜지스터들을 하나의 칩 안에 장착할 수 있는 수준으로 발전하였다. 하지만 트랜지스터 수의 급격한 증가는 프로세서가 소프트 에러에 취약해지는 원인이 되었으며, 이는 신뢰성이 프로세서의 설계 단계에서 주요한 고려사항 중 하나가 되었음을 의미한다.

소프트 에러는 외부의 요인에 의하여 SRAM으로 구성된 메모리 내의 비트 값이 역전되는 현상이며 이는 불안정한 공급 전원, 타이밍 에러 또는 외부로 부터의 알파 입자(Alpha Particle) 유입 등으로 인하여 발생한다[1]. 프로세서는 이와 같은 발생원인들 중 알파입자 유입으로 인한 소프트 에러에 가장 취약하며 또한 예측하기가 어

*정회원, 한성대학교 컴퓨터공학과
접수일자 2012.2.13, 수정일자 2012.3.20
게재확정일자 2012.4.13

Received: 13 February, 2012, Revised: 20 March, 2012
Accepted: 13 April, 2012
*Corresponding Author: yukim@hansung.ac.kr
Dept. of Computer Engineering, Hansung University, Korea

렵다. 물리적인 특성 상 메인 메모리에 사용되는 커패시터(Capacitor)는 알파입자에 의한 소프트 에러에 강인한 특성을 보이는 반면, 트랜지스터를 사용하는 온칩 캐쉬 메모리 (On-chip Cache Memory)는 상대적으로 더 취약하다. 프로세서의 미세 공정 능력의 발전과 고성능 프로세서에 대한 수요로 인하여 현대의 프로세서는 더 큰 온칩 캐쉬 메모리를 탑재하고 있으며, 이러한 양상은 네트워크 프로세서부터 스마트폰과 같은 고성능 소형 임베디드 디바이스까지 동일하게 나타나고 있다.[2]

일반적으로 임베디드 시스템은 휴대성을 위하여 공간적 제약사항과 배터리 효율성을 위하여 전력 소모에 대한 강한 제약사항이 요구되고 있다. 기존에 제안된 고신뢰성 캐쉬 메모리를 위한 소프트 에러 검출 및 복원 기법들은 대체로 추가적인 공간을 필요로 하거나, 높은 전력 소모를 요구한다[3]. 이와 같은 기법들은 소형 임베디드 시스템 환경에서는 적용하기가 어려우며, 높은 효율성을 위한 추가적 고려가 필요하다.

소프트 에러를 극복하기 위하여 기존에 제안된 방법들 중 비교적 낮은 공간적 요구사항과 성능 저하를 보이는 기법에는 In-Cache Replication (ICR)[4]이 있다. 이 기법은 데드 블록 예측(Dead Block Prediction) 기법을 사용하여 더 이상 사용하지 않을 것이라고 예측되는 다른 캐쉬 메모리 블록에 현재 활발히 사용하고 있는 데이터의 복사본을 저장하는 기법이다. 하지만 이 기법은 레벨 1 캐쉬에서만 그 성능과 오버헤드가 검증되었다. Last Level Cache(LLC)는 레벨 1 캐쉬와 비교하여 접근 빈도, 읽기 및 쓰기의 양상이 서로 상이하므로, 레벨 1 캐쉬에서 잘 동작하는 ICR도 LLC에서는 그 효율성이 떨어진다.

본 논문에서는 LLC에 대한 ICR의 성능 하락과 문제점을 분석하고, 이에 대한 보완 기법으로 작은 값 복사 (Narrow Value Duplication)[5] 기법을 추가적으로 적용하여 저비용으로 높은 신뢰성의 캐쉬 메모리를 구현하였다. 작은 값 복사 기법은 캐쉬 블록에 저장된 하나의 워드 값이 전체 비트 중 하위 절반 비트만으로 표현 가능할 때, 그 값을 상위 절반 비트로 복제하여 함께 저장하는 기법이다. LLC 에서 ICR만 적용한 경우 소프트 에러 보호율은 67.4%이며 이는 레벨 1 캐쉬에서 적용했을 때의 보호율인 92.65% 보다 25.25% 더 낮은 보호율이다. ICR과 작은 값 복사 기법을 함께 적용한 경우에는 95.4%로 만족할 만한 보호율을 보였으며, 시스템 성능은 단지 0.26%만 하락하였다. 또한 단지 2.96%의 추가적인 에너지만을 소모하였다.

본 논문의 구성은 다음과 같다. 제 2장에서는 소프트 에러를 극복하기 위한 기존의 기법들을 간략히 소개한다, 제 3장에서는 제안한 기법에 대하여 상세히 설명하며, 제 4장에서는 제안한 기법의 실험 및 결과에 대하여 기술하였다. 제 5장에서는 결론을 맺는다.

II. 배 경

1. 소프트 에러

소프트 에러는 캐쉬 내 저장된 비트 값이 불안정한 공급 전원, 인터커넥션 노이즈, 타이밍 에러 및 외부로부터의 알파입자 유입으로 인하여 0에서 1 또는 1에서 0으로 역전되는 현상을 말한다. 이러한 에러는 일시적으로 발생하며, 사용자나 시스템으로부터 관측이 불가능하기 때문에 잠재적 위험성을 가진다.

레벨 1 캐쉬는 성능에 민감하므로 패리티 비트를 사용하여 에러를 검출하고, LLC에서는 SECDED(Single Error Correction Double Error Detection)와 같은 에러 검출 코드(Error Correction Code: ECC) 코드를 사용하여 에러를 검출한다[5]. 패리티 비트는 단일 워드 또는 캐쉬 블록에 대하여 하나의 비트 에러만을 검출할 수 있으며, SECDED는 두 개의 비트 에러를 검출할 수 있고 하나의 비트 에러를 복구 할 수 있다. 만약 에러가 검출된 캐쉬 블록의 값이 아직 수정되지 않은 clean 상태라면 단지 다음 레벨의 메모리에서 다시 값을 가져오는 것만으로 복원할 수 있다. 하지만 이미 수정된 dirty 상태라면 에러로부터 복구될 수 없다. 멀티 코어 프로세서의 경우 레벨 1 캐쉬를 파워 절감의 이유로 인하여 Write through 정책으로 설정하는 것이 일반적이다[6]. 이 경우 레벨 1 캐쉬 내의 모드 블록은 clean 상태이므로 소프트 에러로부터 복구가 가능하다.

2. In-Cache Replication

ICR은 데드 블록 예측(Dead Block Prediction) 기법을 사용하여 더 이상 사용하지 않는 캐쉬 블록에 현재 활발히 사용되는 캐쉬 블록을 복제하여 에러 발생 시 복원하는 기법이다. 그림 1은 ICR의 예를 나타내고 있다. 데이터 패스로부터 캐쉬에 쓰기가 수행되거나 다음 레벨

메모리로부터 새로운 캐쉬 값이 적재될 때, 현재 위치로부터 전체 way 개수의 절반만큼 떨어진 캐쉬 라인에 대하여 더 이상 사용되지 않는 블록인지 검사한다.

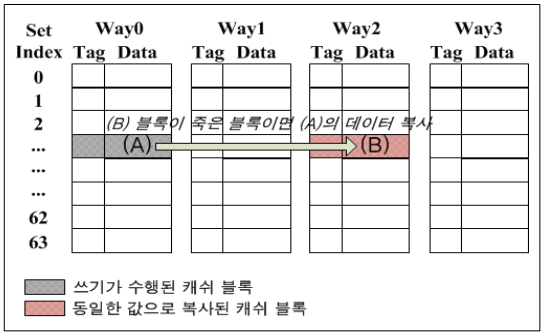

그림 1. In-Cache Replication의 예
Fig. 1. Example of the In-Cache Replication

예를 들어 0부터 3까지의 4-Way 연관을 가지는 캐쉬 메모리가 존재할 때, 첫 번째 Way의 캐쉬 라인에 쓰기가 수행될 경우 동일 Set의 세 번째 Way에 위치하는 캐쉬 라인에 대하여 데드 블록 여부를 검사한다. Zhang의 연구[4]에서는 마지막 캐쉬 라인 접근으로부터 1000 사이클 안에 다시 접근하지 않으면 해당 블록을 더 이상 쓰지 않는다고 판단하고 복제를 수행한다. 데드 블록이 아닐 경우에는 복제하지 않는다.

Ⅲ. 제 안 기 법

1. LLC 에서의 ICR 성능 예측

비록 레벨 1 캐쉬에서 ICR이 잘 동작한다고 해서 LLC에서도 잘 동작한다는 것을 보장할 수 없다. 이는 두 캐쉬 메모리가 서로 접근 빈도, 읽기/쓰기에 대한 양상이 다르기 때문이다. 데이터 패스에 가장 가까이 있는 레벨 1 캐쉬는 데이터를 자주 읽고 쓰므로 빈번하게 접근된다. 하지만 LLC의 경우 데이터가 이전 레벨의 캐쉬 메모리에서 충분히 사용된 후 캐쉬 라인이 교체될 때만 LLC에 쓰기를 수행하므로 최초 읽기 후 거의 대부분의 시간동안 다시 접근되지 않는다. 표 1은 레벨 1 캐쉬와 레벨 2 캐쉬에 대한 접근 빈도를 나타낸다. 레벨 1 캐쉬에서의 평균 접근 횟수는 수십만 번인데 비해, LLC에서의 평균 접근 횟수는 수백 번으로 LLC의 접근 자체가 현저히 낮음을 알 수 있다. ICR은 특정 데이터가 캐쉬 내에 적재된 후 일정시간 동안 활발히 사용된 후 더 이상 사용되지 않는다는 특성을 이용한 것이다.

표 1. 레벨 1 캐쉬와 LLC에서의 접근 빈도
Table 1. Access Frequencies of Level 1 and Last Level Caches

Benchmarks	L1 캐쉬 접근 빈도	L2 캐쉬 접근 빈도
vpr	3979	4
mcf	714706	3
parser	132	3
eon	13808	4080
vortex	53326	26
bzip2	15549	1
twolf	57	1
mgrid	38	1
applu	51	6
mesa	146	9
galgel	56	4
equake	570329	16
fma3d	366702	5135
apsi	253	24
평균값	124223	665

LLC에서는 데이터에 한번 접근된 후 재사용 되는 사이의 시간이 상대적으로 훨씬 길며, 경우에 따라서는 다시 사용되기도 전에 다른 캐쉬 블록으로 교체되는 경우도 빈번히 발생한다. 이는 ICR이 LLC에서는 효율적이지 않음을 의미하며 따라서 저비용으로 LLC에서의 ICR기법이 가지는 비효율성을 극복할 수 있는 다른 기법이 필요함을 의미한다.

2. 작은 값 복사

작은 값 복사(Narrow Value Duplication)[5] 기법은 캐쉬에 저장되는 데이터 워드의 값이 작은 값일 때 이를 복제하여 저장하는 기법이다. 작은 값은 워드 값이란 전체 워드 비트의 하위 절반만으로 표현이 가능한 값을 말한다. 그림 2는 하나의 워드가 16 비트인 작은 값 복제 기법을 나타낸다. 예를 들어, 하위 절반 비트만으로 전체 값을 나타낼 수 있는 작은 값이라면 상위 절반 비트는 전부 0이 된다. 이 경우 하위 절반 비트를 상위 절반 비트로 복제한 후 추가로 장착한 비트에 표시를 해둔다면 동일한 워드 공간에 원본과 복사본을 모두 저장할 수 있다. 이

값은 워드 당 1비트씩 필요한 것을 제외하면 추가적인 공간적 요구사항이 없으며, 성능이나 에너지 오버헤드도 거의 유발하지 않는다. 하지만 작은 값에만 사용할 수 있기 때문에 LLC에서는 그 효율이 떨어질 수 있는데, 이는 많은 프로세서들이 LLC를 명령어와 데이터를 동시에 적재하는 통합 캐쉬(Unified Cache) 구조를 채택하고 있기 때문이다[7]. 명령어의 경우 작은 값이 거의 존재하지 않으므로 이 기법은 사용할 수 없다. 이 기법과 ICR은 각각 독립적으로 보완하여 사용할 수 있으므로 모두 사용하였을 경우 소프트 에러에 대한 보호를 높일 수 있다.

그림 2. 작은 값 복사
Fig. 2. Narrow Value Duplication

3. 동작 및 소프트 에러 처리

ICR은 예측 기반의 기법이기 때문에 예측이 실패했을 경우 성능과 에너지 측면에서 손실이 발생할 수 있다. 만약 데이터를 복사하려고 하는 위치의 블록을 데드 블록이라고 판단하고 교체하려고 시도할 때, 데드 블록이 dirty 상태이면 해당 블록의 데이터를 다음 레벨의 메모리로 보내야 한다. LLC의 경우 다음 레벨 메모리는 메인 메모리이며 접근 시간도 길고 에너지 소모도 심하여 전체적인 처리 비용이 캐쉬 메모리에 비해 높다. 이러한 상황에서 만약 예측이 빗나간다면 메인 메모리로부터 LLC로 다시 데이터를 적재한 후 사용하여야 한다. 반면 작은 값 복제 기법은 해당 데이터가 작은 값인지 아닌지에 대한 판별 비용만 존재하고 이후 추가적인 오버헤드를 발생시키지 않는다. 따라서 LLC에 대하여 두 기법 중 작은 값 복제 기법을 먼저 적용한 후 작은 값이 아닌 데이터에 대해서 ICR을 수행하는 것이 저 비용으로 소프트 에러를 방지할 수 있는 방법이다. 데이터에 수정에 이루어지는 쓰기 명령 시에만 복사를 시도하며 수정이 이루어지지 않은 clean 블록은 에러 발생 시 다음 레벨 메모리에서 다시 가져오는 것으로 복구할 수 있다.

패리티 코드 또는 ECC 코드 역시 트랜지스터에 저장되는 비트의 일부로써 소프트 에러 또한 발생할 수 있다. clean 데이터 블록 또는 이 블록의 ECC 코드에 소프트 에러가 발생할 경우 위치 여부에 상관없이 다음 레벨에서 데이터를 새로 가져오면 된다. 작은 값 복제 기법으로 저장된 경우 상위 절반 워드와 하위 절반 워드에 저장된 값은 서로 같아야 한다. 이 값들이 서로 다르면 소프트 에러가 발생한 것이며, 이 경우 각각의 절반 크기의 워드로 저장된 값들을 원래 길이로 복원시킨 후 ECC 코드로 비교하여 에러가 없는 쪽을 선택하면 된다. 만약 두 값이 같지만 에러라고 검출된 경우 ECC 코드에 에러가 발생한 것이므로 ECC 코드를 새로 생성하여 저장한다. ICR의 경우 역시 에러 발생 시 복제한 위치의 값과 비교하여 에러가 없는 데이터를 선택하여 사용하면 된다.

Ⅳ. 실험 및 결과

1. 실험 환경

실험은 Simplescalar 3.0e[8] 시뮬레이터와 SPEC CPU2000[9] 벤치마크를 사용하여 측정하였다. 10억 사이클을 Fast-forward 시킨 후 실행하였으며, 프로세서 구성은 현재 활발히 사용되고 있는 임베디드 프로세서인 ARM Cortex-A8을 기본으로 구성하였다. 표 2는 본 논문의 실험에서 설정한 프로세서 구성을 보여준다.

표 2. 시뮬레이터 구성
Table 2. Simulator Configuration

Processor Core	
Datapath Width	2
Functional Units	2 IALU, 1 IMULT/DIV 1 FPALU, 1 FPMULT/DIV
Branch Predictor	
Predictor	2 Level Globla, 8-entry RAS
BTB	512-entry, 2-way
Memory	
L1 D-Cache	16KB, 4 ways 64B blocks, 2 cycles
L2 Cache	256KB, 8 ways 64B blocks, 8 cycles
Memory	80 cycles

2. 에러 보호율

캐쉬 내에 저장된 데이터에 소프트 에러가 발생하였을 경우 실제 시스템에 영향을 미치는 순간은 데이터 패스로부터 읽히거나 또는 캐쉬 블록이 교체되는 순간이다. 이와 같은 상황에서 블록이 clean 상태이거나, 작은 값 또는 ICR에 의해 복제된 경우라면 소프트 에러가 발생하여도 복구할 수 있다. 그림 3은 제안된 기법에 대한 에러 보호율을 나타낸다. 보호율은 LLC로부터 데이터가 읽혀지거나 다음 레벨의 메모리로 writeback 될 때, 데이터가 보호되는 횟수를 세어 측정하였다. ICR-NVP 뒤의 숫자는 Decay Window이며 이는 캐쉬 블록이 데드 블록으로 변할 때 까지 사이클 수를 나타낸다. 예를 들어, ICR-NVP-1000은 특정 블록의 마지막 접근으로부터 1000 사이클이 지나면 데드 블록으로 판단한다는 의미이다. NVP는 Narrow Value Protection 으로 작은 값 복제를 적용한 경우를 의미한다. 이 경우 Decay Window가 각각 ICR-NVP-5000, ICR-NVP-10000일 때 보호율은 각각 95.12%, 94.94%로 나타났다.

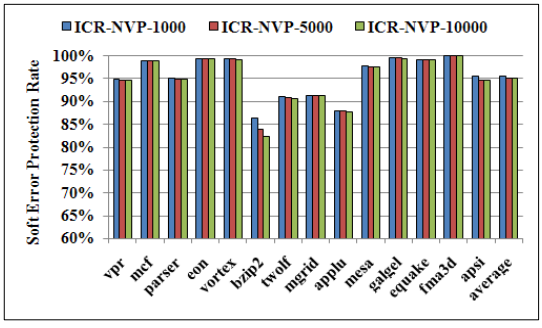

그림 3. Decay Window별 소프트 에러 보호율
Fig. 3. Protection Rate per each Decay Window

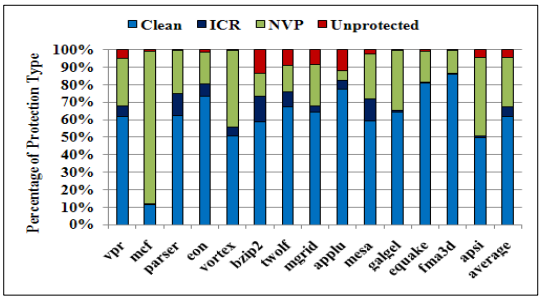

그림 4. ICR-NVP-1000의 타입 별 에러 보호율
Fig. 4. Percentage of Protection Type

그림 4는 ICR-NVP-1000의 보호율을 상세히 분류한 것이다. 평균 보호율은 95.4%이며 그중 블록이 clean 상태이기 때문에 보호받는 비율은 61.93%, 작은 값 복제 기법으로 보호받는 비율은 28.25%, ICR로 보호받는 비율은 5.47%이다. 레벨 1 캐쉬에서 ICR로 인한 보호율이 35.77%인 것과 비교하면 LLC에서는 ICR 기법이 효과적이지 못함을 알 수 있다.

3. 성능 평가

그림 5는 ICR과 작은 값 복제 기법을 적용하였을 때의 성능 저하 정도를 나타낸다. 작은 값 복제 기법은 추가적인 성능 저하는 발생하지 않으며 ICR의 경우는 3.3절에서 설명한 바와 같이 데드 블록에 대한 예측이 실패할 경우 발생한다. 성능 측면에서는 거의 저하가 발생하지 않았는데, ICR 복제 중 추가적인 쓰기 명령은 데이터 패스의 성능에 직접적인 영향을 미치지 않으므로 복제에 대한 성능적인 비용은 거의 들지 않는다. 전체 성능은 0.26%만 저하되었다.

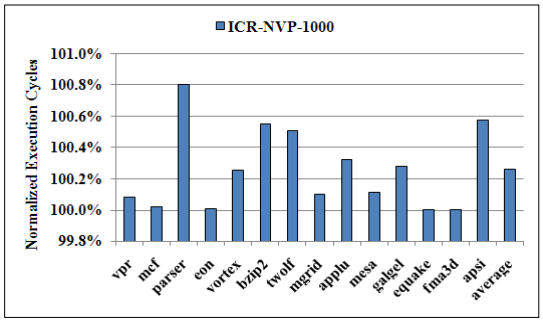

그림 5. 기존 대비 ICR-NVP-1000의 실행 사이클
Fig. 5. Normalized Execution Cycles

4. 에너지 소모

그림 6은 본 연구에서 제안된 기법에 대한 LLC 내의 추가적인 에너지 소모율을 나타낸다. 캐쉬 메모리의 에너지 소모 값은 CACTI 6.5[10] 시뮬레이터로 추출하였으며, 메모리 공정 크기는 45 나노미터로 설정하였다. ICR의 경우 이전 레벨의 메모리로부터 쓰기 명령이 수행될 경우 추가적인 에너지가 소모된다. 복제될 위치의 캐쉬 블록이 데드 블록 여부를 한 후, 데드 블록이면 캐쉬 블록 교체를 수행한 후 교체한 위치에 복제될 값을 쓴다. 만약 복제될 위치의 값이 dirty 값이라면 writeback까지 수행하여야 한다. 이로 인한 동적 에너지 소모는 기존 대비 0.13% 더 늘어났으며, 실행시간 증가로 인한 정적 에

너지는 2.82% 증가하여 전체 2.96%의 추가적인 에너지가 소모되었다.

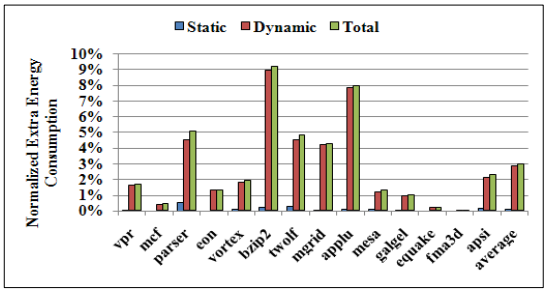

그림 6. ICR-NVP-1000 추가 에너지 소모비율
Fig. 6. Normalized Extra Energy Consumption

5. 공간 오버헤드

본 연구에서는 LLC를 256KB로 구성하였으며, 각 캐쉬 블록은 64바이트로 구성하였고 하나의 워드는 64비트로 설정하였다. ICR을 사용하기 위해서 워드 당 2비트의 saturation counter가 필요하며 복사본이 있는지 표시하기 위하여 1 bit가 추가로 필요하다. 작은 값 복제 기법을 사용하기 위해서는 워드 당 1비트가 필요하다. 두 기법을 모두 사용하면 워드 당 4비트가 필요하며 이는 전체 LLC 크기에서 6.25%만의 공간적 오버헤드를 가짐을 의미한다. 두 기법을 사용하기 위한 제어 로직은 상대적으로 거의 공간을 차지하지 않으므로 공간적 오버헤드는 없다고 가정하였다.

V. 결 론

본 연구에서는 Last Level Cache에서 저비용의 에너지 및 공간적 오버헤드를 가지는 소프트 에러 극복 기법에 대하여 제안하고 분석하였다.

In-Cache Replication은 캐쉬 메모리 내의 데드 블록 예측 기법을 사용하여 더 이상 사용하지 않는 메모리 블록에 대해 현재 사용하고 있는 캐쉬 블록의 데이터를 여분으로 저장하는 기법이다. 하지만 ICR 기법은 레벨 1 캐쉬에서는 만족할 만한 성능을 보였으나, LLC에서는 동일 데이터의 재사용에 대한 긴 대기시간으로 인하여 효율성이 떨어진다.

이에 본 연구에서는 작은 값 복제 기법을 함께 적용하여 이와 같은 비효율성을 극복한 기법을 제안하였다. ICR과 작은 값 복제 기법은 각각 LLC에서의 성능이 떨어지지만, 중첩하여 사용하면 ICR만 사용했을 때 보다 27.9% 더 높은 보호율인 95.4%의 호율을 보인다. 추가적인 에너지 소모는 2.82%이며 기존의 캐쉬 메모리에 비하여 단지 6.25%만의 추가적인 공간을 요구한다.

참 고 문 헌

[1] H. Asadi, V. Sridharan, M. B. Tahoori, and D. Kaeli, “Vulnerability analysis of L2 cache elements to single event upsets,” in *Proc. Des., Autom., Test Eur.*, Mar. 2006, pp. 1 - 6.

[2] ARM Cortex A8 processor, “http://www.arm.com/products/processors/cortex-a/cortex-a8.php”

[3] K. Bhattacharya, N. Ranganathan, and S. Kim, "A Framework for Correction of Multi-Bit Soft Errors in L2 Caches Based on Redundancy," IEEE Trans. VLSI., Vol. 17, issue. 2, pp. 196-206, Feb. 2009.

[4] W. Zhang, S. Gurumurthi, M. Kandemir, and A. Sivasubramaniam, ICR: In-cache replication for enhancing data cache reliability,” in *Proc. Int. Conf. Depend. Syst. Netw.*, 2003, pp. 291 - 300.

[5] O. Ergin et al, “Exploiting narrow values for soft error tolerance,” IEEE Computer Architecture Letters, 2006.

[6] R. Phelan, “Addressing soft errors in ARM core-based soc,” ARM White Paper, ARM Ltd., Dec 2003.

[7] ARM Cortex A9 processor, “http://www.arm.com/products/processors/cortex-a/cortex-a9.php”

[8] D. Burger and T. M. Austin. The SimpleScalar Tool Set, Version 2.0. Computer Architecture News, pages 13 - 25, June 1997.

[9] The Standard Performance Evaluation Corporation. Spec CPU2000 suite. http://www.specbench.org/osg/cpu2000/.

[10] N. Muralimanohar, R. Balasubramonian, and N.P. Jouppi, “CACTI 6.5,” HP Laboratories, Technical Report, 2009.

※ 본 연구는 한성대학교 교내연구비 지원 과제임.

저자 소개

김 영 웅(정회원)

• 1993년 KAIST 전산학과 박사
• 1984 ~ 1997년 KT 통신망연구소
• 1997년 ~ 현재 한성대학교 컴퓨터공학과 교수

<주관심분야 : 소프트웨어 신뢰도, 소프트웨어 설계, 데이터 모델링>