논문 2014-09-04

저전력 집합연관 캐시를 위한 효과적인 알고리즘

(Effective Algorithm for the Low-Power Set-Associative Cache Memory)

정 보 성, 이 정 훈*
(Bo-Sung Jung, Jung-Hoon Lee)

Abstract : In this paper, we proposed a partial-way set associative cache memory with an effective memory access time and low energy consumption. In the proposed set-associative cache memory, it is allowed to access only a 2-ways among 4-way at a time. Choosing ways to be accessed is made dynamically via the least significant two bits of the tag. The chosen 2 ways are sequentially accessed by the way selection bits that indicate the most recently referred way. Therefore, each entry in the way has an additional bit, that is, the way selection bit. In addition, instead of the 4-way LRU or FIFO algorithm, we can utilize a simple 2-way replacement policy. Simulation results show that the energy*delay product can be reduced by about 78%, 14%, 39%, and 15% compared with a 4-way set associative cache, a sequential-way cache, a way-tracking cache, and a way cache respectively.

Keywords : Data cache, Dynamic access time, Low-power consumption.

Ⅰ. 서 론

오늘날, 대용량 및 다양한 응용프로그램의 등장으로 데이터 처리를 위해 컴퓨팅 시스템은 더욱 빠른 처리 속도를 위해 점점 고성능의 프로세스를 요구하고 있다. 고성능 프로세서를 위한 가장 간단한 방법은 높은 클럭 주파수를 가지는 것이다. 하지만, 프로세서의 높은 주파수는 오히려 높은 에너지 소비를 야기 시킬 뿐 아니라 높은 발열현상으로 프로세서의 수명을 단축시키는 원인이 된다. 프로세서의 공정기술 발달은 이러한 에너지 소비 및 발열을 줄이는 한 방법으로 현재 22nm의 공정기술을 사용하고 있다. 특히, 인텔은 트라이게이트 트랜지스터 기술을 바탕으로 22nm공정뿐 아니라 차후 14nm, 5 nm의 공정을 계획하고 있다[1].

또 다른 대표적인 방법은 시스템의 성능향상 및 에너지 소비에 효과적인 새로운 구조 캐시의 접근이다. 이미 알고 있듯이, 캐시는 시스템의 성능향상 및 저전력을 위한 중요한 메커니즘으로 자리 잡고 있다. 캐시 성능향상의 대표적인 방법은 구조적인 모델링으로 접근실패율의 가장 큰 원인인 충돌 접근실패율을 줄이는 것이다. 성능향상을 위한 효과적인 구조인 N-웨이 집합연관 캐시는 이러한 충돌 접근실패율을 줄일 수 있다. 하지만, N-웨이 집합연관 캐시는 높은 접근성공률을 가지는 반면, 그에 따른 높은 에너지 소비와 긴 접근 시간을 가진다.

이러한 기존의 웨이 집합연관 캐시의 에너지 소비 및 성능향상을 위해 많은 연구가 이루어졌다. 가장 기본적인 N-웨이 집합연관 캐시의 성능향상 방법은 웨이를 증가 시키는 것이다. 예로, 임베디드 시스템의 모토로라 MPC7450[2]과 고성능을 위한 인텔의 core i7[3]의 경우 8-웨이 집합연관 캐시를 사용하고 있다. 하지만, 앞서 언급한 것과 같이 높은 웨이의 사용은 빠른 클록 주파수를 가지는 프로세서에서 높은 에너지 소비와 발열 그리고 추가적인 접근 시간이 요구된다.

웨이 예측 알고리즘은 이러한 N-웨이 집합연관 캐시의 에너지 소비 및 빠른 접근시간을 위해 가장 활발하게 연구되어진 부분이다. 웨이 예측 알고리즘은 올바른 예측시 1/N의 에너지 소비와 직접사상

*Corresponding Author (leejh@gnu.ac.kr)
Received: 24 July 2013, Revised: 24 Sep. 2013, 25 Oct. 2013, Accepted 5 Nov. 2013.
B.S. Jung, J.H. Lee: Gyeongsang National University

ISSN : 1975-5066
http://dx.doi.org/10.14372/IEMEK.2014.9.1.25

캐시만큼의 빠른 접근시간을 보장할 수 있다. 하지만, 접근실패시, 추가적인 에너지 소비와 접근시간이 필요하다. 비록, 웨이 예측 알고리즘이 다양한 방법을 제시해주지만, 많은 용량의 추가적인 하드웨어나 복잡한 알고리즘이 가졌다. 하지만, 최근 연구에서는 간단한 알고리즘 및 추가적인 하드웨어를 줄이면서, 웨이 예측 알고리즘이 연구되어지고 있다[4-8].

본 논문에서는 각 웨이 블록에 추가적인 1비트를 이용한 간단한 알고리즘으로 낮은 에너지 소비 및 빠른 접근시간을 가지는 4-웨이 집합연관 캐시를 제안한다. 제안된 캐시는 단지 4-웨이 집합연관 캐시 중 하위 2-웨이를 태그 하위 2비트를 이용하여 선택되어진다. 그리고 선택되어진 하위 2-웨이는 최근 접근성공이 발생된 웨이부터 순차적으로 접근이 이루어진다. 추가된 1비트는 하위 2웨이의 최근 접근성공 웨이를 나타내게 된다.

시뮬레이션 결과, 제안된 캐시 알고리즘은 에너지*지연시간에 대해 기존 4-웨이 집합연관 캐시에 비해 약 78%의 성능향상을 보였다. 그리고 웨이 캐시(Way Cache), 웨이 추적 캐시(Way-Tracking Cache) 및 순차적 웨이 접근 캐시(Sequential Way Access Cache)에 비해 각각 15%, 39% 그리고 14%의 성능향상을 이루었다.

이 논문의 구성은 다음과 같다. II장에서는 관련연구에 대해 설명한다. III장에서는 제안된 캐시의 제안동기 및 구조와 알고리즘 및 동작에 대해 설명한다. IV장에서는 시뮬레이션 결과 및 평가 그리고 V장에서 결론을 맺는다.

II. 관련연구

오늘날, 빠른 CPU 클럭 속도에 따라 저전력 및 고성능 내장형 캐시에 대한 다양한 연구가 이루어지고 있다. 기본적으로 오늘날 프로세서는 높은 접근성공률을 위해 웨이 집합연관 캐시를 사용하고 있다.

연구[6, 7, 9]는 웨이 집합연관 캐시로 구성된 큰 용량의 L2 캐시의 성능 및 에너지 소비에 대한 연구이다. 하지만, L2 캐시의 접근은 5%미만으로 L1 캐시의 성능향상 및 에너지 소비를 줄이는 것이 더욱 효과적이다.

Way-tracking Cache(WTC)[8]는 WTT(Way Tracking Table)를 이용하여 집합연관 캐시를 선택적으로 접근하여 성능과 에너지 소비에 대한 향상을 이루었다. WTT는 태그(Tag) 부분의 하위 비트 정보를 가지며, 이 정보는 웨이 집합연관 캐시의 접근 전 요청된 주소와 비교하여 선택되어진 웨이만 접근하게 된다. 비록, WTC가 기존 N-웨이 집합연관 캐시에 비해 부분적인 웨이 접근으로 에너지 소비에 효과적이지만, 하나의 웨이 접근을 제외하고는 기존 N-웨이 집합연관 캐시와 동일한 접근 시간을 가진다.

Way-Cache(WC)[10]의 연구는 CAM(Contents Address Memory)으로 구성된 WC 메모리에 최근 접근된 집합연관 캐시의 정보를 저장하므로 웨이 캐시 접근 전 각 웨이의 접근 여부를 판단하게 된다. 만약 WC에 웨이 접근 정보가 있다면, 웨이들 중 하나를 선택하여 빠른 접근 시간과 낮은 에너지 소비를 줄인 연구이다. 더욱이 WC는 LSQ (Load/Store Queue)에 대한 접근이 동시에 이루어지므로 기존 4-웨이 집합연관 캐시 중 한 웨이만의 접근성공률을 높였다. 하지만, 웨이 집합연관 캐시 접근을 위해 CAM의 구동은 실제 많은 에너지 소비를 초래한다. 실제, WC는 32~64개의 블록을 제안하였다. 이는 32 Kbyte의 4-웨이 집합연관 캐시와 비슷한 에너지 소비를 보인다.

Sequential Way-Access Cache(SWAC)[11]는 기존 2-웨이 집합연관 캐시에서 가장 최근에 접근 웨이를 순차적으로 접근 하여 에너지 소비 및 평균 메모리 접근 시간을 향상 시켰다. 비록 SWAC가 간단한 알고리즘으로 에너지 소비 및 메모리 접근 시간의 성능을 향상 시켰지만, 여전히 SWAC의 접근 실패율은 기존 2-웨이 집합연관 캐시의 실패율을 가지며, 이는 실제 2-웨이 집합연관 캐시에 한정된 구조라고 할 수 있다.

본 논문에서 제안된 아이디어를 위한 선행연구로 캐시에 대한 접근성공 패턴 및 접근성공에 대한 웨이 비율등 다양한 시뮬레이션을 수행하였다. 선행연구는 범용 컴퓨팅 시스템 및 임베디드 시스템의 고성능 프로세서에서 채택하고 있는 4-웨이 집합연관 캐시이며, 32byte의 블록 크기를 가지는 32 Kbyte 캐시에 대해 시뮬레이션을 수행 하였다.

III. 본 론

1. 제안된 캐시의 방법 및 구조

본 논문의 주목적은 빠른 메모리 접근 시간 및 저전력을 가지는 효과적인 웨이 집합연관 캐시를 위한 새로운 알고리즘 설계 및 구현에 그 목적으로

하고 있다. 제안된 캐시의 기본 아이디어는 크게 2 부분으로 나누어진다.

① 최근 참조된 블록은 가까운 시간에 다시 참조될 가능성이 높다는 시간지역성 특성을 이용하는 것이다.
② 순차적인 접근은 특정 웨이를 접근 하는 높은 비율의 선택비트가 존재한다고 가정하였다.

만약 가장 최근에 접근성공이 발생한 웨이 블록이 다시 접근성공 율이 높다라면, SWAC와 같이 간단한 웨이 예측알고리즘으로 좋은 성능향상을 가질 수 있다. 선행 연구결과 가장 최근에 접근성공이 발생한 웨이 블록의 다시 접근성공이 발생할 경우가 평균 70%를 보였다. 따라서 가장 최근에 접근성공이 발생한 웨이 블록을 우선 참조하는 것은 성능향상에 효과적이다. 하지만, 많은 웨이의 순차적인 접근은 오히려 성능 저하의 원인이 된다.

선행연구 시뮬레이션 결과 태그 하위 2비트 채택시 2-웨이 선택에 가장 효과적인결과를 보였다. 그림 1은 미디어밴치(MediabenchII)에서 접근실패시 동일한 태그 하위 2비트를 가지는 웨이의 비율을 나타낸 그림이다. 그림 1에서 보듯이, 태그 하위 2비트를 웨이 선택 비트로 이용할 경우 전체 접근의 약 90%를 차지한다.

따라서 제안된 캐시는 기존 4-웨이 집합연관 캐시와 동일한 구조를 가진다. 하지만 기존 4-웨이 집합연관 캐시와 달리, 제안된 캐시는 하위 2비트를 이용하여 단지 2-웨이 접근만이 이루어진다. 예로, 태그 최하위 2비트가 '00'이면 Way0와 Way1에 접근하며, '01'이면 Way1과 Way2, '10'이면 Way2와 Way3 그리고 '11'이면 Way3과 Way0에만 접근이 일어난다.

위의 예와같이, 제안된 캐시는 4-웨이 집합연관 캐시중 단지 하위 2-웨이만이 접근이 이루어진다. 따라서 제안된 캐시의 각 웨이는 서로 다른 태그 하위 2비트를 가지는 공유웨이가 된다. 즉 Way0는 하위 태그 2비트 '00'과 '11'에 대한 공유 웨이며, Way1은 '00'과 '01'의 공유 웨이가 된다. 그리고 Way2는 '01'과 '10'의 공유 웨이, Way3은 태그 하위 2비트 '10'과 '11'의 공유 웨이가 된다.

선택되어진 하위 2-웨이 접근은 최근 접근성공이 발생한 웨이부터 순차적으로 접근이 이루어진다. 따라서 제안된 캐시는 선택되어진 하위 2-웨이중 최근 접근성공이 발생한 웨이 선택을 위한 추가적

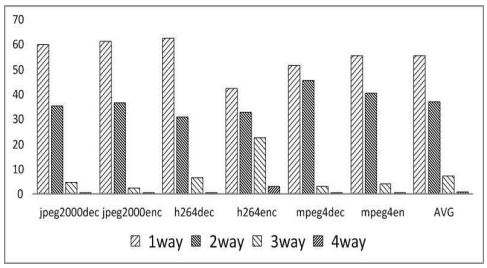

그림 1. 4-웨이 집합연관 캐시 접근실패시 동일한 태그 하위 2비트의 비율

Fig. 1 The ratio of lower two bits of tag that when 4-way set associative cache occur miss

인 1비트의 웨이 선택비트(S.B)를 가진다. 그리고 이 웨이 선택비트는 오직 하위 2-웨이 접근에 대한 우선순위 정보만을 가진다. 따라서 선택되어진 하위 2-웨이 접근실패시, 데이터 교체정책은 웨이 선택비트에 의해 요청된 데이터가 저장되어진다. 이때 새로운 데이터가 저장된 웨이 블록은 가장 최근에 접근이 발생한 웨이 블록을 표시하기위해 웨이 선택 비트가 갱신된다. 웨이 선택비트의 갱신은 하위 태그 2비트가 가리키는 웨이 선택비트가 갱신되어진다. 예로 하위 태그 2비트가 '00'이고, 만약 Way1에서 접근성공이 발생하면 하위 태그 2비트가 '00'이기 때문에 Way0의 웨이 선택비트가 갱신되며, Way1이 최근 접근성공 웨이를 가리키는 Way0의 선택비트가 '1'로 갱신된다. 만약 Way0에서 접근성공이 발생하면, Way0의 웨이 선택비트가 Way0를 가리키는 선택비트가 '0'으로 갱신된다.

접근실패시 교체정책은 선택비트에 의해 하위 2-웨이에 대한 동작이지만, 제안된 캐시는 각 웨이가 태그 하위 2비트에 의해 공유웨이 이다. 따라서 추출되는 웨이 교체 블록은 이웃 웨이의 교체 블록에 이동이 가능 여부를 판단해야만 한다. 만약 추출 웨이 블록이 이웃 웨이 블록과 같은 태그 하위 2비트를 가고, 최근 접근성공이 발생한 블록이면, 그 추출 웨이 블록의 데이터는 이웃 웨이 블록으로 저장되어진다. 이동되어지는 웨이 블록은 최근 접근이 발생된 웨이를 나타내기 위해 선택비트 역시 갱신이 된다.

2. 제안된 캐시의 동작

제안된 캐시는 기존 4-웨이 집합연관 캐시에 간단한 알고리즘으로 구현 할 수 있다. 제안된 캐시는 4-웨이 집합연관 캐시중 하위 2-웨이 접근이

이루어지며, 하위 2웨이 접근은 태그 하위 2비트에 의해 결정되어진다. 이때 각 웨이는 하위 2비트에 대한 공유 웨이를 제공한다.

제안된 캐시의 선택 되어진 하위 2-웨이 중에서 가장 최근에 접근이 발생한 웨이부터 순차적으로 접근이 이루어진다. 이때, 웨이의 순차적인 접근은 최근 접근이 발생한 웨이를 나타내는 선택비트에 의해 결정된다. 따라서 2-웨이 중 최근에 참조되어진 웨이 선택을 위한 추가적인 1비트를 가진다.

제안된 캐시의 동작은 다음과 같습니다.

1) 요청된 주소의 태그 하위 2비트를 이용하여 4-웨이 집합연관 캐시 중 하위 2웨이를 선택하게 된다
2) 선택 되어진 하위 2웨이 중 태그 하위 2비트가 가리키는 웨이의 선택비트에 의해 하위 2웨이가 순차적으로 접근이 이루어진다. 선택비트가 '0'이면 태그 하위 2비트가 가리키는 웨이가 우선 접근되며, '1'이면 다른 웨이가 우선적으로 접근이 일어난다.
3) 선택되어진 하위 2웨이중 접근성공이 발생하면, 접근성공이 발생된 웨이 선택비트가 현재 웨이가 최근에 접근성공이 발생한 웨이를 가리키는 '0'으로 갱신이 되며, 접근성공 웨이에 공유 될 수 있는 다른 태그 하위 2비트가 가리키는 웨이 역시 접근성공이 발생한 웨이가 최근 접근되었다는 것을 나타내는 선택비트 '1'의 값으로 갱신된다. 예로, Way1에서 접근성공이 발생하며, Way1의 선택비트는 '0'으로 갱신되며, Way1에 공유되어 질수 있는 하위 태그 '00'을 가리키는 Way0의 선택비트는 '1'로 갱신이 발생합니다.
4) 접근실패가 발생하면, 선택되어진 하위 2웨이 중 태그 하위 2비트가 가리키는 웨이의 선택비트에 의해 LRU동작을 수행된다. 이때, 선택되어진 웨이 교체 블록은 공유 웨이 블록이기 때문에 선택되어진 교체 블록은 이웃 웨이 블록으로 이동 여부를 판단하게 된다. 예로, Way0의 추출 블록의 태그 하위 2비트가 '11'이고 Way3의 선택비트가 '1'이라면, Way0의 추출 블록은 Way3에 저장 된다. 이때, Way3의 추출 블록이 Way2에 저장 될 수 있다면, Way3의 추출 블록은 Way2에 저장 된다. 하지만, Way2의 블록이 Way1에 저장 될 수 있는 조건이라도 현재 접근이 이루어진 하위 2웨이중 가장 최근 접근이 이루어진 블록으로 데이터 이동은 일어나지 않는다. 따라서 제안된 캐시의 데이터 이동은 선택되어진 하위 2-웨이 접근에서 가장 최근에 접근이 발생된 웨이까지 이루어지며, 제안된 캐시에서 데이터 이동은 최대 2번 발생하게 된다.

2.1 접근성공이 발생할 경우.

그림 2(a)는 제안된 캐시의 접근성공을 나타낸 그림이다. 만약 CPU로부터 요청된 주소가 그림 2(a)와 같이 '1100010'이면, 인덱서는 '10', 태그는 '11000'이다. 이때 태그 '11000'의 하위 2비트('00')에 의해 2-웨이 접근은 Way0와 Way1에서 이루어진다. 하위 2-웨이의 순차적인 접근은 태그 최하위 비트가 '00'이므로, 순차적인 접근의 판별은 Way0의 선택비트에 의해 결정된다. Way0의 선택비트가 '0'이므로 Way0가 최근 접근성공이 발생한 웨이로, Way0부터 Way1로 접근이 이루어진다. 따라서 그림 2(a)처럼, 4-웨이 집합연관 캐시에서 Way1에서 접근성공이 발생하며, 이때 순차적인 접근으로 추가적인 접근시간이 발생한다.

그리고 Way1의 선택비트는 현재 웨이가 최근 접근성공이 발생한 웨이 블록을 나타내는 '0'으로 갱신이 일어난다. Way1은 태그 하위 2비트 '00'과 '01'의 공유 웨이로 태그 하위 2비트 '00'에 대한 Way0의 선택비트 Way1을 나타내는 '1'로 갱신된다.

2.2 접근실패가 발생할 경우.

그림 2(b1, b2)는 제안된 캐시 접근실패를 나타낸 그림이다. 제안된 캐시의 접근실패시 공유 웨이를 통한 데이터 이동이 발생 할 수 있다. 그림 2(b1)은 제안된 캐시에서 데이터 이동이 없는 접근실패 결과를 나타내고 있다. 만약 요청된 주소가 '1100011'이라면, 태그 하위 2비트'00'에 의해 제안된 하위 2-웨이 접근은 그림 2(a)에서 Way0와 Way1에서 이루어진다. 태그 하위 2비트가 '00'임으로, 그림 2(a)의 결과 Way0의 선택비트가 '0'으로 Way0가 최근 접근 발생 된 웨에 블록이다. 따라서 요청된 데이터는 Way1에 저장되게 된다. 그리고 Way1의 선택비트가 '1'이기 때문에 Way2가 Way1보다 최근 접근성공이 발생한 웨이로 Way1로부터 데이터 이동이 발생하지 않는다. 요청된 데이터가 Way1에 저장된 후, Way1의 선택비트는 현재 웨이가 최근 접근이 발생한 웨이를 나타내는 '0'으로 갱신되고, Way1에 공유 될 수 있는 태그 하위 2비트 '00'이 가리키는 Way0의 선택비트는 Way1이 최근 접근된 웨이를 가리키는 '1'로 갱신되어진다.

그림 2(b2)는 접근실패시 데이터 이동을 가지는 동작을 나타내고 있다. 요청된 주소가 그림 2(b2)와

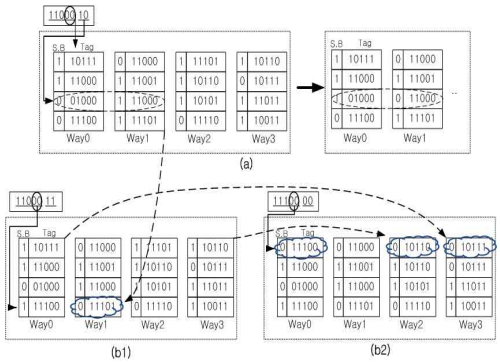

그림 2. 제안된 캐시 접근성공 및 실패 동작
Fig. 2 The Hit and Miss operation of proposed cache memory

표 1. 시스템 환경
Table 1. System Environment

항목	값
CPU 처리 속도	3GHz
캐시 접근 시간	1 ~2 CPU cycle
버스 폭	64bit
메인 메모리 접근 시간	196 CPU cycle

같이 '1110000'일 때, 태그는 '11100'로 하위 2비트 '00'에 의해 하위 2-웨이 접근 역시 Way0와 Way1에서 일어난다. 태그 하위 2비트'00'에 의해 Way0 선택비트에 의해 교체 웨이를 선택하게 된다. 따라서 현재 상태(그림 2(b1))에서 Way1이 최근 접근성공이 발생된 웨이로 요청된 주소 및 데이터는 Way0에 저장하게 된다.

하지만, Way0의 태그의 최하위 2비트는 '11'로 Way3에 저장 될 수 있다. 또한 Way3의 선택비트가 '1'로 Way0이 Way3보다 최근 접근성공이 발생한 웨이로 Way0는 Way3으로 데이트 이동이 발생한다. 그리고 Way3의 태그 하위 2비트 역시 '10'으로 Way2에 이동이 가능하다. Way2의 선택비트 역시 '1'로, Way3이 Way2보다 최근 접근성공이 발생된 웨이 블록으로 Way3의 데이터는 Way2에 이동되어진다. 반면, Way2의 태그 하위 2비트가 '01'로 Way2의 데이터가 Way1로 이동 가능하지만, Way1의 선택비트가 '0'으로 Way2보다 최근 접근성공이 발생된 웨이로 데이터 이동은 발생하지 않는다.

시뮬레이션 결과, 제안된 캐시에서 이러한 데이터 이동은 약 13%가 발생하며, 그중 위 예와 같이 2번의 데이터 이동은 단지 2%만 발생하였다. 제안된 캐시는 기존 4-웨이 집합연관 캐시중 태그 하위 2비트를 이용하여 하위 2-웨이 접근만 이루어지며, 선택되어진 하위 2웨이중 태그 하위 2비트가 가리키는 선택비트 1비트에 의해 최근 접근성공이 발생된 웨이부터 순차적으로 이루어진다. 더욱이, 제안된 캐시의 교체정책은 태그 하위 2비트가 가리키는 선택비트에 의한 교체웨이와 교체웨이에 공유 되어지는 태그 하위 2비트가 가리키는 웨이의 선택비트 값만을 변경하게 된다. 실제 제안된 선택비트는 전체 웨이들의 우선순위를 나타내지 않고, 단지 하위 2-웨이 접근 및 공유 웨이에 대한 우선순위만을 나타낼 수 있다.

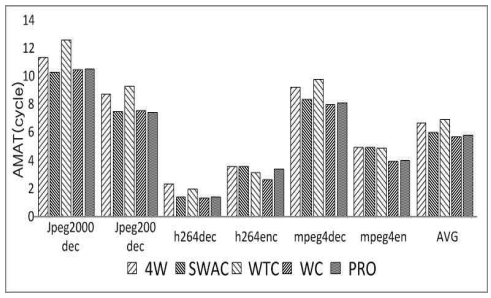

그림 3. 평균메모리 접근 시간(Cycle)
Fig. 3. Average Memory Access Time(Cycle)

Ⅳ. 성능평가

본 논문에서 제안된 캐시의 구조의 성능평가로 평균 메모리 접근 시간(Average Memory Access Time), 에너지 소비 그리고 평균 메모리 접근 시간을 고려한 에너지*지연시간(energy*delay product을) 지표로 하였다. 표 1는 제안된 캐시의 성능 평가를 위한 시스템 환경 변수를 나타낸 표이다.

제안된 캐시의 성능 평가를 위해 본 논문에서는 미디어벤치II[12]를 인텔의 메모리 분석 툴인 pin-tools[13] 시뮬레이터를 수정하여 평가하였다. 본 논문에서 메인 메모리는 다른 요소들은 배제한 CPU 사이클에 대한 전체 메모리의 최소 동작 사이클을 사용하였다. 메모리 계층은 제안된 구조의 성능 비교를 위해 CPU, L1 캐시 그리고 메인 메모리로 가정하였다. 제안된 캐시의 성능 평가를 위해 본 논문에서는 비교 캐시로 최근에 연구되어진 WC, WTC, SWAC 그리고 기존 4-웨이 집합연관 캐시(4W)를 사용하였다. 제안된 캐시와 비교 캐시는 모두 32Kbyte의 동일한 캐시를 가지며, 블록 크기 역시 동일한 32Byte를 가진다. 또한 성능평가는 본 논문에서는 단지 캐시에 대한 평가만 이루어진다.

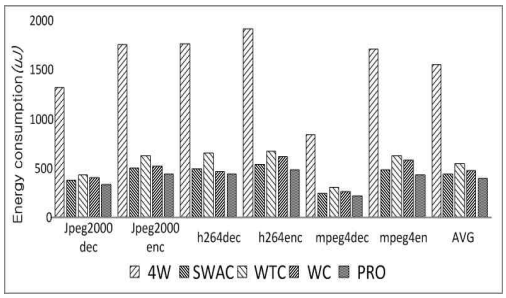

그림 4.에너지 소비(μJ)
Fig. 4. Energy consumption(μJ)

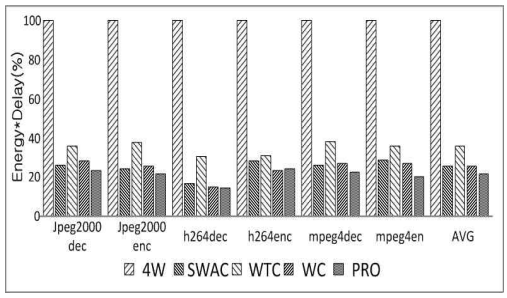

그림 5. 기존 4-웨이 캐시 기준에 대한 에너지*지연시간(%)
Fig. 5. Energy*Delay(%) by various 4-way set associative cache memories

즉, WTC의 경우 추가적인 테이블이 존재하며, WC의 경우 기존 4-웨이 집합연관 캐시의 접근성공이 발생한 웨이 정보를 저장된 추가적인 하드웨어이다.

그림 3은 평균 메모리 접근 시간을 나타낸 그림이다. 제안된 캐시는 WC를 제외한 비교 캐시에 비해 3~16%의 성능향상을 보이고 있다. 제안된 캐시의 경우, 4-웨이 집합연관 캐시중 하위 2-웨이 접근이 이루어지며, 특히 하위 2-웨이 접근중 최근 접근성공이 발생된 웨이를 순차적으로 접근한다. 만약 순차적인 접근에 의해 처음 접근이 성공이 되면, 1cycle 동작이 이루어진다. 시뮬레이션 결과, 제안된 캐시의 1cycle 접근은 전체 접근성공의 평균 97%를 보였다. SWAC와 WC의 경우 역시 약 96%의 1cycle 접근 동작을 보였다. 하지만, SWAC의 경우 높은 접근실패율로 제안된 캐시에 비해 높은 메모리 접근시간을 보이고 있다. WC의 경우, 높은 접근성공률과 1cycle접근으로 좋은 성능을 보이고 있다. 하지만 WC에 웨이 정보가 존재하지 않는다면 기존 4-웨이 집합연관 캐시의 접근성공 시간과 에너지 소비를 가진다. WTC의 경우, 전체 접근성공에 대해 약45%의 1cycle 접근성공률을 가진다.

그림 4는 에너지 소비를 나타낸 그림이다. 제안된 캐시는 높은 1cycle 접근성공과 하위 2웨이의 접근으로 기존 4-웨이 집합연관 캐시에 비해 약 75%의 에너지 소비를 줄였다. SWAC는 제안된 캐시에 비해 약10%의 높은 에너지 소비를 보이며, 이는 높은 접근실패율 뿐 아니라 제안된 캐시에 비해 한 웨이 용량이 크기 때문이다. 그럼에도 불구하고 SWAC의 순차적인 접근은 평균메모리 접근시간에 비해 좋은 에너지 소비를 가진다.

WTC는 기존 4-웨이 집합연관 캐시를 제외하고 가장 높은 에너지 소비를 보인다. 비록 WTC가 기존 4-웨이 집합연관 캐시에 비해 좋은 성능을 보이지만, 다른 비교 캐시에 비해 1cycle 접근비율이 낮을 뿐 아니라 다양한 웨이 접근을 가지므로 다른 비교 캐시에 비해 높은 에너지 소비를 가진다.

WC는 평균 메모리 접근 시간에서 가장 좋은 성능을 보였지만, 에너지 소비에서는 오히려 제안된 캐시에 비해 약17%의 높은 에너지 소비를 가진다. WC에 웨이 정보가 없으면, 기존의 4-웨이 집합연관 캐시의 에너지 소비를 가지기 때문이다. 실제 4-웨이 집합연관 캐시의 에너지 소비는 제안된 캐시에 비해 3~5배정도 높은 에너지 소비를 가진다.

시뮬레이션 결과 제안된 캐시는 비교 캐시(4-웨이 집합연관 캐시 제외)에 비해 평균 11%~29%의 에너지 소비를 줄였다.

그림 5는 전체 성능평가를 위한 에너지*지연시간을 나타낸 그림으로 기존 4-웨이 집합연관 캐시에 대해 평준화한 그림이다. 에너지*지연시간 지표는 에너지 및 성능의 효과적인 평가지표로 널리 사용되고 있다.

시뮬레이션 결과, 제안된 캐시는 기존 4-웨이 집합연관 캐시에 비해 78%의 성능향상을 이루었다. SWAC에 비해 14%, WTC에 비해 39%, 그리고 WC에 비해 15%의 성능향상을 이루었다. 앞서 언급처럼, SWAC는 순차적인 접근임에도 2-웨이 집합연관 캐시의 높은 실패율과 제안된 캐시에 비해 큰 용량의 웨이 때문에 높은 에너지 소비를 가진다. 또한 WTC의 경우, 제안된 캐시에 비해 다양한 웨이 접근으로 높은 에너지*지연시간을 가진다. 더욱이, WC는 높은 1cycle 접근 비율을 가짐에도 WC에 캐시 접근정보가 없다면, 기존 4-웨이 집합연관 캐시의 높은 에너지 소비를 가지기 때문에 에너지*지연시간에서 제안된 캐시 비해 낮은 성능향상을 보이고 있다.

V. 결론

본 논문에서는 높은 접근성공률을 가지는 4-웨이 집합연관 캐시에 대해 저전력 및 빠른 접근 시간을 보장 할 수 있는 효과적이고 간단한 알고리즘을 제안하였다.

제안된 알고리즘은 태그의 하의 2비트를 이용하여 기존 4-웨이 집합연관 캐시 중 2-웨이 접근으로 에너지 소비를 줄였으며, 더욱이 선택 되어진 2-웨이는 최근 접근성공이 발생된 웨이부터 순차적으로 접근하므로 빠른 접근시간 및 에너지* 지연시간의 성능향상을 이루었다.

시뮬레이션 결과, 제안된 캐시는 기존 4-웨이 집합연관 캐시에 비해 약 78%의 에너지*지연시간의 성능향상을 이루었다. 또한 다른 비교 캐시에(SWAC, WTC, WC) 비해 약 14~39%의 에너지*지연시간의 성능향상을 이루었다. 따라서 제안된 캐시는 다른 비교캐시에 비해 작은 용량의 추가적인 비트로 에너지*지연시간의 성능향상에 좋은 구조라 할 수 있다.

References

[1] http://www.intel.com/content/www/us/en/homepage.html.

[2] Motorola, “Errata to the MPC7450 RISC Microprocessor Family Reference Manual.”

[3] http://www.intel.co.kr/content/www/kr/ko/processors/core/core-i7-processor.html.

[4] S. Raghunath, L.D. Bobbala, N. Davanam, B.K. Lee, “Divide-and-Conquer Way Access for Low Power Mobile Cache,” IJICC, Vol. 3, No. 2, pp.297-302, 2011.

[5] O. Yoshiyasu, “An effective Replacement Strategy of Cache Memory for an SMT Processor,” Proceedings of the Euromicro Conference on Digital System Design, pp.19-25, 2009.

[6] C.J. Janraj, T.V. Kalyan, T. Warrier, M. Mutyam, “Way Sharing Set- Associative Cache Architecture,” Proceedings of the International Conference on VSLI Design, pp.251-256, 2012.

[7] D. Rolan, B.B. Fraguela, R. Doallo, “Adaptive line placement with the Set Balancing Cache,” Proceedings of the ACM International Symposium on Microarchitecture, pp.529-540, 2009.

[8] J. Kang, S. Lee, I. Lee, “Way-tracking set-associative caches,” Electronics Letters, Vol. 46, No. 22, pp.1497-1499, 2010.

[9] M.K. Qureshi, D. Thompson, Y.N. Patt, “The V-way Cache: Demand-based Associativity via Global Replacement,” Proceedings of the ISCA, pp.544-555, 2005.

[10] D. Nicolaescu, A. Veidenbaum, A. Nicolau, “Using a Way Cache to Improve Performance of Set-Associative Cache,” High-Performance Computing Lecture Notes in Computer Science Vol. 4759, pp.93-104, 2008

[11] C.H. Ting, J.D. Huang, Y.H. Kao, “Cycle-Time-Aware Sequential Way-Access Set-Associative Cache for Low Energy Consumption,” Proceedings of the IEEE Asia Pacific Conference on Circuits and Systems, pp.854-587, 2008.

[12] J.E. Fritts, F.W. Steiling, J.A. Tucek, “MediaBenchII video: Expediting the next generation of video systems research,” Vol. 33, No. 4, pp.301-318, 2009.

[13] C.K. Luk, R. Cohn, R. Muth, H. Patil, A. Klauser, C. Lowney, S. Wallace, V.J. Pedda, and H. Kimg, “Pin: building customized program analysis tools with dynamic instrum- entation,” Proceedings of ACM SIGPLAN Conference on PLDI. Vol. 40, pp.190-200, 2005.

저 자 소 개

정 보 성

2008년 2월: 경상대학교 제어 계측공학과 석사.
2008년~현재: 경상대학교 제어계측공학과 박사과정.
관심분야: 캐시 및 플래시 메모리.
Email: blueking80@gnu.ac.kr

이 정 훈

2004년 2월: 연세대학교 컴퓨터과학과 박사.
2004년~현재: 경상대학교 제어계측공학과(ERI) 부교수.
관심분야: 고성능 컴퓨팅, 내장형 시스템 및 SOC 시스템.
Email: leejh@gnu.ac.kr