

캐시 주소의 태그 이력을 활용한 에너지 효율적 고성능 데이터 캐시 구조

문 현 주[†] · 지 승 현^{††}

요 약

모바일 기기와 같이 배터리에 의존적인 시스템에서 사용되는 임베디드 프로세서는 총 소모 전력의 많은 부분을 캐시에서 소모한다. 본 논문에서는 임베디드 프로세서용 고성능 선인출 데이터캐시의 저전력화 방안을 연구하였다. 고성능 선인출 데이터캐시에서 메모리 참조명령의 수행에 앞서 참조예측의 결과로 발생하는 선인출 명령은 캐시 적중률을 높여 메모리 참조 시간을 단축하는 반면 선인출 명령의 수에 비례하여 전력 소모가 증가한다. 본 논문에서는 선인출 데이터캐시에 태그이력표(tag history table)를 구비하여 병렬태그탐색을 최소화함으로써 전력 소모를 줄이는 캐시 구조를 제안하였다. 실험을 통해 확인한 결과 제안한 데이터캐시 구조가 기존 데이터캐시 구조에 비하여 수행 시간과 전력 소모를 모두 줄일 수 있음을 확인하였다.

키워드 : 임베디드 프로세서, 저전력, 동적참조예측기, 데이터 선인출, 영역결정장치, 태그이력표

An Energy Efficient and High Performance Data Cache Structure Utilizing Tag History of Cache Addresses

Hyun Ju Moon[†] · Sung Hyun Jee^{††}

ABSTRACT

Uptime of embedded processors for mobile devices are dependent on battery consumption. Especially the large portion of power consumption is known to be due to cache management in embedded processors. This paper proposes an energy efficient data cache structure for high performance embedded processors. High performance prefetching data cache issues prefetching instructions before issuing demand-fetch instructions based on reference predictions. These prefetching instructions bring reduction on memory delay by improving cache hit ratio, but on the other hand those increase energy consumption in proportion to the number of prefetching instructions. In this paper, we adopt tag history table on prefetching data cache for reducing energy consumption by minimizing parallel tag comparison. Experimental results show the proposed data cache improves performance on energy consumption as well as memory delay.

Key Words : Embedded Processor, Low-energy, RPT, Data Prefetching, WDU, Tag History Table

1. 서 론

모바일 분야에 주로 사용되는 임베디드 프로세서는 배터리에 의존적이므로 데이터참조에 따른 실행지연이나 집중적인 전력소모 시에 전력한계 문제를 자주 발생시킨다[1]. 임베디드 프로세서는 사용하는 총 에너지소모율 가운데 많은 부분을 캐시에서 소모하고 있는데, 예를 들어 StrongARM-1의 경우 총 에너지소모율 중 캐시참조로 인한 에너지소모율이 43%를 차지한다[2]. 따라서 많은 연구자들이 임베디드 프로세서에 적합한 에너지 효율적 고성능 캐시구조를 설계

하여 시스템의 성능향상과 저전력 문제를 해결하는 연구를 진행하고 있다.

인텔(Intel)은 임베디드 프로세서의 성능향상을 위하여 캐시설계 시에 데이터 선인출(data prefetching) 기법을 채용하고 있다[3-6]. 선인출 기법은 캐시적중률을 높이고 CPU 대기시간을 단축시켜서 전체 성능을 향상시키는 장점을 가지나 선인출을 위한 별도의 에너지를 필요로 하므로 에너지 효율 면에서는 성능향상에 저해가 된다. 임베디드 프로세서의 주요 성능지표인 저전력 구현방안은 캐시접근 시에 태그(tag)를 예측함으로써 태그메모리에서의 병렬태그탐색(parallel tag lookup)을 최소화하는 방향으로 진행되고 있다. 대표적인 저전력 구현 방안인 영역결정장치(WDU: Way Determination Unit) 기법은 앞서 참조한 메모리주소의 태그(tag)와 영역

[†] 정 회 원 : 남서울대학교 컴퓨터학과 전임강사
^{††} 정 회 원 : 한국고용정보원 정보화전략팀 부연구위원
논문접수 : 2006년 8월 8일, 심사완료 : 2007년 1월 11일

(way)정보를 이용하여 앞으로 참조할 메모리 블록의 영역을 결정하는 방법이다. 즉, 태그이력표(tag history table)에 저장된 태그-영역정보를 이용하여 병렬태그탐색 없이 해당 영역을 참조하여 자료를 읽을 수 있다. 이와 같은 기법은 캐시참조시 태그를 병렬로 탐색할 때 필요한 전력을 현저히 감소시킬 수 있다[7].

본 논문에서는 지금까지의 관측을 바탕으로 임베디드 프로세서에 적합한 고성능 저전력 데이터캐시 구조를 제안하고 성능을 평가하였다. 제안한 데이터캐시 구조는 기존 데이터캐시 구조에 선인출(prefetch) 장치와 태그이력표(tag history table)를 구비함으로써 캐시적중률을 높이며 캐시참조시 태그메모리에 대한 병렬탐색 횟수를 최소화하였다. 이러한 전략적인 데이터캐시 구조는 캐시적중률을 높여서 시스템의 전체 실행시간을 단축시킬 뿐 아니라 태그메모리에 대한 병렬탐색 횟수를 줄임으로써 병렬탐색에 필요한 전력을 절약함으로써 전체 전력소모를 줄일 수 있다.

본 논문의 제 2절에서는 기존 데이터캐시의 설계 전략을 요약하고, 에너지 효율적인 고성능 데이터캐시의 구조를 제안한다. 제 3절에서는 이미지 처리 프로그램의 예를 바탕으로 캐시구조별 수행 방식 및 성능을 비교한다. 제 4절에서는 제안한 캐시구조와 기존의 데이터캐시 구조의 시뮬레이터를 구성하고 다양한 벤치마크 프로그램들을 수행한 결과를 제시하고 캐시구조별 성능을 비교·분석한다. 마지막으로 제 5절에서는 논문의 결론을 맺는다.

2. 에너지 효율적 고성능 데이터캐시 구조 설계

임베디드 프로세서에 적합한 데이터캐시 구조를 선택하기 위해서는 데이터캐시가 시스템의 성능과 저전력에 미치는 영향을 모두 고려해야 한다. 일반적으로 시스템의 성능을 향상시키려는 데이터선인출 등의 시도들은 별도의 전력을 필요로 하므로 높은 에너지소모율을 요구한다. 따라서 데이터캐시 설계는 시스템성능 또는 저전력의 성능지표를 모두 고려해야 한다.

2.1 데이터캐시 설계 전략

2.1.1 데이터선인출을 통한 고성능 전략

인텔을 포함한 임베디드 프로세서 설계자들은 캐시를 설계할 때 데이터선인출 기법을 채용하여 시스템의 전체 성능을 향상시킨다. 대표적인 데이터선인출 기법인 동적참조예측기(RPT: Reference Prediction Table) 기법[6]은 참조예측표를 포함하는 참조예측기를 이용하여 반복해서 참조하는 메모리정보를 저장 및 갱신하면서 미래에 참조될 메모리주소를 예측한다. 만약 예측된 주소가 캐시에 존재하지 않으면 미리 원하는 메모리주소 정보를 메모리로부터 캐시에 가지고 옴으로써 캐시미스를 발생시키지 않는다. 이와 같은 선인출의 장점에도 불구하고, 캐시에서의 데이터선인출은 참조할 메모리주소가 캐시 내에 있는지를 미리 참조하고 후에 실제로 캐시를 참조하므로 기존 캐시참조 횟수에 비해

두 배의 캐시참조횟수가 필요하다. 그러므로 데이터선인출은 캐시적중률을 높여서 시스템의 성능을 향상시키는 반면, 별도의 전력을 필요로 하므로 저전력 측면에서는 저해 요소가 될 수 있다.

2.1.2 태그메모리의 병렬탐색 최소화를 통한 저전력 전략

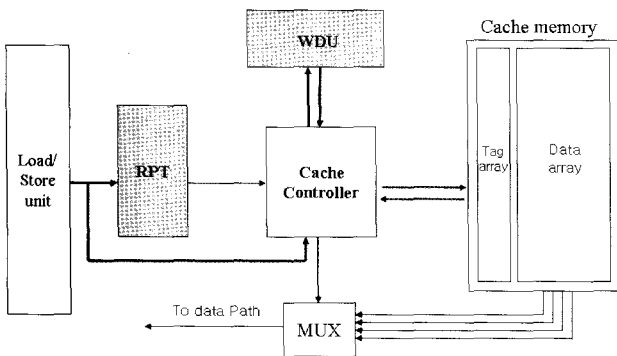
임베디드 프로세서의 저전력을 구현하기 위하여, 캐시 설계자는 캐시접근 시에 태그메모리의 병렬탐색을 최소화하는 연구를 진행하고 있다. 대표적인 병렬탐색 최소화 연구인 영역결정장치(WDU: Way Deterministic Unit) 기법[7]은 캐시참조시 태그이력표(tag history table)에 저장된 태그정보를 이용하여 태그메모리의 병렬탐색 없이 해당 영역을 참조하여 자료를 읽는다. 따라서 이 기법은 태그예측 방법에서 발생할 수 있는 페널티도 고려할 필요가 없으며 다 기법과 비교할 때 최소한의 하드웨어 비용으로도 전력소모를 줄일 수 있는 장점을 지니고 있으나 메모리의 참조 속도 향상과는 무관하다.

이상과 같은 캐시연구는 시스템의 성능향상 혹은 낮은 전력소모에 관한 한 분야의 성능지표 연구에만 집중되어 있다 [10-12]. 본 논문에서는 이 두 가지 성능지표들을 고려하여 고성능 및 저전력을 함께 추구하는 데이터캐시 구조를 제안한다. 제안하는 고성능-저전력 데이터캐시 구조는 기존 데이터캐시의 선인출 기법과 태그메모리의 병렬탐색 최소화 기법을 적용하여 시스템의 성능을 양각에서 향상시킨다. 이하 기존 전통적 데이터캐시 구조를 T-캐시(Traditional Cache), 기존 선인출 데이터캐시 구조를 R-캐시(RPT-Cache), 기존 영역결정장치(WDU)를 갖춘 데이터캐시 구조를 W-캐시(WDU-Cache), 본 논문에서 제안하는 에너지 효율적 고성능 데이터캐시 구조를 WR-캐시(WDU-RPT Cache)라고 명한다.

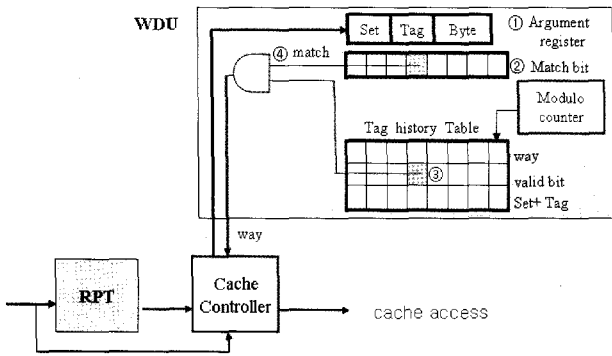
2.2 캐시 구조 설계

WR-캐시 구조는 (그림 1)과 같이 동적참조예측기(RPT), 영역결정장치(WDU), 캐시컨트롤러(cache controller) 및 캐시메모리로 구성된다. 여기서 동적참조예측기는 데이터 선인출을 수행하기 위하여 이전에 참조한 주소 값으로부터 규칙성을 발견하여 다음에 참조할 메모리주소를 미리 계산한다. 영역결정장치는 캐시참조시 태그이력표에 저장된 태그정보를 이용하여 태그메모리의 병렬탐색 없이 원하는 정보를 읽을 수 있다. 이 때 영역결정장치의 엔트리 개수가 8~16 개인 경우, 가장 효과적인 에너지감소 효과와 성능향상을 가진다[7]. 캐시컨트롤러는 참조예측기로부터 전달된 메모리주소를 캐시라인주소로 변환(decode)하고 구성요소간 캐시라인주소의 전달 및 장치 간 동작을 관리한다.

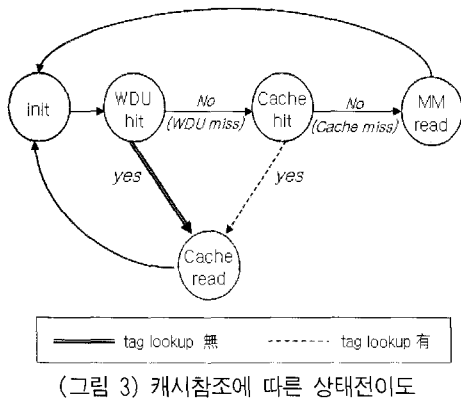
(그림 2)는 WR-캐시에서 영역결정장치내 태그이력표를 이용한 태그탐색 과정을 보여준다. 태그이력표 내 각 항목은 집합(set)과 태그(tag), 영역(way) 및 유효비트(valid bit)로 구성된다. 각 항목은 모듈러 카운터(modulo counter)를 이용하여 LRU 기법으로 교체된다. 먼저 캐시컨트롤러는 동



(그림 1) WR-캐시(WDU-RPT Cache) 구성도



(그림 2) 영역결정장치치의 태그탐색



(그림 3) 캐시참조에 따른 상태전이도

적참조예측기(선인출시) 또는 로드/스토어장치(요구인출시)로부터 전달받은 주기억장치 주소를 캐시라인주소로 변환하여 인수레지스터(argument register)에 저장시킨다(①). 인수레지스터(집합, 태그)와 태그이력표의 엔트리가 일치(적중)하고(②) 유효비트가 설정되면(③), 찾은 태그정보가 영역결정장치에 저장됨을 의미하므로 캐시컨트롤러에게 영역정보를 전달한다(④). 캐시컨트롤러는 영역정보를 이용하여 병렬 태그탐색 없이 원하는 정보를 읽어온다.

(그림 3)은 WR-캐시 구조에서의 캐시참조 과정을 보여준다. 먼저, 초기상태에서 태그이력표에 원하는 태그정보가 존재하는지를 조사한다. 만약 태그정보가 존재하면(WDU hit), 캐시컨트롤러는 태그정보를 이용하여 태그메모리에 대한 병렬탐색 없이 캐시로부터 정보를 읽어 온다(cache read). 만약 영역결정장치에서 동일한 태그가 발견되지 않으면(WDU miss), 캐시컨트롤러는 캐시에 원하는 정보가 존재하는지 조사한다. 만약 캐시에 원하는 정보가 저장되었다면 (Cache hit), 캐시컨트롤러는 캐시에서 해당 정보를 읽고 동시에 태그와 영역정보를 영역결정장치에 전달하여 태그이력표에 저장한다. 만약 캐시미스(cache miss)가 발생되면, 원하는 데이터 블록을 메모리로부터 데이터캐시로 가져온다.

WR-캐시 구조는 태그이력표에 유용한 태그정보가 많을수록, 캐시참조 시 태그메모리에 대한 병렬탐색 횟수도 줄어든다. 만약 태그이력표에 일치하는 태그정보가 없을 경우, 캐시컨트롤러는 기존 캐시참조와 동일하게 태그메모리에 대한 병렬탐색을 수행하여 원하는 자료를 읽어온다. 태그이력표는 완전연관사상 캐시(full-associative cache)를 이용하여 용이하게 구현할 수 있으며, 태그이력표의 항목 개수가 집합연관사상 캐시의 연관도보다 적은 경우에는 병렬태그탐색에 비하여 적은 전력으로 캐시 참조를 수행할 수 있다. 뿐만 아니라, 태그이력표를 CAM-tag 구조의 완전연관사상 캐시로 구현함으로써 소모 전력을 더욱 줄일 수 있다[7].

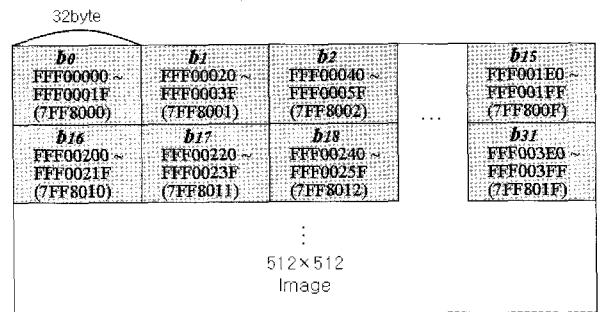
3. 캐시 구조 변화에 따른 성능 영향

모바일 시스템과 같이 임베디드 프로세서를 채용하는 플랫폼에서 핵심적으로 사용되는 이미지 처리 프로그램의 예를 들어 선인출 및 영역결정장치(WDU)의 채용에 의한 성능 변화를 설명할 것이다. (그림 4)(a)는 2차원 배열형태로 적재된 이미지의 각 원소를 읽고 처리하는 프로그램의 일부이고, (그림 4)(b)는 프로그램의 수행시점에서 배열이 적재된 메모리의 상태를 2차원 형태로 표시한 것이다. 배열 image는 0xFFFF0000 번지를 시작주소로 하여 메모리에 적재되었으며 메모리 블록 크기는 32byte로 가정하였다. b_i 는 각 메모리 블록을 나타내며, 각 메모리 블록에는 메모리주소 범위를 표시하고 캐시라인주소를 ()에 표시하였다.

```

unsigned char image[512][512];
for(i=0; i<512; i++)
for(j=0; i<512; j++)
... = image[i][j];
    
```

(a) 이미지 처리 프로그램의 일부



(b) 메모리 상태

(그림 4) 이미지 처리 프로그램 예제

3.1 메모리 참조 명령 수행 과정

- (1) T-캐시 : 배열 image의 각 원소가 참조될 때마다 요구인출 명령이 발생한다. 참조되는 원소가 메모리 블록의 첫 번째 원소(바이트)인 경우에는 태그비교의 결과에 따른 캐시 적중 여부에 따라 캐시 미스시에 해당 메모리 블록이 캐시로 적재되며, 나머지 원소에 대한 요구인출 시에는 메모리 블록이 이미 캐시에 적재되었으므로 태그 비교만이 수행된다. 따라서 T-캐시에서 요구인출 및 태그비교 횟수는 총 배열 원소수인 $512 \times 512 (= 262,144)$ 회가 된다.
- (2) R-캐시 : 배열 image의 처음 3개 원소(0xFFFF0000h ~ 0xFFFF0002h)가 참조되는 동안에는 각 원소에 대한 요구인출 명령이 발생된다. 각 요구인출 명령이 수행될 때마다 참조된 메모리 주소는 참조예측기로 전달되고 이들 간에 선형 규칙성이 존재하므로 4번째 원소, 즉 FFF00003h번지부터는 요구인출 이전에 선인출 명령이 먼저 발생된다. 참조되는 메모리 주소간의 선형 규칙성은 배열의 마지막 원소가 참조될 때까지 유지되므로 배열의 4번째 ~ 마지막 원소까지 선인출 명령이 발생된다. 선인출 명령에 의하여 참조되는 메모리 주소는 우선 태그비교가 수행되고 캐시 미스시에는 해당 메모리 블록을 캐시로 적재한다. 따라서 각 메모리 블록의 첫 번째 원소에 대한 선인출 명령이 발생할 때마다 캐시로의 적재가 이루어진다. 뿐만 아니라, 각 배열 원소의 요구인출시에도 해당 메모리 블록의 캐시 적중 여부를 확인하기 위하여 태그비교가 수행된다. 따라서 R-캐시에서 요구인출 횟수는 배열 원소수인 262,144회이고, 선인출 횟수는 262,141회이며, 태그비교 횟수는 이들의 합이 된다.
- (3) W-캐시 : 요구인출 되는 메모리 주소에 대한 태그비교를 수행하기 전에 영역결정장치(WDU)의 태그이력표를 먼저 검색한다. 각 메모리 블록의 첫 번째 원소가 참조되는 시점에서는 태그이력표에 해당 메모리 블록의 영역 정보가 적재되지 않았으므로 WDU 미스가 발생한다. 이에 따라 태그 비교를 수행하여 메모리 블록이 적재되었거나 적재될 영역(way)을 결정하고 이 정보를 태그이력표에 갱신한다. 이후부터는 메모리 블록의 마지막 원소가 참조될 때까지 WDU에서 영역 정보가 검색되므로 매 요구인출시 필요로 되는 태그비교를 생략할 수 있다. 따라서 W-캐시에서 요구인출 횟수는 262,144회이고, 이 중 각 메모리 블록의 첫 번째 원소에 대한 요구인출시에만 태그비교가 수행되므로 태그비교 횟수는 $262,144/32 (= 8,192)$ 회이다. 단, 태그비교가 수행되려면 이에 앞서 WDU 미스가 전제되므로 WDU 검색 횟수는 총 배열 원소수와 같다.
- (4) WR-캐시 : 요구인출 및 선인출 되는 메모리 주소에 대한 태그비교를 수행하기 전에 영역결정장치(WDU)의 태그이력표를 먼저 검색한다. 첫 번째 메모리 블록을 제외한 모든 메모리 블록은 선인출 된 후에 요구

인출 되므로 각 메모리 블록의 첫 번째 원소에 대한 선인출시 WDU 미스가 발생한다. 이 때 영역(way)이 결정되면 블록내 나머지 원소에 대한 선인출 및 블록내 모든 원소의 요구인출시에는 WDU에서 적중되므로 태그비교를 생략할 수 있다. 따라서 WR-캐시에서 총 인출횟수는 (262,144+262,141)이고 태그비교 횟수는 첫 번째 블록을 제외한 블록의 총 수, 즉 8,191회이며, WDU 검색 횟수는 총 인출 횟수와 같다.

3.2 성능 비교

3.2.1 캐시 구조별 전력 소모

메모리 참조시 태그 비교를 위한 캐시구조별 전력 분석 모델은 다음과 같이 정의할 수 있다.

$$\text{T-캐시} : N_D \times E_T$$

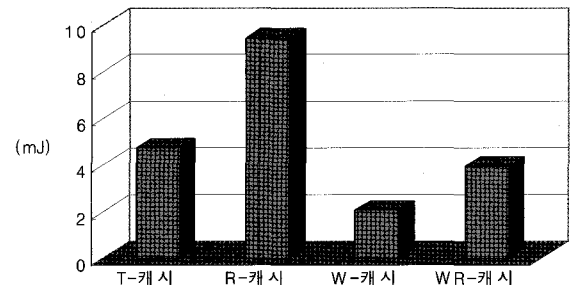
$$\text{R-캐시} : (N_D + N_P) \times E_T$$

$$\text{W-캐시} : (N_D \times E_W) + (N_D^{\overline{W}} \times E_T)$$

$$\text{WR-캐시} : (N_D + N_P) \times E_W + (N_D^{\overline{W}} + N_P^{\overline{W}}) \times E_T$$

여기서, N_D 는 요구인출 횟수이고 N_P 는 선인출 횟수를 나타내며, $N_D^{\overline{W}}$ 는 요구인출 중 WDU 미스가 발생한 횟수이고 $N_P^{\overline{W}}$ 는 선인출 중 WDU 미스가 발생한 횟수를 나타낸다. 또한 E_T 는 태그비교에 소요되는 단위 전력이고 E_W 는 WDU 검색에 소요되는 단위 전력을 나타낸다.

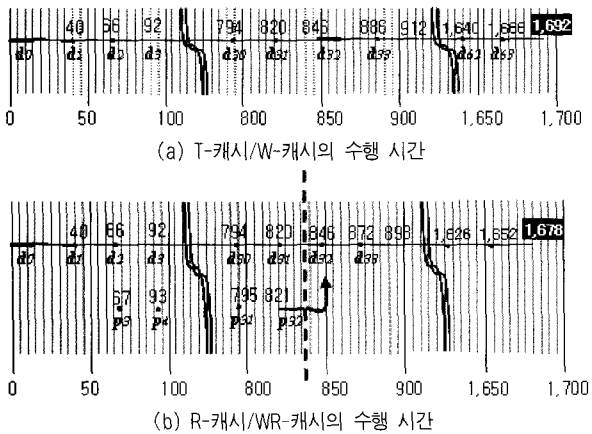
각 캐시의 크기가 32Kbyte이고 32-way 세트연관사상 방식을 채택한 경우를 가정하고, W-캐시 및 WR-캐시에서 사용하는 영역결정장치(WDU)의 태그이력표를 1Kbyte의 완전연관사상 캐시로 구현한 경우 CACTI 모델[10]을 바탕으로 하면 캐시의 태그비교 단위 전력 $E_T = 17.98nJ$ 과 WDU 검색의 단위전력 $E_W = 7.23nJ$ 을 구할 수 있다. 이를 바탕으로, 각 캐시구조에서 (그림 4)의 응용프로그램을 수행하는 동안 태그비교에 소모되는 총에너지를 계산한 결과가 (그림 5)와 같다.



(그림 5) 캐시구조별 전력 소모량

3.2.2 캐시 구조별 수행 시간

(그림 6)은 (그림 4)의 프로그램에 대한 캐시구조별 수행과정 중 2개의 메모리 블록에 대한 참조와 처리 과정을 시간의 흐름에 따라 나타낸 것이다. 그림에서 하나의 배열 원



(그림 6) 시간흐름에 따른 캐시구조별 수행 과정

소에 대한 수행을 —와 같이 표시하였는데 굵은 선은 배열원소에 대한 캐시 또는 메모리 참조시간을 나타내며 가는 선은 이를 활용한 처리시간을 나타낸다. d_i 와 p_i 는 각각 배열원소 i 에 대한 요구인출과 선인출을 나타내며, 캐시적중과 캐시미스의 지연시간은 각각 1cycle과 15cycle, 각 배열원소에 대한 처리시간은 25cycle로 가정하였다.

(그림 6)(a)는 T-캐시와 W-캐시의 경우로, 배열 원소가 순차적으로 참조됨에 따라 요구인출이 순차적으로 수행되는 것을 나타낸다. 이 때, 각 메모리 블록의 첫 번째 원소(바이트)에서 캐시미스로 인하여 메모리 접근을 위한 지연이 발생하고 있다. 나머지 경우에는 캐시에서 적중되므로 참조시간이 1cycle로 표시되었다. T-캐시와 비교하여 W-캐시의 경우에는 캐시적중 여부를 결정하는 과정에서 WDU 참조와 태그비교가 중복하여 일어날 수 있으나 이것이 전체 참조시간에서 차지하는 비중이 크지 않으므로 별도로 표시하지 않았다.

(그림 6)(b)는 R-캐시와 WR-캐시의 경우로, 요구인출 d_i 가 수행된 직후부터 배열원소의 처리시간에 선인출을 중복하여 수행하고 있다. $p_3 \sim p_{31}$ 까지의 선인출은 전체 수행시간에 영향을 미치지 못하나, 캐시미스가 발생하는 p_{32} 의 선인출은 메모리 참조를 이전 배열 원소의 처리와 중복하여 수행함으로써 d_{32} 의 지연을 제거하고 있다.

3.2.3 캐시 구조별 성능 비교

R-캐시는 T-캐시에 비하여 메모리 참조 지연을 줄이는 반면 캐시 참조 횟수가 증가하여 전력 소모가 크게 증가하는 부작용을 내포하고 있다. 또한 W-캐시는 병렬태그비교 과정을 이보다 전력소모가 적은 WDU 검색 과정으로 대체함으로써 전체 소모 전력을 감소시키는 반면 메모리 참조속도 측면에서는 T-캐시에 비하여 성능을 향상시키지 못한다. 이와 같이 R-캐시 또는 W-캐시는 상반되는 두 가지 성능 목표에 대하여 하나의 목표만을 향상시키고 있다. 이에 반해 WR-캐시는 T-캐시에 비하여 메모리 참조 지연을 줄임과 동시에 전체 소모 전력도 감소됨으로써 총체적인 성능 향상을 이룰 수 있다.

4. 실험 및 성능평가

4.1 실험환경과 벤치마크

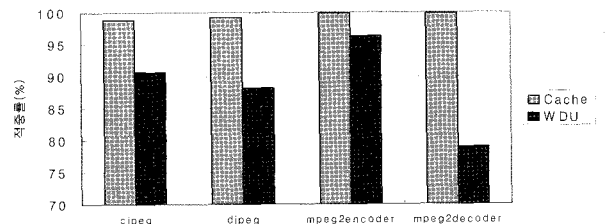
본 논문에서는 WR-캐시 구조의 성능을 분석하기 위하여, T-캐시, R-캐시, W-캐시, WR-캐시 구조별 시뮬레이터를 구현하고 벤치마크 수행 결과를 분석하였다. 캐시 시뮬레이터는 위스콘신 대학에서 개발한 트레이스 구동형 캐시 시뮬레이터인 Dinero III[8]를 기반으로 영역결정장치 및 선인출 메커니즘을 추가하여 구현하였다. 실험은 벤치마크 프로그램의 명령어 트레이스 생성, 데이터캐시 시뮬레이션, 소모전력 계산으로 구성된다.

벤치마크 프로그램의 명령어 트레이스는 Alpha CPU용 목적코드 분석도구인 ATOM 시뮬레이터[9]를 사용하여 생성하였다. 명령어 트레이스는 연산, 분기 및 메모리 참조 명령어가 혼합된 형태로 생성되는데, 메모리참조 명령어는 캐시의 수행과정 모의실험으로 사용되며 연산/분기 명령어는 메모리참조 지연시간을 측정목적으로 사용된다. 데이터캐시 시뮬레이션은 캐시크기, 블록크기, 연관도, 교체정책 등을 매개변수로 입력 받아 벤치마크 수행시 캐시, 동적참조예측기, 영역결정장치의 동작을 모의실행한 후 캐시적중률, 실행사이클, 선인출 적중률, 영역결정장치 적중률 등의 실험결과를 얻는다. 각 캐시의 소모전력은 캐시 시뮬레이션으로부터 도출된 실험 결과와 Cacti 모델[10]에 의하여 계산한 단위 전력을 3.2절의 전력 분석 모델에 적용하여 계산하였다.

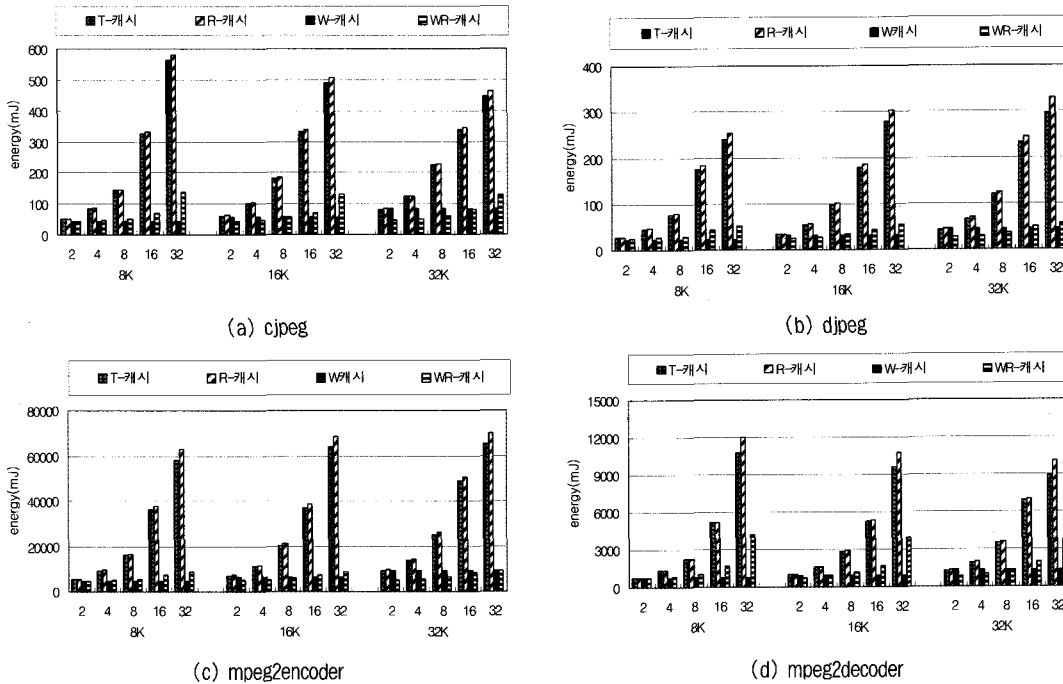
실험에 사용된 벤치마크는 멀티미디어 처리의 대표적 프로그램인 MPEG과 JPEG으로써 각 프로그램은 압축모듈과 해제모듈로 구성되며 각 워크그룹이 제공하는 데이터를 대상으로 실험하였다. 실험은 캐시 크기가 8K, 16K, 32K이고 캐시 연관도가 2~32 인 경우에 대하여 수행하였으며 영역결정장치의 엔트리 수는 8개로 구현하여 실험하였다.

4.2 성능평가 및 분석

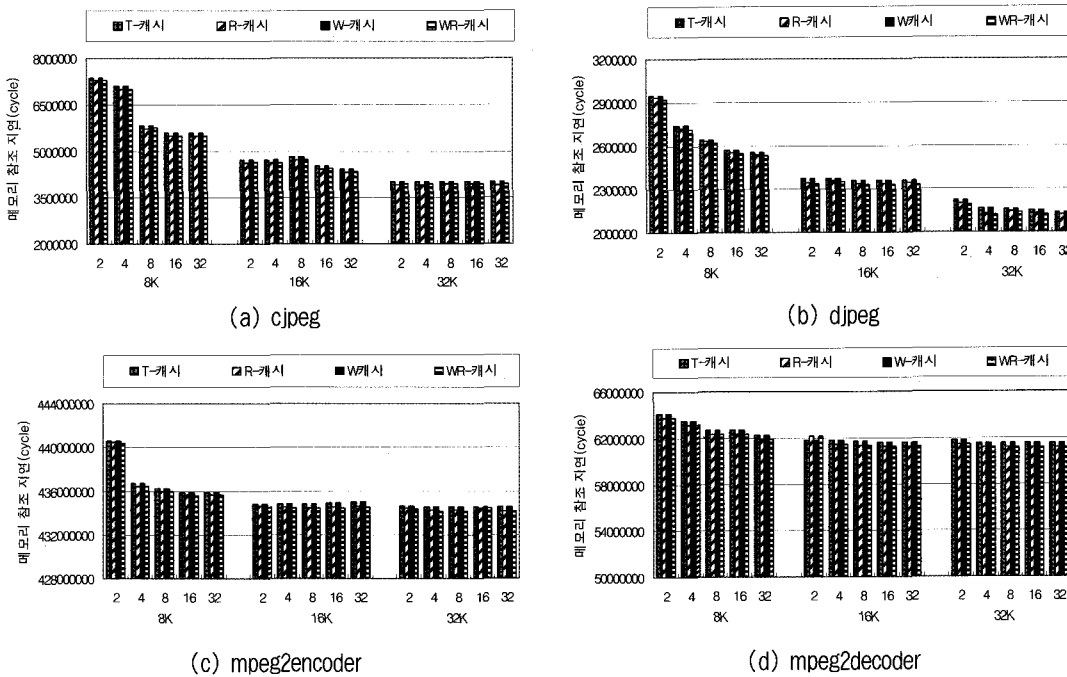
(그림 7)은 캐시 크기가 8K~32K이고 연관도가 2~32인 WR-캐시 구조에서 각 벤치마크 프로그램을 수행한 경우의 캐시 적중률에 대한 평균값과 영역결정장치(WDU) 검색의 적중률에 대한 평균값을 나타낸다. (그림 7)에서 각 벤치마크에 대한 캐시 적중률이 98.75~99.9로 매우 높은 것을 볼 수 있으며, 매 캐시 참조에 앞서 수행되는 영역결정장치 검색의 적중률은 캐시 적중률에 비하여 낮은 78.96~96.32%를 나타냈다. 그러나 실험에 사용된 영역결정장치의 엔트리 수가 8개로 매우 적은 것을 감안한다면 영역결정장치 또한 매



(그림 7) 영역결정장치(WDU) 검색 적중률



(그림 8) 캐시 구조별 전력 소모량



(그림 9) 캐시 구조별 메모리 참조 시간

우 높은 적중률을 나타내는 것으로 볼 수 있다. 이러한 높은 적중률은 동일한 캐시 라인에 포함된 메모리 블록간의 지역성이 높다는 분석을 가능하게 하며, 배열 인덱스와 같이 반복적으로 사용되는 메모리 블록이 처음 참조된 이후 사용될 때마다 캐시 뿐 아니라 영역결정장치에서도 적중되기 때문에 판단할 수 있다. 벤치마크별 영역결정장치 적

중률을 비교할 때 mpeg2encoder의 적중률이 가장 높은 반면 mpeg2decoder의 적중률이 가장 저조한 것을 볼 수 있다.

(그림 8)은 각 캐시 구조에서 프로그램 수행에 소모되는 전력량을 비교한 것이다. T-캐시를 기준으로 R-캐시는 전력 소모량이 증가하였으며, W-캐시와 WR-캐시는 전력 소모량이 감소한 것을 볼 수 있다. 영역결정장치를 채용하지

얇은 캐시구조와 채용한 캐시구조의 성능을 비교하면 T-캐시에 대하여 W-캐시는 전력 소모량이 평균 51.6% 감소하였으며 R-캐시에 대하여 WR-캐시는 전력 소모량이 평균 60.93% 감소하였다. 이는 RPT 선인출 기법이 근본적으로 메모리 참조의 선형 규칙성을 활용하는 방법이므로 선인출되는 메모리 블록들이 참조의 지역성을 지니고 이것이 영역결정장치에서도 활용되어 높은 성능 향상을 나타낸 것으로 분석할 수 있다.

응용 프로그램별로 영역결정장치를 채용하지 않은 캐시구조(T-캐시, R-캐시)에 대하여 영역결정장치를 채용한 캐시구조(W-캐시, WR-캐시)의 전력 감소율을 살펴보면 mpeg2encoder의 경우가 가장 높고 mpeg2decoder의 경우가 가장 낮은 것으로 나타났다. 이는 (그림 7)에 제시한 바와 같이 영역결정장치 적응률이 높을수록 전력 감소율도 높아짐을 나타낸다.

(그림 9)는 각 캐시 구조에서의 메모리 참조 시간을 비교한 것이다. T-캐시를 기준으로 R-캐시와 WR-캐시는 메모리 참조 시간이 감소한 반면 W-캐시는 메모리 참조 시간에 변화가 없는 것을 알 수 있다.

이상의 실험 결과로부터 WR-캐시가 선인출과 영역결정장치의 장점을 혼합함으로써 캐시의 상반되는 두 가지 목표 즉, 고성능과 저전력을 성취함을 알 수 있다.

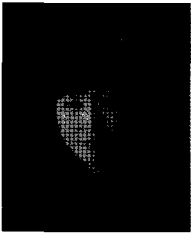
5. 결 론

본 논문에서는 임베디드 프로세서에 적합한 에너지 효율적 고성능 데이터 캐시 구조를 제안하였다. 메모리 시스템의 참조속도 향상을 위하여 선인출 장치를 채용하는 기존의 고성능 캐시구조는 선인출 명령으로 인하여 태그 병렬 탐색 횟수가 증가함으로써 전력 소모가 증가하는 문제점을 지닌다. 이러한 문제점을 해결하도록 본 논문에서는 태그이력표(tag history table)를 포함하는 영역결정장치(WDU:Way Determination Unit)를 데이터 캐시에 추가하였다.

제안한 캐시 구조에서 8~16개에 해당하는 소수개의 엔트리로 구성된 태그이력표의 검색은 태그 메모리의 병렬탐색에 의하여 전력 소모가 낮으므로 참조되는 메모리 주소의 영역정보가 태그이력표에서 검색되는 경우에 전력 소모를 줄일 수 있다. 특히 영역결정장치에 의한 전력감소 효과는 메모리 참조의 지역성이 높은 경우에 우수하게 나타나며, 이는 지역성에 기반을 둔 선인출의 효과와 부합하여 성능을 극대화 할 수 있다. 다양한 캐시 구조의 시뮬레이터에 의한 실험 결과, 메모리 참조 속도 및 전력 소모의 측면에서 제안한 캐시 구조가 성능을 향상시킴을 알 수 있었다.

참 고 문 헌

- [1] T. Mudge, "Power: A First-Class Architectural Design Constraint," *IEEE Computer*, Vol.34, No.4, pp.52-58, Apr., 2001.
- [2] M. Baron, "Analog and CPU Wizards Reduce Digital Power: National Semiconductor and ARM Increase Battery Life," *Microprocessor Report*, Vol.17, No.1, pp.10-14, Jan., 2003.
- [3] A. Smith, "Sequential Program Prefetching in Memory Hierarchies," *IEEE Computer*, Vol.11, No.2, pp.7-21, 1978.
- [4] B. Mathew and A. Davis, "An Energy Efficient High Performance Scratch-pad Memory System" *Proceedings of the Design Automation Conference (DAC)*, 2004.
- [5] F. Dahlgren, M. Dubois and P. Stenstrom, "Fixed and Adaptive Sequential Prefetching in Shared-memory Multiprocessors," *Proceedings of the International Conference on Parallel Processing*, pp.156-63, Aug., 1993.
- [6] T. F. Chen and J. L. Baer, "Effective Hardware-Based Data Prefetching for High Performance Processors," *IEEE Transactions on Computers*, Vol. 44, No.5, pp.609-623, May, 1995.
- [7] D. Nicolaescu, Al. Veidenbaum and A. Nicolau, "Reducing Power Consumption for High-Associativity Data Caches in Embedded Processors," *Proceedings of the conference on Design, Automation and Test in Europe*, pp.11064-11069, Mar., 2003.
- [8] M. D. Hill, Dinero III Cache Simulator, <http://www.ece.cmu.edu/~ece 548/tools/dinero>
- [9] A. Srivastava and A. Eustace, "ATOM: A System for Building Customized Program Analysis Tools," *Proceedings of the ACM SIGPLAN 94*, pp.196-205, 1994.
- [10] P. Shivakumar and N. P. Jouppi, "CACTI 3.0: An Integrated Cache Timing, Power, and Area Model," *HP Western Research Labs, Tech. Rep.* 2001.
- [11] M. Zhang and K. Asanovic, "Highly-Associative Caches for Low-Power Processors," *33rd International Symposium on Microarchitecture*, pp.196-205, Dec., 2000.
- [12] W. T. Shiu, "Memory Exploration for Low Power Embedded Systems", *Proceedings of the 1999 IEEE International Symposium on Circuits and Systems*, pp.250-253, May, 1999.



문 현 주

e-mail : semulset@nsu.ac.kr

1995년 충북대학교 컴퓨터과학과
(이학사)

1997년 충북대학교 전자계산학과
(이학석사)

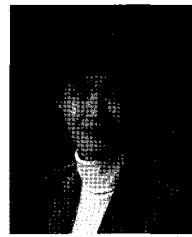
2003년 충북대학교 전자계산학과
(이학박사)

2003년~2005년 나사렛대학교 정보과학부 전임강사

2005년 3월~2005년 8월 유비쿼터스바이오정보기술연구센터
연구원

2005년 9월~현재 남서울대학교 컴퓨터학과 전임강사

관심분야: 캐시구조, 임베디드 시스템



지 승 현

e-mail : jshjsh@work.go.kr

1993년 충북대학교 전자계산학과(이학사)

1995년 충북대학교 전자계산학과(이학석사)

2000년 충북대학교 전자계산학과(이학박사)

2000년~2003년 미국 미주리주립대
연구교수

1999년~2005년 백석대학 디지털정보학부 전임강사

2005년 9월~2006년 8월 한국산업기술대학교 교육학과(컴퓨터교육)
초빙교수

2006년 9월~현재 한국고용정보원 정보화전략팀 부연구위원

관심분야: 캐시구조, 임베디드 시스템, 유비쿼터스 컴퓨팅