기억장치 (2)

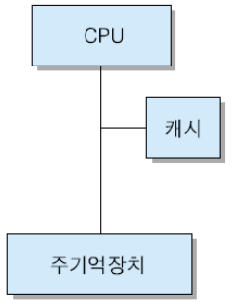
컴퓨터구조론 5장 후반부

연세대학교

5.5 캐시 기억장치(Cache memory)

연세대학교

- 사용 목적 :
  - CPU와 주기억장치의 속도 차이로 인한 CPU 대기 시간을 최소화 시키기 위하여 CPU와 주기억장치 사이에 설치하는 고속 반도체 기억장치
- 특징
  - 주기억장치보다 액세스 속도가 빠른 칩 사용
  - 가격 및 제한된 공간 때문에 용량이 적다



캐시 기억장치

연세대학교

- 캐시 적중(cache hit)과 캐시 미스(cache miss)
  - 캐시 적중: CPU가 원하는 데이터가 캐시에 있는 상태
  - 캐시 미스: CPU가 원하는 데이터가 캐시에 없는 상태.
    이 경우에는 주기억장치로부터 데이터를 읽어온다.
- 적중률(hit ratio) : 기억장치 액세스 중에서 캐시에 적중되는 비율(H)

$$H = \frac{\text{캐시에 적중되는 횟수}}{\text{전체 기억장치 액세스 횟수}}$$

- 캐시의 미스율(miss ratio) = 1 - H

- 평균 기억장치 액세스 시간(Ta) :

$T_a = H \times T_c + (1 - H) \times T_m$

(T_c는 캐시 액세스 시간, T_m은 주기억장치 액세스 시간)

평균 기억장치 액세스 시간의 예

연세대학교

예제 5-1

$Tc = 10ns$, $Tm = 100ns$인 시스템에서 캐시 적중률(H)이 70%, 80%, 90%, 95% 및 99%일 때의 평균 기억장치 액세스 시간(Ta)을 구하라.

[풀이]

식 (5-2)를 이용하여 구하면, 아래와 같아진다.

$H = 70\%$의 경우 : $Ta = 0.7 \times 10ns + 0.3 \times 100ns = 37ns$

$H = 80\%$의 경우 : $Ta = 0.8 \times 10ns + 0.2 \times 100ns = 28ns$

$H = 90\%$의 경우 : $Ta = 0.9 \times 10ns + 0.1 \times 100ns = 19ns$

$H = 95\%$의 경우 : $Ta = 0.95 \times 10ns + 0.05 \times 100ns = 14.5ns$

$H = 99\%$의 경우 : $Ta = 0.99 \times 10ns + 0.01 \times 100ns = 10.9ns$

- 캐시 적중률은 프로그램과 데이터의 지역성(locality)에 따라 달라짐

지역성(locality)

연세대학교

- 시간적 지역성(temporal locality)
  - 최근에 액세스된 프로그램이나 데이터가 가까운 미래에 다시 액세스 될 가능성이 높다
- 공간적 지역성(spatial locality)
  - 기억장치 내에 인접하여 저장되어 있는 데이터들이 연속적으로 액세스 될 가능성이 높다
- 순차적 지역성(sequential locality) :
  - 분기(branch)가 발생하지 않는 한, 명령어들은 기억장치에 저장된 순서대로 인출되어 실행된다
    (순차 실행 : 비순차 실행 = 5:1 정도)

캐시 설계에 있어서의 공통적인 목표

연세대학교

- 캐시 적중률의 극대화
- 캐시 액세스 시간의 최소화
- 캐시 미스에 따른 지연 시간의 최소화
- 주기억장치와 캐시 간의 데이터 일관성 유지 및 이에 따른 오버헤드의 최소화

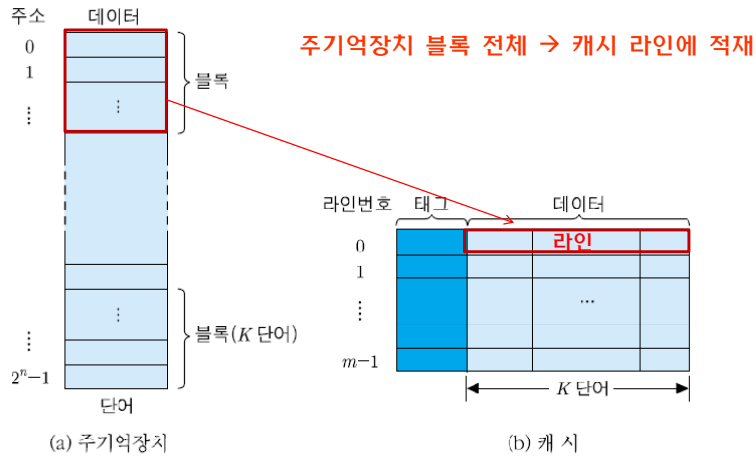
캐시의 크기 / 인출 방식

연세대학교

- 캐시의 크기(용량)
  - 용량이 커질수록 적중률이 높아지지만, 비용이 증가
  - 용량이 커질수록 주소 해독 및 정보 인출을 위한 주변 회로가 더 복잡해지기 때문에 액세스 시간이 다소 더 길어진다
- 인출 방식
  - 요구 인출(demand fetch) 방식
    - 필요한 정보만 인출해 오는 방법
  - 선인출(prefetch) 방식
    - 필요한 정보 외에 앞으로 필요할 것으로 예측되는 정보도 미리 인출 → 블록 단위 인출
    - 지역성이 높은 경우에 효과가 높다.

주기억장치와 캐시의 조직

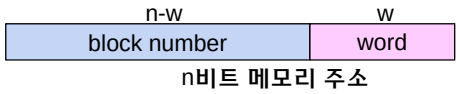
연세대학교



주기억장치와 캐시의 조직 (계속)

연세대학교

- 블록(block) : 주기억장치로부터 동시에 인출되는 정보들의 그룹
  - 주기억장치 용량 = 2^n 워드, 블록 = K 워드 ($K=2^w$)
    → 블록의 수 = 2^n / K 개 = 2^{n-w} 개
- 라인(line) : 캐시에서 각 블록이 저장되는 장소
- 태그(tag) : 라인에 적재된 블록을 구분해주는 정보
  - 캐시의 각 라인은 주기억장치의 여러 블록에 의해서 공유됨
  - 태그는 라인을 공유하는 블록들 중 현재 어느 블록이 라인에 적재 되었는지를 알려줌

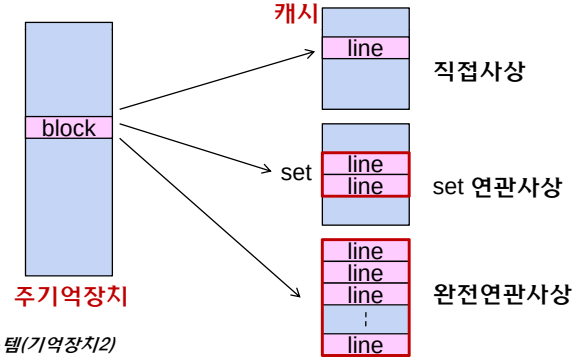


- 메모리 주소의 상위 n-w 비트 → 블록 번호
- 메모리 주소의 하위 w 비트 → 블록 내의 word 주소

사상(mapping) 방식

연세대학교

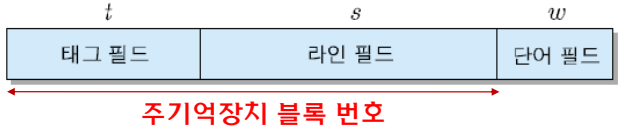
- 어떤 주기억장치 블록들이 어느 캐시 라인을 공유할 것인 지를 결정해 주는 방법
  - 직접 사상(direct mapping) : 1 라인 사용
  - 완전-연관 사상(fully-associative mapping): 모든 라인 공유
  - 세트-연관 사상(set-associative mapping) : 일부 라인 공유



직접 사상

연세대학교

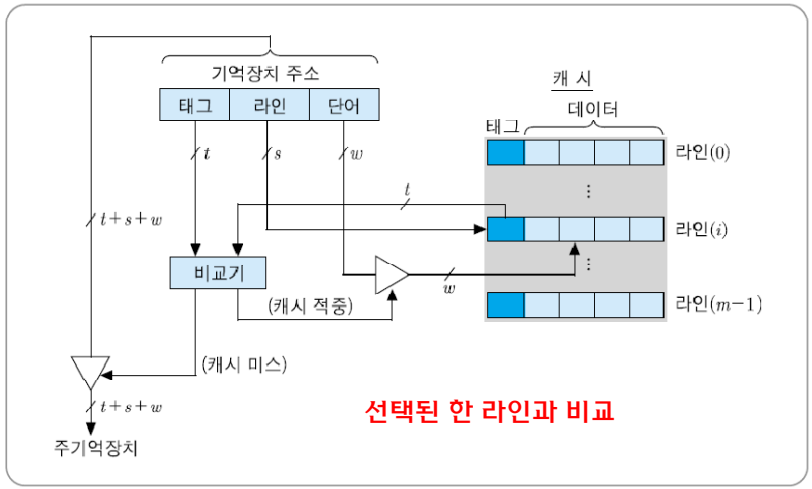
- 주기억장치의 블록들이 지정된 하나의 캐시 라인으로만 적재
- 주기억장치 주소 형식



- 캐시 메모리 크기: m = 2^s 워드, 블록 크기 = 2^w 워드
  - 단어 필드(w 비트) : 각 블록 내 2^w 개 단어들 중의 하나를 구분
  - 라인 필드(s 비트) : 블록 번호의 하위 s비트
    - 캐시의 m = 2^s 개의 라인들 중의 하나를 지정
  - 태그 필드(t 비트) : 블록 번호의 라인 필드를 제외한 상위 비트
    - 캐시 라인의 태그필드에 저장하여 적재된 캐시 라인의 주기억장치 블록번호 정보를 제공함

직접 사상 캐시의 조직

연세대학교



직접 사상 캐시의 동작 원리

연세대학교

- 캐시로 기억장치 주소가 보내지면,
  - 라인 번호(s비트)를 이용하여 캐시의 라인을 선택
- 선택된 라인의 태그 비트들을 읽어서 주소의 태그 비트들과 비교
  - 캐시 적중 (두 태그값이 일치하면)
    - 주소의 워드번호(w 비트)들을 이용하여 라인내의 워드들 중에서 하나를 선택하여 인출함
  - 캐시 미스 (태그값이 일치하지 않는다면)
    - 주소를 주기억장치로 보내어 요청한 워드를 포함한 블록을 액세스
    - 인출된 블록을 지정된 캐시 라인에 적재하고, 주소의 태그 비트들을 그 라인의 태그 필드에 기록
    - 만약 그 라인에 다른 블록이 이미 적재되어 있다면, 그 내용은 지워지고 새로이 인출된 블록을 적재

직접 사상 캐시의 장단점

연세대학교

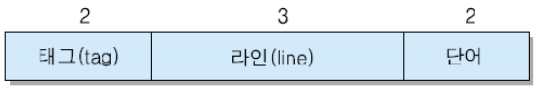
- 장점
  - 하드웨어가 간단하고, 구현 비용이 적게 든다
- 단점
  - 각 주기억장치 블록이 적재될 수 있는 캐시 라인이 한 개뿐이기 때문에, 그 라인을 공유하는 다른 블록이 적재되는 경우에는 swap-out 됨 → 캐시 적중률이 낮아짐

(예) 직접 사상 캐시의 예

연세대학교

- 주기억장치 용량 = 128(= 2^7) 바이트
  - 주기억장치 주소 = 7 비트 (바이트 단위 주소 지정)
- 블록 크기 = 4 바이트 (2^2) = 캐시라인 크기
  - 주기억장치의 블록 수 = 128/4 = 32 개 = 2^5
- 캐시 크기 = 32 바이트
  - 전체 캐시 라인 수
    m = 32/4 = 8개 = 2^3
  - 태그 비트수 : 5 - 3= 2

➔기억장치 주소 형식



직접 사상 캐시의 예(계속)

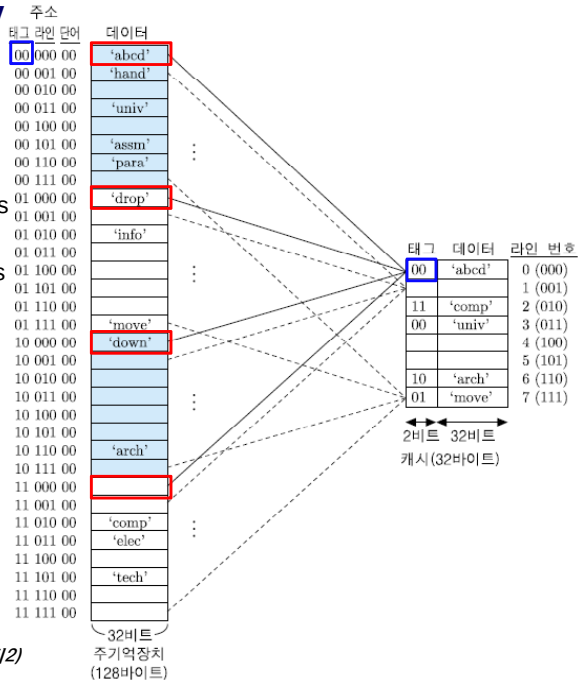
연세대학교

- 각 기억장치 블록이 공유하게 될 캐시 라인 번호
  블록 번호(5비트)의 하위 3비트

[표 5-5] 각 캐시 라인을 공유하는 주기억장치 블록들

캐시 라인	주기억장치 블록 번호			
0 (000)	00000	01000	10000	11000
1 (001)	00001	01001	10001	11001
2 (010)	00010	01010	10010	11010
3 (011)	00011	01011	10011	11011
4 (100)	00100	01100	10100	11100
5 (101)	00101	01101	10101	11101
6 (110)	00110	01110	10110	11110
7 (111)	00111	01111	10111	11111

직접 사상 캐시의 예(계속)



적중검사

(1) 01010100 → miss

(2) 00011100 → hit

(3) 11101100 → miss

(4) 10111000 → hit

완전-연관 사상

- 주기억장치 블록이 캐시의 어떤 라인으로든 적재 가능
- 기억장치 주소 형식

  - 태그 필드 = 주기억장치 블록 번호
  - 라인 필드는 없음 (라인을 선택할 필요가 없으므로)

- (예) 직접 사상 캐시의 예에 완전-연관 사상 방식을 적용하면,

➔기억장치 주소 형식

완전-연관 사상 캐시의 조직

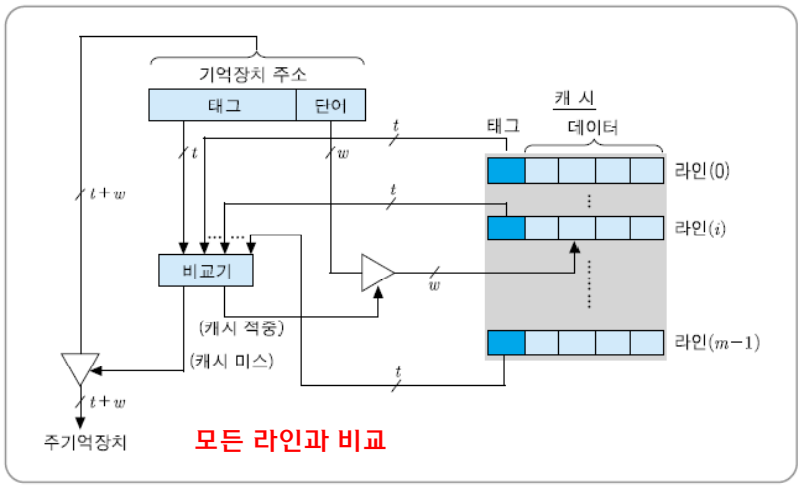
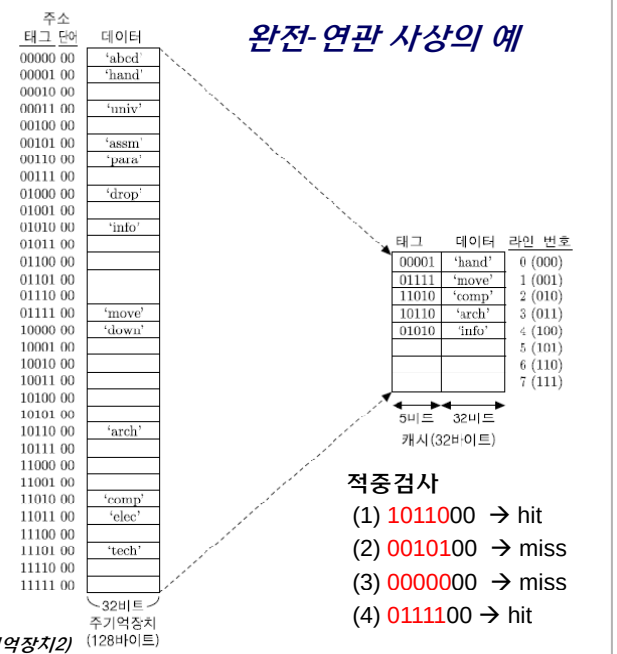


그림 5-20 완전-연관 사상 캐시의 조직

완전-연관 사상 캐시의 장단점

- 장점
  - 새로운 블록이 캐시로 적재될 때 라인의 선택이 매우 자유롭다
  - 지역성이 높다면, 적중률이 매우 높아진다
- 단점
  - 캐시 라인들의 태그들을 병렬로 검사하기 위하여 가격이 높은 연관 기억장치(associative memory) 및 복잡한 주변 회로가 필요

완전-연관 사상의 예



적중검사

(1) 1011000 → hit

(2) 0010100 → miss

(3) 0000000 → miss

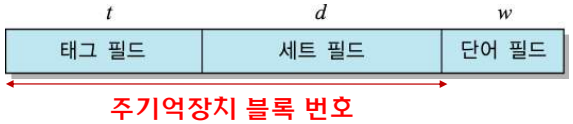
(4) 0111100 → hit

세트-연관 사상

- 캐시 세트
  - 여러 개(k개)의 캐시 라인으로 구성된 세트 (k=2, 4, 8 …)
- k-way 세트 연관 사상
  - 주기억장치 각 블록은 k개의 캐시 라인으로 구성된 하나의 캐시 세트를 공유함
  - 각 블록은 연관되는 캐쉬 세트에 속한 라인들 중 하나의 라인에 적재됨
- v개의 세트들로 구성되는 k-way 세트 연관 사상 캐시
  - 캐시 라인의 수: m = v x k
    - v : 캐시의 전체 세트 수 ($v = 2^d$)
    - k : 세트 내의 라인 수 ($k = 2^r$)
  - 세트 수 = 캐시 라인 수(m) / k

세트-연관 사상 (계속)

- 기억장치 주소 형식

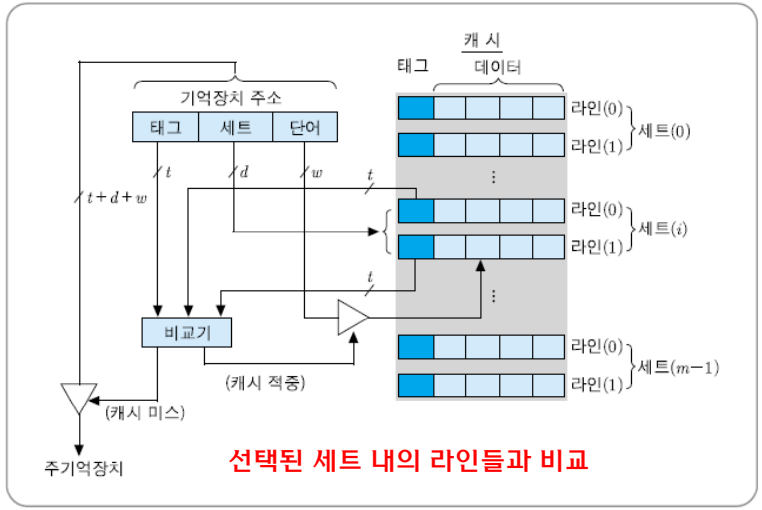


  - 세트 필드(d 비트): 블록 번호의 하위 d비트
  - 태그 필드(t 비트) : 블록 번호의 세트필드를 제외한 상위 비트
    - 캐시 라인의 태그필드에 저장하여 적재된 캐시 라인의 주기억장치 블록번호 정보를 제공함
- (예) 직접 사상 캐시의 예에 2-way 세트연관 사상 방식을 적용한 경우
  - 세트 수 = 8 라인 / 2 way = 4 세트 (= 2^2)
  - 태그 비트 수 = 5 – 2 = 3

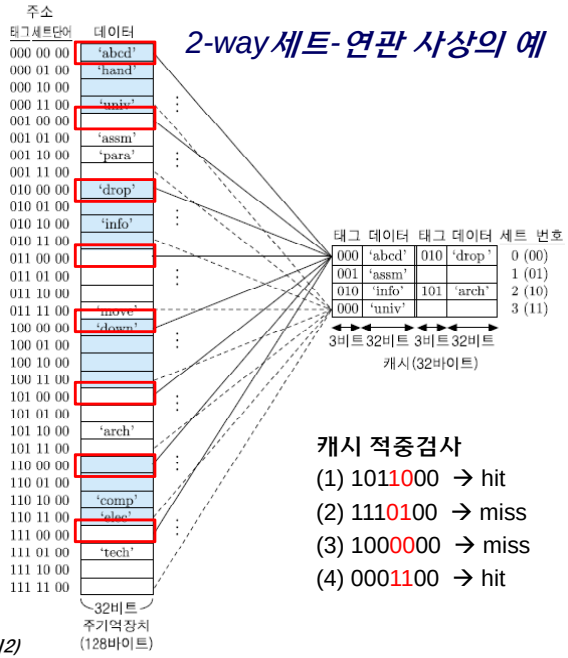
세트-연관 사상의 동작 원리

- 기억장치 주소의 세트 번호를 이용하여 캐시의 세트를 선택
- 주소의 태그 필드 내용과 선택된 세트 내의 태그들을 비교
  - 일치하는 것이 있으면 (캐시 적중)
    - 그 라인내의 한 단어를 w 비트에 의해 선택하여 인출
  - 일치하는 것이 없다면 (캐시 미스)
    - 주기억장치를 액세스
    - 세트 내의 라인들 중의 어느 라인에 새로운 블록을 적재할 것인 지를 결정하여 교체 → 교체 알고리즘 필요

2-way 세트-연관 사상 캐시의 조직



2-way세트-연관 사상의 예



캐시 적중검사

(1) 1011000 → hit
(2) 1110100 → miss
(3) 1000000 → miss
(4) 0001100 → hit

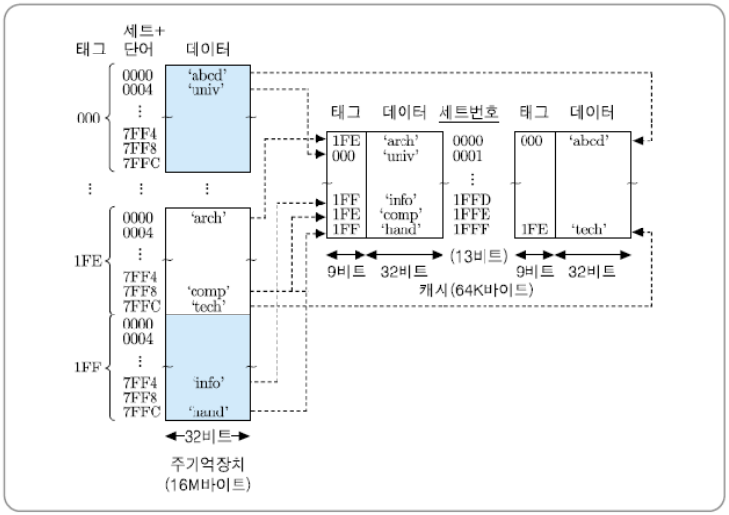
직접, 세트-연관, 완전-연관 사상 관계

- 직접 사상, 완전 연관 사상은 세트-연관 사상의 특수한 경우임
- k-way 세트-연관 사상에서
  - 직접 사상: k = 1 인 경우 (세트 수 = 캐시 라인 수)
  - 완전연관 사상: k = 캐시 전체 라인 수(m) 인 경우 (세트 수 = 1)

예: 큰 용량의 캐시

- 주기억장치의 용량 = 16MB (2^{24})바이트
- 캐시의 용량 = 64KB (2^{16})바이트
- 블록 크기 = 라인 크기 = 4B (2^2)
- (풀이)
  - 주기억장치 주소 = 24비트 (바이트 단위로 주소가 지정됨)
  - 주기억장치 블록 수 = 16M / 4 = 2^{24} / 2^2 = 2^{22} (4M) 개
  - 캐시 라인의 수 m = 64K / 4 = 2^{16} / 2^2 = 2^{14} (16K) 개
- 2-way 세트-연관 캐시의 주소 형식
  - 캐시의 세트 수 = m / k = 2^{14} / 2 = 2^{13}
  - 태그 비트 수 = 22 – 13 = 9

예: 큰 용량의 세트-연관 캐시의 조직



교체 알고리즘

- 세트-연관 사상 캐시에서
  - 주기억장치로부터 새로운 블록이 캐시로 적재될 때, 만약 세트 내 모든 라인들이 다른 블록들로 채워져 있다면, 그들 중의 하나를 선택하여 새로운 블록으로 교체
- 교체 알고리즘 :
  - 교체할 블록을 선택하기 위한 알고리즘
  - 캐시 적중률을 극대화할 수 있도록 알고리즘을 선택
  - 최소 최근 사용(Least Recently Used: LRU) 알고리즘이 가장 효과적임
    - 사용되지 않은 채로 가장 오래 있었던 블록을 교체하는 방식

쓰기 정책 (write policy)

- 캐시의 블록이 변경되었을 때 변경된 내용을 주기억장치에 갱신하는 시기와 방법의 결정
- Write-through 방법 : 모든 쓰기 동작들이 캐시로 뿐만 아니라 주기억장치로도 동시에 수행되는 방식
  - 장점: 캐시에 적재된 블록의 내용과 주기억장치에 있는 그 블록의 내용이 항상 같다
  - 단점: 모든 쓰기 동작이 주기억장치 쓰기를 포함하므로, 쓰기 시간이 길어지고, 쓰기 횟수가 많아진다.
- Write-back 방법: 캐시에서 데이터가 변경되어도 주기억장치에는 갱신되지 않는 방식
  - 장점: 기억장치에 대한 쓰기 동작의 횟수가 최소화되고, 쓰기 시간이 짧아진다
  - 단점: 캐시의 내용과 주기억장치의 해당 내용이 서로 다르다.
  - 블록을 교체할 때는 캐시의 수정상태를 확인하여 주기억장치를 갱신하는 동작이 선행되어야 함 → 상태비트(Modified) 필요

쓰기 정책 (계속)

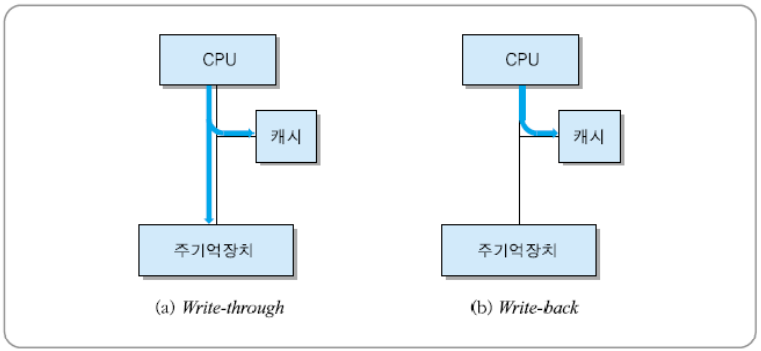


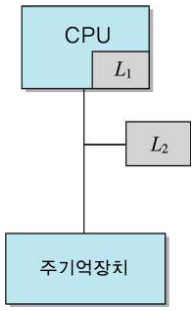
그림 5-26 쓰기 정책에 따른 쓰기 동작의 비교

다중 캐시(multiple cache)

연세대학교

- 온-칩 캐시(on-chip cache) :
  - 액세스 시간을 단축시키기 위하여 CPU 칩 내에 포함시킨 캐시
- 계층적 캐시(hierarchical cache)
  - 여러 단계로 설치된 캐시
  - 대개, 온-칩 캐시를 1차(L1) 캐시로 사용, CPU 외부에 더 큰 용량의 2차(L2) 캐시를 설치함
  - 최근에는 L2캐시도 온-칩 캐시로 구현됨
- 2단계 캐시 시스템의 평균 기억장치 액세스 시간

  Ta = H_1 x T_{L1} + (H_2 - H_1) x T_{L2} + (1 - H_2) x T_m
  - H_1 , H_2 : 주기억장치에 대한 L1, L2 캐시의 적중률



예제

연세대학교

예제 5-8

L_1, L_2 캐시 및 주기억장치의 액세스 시간이 각각 2ns, 10ns 및 100ns이고, L_1의 적중률(H_1)이 0.7, L_2의 적중률(H_2)은 0.9라고 할 때, 평균 기억장치 액세스 시간을 구하라. 단, 식 (5-6) 및 식 (5-7)을 이용하는 경우에 대하여 각각 답하라.

[풀이]

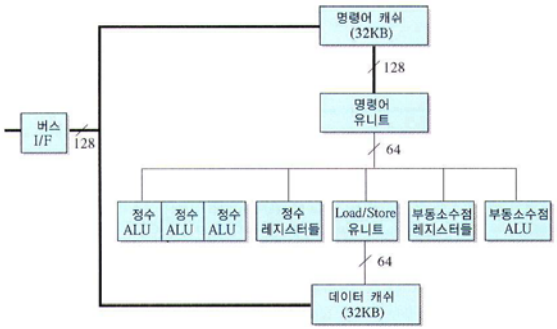
먼저, 식 (5-6)에서와 같이 H_2가 전체 기억장치 액세스들에 대한 L_2의 적중률이라면, 평균 기억장치 액세스 시간(Ta)은 아래와 같아진다.

$$Ta = (0.7\times 2ns) + \{(0.9-0.7)\times 10ns\} + \{(1-0.9)\times 100ns\}$$
$$= 1.4ns + 2ns + 10ns = 13.4ns$$

다중 캐시 - 분리 캐시 (split cache)

연세대학교

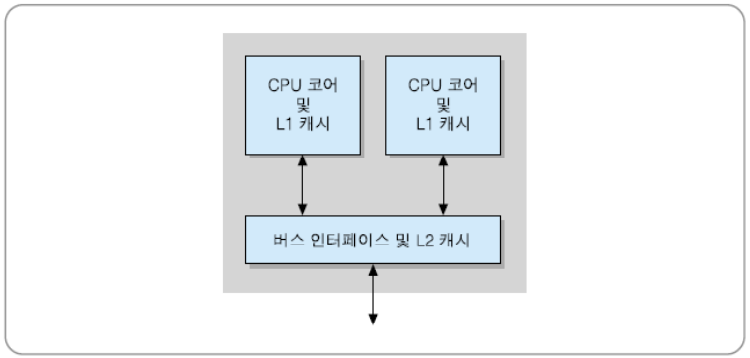
- 분리 캐시
  - 캐시를 명령어 캐시와 데이터 캐시로 분리
  - 명령어 인출 유니트와 실행 유니트 간에 캐시 액세스 충돌 현상 제거함
  - 대부분의 고속 프로세서들(Pentium 계열, PowerPC 등)에서 사용



듀얼-코어(dual-core) 프로세서의 캐시 구조

연세대학교

- 각 코어는 별도의 L1 캐시를 포함
- L2 캐시는 두 코어들에 의해 공유



5.6 최신 기억장치 기술

연세대학교

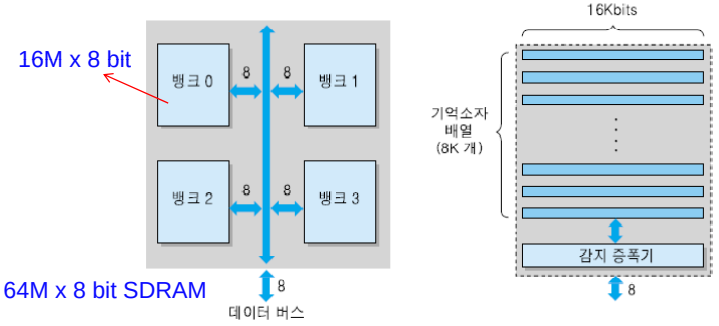
- 기억장치의 액세스 속도는 CPU에 비하여 현저히 낮음
- 동영상 편집, 음성/영상 압축과 같은 대규모 데이터 처리 응용 환경
  - 주기억장치 병목 현상 심화
  - 새로운 기억장치 유형 개발 시급

- 최신 기억장치
  - SDRAM
  - DDR SDRAM
  - PRAM (Phase-Change RAM)
  - FRAM (Ferroelectric RAM)
  - MRAM (Magnetic RAM)

SDRAM

연세대학교

- 동기식 DRAM(Synchronous DRAM: SDRAM) :
  - 액세스 동작들이 시스템 클록에 동기화 되어 수행되는 DRAM
- SDRAM의 내부 조직
  - 다수의 뱅크(bank)들로 구성, 뱅크 별로 동시 액세스 ← pipelining

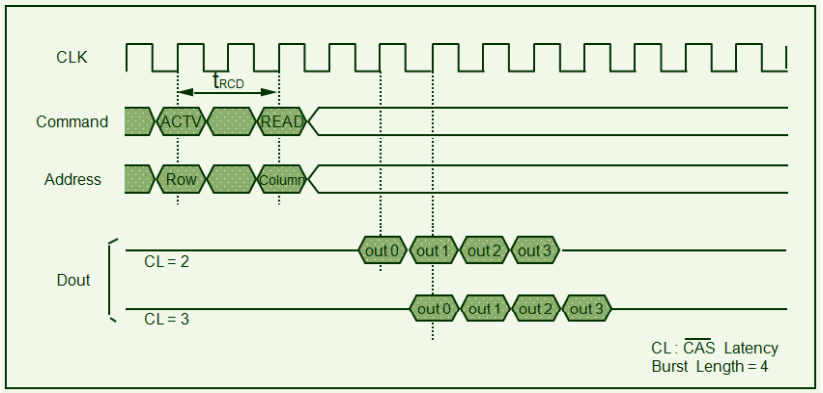


(a) 네 개의 뱅크들로 구성된 SDRAM　　(b) 각 뱅크의 내부 조직

SDRAM (계속)

연세대학교

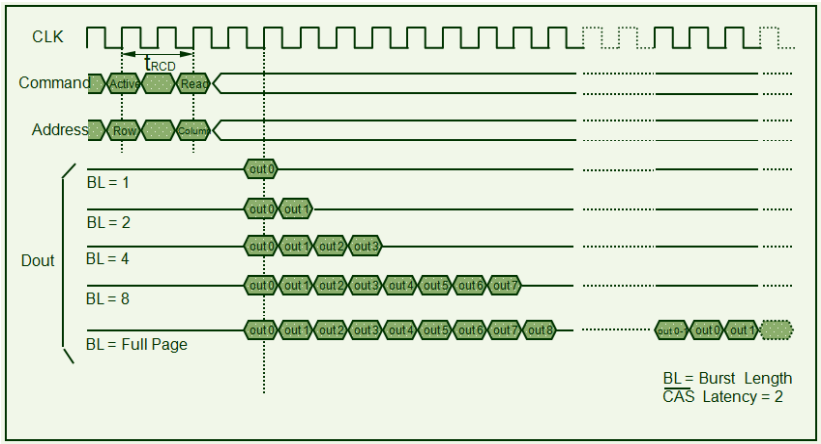
- 읽기 동작
  - CAS latency 후에 데이터 출력



SDRAM (계속)

연세대학교

- 읽기 동작 (Burst 모드)
  - 연속된 주소의 데이터를 연속하여 읽어서 출력

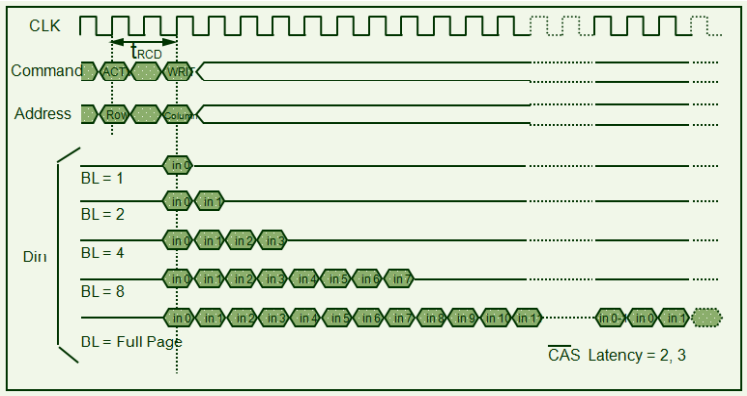


CL과 BL은 초기에 설정해야 함

SDRAM (계속)

연세대학교

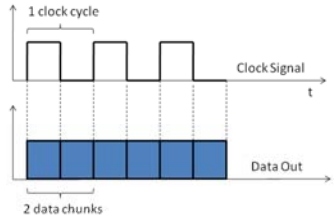
- 쓰기동작



DDR SDRAM

연세대학교

- 기억장치 대역폭(memory bandwidth)을 향상시키기 위한 기술
  - 대역폭: 단위 시간 당 전송되는 데이터 량, 단위: bps [bits/sec]
- DDR(double data rate) SDRAM : 버스 클록 당 두 번의 데이터 전송 (클록의 상승-에지 및 하강-에지에서 각각 전송)
  - [비교] SDRAM: SDR(single data rate) SDRAM이라 부름
- DDR2 SDRAM : DDR SDRAM과 같으며, 버스 클록 주파수만 두 배 높여 대역폭을 향상



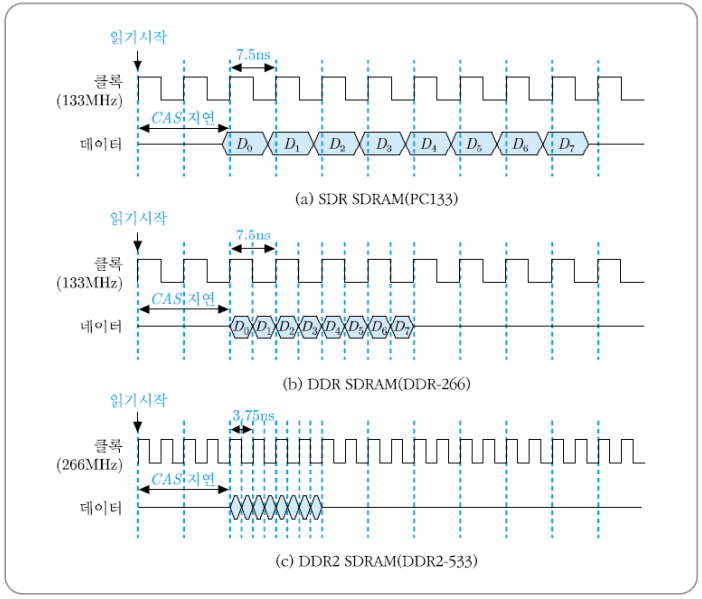


그림 5-34 DDR SDRAM 읽기 동작의 타이밍(버스트 길이 = 8)

SDRAM 명칭 부여 방법

연세대학교

- SDR SDRAM
  - PC66, PC100, PC133
    - 66, 100, 133 : clock 속도 (MHz)
- DDR SDRAM
  - DDRx-yyyy
    - x: technology generation
    - yyyy: DDR clock 속도 (사용 clock 속도의 2배) (MHz)

DDR SDRAM (계속)

연세대학교

- 읽기 시간 비교 (예: 버스트 길이 = 8)
  - SDR @ 133 MHz [PC133]
    - 매 클록 주기(7.5ns)마다 64비트씩 전송
    - 8번의 데이터 전송에 걸리는 시간
      = CAS 지연(15ns) + 7.5ns x 8 = 75ns
  - DDR @ 133 MHz [DDR-266]
    - 상승 및 하강 에지(3.75ns)마다 64비트씩 전송
    - 8번의 데이터 전송에 걸리는 시간
      = CAS 지연(15ns) + 3.75ns x 8 = 45ns

DDR SDRAM (계속)

연세대학교

- DDR 기술의 기본 원리
  - 버스 클록의 상승 에지 및 하강 에지에서 모두 데이터 전송
  - 기억장치 제어기(memory controller) 및 버스 인터페이스 회로의 개선을 통하여 버스 클록 주파수 향상
  - DDR3 및 DDR4도 같은 원리를 적용하여 설계

DDR SDRAM 모듈

연세대학교

- 기억장치 대역폭(memory bandwidth)
  - 단위 시간 당 데이터 전송량 = 버스 폭(bus width) x 클록 주파수

  [예] 데이터 버스 = 64비트, 버스 클록 주파수 = 100MHz
  ➔ 대역폭 : (100MHz x 64) / 8비트 = 800 [Mbytes/sec]
  - SDR@133MHz : (133MHz x 64) / 8 = 1064 [Mbytes/sec]
  - DDR@133MHz : (133MHz x 2 x 64) / 8 = 2128 [Mbytes/sec]
  - DDR2@266MHz : (266MHz x 2 x 64) / 8 = 4256 [Mbytes/sec]
- DDR SDRAM module의 명칭: **PCx-zzzz**
  - x: technology generation
  - zzzz: 이론적인 transfer rate (memory bandwidth) (MB/s)
  - DDR-266 => PC-2100
  - DDR2-533 => PC2-4200
  - DDR3-1066 => PC3-8500

연세대학교

- 전압

Technology	Typical Voltage
DDR	2.5 V
DDR2	1.8 V
DDR3	1.5 V

- Latency

Technology	Typical Latency	Other Common Latencies Available
DDR	3	2, 2.5
DDR2	5	3, 4
DDR3	7	6, 8, 9

차세대 비휘발성 기억장치

연세대학교

- PRAM(Phase-change), FRAM(Ferroelectric), MRAM(Magnetic)
  - **비휘발성**(nonvolatile)
  - **플래시 메모리에 비하여 액세스 속도가 1000배 가량 높음**
  - DRAM**보다 느리지만, 집적도는 비슷하며, 전력 소모가 더 낮음**

구분	PRAM	FRAM	MRAM
동작 원리	특정 물질의 상태 변화	강유전체의 분극 특성	전극의 자화 방향
쓰기 시간	500*ns*	50*ns*	10*ns*
장점	비휘발성, 고속, 고집적도	비휘발성, 고속, 저전력	비휘발성, 고속, 내구성
단점	느린 쓰기 속도	내구성 취약	상대적 고비용