가상 및 연관 기억장치

5 라인 크기

❖ 블록 크기에 따른 특성

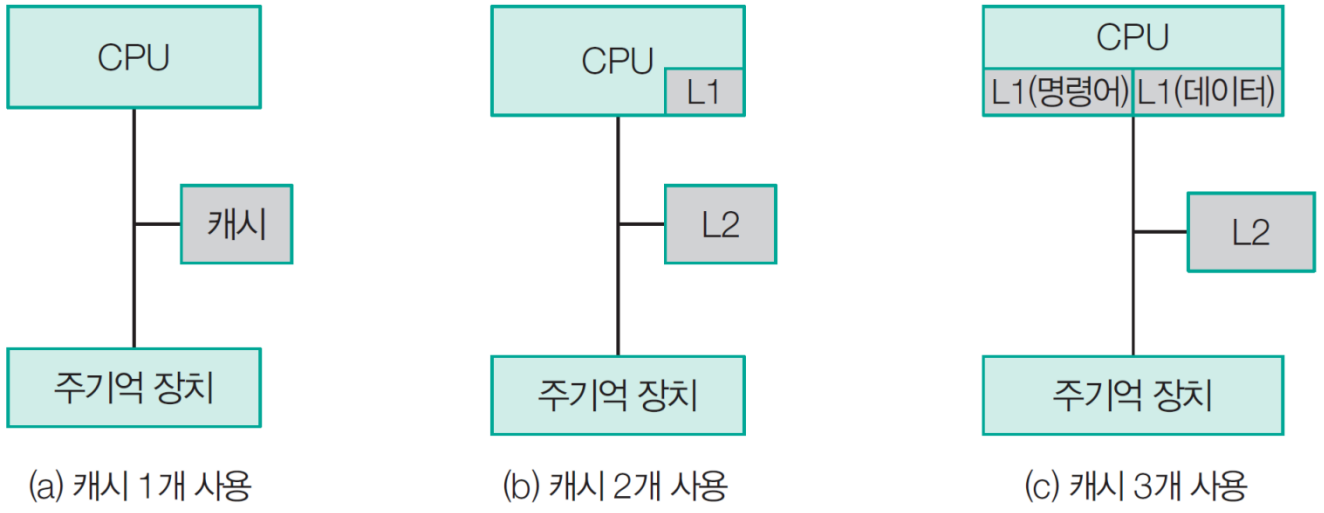
- 캐시 용량은 한정되어 있기 때문에 블록 크기가 커질수록 캐시에 들어올 수 있는 블록의 수는 줄어든다. 새로운 블록을 읽어 올 때마다 원래 캐시 내용은 교체되기 때문에 블록 수가 적으면 인출된 직후에 다른 블록과 다시 교체된다.
- 블록 크기가 커질수록 원하는 워드에서 멀리 떨어져 있는 워드들도 같이 읽혀 오며, 그들은 가까운 미래에 사용될 가능성은 낮다.

⇨ 대략적으로 블록 크기가 8~32바이트 정도가 최적에 가까운 것으로 알려져 있다.

6 캐시 수

❑ 계층적 캐시

- **온-칩 캐시**(on-chip cache) : 캐시 액세스 시간을 단축시키기 위하여 CPU 칩 내에 포함시킨 캐시 (그림의 L1)
- **계층적 캐시**(hierarchical cache) : 온-칩 캐시를 1차(L1) 캐시로 사용하고, CPU 외부에 더 큰 용량의 2차(L2) 캐시를 설치하는 방식
- **분리 캐시**(split cache) : 캐시를 명령어 캐시와 데이터 캐시로 분리



(a) 캐시 1개 사용 (b) 캐시 2개 사용 (c) 캐시 3개 사용

그림 6-35 다양한 계층적 캐시 구조

03 캐시 기억 장치

❖ 계층적 캐시에서의 히트율

- L2의 용량이 L1보다 크며, L1의 모든 내용이 L2에도 존재
- 먼저 L1을 검사하고, 만약 원하는 정보가 L1에 없다면 L2를 검사하며, L2에도 없는 경우에는 주기억 장치를 액세스
- L1은 속도가 빠르지만, 용량이 작기 때문에 L2 보다 히트율은 더 낮다.
- 평균 기억 장치 액세스 시간(T_a)

$$T_a = H_1 \times T_{L1} + (H_2 - H_1) \times T_{L2} + (1 - H_2) T_m$$

L1 캐시의 액세스 시간과 히트율 : T_{L1}, H_1
L2 캐시의 액세스 시간과 히트율 : T_{L2}, H_2
주기억 장치의 액세스 시간 : T_m
H_2 는 전체 기억 장치 액세스에 대한 L2의 히트율을 의미한다.

$$H_1+(H_2-H_1)+(1-H_2)=1$$

예제 6-12 L1 캐시, L2 캐시, 주기억 장치의 액세스 시간이 각각 2ns, 5ns, 50ns이고, H_1=0.7, H_2=0.9라고 할 때, 평균 기억 장치 액세스 시간을 구하여라.

풀이

$$T_a = H_1 \times T_{L1} + (H_2 - H_1) \times T_{L2} + (1 - H_2) T_m$$

$$T_a = 0.7 \times 2ns + (0.9 - 0.7) \times 5ns + (1 - 0.9) \times 50ns$$
$$= 1.4ns + 1ns + 5ns = 7.4ns$$

End of Example

03 캐시 기억 장치

❑ 통합 캐시와 분리 캐시

❖ 통합 캐시(unified cache)

- 온-칩 캐시가 처음 출현했을 때, 대부분은 명령어와 데이터를 하나의 캐시에 저장

❖ 분리 캐시(split cache)

- 캐시를 명령어 캐시와 데이터 캐시로 분리
- 명령어 인출 유니트와 실행 유니트 간의 캐시 액세스 충돌 제거
- 고성능 파이프 라인 프로세서들에서 사용

04 가상 기억 장치(virtual memory)

❖ 가상 기억 장치(virtual memory)

- 하드 디스크처럼 용량이 큰 보조 기억 장치를 주기억 장치처럼 사용하는 개념
- CPU가 참조하는 가상 주소를 주기억 장치의 실제 주소로 변환하는 주소 매핑이 필요하다.

❖ 주소 매핑(address mapping)

- 프로그래머가 프로그램에 표시한 주소를 가상 주소 또는 논리 주소라 하고, 이들의 주소 집합을 **주소 공간**이라 한다.
- 실제 프로그램이 적재되는 주기억 장치의 주소를 물리 주소라 하고, 이들의 집합을 **메모리 공간**이라 한다.

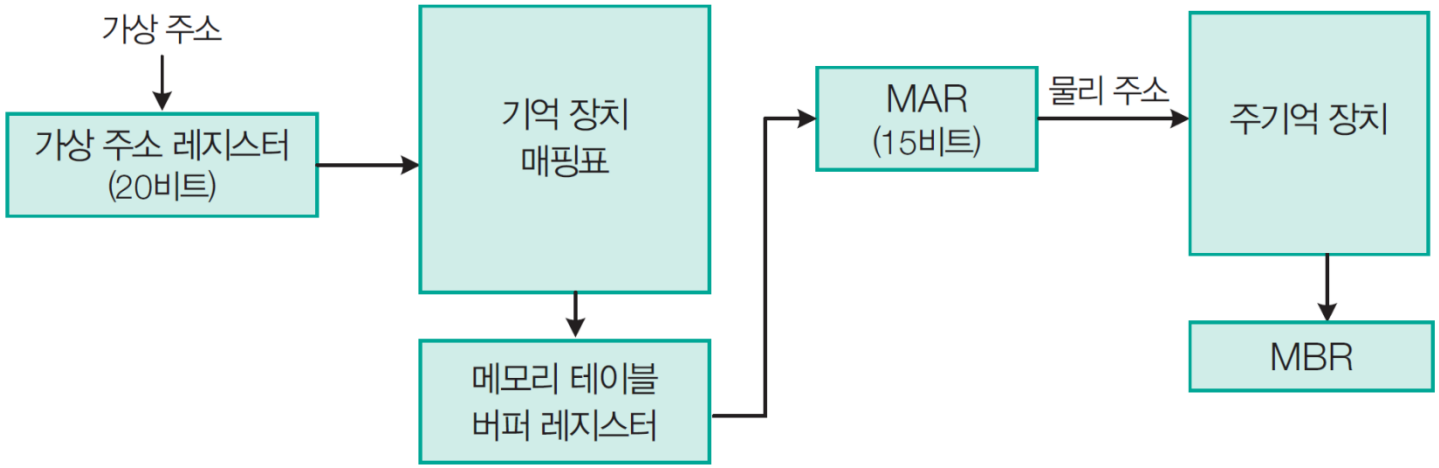


그림 6-36 가상 주소를 물리 주소로 변환하여 정보를 얻는 과정

1 가상 기억 장치의 매핑

❑ 페이지에 의한 매핑

- **페이지**(page) : 주소 공간을 고정 크기로 나눈 것
- **블록**(block) : 메모리 공간을 고정 크기로 나눈 것
- 페이지에 대한 기억 장치 매핑표를 가지고 페이지를 블록으로 변환한다.
- 가상 주소 페이지가 주기억 장치에 존재하지 않는 경우에는 **페이지 오류**(page fault)가 발생되었다고 하며, 이러한 페이지 오류가 자주 발생되는 경우를 **스래싱**(thrashing)이라고 한다.
- 주소 공간이 8K, 메모리 공간이 4K인 컴퓨터 시스템의 예(최대 페이지 4개가 메모리 공간의 블록 4개 중 하나에 들어간다.)

그림 6-37 주소 공간과 메모리 공간의 분리(1K)

❖ 기억 장치 매핑표

- 페이지 번호에서 블록 번호로의 변환만 하면 된다.
- 그림에서 페이지 1, 2, 5, 6은 주기억 장치 블록 2, 0, 1, 3에 각각 저장되어 있다.
- 현존 비트가 1이면 해당 페이지가 주기억 장치에 존재함을 의미

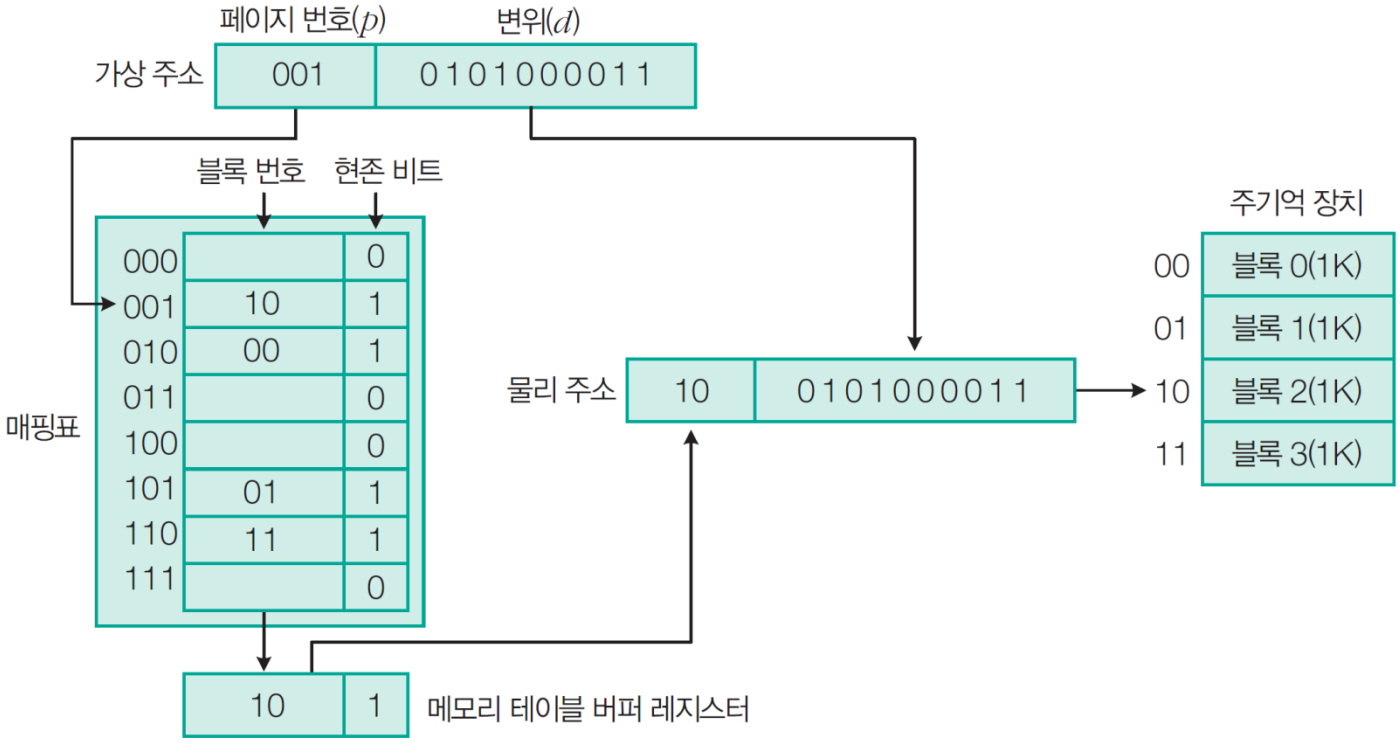


그림 6-38 기억 장치 매핑표 구조

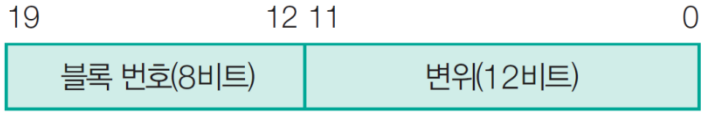
예제 6-13 용량이 1MB인 주기억 장치를 가진 컴퓨터 시스템에서 32비트의 가상 주소를 사용한다고 하자. 페이지 크기가 1K 워드고, 1워드는 4바이트라고 할 때 가상 주소와 물리 주소의 구성은 어떻게 될까?

풀이

한 페이지는 1K(=2^{10}) 워드고, 한 워드가 4(=2^{2})바이트이므로 한 페이지의 크기는 2^{12}이므로 변위는 12비트이어야 한다. 주기억 장치의 용량이 1M(=2^{20})바이트고, 한 페이지의 크기는 2^{12}바이트이므로 주기억 장치의 페이지 수는 256(=2^{8})개다.

$$(\text{주기억 장치의 용량})/(\text{한 페이지의 크기}) = 2^{20}/2^{12} = 2^{8}$$

따라서 물리 주소는 20비트로 구성되므로 페이지 주소에 8비트, 변위에 12비트가 할당된다.



가상 주소는 32비트이므로 페이지 번호에 20비트가 할당된다.

End of Example

❑ 연관 기억 장치를 이용한 기억 장치 매핑

- 앞 방법에서 기억 장치 매핑표는 기억 장치의 이용상 비효율적이다.
- 이를 위해 연관 기억 장치를 사용해 기억 장치 매핑표의 워드 수와 주기억 장치의 블록 수를 같게 한다.
- 가상 주소는 인자 레지스터로 전송되며, 키 레지스터의 마스크 값에 따라 이 레지스터의 페이지 번호 비트는 연관 기억 장치의 모든 페이지 번호 부분과 비교된다.
- 매치가 일어나지 않으면 운영체제에 의해 요구되는 페이지가 보조 기억 장치에서 주기억 장치로 옮겨진다.

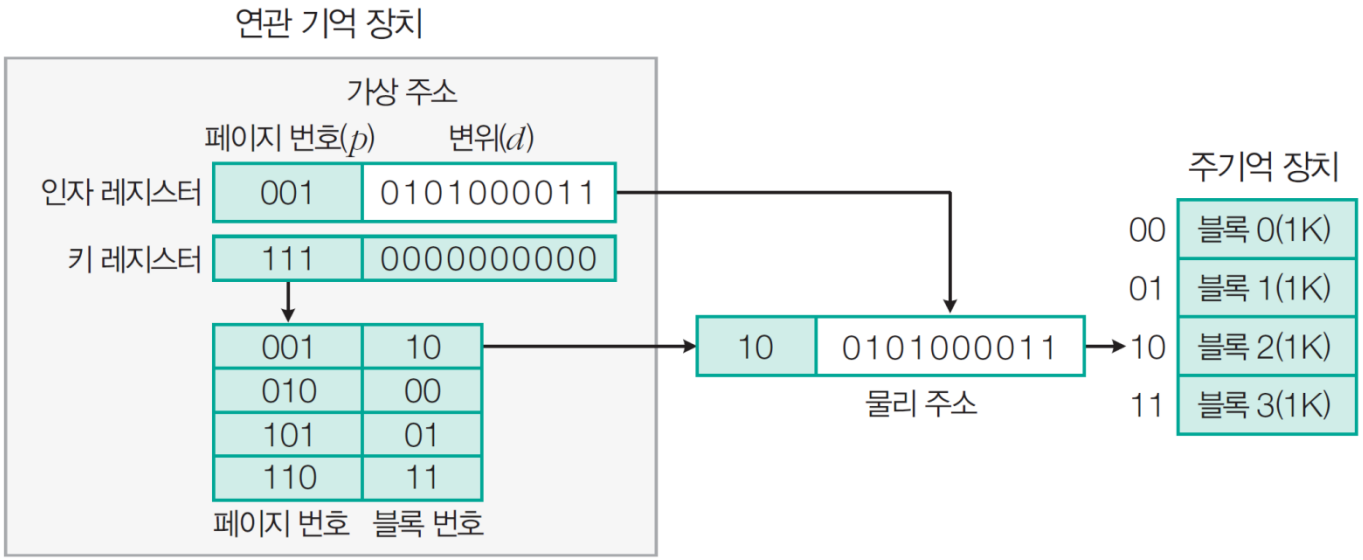


그림 6-39 연관 기억 장치를 이용한 매핑표 구조

❑ 세그먼트에 의한 매핑

- 고정 길이의 페이지가 아니라 가변 길이의 **세그먼트**로 매핑한다.
- 세그먼트로 된 프로그램에 의해 지정되는 주소를 논리 주소라고 한다.
- 프로그래머가 프로그램을 세그먼트화하고, 다시 시스템이 각 세그먼트를 페이지화한다.
- 세그먼트의 논리 주소에서 물리 주소로 매핑은 기억 장치 매핑표와 비슷한 세그먼트 테이블을 참조해서 이루어진다.
- 속도 향상을 위해 세그먼트 표나 페이지 표를 접근 시간이 빠른 연관 기억 장치에 저장 한다.

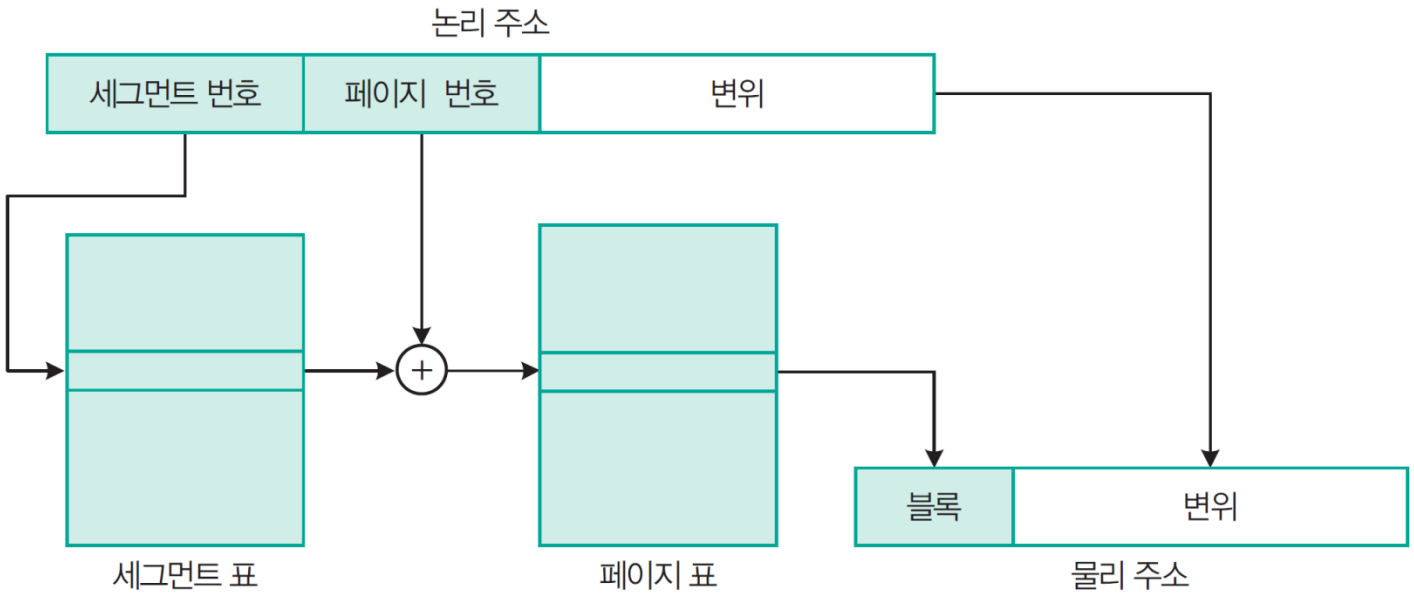


그림 6-40 세그먼트에 의한 매핑에서 논리 주소를 물리 주소로 매핑

2 페이지 교체 알고리즘

- 참조하고자 하는 페이지가 주기억 장치에 없을 경우 **페이지 오류**(page fault)라 하고, 이 조건이 발생하면 요구된 페이지가 주기억 장치로 옮겨질 때까지 프로그램 수행이 중단된다.
- 보조 기억 장치에서 주기억 장치로 페이지를 전송하는 것은 입출력 동작이므로 운영체제는 이 일을 I/O 프로세서에 맡긴다.
- **페이지 교체 알고리즘** : 새로운 페이지가 주기억 장치로 전송될 때, 주기억 장치가 꽉 차 있으면 제거할 페이지를 선택(FIFO, LRU, LFU 알고리즘 등)

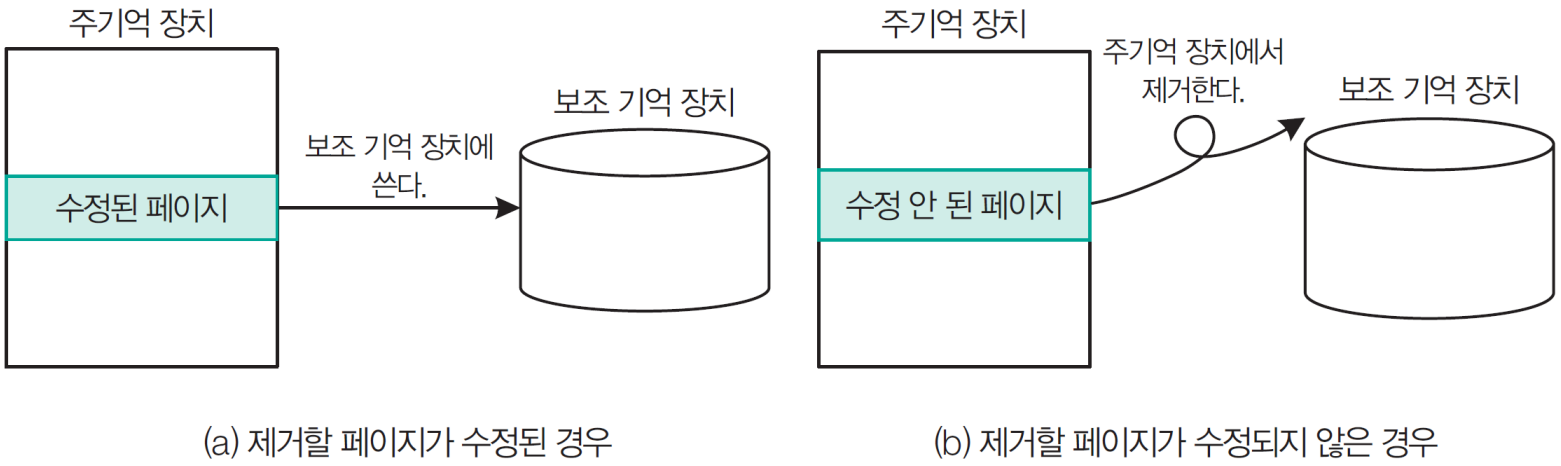


그림 6-41 페이지 교체할 때 동작

- 다음 예를 통해 각 알고리즘을 살펴보자.

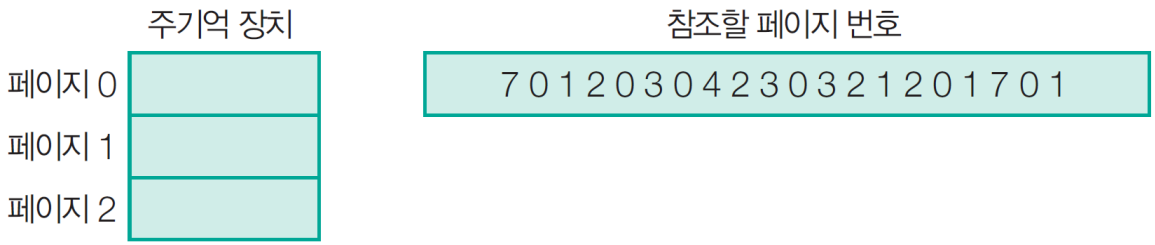


그림 6-42 페이지 교체 알고리즘 동작 예

❖ FIFO(First-In First Out) 알고리즘

- 주기억 장치에 가장 오래 있었던 페이지를 교체
- 구현하기가 쉽다는 장점이 있으나, 어떤 상황에서는 페이지가 너무 자주 교체되는 단점이 있다.

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	0	0	0	0	7	7	7
	0	0	0	0	3	3	3	2	2	2	2	2	1	1	1	1	1	0	0
		1	1	1	1	0	0	0	3	3	3	3	3	2	2	2	2	2	1

페이지 폴트 15회 발생

❖ LRU(Least Recently Used) 알고리즘

- 최근까지 가장 오랫동안 사용되지 않았던 페이지를 선택하여 제거하는 방법

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	4	0	0	0	1	1	1	1	1	1	1
	0	0	0	0	0	0	0	0	3	3	3	3	3	3	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	2	2	2	2	7	7	7

페이지 폴트 12회 발생

❖ LFU(Least Frequently Used) 알고리즘

- 사용 빈도가 가장 낮은 페이지를 선택해서 제거하는 방법

7	0	1	2	0	3	0	4	2	3	0	3	2	1	2	0	1	7	0	1
7	7	7	2	2	2	2	4	4	3	3	3	3	3	3	3	3	3	3	3
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
		1	1	1	3	3	3	2	2	2	2	2	1	2	2	1	7	7	1

페이지 폴트 13회 발생

❖ NUR(Not Used Recently) 알고리즘

- LRU와 비슷한 알고리즘으로 참조 비트와 변형 비트를 사용하여 최근에 사용하지 않은 페이지를 교체하는 방법

표 6-10 참조 비트와 변형 비트에 따른 교체 우선순위

참조 비트	변형 비트	교체 순서
0	0	1
0	1	2
1	0	3
1	1	4

참조비트는 최근 사용여부 확인 위해 해당 페이지가 엑세스될 때마다 세트되고 주기적으로 리셋

페이지가 수정되면 변형비트가 세트됨

04 가상 기억 장치

❖ 캐시 메모리, 주기억 장치, 가상 기억 장치 간의 데이터 이동

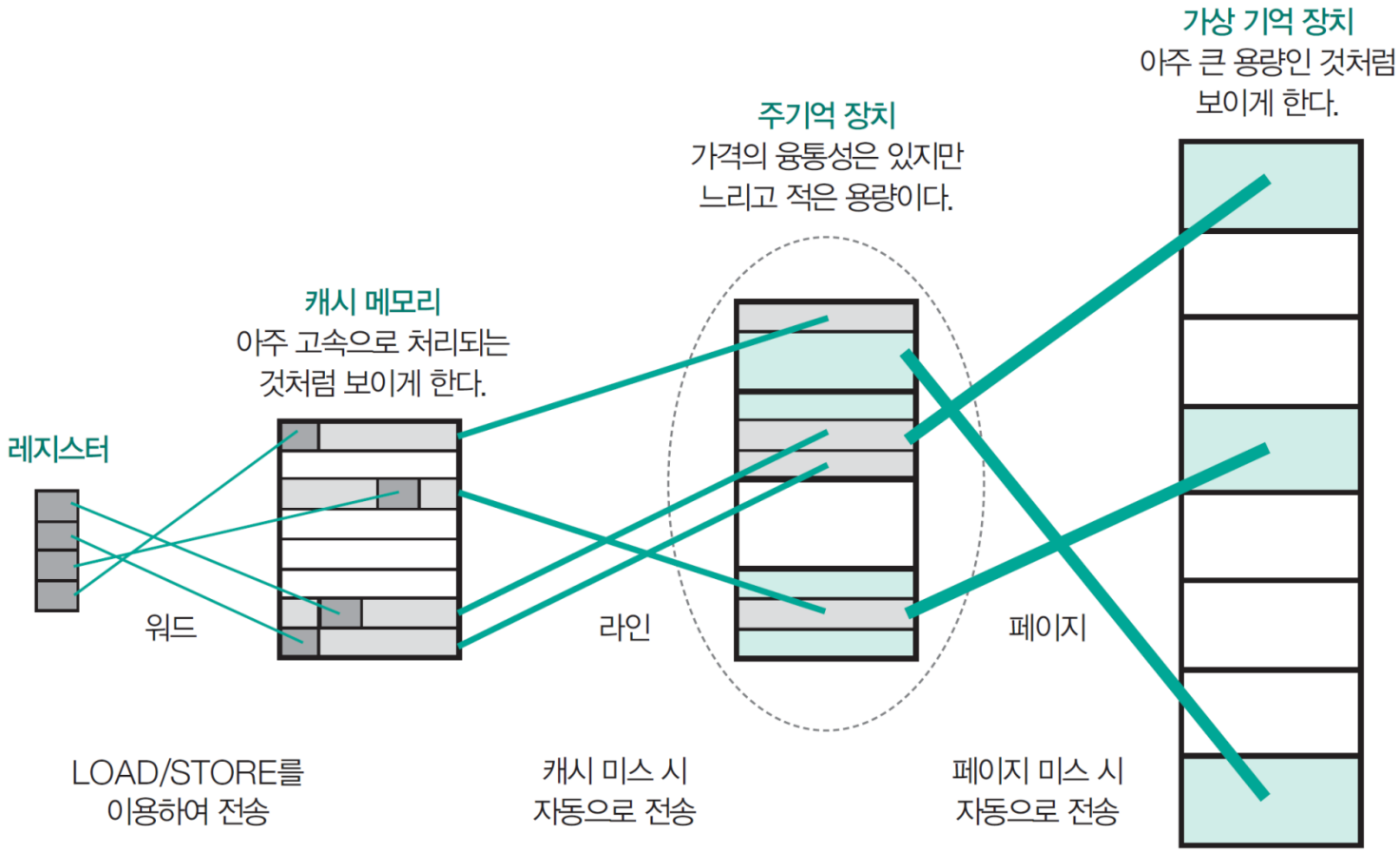


그림 6-43 메모리 계층 구조에서의 데이터 이동

05 연관 기억 장치(associative memory)

- **연관 기억 장치**(associative memory)는 메모리에 저장된 내용의 일부분을 이용해 원하는 정보가 저장된 위치를 알아낸다.
- CAM(Content Addressable Memory) 또는 병렬 탐색 기억 장치라고도 한다.
- 기억 장치의 모든 단자를 동시에 읽어 주어진 특성과 비교하므로 주소로만 접근할 때보다 훨씬 빠르다.
- 하지만 비교 회로 추가로 비싸져 탐색 시간이 중요하고 빠른 처리에만 주로 사용된다.

❖ 연관 기억 장치의 구조

- 인자 레지스터는 n비트, 키 레지스터는 n비트, 매치 레지스터는 m비트로 구성된다.
- 인자 레지스터에는 검색하려는 정보를 저장하고, 키 레지스터는 인자 레지스터의 특정 영역에 대한 마스크(mask)를 제공한다.

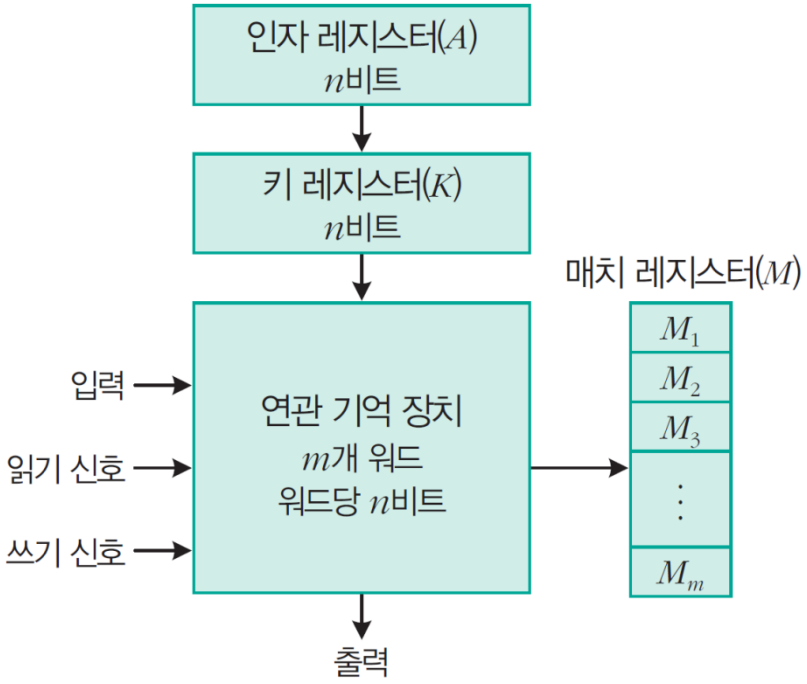


그림 6-44 연관 기억 장치의 구조

인자 레지스터(A)	101	11110	
키 레지스터(K)	111	00000	
워드 1	101	00100	일치 M_1=1
워드 2	111	11011	불일치 M_2=0
워드 3	010	01101	불일치 M_3=0
…		…	…
워드 m	100	11010	불일치 M_m=0

워드 1이 101이므로 매치 레지스터의 비트 M_1=1이다.

❖ 메모리 배열과 외부 레지스터의 관계

- C_{ij} : i워드 내의 j번째 셀
- 인자 레지스터 A_j비트는 K_j=1이라면 j 번째 칸의 모든 비트와 비교된다.
- 마스크되지 않은 모든 인자의 비트와 i워드 비트들의 매치가 이루어지면 매치 레지스터의 대응하는 비트 M_i=1이 된다.

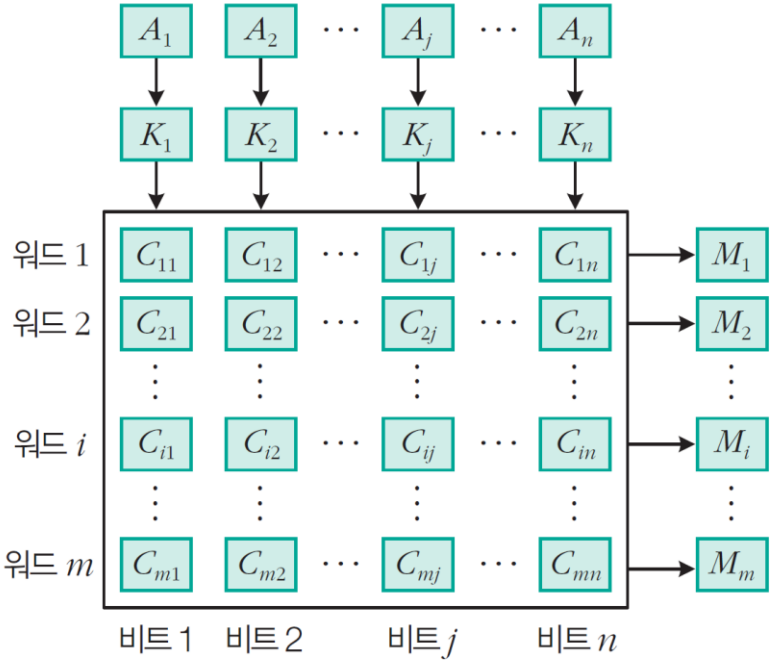


그림 6-45 워드당 n개 셀을 가진 워드 m개의 연관 기억 장치

❖ 전형적인 셀 C_{ij}의 내부 구조

- 매치 논리 회로는 셀의 내용과 마스크되지 않은 인자의 비트를 비교하여 M_i를 세트 또는 리셋시킨다.

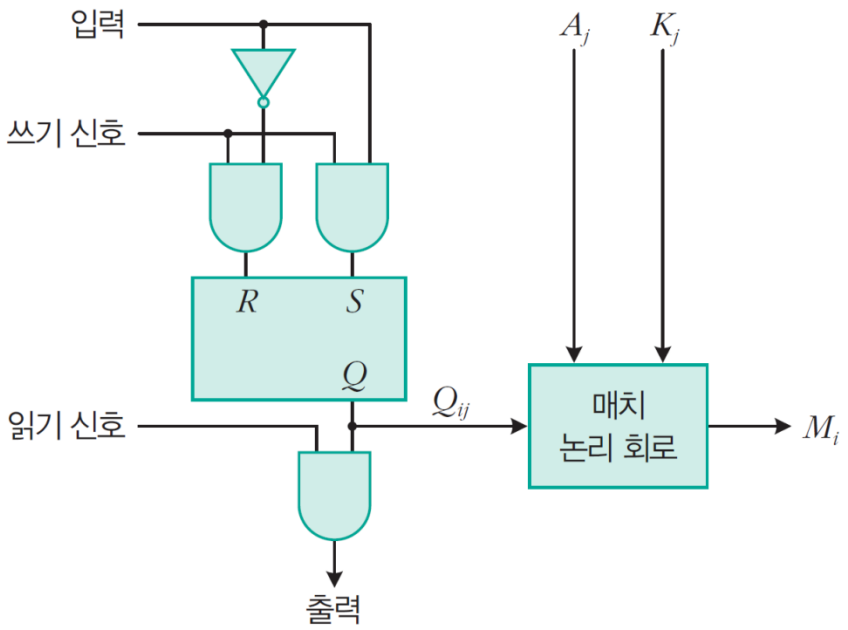


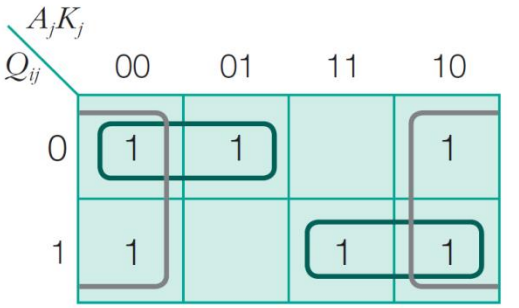
그림 6-46 연관 기억 장치의 셀 구조

❖ 매치 논리 회로 구조

- j =1, 2, ..., n에 대하여 K_j= 0이면 마스크된 경우이므로 M_j는 1이고, K_j=1이면 마스크되지 않은 경우이므로 Q_{ij} 와 A_j 를 비교하여 같으면 M_j=1, 다르면 M_j=0이다.

Q_{ij}	A_j	K_j	M_j
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	0
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

(a) 진리표



(b) 카르노 맵

그림 6-47 매치 논리 회로의 간소화

- 연관 기억 장치 내의 워드 i를 위한 매치 논리 회로의 불 대수식

$$M_i = (A_1Q_{i1} + \overline{A}_1\overline{Q}_{i1} + \overline{K}_1)(A_2Q_{i2} + \overline{A}_2\overline{Q}_{i2} + \overline{K}_2)\cdots(A_nQ_{in} + \overline{A}_n\overline{Q}_{in} + \overline{K}_n)$$

$$= \prod_{j=1}^{n}(A_jQ_{ij} + \overline{A}_j\overline{Q}_{ij} + \overline{K}_j)$$

❖ 한 워드에 대한 매치 논리 회로

- 매치가 일어나면 M_i=1이 되며, 그렇지 않으면 M_i=0이 된다.

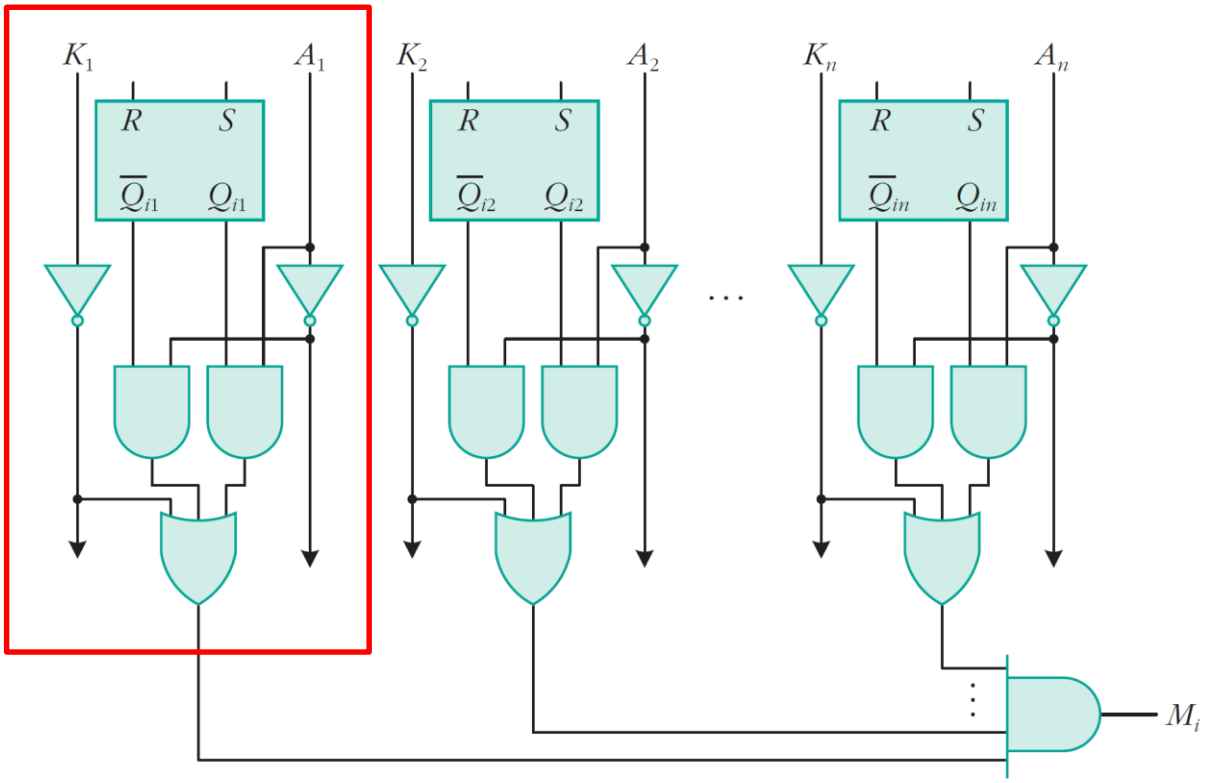


그림 6-48 연관 기억 장치의 한 워드에 대한 매치 논리 회로

수고하셨습니다!