Slide 0

12장. 메모리 구조

메모리 계층

- 메모리의 분류
  - 주기억 장치 : 보통 DRAM 으로 구성되며 CPU 가 실행할 프로그램 저장
  - 보조기억 장치 : 보통 자기디스크, 자기 테이프등이 있음
- 메모리의 계층 (그림 12-1)

Slide 1

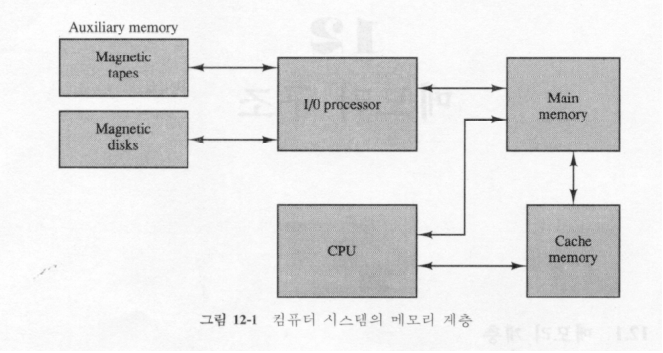


그림 12-1 컴퓨터 시스템의 메모리 계층

  - 속도가 느리나 대용량의 보조 메모리 (보조 기억 장치)

메모리 계층 (계속)

Slide 2

    * CPU 가 사용하지 않는 프로그램이나 데이타 저장
    * 전원이 꺼져도 데이타는 지워지지 않음
  - 속도가 빠르나 소용량의 메인 메모리 (주기억 장치)
    * CPU 가 사용하는 프로그램 저장
    * 전원이 꺼지면 데이타는 지워짐
  - 초고속 초소용량의 버퍼 메모리 (캐쉬)
    * CPU 가 사용하는 프로그램 중 일부를 저장
    * 전원이 꺼지면 데이타는 지워짐
    * 캐쉬에 비하여 보통 주 기억장치는 7배 보조기억장치는 1000배의 접근 속도를 가짐
- 멀티 프로그래밍 (multiprogramming)
  - 독립적인 여러개의 프로그램을 동시에 수행
  - 메모리 관린 시스템이 필요 →MMU (memory management unit)

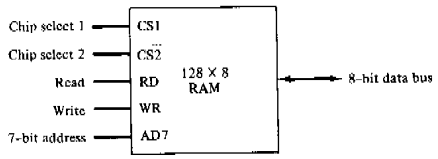
주기억 장치

Slide 3

- 메모리의 종류
  - SRAM (static RAM)
    * 플립플롭에 정보저장 →비교적 소 용량
    * 전원이 꺼지지 않는이상 정보 유지
    * 빠른 접근 속도
  - DRAM (dynamic RAM)
    * 콘덴서에 정보 저장 →비교적 대 용량
    * 방전하기 때문에 주기적으로 재 충전 필요
    * 느린 접근 속도
  - ROM (read only memory)
    * 하드웨어적으로 정보 저장
    * 전원이 꺼져도 정보 유지
    * 초기화 프로그램 저장
- RAM 과 ROM 칩
  - RAM 칩의 블럭도 (그림 12-2)

주기억 장치 (계속)

Slide 4



(a) Block diagram

CS1	$\overline{CS2}$	RD	WR	Memory function	State of data bus
0	0	×	×	Inhibit	High-impedance
0	1	×	×	Inhibit	High-impedance
1	0	0	0	Inhibit	High-impedance
1	0	0	1	Write	Input data to RAM
1	0	1	×	Read	Output data from RAM
1	1	×	×	Inhibit	High-impedance

(b) Function table

그림 12-2 전형적인 RAM 칩

- ROM 칩의 블럭도 (그림 12-3)

주기억 장치 (계속)

Slide 5

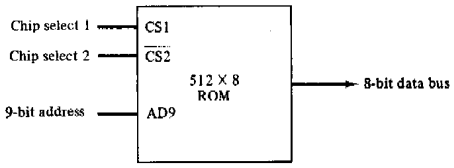


그림 12-3 전형적인 ROM 칩

- 메모리 주소 맵
  - 메모리 소자 별로 할당된 주소 표 (표 12-1)

주기억 장치 (계속)

Slide 6

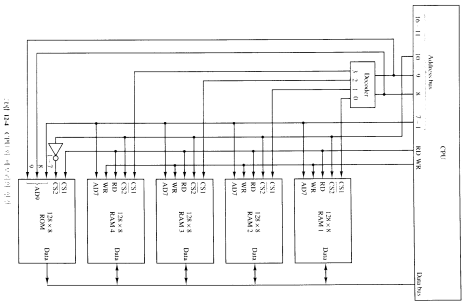
표 12-1 마이크로 컴퓨터의 메모리 주소 맵

Component	Hexadecimal address	Address bus 10	9	8	7	6	5	4	3	2	1
RAM 1	0000–007F	0	0	0	x	x	x	x	x	x	x
RAM 2	0080–00FF	0	0	1	x	x	x	x	x	x	x
RAM 3	0100–017F	0	1	0	x	x	x	x	x	x	x
RAM 4	0180–01FF	0	1	1	x	x	x	x	x	x	x
ROM	0200–03FF	1	x	x	x	x	x	x	x	x	x

- CPU 로의 메모리 연결
  - 메모리 주소 맵에 근거한 CPU 와의 연결 (그림 12-4)

주기억 장치 (계속)

Slide 7



보조 기억 장치

Slide 8

- 자기 디스크
  - 자기 디스크의 구조 (그림 12-5)

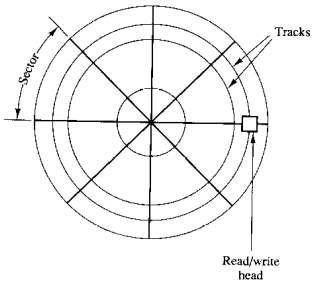


그림 12-5 자기 디스크

  - 자기 디스크의 종류
    - 하드 디스크 (hard disk)
    - 플로피 디스크 (floppy disk) →3.5 인치 5.25인치

어소시어티브 메모리

Slide 9

- 어소시어티브 메모리의 필요성
  - 많은 데이타 처리는 메모리에 저장되 표의 탐색이 필요
  - 각 주소의 메모리의 내용을 읽어서 해당 항목 찾기 →시간이 많이 필요
  - 메모리 구성 자체를 주소가 아니라 항목 내용으로 찾으면 빠르게 동작
  - 메모리 내용에 의하여 접근 되는 메모리
    →어소시어티브 메모리 (associative memory) 또는
    →내용 지정 메모리 (content addressable memory) 로 부름
  - 비교회로가 들어감으로 RAM 보다 비쌈
- 하드웨어 구성
  - 어소시어티브 메모리 구성 (그림 12-6)

어소시어티브 메모리 (계속)

Slide 10

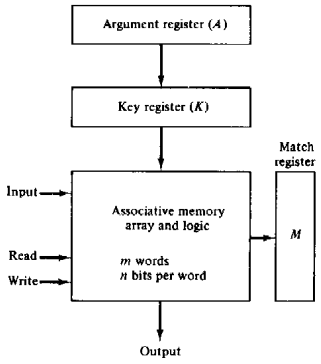


그림 12-6 어소시어티브 메모리의 블럭도

- Argument register (A) →찾을 데이타 저장
- Key register (K) →마스크 지정
- Match register (M) →매칭된 워드 표시
- 예)

어소시어티브 메모리 (계속)

Slide 11

A	101 111100	
K	111 000000	
워드 1	100 111100	no match
워드 2	101 000001	match

- 어소시어티브 메모리 구성 (그림 12-7)

어소시어티브 메모리 (계속)

Slide 12

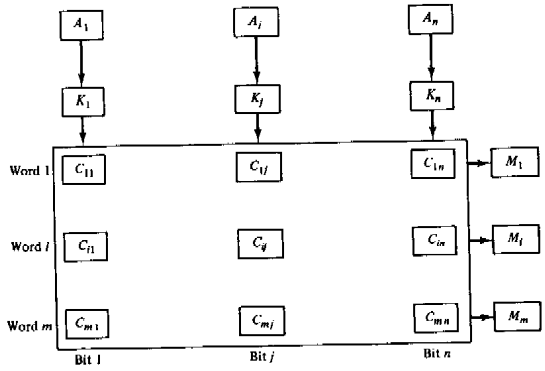


그림 12-7 워드당 n개 셀을 가진 m개 워드의 어소시어티브 메모리

- 셀의 내부구조 (그림 12-8)

어소시어티브 메모리 (계속)

Slide 13

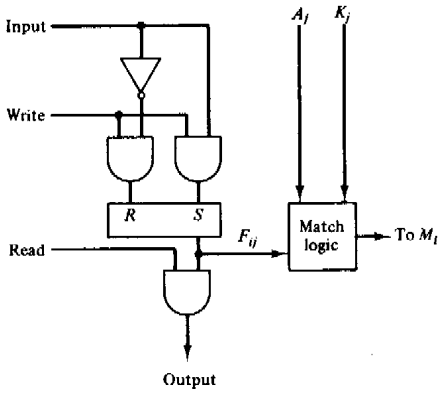


그림 12-8 어소시어티브 메모리의 한 셀

- 매치 논리
  - A_j 와 F_{ij} 가 같을 조건 $\rightarrow \boxed{x_j = A_j F_{ij} + A_j^{'} F_{ij}^{'}}$

어소시어티브 메모리 (계속)

Slide 14

- M_i 를 1 로 하는 조건 $\rightarrow \boxed{M_i = x_1 x_2 x_3 \cdots x_n}$
- K_j 를 고려 하면 $\rightarrow$

$$x_j + K_j^{'} = \begin{cases} x_j & \text{if } K_j = 1 \\ 1 & \text{if } K_j = 0 \end{cases}$$

- 결국 $\boxed{M_i = (x_1 + K_1^{'})(x_2 + K_2^{'}) \cdots (x_n + K_n^{'})}$
- 또다른 표현 $\boxed{M_i = \Pi_{j=1}^{n}(A_j F_{ij} + A_j^{'} F_{ij}^{'} + K_j^{'})}$
- 한 워드의 매칭 회로 (그림 12-9)

어소시어티브 메모리 (계속)

Slide 15

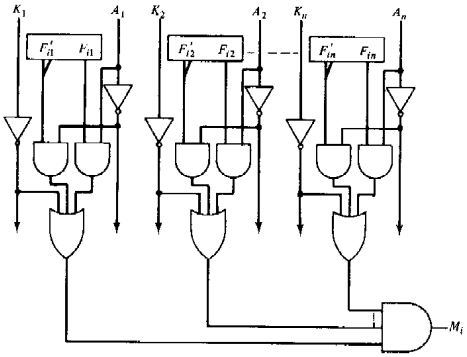


그림 12-9 어소시어티브 메모리의 한 워드에 대한 매칭 논리

- 읽기 동작
  - 매칭 레지스터 비트가 1로 세팅되는 워드를 읽음

어소시어티브 메모리 (계속)

  - 한표에는 하나의 매칭만 되게 표를 구성
  - matching 비트를 해당 워드의 read 신호로 연결 읽음
  - 모두 0 인 워드 출력 →매칭되는게 없음

- 쓰기 동작

Slide 16

  - 내용 갱신시 필요
  - $m = 2^d$ 임으로 d개의 주소 비트 필요 (m 은 워드의 수)
  - m 비트의 tag 레지스터를 사용 활성워드 및 비활성워드 구별
  - 워드 삭제 tag 레지스터의 해당 비트를 0 으로
  - 워드 삽입 tag 레지스터의 0 인 비트를 찾아 해당 워드에 삽입
  - 매칭 비교시 tag 레지스터의 상태도 필요

캐쉬 메모리

- 캐쉬 메모리의 필요성
  - 프로그램은 국한된 영역을 자주 액세스하는 경향이 있다 →locality of reference
  - 자주 액세스 되는 프로그램이나 데이타를 소용량의 빠른 메모리에 저장 →캐쉬 메모리
- 가상 메모리 (virtual memory) 와 캐쉬

Slide 17

  - 가상 메모리
    * 보조기억장치와 주기억장치 사이에 존재
    * multiprocessing 보조
    * 하드웨어 (MMU) 와 소프트웨어 (OS) 의 조합으로 구성
  - 캐쉬 메모리
    * 주기억장치와 CPU 사이에 존재
    * 빠른 액세스를 위함
    * 고속을 위하여 하드웨어제어기로만 구성
- 캐쉬의 특성

캐쉬 메모리 (계속)

Slide 18

- 캐쉬의 성능
  - 히트(미스) →찾고자하는 항목이 캐쉬에 있을때 (없을때)
  - 히트율 →캐쉬에 항목이 있는 총수/메모리 액세스 총수
  - 보통 히트율은 0.9
- 캐쉬의 매핑 →주 메모리의 항목을 캐쉬에 전송하는것
- 매핑의 3가지 방법
  1. 어소시어티브 매핑 (associative mapping)
  2. 직접 매핑 (direct mapping)
  3. 세트-어소시어티브 매핑 (set-associative mapping)

- 어소시어티브 매핑
  - 구성 (그림 12-11)

캐쉬 메모리 (계속)

Slide 19

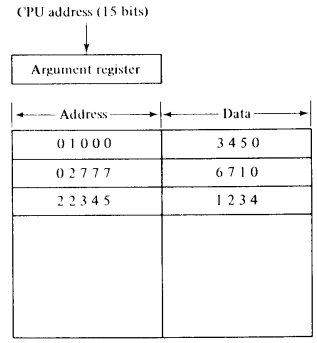


그림 12-11 어소시어티브 매핑 캐시(모든 숫자는 8진수)

  - 단점 : 비교기가 많이 들어감으로 비용 증가

- 직접 매핑
  - 주기억 장치와 캐쉬 메모리 사이의 주소 관계 (그림 12-12)

캐쉬 메모리 (계속)

Slide 20

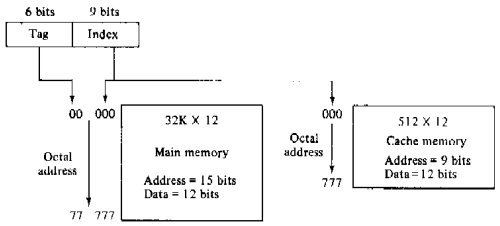


그림 12-12 주기억 장치와 캐시 메모리 사이의 주소 관계

– 구조 (그림 12-13)

캐쉬 메모리 (계속)

Slide 21

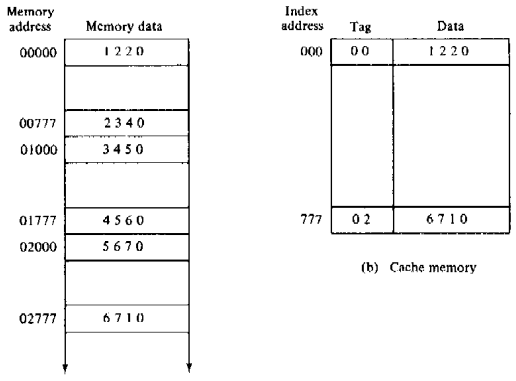


그림 12-13 직접 매핑 캐시 구조

– 단점

* 두개의 인덱스를 가진 다른 tag 가 번갈아 접근되면 힛트율이 떨어짐

캐쉬 메모리 (계속)

Slide 22

→쓰레싱 (thrashing) 이라함

- 세트-어서시어티브 매핑
  - 두개의 세트 크기를 가진 세트-어소시어티브 개쉬의 구조 (그림 12-15)

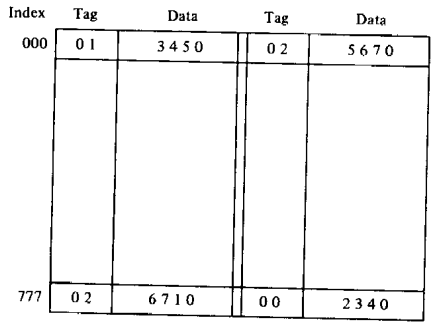


그림 12-15 두 개의 세트 크기를 가진 세트-어소시어티브 개쉬

  - 비교기는 세트수 만큼만 필요

캐쉬 메모리 (계속)

Slide 23

  - 쓰레싱 방지됨

- 캐쉬에 기록
  - 두가지 기록 방법
    1. write-through : 캐쉬 내용 갱신 곧바로 주메모리 내용 갱신
       →캐쉬 일관성이 보장됨
    2. write-back : 캐쉬 내용 갱신 나중에 주메모리 내용 갱신
       →캐쉬 일관성이 보장되지 않음

- 캐쉬의 초기값 설정
  - 전원이 켜지면 임의의 값을 가짐
  - 모든 유효비트 (valid bit) 를 0으로 초기화

가상 메모리

Slide 24

- 가상 메모리 사용목적
  - 보조메모리 (하드디스크) 크기 만큼의 기억장소 (main memory) 가 있는것처럼 만듦
  - multiprocessing 이나 multiuser 환경을 지원
- 주소공간과 메모리 공간
  - 가상 주소 (virtual address) : 프로그래머에 의하여 쓰여진 주소
  - 물리적 주소 (physical address) : 주기억 장소의 주소
  - 주소공간 (address space) : 가상주소의 집합
  - 메모리공간 (memory space) : 주기억장소의 주소 집합
  - 20비트 가상주소 15비트 물리적주소를 갖는 환경 (그림 12-16)

가상 메모리 (계속)

Slide 25

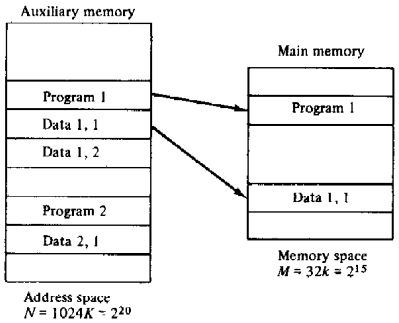


그림 12-16 가상 메모리 시스템에서 주소와 메모리 공간 사이의 관계

  - 가상주소를 매핑하기위한 메모리 표 (그림 12-17)

가상 메모리 (계속)

Slide 26

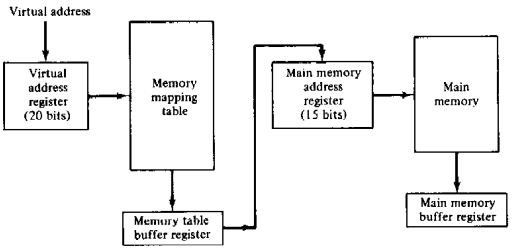


그림 12-17 가상 주소를 매핑하기 위한 메모리표

- 페이지를 사용하는 주소 매핑
  - 블럭 (block) : 물리적 메모리를 같은 크기로 나눈것
  - 페이지 (page) : 주소공간을 같은 크기로 나눈것
  - 8K 주소공간, 4K 의 메모리공간 을 1K의 크기로 나눈 구성 (그림 12-18)

가상 메모리 (계속)

Slide 27

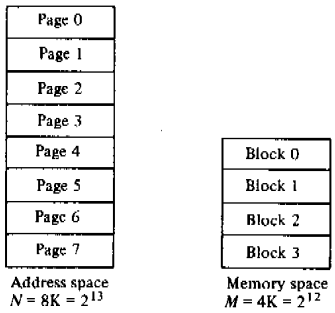


그림 12-18 주소 공간과 메모리 공간을 1K 워드의 그룹으로 분리

  - 페이지내의 라인주소는 블럭내의 라인주소와 동일
    페이지 번호에서 블럭번호로의 변환만 필요
  - 메모리 변환표의 구성도 (그림 12-19)

가상 메모리 (계속)

Slide 28

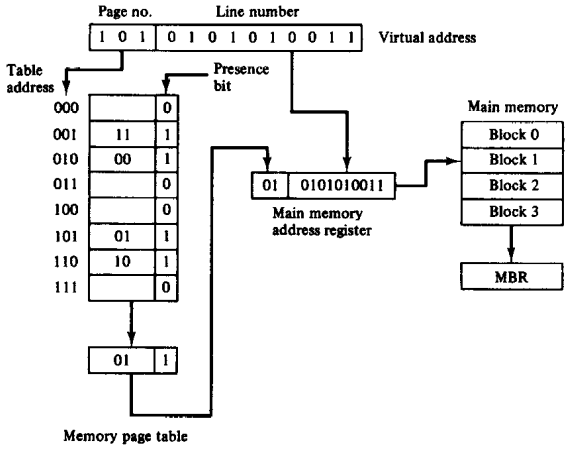


그림 12-19 페이지로 된 시스템에서의 메모리표

- 어소시어티브 메모리 페이지표

가상 메모리 (계속)

– 어소시어티브 메모리를 이용하여 페이지표를 구성 (그림 12-20)

Slide 29

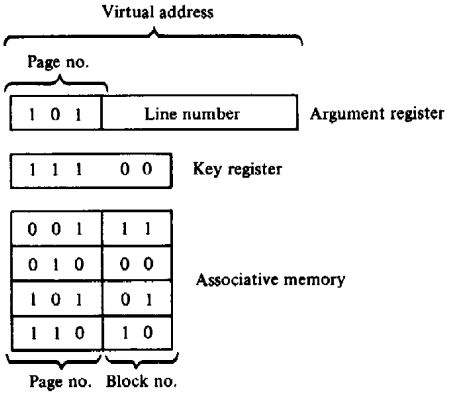


그림 12-20 어소시어티브 메모리 페이지표

– 빠른 검색이 가능

가상 메모리 (계속)

- 페이지의 교체
  - 가상메모리에서 소프트웨어의 역할
    - 새로운 페이지 요구시 주기억 장치의 페이지중 어떤 페이지를 보조기억장치로 옮길것인가 결정
    - 언제 새 페이지가 보조 기억장치로 부터 주기억장치로 옮겨지는가 결정
    - 주기억 장치에 어디에 저장할것인가 결정

Slide 30

  - 하드웨어 주소 변환장치 + 관리 소프트웨어 →가상 메모리 구성
  - 페이지 결함 (page fault) : 주기억장치에 해당 페이지가 없을경우
    - 해당 페이지가 보조기억장치로 부터 주기억장치로 모두 옮겨질때까지 기다림
    - I/O 프로세서에게 요구 →타 프로그램 수행
    - 전송완료 →해당 프로그램 계속 수행
  - 교체 알고리즘 (replacement algorithm)
    - 주기억 장치가 유효한 페이지로 모두 차고
    - 새로운 페이지요구가 발생시
    - 어떤 페이지를 보조기억장치로 밀어낼것인가 결정하는 알고즘

가상 메모리 (계속)

Slide 31

- 종류
  1. FIFO : 먼저 올라온것을 먼저 제거
  2. LRU (least recently used) : 가장 오랫동안 사용안된것을 먼저 제거

메모리 관리 하드웨어

Slide 32

- 역할
  1. 논리 메모리 참조를 물리 메모리 주소로 변환하는 동적 저장 장소 재배치를 위한 기능
  2. 메모리네에서 서로 다른 사용자가 하나의 프로그램을 같이 사용하기 위한 편의
  3. 사용자간의 허락되지 않은 접근을 방지하고 사용자가 O.S 의 기능을 변경못하도록 하는 정보의 보호
- 세그멘트 (segment)
  - 논리적으로 연관된 명령과 데이타의 집합
  - 예) 서브루틴, 데이타의 배열, 기호표, 사용자의 프로그램등
  - 세그먼트로 된 프로그램에 의하여 지정되는 주소를 논리 주소라고 함
- 세그먼트내의 페이지 매핑
  - 세그멘트의 크기는 실행되는 프로그램의 필요에 의하여 변화할수 있음
  - 세그멘트를 여러개의 페이지로 나눔

메모리 관리 하드웨어 (계속)

Slide 33

- 논리주소에서 물리 주소로의 매핑 (그림 12-21)

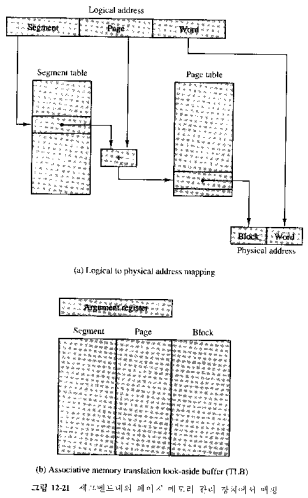


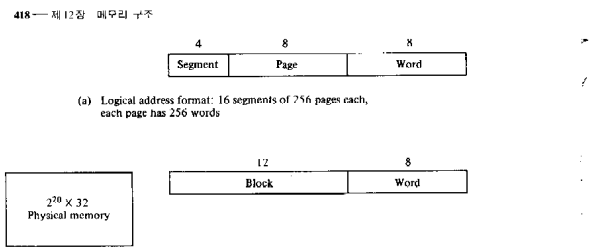
그림 12-21 세그멘트와 페이지 메모리 관리 장치에서 매핑

- 수치 예

메모리 관리 하드웨어 (계속)

Slide 34

– 논리 및 물리 주소의 예 (그림 12-22)



(a) Logical address format: 16 segments of 256 pages each, each page has 256 words

(b) Physical address format: 4096 blocks of 256 words each, each word has 32 bits

그림 12-22 논리 및 물리 주소의 예

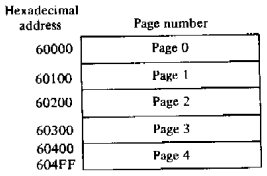
논리 주소의 범위는 60000~604FF이다. 프로그램이 물리 메모리로 적재될 때 OS는 빈 공간을 찾아 임의로 다섯 개의 블럭에 저장한다. 논리 페이지 번호와

– 논리 및 메모리 주소 할당의 예 (그림 12-23)

메모리 관리 하드웨어 (계속)

Slide 35

트 6은 다섯 개의 페이지로 구성되어 있는데 페이지표에서 보듯이 35번지에서 39번지에 있다. 세그먼트 6의 페이지 2는 35+2=37번지이다. 물리 메모리 블럭은 페이지표에서 찾을 수 있으며 값은 019이다. 따라서 블럭 19의 워드



(a) Logical address assignment

Segment	Page	Block
6	00	012
6	01	000
6	02	019
6	03	053
6	04	A61

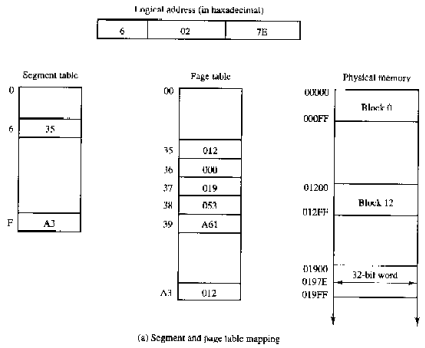
(b) Segment-page versus memory block assignment

그림 12-23 논리 및 메모리 주소의 할당의 예

– 논리 메모리를 물리 메모리로 매핑하는 예 (그림 12-24)

메모리 관리 하드웨어 (계속)

Slide 36



(a) Segment and page table mapping

Segment	Page	Block
6	02	019
6	04	A61

(b) Associative memory (TLB)

그림 12-24 논리 메모리를 물리 메모리로 매핑하는 예(모두 숫자 16진수)

메모리 관리 하드웨어 (계속)

Slide 37

- 메모리 보호
  - 논리주소나 물리주소를 통한 메모리 접근 보호가능
  - 논리주소를 통한 메모리 접근이 더 좋음
  - 논리주소 세그멘트표 안에 보호에 대한 정보 저장
  - 전형적인 세그멘트 설명자의 형식 (그림 12-5)
  - 접근 권한의 종류
    1. Full read and write 특권 : 자신의 프로그램
    2. Read only (write protection) : 유틸리티나 라이브러리 프로그램
    3. Execute only (program protection) : 프로그램 복사 방지 프로그램
    4. System only (operating system protection) : OS 프로그램