기억장치(메모리)

계층적 메모리 구조 (Memory Hierarchy)

개념

계층적 메모리 구조는 컴퓨터 시스템에서 메모리를 속도, 용량, 비용 간의 절충을 고려하여 계층적으로 나누는 방식입니다. 이를 통해 처리 속도와 효율성을 극대화합니다. 메모리는 아래에서 위로 갈수록 속도는 빠르지만 용량은 작고 비용은 높아지는 특징을 가집니다.

계층 구성

레지스터(Register)

- CPU 내부에 위치하며 가장 빠른 메모리입니다.
- 용량이 매우 작고 휘발성입니다.
- 예: 연산 중 필요한 데이터 저장.

캐시 메모리(Cache Memory)

- CPU와 메인 메모리 사이에 위치하며, 자주 사용하는 데이터를 저장하여 접근 속도를 높입니다.
  예: L1, L2, L3 캐시.

주 기억 장치(Main Memory)

RAM으로 구성되며, 실행 중인 프로그램과 데이터를 저장합니다.
속도는 캐시보다 느리지만 용량이 더 큽니다.

보조 기억 장치(Secondary Storage)

- HDD, SSD와 같은 저장 장치로, 데이터 영구 저장에 사용됩니다.
- 속도는 느리지만 용량이 매우 큽니다.
- 외부 저장 장치(Tertiary Storage)
- 테이프 드라이브와 같은 장치로, 백업 및 장기 저장에 사용됩니다.

지역성의 원리

- 계층적 메모리 구조는 시간 지역성과 공간 지역성을 활용합니다.
- 시간 지역성: 최근에 사용된 데이터는 다시 사용될 가능성이 높음.

- 공간 지역성: 특정 데이터 근처의 데이터도 함께 사용될 가능성이 높음.

캐시 메모리 (Cache Memory)

개념

- 캐시 메모리는 CPU와 메인 메모리 간의 속도 차이를 줄이기 위해 사용되는 고속 메모리입니다. 자주 사용하는 데이터를 저장하여 CPU가 빠르게 접근할 수 있도록 합니다.

특징

- 캐시는 일반적으로 SRAM으로 구성되어 있으며, 속도가 빠릅니다.
- CPU 내부(L1 캐시) 또는 외부(L2, L3 캐시)에 위치합니다.

캐시 레벨

- L1 캐시: CPU 코어에 내장, 가장 빠르고 용량이 작음.
- L2 캐시: CPU 외부에 위치, L1보다 느리지만 용량이 큼.
- L3 캐시: 모든 코어가 공유, 가장 느리지만 용량이 큼.

캐시 정책

쓰기 정책: Write-Through(캐시와 메모리에 동시에 기록), Write-Back(캐시에만 기록 후 필요 시 메모리에 기록).

교체 알고리즘: FIFO, LRU, LFU, Random 등.

예시

CPU가 특정 데이터를 요청할 때, 캐시에서 데이터를 찾으면 **캐시 히트(Cache Hit)**가 발생하여 빠르게 처리됩니다.

캐시에 데이터가 없으면 **캐시 미스(Cache Miss)**가 발생하며, 메인 메모리에서 데이터를 가져옵니다.

정보처리산업기사 필기시험과 관련하여 계층적 메모리 구조, 캐시 메모리, 가상 메모리, 그리고 메모리 주소지정방식에 대해 자세히 설명하겠습니다.

계층적 메모리 구조

계층적 메모리 구조는 컴퓨터 시스템에서 속도, 용량, 비용 간의 균형을 맞추기 위해 설계된 구조입니다. 이 구조는 다음과 같이 구성됩니다:

레지스터: CPU 내부에 위치한 가장 빠른 메모리로, 소량의 데이터를 저장합니다.

캐시 메모리: CPU와 주 메모리 사이에 위치하며, 자주 사용되는 데이터를 저장합니다.

주 메모리(RAM): 프로그램과 데이터를 임시로 저장하는 휘발성 메모리입니다.

보조기억장치(HDD, SSD): 대용량 데이터를 영구적으로 저장하는 비휘발성 저장소입니다.

이 구조에서 상위로 갈수록 속도는 빨라 지지만 용량은 작아지고 비용은 증가합니다.

캐시 메모리

캐시 메모리는 CPU와 주 메모리 사이의 속도 차이를 줄이기 위한 고속 버퍼 메모리입니다. 주요 특징은 다음과 같습니다:

계층 구조: L1, L2, L3 캐시로 구분되며, L1이 가장 빠르고 용량이 작습니다.

지역성 원리: 시간적 지역성(최근 사용된 데이터가 곧 다시 사용될 가능성)과 공간적 지역성(인접한 데이터가 순차적으로 사용될 가능성)을 활용합니다.

예를 들어, 반복문에서 자주 사용되는 변수는 캐시에 저장되어 빠르게 접근할 수 있습니다.

가상 메모리

가상 메모리는 물리적 메모리의 한계를 극복하기 위한 기술로, 다음과 같은 특징을 가집니다:

페이징: 메모리를 일정 크기의 페이지로 나누어 관리합니다.

스와핑: 필요한 페이지만 주 메모리에 로드하고, 사용하지 않는 페이지는 보조 기억장치로 이동시킵니다.

예를 들어, 4GB RAM을 가진 컴퓨터에서 8GB 크기의 프로그램을 실행할 수 있게 해줍니다.

메모리 주소지정방식

메모리 주소지정방식은 CPU가 메모리의 특정 위치를 참조하는 방법을 말합니다. 주요 방식은 다음과 같습니다:

직접 주소지정: 명령어에 실제 메모리 주소를 직접 지정합니다.
예: MOV AX, [1000H] (1000H 주소의 데이터를 AX 레지스터로 이동)

간접 주소지정: 주소를 가리키는 포인터를 사용합니다.
예: MOV AX, [BX] (BX 레지스터가 가리키는 주소의 데이터를 AX로 이동)

즉시 주소지정: 데이터를 직접 명령어에 포함시킵니다.
예: MOV AX, 100 (100을 AX 레지스터에 직접 저장)

상대 주소지정: 현재 위치에서 상대적인 주소를 사용합니다.
예: JMP +10 (현재 위치에서 10바이트 앞으로 점프)

이러한 주소지정방식을 이해하고 적절히 사용하면 효율적인 메모리 접근이 가능해집니다.