1. 메모리와 저장장치
   1. SRAM
      1. 하나의 Cell(1비트 저장단위) 는 6개의 트랜지스터로 구성된다.
      2. 전원이 공급되는 동안 저장된 값을 무한히 보존
      3. 외부 충격에 내성이 있다.
      4. 빠르고 가격이 비싸다
      5. cpu안에 들어가는 레지스터에 사용.
   2. DRAM
      1. 배터리의 동작원리를 빌려서 만든 램
      2. 하나의 셀은 하나의 캐패시터로 구성된다. 이걸 읽기 위한 하나의 트랜지스터가 필요하다.
      3. 10~100ms마다 refresh해야한다. –누설 전류
      4. 외부 자극에 민감하다.
      5. 주 메모리에 사용된다.
   3. Enhanced DRAM
      1. SDRAM 클럭을 이용하여 메모리 제어
      2. DDR SDRAM 클럭의 상승/하강을 모두 이용하여 메모리 제어
   4. NV MEMORY(비휘발성)
      1. ROM, EPROM, FLASH MEMORY 등
      2. 전원이 공급되지 않아도 데이터가 저장됨
      3. 새로운 값을 넣는데 큰 비용이 들어간다.
      4. 펌웨어(고정된) 프로그램 저장용
      5. DRAM 수준의 접근 속도를 제공하는 비휘발성 메모리 개발중
   5. 버스 구조
      1. 메모리는 I/O 브릿지(사우스 브릿지)에 버스로 연결되어 있다
2. 메모리 계층 구조
   1. 시스템 하드웨어간 큰 성능차이 (용량 / 성능 반비례 관계)
   2. 프로세스 사이클의 병목문제
   3. 해결방안 – 메모리 계층 구조
      1. 컴퓨터 프로그램의 특징 : 참조의 지역성(Locality)
      2. 지역성을 활용하여 마치 빠르고 용량이 많은 메모리가 있는 것 처럼
   4. 지역성
      1. 지역성의 원리
         1. 프로그램이 최근 사용한 것과 같은 주소 또는 근처 주소의 데이터와 명령어를 계속 사용하려는 경향이 있다.
         2. 메모리 계층구조가 효과적으로 동작하는 근거
         3. 전체 소프트웨어에 필요한 데이터 양이 많더라도 시간을 작게 쪼개어 보면 지금 현재 동작에 사용되는 메모리는 그렇게 크지 않다.
         4. 그리고 그 사용되는 메모리의 영역은 참조 지역성에 의해 예측될 수 있다.
      2. 시간 지역성
         1. 최근에 참조한 항목이 가까운 미래에 다시 참조하게 되는 특성
         2. 대표적인 사례 : 반복문
      3. 공간 지역성
         1. 일정 시간동안 근접한 주소의 항목들이 함께 참조되는 특성
         2. 대표적인 사례 Array
      4. 프로그램에서 지역성
         1. SUM PROC
            1. sum = 0;
            2. For(int i = 0 ; i < 100; ++i)

sum += arr[i];

//데이터 참조

//공간 지역성(바로 다음 메모리 참조)

//시간 지역성(같은 변수 sum을 반복적으로 참조)

* + - * 1. Return sum
        2. 명령어 참조

명령어를 순차적으로 참조 : 파이프라이닝 적용가능

순환문 내의 명령어가 반복적으로 참조

* + - 1. 2차원 배열의 경우
         1. For(int i =0 ;i < MAX; i++)

For(int j =0 ; j < MAX; j++)

Sum += a[i][j]; //열++: 좋은 지역성

Sum += a[j][i]; //행++: 나쁜 지역성

* 1. 캐시
     1. 메모리 계층 구조에서 아래쪽 저장소에 대해서 위쪽 저장소가 하는 역할
        1. 아래단계의 데이터 중 일부를 더 빠른 상위계층의 저장장치가 저장하는 것
        2. 가능한 상위 계층의 메모리에 접근하여 작업이 진행 될 수 있도록 만드는 것
     2. Hit / Miss
        1. 상위 계층의 메모리(캐시)에 저장한 데이터가 호출된 경우 : HIT
        2. 캐시에 호출된 데이터가 없는 경우 : MISS
           1. 하위 계층의 메모리에서 다시 호출
           2. 상위 캐시에서 데이터 Replacement

상위캐시에서 시간/ 공간 로컬리티에서 가장 먼 아이템(victim)을 새롭게 호출된 데이터와 교체

* + - * 1. 종류

콜드 미스 : 캐시가 비어있을 때

충돌 미스 :

하위 단계의 블록들은 배치 정책에 따라 캐시에 저장됨.

여러 블록이 동일한 위치에 배치됨으로써 발생하는 캐시 미스

용량 미스 : 상위 캐시가 용량이 부족하기 때문에 어쩔 수 없이 발생하는 캐시 미스