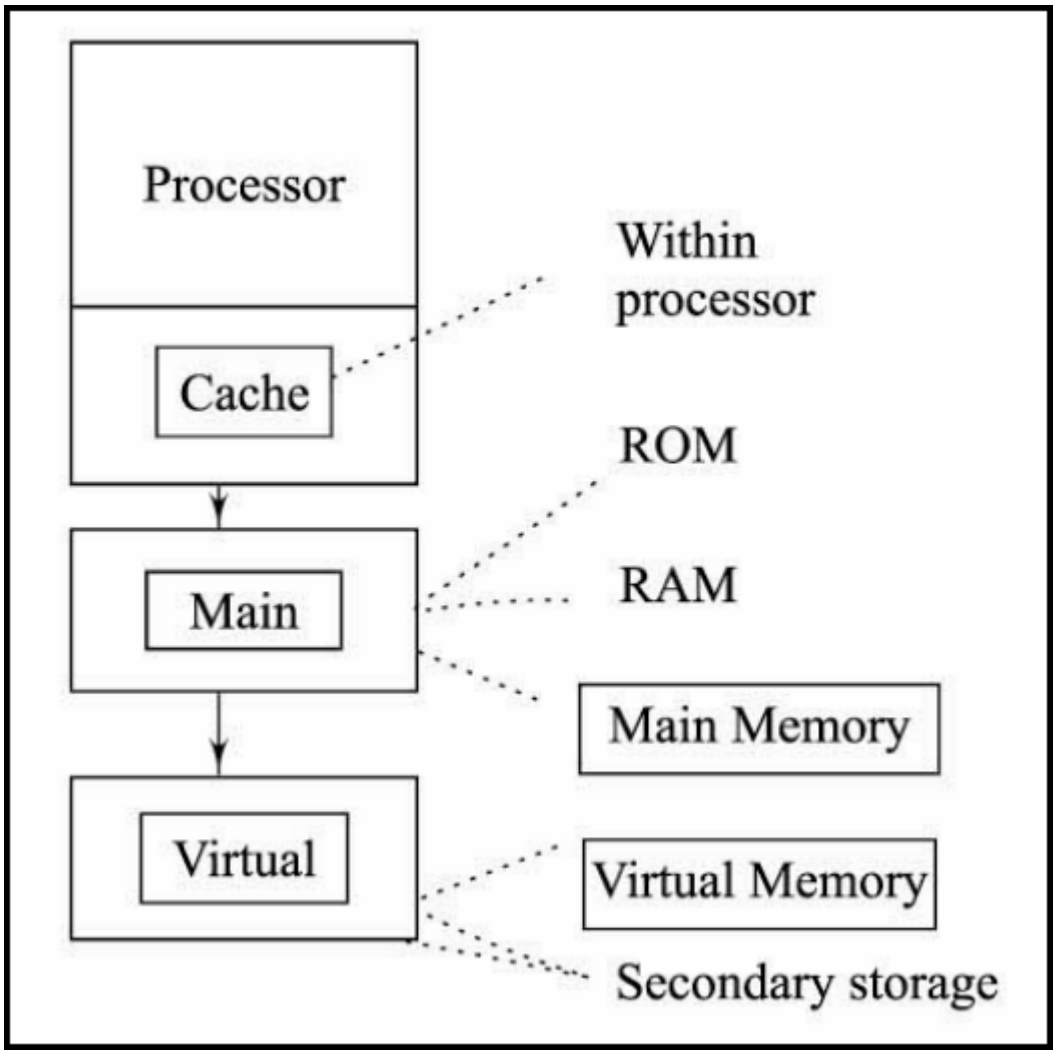
Write Through Vs Write Back

Store operation



Write Through

▶Write Through란?

CPU가 주기억장치 또는 디스크로 데이터를 기입하고자 할 때

그 데이터는 먼저 캐시로 기입된다.

이때 데이터가 캐시 됨과 동시에 주기억장치 또는 디스크로 기입되는 방식을

지원하는 구조의 캐시이다.

즉 캐시와 메모리 둘다에 업데이트를 해버리는 방식이다.

장점은 캐시와 메모리에 업데이트를 같이 하니 inconsistency현상이 발생하지 않고

안정적이다.

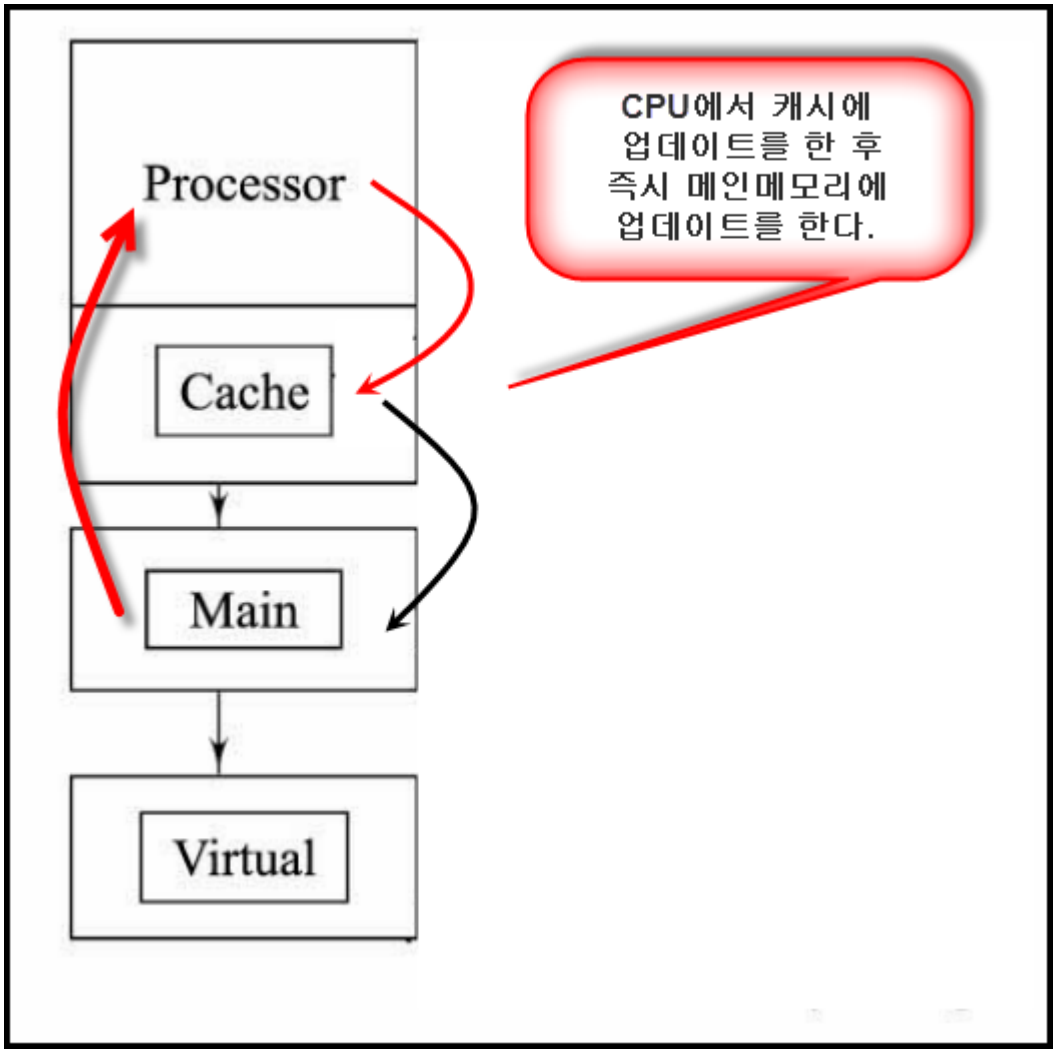
단점은 속도가 느린 주기억장치 또는 디스크로 데이터를 기록하는 조작이 완료될 때까지

CPU가 대기하는 시간이 필요하기 때문에 성능이 떨어진다.

▶데이터 로스의 리스크가 있으면 안되는 상황에서는 Write Through를 사용하는 것이

바람직하다.

▶Write Through를 사용하여 데이터를 처리하는 구조



Write Back

▶Write Back이란?

위에서 봤듯이 CPU가 주기억장치 또는 디스크로 데이터를 기록하고자 할 때,

그 데이터는 먼저 캐시로 기록되는데 캐시 내에 일시적으로 저장된 후에

블록 단위로 유휴 머신 주기 동안에 캐시로부터 해제되는 때(캐시안에 있는 내용을

버릴시) 에만 주기억장치 또는 디스크로 기록되는 방식이다.

즉 데이터를 쓸 때 메모리에는 쓰지 않고 캐시에만 업데이트를 하는 방법이다.

장점은 Write Through보다 훨씬 빠르다.

단점은 속도가 빠른 대신에 캐시에 업데이트 하고 메모리에는 바로 업데이트를

하지 않기 때문에 캐시와 메모리가 서로 값이 다른 경우가 발생할 때가 있다.

이걸 inconsistency라고 하는데 이렇게 되면 캐시에만 써놓고 Device에 값을 안넘기는

경우가 생긴다. 예로 LCD같은 곳에다가 데이터를 뿌리고 싶은데 LCD에 값을 썻더니

캐시에만 업데이트되고 LCD에는 값을 넘겨주지 않아서 LCD에 그림이 뿌려지지 않는

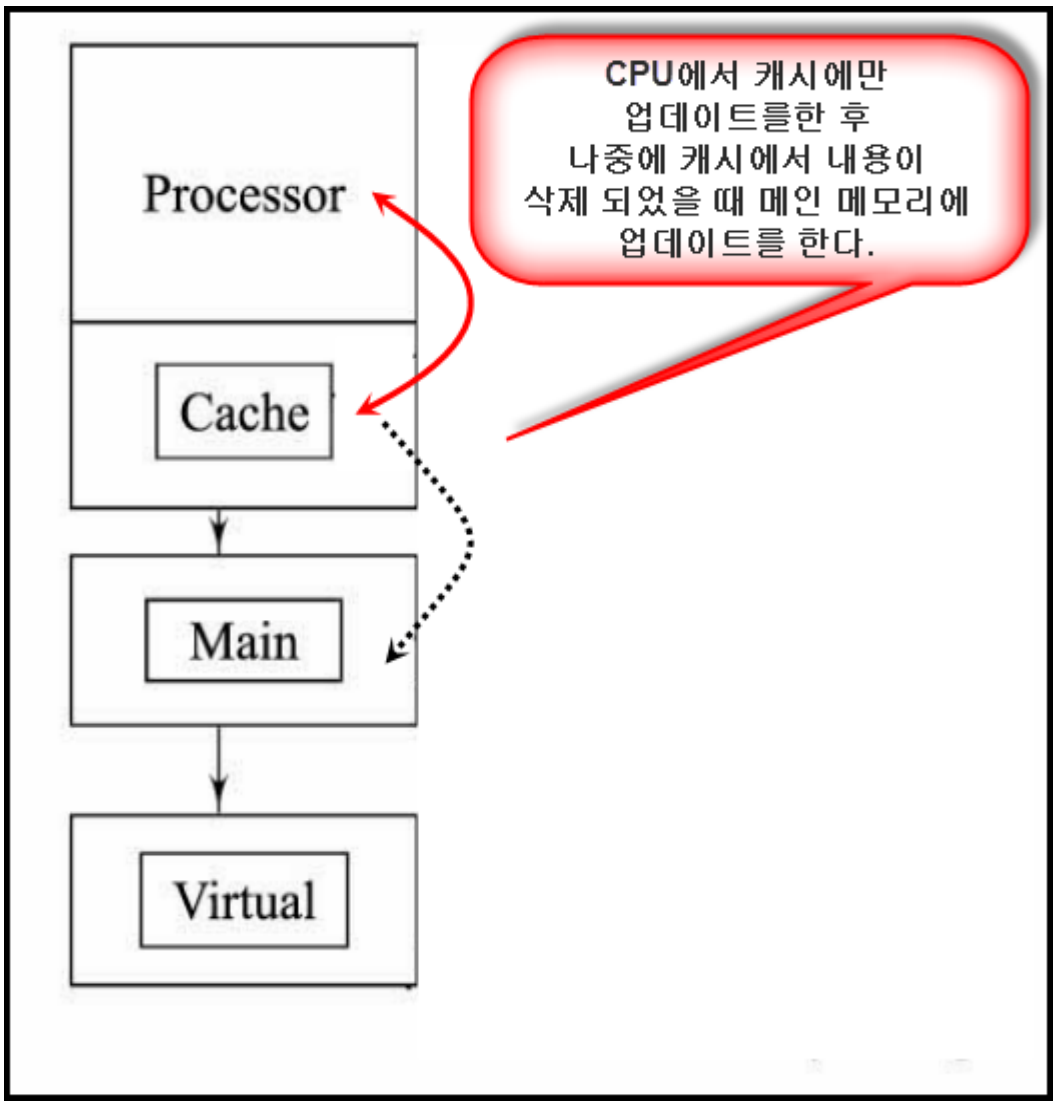
현상이 발생한다.

이런 현상을 해결하기 위하여 Cache Flush 또는 Cache clean을 사용한다.

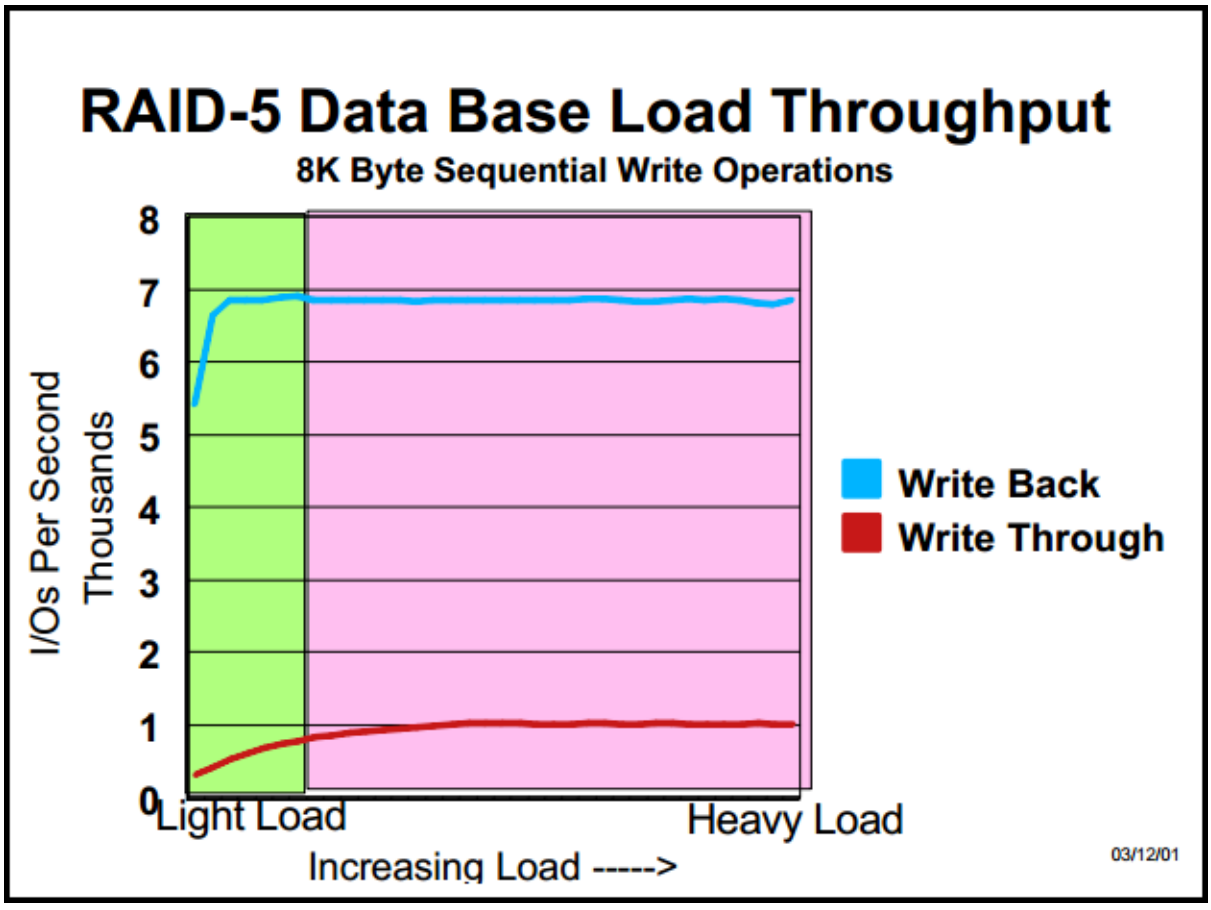
▶데이터 로스의 리스크를 조금 감수하더라도 빠른 서비스를 요하는 상황에서는

Write Back을 사용하는 것이 바람직하다.

▶Write Back을 사용하여 데이터를 처리하는 구조



Write Through vs Write Back



<RAID 5 구성상에서 데이터 베이스 로드 처리량 비교>