HW-3

[1] Cache memory 구조 설계에서 블록 매핑 방식에 따라 3가지로 구분된다. 이러한 3가지 Cache 내 블록 배치/매핑 방법을 설명하시오.

공통적용 개념 :
* Cache Index : Cache memory 내에 access 할 수 있는 block의 주소 (Cache 내 block 수 만큼 존재)
* Set : Cache index로 추출해서 cache access 하는 용도로 사용할 때 이 주소에 mapping이 될 수 있는 공간
* Cache 내 block 배치 방법 : Main memory block number를 각각의 mapping 구조에 따른 set의 개수로 나눈 나머지 값을 취하여 어디에 배치해야 할지 구할 수 있음 (modulo operation)

1. Direct mapped : CPU가 생성해주는 메모리 주소에서 cache index를 추출하여, 그에 맞는 cache memory block에 일대일 mapping 하는 것을 말한다. 한 개의 set에 한 개의 block이 존재하기에 여기서 set의 수는 cache memory의 block 개수와 동일하다.
2. Set associative : CPU가 어드레스 생성하면 그 주소로부터 cache index를 추출하여, 그 위치에 N개의 block entry가 존재하는 구조를 N-way set associative cache 구조라고 한다. 한 개의 set에 N 개의 block이 존재하기에 set의 수는 cache memory block의 총 개수를 N개로 나누어서 구할 수 있다. 이 한 set 안에서는 그 어디든지 mapping 할 수 있다. 보통 N은 2개에서 4개 정도 취한다.
   이 경우 Direct mapped와 다른 점은 N의 개수만큼 comparators가 있고 (Directed mapped의 경우 entry가 한 개라서 한 개의 comparator만 필요), Extra MUX delay가 발생할 수 있으며, Hit/Miss 판단 후에 Data가 다음 단계로 넘어갈 수 있다.
3. Fully associative : Cache 내 block에 자유롭게 어디에든지 넣을 수 있는 mapping 방법이다. Cache 자체가 한 개의 set이 되고 전체 cache block 이 이 set에 속한다.

[2] Cache memory에서 2가지 쓰기 방식에 대해 설명하고, 각 장단점을 제시하시오.

1. Write through : Cache memory에 해당하는 block에 어떤 데이터를 쓰게 되면, 동시에 그 block에 해당하는 main memory block에도 똑같이 write가 된다.
   - 장점 : Cache 내에 read miss가 발생했을 때, write 동작을 유발하지 않는다.
   - 단점 : Cache와 main memory 사이에 있는 write buffer를 통해서 main memory에 쓰이기 때문에 main memory에 쓰이는 속도보다 CPU로부터 들어오는 data 속도가 빠르면 문제가 된다.

2. Write back : 새로운 information이 cache에만 쓰이고, main memory에는 업데이트가 되지 않다가, 이 block이 cache에서 교체가 되는 경우 그때 main memory의 해당 block에 쓰는 동작을 수행하는 것이다. Main memory에 대한 write 동작이 delay 되는 방법이다.
   - 장점 : Data가 업데이트되었을 때, 마지막 replace 될 때만 write 동작을 수행시키기 때문에 반복적으로 main memory에 쓰이지 않는다.
   - 단점 : Cache 내에 read miss가 발생했을 때, write 동작을 유발한다.

[3] Cache memory 내에서 대표적인 블록 교체 방법을 제시하고, Random 교체 방식이 의미가 있는 방식인지 캐시 용량에 따라 다른 방법과 비교하여 설명하시오.

Cache memory 내에서 block을 교체할 경우 Direct mapped 구조에서는 해당 entry를 메모리로 보내고 miss 난 새로운 block을 해당 위치로 읽어오면 된다. Set Associative 혹은 Fully Associative로 mapping 된 구조는 각각 N개의 set, 혹은 캐시 전체 block 수 만큼의 entry가 존재하여서 무엇을 교체 해야 하는지 명확하지 않다.

이 경우 첫 번째 해결 방법으로는 LRU (Least Recently Used) 방법을 사용해서 가장 오랫동안 사용하지 않은 block이 무엇인지 찾아서 그 entry를 교체 할 수 있다.
두 번째 해결 방법으로는 Random으로 해당 set 내 아무 entry나 무작위로 선택하여 교체하는 방법이 있다.

Random은 set 내 아무거나 하나 선택해서 교체하기 때문에 cost가 별로 들지 않지만, LRU는 언제 마지막으로 사용되었는지를 알아야 하므로 이전 history를 모두 유지 관리해야 하고 이에 따라 비용이 발생한다. 하지만 Cache의 크기가 커질수록, Associativity가 커질수록 두 방법이 차이가 없다는 결론이 나온다 (miss ratio 차이가 없다). 따라서 Random 교체 방식이 의미가 있는 방식이라고 할 수 있다.

[4] Cache memory에서 Write allocate 방식과 No write allocate 방식에 대해 설명하시오.

1. Write allocate : Write를 read miss와 똑같이 취급하는 것이다. Write miss가 발생하면 cache block을 cache 내에 할당하고, write back과 함께 사용된다.
2. No write allocate : Write miss가 발생해도 cache 내에 block 할당을 하지 않는다. 해당하는 block write가 cache 내에 할당이 되지 않았다고 하면, lower level memory에 바로 수정이 이루어지게 된다. 그리고 반드시 cache로 올라갈 경우에는 read miss가 발생할 경우에만 올라가며, write through 방식과 결합하여 사용된다.