Homework #3 - Solution

1. A two-part question.

A. Assume the following 10-bit address sequence generated by the microprocessor:

The cache uses 4 bytes per block. Assume a 2-way set associative cache design that uses the LRU algorithm (with a cache that can hold a total of 4 blocks). Assume that the cache is initially empty. First determine the TAG, SET, BYTE OFFSET fields and fill in the table above. In the figure below, clearly mark for each access the TAG, Least Recently Used (LRU), and HIT/MISS information for each access.

Time	0	1	2	3	4	5	6	7
Access	10001101	10110010	10111111	10001100	10011100	11101001	11111110	11101001
TAG	10001	10110	10111	10001	10011	11101	11111	11101
SET	1	0	1	1	1	0	1	0
INDEX	01	10	11	00	00	01	10	01

In the following each access is represented by a triple: (Tag, LRU Bit, Hit Bit)

LRU bit = 1 if the current block is the least recently used one.
Hit Bit = 1 if the current reference is a hit

Initial

	Block 0		Block 1
Set 0			
Set 1			

Access 0

	Block 0		Block 1
Set 0			
Set 1	10001,0,0		

Access 1

	Block 0		Block 1
Set 0	10110,0,0		
Set 1	10001,0,0		

Access 2

	Block 0		Block 1
Set 0	10110,0,0		
Set 1	10001,1,0		10111,0,0

Access 3

	Block 0		Block 1
Set 0	10110,0,0		
Set 1	10001,0,1		10111,1,0

Access 4

	Block 0		Block 1
Set 0	10110,0,0		
Set 1	10001,1,0		10011,0,0

Access 5

	Block 0		Block 1
Set 0	10110,1,0		11101,0,0
Set 1	10001,1,0		10011,0,0

Access 6

	Block 0		Block 1
Set 0	10110,1,0		11101,0,0
Set 1	11111,0,0		10011,1,0

Access 7

	Block 0		Block 1
Set 0	11101,0,0		11101,0,0
Set 1	11111,0,0		10011,1,0

B. Derive the hit ratio for the access sequence in Part A.

The hit ratio for the above access sequence is given by : (1/8) = 0.125

2. Explain the two characteristics of program memory accesses that caches exploit.
   Temporal locality와 spatial locality에 관한 설명이면 정답

3. Design a 128KB direct-mapped data cache that uses a 32-bit address and 16 bytes per block. Calculate the following:
   A. How many bits are used for the byte offset? 2^4=16, 따라서 4 bits
   B. How many bits are used for the set (index) field? 128KB/16B = $2^{17}/2^4 = 2^{13}$, 따라서 13 bits
   C. How many bits are used for the tag? 32-4-13 = 15 bits

4. The Average Memory Access Time equation (AMAT) has three components: hit time, miss rate, and miss penalty. For each of the following cache optimizations, indicate which component of the AMAT equation is improved.
   A. Using a second-level cache: Using a second-level cache improves miss rate
   B. Using a direct-mapped cache: Using a direct-mapped cache improves hit time
   C. Using a 4-way set-associative cache: Using a 4-way set-associative cache improves miss rate
   D. Using larger blocks: Using larger blocks improves miss rate

5. A three-part question. This question covers cache and pipeline performance analysis.
   A. Write the formula for the average memory access time assuming one level of cache memory:

Average Memory Access Time = Time for a hit + Miss rate × Miss penalty

   B. For a data cache with a 92% hit rate and a 2-cycle hit latency, calculate the average memory access latency. Assume that latency to memory and the cache miss penalty together is 124 cycles. Note: The cache must be accessed after memory returns the data.

AMAT = 2 + 0.08*124 = 11.92 cycles

   C. Calculate the performance of a processor taking into account stalls due to data cache and instruction cache misses. The data cache (for loads and stores) is the same as described in Part B and 30% of instructions are loads and stores. The instruction cache has a hit rate of 90% with a miss penalty of 50 cycles. Assume the base CPI using a perfect memory system is 1.0. Calculate the CPI of the pipeline, assuming everything else is working perfectly. Assume the load never stalls a dependent instruction and assume the processor must wait for stores to finish when they miss the cache. Finally, assume that instruction cache misses and data cache misses never occur at the same time. Show your work.

• Calculate the additional CPI due to the icache stalls: The additional CPI due to icache stalls = Hit Rate * Hit Latency + Miss Rate* Miss Penalty = 0.9*2 + 0.1*50 = 1.8 + 5 = 6.8
• Calculate the additional CPI due to the dcache stalls:
The additional CPI due to dcache stalls = 0.92*2 + 0.08*124 = 11.76
• Calculate the overall CPI for the machine. The overall CPI = 0.3*11.76 + 0.7*1.0 + 1.0*6.8 = 11.03

6. Caches: Misses and Hits

```
int i;
int a[1024*1024];
int x=0;
for(i=0;i<1024;i++){
       x+=a[i]+a[1024*i];
}
```

Consider the code snippet in code above. Suppose that it is executed on a system with a 2-way set associative 16KB data cache with 32-byte blocks, 32-bit words, and an LRU replacement policy. Assume that `int` is word-sized. Also assume that the address of `a` is 0x0, that `i` and `x` are in registers, and that the cache is initially empty. How many data cache misses are there? How many hits are there?

주어진 캐시 시스템의 set의 개수는 16x1024/(32x2) = 2^8이므로 set의 index는 0~255까지 존재한다. 변수 i와 x는 register에 있다고 가정했으므로 캐시 메모리에 영향을 미치지 않는다. 배열 a는 int type이므로 a[0]~a[7]이 캐시의 set 0, a[8]~a[15]가 캐시의 set 1, a[16]~a[23]이 캐시의 set 2... 와 같은 방법으로 저장된다. 반면 a[1024], a[1024x2], a[1024x3], a[1024x4]... 는 캐시에서 set 128, set 0, set 128, set 0... 과 같은 순서로 저장된다.
따라서 a[i]를 읽을 때는 8번마다 한번의 cache miss와 7번의 cache hit가 발생한다. 그러므로 1024/8=128번 miss와 128*7=896번 hit가 발생한다.
a[1024*i]의 경우 주소의 증가폭이 cache block 하나의 크기보다 크므로 cache block을 읽은 후 재사용하지 않는다. 따라서 매번 cache miss가 발생한다. 따라서 1024번의 miss가 발생한다.
또한 a[i]와 a[i*1024]들은 2-way set associative cache를 사용하므로 같은 cache block을 접근하기 위하여 생기는 충돌은 발생하지 않는다.
그러므로 cache miss 횟수는 1024 + 128 = 1152, cache hit 횟수는 896.