EXERCISES -- CACHE MEMORY

1.- Given a computer with split instructions and data caches, of sizes 32KB each and 256B blocks, we want to execute the following C program:

```
#define N 128
int A[N][N];
int B[N][N];
int C[N][N];

for (i=0; i < N; i++)
    for (j=0; j < N; j++)
       C[i][j] = A[i][j] + B[i][j];
```

Assume that matrix A is mapped to address **0x0C000000**, matrices B and C are mapped sequentially after A, variable *i* is stored in a register, and the starting addresses of the matrices are stored in registers.

a) Obtain the amount of cache misses due to read operations (load) and write operations (store) if the cache uses direct-mapping and no-write allocate policies.
b) Repeat the previous calculation for a 2-way set-associative cache, that uses an LRU replacement policy, for the following cases: No-Write allocate and Write allocate. Obtain the performance improvement in each case.
c) A programmer suggests the following modification for the program running in a write-allocate, 2-way set-associative cache.

```
#define N 128
struct {
 int A;
 int B;
} AB[N][N];
int C[N][N];

for (i=0; i < N; i++)
 for (j=0; j < N; j++)
     C[i][j] = AB[i][j].A + AB[i][j].B;
```

Assuming that array AB is mapped to 0x0C000000 and array C is mapped sequentially after it, obtain the amount of read and write misses and compare them with the result of the original program for the same cache configuration.
d) Could we obtain a similar result to the one obtained in the previous item, without changing the program or the cache size?

2.- Given a computer with 64KB of main memory, byte-addressable, and a 2KB cache, with 256B blocks, and HW prefetching (when there is a miss, two blocks are fetched).
Given the following sequence of accesses: 0x3345, 0x1244, 0x3374, 0x5BAC, 0x132A, 0x5C41 y 0x13AF.
Obtain the cache evolution, showing the tags, the misses and the prefetches, under the following situations:
a) Direct-mapping
b) 2-way set-associative cache, with LRU replacement policy
c) Direct-mapping with a 512B buffer where the prefetched block is temporarily stored until it is referenced. The buffer is fully associative with a FIFO replacement policy.

3.- Given a computer with a 4MB main memory, byte-addressable, with a 2-way 4KB cache, 512B blocks and a fully-associative victim cache of 1024B. In both caches, the replacement policy is a FIFO. The initial cache state is the following:

Set	Tag	FIFO state
0	668	0
0	008	1
1	297	0
1	368	1
2	5FF	0
2	668	1
3	297	1
3	200	0

Tag/Set in Main Memory	FIFO state
0A80	0
3FFF	1

Given the following sequence of accesses: 0x334500, 0x14BF00, 0x150084, 0x004021, 0x0540AB y 0x0041F1.
Obtain the cache evolution, showing the tags, misses and transfers among both caches.

4.- Given a computer with 32-bit addresses, with a no-write allocate direct-mapping 128B cache, with blocks of 16B. Assume that variable a is stored in the register file. The following program is executed:

```
int nota[128];    // nota[0] is stored in address 0x00000000
int media[128];   // media [0] is store after nota[127]

for (i=0;i<128;i++) {
       if (i>7 && i<64) {
         nota[i] = media[i]/2;
         }
       else {
         a = nota[i]*media[i]
         }
       }
```

a) Obtain the amount of misses
b) Propose 2 code optimizations and obtain the amount of misses.
c) Assume that: time to access a word in cache is 1ns, time to read or write a word in main memory is 50ns and time to transfer a MM block to cache is 200ns. What is the time needed for all the references in the program?
d) If a second level cache memory with the following features is added to the computer: 1MB, 4-ways, block of 16B, with write allocate and write-back policies. Assuming the time to transfer a block from the second level cache to the first level one is 15ns, what would be the time to access all the references in the program?

5.- We want to execute the following C program in a computer with a byte-addressable 256KB Main Memory, and a 128B cache, with 16B blocks:

```
int A[16][16];
int B[32];
int C[16][16];

for (i=0; i< 16; i++)
 C[0][i] = A[0][i] + B[4];
```

Assuming that the A matrix is stored at address **0x10000**, B and C are stored sequentially after A, and that variables i, j and the pointers to A, B and C are stored in a register:

a. Obtain the amount of cache read (load) and write (store) misses for a direct-mapped write-allocate cache.
b. Obtain the amount of cache read (load) and write (store) misses for a 2-way set-associative write-allocate cache, with a FIFO replacement policy.
c. Could we reduce the amount of misses without changing the program nor the cache size?
d. For the cache configuration of Section a, given an access time of 2ns to the cache and a penalty of 150ns, both for read and write misses, what´s the time needed for performing all the references in the program?