Department of Computer Science and Engineering
Bachelor in Computer Science and Engineering
Computer Architecture

Exam

ATTENTION:

- **Time limit is 150 minutes.**

NAME:
FAMILY NAME:
NIA:

Exercise 1 [3 points]:

A computer has an L1 data cache which is 2 ways set-associative and with a size of 32KB and an L1 instructions cache with the same characteristics. It also has an L2 unified cache which is 4-ways set-associative with a size of 256 KB. In both cases the line size is 64 bytes. It is assumed that hit in L1 cache requires 3 cycles and a hit in L2 cache requires 9 additional cycles and a penalty for bringing a block from main memory to L2 cache of 60 cycles. All caches have a write-back policy.

On this machine we have the following code fragment:

```
struct body {
  double x, y, z;
  double vx, vy, vz;
  double ax, ay, az;
  double m;
};
void f(std::vector<body> & v, double dt) {
  const size_t sz = v.size();
  for (int i=0; i< sz; ++i) {
    v[i].vx += v[i].ax * dt;
    v[i].vy += v[i].ay * dt;
    v[i].vz += v[i].az * dt;
    v[i].x += v[i].vx * dt;
    v[i].y += v[i].vy * dt;
    v[i].z += v[i].vz * dt;
  }
}
```

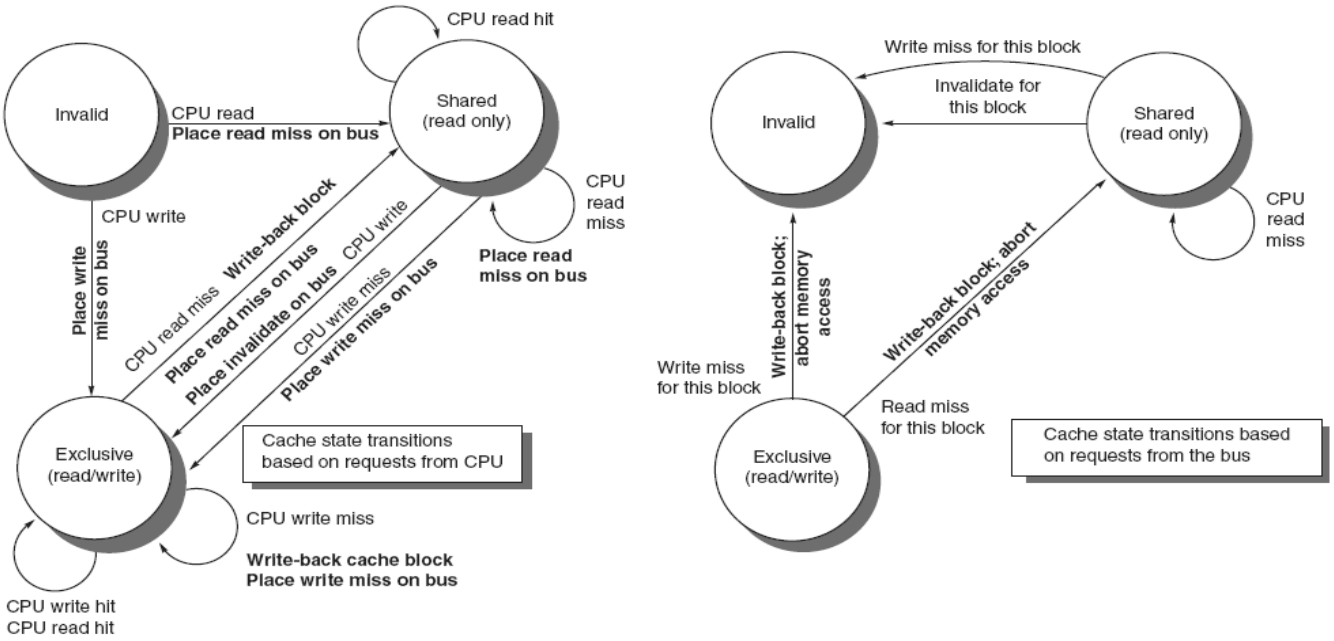
1.1 [0.5 points]: Determine which should be the average access time for data assuming the following miss rates:

- L1 data: 3%
- L2: 1%

1.2 [1.5 points]: Determine the number of misses in data access during the execution of the loop in function f() for cache L1 (initially empty) if vector size is 64 elements. Assume that variables I and sz are allocated to registers and do not generate data accesses. The data block from vector v is assumed to be aligned at 64 bytes limit.

1.3 [1 point]: Propose a modification in code that uses parallel arrays (i.e., eliminating the array merge and using an independent array for each field in the original structure) instead of array merge and determine which would be the new miss rate. Assume that all data blocks from arrays are stored consecutive in memory.

Exercise 2 [2.5 points]: A processor with three cores and symmetric shared memory architecture uses a bus with snooping protocol. Each core has a single level of cache memory which is private for that core and whose coherence is kept by using a MSI protocol. Cache memories use direct mapping and each cache line stores two words.



The following scenarios show different execution situations of different threads in each core. Assume that:

- Each scenario is independent and for each one all cache memories are initially empty.
- Vectors a and b have two entries, are memory aligned storing a single block each one.
- Due to mapping policy, vectors a and b are stored in the same cache line.
- You are asked to specify for each scenario the contents of cache memories from each core across time following the format in the provided table (make one for each scenario). **Besides, you must justify your answers.**

Time	Core 1		Core 2		Core 3		Bus Traffic
	State	Content (a or b)	State	Content (a or b)	State	Content (a or b)	

Universidad Carlos III de Madrid

Department of Computer Science and Engineering
Bachelor in Computer Science and Engineering
Computer Architecture

Exam

- **Scenario 1:**

Time	Thread 1 (running in core 1)	Thread 2 (running in core 2)	Thread 3 (running in core 3)
0	Read a[0]		
1		Read a[0]	
2			Write a[0]

- **Scenario 2:**

Time	Thread 1 (running in core 1)	Thread 2 (running in core 2)	Thread 3 (running in core 3)
0		Write a[0]	
1			Write a[0]
2			Write a[0]
3			Read a[0]

- **Scenario 3:**

Time	Thread 1 (running in core 1)	Thread 2 (running in core 2)	Thread 3 (running in core 3)
0		Read a[0]	
1			Read a[0]
2	Write a[1]		
3			Read a[0]

- **Scenario 4:**

Time	Thread 1 (running in core 1)	Thread 2 (running in core 2)	Thread 3 (running in core 3)
0		Write a[0]	
1		Read b[0]	
2			Write a[0]
3		Write b[0]	
4		Write a[0]	

Exercise 3 [1 point]: Given the following code, answer the questions below:

```
01  void exec(int a){
02    /*
03      * This code needs "a" seconds of CPU to be executed
04      */
05  }
06
07  int main(){
08
09    #pragma omp parallel for
10    for(int i = 0; i < 100; i++){
11              exec(i);
12    }
13
14    return 0;
15  }
```

3.1 [0.5 points]: What is done in line 9? Consider that no environment variable related to OPenMP has be set in the session where the program is running.

3.2 [0.5 points]: Which is the best OpenMP scheduler for directive in line 9 using a chunk size of 10? Justify your answer comparing the behavior of all possible schedulers.

Exercise 4 [1.5 points]: Given the following code, answer the questions below:

```
01  #include <iostream>
02  #include <iomanip>
03  #include <thread>
04  #include <vector>
05
06  double sum;
07  double step;
08  int n_threads;
09
10  void calculate_steps(int thread_id, long nsteps) {
11      for (int i=thread_id; i<nsteps; i=i+n_threads) {
12        double x = (i+0.5) * step;
13        sum += 4.0 / (1.0 + x * x);
14      }
15  }
16
17  int main(){
18      using namespace std;
19
20      // Init global variables
21      n_threads = 4;
22      long nsteps = 100000000;
23      step = 1.0 / double(nsteps);
24      sum = 0.0;
25      std::vector<std::thread> threads;
26
27      // Create threads
28      for(int i = 0; i < n_threads; i++){
29        threads.push_back(std::thread(calculate_steps,i,nsteps));
30      }
31      for (int i = 0; i < n_threads; ++i) {
32         threads[i].join();
33      }
34
35      // Calculate reduction
36      double pi = step * sum;
37      cout << "PI= " << setprecision(10) << pi << endl;
38
39      return 0;
40  }
```

4.1 [0.25 points]: Specify the lines of code composing a critical section. Justify your answer.

4.2 [0.25 points]: Propose a lock based solution to the critical section. Justify your answer.

4.3 [0.5 points]: Propose a lock-free solution to the critical section. Justify your answer.

4.4 [0.5 points]: If the program is run in a 2-core machine. Which from the following solutions should have a better performances using two threads (n_threads = 2). And using 32 threads (n_threads=32). Justify both answers.