In this project, you will use the high-level programming language of your choice to simulate cache coherence protocols for bus-based shared memory multicomputers. The simulation parameters are the number of processors and the coherence protocol. Assume the following:

- 1 GHz dual-issue 32-bit processors with two 5-stage pipelines which access data memory (if necessary) during the 4th stage
- pipelines stall only for cache misses and if either pipeline in a processor is stalled for a cache miss, both pipelines stall
- 10% of the instructions are store word and 10% are load word
- each processor has 1 instruction cache and 1 data cache
- all caches are 64 KB in size, are direct mapped, and have a block size of 16 words
- each processor executes a single thread, the code for all threads is identical and begins at address 0, thread initiation times on the various processors are spaced 1000 cycles apart, and thread execution time (before considering processor stalls) is 100,000 processor cycles
- the processor-memory bus is 64 bits wide and operates at 500 MHz
- main memory is 64 bits wide, has 256 MB total size, and has the following organization and properties:
  - the memory is word-addressed for single word writes with write-through caches
  - for block transfers, the memory is double-word-addressed
  - access time is 10 ns for the first double word in a block and 2 ns for each additional double word in a block
  - the thread code is stored from word address 0 to word address 4M – 1
  - shared variables are stored from word address 4M to word address 32M – 1
  - the rest of the memory is split evenly among the processors for private variables, i.e. the remaining blocks are not shared among processors at all

Threads access memory according to the following model:

- instructions are always accessed in pairs and both instructions in a pair can always be issued simultaneously
- the double-word address of the next instruction pair is PC+2 with probability 0.9 and is uniformly distributed over the entire thread code with probability 0.1
- the first load word or store word accesses word address 4M
- subsequent word addresses to be operated on by load word and store word instructions are chosen as follows:
• with probability 0.6, the address is k+1, where k was the word address for the last load word or store word instruction

• with probability 0.2, the address is uniformly distributed over the thread’s private variable space

• with probability 0.1, the address is uniformly distributed over the shared variable space

• with probability 0.1, the address is 4M

A few last notes about the system you will simulate. You will need to assume a bus protocol and bus arbitration scheme. Any of several choices are acceptable in this regard (see, for example, Patterson and Hennessy, Chapter 8). Be sure to describe your selection in the project report. To simplify the simulation program, you should assume that the system is synchronous, i.e. the simulation can proceed in steps of a single global clock cycle in which each processor completes two instructions (unless it is waiting for a cache miss to be handled). Every two global cycles equals one bus clock cycle.

In your simulations, you should set the number of processors to both 4 and 8 and you should simulate:

1) the simple cache-coherent protocol with write-through caches

2) the MESI protocol with write-back caches

3) the Dragon protocol with write-back caches

For each choice of protocol and number of processors, run your simulation program to completion 10 times and calculate, in clock cycles, the mean completion time and mean cache miss penalty. Write a 5-page report that describes your simulation program, and presents and interprets the results. Include your source code as an appendix.