CC-UMA-Simulator Report

Junchen (Daniel) Lu, Nov 9th, 2023

INTRODUCTION:

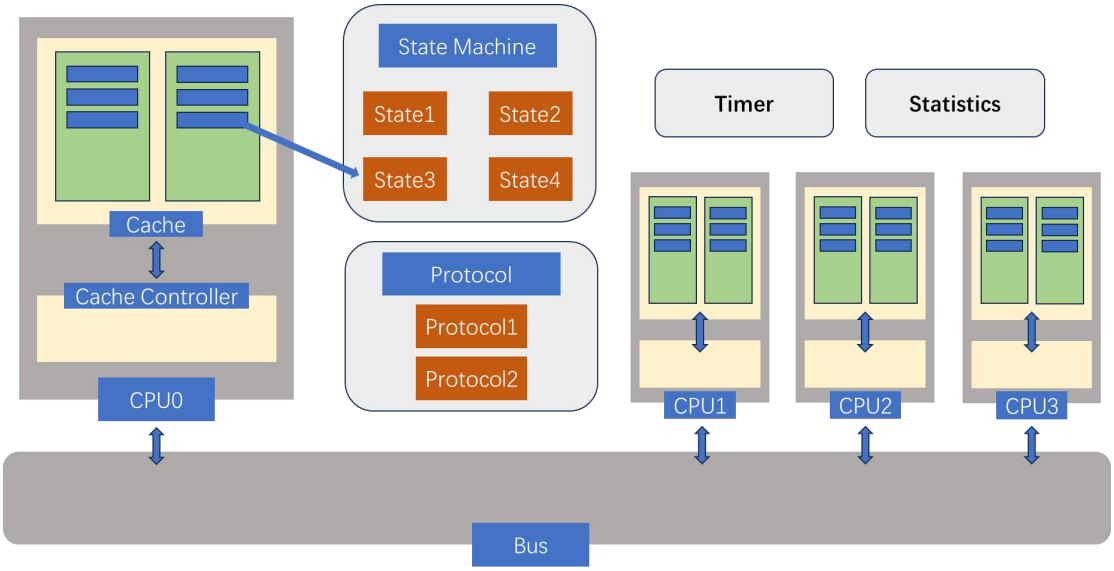
The CS4223 mini-project assignment focuses on cache coherence protocols and simulator development. The goal of the project is to enhance understanding of cache coherence protocols and how simulators are built. The simulator will be trace-driven, meaning it will execute benchmark traces provided from the PARSEC benchmark suite. The benchmark traces include blackscholes, bodytrack, and fluidanimate. The assignment includes implementing MESI and Dragon cache coherence protocols, analyzing the protocols quantitatively, and optionally implementing an optimization to the protocols and evaluating its impact.

IMPLEMENTATION:

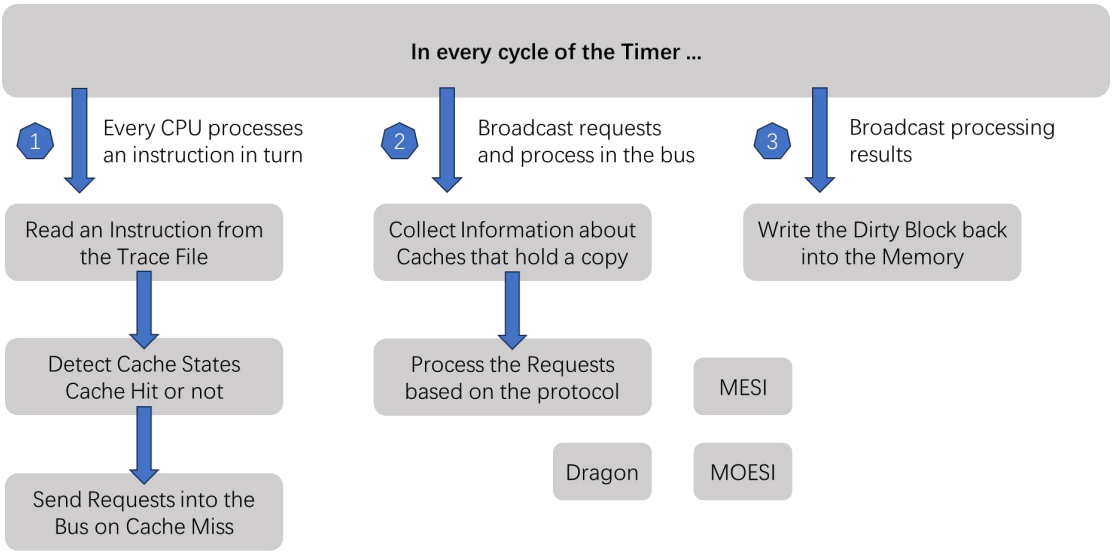
Programming Language: C++

Development Environment: Linux Ubuntu Server

Data Structures and Classes Defined:



Flow Charts:



OPTIMIZATION (MOESI) ANALYSIS:

1. **Modified**: on processor read or write, do nothing and stay in Modified state
2. **Exclusive**: on processor read, do nothing; on processor write, transfer to Modified state; in case 1 and 2, there should be only one copy of data in the system
3. **Shared**: up to two types of states can exist in the system, if owned doesn't exist, cache data is coherent with memory; if owned exists, only owned is responsible for writing data back to the memory; on processor read, do nothing; on processor write, invalidate all the other copies and transfer to Modified state
4. **Owned**: on processor read, do nothing; on processor write, invalidate all the other copies and transfer to Modified state; only the dirty bit of owned state should be set to true, not the Sharers
5. **Invalid**: on processor write, invalidate all the other copies and transfer to Modified state; on processor read, decide the states among Owned/Shared/Exclusive, but the return value needn't take "write back to memory" into consideration

RESULTS:

In this section, three protocols (MESI, MOESI, Dragon) will be discussed and they will be compared from different dimensions. The question that which protocol is better for each

benchmark will also be investigated.

Present the Results Graph:

The "blackscholes" simulation chart presented the cycles and data traffic for three different coherence protocols: MESI, MOESI, and Dragon. Here is the analysis and comparison from both angles:

1. In terms of "sum execution time":

- The MESI protocol completed its simulation in 79,586,954 cycles.
- The MOESI protocol was marginally more efficient, completing its run in 79,565,834 cycles, which is 21,120 cycles fewer than MESI.
- The Dragon protocol showed a slightly higher cycle count than MESI, finishing at 79,591,513 cycles, which is 4,559 cycles more than MESI.

From a cycle count perspective, the MOESI protocol is the most efficient, with the lowest cycle count indicating a potentially faster completion time for the simulation compared to MESI and Dragon protocols. The differences between the cycle counts are minimal, suggesting that all three protocols have similar efficiency in terms of computation time.

2. In terms of "data traffic on the bus":

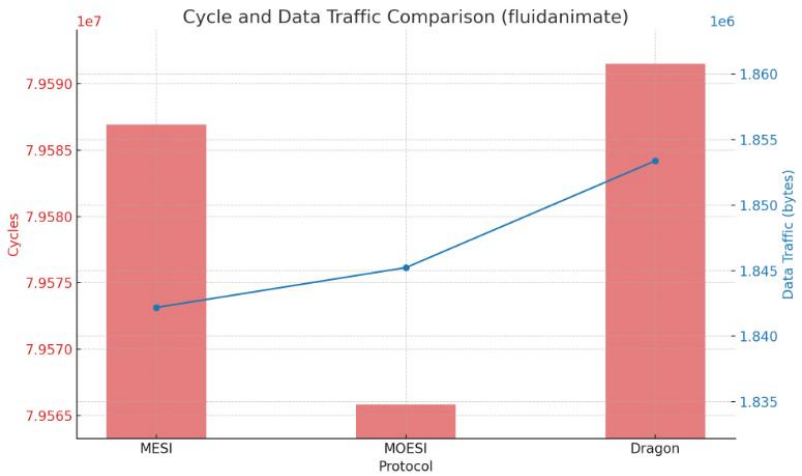
- The MESI protocol generated 1,842,192 bytes of data traffic.
- The MOESI protocol generated slightly more data traffic at 1,845,232 bytes, an increase of 3,040 bytes over MESI.
- The Dragon protocol produced the most data traffic at 1,853,384 bytes, which is 11,192 bytes more than MESI and 8,152 bytes more than MOESI.

In terms of data traffic, MESI is the most economical, generating the least amount of data. This

could be advantageous in systems where bandwidth is limited or where minimizing data traffic is a priority. The Dragon protocol, while having a comparable cycle count to the others, generates the most data traffic, which might be a drawback in bandwidth-constrained environments.

3. Overall Comparison:

MOESI offers a slightly better performance in terms of cycle efficiency but generates slightly more data traffic than MESI. The Dragon protocol, while not as cycle-efficient as MOESI, finishes in a comparable number of cycles to MESI but with higher data traffic. The trade-offs between these protocols would depend on the specific requirements of a system, such as whether cycle efficiency or data traffic minimization is more critical.



For the "fluidanimate" simulation, we can analyze the performance based on cycles and data traffic as follows:

1. In terms of “sum execution time”:

- The MESI protocol completed its simulation in 79,586,954 cycles.
- The MOESI protocol finished very close to MESI with 79,565,834 cycles, only 21,120 cycles less.
- The Dragon protocol completed its simulation in 79,591,513 cycles, slightly more than MESI, with an additional 4,559 cycles.

In terms of cycle efficiency for the "fluidanimate" simulation, the differences among the three protocols are very narrow, indicating similar performance. MOESI had a slight edge with the fewest cycles, suggesting it may be marginally more efficient in terms of processing time.

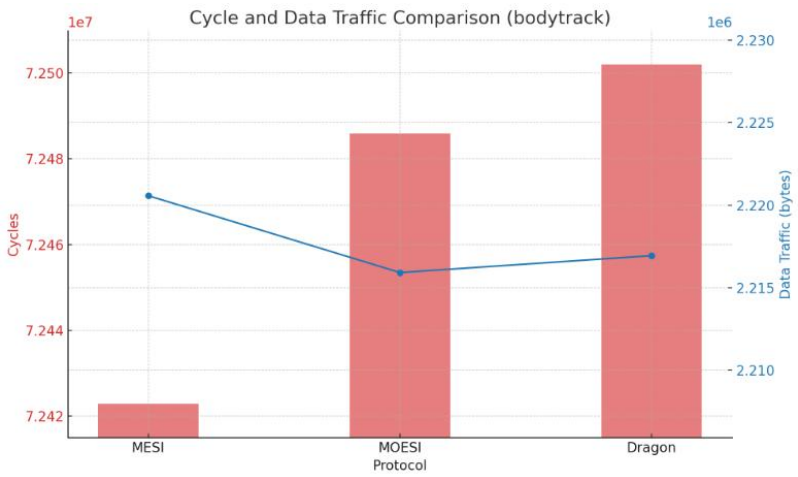
2. In terms of “data traffic on the bus”:

- MESI generated 1,842,192 bytes of data traffic.
- MOESI generated a bit more at 1,845,232 bytes, which is 3,040 bytes above MESI.
- Dragon produced the highest data traffic at 1,853,384 bytes, which is 11,192 bytes more than MESI and 8,152 bytes more than MOESI.

From the perspective of data traffic, MESI was the most efficient, generating the least amount of data. This might be preferable in systems where data traffic needs to be minimized. Dragon, while comparable in cycle count, generated the most data traffic, which could be less ideal in systems where bandwidth is a concern.

3. Overall Comparison:

For the "fluidanimate" simulation, MOESI and MESI are very close in both cycles and data traffic, with MOESI being slightly more cycle-efficient but also generating a slightly higher amount of data traffic. Dragon is comparable in cycle count to MESI but generates more data traffic. The choice between these protocols would likely depend on the specific optimization requirements of the system in question, whether that's cycle efficiency or data traffic minimization.



For the "bodytrack" simulation, the performance based on cycles and data traffic is as follows:

1. In terms of “sum execution time”:

- MESI protocol finished its simulation in 72,422,886 cycles.
- MOESI protocol completed the simulation in 72,485,909 cycles, which is slightly higher by 63,023 cycles compared to MESI.
- Dragon protocol ended with 72,501,973 cycles, the highest among the three, with a difference of 79,087 cycles from MESI.

Here, the MESI protocol is the most cycle-efficient, completing the simulation with the fewest cycles. Although the differences are relatively small, in high-performance computing environments, even small efficiencies can be significant. MOESI and Dragon are closely matched, but Dragon has the highest cycle count, which suggests it is the least efficient in processing time for the "bodytrack" simulation.

2. In terms of “data traffic on the bus”:

- MESI protocol generated 2,220,576 bytes of data traffic.
- MOESI protocol produced slightly less data traffic at 2,215,920 bytes, a difference of 4,656 bytes less than MESI.
- Dragon protocol generated 2,216,944 bytes of data traffic, which is marginally higher than MOESI by 1,024 bytes but still less than MESI.

For data traffic, MOESI is the most economical, generating the least amount, followed very closely by Dragon. MESI, while being the most cycle-efficient, generated the most data traffic among the three.

3. Overall Comparison:

In the "bodytrack" simulation, there is a trade-off between the cycles completed and the data traffic generated. MESI is the most efficient in terms of cycle count but generates the most data traffic. MOESI, while slightly less efficient in cycle count, is the most efficient in terms of data traffic. The Dragon protocol is less efficient than both MESI and MOESI in terms of cycle count but is more efficient than MESI in data traffic. The optimal protocol would depend on whether the priority is on reducing the cycle count or minimizing data traffic.

CONCLUSION:

In comparing the MESI, MOESI, and Dragon cache coherence protocols across the simulations, it's evident that cycle count and data traffic efficiencies are closely contested. MOESI often leads marginally in cycle efficiency, while data traffic generation varies, with MESI typically being the least efficient. These small differentials underscore that each protocol's effectiveness is workload-dependent, and the choice among them would hinge on the specific balance between cycle time optimization and data traffic minimization required by the application.