**Blue Waters Petascale Semester Curriculum v1.0**

**Unit 9: Optimization**

**Lesson 4: Multiprocessor Caching and False Sharing**

**Sample Assessment**

*Developed by* *David P. Bunde for the Shodor Education Foundation, Inc.*



*Except where otherwise noted, this work by The Shodor Education Foundation, Inc. is licensed under CC BY-SA 4.0. To view a copy of this license, visit*[*https://creativecommons.org/licenses/by-sa/4.0*](https://creativecommons.org/licenses/by-sa/4.0)

*Browse and search the full curriculum at*[*http://shodor.org/petascale/materials/semester-curriculum*](http://shodor.org/petascale/materials/semester-curriculum)

*We welcome your improvements! You can submit your proposed changes to this material and the rest of the curriculum in our GitHub repository at*[*https://github.com/shodor-education/petascale-semester-curriculum*](https://github.com/shodor-education/petascale-semester-curriculum)

*We want to hear from you! Please let us know your experiences using this material by sending email to* [*petascale@shodor.org*](mailto:petascale@shodor.org)

1. Draw diagrams showing shared caching and private caching for a quad core system. Explain the difference between them.
2. Explain the cache coherence problem for private caches and why multicore systems need to solve it.
3. Define false sharing.
4. For each of the code segments below, explain whether false sharing is likely to occur and any steps that could be taken to reduce or eliminate it. (Note that neither “yes” nor “no” is an explanation.) Recall that when OpenMP runs a sections block, each section sunblock is assigned to a single thread. For all the code segments, assume that array and count are pre-allocated integer arrays sufficiently long for the way they are used.
   1. #pragma omp parallel sections

{

#pragma omp section

{

for(int i=0; i<num/2; i++)

if(array[i] == 0)

count[i/5]++;

}

#pragma omp section

{

for(int i=num/2; i<num; i++)

if(array[i] == 0)

count [i/5]++;

}

}

* 1. #pragma omp parallel sections

{

#pragma omp section

{

for(int i=0; i<num; i+=2)

if(array[i] < 0)

count[0]++;

}

#pragma omp section

{

for(int i=1; i<num; i+=2)

if(array[i] < 0)

count[1]++;

}

}

* 1. #pragma omp parallel sections

{

#pragma omp section

{

int ones = 0;

for(int i=0; i<num; i++)

if(array[i] == 1)

ones++;

printf(“Number of ones: %d\n”, ones);

}

#pragma omp section

{

int twos = 0;

for(int i=0; i<num; i++)

if(array[i] == 2)

twos++;

printf(“Number of twos: %d\n”, twos);

}

}