ECE563 Final Exam, Spring 2009

NAME ____________________________

1. (10+ points) What is a typical overhead, in milliseconds, for
   (a) an OMP PARALLEL construct __________
   (b) fork/join of a thread in pthreads __________
   (c) Unix-type fork() ___________
   (bonus) In all three cases, explain briefly where you think the majority of this time gets spent.

   ____

   ____

   ____

2. (5 points) Mark all correct answers:
   __ The tasking model in pthreads is less dynamic/flexible than in OpenMP.
   __ In MPI programs, the amount of false sharing depends on the programmer.
   __ In a system of networked multicores, it makes sense to use OpenMP to exploit parallelism within nodes and MPI across nodes.
   __ A typical MPI programs executes one fork and one join operation.
   __ In OpenMP, a parallel region must contain at least one parallel loop or one parallel section.
   __ MPI is used for messaging within CUDA.

3. (5 points) Mark all correct answers:
   __ In practice, it is not possible to use standard programming languages to write efficient programs for parallel machines.
   __ Parallelizing compilers are most successful on regular, science/engineering applications.
   __ Parallelizing compilers are successful on irregular integer applications.
   __ For high-resolution video games, multicores may work efficiently at executing full-physics simulations.
   __ When parallelism can be hidden in libraries, efficient parallel programs can be written with high productivity.

4. (15 points) Convert the following program into MPI parallel form.
   The precise syntax of parallel constructs is not essential.
   Assume that arrays X and Y exist in block distribution.
   State all other assumptions you make.

```
SUBROUTINE FinalExam(X,Y,Z,sum,n)
REAL X(1000), Y(1000), Z(1000), sum, tmp
INTEGER n, i
DO i=1,n
   tmp = X(i)+Y(i)
   Z(i) = tmp + tmp**2
   sum  = sum + Y(i)
ENDDO
END
```

5. Consider the following loop

```
DO i=1,n {
   k += a[k]+1;
   b[k] = c[i];
}
PRINT k;
```

a) (5 points) Is k an induction variable? Explain why or why not.

b) (5 points) If your answer to (a) was yes: write the parallel code. If the answer was no: can you write this code so that it exploits partial parallelism? Do so or explain why not.

6. (10 points) Barak is a novice parallel programmer. His serial program exhibits a small amount of input/output. However, after some parallelization work, the profile of the program suddenly shows that IO dominates. Barak is worried that his disk became defective. What advice would you give him?

7. Consider the following loop and its parallel transformation

   Serial:
```
     DO i=1,n
        call bigcomputation(i)
        sum = sum + B(i)
     ENDDO
```

   Parallel:
```
   C OMP DO PARALLEL
     DO i=1,n
        call bigcomputation(i)
        wait(i-1)
        sum = sum + B(i)
        post(i)
     ENDDO
  C OMP END PARALLEL
```

   Assume that `Bigcomputation` takes milliseconds of execution time.

   a) (5 points) Explain the parallelism this program exhibits

   b) (5 points) Explain the efficiency at which the program can exploit multicores

8. (10 points) Consider GPU and CUDA. Mark all true answers:
   __ Similar to OpenMP, where serial and parallel regions alternate, CUDA programs can alternate between CPU and GPU code.
   __ Unlike OpenMP programs, CUDA programs may require substantial communication at the boundaries between CPU and GPU code.
   __ The code executed on the GPU device is called *kernel function.*
   __ Code that exhibits stride-1 data accesses does not run efficiently on GPUs
   __ Programs that are suitable for execution on GPUs may run more than an order of magnitude faster than on common multicores.
   __ Irregular programs do not run efficiently on GPUs.
   __ GPUs are faster because they contain more transistors on the chip.