Hadassah Academic College
Department of Computer Science

Advanced Computer Architecture Exercise 7

1. Consider the following code to numerically integrate

$$\int_0^1 \frac{4}{1+x^2}\,dx = \pi$$

```
static long num_steps = 100000;
double step, pi;
void main()
{  int i;
   double x, sum = 0.0;
   step = 1.0/(double) num_steps;
   for (i=0; i < num_steps; i++)
   {
      x = (i+0.5)*step;
      sum = sum + 4.0/(1.0 + x*x);
    }
   pi = step * sum;
   printf("Pi = %f\n",pi);
}
```

A. Using OpenMP methods, write a parallel version of the code. Leave undetermined the number of CPUs and threads as N.

B. Assume that after compilation for the simple DLX (with 1 CPU) the code divides into a header, loop, and tail as follows:

Head	5 CC
Loop	10 CC per iteration
Tail	100 CC (**printf** library call)

Assuming that the parallel version runs without additional overhead, find the number of clock cycles to run the program as a function of N.

C. Now assume that the overhead in the parallel version is as follows:

Fork	4 CC
Communication	Each processor loads instructions from RAM to cache over shared bus 1 CC per instruction per processor
Join	Each processor writes partial result to RAM over shared bus 1 CC per processor

Find the number of clock cycles to run the program as a function of N, given the overhead. Find the speedup as a function of N.

C. Find the optimum number of threads.

D. Find the speedup for the optimum number of threads.

2. An SMP computer has 2 simple (single core) DLX CPUs with CPI = 1. Ignore stall behavior except for a cache miss that causes a miss penalty of 10 CC.

A. The following code runs on the SMP. Assuming interleaved execution, show (as in the example) the order of instruction executions, with an emphasis on cache behavior. At the start of execution, there are no blocks in cache. Find the number of clock cycles to run the 2 threads.

```
L: TEST_AND_SET mutex, R1
   BNEZ R1, L
   LW R2, x                    ; critical section
   ADDI R2, R2, #1
   SW x, R2
   SWAP mutex, R1
```

CPU 0		CPU 1	
L: TEST_AND_SET	Cache allocate stall 10 CC CPU 0 ← Mutex = 0 R1 ← 0 Mutex ← 1		
		L: TEST_AND_SET	Cache allocate stall 10 CC CPU 1 ← Mutex = 1 R1 ← 1
Total CC =		**Total CC =**	
Total CC =			

B. The following code runs on the SMP. Assuming interleaved execution, show (as in the example) the order of instruction executions, with an emphasis on cache behavior. At the start of execution, there are no blocks in cache. Find the number of clock cycles to run the 2 threads.

```
L: LOAD-RESERVE R2, x
   ADDI R2, R2, #1             ; critical section
   STORE-CONDITIONAL x, R2
   BEQZ status, L
```

CPU 0		CPU 1	
L: LOAD-RESERVE R2, x	Cache allocate x, flag stall 10 CC CPU 0 R1 ← x flag ← 1		
		L: LOAD-RESERVE R2, x	Cache allocate x, flag stall 10 CC CPU 1 R1 ← x flag ← 1
Total CC =		**Total CC =**	
Total CC =			

C. What is the largest number of instructions (without memory accesses) that can be put into the critical section, such that the second version will be more efficient than the first.