Multithreading and Multiprocessing

1) [15 points] Assume you have a 2-wide superscalar core with dynamic instruction scheduling. Assume that the dynamic scheduler is optimal - it chooses the very best possible schedule (instead of, say, scheduling the oldest available instructions first). Assume that an instruction must wait for 2 cycles before consuming the data provided by a load instruction. For example, if a load writes to r6 on cycle 1, then an instruction cannot consume r6 until cycle 3. Assume that all instructions take one cycle and that data-dependent instructions cannot execute on the same cycle.

(a) Show the execution of Thread 1 on this core - that is, on what cycle does each instruction execute. What is the utilization of the core, measured in terms of IPC/(superscalar width)? Do the same thing for Thread 2.

(b) Show the simultaneous execution of both Thread 1 and Thread 2 on an SMT core. Assume that the core can fetch instructions from both threads during the same cycle. The core is identical to the core for part (a), except that it has 2 thread contexts. What is the utilization of the core now?

Thread 1

```
add r1, r2, r3 // r1=r2+r3
load r4, A // r4=Mem[A]
sub r5, r1, r4
mul r6, r5, r2
```

Thread 2

```
load r11, B
load r12, C
xor r13, r11, r12
and r16, r13, r12
```

2) [15 points] In approximately one page, discuss the Multiscalar paper. Provide a brief summary, describe its contributions to the field of computer architecture, and discuss at least one weakness of the paper.

3) [10 points] A multicore chip has 16 cores. If they are connected together in a 2D torus, what is the average number of hops that a message from one core must travel before reaching another core? Explain your answer.

4) [15 points] Assume a CMP with in-order cores, blocking private caches, and no write buffers, LSQ, or other buffering between the core and the cache (i.e., a core’s cache sees and performs memory requests precisely in program order). Write a shared memory program (on paper, not on a computer) with two threads that works correctly (produces the desired result) if cache coherence is maintained and that might not work correctly without cache coherence. You must explain why it might not work if cache coherence isn’t maintained. Please keep your program as small as possible, to simplify grading it.
5) [10 points] What are the advantages and disadvantages of having heterogeneous (asymmetric) cores on a chip, as opposed to a homogeneous multicore chip (where all cores are identical)?