1 Processor (25 points)

(a) (7 points) Why do loads, that are issued out of program order, search the load-store queue at commit?

(b) (8 points) Give an example of code pattern where compiler scheduling techniques like trace scheduling and software pipelining would perform **better** than hardware scheduling using out-of-order issue. You may state general code characteristic **without** detailed code. Examples such as regular, parallel loops where hardware would do as well as compiler will **not** receive any points.

(c) (10 points) Assume that a processor has two adders each of which takes one cycle for an add and two multipliers each of which takes four cycles for a multiplication. Assuming that 25% of instructions in a program are multiplies and the rest are adds, compute the range of CPI (cycles per instruction) that may be achieved by the program, given that there are no delays in the system other than the above operational latencies. State the conditions under which the end-points of your range would be achieved.

2 Memory hierarchy (25 points)

(a) (10 points) Describe a hardware optimization to improve the performance of the following code for large values of N **without** modifying the cache or the code. You may just describe the optimization at a high level **without** providing details.

```
for (i=0; i<N; i++) {

    B[i] = f (A[i]);  /* A[] and B[] incur both capacity and conflict misses */

}
```

(b) (8 points) What is the purpose of lock-up free caches (or non-blocking caches)?

(c) (7 points) Why do modern cache hierarchies enforce inclusion?

3 Virtual Memory (25 points)

(a) (10 points) Assuming a 64-bit virtual address space which is populated sparsely and **uniformly**, how would you organize the page table?

(b) (10 points) Modern L2 caches are not flushed upon context switches. What feature of the cache enables this avoidance and why does the feature not slow down cache access?

(c) (5 points) Why do TLBs use process IDs (PIDs)?

4 Fundamentals (25 points)

Assume that the solid-state area at Purdue ECE has developed a new process technology which provides faster transistors than conventional process. However, the new technology is such that while activating one circuit block consumes some energy, **simultaneously** activating multiple circuit blocks consumes exponentially more energy (e.g., n circuit blocks consume 2^n times the energy of activating one block). Because of the higher speed, we wish to build computer systems using the new technology.

(a) (5 points)

What feature that is considered to be good in unlimited amounts in conventional systems needs to be controlled carefully in the new systems?

(b) (10 points)

What is the opportunity for reducing energy without hurting performance in the new systems. You may describe the opportunity through a simple example.

(c) (10 points)

Assuming that the workloads of interest can be scheduled by the compiler and the operation latencies are known to the compiler, what would be your scheduling strategy to maximizing performance while reducing energy? You may draw a figure to explain. Any strategy that reduces energy but results in poor performance will **not** receive any points.