COE 501: Computer Architecture

Problem Set 4: Caches and Virtual Memory

1) (11 pts) The transpose of a matrix interchanges its rows and columns. Here is the code:

```
for (i=0; i<N; i++)
  for (j=0; j<N; j++)
    output[j][i] = input[i][j];
```

Both the input and output matrices are stored in row-major order. Assume that you are executing *N*×*N* double-precision (8 bytes per element) matrix transpose on a processor with 16 KB D-Cache, which is 2-way set-associative, and 64-byte blocks. The D-Cache is a write-back with write-allocate policy on a write miss.

a) (2 pts) Assume each set in the D-Cache stores one block from the input matrix and a second block from the output matrix. How many sets exist in the D-Cache? What is the maximum value of *N* such that both the input and output matrices can fit in the 16-KB D-Cache?

b) (3 pts) A compulsory cache miss occurs when a block is referenced for the first time. Given that *N*=16, how many cache misses are caused in the 16 KB 2-way set associative cache? If each cache miss stalls the processor for 8 cycles (assuming hit in L2 cache) then what is the total number of stall cycles for matrix transpose when *N*=16? What is the impact on the CPI if the execution CPI = 1.1 (excluding cache misses)? Assume six instructions are fetched and executed per inner loop iterate plus 2 instructions per output loop iterate.

c) (2 pts) If *N* is large then cache blocks are replaced, even though they are still needed. They are later brought into the cache causing additional cache misses. Estimate the total number of cache misses as a function of *N*, given that the cache block size is 64 bytes.

d) (4 pts) Loop interchange does not work for matrix transpose, because either the input matrix or the output matrix will be traversed by column, which is non-ideal. Transform the above code to perform matrix transpose, which uses $B \times B$ blocks (Block parameter *B*).

2) (12 pts) A processor with in-order execution runs at 1 GHz and has a CPI of 1.2 without counting the memory stall cycles. The only instructions that access memory are loads (20% of all instructions) and stores (5% of all instructions). The memory system consists of an I-cache and a D-cache that each has a hit time of 1 clock cycle.

The I-cache has a 2% miss rate and 32-byte blocks. The D-cache is write-through with a 5% miss rate for reads and 16-byte blocks. There is a write buffer on the D-cache that eliminates stalls for 99% of all writes. The 512 KB write-back, unified L2 cache has 64-byte blocks and an access time of 15 ns. It is connected to the I-cache and D-cache by a 128-bit data bus that runs at 400 MHz and can transfer 128 bits (16 bytes) per L2 bus cycle. Of all memory references sent to the L2 cache in the system, 20% miss in the L2 cache and require main memory access. The main memory has an access latency of 60 ns, after which any number of bytes many be transferred at the rate of 16 bytes per cycle on the 128-bit-wide main memory data bus, running at 200 MHz.

a) (3 pts) What is the average memory access time for instruction fetching?
b) (3 pts) What is the average memory access time for data reads?
c) (3 pts) What is the average memory access time for data writes?
d) (3 pts) What is the overall CPI, including memory stall cycles?

3) (9 pts) Cache organization is often influenced by the desire to reduce the cache's energy consumption. For that purpose we assume that the cache is physically distributed into a data array (holding the data), tag array (holding the tags), and replacement array (holding information needed by replacement policy). In addition, each one of these arrays is physically distributed into multiple sub-arrays (one per way) that can be individually accessed. For example, a four-way set associative LRU cache has four data sub-arrays, four tag sub-arrays, and four replacement sub-arrays. The replacement sub-arrays should be accessed on every cache access when the LRU replacement is used. However, it is not needed when Random replacement is used. For a specific cache, it was determined that the accesses to the different arrays have the following energy consumption:

Array	**Energy Consumption Per Way Accessed**
Data Array	20 energy units
Tag Array	5 energy units
Replacement Array	1 energy unit

Estimate the energy consumption for the following configurations. The cache is 4-way set associative. Main memory access and cache refill (although important) are not considered here. Provide answers for the LRU and Random replacement policies.

a) (3 pts) A cache read hit. All arrays are read simultaneously.

b) (3 pts) The cache access is now split across two cycles. In the first cycle, all tag sub-arrays are accessed. In the second cycle, only the sub-array whose tag matched will be accessed. Repeat part (a) for a cache read hit.

c) (3 pts) Repeat part (b) for a cache read miss, assuming that no data array access in the second cycle for a cache miss.

4) (8 pts) The following table shows parameters of a virtual memory system:

Virtual Address	Maximum Physical Memory	Page Size	Page Table Entry
50 bits	64 GB	16 KB	4 bytes

a) (2 pts) For a single-level page table, how many page table entries are needed? How much physical memory is needed for storing the page table?

b) (2 pts) Using a multilevel page table can reduce the physical memory allocation of page tables. How many levels of page tables will be needed, given the size of the page table at any level is the size of a page?

c) (2 pts) The architect wants to support a large page size, what will be the best choice for a large page size and why?

d) (2 pts) The architect wants to design a 64 KB cache that should be indexed in parallel with the TLB (address translation). The cache should be physically tagged and should not have any aliasing problem. What should be the minimum associativity of the cache to avoid aliases?