CSE 30321 – Computer Architecture I – Fall 2010

Homework 08 – Multiprocessing

Assigned: November 30, 2010**, Due:** December 9, 2010, **Total Points:** 60

Problem 1: (20 points)

You're trying to decide whether or not you should buy a laptop with a dual core processor or a single core processor. The specs for each machine – and benchmark data – are listed below.

To execute your application of choice on the 2 GHz **dual core machine**, you will need to execute 2,000,000,000 total instructions:

- 60% of the instructions will run on Core 1
- 40% of the instructions will run on Core 2
- The instructions will run in parallel.

Of the instructions that run on Core 1:

- 20% are branch instructions
- 40% are ALU instructions
- 40% of the instructions reference memory.

Of the instructions that run on Core 2:

- 30% are branch instructions
- 40% are ALU instructions
- 30% of the instructions reference memory

On the dual core machine:

- A branch instruction requires 7 CCs to execute
- An ALU instruction requires 10 CCs to execute
- A memory reference instruction requires:
    - 8 CCs if there is an L1 cache hit.
    - 8 CCs + 14 more CCs if there is an L1 cache miss but data is found in the L2 cache
    - 8 CCs + 14 CCs + 100 more CCs if the memory reference misses in L1 and L2.

On both cores, the L1 miss rate is 10% and the L2 miss rate is 7%

To execute your application of choice on the 2.5 GHz **single core machine**, you will need to execute 1,600,000,000 total instructions:

- 1/3rd of the instructions are branch instructions
- 1/3rd of the instructions are ALU instructions
- 1/3rd of the instructions reference memory.

On the single core machine:

- A branch instruction requires 6 CCs to execute
- An ALU instruction requires 6 CCs to execute
- A memory reference instruction requires:
    - 8 CCs if there is an L1 cache hit.
    - 8 CCs + 14 more CCs if there is an L1 cache miss but data is found in the L2 cache
      8 CCs + 14 CCs + 150 more CCs if the memory reference misses in L1 and L2.
- The L1 miss rate is 15% and the L2 miss rate is 2%

For this benchmark suite, which processor is better? Why?

Problem 2: (10 points)

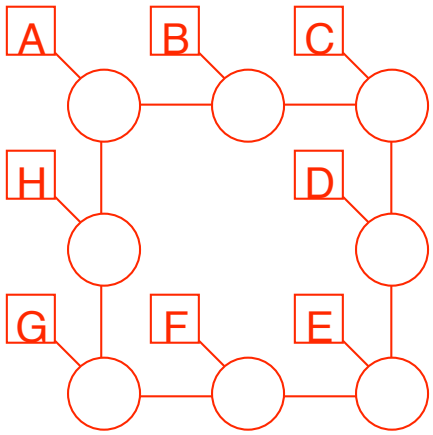
Assume that different processor cores are connected via an on-chip ring network (see below).

- Note: Circles are routers, squares (with letters) are cores

Part A: (5 points)

How long does it take to send a 140-bit message from terminal node A to terminal node D?

Assume that the time required to traverse a router is 2 CCs, and that it takes 1 CC to traverse the distance between routers. Assume that there are 64 bits of available bandwidth between terminals.



Part B: (5 points)

Assume that all 8 cores are working in conjunction to solve a problem.

- The amount of *computation* (not "overhead") associated with the job that each core must do requires 100,000 CCs per core
- Each core must send 500 messages to a core 4 routers away (use the number of clock cycles calculated above) to facilitate parallelization.
- What speedup is obtained?
- What speedup is obtained if the communication overhead is the same, but each core instead requires 500,000 CCs of computation per core?

Problem 3: (10 points)

Assume that you are running a program on a GPU similar to the NVIDIA GeForce 8 and/or GeForce 500 series. The GPU has:

- 16 *multiprocessors* (MPs)
- Each *multiprocessor* is composed of 16 *streaming processors* (SP).
- It is possible to execute just 1 floating point instruction / SP / clock cycle
- On average, the GPU achieves 85% of its peak performance

The GPU is supported by a memory:

- That is 2 GBytes
- That is 8 bytes wide – i.e. each can send 8 bytes / CC
- That operates at 1.2 GHz

Each floating point operation on average needs 6 bytes of data

Question A: (5 points)

To supply the processor with needed data, how should the memory be organized?

Question B: (5 points)

Recall from Lecture 23, I noted that common Byte:FLOP ratios for supercomputing applications include: 1 byte:1 FLOP and 0.3 bytes:1 FLOP. How would your answer to Part A change given these ratios?

Problem 4: (20 points)
Consider the multiprocessor cache and memory state shown in the tables below:

CACHES

P0:

Block Number	Coherence State	Address Tag	Data	
Block 0	I	100	00	10
Block 1	S	108	00	08
Block 2	M	110	00	30
Block 3	I	118	00	10

P1:

Block Number	Coherence State	Address Tag	Data	
Block 0	I	100	00	10
Block 1	M	128	00	68
Block 2	I	110	00	10
Block 3	S	118	00	18

…

P7:

Block Number	Coherence State	Address Tag	Data	
Block 0	S	120	00	20
Block 1	S	108	00	08
Block 2	I	110	00	10
Block 3	I	118	00	10

MEMORY

Address	Data	
100	00	00
108	00	08
110	00	10
118	00	18
120	00	20
128	00	28
130	00	30

Each part of this exercise specifies a sequence of one or more CPU operations in the form:
P#: <op> <address> [← <value>]

Here:

"P#"	designates the CPU number (e.g. P0)
"<op>"	is the CPU operation (e.g. read or write)
"<address>"	denotes the memory address
"<value>"	indicates the new word to be assigned on a write operation

Question A: (15 points)
Given the initial state shown above, ***what is the resulting state (i.e. coherence state, tags, and data) of the caches and memory after the given action?*** Show only the blocks that change.

For example, "P0.B0: (I, 120, 00 ll 01)" indicates that CPU PO's block B0 has the final state of I, tag of 120, and data words 00 and 01.

Also, ***what value is returned by each read operation?***

You should treat each part below independently (i.e. as you begin a new part, you should assume the same, initial state).

Part 1:
p0: read 100
p0: write 100 ← 40:

Part 2:
p0: read 120
p0: write 120 ← 60

Part 3:
p0: read 100
p0: write 100 ← 60
p1: write 100 ← 40

Question B: (5 points)
Based on your answers to Question A, what are the total stall cycles for each part?

You should assume the following latencies:

Parameter	Latency (CCs)
Memory	100
Cache	70
Invalidate	15
Write back	10

Part 1:
p0: read 100
p0: write 100 ← 40

Part 2:
p0: read 120
p0: write 120 ← 60

Part 3:
p0: read 100
p0: write 100 ← 60
p1: write 100 ← 40