# EECS 570 *Final Exam*

Winter 2024

Name: _____ unique name: _____

Sign the honor code:

I have neither given nor received aid on this exam nor observed anyone else doing so.

_____

---

Scores:

| # | Points |
|---|---|
| 1 | **/ 20** |
| 2 | **/ 30** |
| 3 | **/ 30** |
| 4 | **/ 20** |
| ***Total*** | ***/ 100*** |

## NOTES:

- Closed book. One 8.5" x 11" page of notes is allowed.
- Calculators are allowed
- Don't spend too much time on any one problem.
- You have about 120 minutes for the exam (avg. 30 minutes per problem).
- There are 10 pages in the exam (including this one), Please ensure you have all pages.
- **Be sure to show work and explain what you've done when asked to do so.**

**1. Short Answers [20 points]**

a. A memory system is coherent if there is a total order on all _____ to any given address.

b. This isn't enough for consistency. Sequential Consistency (SC) assumes there is a total order among _____

c. Sun's "Relaxed Memory Order" (RMO) allows unordered, coalescing post-retirement store buffers, but the compiler cannot reorder loads between stores.

◯ True ◯ False

d. Sequential Consistency (SC) makes nearly all compiler optimizations illegal, but (in c/c++ programming model) operations within an expression may be reordered.

◯ True ◯ False

e. Order the following four network domains in terms of their latency requirements:

| | low latency ==========> High latency | | | |
|---|---|---|---|---|
| LANs(Local Area Network) | ◯ | ◯ | ◯ | ◯ |
| WAN(Wide Area Networks) | ◯ | ◯ | ◯ | ◯ |
| SANs(Storage Area Networks) | ◯ | ◯ | ◯ | ◯ |
| OCNs/NoCs(Network on Chip) | ◯ | ◯ | ◯ | ◯ |

**2. Murphi for MSI [30 points]**

Answer the questions on the following page about verifying the MSI protocol using Murphi. You should make the same assumptions as PA2, listed here:

1. It uses an interconnect network that supports only point-to-point communication. All communication is done by sending and receiving messages. **The interconnect network may reorder messages arbitrarily.** It may delay messages, but it will always deliver messages eventually. Messages are never lost, corrupted or replicated. Message delivery cannot be assumed to be in the same order as they were sent, even for the same sender and receiver pair.
2. At the receiving side of the interconnect system, messages are delivered to a receive port. Once a message has been delivered to the receive port, it will block all subsequent messages to this port until the message is read. Consider this behavior equivalent to that of a mailbox with room for only one letter: you have to remove the letter from the mailbox before you can receive the next one. On the sending side, there is no such restriction: you can always send messages. The interconnect system has enough buffer space to queue messages.
3. For the purpose of this assignment, you may assume that there is no limit on the buffer space in the interconnect system. However, your protocol will be considered broken if there is a way to generate an infinite number of undelivered messages. Besides, you will not be able to verify your protocol in this case.
4. You may assume that the interconnect network supports multiple lanes. For each lane, you have a separate set of send- and receive-ports for each unit. Traffic on one lane is independent of traffic in the other lanes. Messages will never switch lanes. Note that using fewer lanes is better.
5. Each processor has a dedicated cache that is **not shared with any other processor**. All caches must be kept coherent by your cache coherency protocol. Processors may issue load and store operations only. Because this assignment only deals with cache coherency and not with consistency issues, **you will be concerned with only one storage location** (address). However, you need to model cache conflicts. To do this, you need to model a third operation besides load and store: a cache write-back. Write-backs normally arise from a cache conflict if the old line is dirty. Write-back operations may occur at any time between any pair of load/store operations. If the cache is in a clean state, you may simply set it to be invalid or take the appropriate action according to your CC protocol. Cache replacements of dirty lines must obviously write the line back to memory.
6. You should assume that the coherency unit is equal to one word and that all loads and stores read or write the entire word.
7. Besides processors with their caches, there is one memory unit in your system. The memory unit has a directory-based cache-consistency controller which ensures that only one processor can write to the memory block at a time (exclusive-ownership style protocol). The directory representation is unimportant for this assignment. You should assume that you have a full directory (bit vector) that can keep track of all sharers.
8. The interconnect system can send messages from any unit to any other unit. It is OK if your protocol requires that a cache controller has to send a message to another cache controller.

a. Assume core 0 is in the S state and core 1 wants to transition from I state to S state. Only core 0 is on the sharer list. What messages are sent if this is a 4-hop system? (from whom to whom, message name, and what information is included)

b. What about a 3-hop system? (Note: PA2 implements a 3-hop system)

c. [Race conditions] Assume core 0 and core 1 both send a GetM message to the home node. What should the Home node do to avoid race conditions?

d. PA2 setup allows multiple virtual channels on the interconnect network. Is it okay if all the messages are sent on a single channel? Why and why not (use examples to explain)?

e. Processor-initiated rules. Specify all possible rules a processor can fire and what coherence messages are sent when it is in M, S, or I state.

   p.state = **Invalid**

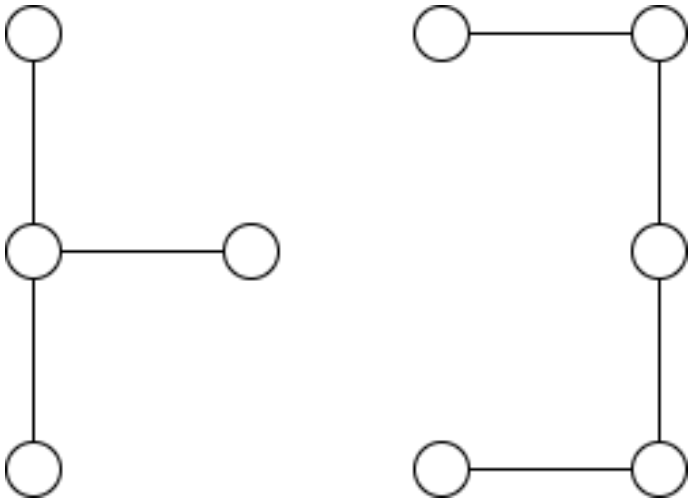
   (example) rule “write request at invalid state.” Send GetM to HomeNode.

   p.state = **Modified**

   p.state = **Shared**

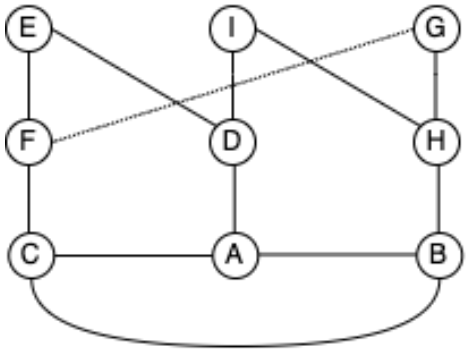
**3) Network-on-Chip [30 points]**

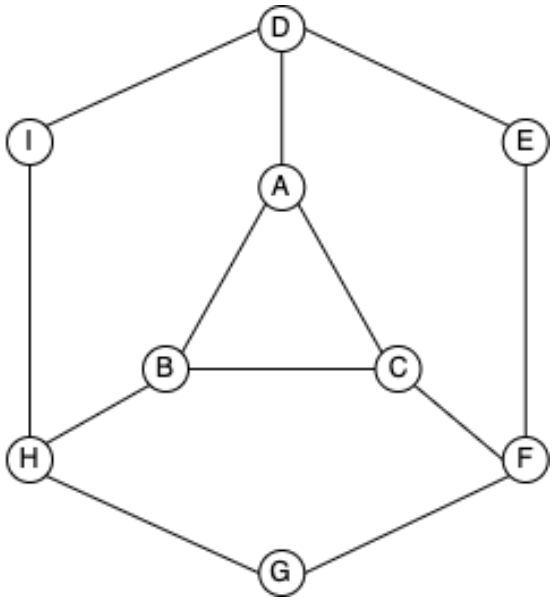
When designing topologies, there are several important properties to consider that can affect NoC performance. One of these considerations is *average hop count* which measures the average number of hops required to travel between any two processors on the chip. Consider the following *incomplete* NoC topology where each node represents a processor and each line represents a link.



a. As a NoC topology designer, what *single link* would you add between *any two processors* in this network to minimize the average hop distance in the NoC? Draw the link above and give a one-sentence explanation. (Note: the length of each link is not a consideration for this question.)

In class we have covered many different topologies. One of the most basic topologies is an NxN grid. Researchers at the University of Michigan are investigating alternatives to this using 3D topologies like the one below (these are two representations of the same NoC topology).




b. In one sentence each, name an advantage and a disadvantage of this topology compared to a 3x3 grid topology.

( + )

( – )

c. Below is the state of the link queues. Identify the cyclical resource dependency in the network that is causing a deadlock by **circling** the links involved - you can do this on either representation.

| Link | A→B | A→C | A→D | B→A | B→H | C→F | C→B | F→E | E→D | D→A |
|---|---|---|---|---|---|---|---|---|---|---|
| Packet Destination | H | F | D | G | I | E | B | D | A | C |

In one sentence explain why this is a cyclical resource dependency:

d. We now want you to design a deadlock-free routing algorithm for this topology. Assume there is only one message class and you can use at most two virtual channels (Hint: work on making routing deadlock-free within each 2D layer first, then think of how to connect them in a deadlock-free manner.)

(i) How is routing done within each 2D layer? Explain how VCs are used within a layer.

(ii) How is routing done between each 2D layer? Explain how VCs are used between layers.

e. Argue why your algorithm is deadlock-free in 3-4 sentences.

**4) Consistency [20 points]**

Given the following litmus test, for each given output value set, mark (by shading the box) whether it is a valid outcome or not for the different consistency models.

| Core 0 | Core 1 | Core 2 | Core 3 |
|---|---|---|---|
| (i1) a = 1<br>(i2) r1 = a<br>(i3) r2 = b | (i4) b = 1<br>(i5) r3 = b<br>(i6) r4 = a | (i7) r5 = a<br>(i8) r6 = b | (i9) r7 = b<br>(i10) r8 = a |

Example

| | |
|---|---|
| r1 | 1 |
| r2 | 1 |
| r3 | 1 |
| r4 | 1 |
| r5 | 1 |
| r6 | 1 |
| r7 | 1 |
| r8 | 1 |

| | Valid | Not Valid |
|---|---|---|
| SC | ■ | |
| TSO – MCA Style (IBM 370) | ■ | |
| TSO – rMCA Style (Sun) | ■ | |
| PC – nMCA Style | ■ | |
| PSO | ■ | |

a. Output of

| | |
|---|---|
| r1 | 1 |
| r2 | 1 |
| r3 | 1 |
| r4 | 1 |
| r5 | 1 |
| r6 | 0 |
| r7 | 1 |
| r8 | 0 |

| | Valid | Not Valid |
|---|---|---|
| SC | | |
| TSO – MCA Style (IBM 370) | | |
| TSO – rMCA Style (Sun) | | |
| PC – nMCA Style | | |
| PSO | | |

b. Output of

| | |
|---|---|
| r1 | 1 |
| r2 | 0 |
| r3 | 1 |
| r4 | 0 |
| r5 | 1 |
| r6 | 1 |
| r7 | 1 |
| r8 | 1 |

| | Valid | Not Valid |
|---|---|---|
| SC | | |
| TSO – MCA Style (IBM 370) | | |
| TSO – rMCA Style (Sun) | | |
| PC – nMCA Style | | |
| PSO | | |

c. Output of

| | |
|---|---|
| r1 | 1 |
| r2 | 1 |
| r3 | 0 |
| r4 | 1 |
| r5 | 1 |
| r6 | 1 |
| r7 | 1 |
| r8 | 1 |

| | Valid | Not Valid |
|---|---|---|
| SC | | |
| TSO – MCA Style (IBM 370) | | |
| TSO – rMCA Style (Sun) | | |
| PC – nMCA Style | | |
| PSO | | |

d. Output of

| | |
|---|---|
| r1 | 1 |
| r2 | 1 |
| r3 | 1 |
| r4 | 1 |
| r5 | 0 |
| r6 | 0 |
| r7 | 1 |
| r8 | 1 |

| | Valid | Not Valid |
|---|---|---|
| SC | | |
| TSO – MCA Style (IBM 370) | | |
| TSO – rMCA Style (Sun) | | |
| PC – nMCA Style | | |
| PSO | | |

Intentionally Blank Page