

ETH 263-2210-00L COMPUTER ARCHITECTURE, FALL 2022
HW 5: INTERCONNECTS, NOC, ACMP, AND BOTTLENECK ACCELERATION

Instructor: Prof. Onur Mutlu

TAs: Juan Gómez Luna, Mohammad Sadrosadati, Mohammed Alser, Rahul Bera, Nisa Bostanci,
João Dinis Ferreira, Can Firtina, Nika Mansouri Ghiasi, Geraldo Francisco De Oliveira Junior,
Konstantinos Kanellopoulos, Joël Lindegger, Rakesh Nadig, Ataberk Olgun, Abdullah Giray Yaglikci,
Yahya Can Tugrul, Haocong Luo, Banu Cavlak, Aditya Manglik

Given: Monday, December 8, 2022
Due: **Friday, December 19, 2022**

- **Handin - Critical Paper Reviews (1).** You need to submit your reviews to <https://safari.ethz.ch/review/architecture22/>. Please, check your inbox, you should have received an email with the password you should use to login. If you did not receive any email, contact comparch@lists.inf.ethz.ch. In the first page after login, you should click in "Computer Architecture Home", and then go to "any submitted paper" to see the list of papers.
- **Handin - Questions (2-6).** You should upload your answers to the Moodle Platform (<https://moodle-app2.let.ethz.ch/mod/assign/view.php?id=833634>) as a single PDF file.
- If you have any questions regarding this homework, please ask them the Moodle forum (<https://moodle-app2.let.ethz.ch/mod/moodleoverflow/view.php?id=833635>).
- Please note that the handin questions have a hard deadline. However, you can submit your paper reviews till January 31 2023.

1. Critical Paper Reviews [1,000 points]

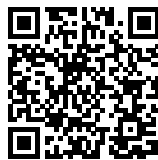
We assign you five **required readings** for this homework. You may access them by *simply clicking on the QR codes below or scanning them*.



Required 1



Required 2



Required 3



Required 4



Required 5

Write an approximately one-page critical review for the readings (i.e., papers from #1 to #5). If you review a paper other than the 5 mandatory papers, you will receive 200 BONUS points on top of 1,000 points you may get from paper reviews (i.e., each additional submission is worth 200 BONUS points with a possibility to get up to 5800 points). Note that you will get **zero** points from the critical paper reviews if you do not submit the required paper reviews (i.e., papers from #1 to #5).

Please read the guideline slides for reviewing papers and watch Prof. Mutlu's guideline video on how to do a critical paper review. We also provide you with sample reviews which you can access using the QR code. A review with bullet point style is more appreciated. Try not to use very long sentences and paragraphs. Keep your writing and sentences simple. Make your points bullet by bullet, as much as possible. **We will give out extra credit that is worth 0.5% of your total course grade for each good review.**



Guideline Slides



Guideline Video



Sample Reviews

1. **(REQUIRED)** Moscibroda et al., “A Case for Bufferless Routing in On-Chip Networks“, ISCA 2009, https://people.inf.ethz.ch/omutlu/pub/bless_isca09.pdf
2. **(REQUIRED)** Suleman et al., “Accelerating critical section execution with asymmetric multi-core architectures“, ASPLOS’09, https://people.inf.ethz.ch/omutlu/pub/acs_asplos09.pdf
3. **(REQUIRED)** Das et al., “Aergia: Exploiting packet latency slack in on-chip networks“, ISCA, 2010, <https://www.microsoft.com/en-us/research/wp-content/uploads/2016/02/ISCA2010.pdf>
4. **(REQUIRED)** Fallin et al., “MinBD: Minimally-Buffered Deflection Routing for Energy-Efficient Interconnect“, NOCS 2012, https://people.inf.ethz.ch/omutlu/pub/minimally-buffered-deflection-router_nocs12.pdf
5. **(REQUIRED)** Joao et al., “Bottleneck Identification and Scheduling in Multithreaded Applications“, ASPLOS’12, https://people.inf.ethz.ch/omutlu/pub/bottleneck-identification-and-scheduling_asplos12.pdf
6. Patel et al., “Performance of Processor-Memory Interconnections for Multiprocessors“, ISCA 1979, <https://safari.ethz.ch/architecture/fall2020/lib/exe/fetch.php?media=p168-patel.pdf>
7. Gottlieb et al., “The NYU Ultracomputer—Designing an MIMD Shared Memory Parallel Computer“, Transactions on Computers 1983, <http://aggregate.org/GPUMC/01676201.pdf>
8. Seitz et al., “The cosmic cube“, CACM 1985, <https://safari.ethz.ch/architecture/fall2020/lib/exe/fetch.php?media=p22-seitz.pdf>
9. Glass and Ni, “The Turn Model for Adaptive Routing“, ISCA 1992 <https://safari.ethz.ch/architecture/fall2019/lib/exe/fetch.php?media=p278-glass.pdf>
10. Hillis et al., “The CM-5 Connection Machine: a scalable supercomputer“, Communications of the ACM 1993, https://course.ece.cmu.edu/~ece740/f13/lib/exe/fetch.php?media=hillis_cm5.pdf
11. Herlihy et al., “Transactional memory: architectural support for lock-free data structures“, ISCA 1993 <https://cs.brown.edu/~mph/HerlihyM93/herlihy93transactional.pdf>
12. Rajwar et al., “Speculative Lock Elision: Enabling Highly Concurrent Multithreaded Execution“, MICRO 2001 https://safari.ethz.ch/architecture_seminar/fall2018/lib/exe/fetch.php?media=p294-rajwar.pdf
13. Dally et al., “Route packets, not wires: on-chip interconnection networks“, DAC 2001, http://cva.stanford.edu/publications/2001/onchip_dac01.pdf
14. Kumar et al., “Single-ISA heterogeneous multi-core architectures: The potential for processor power reduction“, MICRO’03, <https://safari.ethz.ch/architecture/fall2020/lib/exe/fetch.php?media=01253185.pdf>
15. Grochowski et al., “Best of both latency and throughput“, ICCD’04, <https://safari.ethz.ch/architecture/fall2020/lib/exe/fetch.php?media=01347928.pdf>
16. Annavaram et al., “Mitigating Amdahl’s law through EPI throttling“, ISCA’05, <https://safari.ethz.ch/architecture/fall2020/lib/exe/fetch.php?media=01431565.pdf>
17. Das et al., “Application-aware prioritization mechanisms for on-chip networks“, MICRO 2009, https://people.inf.ethz.ch/omutlu/pub/app-aware-noc_micro09.pdf
18. Grot et al., “Express Cube Topologies for On-Chip Interconnects“, HPCA 2009, https://people.inf.ethz.ch/omutlu/pub/mecs_hPCA09.pdf
19. Grot et al., “Preemptive Virtual Clock: A Flexible, Efficient, and Cost-effective QoS Scheme for Networks-on-Chip“, MICRO 2009, https://people.inf.ethz.ch/omutlu/pub/pvc-qos_micro09.pdf
20. Suleman et al., “Data Marshaling for Multi-core Architectures“, ISCA’10, https://people.inf.ethz.ch/omutlu/pub/dm_isca10.pdf
21. Grot et al., “Kilo-NOC: A Heterogeneous Network-on-Chip Architecture for Scalability and Service Guarantees“, ISCA 2011, <https://safari.ethz.ch/architecture/fall2018/lib/exe/fetch.php?media=p401-grot.pdf>
22. Fallin et al., “CHIPPER: A Low-Complexity Bufferless Deflection Router“, HPCA, 2011, https://people.inf.ethz.ch/omutlu/pub/chipper_hPCA11.pdf
23. Chang et al., “HAT: Heterogeneous Adaptive Throttling for On-Chip Networks“, SBAC-PAD 2012. http://users.ece.cmu.edu/~omutlu/pub/hetero-adaptive-source-throttling_sbacpad12.pdf
24. Nychis et al., “On-Chip Networks from a Networking Perspective: Congestion and Scalability in Many-core Interconnects“, SIGCOMM 2012. https://people.inf.ethz.ch/omutlu/pub/onchip-network-congestion-scalability_sigcomm2012.pdf
25. Joao et al., “Utility-Based Acceleration of Multithreaded Applications on Asymmetric CMPs“, ISCA’13, https://people.inf.ethz.ch/omutlu/pub/utility-based-acceleration-acmp_isca13.pdf
26. Mishra et al., “A Heterogeneous Multiple Network-on-Chip Design: An Application-Aware Approach“, DAC 2013. https://people.inf.ethz.ch/omutlu/pub/hetero-multiple-NoC_dac13.pdf
27. Ausavarungnirun et al., “Design and Evaluation of Hierarchical Rings with Deflection Routing“, SBAC-PAD 2014, https://people.inf.ethz.ch/omutlu/pub/hierarchical-rings-with-deflection_sbacpad14.pdf

28. Fattah et al., “A Low-Overhead, Fully-Distributed, Guaranteed-Delivery Routing Algorithm for Faulty Network-on-Chips,” NOCS 2015 https://people.inf.ethz.ch/omutlu/pub/maze-routing_nocs15.pdf
29. Ausavarungnirun et al., “Energy-Efficient Deflection-based On-chip Networks: Topology, Routing, Flow Control,” arXiv 2021. <https://arxiv.org/pdf/2112.02516.pdf>

2. Interconnection Networks - I [200 points]

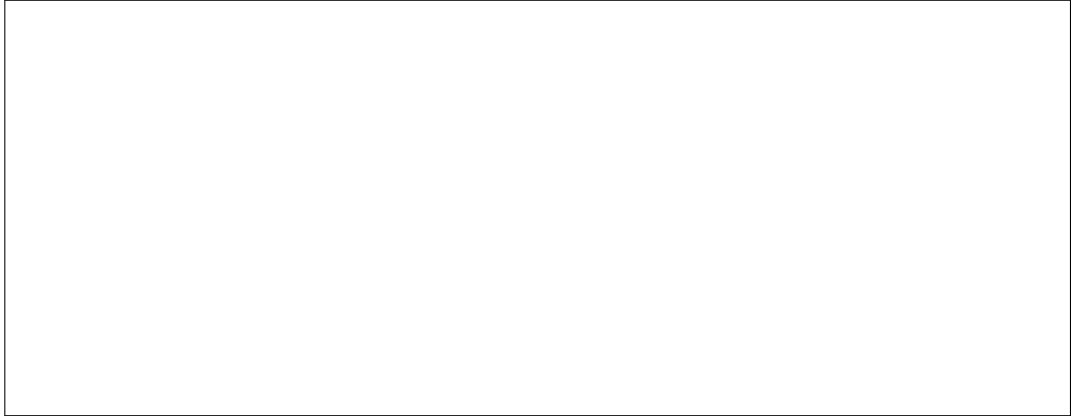
Suppose you would like to connect 2^N processors, and you are considering four different topologies:

- $\sqrt{2^N} \times \sqrt{2^N}$ 2D mesh
- $\sqrt{2^{N-2}} \times \sqrt{2^{N-2}}$ 2D concentrated mesh (Cmesh), where each router serves four processors
- $\sqrt{2^N} \times \sqrt{2^N}$ 2D torus
- Hypercube

Please answer the following questions. Show your work.

- (a) For $N = 4$, please draw how each network looks like. You can use ... (three dots) to avoid repeated patterns.

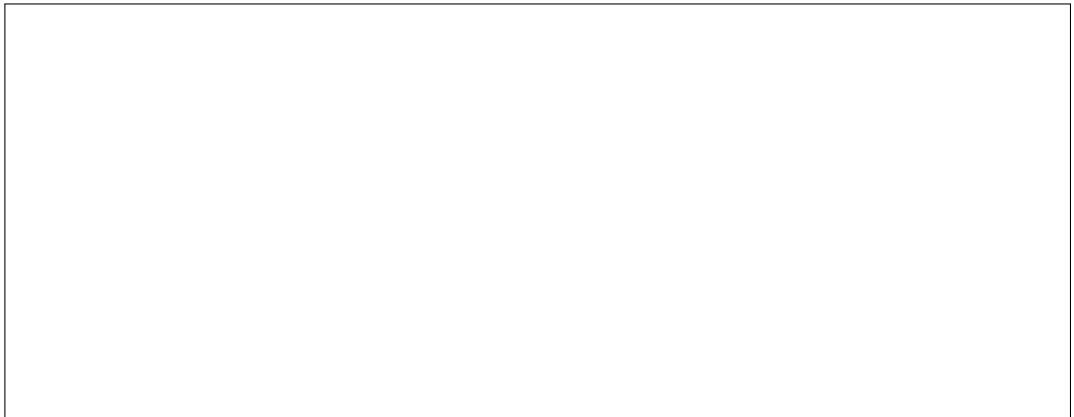
2D mesh



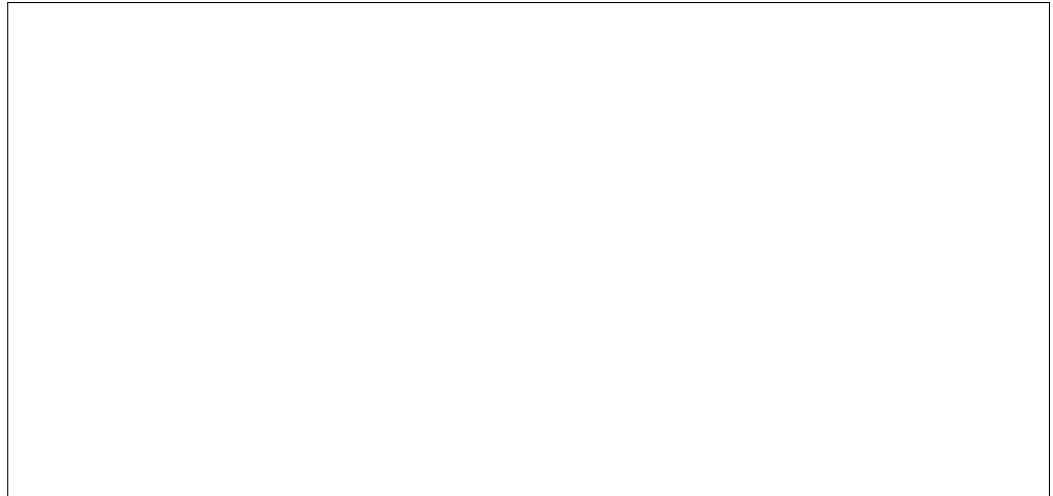
Cmesh



2D torus



Hypercube



For the remaining questions, *assume* $N = 8$.

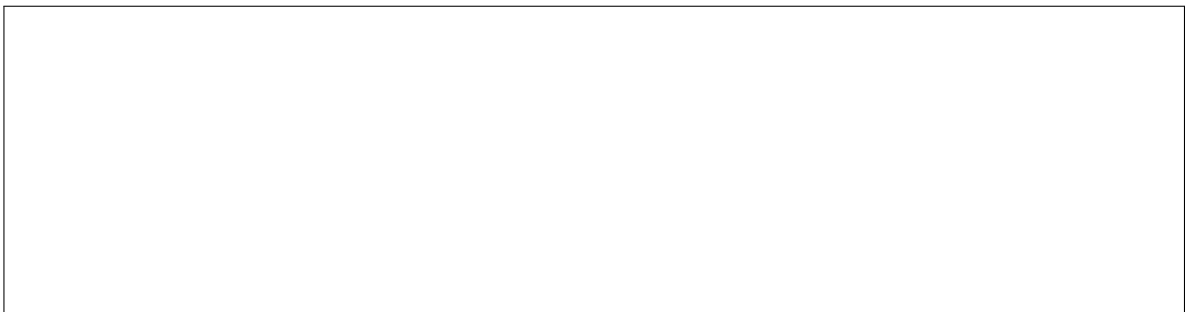
- (b) For $N = 8$, calculate the number of network links for each network. (Hint: a single network link is bi-directional)



- (c) For $N = 8$, calculate the number of input/output ports including the injection/ejection ports for *each router* in these topologies (Hint: give answer to all types of routers that exist in an irregular network).



- (d) Assume a network link can be faulty. For each topology, what is the minimum possible number of faulty links that are needed to make at least one processor unreachable from any other processor?



3. Interconnection Networks - II [200 points]

Suppose you want to connect 16 processors using two topologies: 4×4 Mesh and 4×4 Torus with bi-directional links, similar to Figure 1:

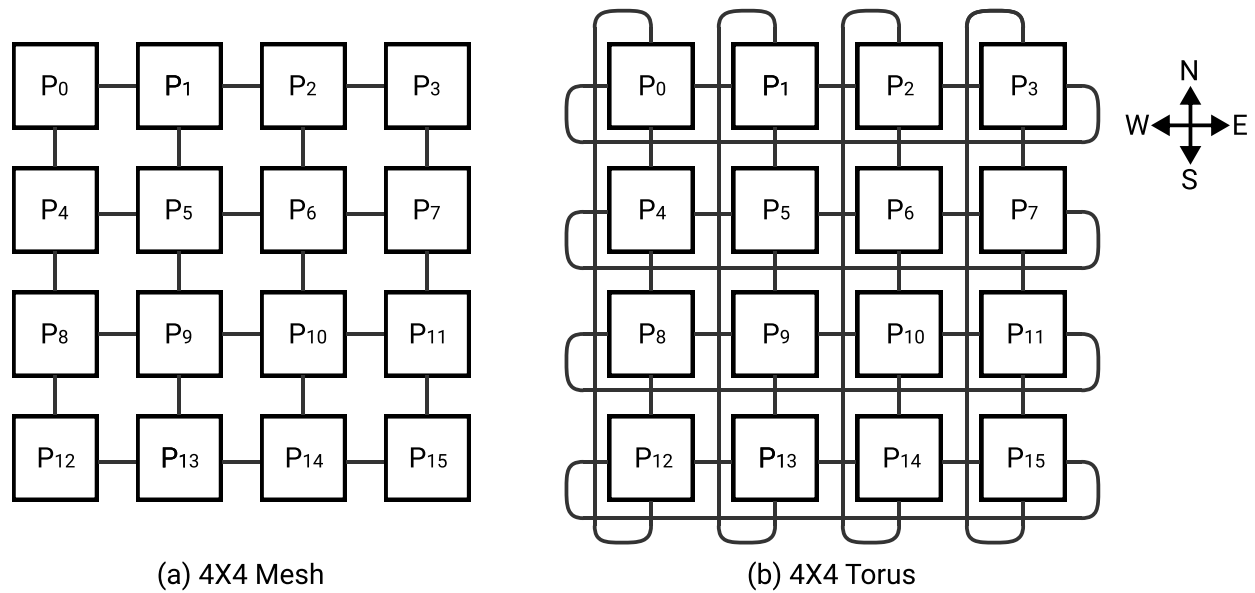


Figure 1. 2D mesh and 2D torus topologies connecting 16 processors.

You know that there are eight possible turns in the 2D mesh and the 2D torus topologies, as shown in Figure 2:

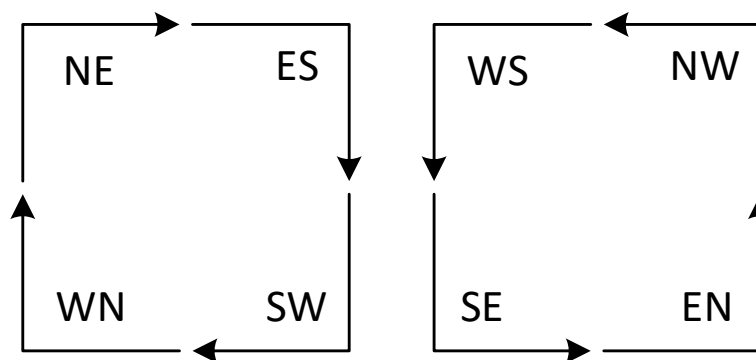


Figure 2. Eight possible turns in 2D mesh and 2D torus topologies.

You are considering three different *minimal* routing algorithms based on which turns are forbidden:

- **Routing Algorithm 1:** Turns NE, SW, NW, and SE are forbidden. The other four turns are allowed.
- **Routing Algorithm 2:** Turns SW and NW are forbidden. The other six turns are allowed.
- **Routing Algorithm 3:** Turns WN and NW are forbidden. The other six turns are allowed.

For the rest of this question, we address source-destination pairs using $\langle \text{src}, \text{dest} \rangle$ representation. Please answer the following questions and show your work.

- (a) Show all possible paths between source and destination pairs $\langle P_5, P_{11} \rangle$ and $\langle P_5, P_{12} \rangle$, using each of the previously mentioned routing algorithms. For each path, mention the source router, intermediate routers, and destination router.

2D Mesh

Algorithm 1, $\langle P_5, P_{11} \rangle$

Algorithm 1, $\langle P_5, P_{12} \rangle$

Algorithm 2, $\langle P_5, P_{11} \rangle$

Algorithm 2, $\langle P_5, P_{12} \rangle$

Algorithm 3, $\langle P_5, P_{11} \rangle$

Algorithm 3, $\langle P_5, P_{12} \rangle$

2D Torus

Algorithm 1, $\langle P_5, P_{11} \rangle$

Algorithm 1, $\langle P_5, P_{12} \rangle$

Algorithm 2, $\langle P_5, P_{11} \rangle$

Algorithm 2, $\langle P_5, P_{12} \rangle$

Algorithm 3, $\langle P_5, P_{11} \rangle$

Algorithm 3, $\langle P_5, P_{12} \rangle$

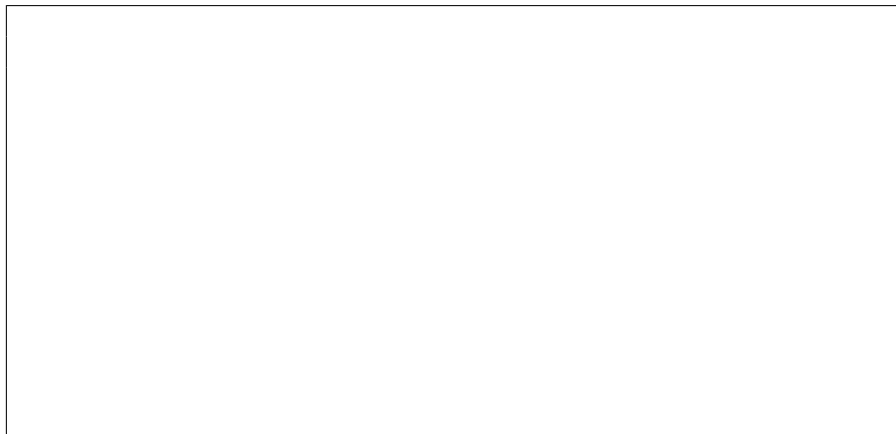
- (b) Which of these three algorithms are deadlock free? For each case, if deadlocks can happen, provide a deadlock scenario. Otherwise, prove that deadlock occurrence is impossible (Hint: Use contradiction).

2D Mesh

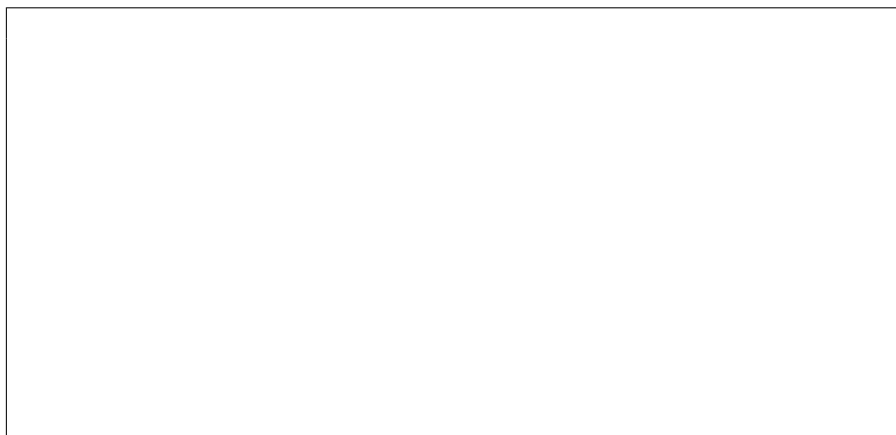
Routing Algorithm 1:

A large, empty rectangular box with a thin black border, intended for the student to provide a deadlock scenario or proof for Routing Algorithm 1.

Routing Algorithm 2:

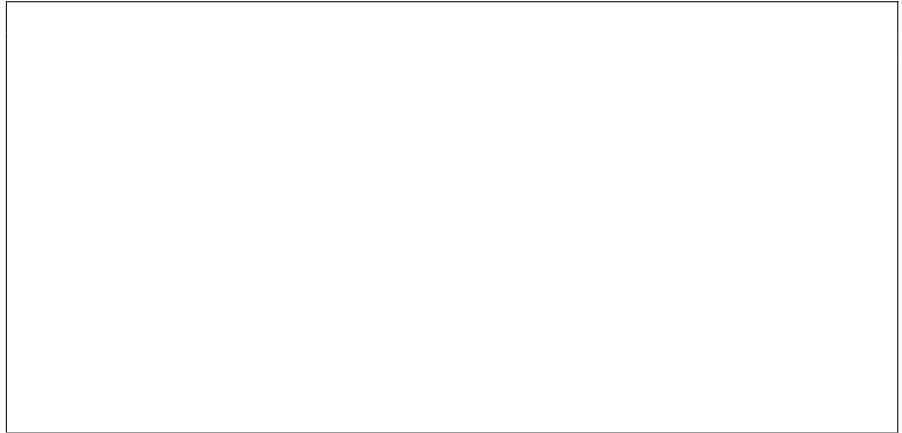
A large, empty rectangular box with a thin black border, intended for the student to provide a deadlock scenario or proof for Routing Algorithm 2.

Routing Algorithm 3:

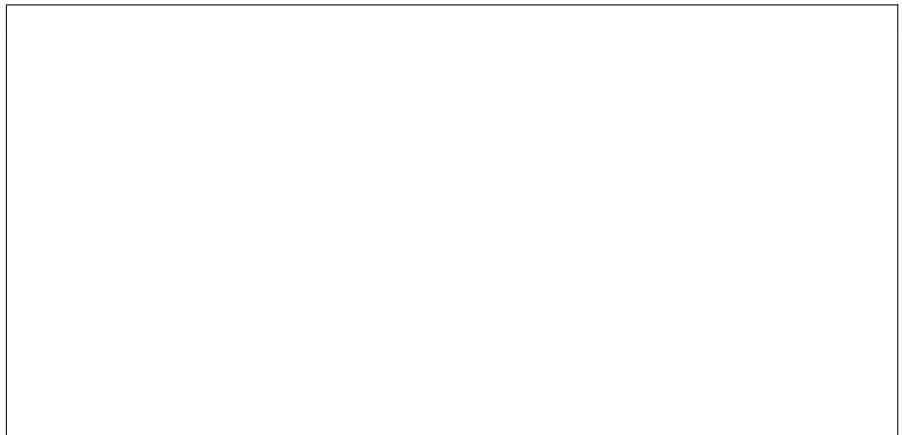
A large, empty rectangular box with a thin black border, intended for the student to provide a deadlock scenario or proof for Routing Algorithm 3.

2D Torus

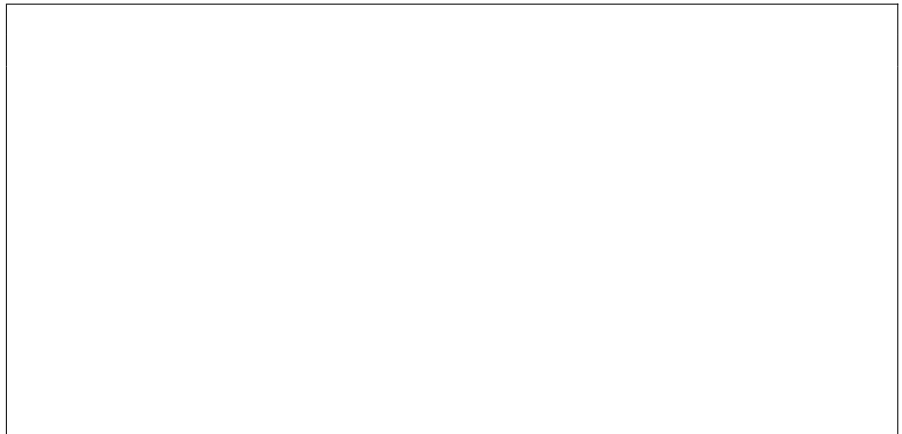
Routing Algorithm 1:



Routing Algorithm 2:



Routing Algorithm 3:



You are about to design a routing algorithm for each of these two topologies with the following two requirements: Your routing algorithm (1) should be deadlock free, and (2) should *not* be deterministic.

- (c) Which routing algorithm out of the three earlier routing algorithms do you choose for each of these two topologies? Explain your reasoning.

2D mesh

2D torus

4. Asymmetric Multicore - I [200 points]

A microprocessor manufacturer asks you to design a multicore processor for modern workloads. You should optimize it assuming a workload with 60% of its work in the parallel portion and 40% in the serial portion. You are tasked to compare two configurations that can fit into the processor's die area: 1) Small Cores (SC), and 2) Large + Small Cores (LSC). These consist of the following:

- *SC*: A design that contains 8 small cores, which share the same die. Seven of these small cores operate at a *baseline* fixed instruction throughput. The eighth small core is an overclockable core, which can operate at either: 1) *baseline* throughput, or 2) *overclocked* with 2× the baseline throughput.
- *LSC*: A design that contains 1 large core and 4 small cores that all share the same die. The four small cores operate at the baseline throughput. The large core is 4× faster than a small core.

In addition, Table 1 provides the static power (i.e., when the core is idle) and the dynamic power (i.e., when the core is active) of each of the cores.

Table 1. Power consumption of cores in different modes.

Core	Mode	Static Power (W)	Dynamic Power (W)
Small	Baseline	0.5	1
	Overclocked	1	8
Large	Baseline	2	4

The SC processor executes the parallel portion on *all* the small cores (including the overclockable core, operating at the baseline throughput), and the serial portion *only* on the overclockable core (using either baseline or overclocked options). The LSC processor executes the parallel portion only on the small cores, and the serial portion only on the large core.

Please answer the following questions.

- (a) Which of the three design configurations (SC, SC with overclocked core, or LSC) results in the highest performance? Show your work.

- (b) The energy consumption should also be a metric of reference in your design. Which of the three design configurations (SC, SC with overlocked core, or LSC) results in the lowest energy consumption? Show your work.

- (c) At least what ratio of a workload should be spent on the parallel section so that the *SC* configuration, *even without overlocking*, performs better than the *LSC* configuration? Show your work.

- (d) In order to improve the performance of the LSC configuration, you come up with hardware design optimizations to improve the throughput of the large core. You expect that these optimizations will increase the throughput of the large core by $T\times$. Given an application with 90% of its work in the parallel portion, is it possible for the LSC configuration to outperform the SC configuration *with* overclocking? If yes, for which values of T ? Show your work.

5. Asymmetric Multicore - II [200 points]

A microprocessor manufacturer asks you to design an asymmetric multicore processor for modern workloads. Your design contains one large core and several small cores, which share the same die. Assume the total die area is A units. The table below describes the area, performance, and power specifications for each core type.

Type of Core	Area (mm^2)	Performance	Dynamic Power (W)	Static Power (W)
Large	S	\sqrt{S}	S	$\frac{1}{4} \times S$
Small	1	1	1	0.5

The serial portion of a workload executes only on the large core, while the parallel portion executes on both large and small cores. On this multiprocessor, we will execute a workload where a fraction P of its work is parallel, and $1 - P$ of its work is serial. You will fit as many small cores as possible, after placing the large core. Consider the following two configurations:

- Configuration X: $A = 32$, $S = 4$.
- Configuration Y: $A = 32$, $S = 16$.

Please answer the following questions. Show your work. Express your equations and solve them.

- (a) For what values of P does the workload run faster on Y than on X? Show your work.

- (b) For what values of P does the workload consumes less energy when running on Y than on X? Show your work.



5.1. Accelerating Single-Thread Execution

Assume that two large cores can operate in a collaborative manner to achieve the single-thread performance of an even “larger” core that is $\mathbf{N} \times$ faster than the largest core on the chip. When executing the serial portion of the workload, the functional units of both large cores are merged into the same pipeline to have a faster core. The collaborative execution mode is only enabled during the serial portion of the workload. During the parallel portion, the two large cores separate from each other (on-the-fly) and operate as two independent cores. The serial portion executes only on the “dual-core”, and the parallel portion executes on all the cores. The table below describes the area and performance specifications for each core for this design.

Type of Core	Area (mm^2)	Performance
$Large_1$	S_1	$\sqrt{S_1}$
$Large_2$	S_2	$\sqrt{S_2}$
$Large_1 + Large_2$	$S_1 + S_2$	$\mathbf{N} \times \sqrt{Max(S_1, S_2)}$
Small	1	1

Consider the following configuration:

- Configuration Z: $A = 32$, $S_1 = 9$, $S_2 = 4$.

(c) For what values of N does the workload run faster on Z than on X? Assume $P = 0.8$. Show your work.

(d) For what values of P does the workload run faster on Z than on X? Assume $N = 1.5$. Show your work.

(e) Suppose you are executing workloads where a fraction P of its work is *infinitely* parallelizable. Which configuration would you choose? Z or Y? Why?

6. Bottleneck Acceleration [200 points]

In this question, you are asked to analyze the performance and scalability benefits of accelerating the critical section in the following piece of code.

```
while (!problem_solved){
    Lock(X) //start of the critical section
        SubProblem = PQ.Dequeue();
    Unlock(X) //end of the critical section

    problem_solved = Solve(SubProblem)
    if(problem_solved)
        break;
}
```

Assume that

- the while loop iterates 15 times.
 - there is no data dependency across iterations.
- (a) We observe that the application's performance saturates when the iterations are distributed across five threads (i.e., running more than five threads does *not* improve the performance). We are interested in the ratio of time spent executing critical sections to the total execution time for each iteration. Calculate the range of all possible values for this ratio. (Hint: The performance saturates with P threads if it can achieve speedup from increasing the number of threads from $P-1$ to P , and not from P to $P+1$.)

- (b) In order to improve the performance and scalability of the program, we decide to accelerate the critical section by migrating its execution to a more powerful core. The execution of the critical section on the powerful core completes faster, including the cost of migration between cores. We use the powerful core only for executing the critical section, while the rest of the iteration executes on smaller less powerful cores (original cores used in part a). Then, we observe that the application's scalability increases up to six cores.

Assume that the critical section initially takes the minimum ratio of execution time that you found in the previous section. How much does the powerful core increase the performance of the critical section in this case? Show the range for possible performance improvements.