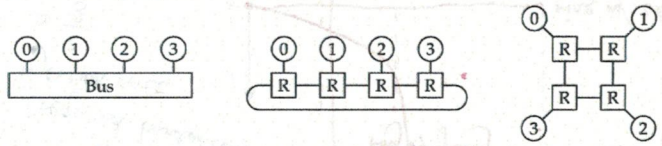
ECE 4750 Computer Architecture, Fall 2016
Problem-Based Learning on Fundamental Networks

http://www.csl.cornell.edu/courses/ece4750
School of Electrical and Computer Engineering
Cornell University

revision: 2016-10-21-10-54

In this section, we will analyze three different network topologies: a four-terminal multiple-writer, multiple-reader (MWMR) bus, a four-node ring (i.e., a 4-ary 1-cube torus) with both a greedy and an odd/even routing algorithm, and a 2x2 mesh (i.e., a 2-ary 2-cube mesh) with an XY dimension-ordered routing (DOR) algorithm. We will assume the same channel bandwidth in all topologies. The three network topologies are shown below.

assume channel bandwidth is 32 bit/cycle



4-Terminal Bus Topology 4-Node Ring Topology 2x2 Mesh Topology

We will be using the following traffic pattern in all examples. Informally for the ring topology, this traffic pattern involves each input terminal sending all of its traffic to the output terminal on the other side of the ring. Note that this traffic pattern is not quite the same as the tornado traffic pattern mentioned in lecture, since in tornado traffic pattern, all traffic from input terminal i is sent to output terminal $i + (\lceil k/2 \rceil - 1) \bmod k$ where k is is the number of nodes in a given dimension. The tornado traffic pattern on a ring with four terminals degenerates into the neighbor traffic pattern.

	0	1	2	3
0			1	
1				1
2	1			
3		1		

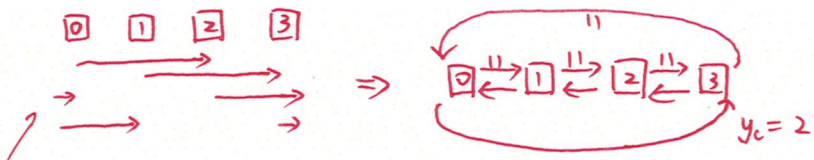
1. Four-Terminal MWMR Bus Topology

Buses are simple and commonly used when interconnecting small numbers of terminals. A bus connects a number of terminals with a single, shared channel that serves as a broadcast medium. At any given time, only one input terminal can transmit a packet over the bus. Assume that the bus bandwidth is the same as the input terminal channel bandwidth. **Calculate the ideal terminal throughput and the zero-load latency in router hops of the bus topology under the given traffic pattern. Sketch a bandwidth vs. latency curve for this topology on the graph at the end of this problem.**

$\Theta_{term} = \frac{32}{4} = 8$ bit/cycle (assume the bus bandwidth is 32 bit/cycle)

$T_\phi = 1$

If a packet is 32 bit, this means on average, each input terminal can send one packet every four cycles.



2. Four-Node Ring Topology

Multi-stage topologies can help improve scalability and throughput compared to simpler bus topologies. In this problem, we will analyze the four-node ring topology assuming two different routing algorithms, and we will also discuss conditions under which deadlock might occur.

2.1. Greedy Routing Algorithm

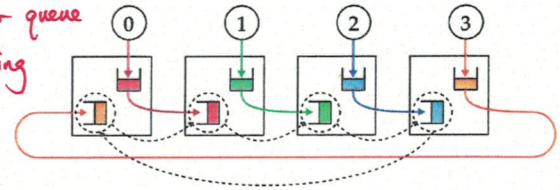
A *greedy routing algorithm* on a ring topology always chooses the direction that will result in a minimal path. Greedy algorithms can vary in how they handle the case when there are two minimal paths (i.e., the destination terminal is equidistant in both directions around the ring from the source terminal). For this problem, assume that when there are two minimal paths our greedy routing algorithm always chooses the clockwise direction. This means our greedy routing algorithm is an oblivious deterministic routing algorithm. **Calculate the ideal terminal throughput and the zero-load latency in router hops of the ring topology under the given traffic pattern with the greedy routing algorithm. Sketch a bandwidth vs. latency curve for this topology on the graph at the end of this problem.** *Hint: Use the tick-mark method to calculate the maximum channel load.*

$\Theta_{term} = \frac{32}{2} = 16$ bit/cycle $T_\phi = 3$

2.2. Deadlock

Deadlock occurs when packets are unable to make forward progress because of a cycle in the "holds" and "waiting for" dependencies. The following diagram illustrates how the greedy routing algorithm can result in deadlock for the given traffic pattern. The dashed circles represent which buffer a packet "holds", and the dashed arrow indicates which buffer a packet is "waiting for". After understanding the diagram, **brainstorm ways to avoid deadlock in this topology.**

① add another queue
② change routing algorithms
③ bubble flow control

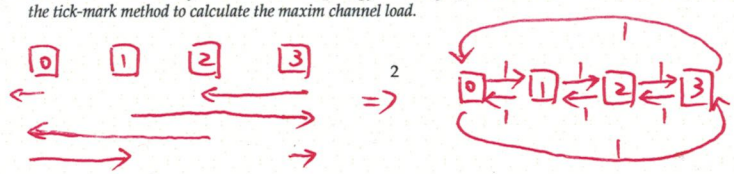


④ change traffic pattern to

but not realistic..

2.3. Odd/Even Routing Algorithm

For the given traffic pattern, the greedy routing algorithm heavily loads all clockwise channels but the corresponding counter-clockwise channels are idle. We can potentially improve the ideal throughput by using a different routing algorithm that better balances the load across both clockwise and counter-clockwise channels. Consider an *odd/even routing algorithm* in which odd input terminals use the clockwise channels when the destination is equidistant from the source, but even input terminals use the counter-clockwise channels when the destination is equidistant from the source. Note that the odd/even routing algorithm still always uses minimal paths and only differs from the greedy routing algorithm in the routes it chooses when the destination is equidistant from the source. **Calculate the ideal terminal throughput and the zero-load latency in router hops of the ring topology under the given traffic pattern with the odd/even routing algorithm. Sketch a bandwidth vs. latency curve for this topology on the graph at the end of this problem.** *Hint: Use the tick-mark method to calculate the maxim channel load.*



$y_c = 1$

$\Theta_{term} = \frac{32}{1}$

$= 32$ bit/cy

$T_\phi = 3$

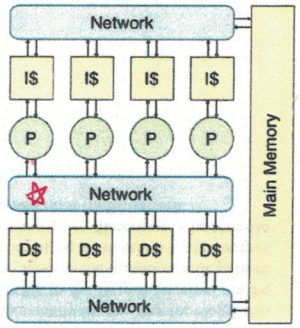
3. 2x2 Mesh Topology

In this problem, we will analyze the 2x2 mesh topology with XY dimension-ordered routing. In XY routing, we always route completely in the X direction before routing in the Y dimension. **Calculate the ideal terminal throughput and the zero-load latency in router hops of the mesh topology under the given traffic pattern with XY dimension-ordered routing. Sketch a bandwidth vs. latency curve for this topology on the graph at the end of this problem.**

Same as a 4-node ring with odd/even routing algorithm

4. Comparative Analysis

In lab 4, you will build the four-terminal bus and the multi-stage topology discussed in this problem. In lab 5, you will use this network to interconnect the pipelined processors you implemented in lab 2 and the blocking caches you implemented in lab 3. The figure to the right illustrates the multicore system you will be composing in lab 5.

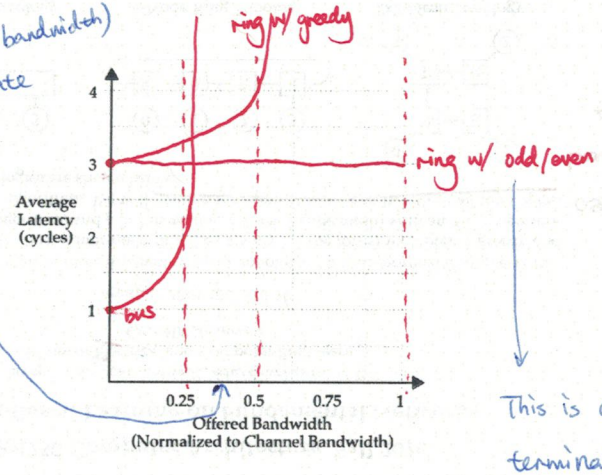


Consider the network between the processors and the L1 data caches and revisit the bandwidth vs. latency curve for the various topologies studied in this problem. Which topology (and routing algorithm) would be the best choice for the multicore processor? *Hint: Revist the vector-vector add assembly sequence and calculate the average number of loads and stores (i.e., read/write memory accesses) per cycle to get an estimate of the possible injection rate for the network.*

We don't know! Experiment in Lab 5!!

vvadd:
lw
lw
add
sw
addi
addi
addi
bne

(required bandwidth) injection rate $\approx \frac{3}{8}$



① Injection rate is approximate
(didn't include initialization & ending instructions, and # iterations is not infinite)

② The curves are theoretical
(didn't include channel latency, queuing delays, etc)

This is constant because channels are not shared and packets from the same input terminal are pipelined (You'll see this in lab 4). But this is because of the specific traffic pattern.