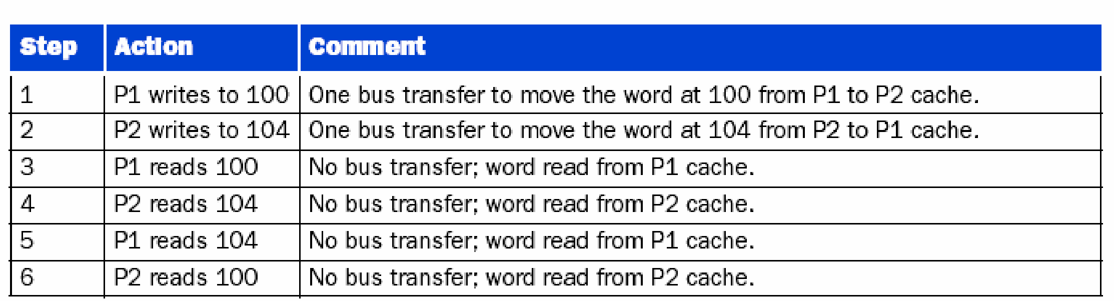
CEG 4131 Assignment 3 - Solutions

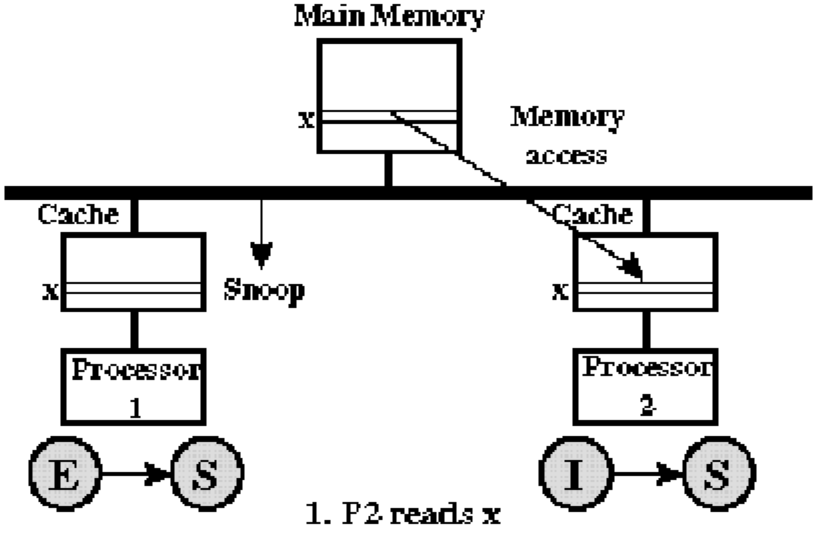
Problem 1

Write-back cache with write-update cache coherency and one-word blocks. Both words are in both caches and are initially clean. Assume 4-byte words and byte addressing.
Total bus transactions = 2

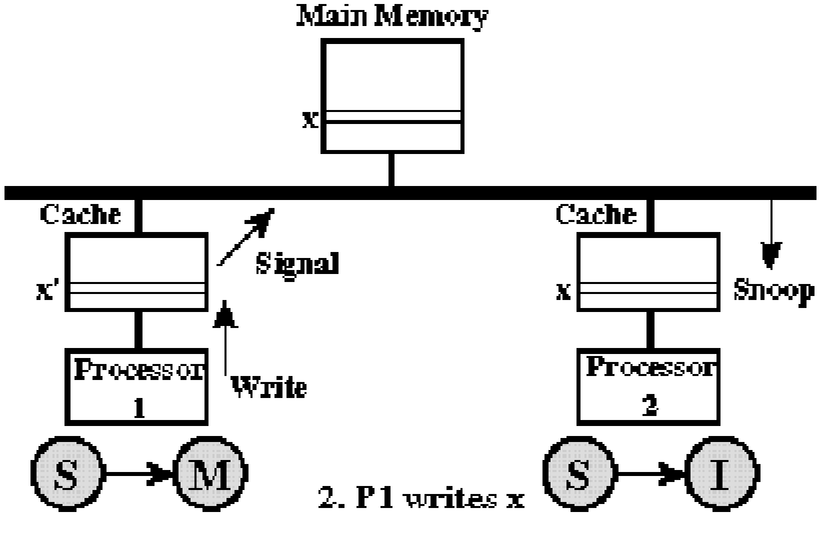


Step	Action	Comment
1	P1 writes to 100	One bus transfer to move the word at 100 from P1 to P2 cache.
2	P2 writes to 104	One bus transfer to move the word at 104 from P2 to P1 cache.
3	P1 reads 100	No bus transfer; word read from P1 cache.
4	P2 reads 104	No bus transfer; word read from P2 cache.
5	P1 reads 104	No bus transfer; word read from P1 cache.
6	P2 reads 100	No bus transfer; word read from P2 cache.

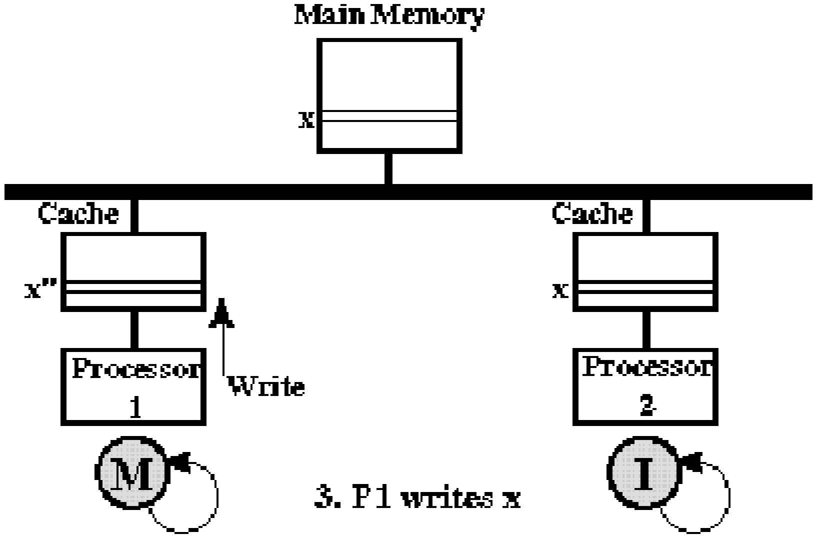
Problem 2



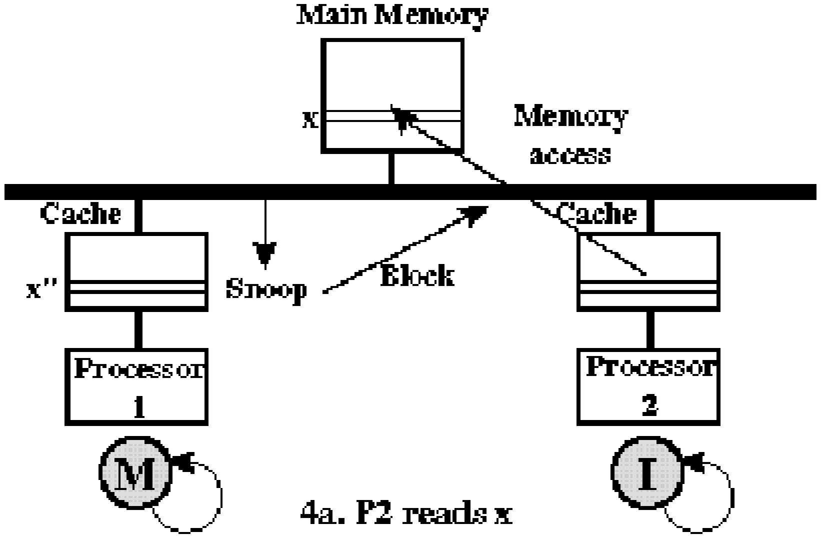
1. P2 reads x



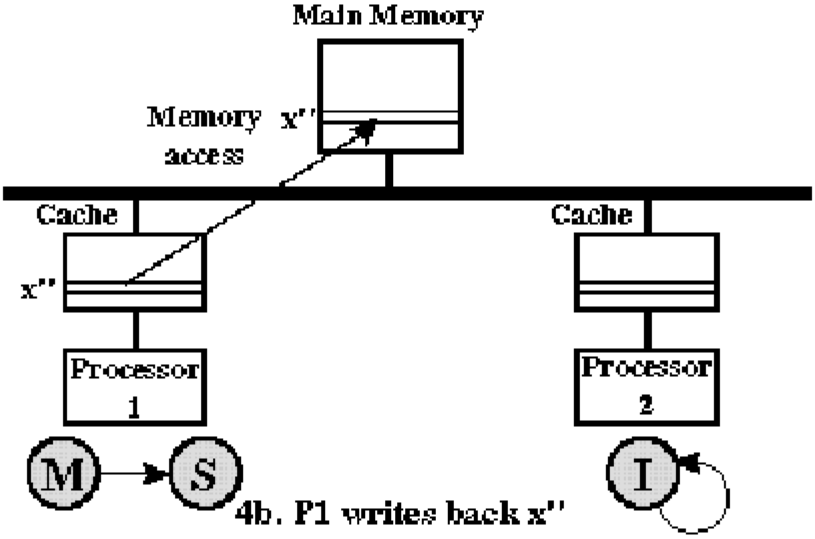
2. P1 writes x



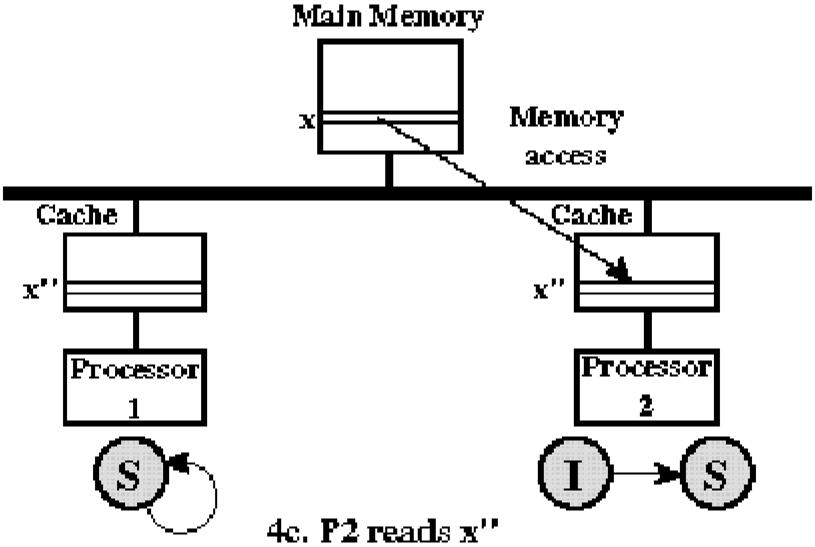
3. P1 writes x



4a. P2 reads x



4b. P1 writes back x''



4c. P2 reads x''

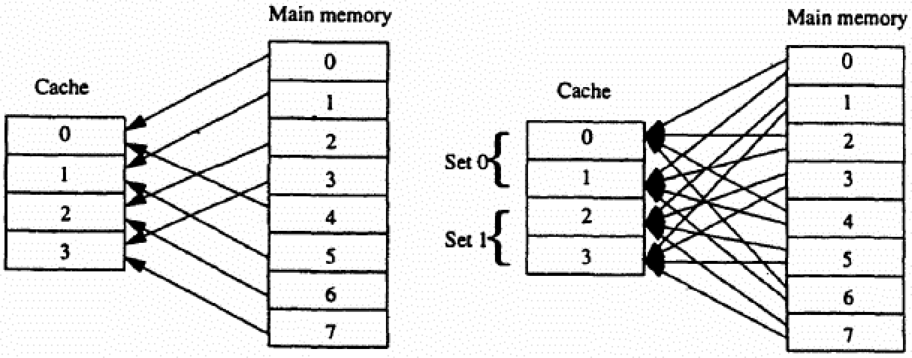
Problem 3

a. The protocol at Slide 4 is the simplest possible cache coherence protocol. It requires that all processors use a write-through policy. If a write is made to a location cached in remote caches, then the copies of the line in remote caches are invalidated. This approach is easy to implement but requires more bus and memory traffic because of the write-through policy.

b. This protocol at Slide 5 makes a distinction between shared and exclusive states. When a cache first loads a line, it puts it in the shared state. If the line is already in the modified state in another cache, that cache must block the read until the line is updated back to main memory, similar to the MESI protocol. The difference between the two is that the shared state is split into the shared and exclusive states for MESI. This reduces the number of write-invalidate operations on the bus.

Problem 4

(a) The mappings are shown in the following figure.

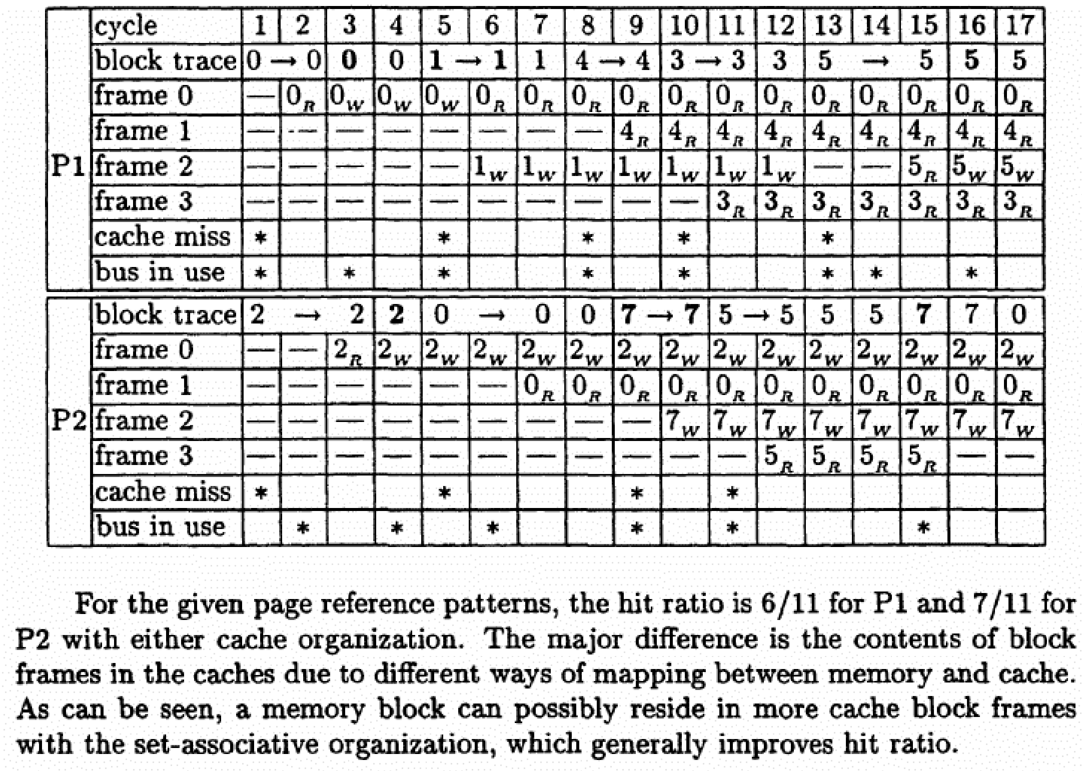


(i) Direct mapping

(ii) Two-way set-associative mapping

(b) The results are shown in the following two tables; the first table corresponds t direct mapping and the second two-way set-associative mapping. In each table, a arrow connecting the same block numbers indicates that the corresponding acces takes more than one cycle due to read/write misses or bus contention. In any case at most 3 cycles are required to complete an access in the case of a read/write mis coupled with bus contention. The subscript associated with a block indicates th state of that block (R for read-only, and W for read-write.)

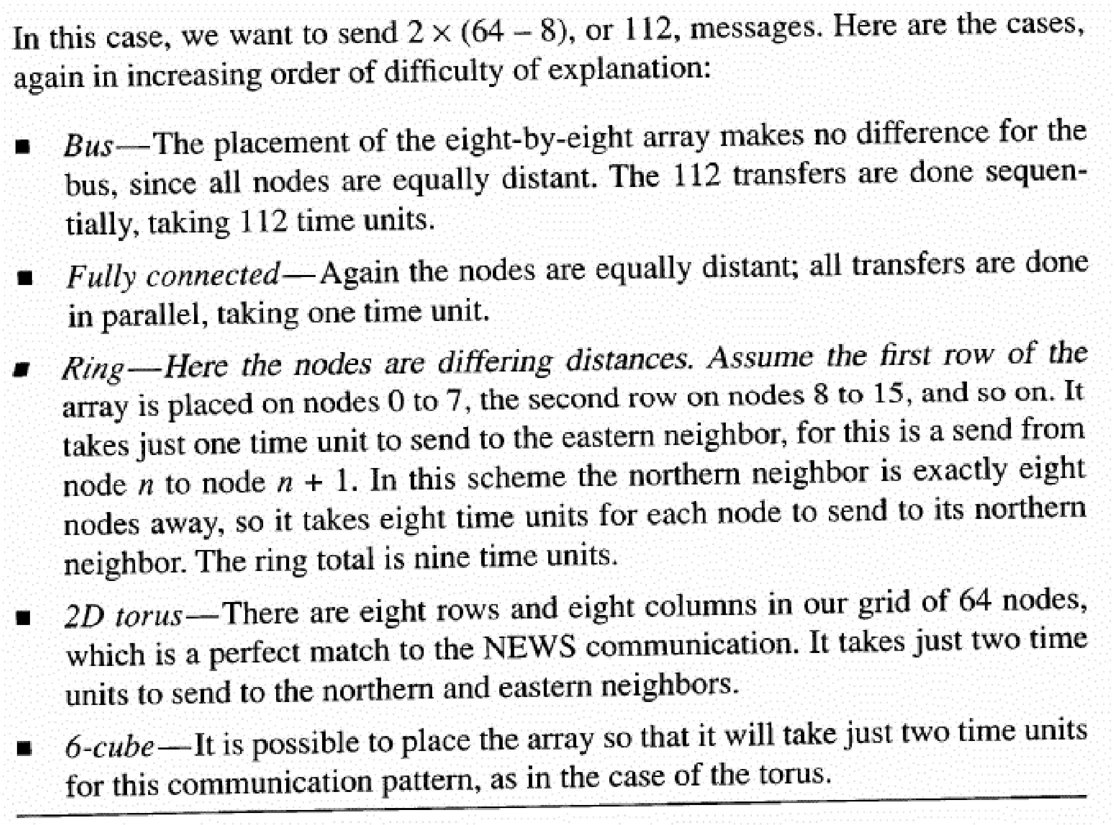
	cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
P1	block trace	0 → 0		**0**	0	**1 → 1**		1	4 → 4		3 → 3		3	5 → 5			**5**	5
	frame 0	—	0_R	0_W	0_W	0_W	0_R	0_R	—	4_R	4_R	4_R	4_R	4_R	4_R	4_R	4_R	4_R
	frame 1	—	—	—	—	—	1_W	1_W	1_W	1_W	1_W	1_W	1_W	—	—	5_R	5_W	5_W
	frame 2	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—	—
	frame 3	—	—	—	—	—	—	—	—	—	—	3_R	3_R	3_R	3_R	3_R	3_R	3_R
	cache miss	*				*			*		*			*				
	bus in use	*		*		*			*		*			*	*		*	
P2	block trace	2 → 2			**2**	0 → 0			0	**7 → 7**		5 → 5		5	5	**7**	7	0
	frame 0	—	—	—	—	—	—	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R
	frame 1	—	—	—	—	—	—	—	—	—	—	—	5_R	5_R	5_R	5_R	—	—
	frame 2	—	—	2_R	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W
	frame 3	—	—	—	—	—	—	—	—	—	7_W	7_W	7_W	7_W	7_W	7_W	7_W	7_W
	cache miss	*				*				*		*						
	bus in use		*		*		*			*		*				*		



	cycle	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
P1	block trace	0 → 0		**0**	0	**1 → 1**		1	4 → 4		3 → 3		3	5 → 5			**5**	5
	frame 0	—	0_R	0_W	0_W	0_W	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R
	frame 1	—	—	—	—	—	—	—	—	4_R	4_R	4_R	4_R	4_R	4_R	4_R	4_R	4_R
	frame 2	—	—	—	—	—	1_W	1_W	1_W	1_W	1_W	1_W	1_W	—	—	5_R	5_W	5_W
	frame 3	—	—	—	—	—	—	—	—	—	—	3_R	3_R	3_R	3_R	3_R	3_R	3_R
	cache miss	*				*			*		*			*				
	bus in use	*		*		*			*		*			*	*		*	
P2	block trace	2 → 2			**2**	0 → 0			0	**7 → 7**		5 → 5		5	5	**7**	7	0
	frame 0	—	—	2_R	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W	2_W
	frame 1	—	—	—	—	—	—	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R	0_R
	frame 2	—	—	—	—	—	—	—	—	—	7_W	7_W	7_W	7_W	7_W	7_W	7_W	7_W
	frame 3	—	—	—	—	—	—	—	—	—	—	—	5_R	5_R	5_R	5_R	—	—
	cache miss	*				*				*		*						
	bus in use		*		*		*			*		*				*		

For the given page reference patterns, the hit ratio is 6/11 for P1 and 7/11 for P2 with either cache organization. The major difference is the contents of block frames in the caches due to different ways of mapping between memory and cache. As can be seen, a memory block can possibly reside in more cache block frames with the set-associative organization, which generally improves hit ratio.

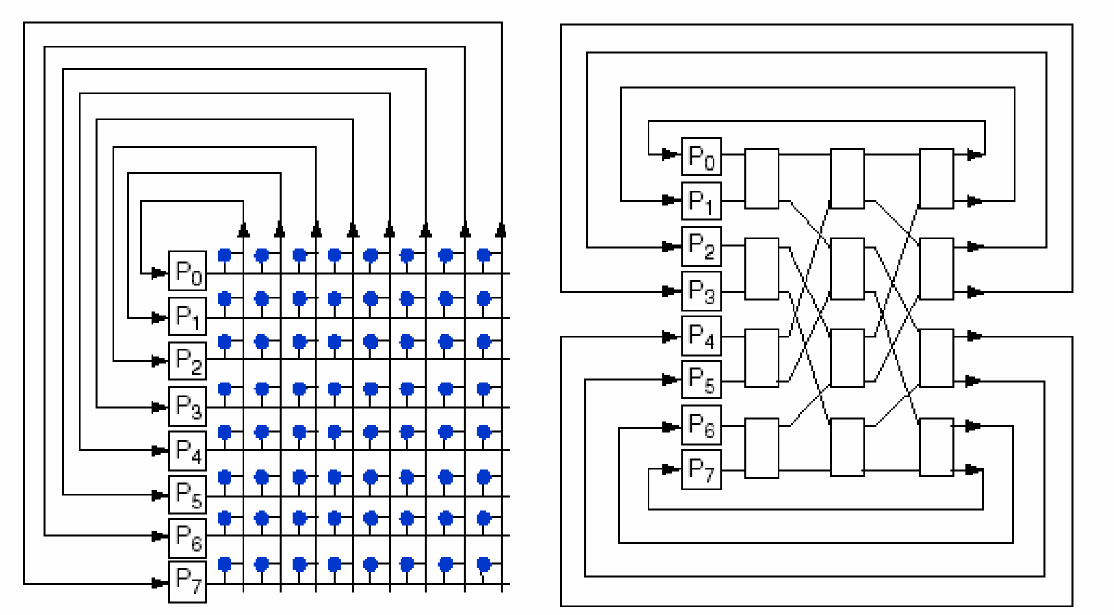
Problem 5



In this case, we want to send 2 × (64 – 8), or 112, messages. Here are the cases, again in increasing order of difficulty of explanation:

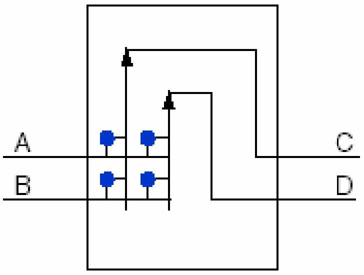
- *Bus*—The placement of the eight-by-eight array makes no difference for the bus, since all nodes are equally distant. The 112 transfers are done sequentially, taking 112 time units.
- *Fully connected*—Again the nodes are equally distant; all transfers are done in parallel, taking one time unit.
- *Ring—Here the nodes are differing distances. Assume the first row of the* array is placed on nodes 0 to 7, the second row on nodes 8 to 15, and so on. It takes just one time unit to send to the eastern neighbor, for this is a send from node *n* to node *n* + 1. In this scheme the northern neighbor is exactly eight nodes away, so it takes eight time units for each node to send to its northern neighbor. The ring total is nine time units.
- *2D torus*—There are eight rows and eight columns in our grid of 64 nodes, which is a perfect match to the NEWS communication. It takes just two time units to send to the northern and eastern neighbors.
- *6-cube*—It is possible to place the array so that it will take just two time units for this communication pattern, as in the case of the torus.

Problem 6



a. Crossbar

b. Omega network



c. Omega network switch box