Solutions for Chapter 9 Exercises

9.1 No solution provided.

9.2 The short answer is that x is always 2, and y can either be 2, 4, or 6. In a load-store architecture the code might look like the following:

Processor 1	Processor 2
load X into a register	load X into a register
increment register	increment register
store register back to X	store register back to Y
load Y into a register	
add two registers to register	
store register back to Y	

When considering the possible interleavings, only the loads/stores are really of interest. There are four activities of interest in process 1 and two in process 2. There are 15 possible interleavings, which result in the following:

111122: x = 2, y = 4
111212: x = 2, y = 4
111221: x = 2, y = 2
112112: x = 2, y = 4
112121: x = 2, y = 2
112211: x = 2, y = 6
121112: x = 2, y = 2
121121: x = 2, y = 2
121211: x = 2, y = 4
122111: x = 2, y = 4
211112: x = 2, y = 2
211121: x = 2, y = 2
211211: x = 2, y = 4
212111: x = 2, y = 4
221111: x = 2, y = 4

9.3 No solution provided.

9.4 No solution provided.

9.5 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.

9.6 Write-back cache with write-update cache coherency and four-word blocks. The block is in both caches and is initially clean. Assume 4-byte words and byte addressing. Assume that the bus moves one word at a time. Addresses 100 and 104 are contained in the one block starting at address 96.

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.

Total bus transactions = 2.

False-sharing appears as a performance problem under a write-invalidate cache coherency protocol. Writes from different processors to different words that happen to be allocated in the same cache block cause that block to ping-pong between the processor caches. That is, a dirty block can reside in only one cache at a time.

If we modify the cache in this exercise to use write-invalidate, the number of bus transactions increases to 9.

Step	Action	Comment
1	P1 writes 100	P1 issues a write-invalidate using one bus transaction to send the address 100; P2 removes the block from its cache; the block is now dirty in the P1 cache alone.
2	P2 writes 104	P2 issues a read-with-intent-to-modify and the block is moved from P1 to P2 using four bus transfers; P1 removes the block from its cache; the block is now dirty in the P2 cache alone
3	P1 reads 100	P1 issues a read miss and the P2 cache supplies the block to P1 and writes back to the memory at the same time using four bus transfers; the block is now clean in both cases.
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.

9.7 No solution provided.

9.8 No solution provided.

9.9 No solution provided.

9.10 No solution provided.

9.11 No solution provided.

9.12 No solution provided.

9.13 No solution provided.

9.14 No solution provided.