CompArch Question (Fall 2023): ECGR 4181

1. Consider a function *sumSquare* that, when given an integer n, returns the summation below. If n is not positive, then the function returns 0. The function *sumSquare* should call a procedure *Square* to calculate the square of each term in the summation:

$$n^2 + (n-1)^2 + (n-2)^2 + \ldots + 1^2$$

   A) Write the C codes for the functions *sumSquare* and *Square*. [5]
   B) Write the RISC-V assembly codes for *sumSquare* and *Square* with the assumption that data and instruction words are 32 bits wide, memory is byte addressable, and Stack Pointer (SP) will have to be used. Make assumptions as you see fit, but state and explain them clearly. Your assembly code should start from a main function which will call the *sumSquare* function, which in turn will call the *Square* function. The control of execution once *Square* ends, will have to move back to the function *sumSquare*, and then on its completion to the main thread. [25]
   C) Assume instruction word addresses of your choice and point out the contents of the PC (program counter) and SP for every line of assembly code. [5]

2. This problem concerns MOESI, an invalidation based snooping cache coherence protocol, for bus-based shared-memory multiprocessors with a single level of cache per processor. The MOESI protocol has five states. A block starting at address Addr can be in one of the following states in cache C:
   a. Modified: The block is present only in cache C and the data in the cache is dirty or modified (i.e., it reflects a more recent version than the copy in memory).
   b. Owned: The block is present in cache C and may also be present in other caches. The memory may not have an up-to-date copy of this block. The block is said to be owned by cache C and C must service the requests of other caches to this block since memory may not have an up-to-date copy.
   c. Exclusive: The block is present in a single cache (C) but is clean (i.e., memory has an up to-date copy of the block).
   d. Shared: The block is present in cache C and possibly present in other caches.
   e. Invalid: The block is not valid in cache C (space for the block may or may not be currently allocated in this cache).

   If the cache has a block in Owned state, then it services any requests to that block from other processors. Assume that the memory does NOT update its copy even if the request is a read by some other cache and the Owner cache has put the block on the bus. Hence, the owner cache remains in Owned state and continues to service other requests, until the block is replaced from its cache.

   Also assume that the only way to reach the Owned state is from the Modified or Exclusive state, when some other cache issues a read request for that block. If a cache C has a block in exclusive or modified state, then it is responsible for servicing any requests to that block from other processors. If the request is a read, then the cache C transitions to the Owned state, and memory does NOT update its copy. On replacement of a block in Owned or Modified state, the block is sent to memory, and memory

resumes responsibility for servicing subsequent requests to that block. Replacement of a block in Exclusive state is similar, except that the block need not be sent to memory (since memory already has a copy).

Assume that after a cache performs a transaction on the bus, there is a mechanism for it to know whether other caches have a copy of the requested block or not at that time. This enables the cache to determine whether to transition to exclusive state.

A) Draw the state transition diagram for the MOESI protocol as described above, and explain the different inter-state transitions. [20]

B) Consider a Block B in Owned state in the cache of processor P. Can B be in a non-invalid state in any other processor's cache? If yes, then what are the possible (non-invalid) states in which B could be in any of the other caches? If no, then explain why not. [3]

C) This part concerns the response of the cache of processor i to bus transactions initiated by the cache of processor j for a block that starts at address B (referred to as block B below). Fill out the following rows for the state transition table for the cache of processor i, showing the next state for block B in the cache and any action taken by the cache. Each entry should be filled out as: Next State/Action (e.g., S/Send block to memory) where Next State = M, O, E, S, or I Action = Send block to memory, Send block to cache, Send block to cache and memory, or No action Note: If an entry is not possible (i.e., the system cannot be in such a state), write "Not Possible". [6]

Current state of block B in cache of processor i	Read of block B by cache of processor j	Invalidate of block B by cache of processor j (with no read request for the block)	Read + Invalidate of block B by cache of processor j
M			
O			

D) Consider the following sequence of operations by two processors for a block that starts at address B. Determine the state of that block in the caches of both the processors after each operation in the sequence for the MOESI protocol. Both caches are initially empty and all lines are in the I state. The table below is provided to help organize your answer. [6]

Step	Operation	P1	P2
1	P1 reads B		
2	P1 writes B		
3	P2 writes B		
4	P1 reads B		
5	P1 writes B		
6	P2 reads B		