

# Shared Memory Consistency Models

## Shared Memory Consistency Models: A Tutorial

Sarita V. Adve\*  
Kourosh Gharachorloo\*

CSE 240B

Dean Tullsen

# Some sample code

```
Initially all pointers = null, all integers = 0.  
  
P1  
while (there are more tasks) {  
  Task = GetFromFreeList();  
  Task → Data = ...;  
  insert Task in task queue  
}  
Head = head of task queue;  
  
P2, P3, ..., Pn  
while (MyTask == null) {  
  Begin Critical Section  
  if (Head != null) {  
    MyTask = Head;  
    Head = Head → Next;  
  }  
  End Critical Section  
}  
... = MyTask → Data;
```

Figure 1: What value can a read return?

- What assumptions is the programmer making?

CSE 240B

Dean Tullsen

# Memory Consistency model impacts

- Cache/memory hierarchy design
- Cache design, write buffers, etc.
- CPU execution core (eg, ooo instruction scheduler)
- Compiler
- Programmer

CSE 240B

Dean Tullsen

# Uniprocessor Consistency

- What are the typical guarantees for uniprocessor consistency?
- How does this effect processor design/optimization?

CSE 240B

Dean Tullsen

# Sequential Consistency

- **Definition:** [A multiprocessor system is *sequentially consistent if*] the result of any execution is the same as if the operations of all the processors were executed in some sequential order, and the operations of each individual processor appear in this sequence in the order specified by its program.

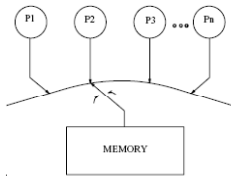


Figure 3: Programmer's view of sequential consistency.

# Sequential consistency implies:

- (1) maintaining program order among operations from individual processors, and
- (2) maintaining a single sequential order among operations from all processors.

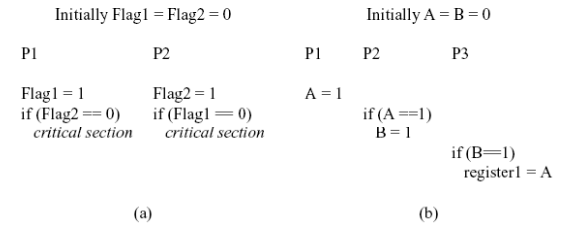
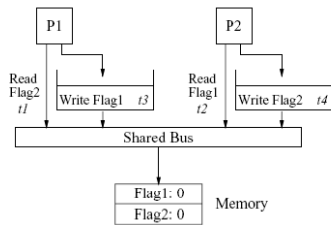
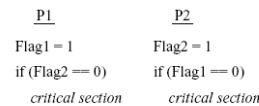


Figure 4: Examples for sequential consistency.

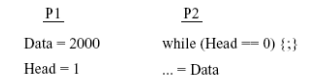
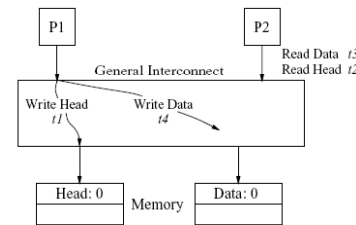
# Sequential Consistency and the Write Buffer



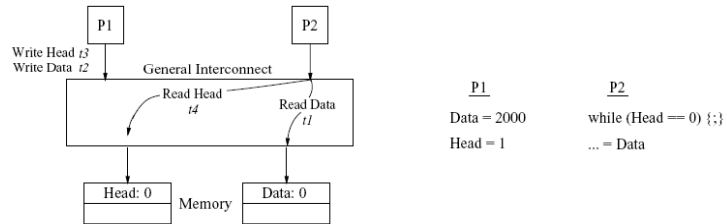
(a) write buffer



# Overlapping writes



## Non-blocking Reads



```

P1
Data = 2000
Head = 1

P2
while (Head == 0) {}
... = Data
    
```

CSE 240B

Dean Tullsen

## Sequential Consistency

- Additional changes because of caches?

CSE 240B

Dean Tullsen

## Sequential Consistency

- What does it mean for a write to “complete” in a multiprocessor?
- Why is sequential consistency tougher in a write-update coherence protocol than write-invalidate?

CSE 240B

Dean Tullsen

## Sequential Consistency and Compilers

- What are the issues?
- How to avoid?

```

P1
Data = 2000
Head = 1

P2
while (Head == 0) {}
... = Data
    
```

CSE 240B

Dean Tullsen



## Release Consistency

operations. For RCsc, the constraints are as follows:

- acquire → all, all → release, and special → special.

For RCpc, the write to read program order among special operations is eliminated:

- acquire → all, all → release, and special → special except for a special write followed by a special read.

## Compiler, again

- What does it imply for the compiler if the processor supports relaxed consistency?