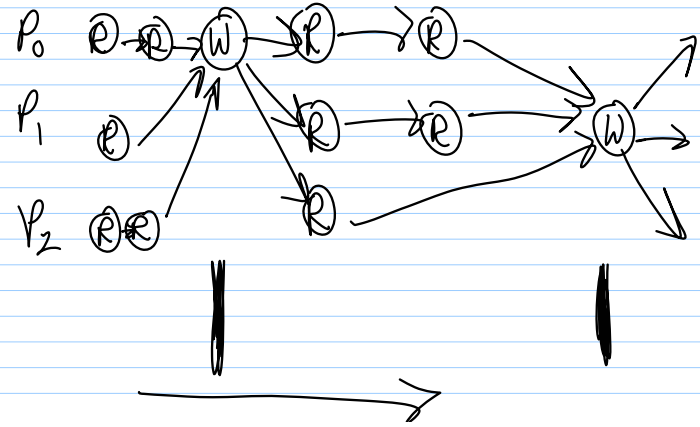


# Memory Consistency

- A mem op  $m_2$  is subsequent mem op  $m_1$  if the ops are issued by same proc. and  $m_2$  follows  $m_1$  in prog. order
- Read  $R$  is subsequent to write  $W$  if  $R$  generates bus xaction that follows  $W$
- Write  $W_2$  is subsequent to op  $M$  if  $M$  generates bus xaction and the xaction for  $W$  follows that for  $M$
- Write  $W$  is subsequent to read  $R$  if  $R$  does not generate bus xaction and is not already separated from  $W$  by another bus xaction

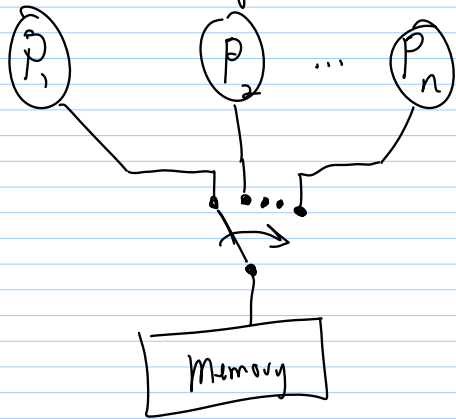


|                   |                    |
|-------------------|--------------------|
| $P_1$             | $P_2$              |
| init values all 0 |                    |
| $A = 1;$          | while (flag == 0); |
| flag = 1;         | print A;           |

|                 |              |
|-----------------|--------------|
| $P_1$           | $P_2$        |
| init values = 0 |              |
| 1a) $A = 1;$    | 2a) print B; |
| 1b) $B = 2;$    | 2b) print A; |

memory consistency model

sequential consistency



"Lamport 1979

A multiprocessor is seq. consistent if the result of any execution is the same as if the ops of all the procs were executed in some seq. order, and the ops of each indiv. proc appear in this sequence in the order specified by its program."

$P_1$                        $P_2$

init values 0

1a)  $A=1;$                       2a) print B;

1b)  $B=2;$                       2b) print A;

legal outcomes  $(A,B) = (0,0), (1,0), (1,2)$  not  $(0,2)$

$1b \rightarrow 1a \rightarrow 2b \rightarrow 2a$  ?  
 $\rightarrow 2a \rightarrow 2b$  ?     $\checkmark$  yes SC

$1b \rightarrow 2a \rightarrow 2b \rightarrow 1a$      $\times$  not SC