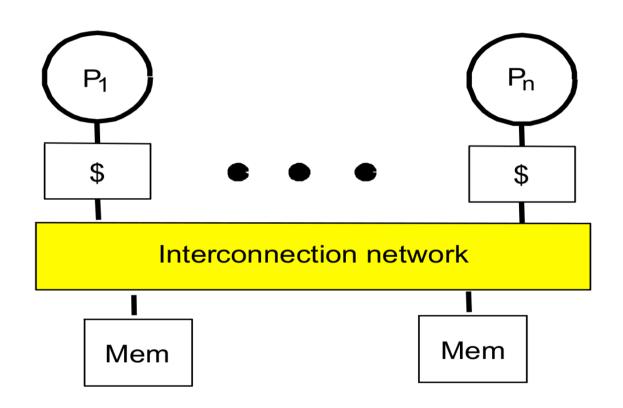
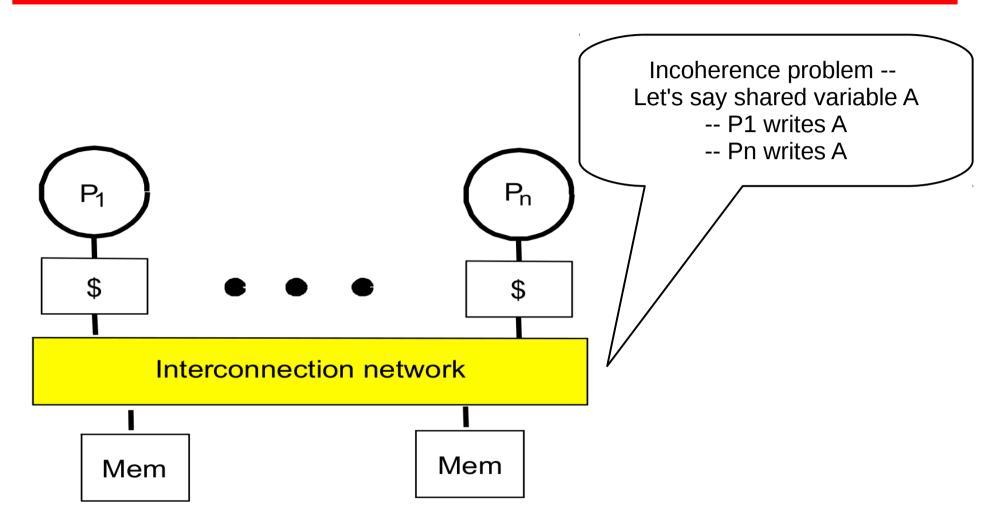
Tutorial 8

ECSE 420 - Tutorial 8

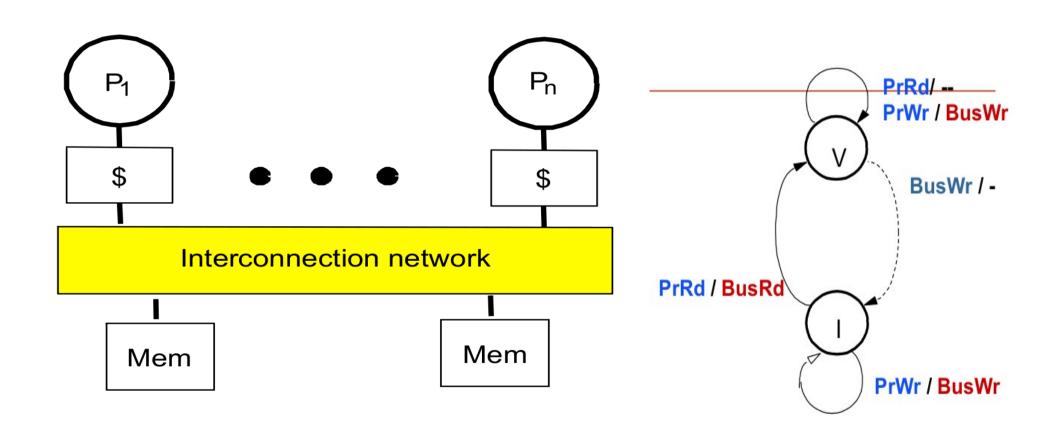
Dimitrios Stamoulis

TR 4110 November 10, 2014

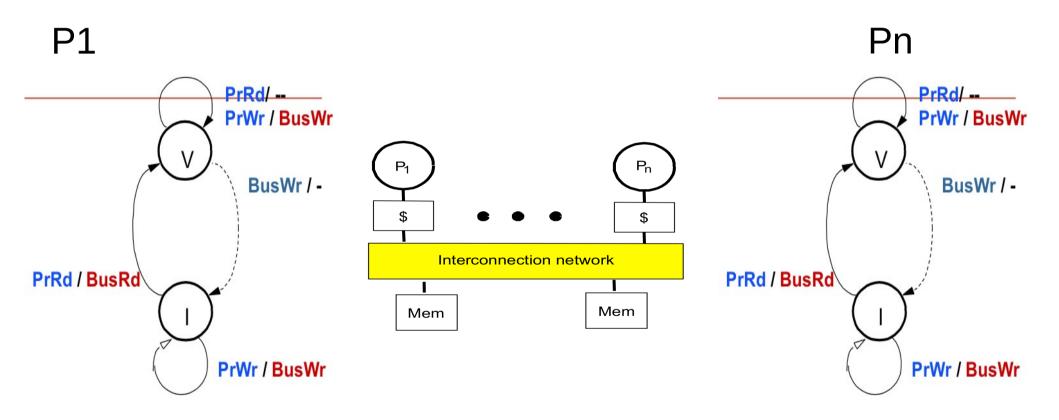




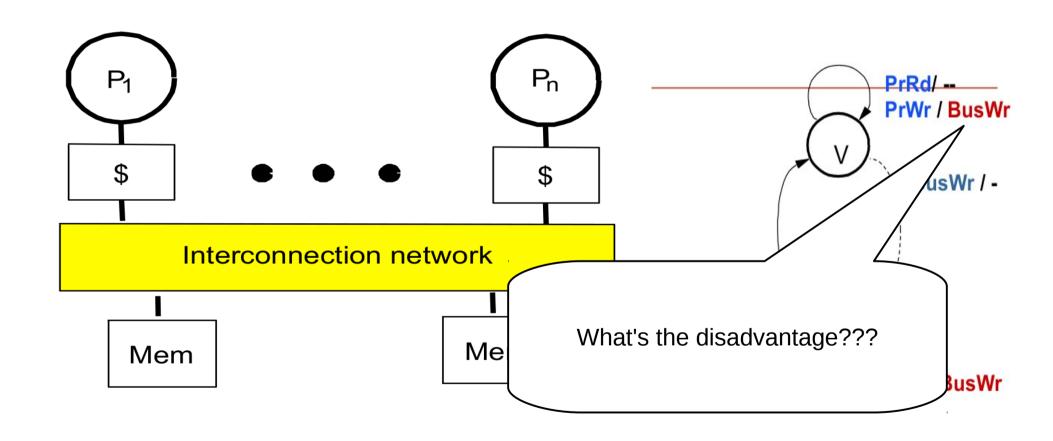
Write-through Invalidate Protocol



Write-through Invalidate Protocol

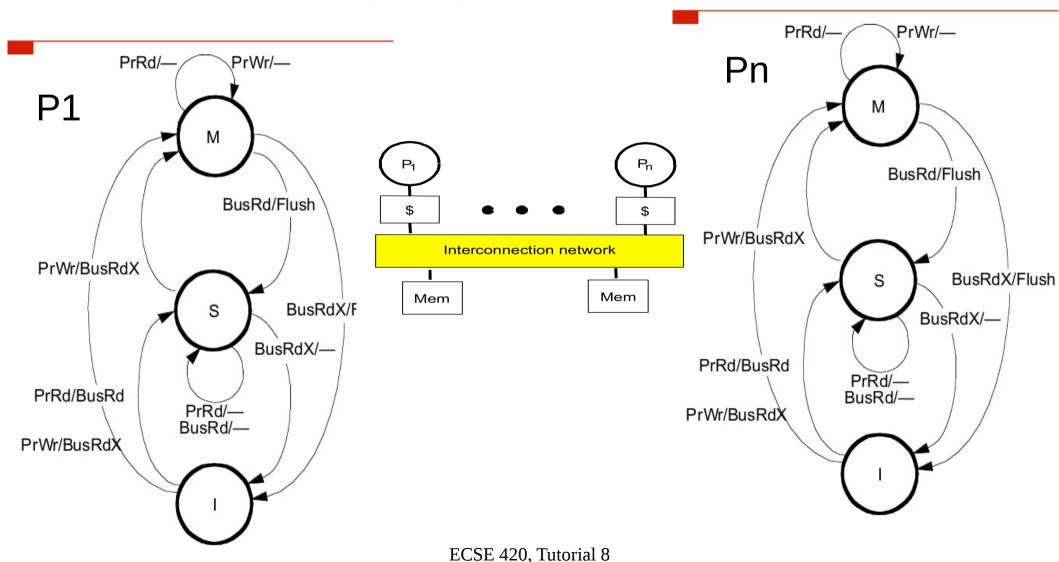


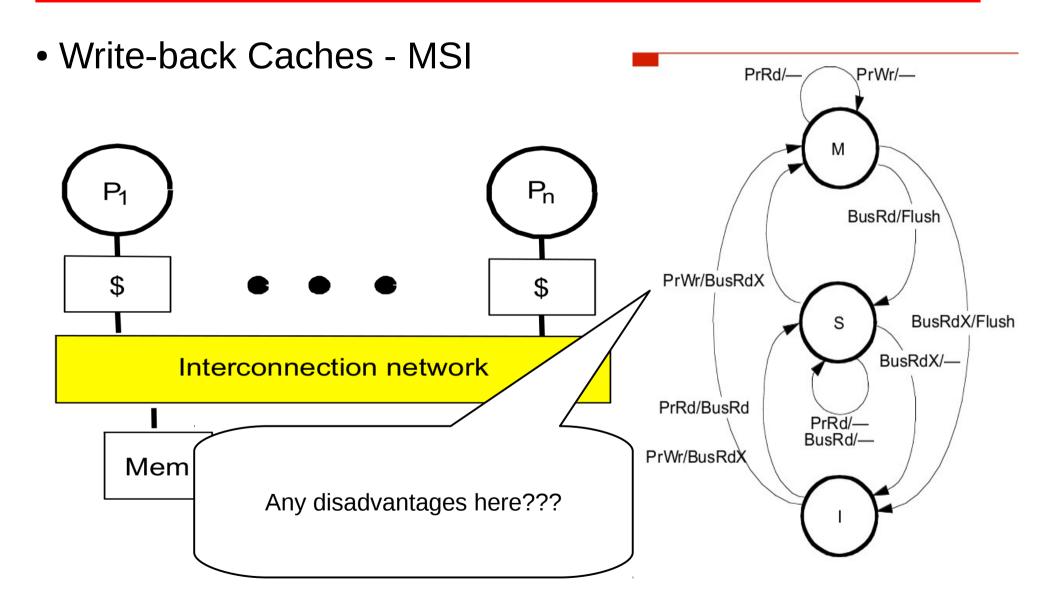
Write-through Invalidate Protocol



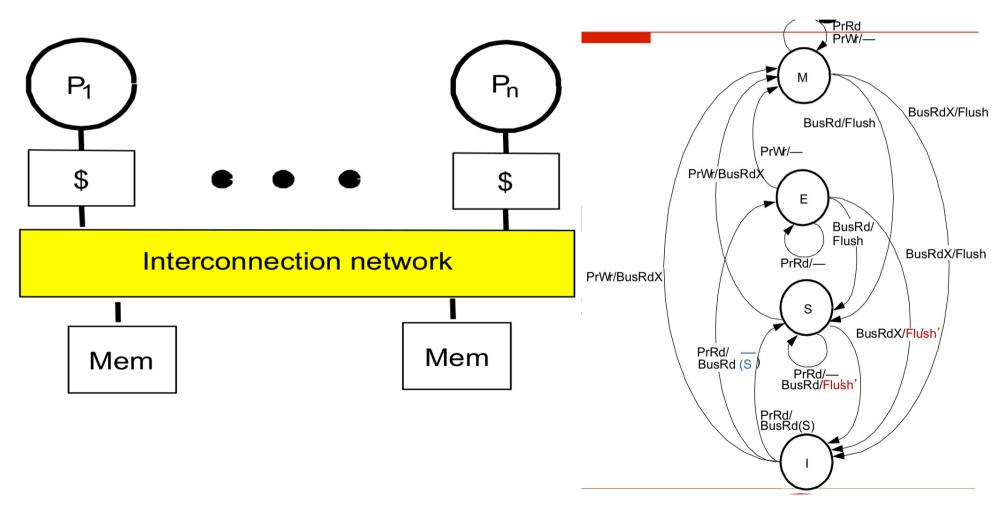
 Write-back Caches - MSI PrRd/-PrWr/— M P_n BusRd/Flush PrWr/BusRdX BusRdX/Flush BusRdX/-Interconnection network PrRd/BusRd PrRd/— BusRd/-PrWr/BusRdX Mem Mem

Write-back Caches - MSI

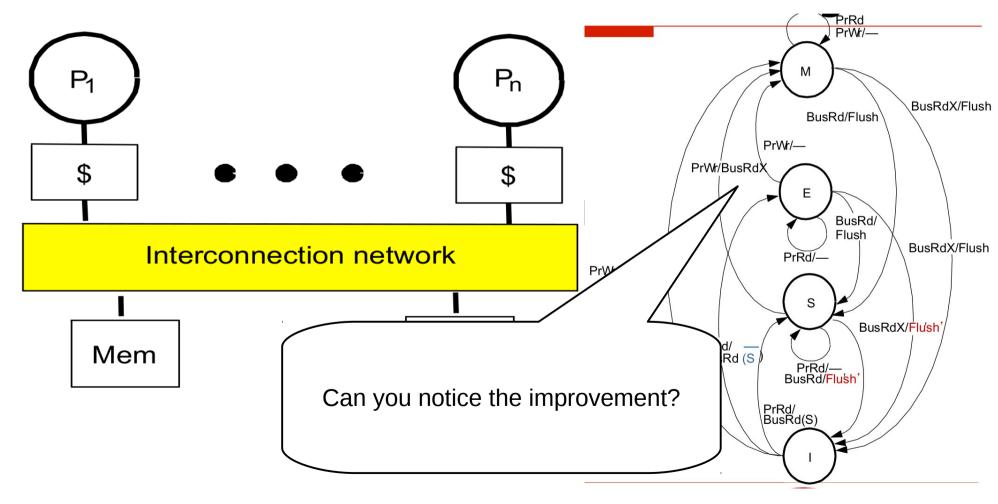




Write-back Caches - MESI



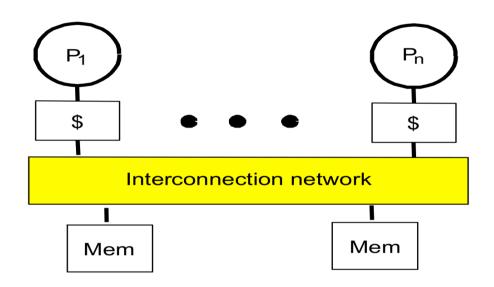
Write-back Caches - MESI



Let's solve by hand an example

- 2 CPUs
- Write-back caches
- MESI protocol
- 16 bytes per line
- Caches initially empty

Instructions
P0 writes to 8c34
P0 reads 7c34
P1 writes to 8c30
P1 reads 7c34
P1 reads 9c38
P0 reads 8c3c
P0 writes to 8c3c



Example

Instructions

P0 w 8c34

P0 r 7c34

P1 w 8c30

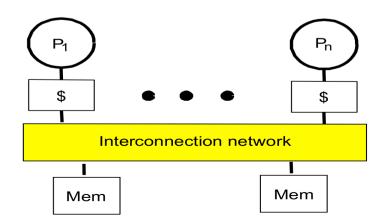
P1 r 7c34

P1 r 9c38

P0 r 8c3c

P0 w 8c3c

Memory						
Location	0	4	8	С		
7c3x	0000	0000	0000	0000		
8c3x	0000	0000	0000	0000		
9c3x	0000	0000	0000	0000		



Example

Instructions

P0 w 8c34

P0 r 7c34

P1 w 8c30

P1 r 7c34

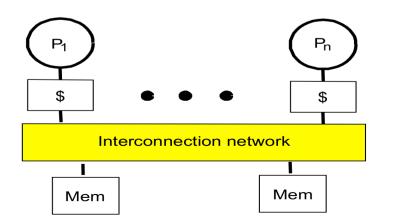
P1 r 9c38

P0 r 8c3c

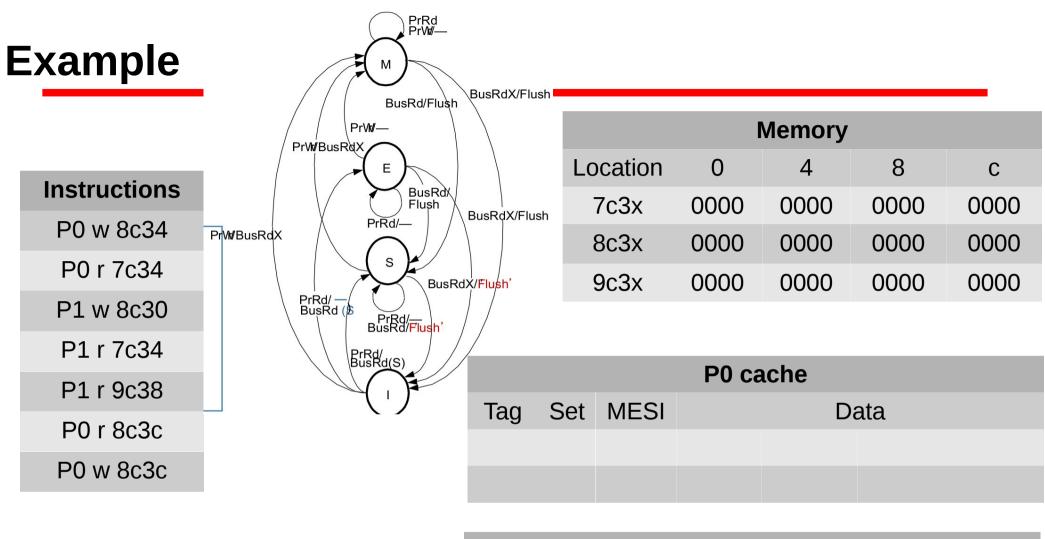
P0 w 8c3c

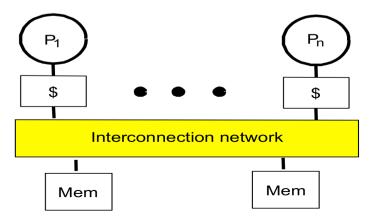
Memory						
Location	0	4	8	С		
7c3x	0000	0000	0000	0000		
8c3x	0000	0000	0000	0000		
9c3x	0000	0000	0000	0000		

P0 cache					
Tag	Set	MESI	Data		

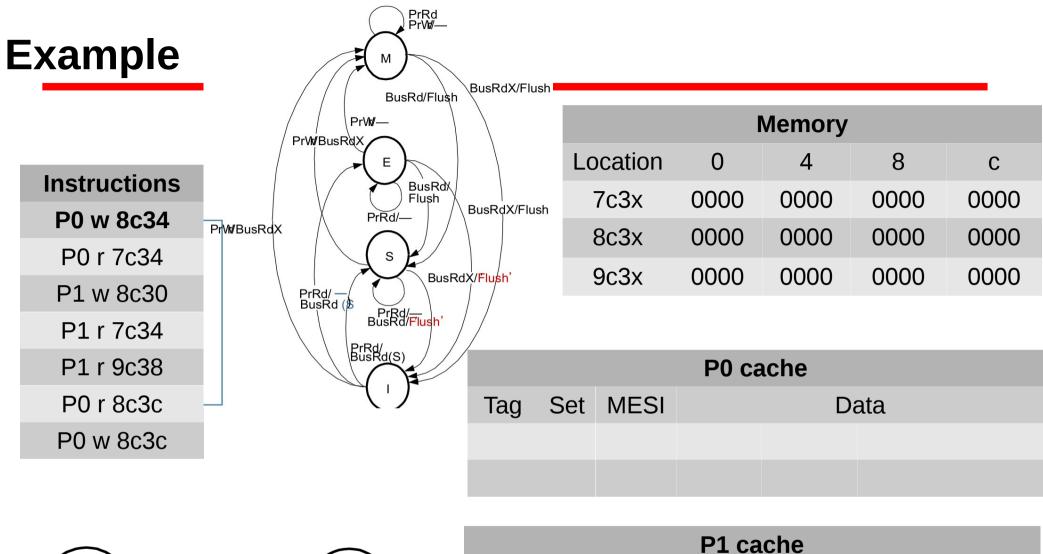


P1 cache				
Tag	Set	MESI	Data	

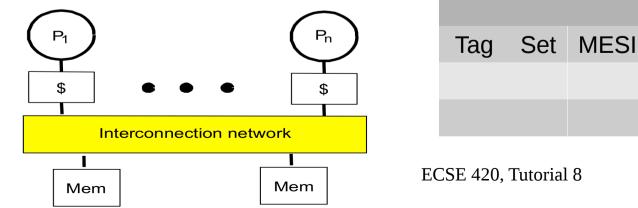




	P1 cache					
Tag	Set	MESI		Da	ata	

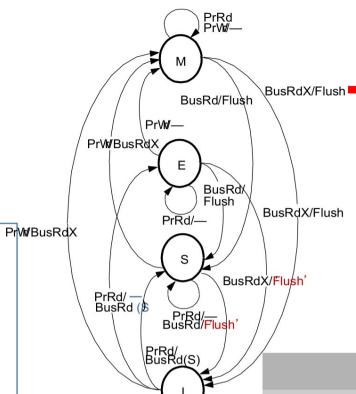


Data



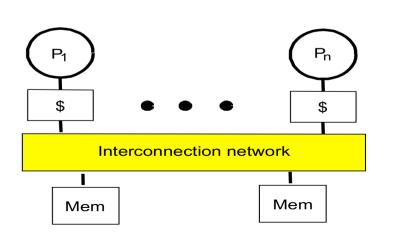




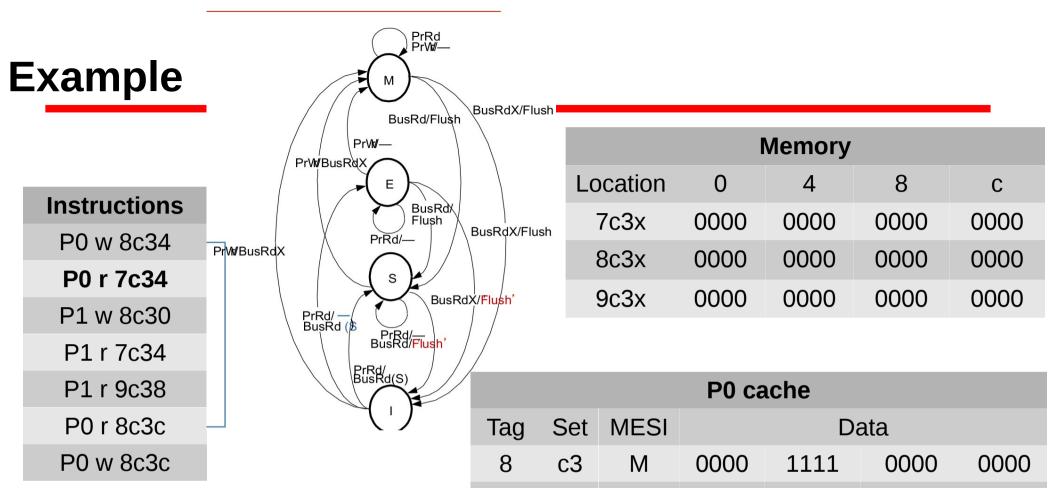


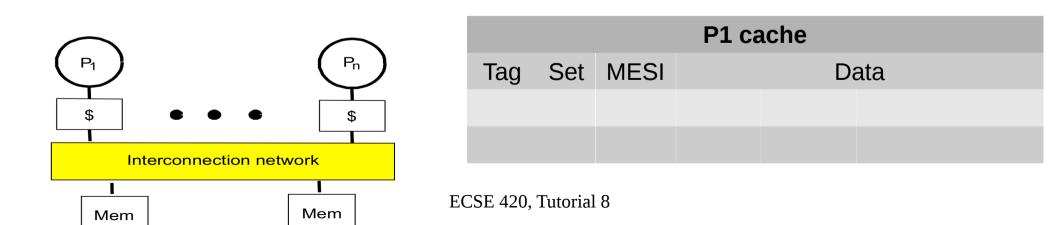
Memory						
Location	0	4	8	С		
7c3x	0000	0000	0000	0000		
8c3x	0000	0000	0000	0000		
9c3x	0000	0000	0000	0000		

P0 cache						
Tag	Set	MESI		Da	ata	
8	c3	М	0000	1111	0000	0000



	P1 cache					
Tag	Set	MESI		Da	ata	







Instructions

P0 w 8c34

P0 r 7c34

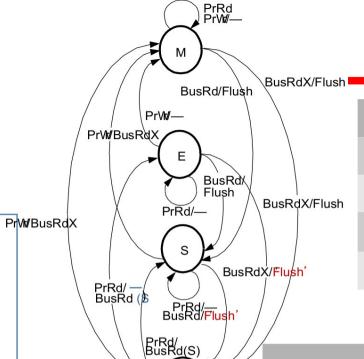
P1 w 8c30

P1 r 7c34

P1 r 9c38

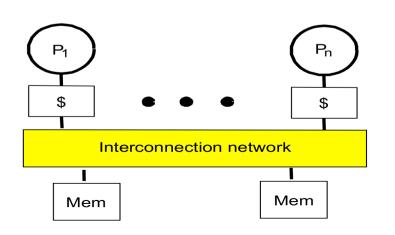
P0 r 8c3c

P0 w 8c3c



Memory						
Location	0	4	8	С		
7c3x	0000	0000	0000	0000		
8c3x	0000	0000	0000	0000		
9c3x	0000	0000	0000	0000		

	P0 cache							
Tag	Set	MESI		Da	ata			
8	c3	М	0000	1111	0000	0000		
7	с3	Е	0000	0000	0000	0000		



P1 cache				
Tag	Set	MESI	Data	



Instructions

P0 w 8c34

P0 r 7c34

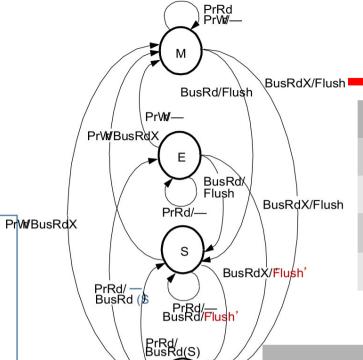
P1 w 8c30

P1 r 7c34

P1 r 9c38

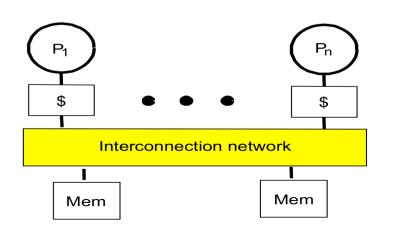
P0 r 8c3c

P0 w 8c3c



Memory									
Location	Location 0 4 8 c								
7c3x	0000	0000	0000	0000					
8c3x	0000	0000	0000	0000					
9c3x	0000	0000	0000	0000					

	P0 cache								
Tag	Set	MESI		Da	ata				
8	c3	M	0000	1111	0000	0000			
7	c3	E	0000	0000	0000	0000			



	P1 cache						
Tag Set MESI Data							



P0 w 8c34 P0 r 7c34

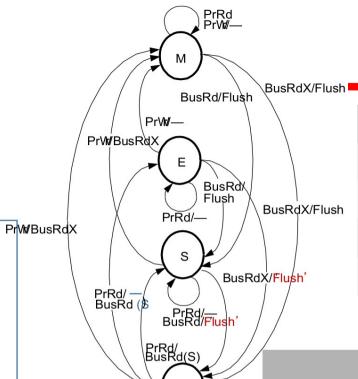
P1 w 8c30

P1 r 7c34

P1 r 9c38

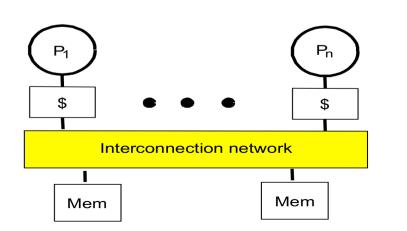
P0 r 8c3c

P0 w 8c3c



Memory									
Location	Location 0 4 8 c								
7c3x	0000	0000	0000	0000					
8c3x	0000	1111	0000	0000					
9c3x	0000	0000	0000	0000					

			P0 ca	che		
Tag	Set	MESI		Da	ata	
8	c3	I				
7	c3	Е	0000	0000	0000	0000



	P1 cache								
Tag	Set	MESI	l Data						
8	сЗ	M	2222	1111	0000	0000			

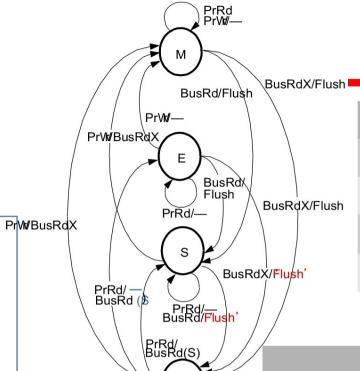


P0 w 8c34 P0 r 7c34 P1 w 8c30

P1 r 7c34

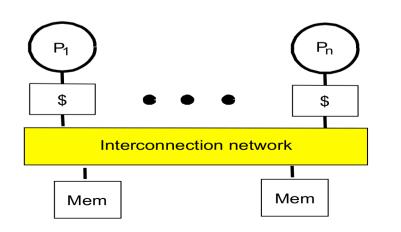
P1 r 9c38 P0 r 8c3c

P0 w 8c3c



Memory									
Location	Location 0 4 8 c								
7c3x	0000	0000	0000	0000					
8c3x	0000	1111	0000	0000					
9c3x	0000	0000	0000	0000					

			P0 ca	che		
Tag	Set	MESI		Da	ata	
8	c3	I				
7	сЗ	Е	0000	0000	0000	0000



P1 cache							
Tag	Set	MESI	MESI Data				
8	c3	М	2222	1111	0000	0000	



Instructions P0 w 8c34

P0 r 7c34

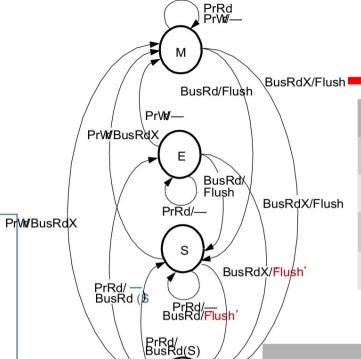
P1 w 8c30

P1 r 7c34

P1 r 9c38

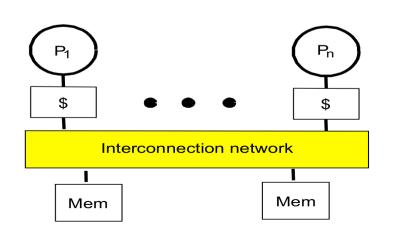
P0 r 8c3c

P0 w 8c3c



Memory									
Location	Location 0 4 8 c								
7c3x	0000	0000	0000	0000					
8c3x	0000	1111	0000	0000					
9c3x	0000	0000	0000	0000					

			P0 ca	che		
Tag	Set	MESI		Da	ata	
8	c3	I				
7	с3	S	0000	0000	0000	0000



	P1 cache							
Tag	Set	MESI	Data					
8	сЗ	М	2222	1111	0000	0000		
7	c3	S	0000	0000	0000	0000		



P0 w 8c34 P0 r 7c34

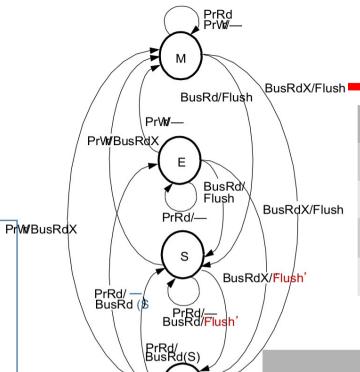
P1 w 8c30

P1 r 7c34

P1 r 9c38

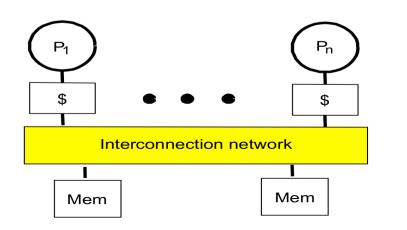
P0 r 8c3c

P0 w 8c3c



Memory										
Location 0 4 8 c										
7c3x	0000	0000	0000	0000						
8c3x	0000	1111	0000	0000						
9c3x	0000	0000	0000	0000						

P0 cache											
Tag	Set	MESI									
8	c3	I									
7	сЗ	S	0000	0000	0000	0000					



	P1 cache										
Tag	g Set MESI Data										
8	c3	M	2222	1111	0000	0000					
7	c3	S	0000	0000	0000	0000					



Po r 7c34 P1 w 8c30

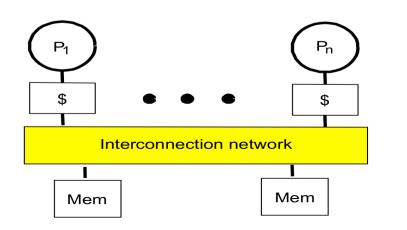
P1 r 7c34

P1 r 9c38

P0 r 8c3c P0 w 8c3c BusRdX/Flush
PrW/BusRdX
PrRd/—
PrRd/—
BusRdX/Flush
PrRd/—
BusRd/Flush'
BusRd/Flush'
PrRd/—
BusRd/Flush'
BusRd(S)

Memory										
Location 0 4 8 c										
7c3x	0000	0000	0000	0000						
8c3x	2222	1111	0000	0000						
9c3x	0000	0000	0000	0000						

	P0 cache										
Tag	Set	MESI	Data								
8	c3	I									
7	сЗ	S	0000	0000	0000	0000					



	P1 cache										
Tag	Set	MESI	Data								
9	c3	Е	0000	0000	0000	0000					
7	c3	S	0000	0000	0000	0000					





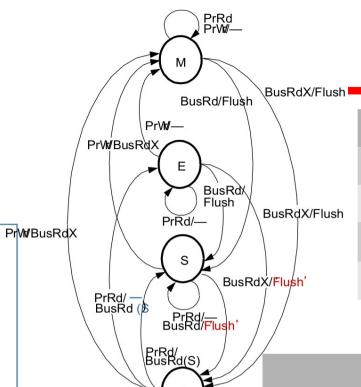
P1 w 8c30

P1 r 7c34

P1 r 9c38

P0 r 8c3c

P0 w 8c3c



Memory										
Location 0 4 8 c										
7c3x	0000	0000	0000	0000						
8c3x	2222	1111	0000	0000						
9c3x	0000	0000	0000	0000						

	P0 cache										
Tag	Set	MESI									
8	c3	I									
7	сЗ	S	0000	0000	0000	0000					

\$ • •	Pn \$
Interconnect	ion network
I Mem	Mem

	P1 cache										
Tag	Set	MESI	Data								
9	c3	E	0000	0000	0000	0000					
7	с3	S	0000	0000	0000	0000					



P0 w 8c34 P0 r 7c34

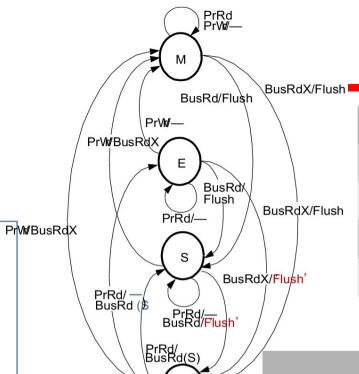
P1 w 8c30

P1 r 7c34

P1 r 9c38

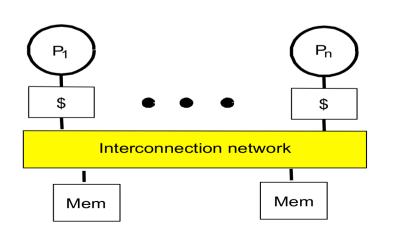
P0 r 8c3c

P0 w 8c3c



Memory										
Location 0 4 8 c										
7c3x	0000	0000	0000	0000						
8c3x	2222	1111	0000	0000						
9c3x	0000	0000	0000	0000						

P0 cache											
Tag	Set	MESI	Data								
8	c3	Е	2222	1111	0000	0000					
7	с3	S	0000	0000	0000	0000					



	P1 cache							
Tag Set MESI Data								
9	c3	Е	0000	0000	0000	0000		
7	c3	S	0000	0000	0000	0000		



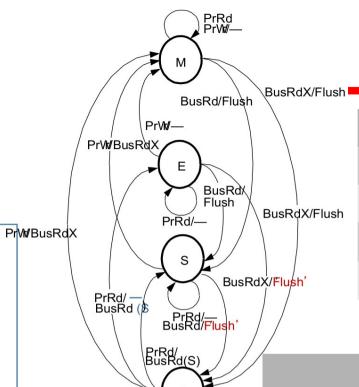


P1 r 7c34

P1 r 9c38

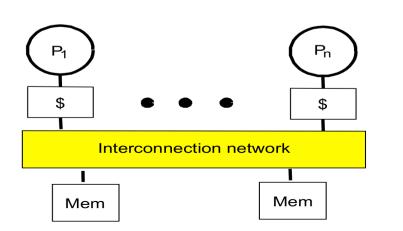
P0 r 8c3c

P0 w 8c3c



Memory								
Location 0 4 8 c								
7c3x	0000	0000	0000	0000				
8c3x	2222	1111	0000	0000				
9c3x	0000	0000	0000	0000				

P0 cache							
Tag	Set	MESI		Da	ata		
8	c3	Е	2222	1111	0000	0000	
7	c3	S	0000	0000	0000	0000	



P1 cache							
Tag	Set	MESI		Da	ata		
9	c3	E	0000	0000	0000	0000	
7	с3	S	0000	0000	0000	0000	



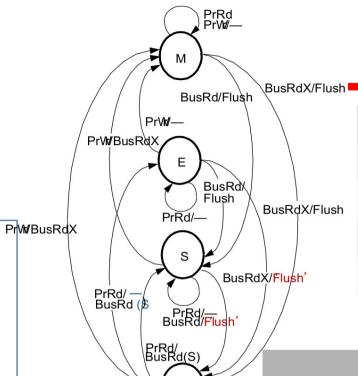
P0 w 8c34 P0 r 7c34 P1 w 8c30

P1 r 9c38

P1 r 7c34

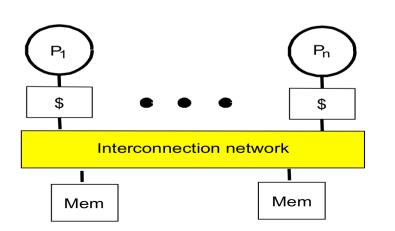
P0 r 8c3c

P0 w 8c3c



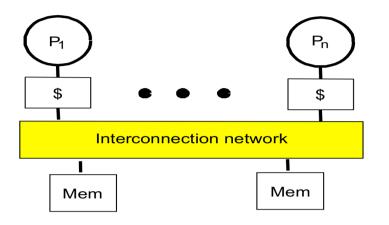
Memory								
Location 0 4 8 c								
7c3x	0000	0000	0000	0000				
8c3x	2222	1111	0000	0000				
9c3x	0000	0000	0000	0000				

P0 cache								
Tag	Set	MESI	Data					
8	c3	M	2222	1111	0000	3333		
7	с3	S	0000	0000	0000	0000		



	P1 cache							
Tag Set MESI Data								
9	c3	Е	0000	0000	0000	0000		
7	c3	S	0000	0000	0000	0000		

Coherence

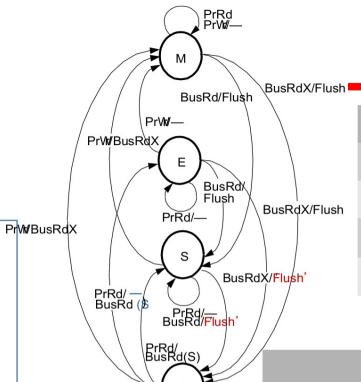


• Coherence: Writes to a single location become visible to all in the same order



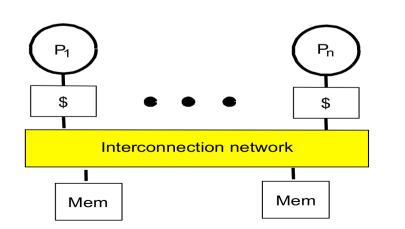
P0 w 8c34 P0 r 7c34 P1 w 8c30 P1 r 7c34 P1 r 9c38 P0 r 8c3c

P0 w 8c3c

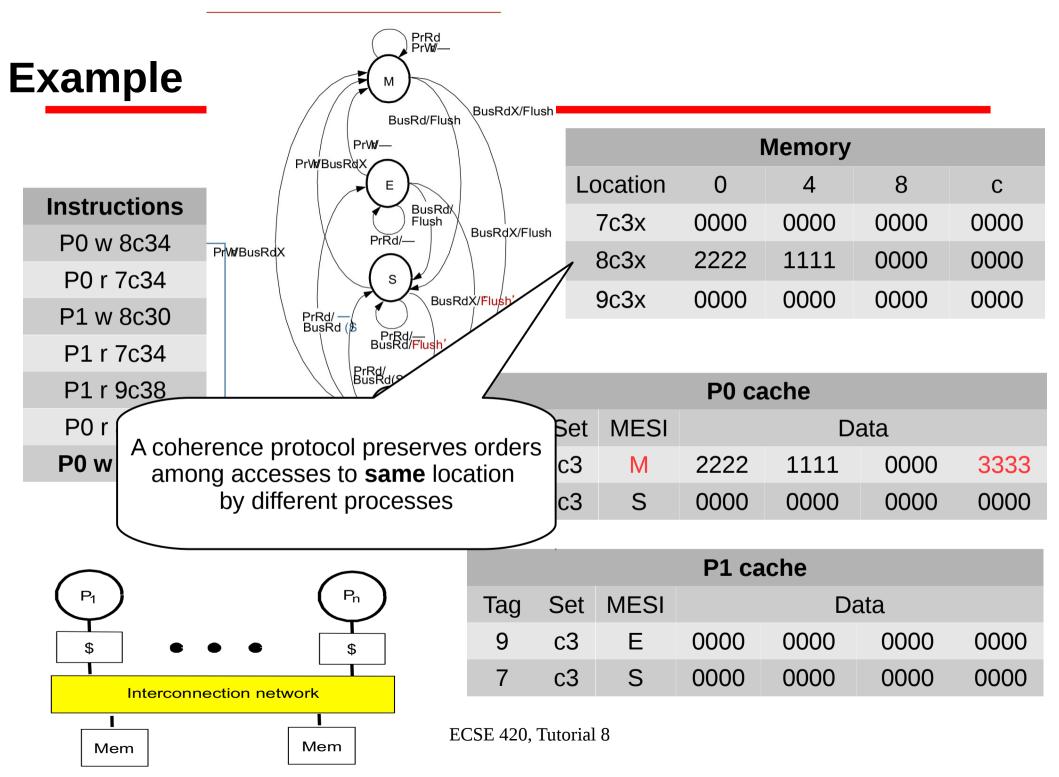


Memory								
Location 0 4 8 c								
7c3x	0000	0000	0000	0000				
8c3x	2222	1111	0000	0000				
9c3x	0000	0000	0000	0000				

/	P0 cache								
Tag Set MESI Data									
	8	c3	M	2222	1111	0000	3333		
	7	сЗ	S	0000	0000	0000	0000		



	P1 cache							
Tag Set MESI Data								
9	c3	Е	0000	0000	0000	0000		
7	c3	S	0000	0000	0000	0000		



Example Instructions

Imagine an OoO execution case!!

Memory

P0 w 8c34

P0 r 7c34

P1 w 8c30

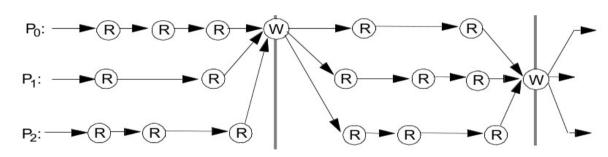
P1 r 7c34

P1 w 9c38

P0 r 8c3c

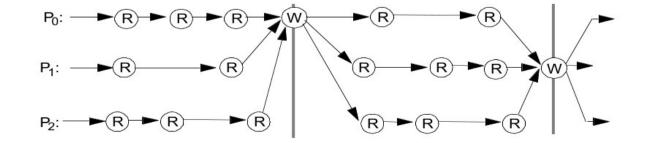
P0 w 8c3c

Location	0	4	8	С
7c3x	0000	0000	0000	0000
8c3x	2222	1111	0000	0000
9c3x	0000	0000	0000	0000



Mem: 8c3x

Mem: 9c3x



Example

Imagine an OoO execution case!!

Memory

Instructions

P0 w 8c34

P0 r 7c34

P1 w 8c30

P1 r 7c34

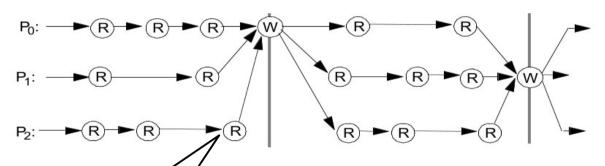
P1 w 9c38

P0 r 8c3c

P0 w 8c3c

Location	0	4	8	С
7c3x	0000	0000	0000	0000
8c3x	2222	1111	0000	0000
9c3x	0000	0000	0000	0000

►(R)



Mem: 8c3x

We will need to respect a "global" order...

Example

Imagine an OoO execution case!!

Memory

Location

0

4

8

C

Instructions

P0 w 8c34

P0 r 7c34

P1 w 8c30

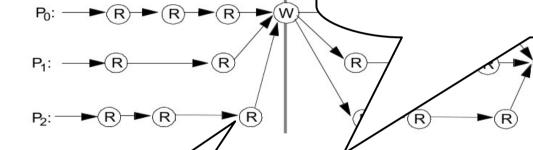
P1 r 7c34

P1 w 9c38

P0 r 8c3c

P0 w 8c3c

We will need an order in which memory operations (from any process) can appear to execute with **respect to one another**



We will need to respect a "global" order...

Memory Consistency

• Let's check a simple example where 2 memory locations are accessed.

• Let's check a simple example where 2 memory locations are accessed.

P ₁	P_2
/*Assume initial values of A and B are 0*/	
(1a) A = 1;	(2a) print B;
(1b) $B = 2;$	(2b) print A;

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

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

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

- (0,0):
- •
- •
- lacktriangle

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): 2a
- •
- •
- lacktriangle

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): 2a \rightarrow 2b
- •

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- •

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0):
- •

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): 1a

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b$

•

lacktriangle

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

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

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

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2):

lacktriangle

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): 1a

•

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): 1a \rightarrow 1b

•

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$

lacktriangle

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0:

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0: 2b → 1a

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0$$
*/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0: $2b \rightarrow 1a$, but from P1 order $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0: $2b \rightarrow 1a$, but from P1 order $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$ To get B=2:

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

(1a) A = 1;

(1b) B = 2;

(2a) print B;

(2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0: $2b \rightarrow 1a$, but from P1 order $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$

To get B=2: $1b \rightarrow 2a$

 Let's check a simple example where 2 memory locations are accessed.

P₁

/*Assume initial values of A and B are
$$0*$$
/

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

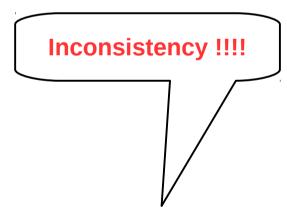
(1b) $B = 2$; (2b) print A;

We can have 4 outcomes for (A,B):

- (0,0): $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$
- (1,0): $1a \rightarrow 2a \rightarrow 2b \rightarrow 1b$
- (1,2): $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$
- (0,2):

To get A=0: $2b \rightarrow 1a$, but from P1 order $2a \rightarrow 2b \rightarrow 1a \rightarrow 1b$

To get B=2: $1b \rightarrow 2a$, but from P2 order $1a \rightarrow 1b \rightarrow 2a \rightarrow 2b$



Lab 2 – Any questions??



