

System Modeling

Introduction

Rugby Meta-Model

Finite State Machines

Petri Nets

Untimed Model of Computation

Synchronous Model of Computation

Timed Model of Computation

Integration of Computational Models

Tightly Coupled Process Networks

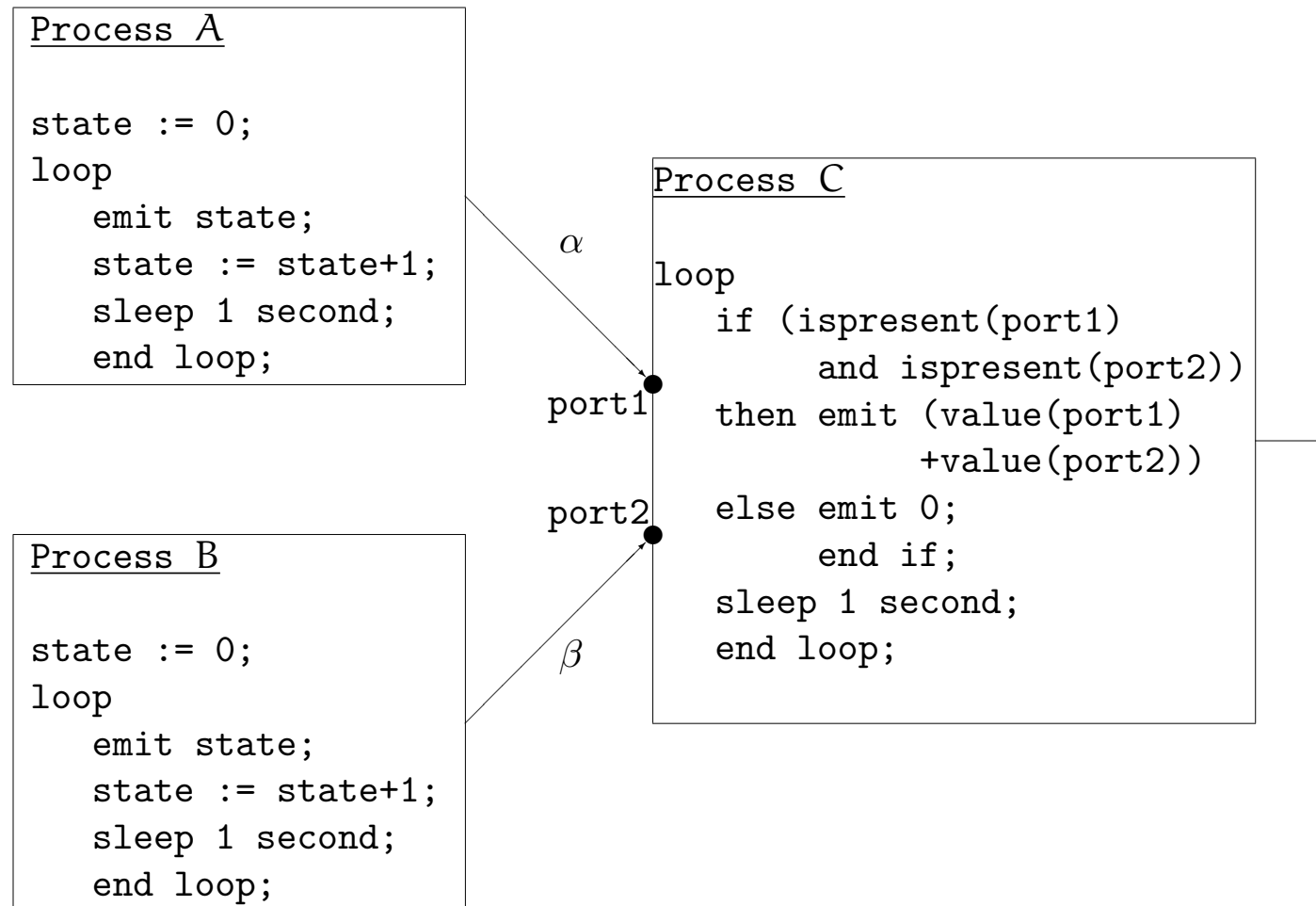


Tightly Coupled Networks

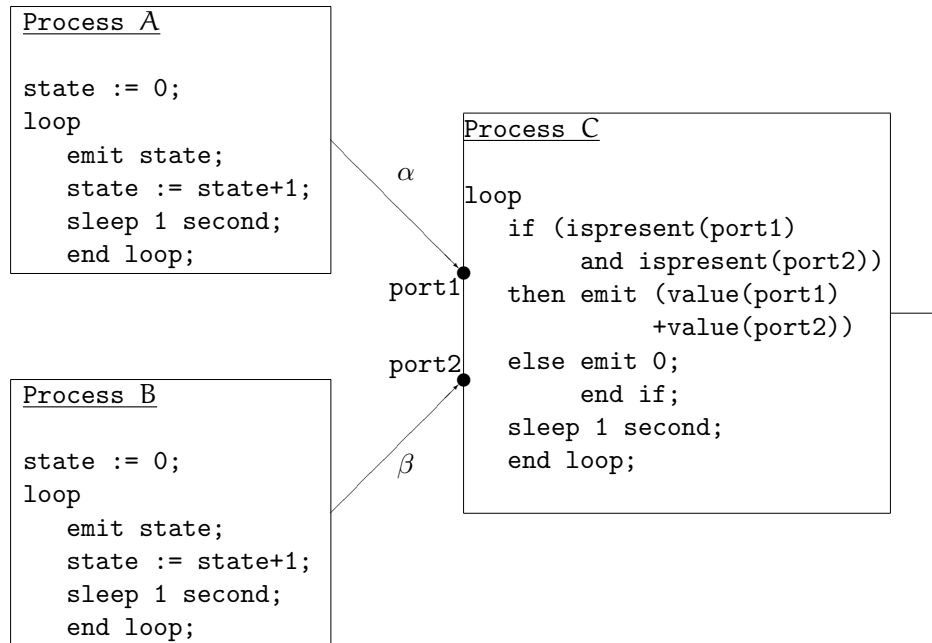
Common assumptions for untimed, synchronous and timed MoC:

- An input channel cannot be probed for the presence of data. The read operation is called a **blocking read**.
- The communication channels are **unbounded FIFO buffers**. The writing to and reading from a channel are entirely decoupled.
- The process network is **determinate** if all processes are determinate, i.e. the connecting network does not introduce nondeterminism.

Non-blocking Read



Non-blocking Read - cont'd



delay(α)=0
 delay(β)=0

delay(α)=0
 delay(β)=1

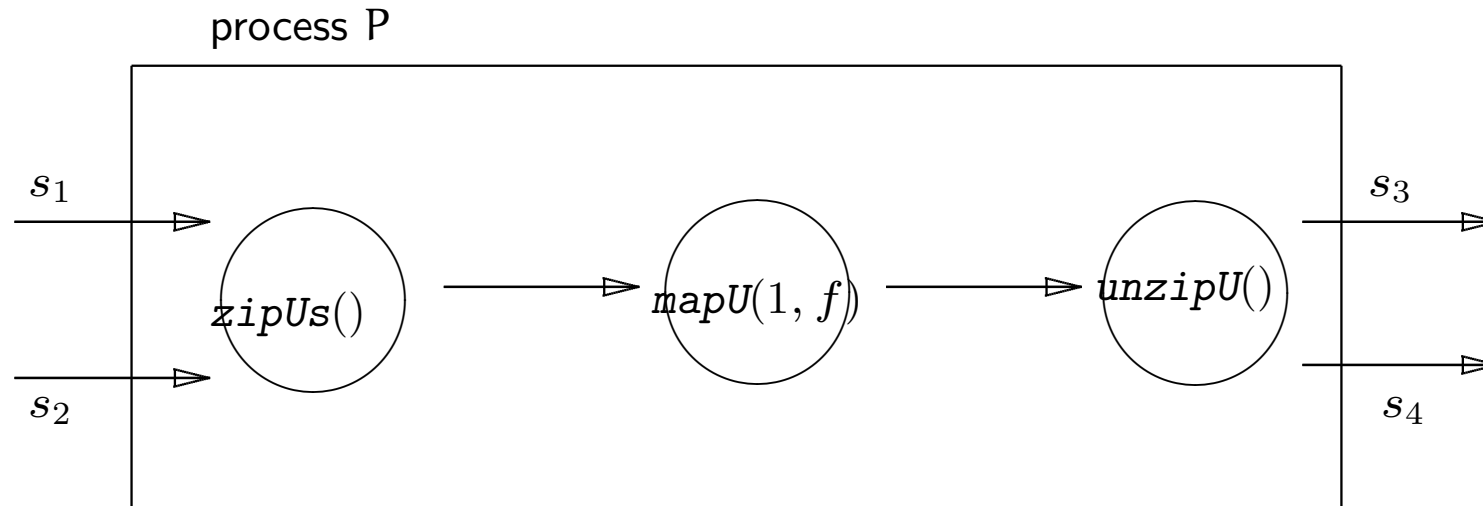
Process ports	Output sequence				
A output	0	1	2	3	4
B output	0	1	2	3	4
C input 1	0	1	2	3	4
C input 2	0	1	2	3	4
C output	0	2	4	6	8
A output	0	1	2	3	4
B output	0	1	2	3	4
C input 1	0	1	2	3	4
C input 2	-	0	1	2	3
C output	0	1	3	5	7



Nondeterministic Channel Delays

- No nondeterminism of functionality if
 - ★ Time is not explicit, and absence of events cannot be sensed,
 - ★ or time is explicitly part of the functionality.
- Deterministic behavior can be realized in a nondeterministic MoC by implementing a blocking read.

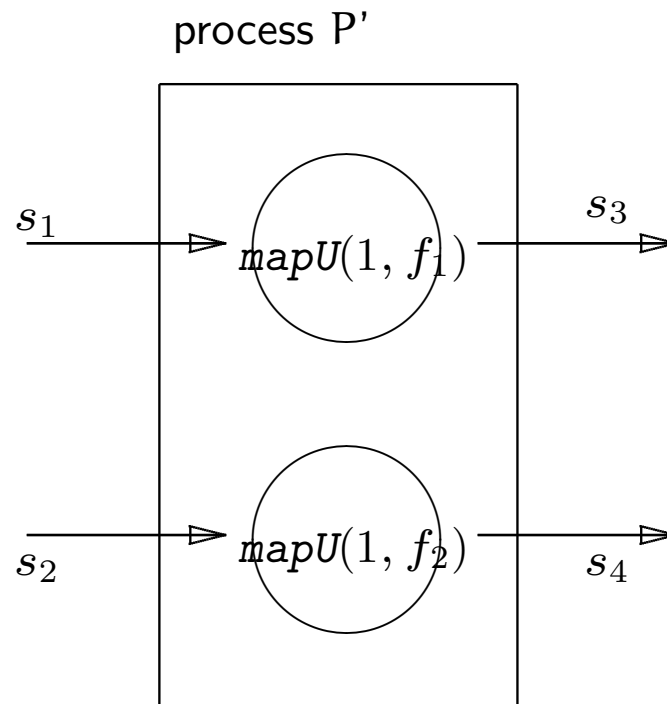
Control Coupling of Independent Data Streams - 1



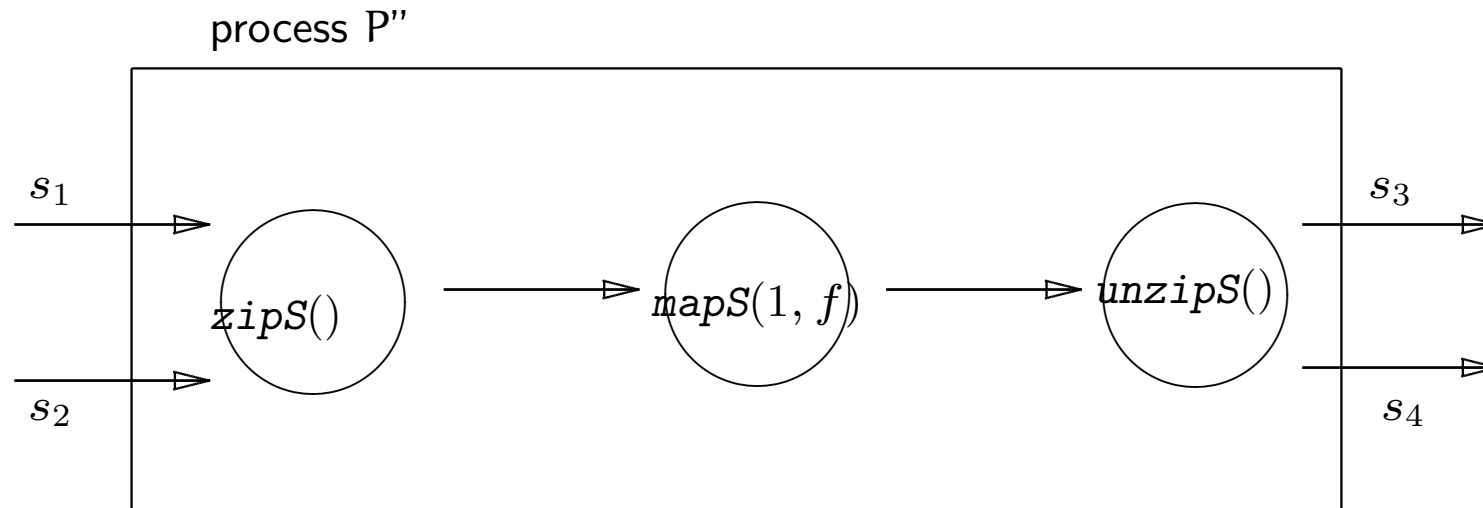
$$f((x, y) = (f_1(x), f_2(y)))$$

Stream $s_1 - s_3$ is independent of $s_2 - s_4$.

Control Coupling of Independent Data Streams - 2



Control Coupling of Independent Data Streams - 3



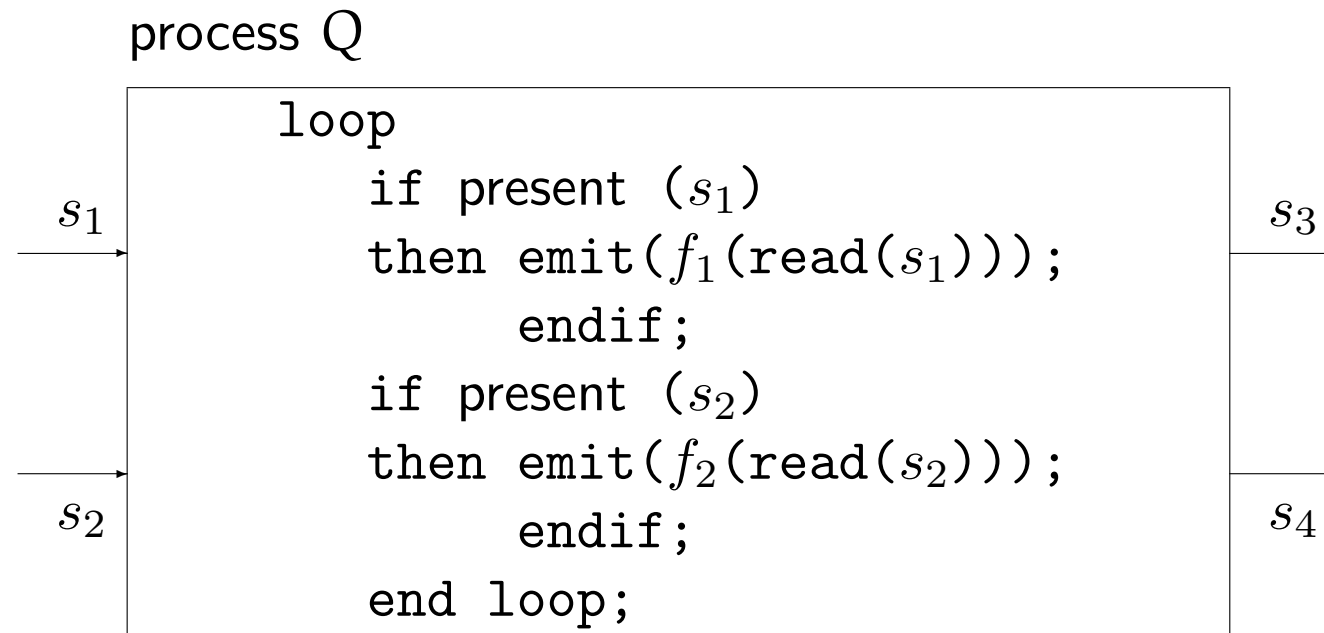
$$f(\perp) = \perp$$

$$f(\perp, y) = (\perp, f_2(y))$$

$$f(x, \perp) = (f_1(x), \perp)$$

$$f(x, y) = (f_1(x), f_2(y))$$

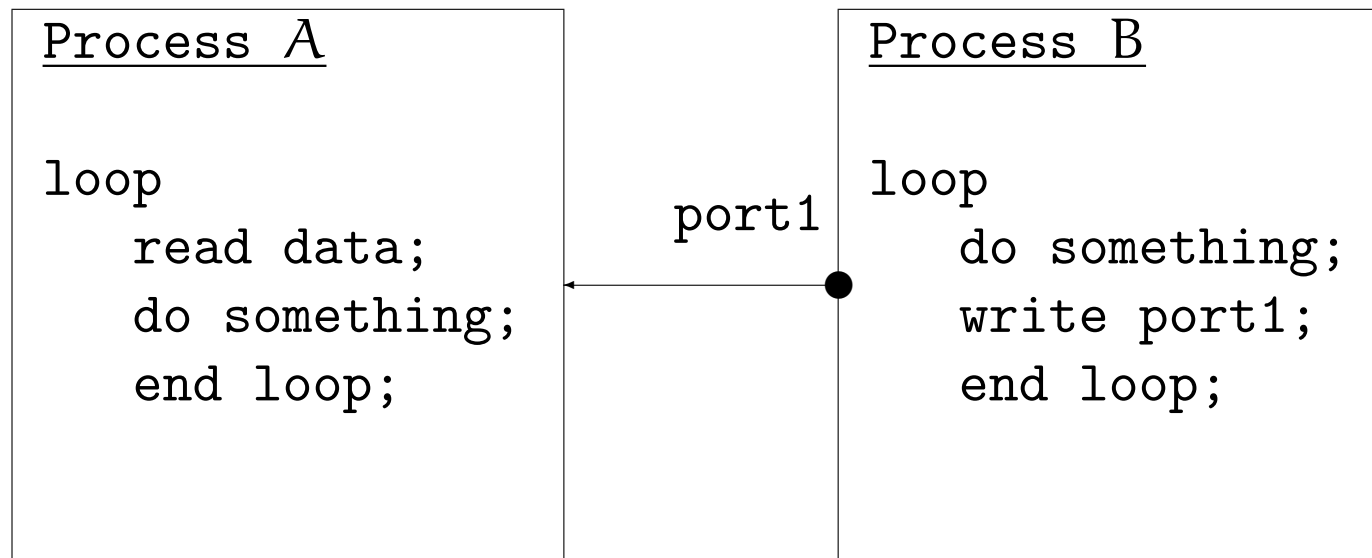
Control Coupling of Independent Data Streams - 4



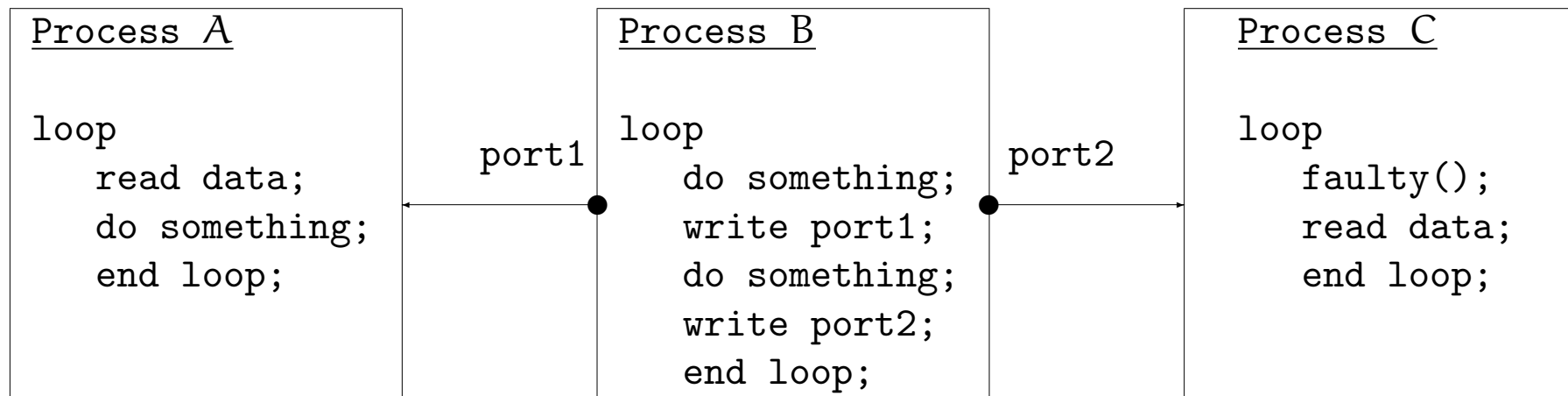
A process with non-blocking read implemented by function present.

Rule: **If s_o does not depend on s_i then the reading of, processing of and waiting for s_i data must not change or block the generation of s_o data.**

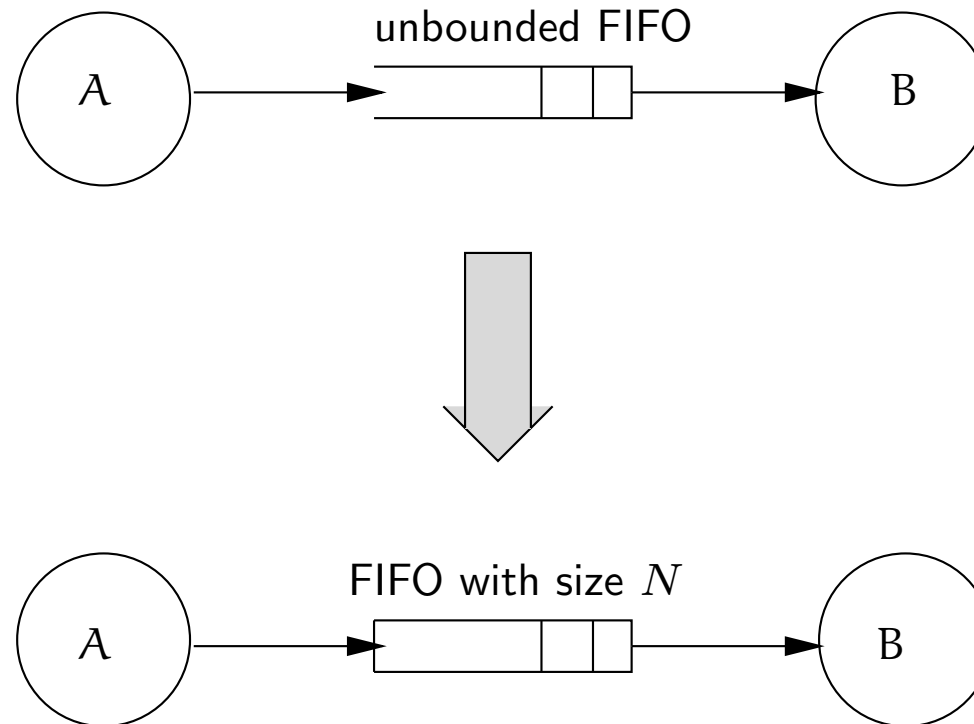
Blocking Read and Blocking Write - 1



Blocking Read and Blocking Write - 2



Oversynchronization FIFO Refinement



Under which condition does the refined system behave identical to the original?

Identity Condition

$\text{ev-time}(p)$... evaluation instances of p .

$\text{emitted}(p, i)$... number of tokens emitted in cycle i .

$\text{consumed}(p, i)$... number of tokens consumed in cycle i .

$$E_{p,t} = \sum_{j=0}^J \text{emitted}(p, j) \quad \text{where} \quad J = \max(k : t_k \leq t)$$

$$C_{p,t} = \sum_{j=0}^J \text{consumed}(p, j) \quad \text{ev-time}(p) = \langle t_0, t_1, t_2, \dots \rangle$$

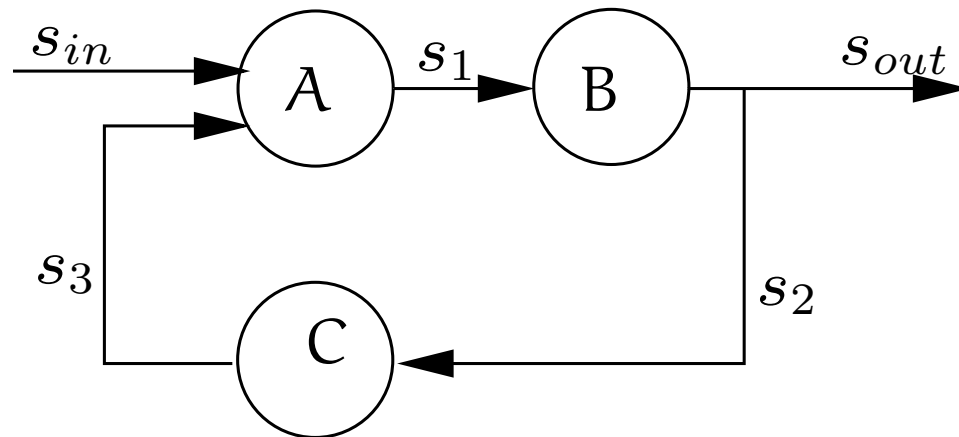
Process A will never be blocked due to a full FIFO of size N if

$$E_{A,t} \geq C_{B,t} \geq E_{A,t} - N \quad \text{for all } t \in \text{ev-time}(A)$$

Effects of Blocking Finite FIFOs

- No feedback loops:
 - ★ All system output signals are eventually the same as in the specification.
 - ★ Output events may be delayed.
 - ★ The shorter the FIFOs the longer the delay may be.
- With feedback loops:
 - ★ The generated output events are identical up to the length of the generated signals.
 - ★ The output signals may be strict prefixes of the corresponding specification output signals.

Finite FIFOs (size 1) with Feedback Loops



$$A = \text{zipUs}(1, 1)$$

$$B = \text{mealyU}(1, f, g, 0)$$

$$\text{where } g(0, (x_1, x_2)) = 1$$

$$g(1, (x_1, x_2)) = 2$$

$$g(n, (x_1, x_2)) = 2 \text{ for } n > 2$$

$$f(0, (x_1, x_2)) = \langle x_1, x_1, x_2 \rangle$$

$$f(1, (x_1, x_2)) = \langle x_1, x_2 \rangle$$

$$f(2, (x_1, x_2)) = \langle x_1 \rangle$$

$$C = \text{initU}(\langle x_0 \rangle)$$

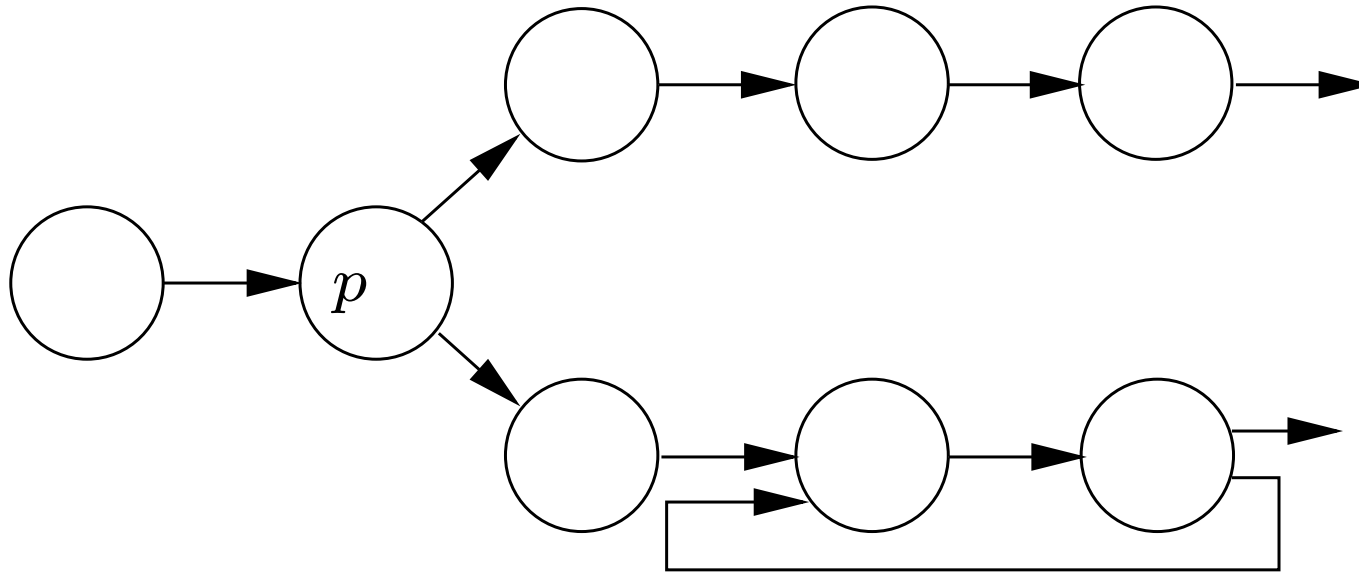
Infinite FIFOs with Feedback Loop

Step	Process	s_{in}	s_1	s_2	s_3	s_{out}
-	-	$\langle x_1, x_2, x_3 \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$
1	C	$\langle x_1, x_2, x_3 \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle \rangle$
2	A	$\langle x_2, x_3 \rangle$	$\langle (x_0, x_1) \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$
3	B	$\langle x_2, x_3 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1 \rangle$
4	C	$\langle x_2, x_3 \rangle$	$\langle \rangle$	$\langle x_0, x_1 \rangle$	$\langle x_0 \rangle$	$\langle x_0, x_0, x_1 \rangle$
5	A	$\langle x_3 \rangle$	$\langle (x_0, x_2) \rangle$	$\langle x_0, x_1 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1 \rangle$
6	B	$\langle x_3 \rangle$	$\langle \rangle$	$\langle x_0, x_1, x_0, x_2 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1, x_0, x_2 \rangle$
7	C	$\langle x_3 \rangle$	$\langle \rangle$	$\langle x_1, x_0, x_2 \rangle$	$\langle x_0 \rangle$	$\langle x_0, x_0, x_1, x_0, x_2 \rangle$
8	A	$\langle \rangle$	$\langle (x_0, x_3) \rangle$	$\langle x_1, x_0, x_2 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1, x_0, x_2 \rangle$
9	B	$\langle \rangle$	$\langle \rangle$	$\langle x_1, x_0, x_2, x_0 \rangle$	$\langle \rangle$	$\langle x_0, x_0, x_1, x_0, x_2, x_0 \rangle$
10	C	$\langle \rangle$	$\langle \rangle$	$\langle x_0, x_2, x_0 \rangle$	$\langle x_1 \rangle$	$\langle x_0, x_0, x_1, x_0, x_2, x_0 \rangle$

Finite FIFOs with Feedback Loop

Step	Process	s_{in}	s_1	s_2	s_3	s_{out}
-	-	$\langle x_1, x_2, x_3 \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$
1	C	$\langle x_1, x_2, x_3 \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle \rangle$
2	A	$\langle x_2, x_3 \rangle$	$\langle (x_0, x_1) \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle \rangle$
3	B blocked	$\langle x_2, x_3 \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle \rangle$	$\langle x_0 \rangle$
4	C	$\langle x_2, x_3 \rangle$	$\langle \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle x_0 \rangle$
5	B resumed; blocked	$\langle x_2, x_3 \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle x_0 \rangle$	$\langle x_0, x_0 \rangle$
6	A	$\langle x_3 \rangle$	$\langle (x_0, x_2) \rangle$	$\langle x_0 \rangle$	$\langle \rangle$	$\langle x_0, x_0 \rangle$
7	C	$\langle x_3 \rangle$	$\langle (x_0, x_2) \rangle$	$\langle \rangle$	$\langle x_0 \rangle$	$\langle x_0, x_0 \rangle$
8	B resumed	$\langle x_3 \rangle$	$\langle (x_0, x_2) \rangle$	$\langle x_1 \rangle$	$\langle x_0 \rangle$	$\langle x_0, x_0, x_1 \rangle$
9	all blocked					

Oversynchronization



A deadlock in the lower branch can block execution of the upper branch.

Summary

- Data dependencies determine the minimum coupling between processes.
- Control dependencies can, sometimes unnecessarily, increase the process coupling.
- Limited and shared resources may further increase process coupling.
- In deterministic process networks:
 - ★ without cyclic dependencies output events may only be delayed due to coupling;
 - ★ with cyclic dependencies deadlock may occur due to coupling.
- In nondeterministic process networks the resulting behaviour may be very different.