Argo – An area-efficient time-division-multiplexed network-on-chip for real-time multicore platforms

Jens Sparsø
jspa@dtu.dk

DTU Compute
Department of Applied Mathematics and Computer Science

Introduction – NoC for real-time

- Hard real-time systems need
  worst case execution time (WCET) guarantees:
  - Execution time of tasks
    - Time predictable processor cores
  - Guaranteed latency and bandwidth of task-to-task communication
    - Time predictable NoC

- Time predictable NoC
  - End-to-end virtual circuits
    - No interference among traffic flows
    - Analysis of individual traffic flows one-by-one
  - Solution:
    - Time division multiplexing (TDM) and static scheduling
    - Rate control and non-blocking routers + network calculus

Introduction – why another NoC

Q: Why yet another NoC after 10-15 years of NoC-research?

Q: Why yet another TDM-based NoC?

- TU/e: Æthereal, Aelite, dAelite
- KTH: Nostrum
- TU Vienna: TTNoC

A: Many NoC designs, both BE og GS, have very large implementations.

- Buffers and flow control
- Size of router + NI often approach size of processor core.

[As I see it] a result of :

- Focus on providing solutions
- Focus on layering and encapsulation

Introduction – Argo

- TDM scheduling requires a common global notion of time.
- Chip technology calls for Globally-Asynchronous Locally-Synchronous (GALS) or Mesochronous timing architecture

- ARGO combines TDM and GALS/Mesochronous
  - Individually clocked processors (GALS)
  - Mesochronous network interfaces
  - Asynchronous routers

- ARGO avoids all buffering and (run time) flow control.
  TDM-mechanism transfer data end-to-end, i.e., SPM to SPM.
  (not just NI to NI as in other NoCs)
  - A novel NI microarchitecture with a very small HW-implementation

Acknowledgements

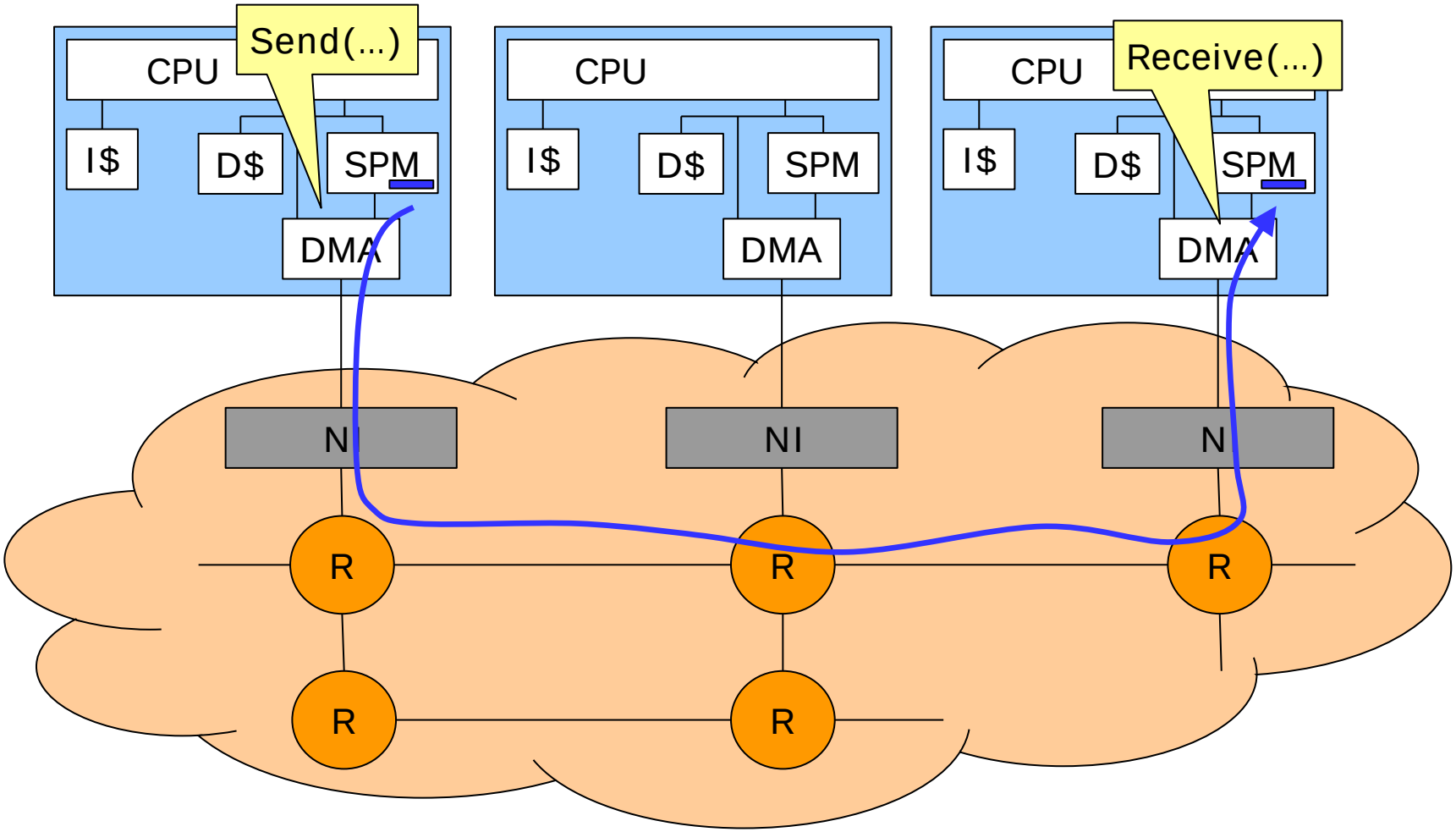
- Presentation is based on joint work with:
  - Associate Professor Martin Schoeberl
  - Wolfgang Puffitsch (Post Doc. 2013-2016)
  - Evangelia Kasapaki (PhD student 2011-2015)
  - Rasmus Bo Sørensen (PhD student 2013-2016)

- Work has been partly funded by:
  - European Union's 7th Framework Programme: Time-predictable Multi-Core Architecture for Embedded Systems (**T-CREST**) under grant agreement no. 288008. [2011-14]
  - "Hard Real-Time Embedded Multiprocessor Platform - **RTEMP**" funded by the Danish Research Council for Technology and Production Sciences under contract no. 12-127600. [2013-2016]

Outline

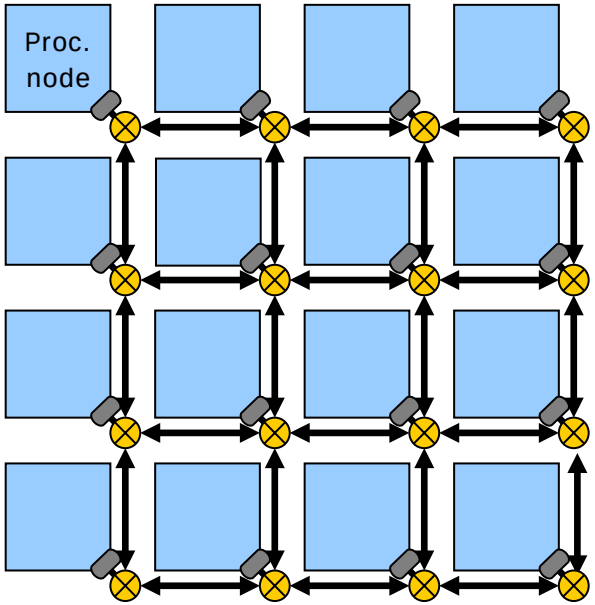
1. Introduction
2. Background
   - The T-CREST multi-core platform
   - Message passing, TDM and static scheduling
3. Architecture and implementation of Argo (globally synchronous)
   - Router
   - NI microarchitecture
   - Results (area)
4. Timing organization of Argo
   - *Individually* clocked processor cores
   - *Mesochronous* NIs
   - Network of *asynchronous* routers
5. Analysis of (clock and reset) skew tolerance
6. Generating schedules
7. Conclusion

Meassage passing using DMAs + virtual circuits

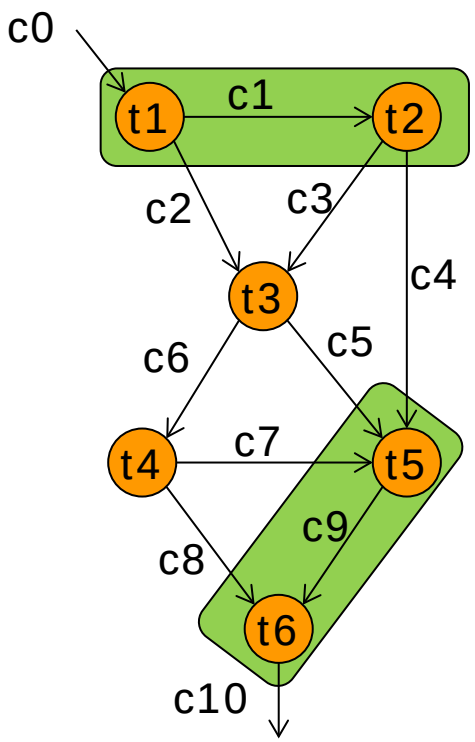


The T-CREST multicore platform

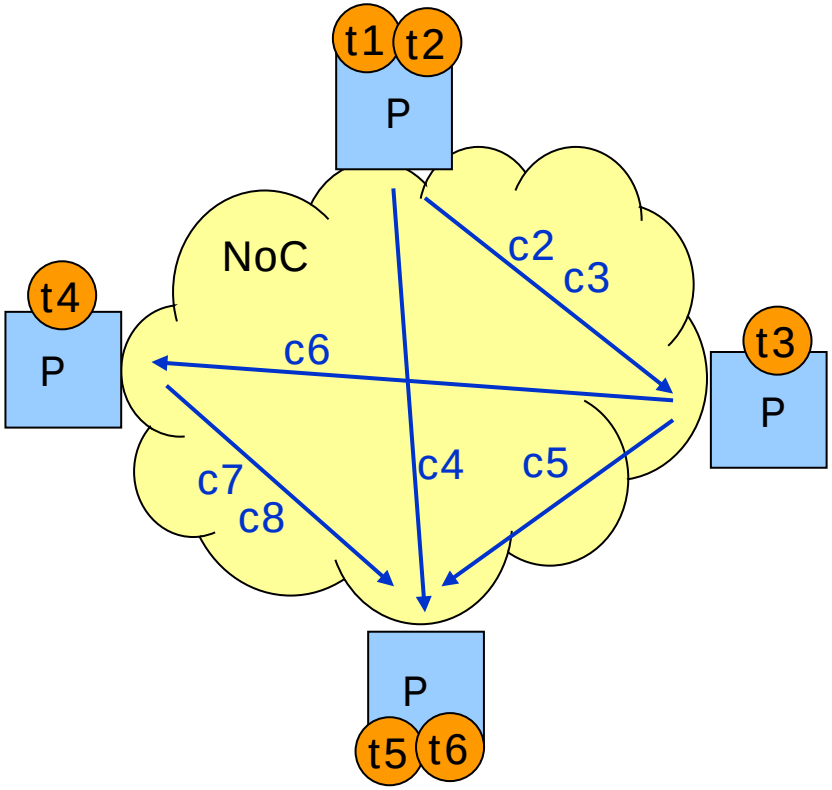
- All components are designed to be time-predictable
- PATMOS processors
  - Dual issue RISC
  - Special caches (method, stack, …)
  - Private scratch pad memories (SPM)
- Argo NoC supporting message passing
  - TDM and static scheduling
  - Supports GALS timing organization
- Memory tree NoC
  - All processors towards one memory
  - TDM and static scheduling



Communicating tasks, communicating processors, and scheduling of packets.

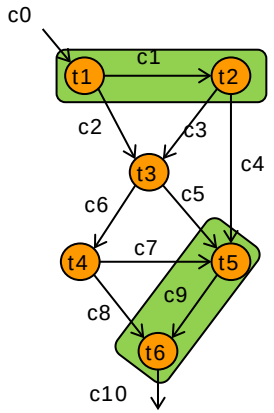


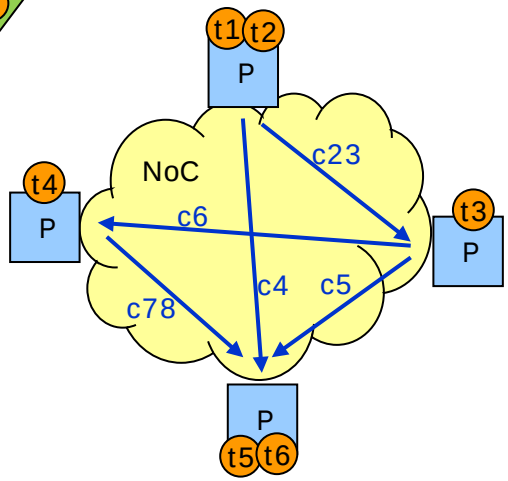
Task graph

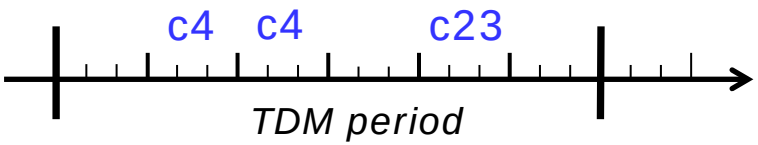


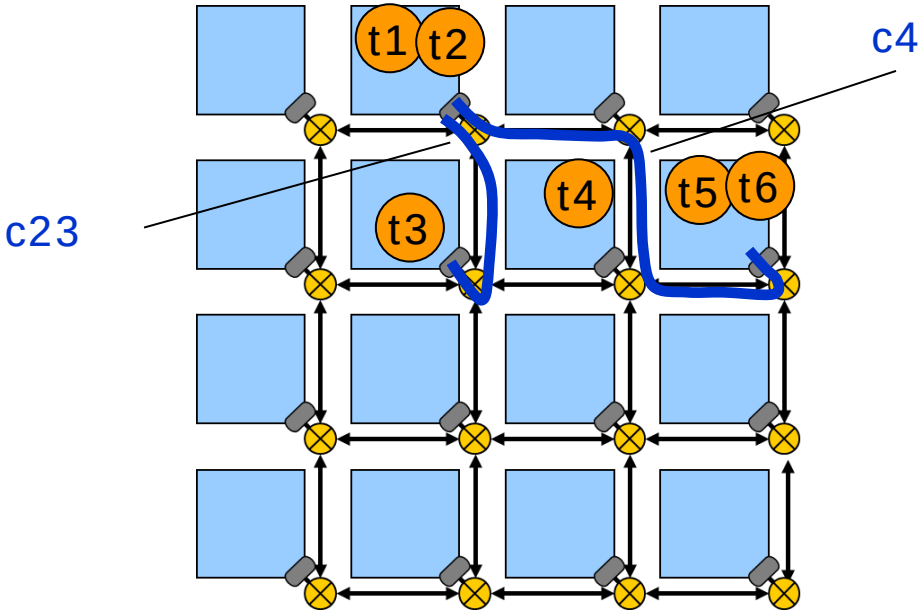
Core communication graph

Communicating tasks, communicating processors, and routing of packets in the NoC

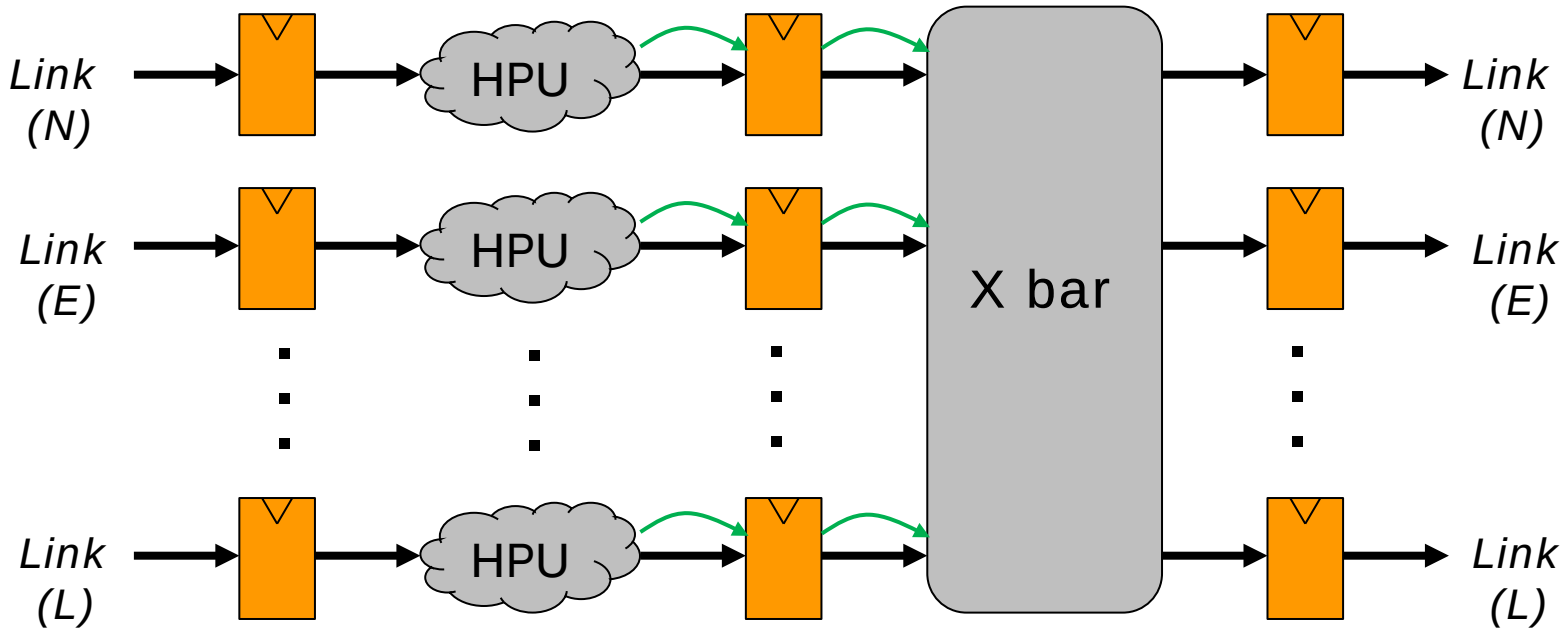




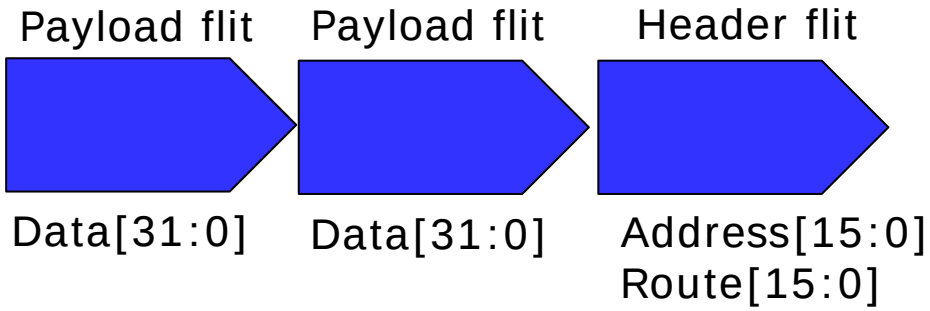




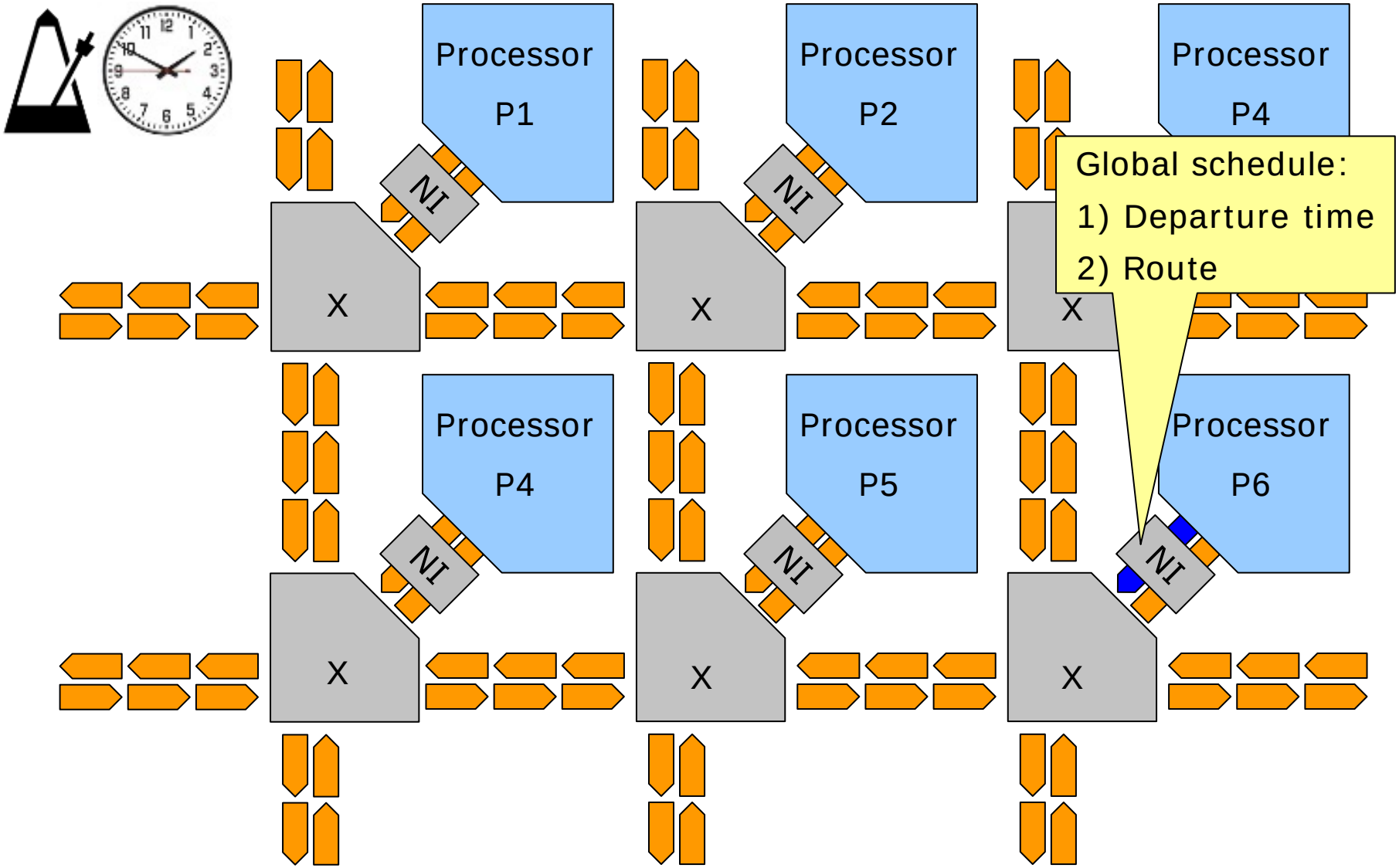
Synchronous router for source-routed TDM-based NoC



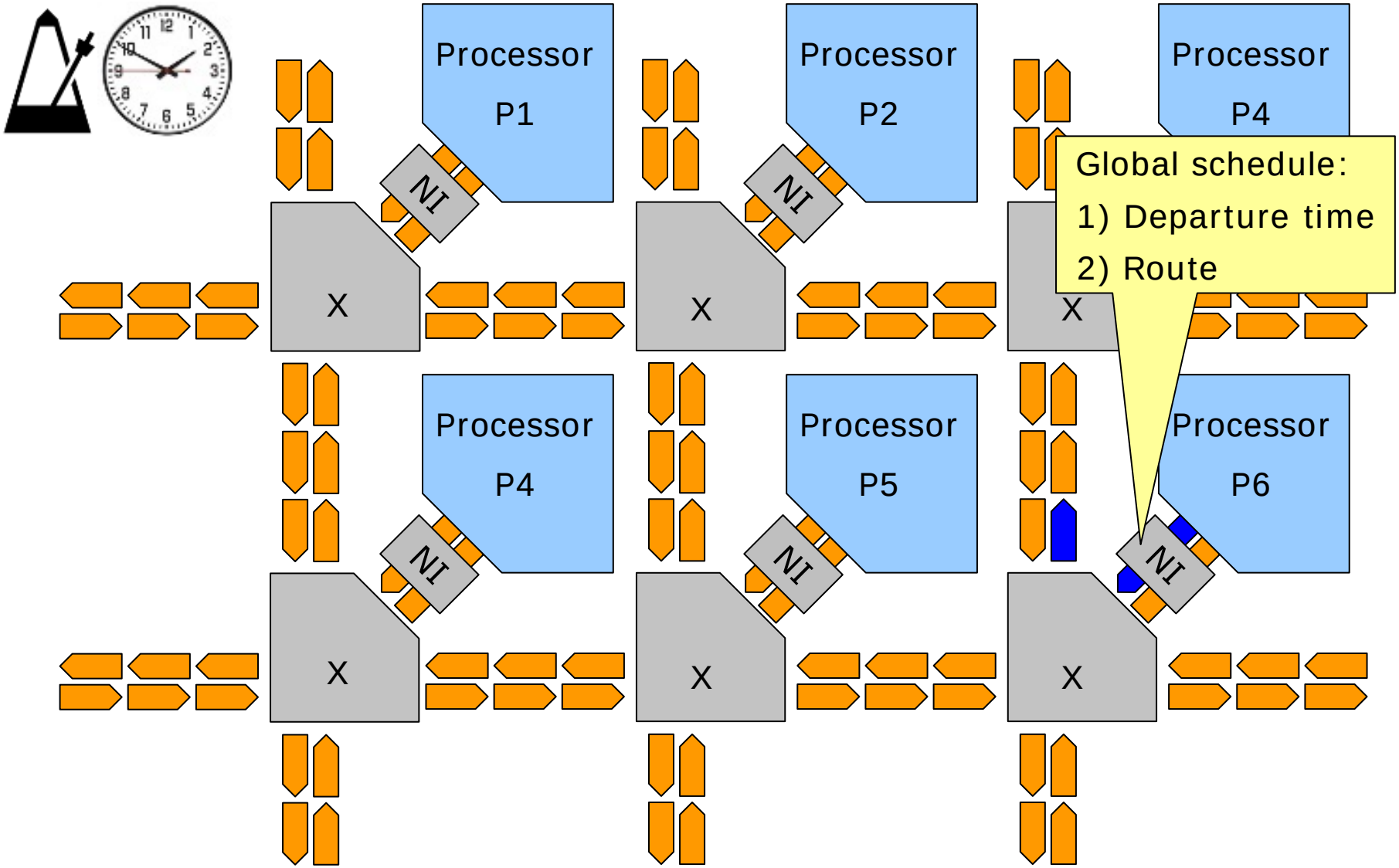
Packet:



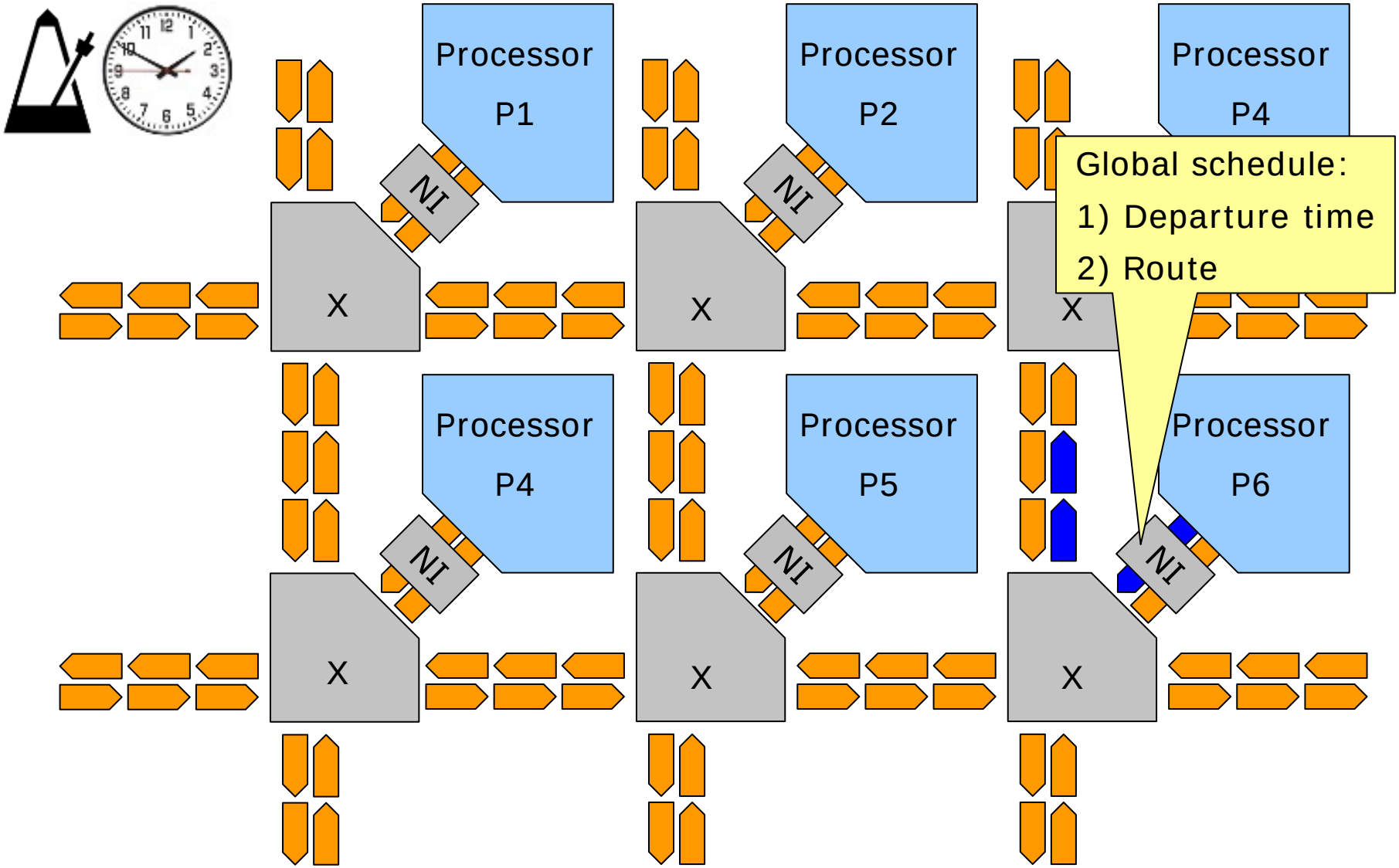
Packet ≈


Processor
P1
Processor
P2
Processor
P4
Processor
P4
Processor
P5
Processor
P6
NI
X
Global schedule:
1) Departure time
2) Route

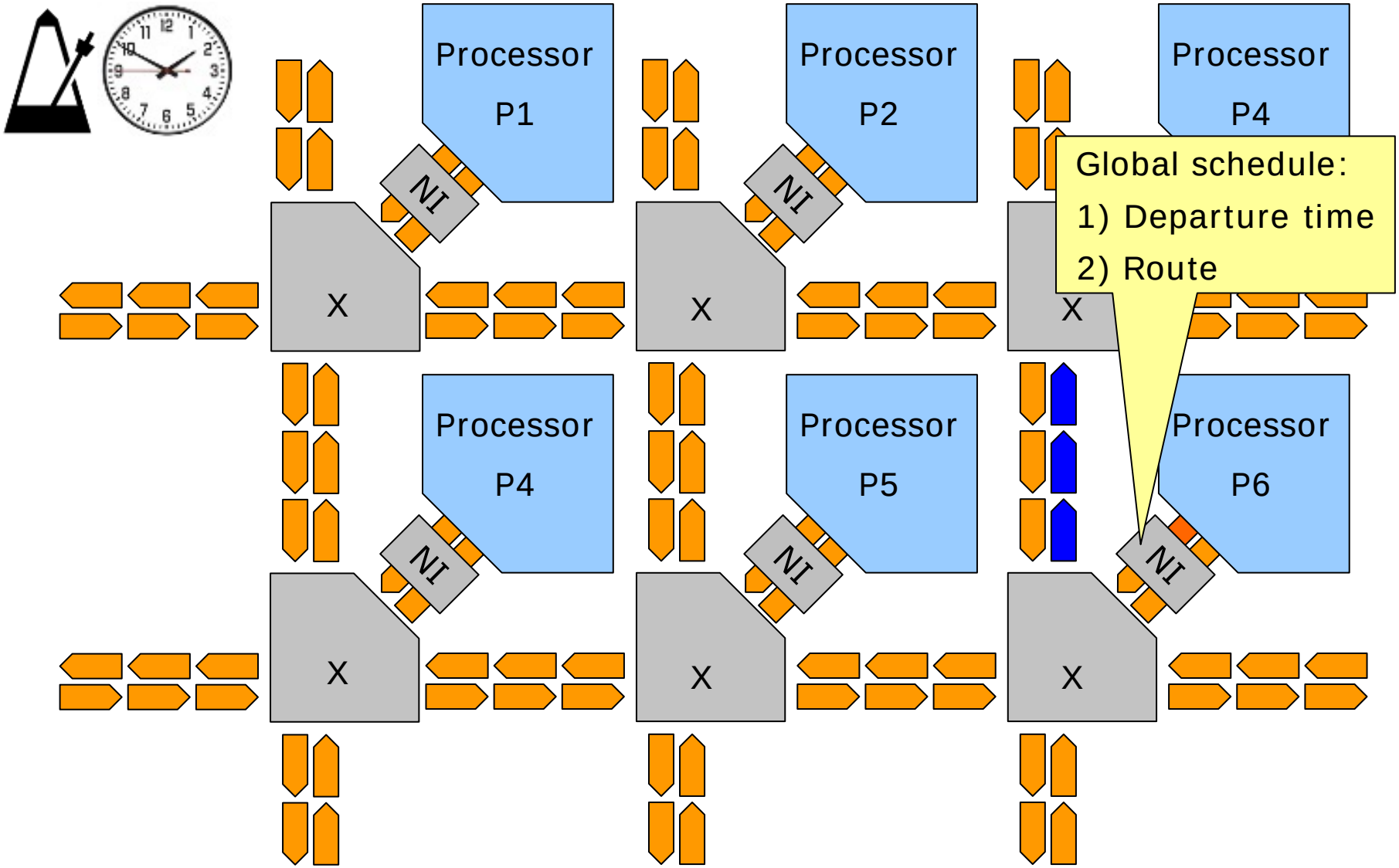
Packet ≈


Processor
P1
Processor
P2
Processor
P4
Processor
P4
Processor
P5
Processor
P6
NI
X
Global schedule:
1) Departure time
2) Route

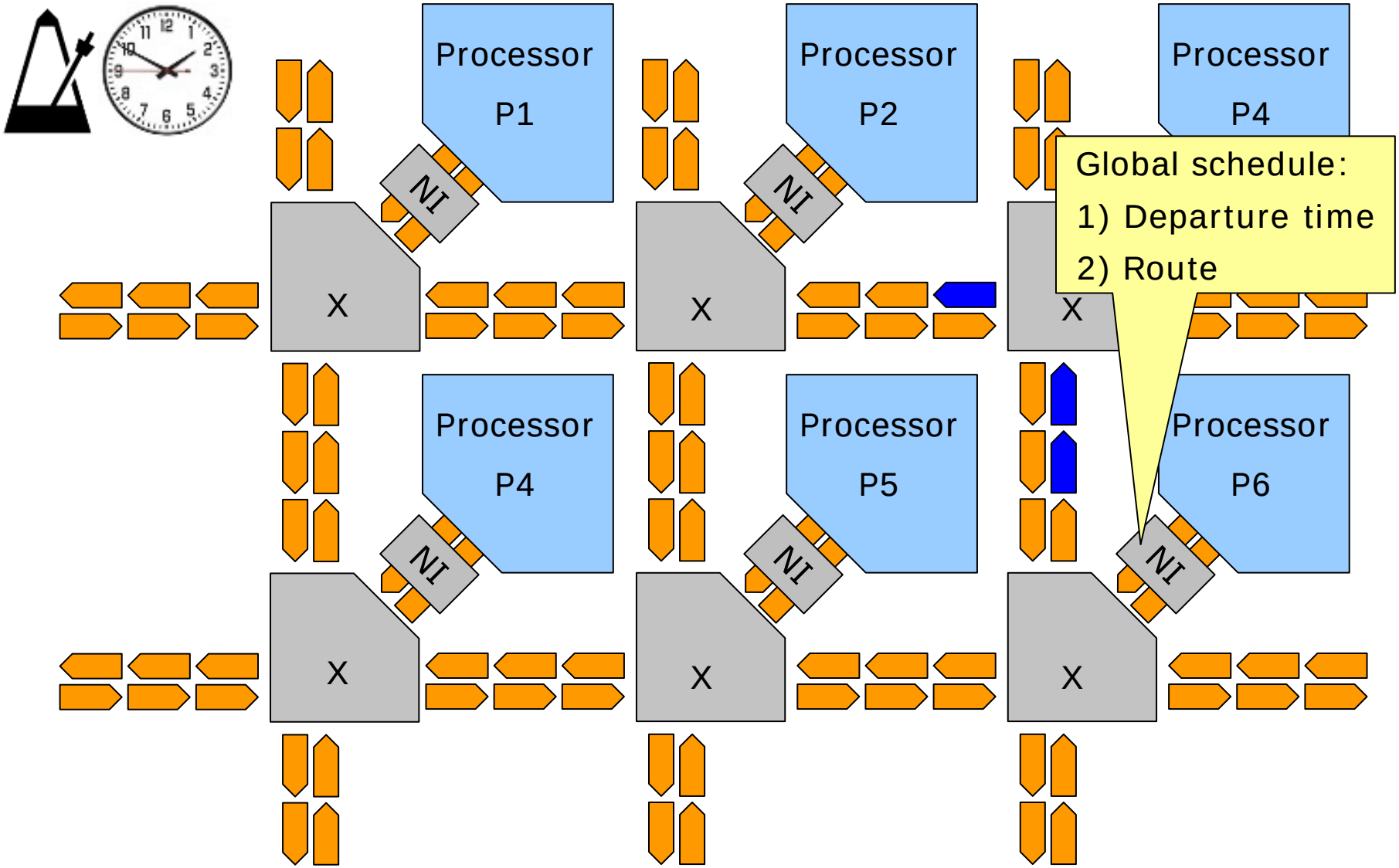
Packet ≈


Processor
P1
Processor
P2
Processor
P4
Processor
P4
Processor
P5
Processor
P6
NI
X
Global schedule:
1) Departure time
2) Route

Packet ≈

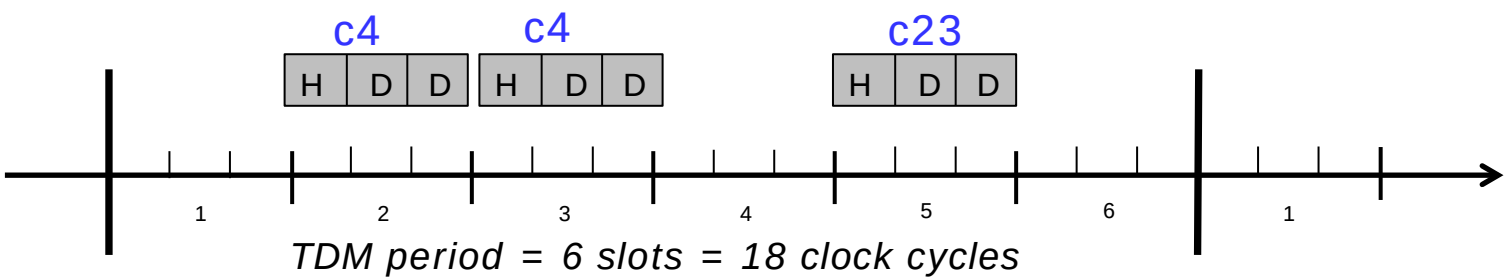

Processor
P1
Processor
P2
Processor
P4
Processor
P4
Processor
P5
Processor
P6
NI
X
Global schedule:
1) Departure time
2) Route

Packet ≈

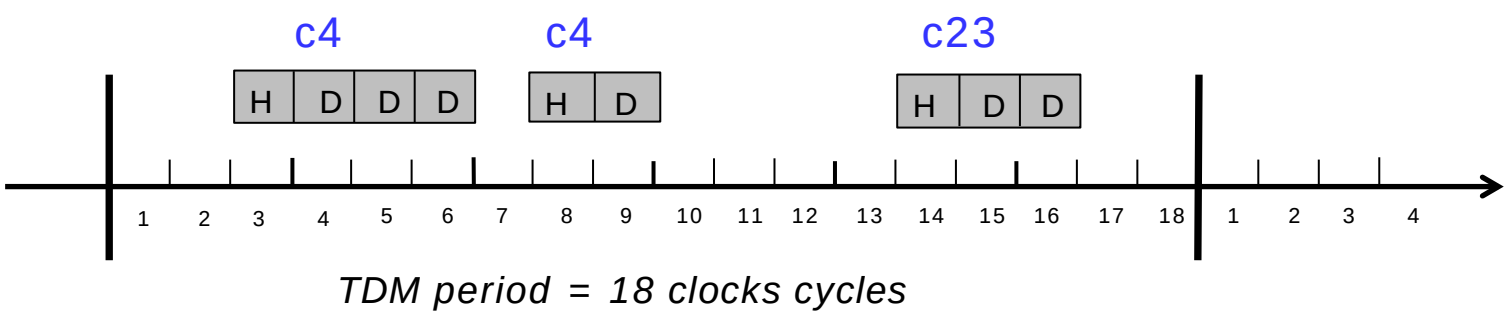

Processor
P1
Processor
P2
Processor
P4
Processor
P4
Processor
P5
Processor
P6
NI
X
Global schedule:
1) Departure time
2) Route

TDM schedule + granularity

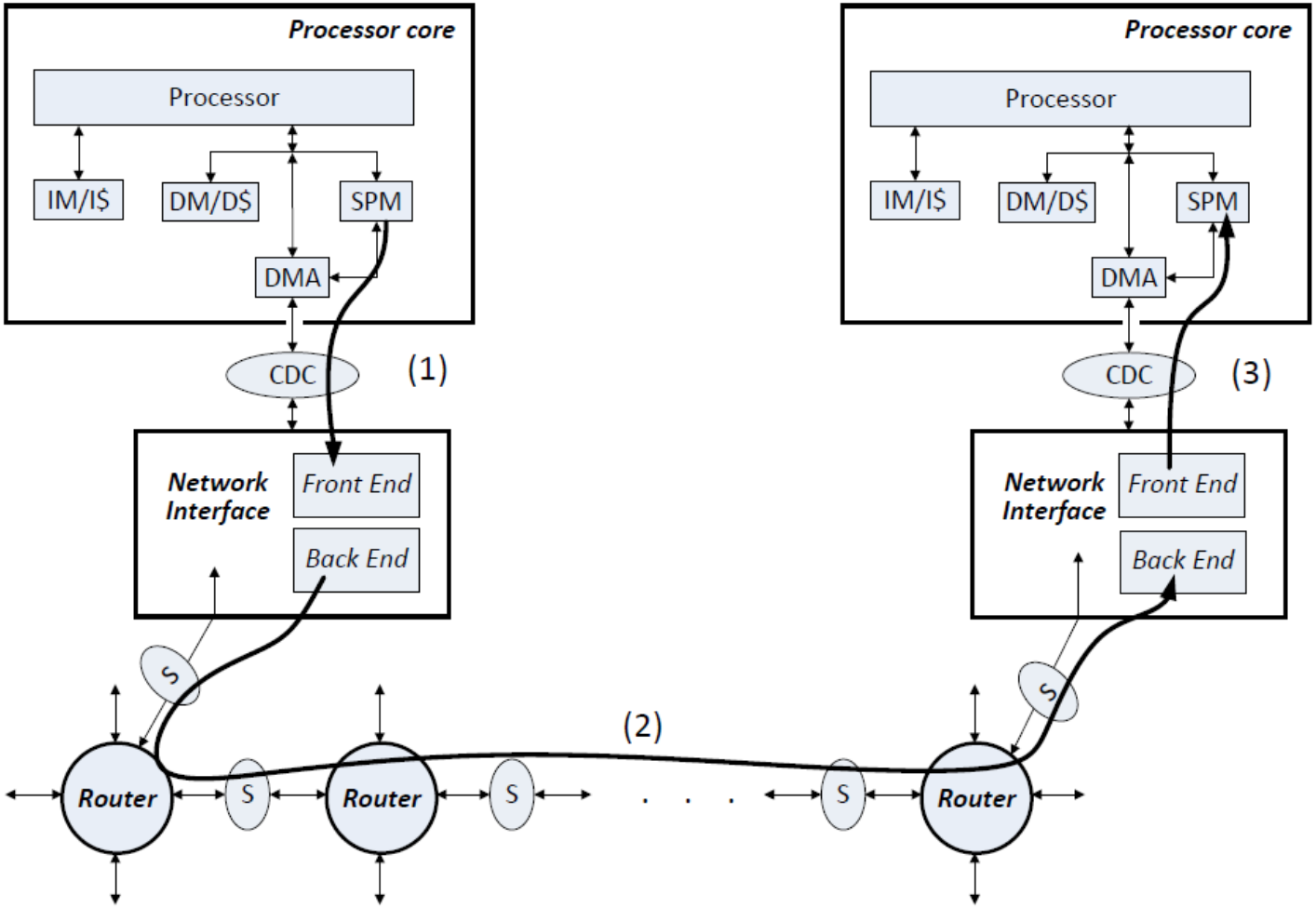
- 3-flit package and 3-stage pipelined router:
  (TDM slot is 3 clock cycles)



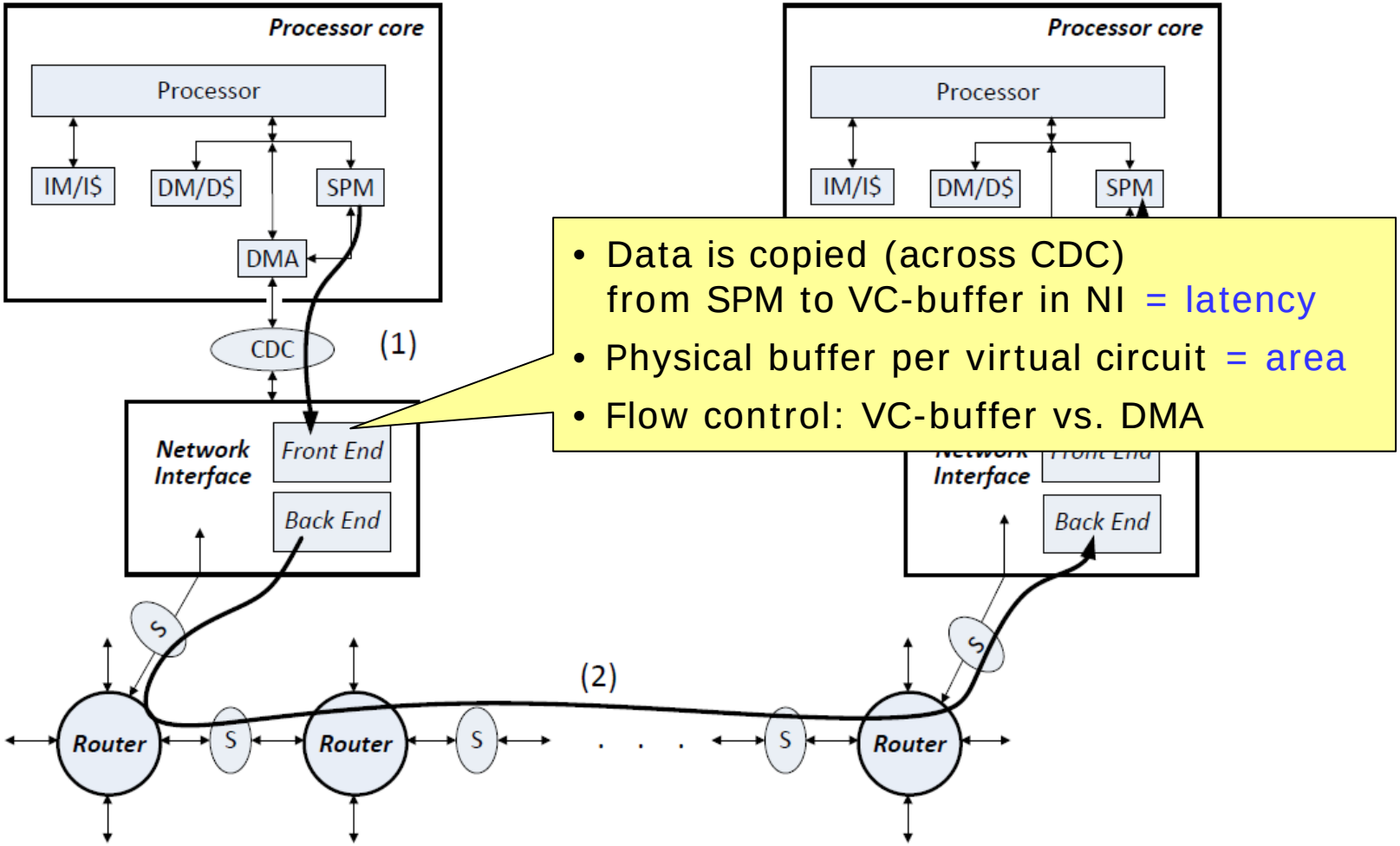
- Variable length packets and arbitrary pipeline depth of router
  (TDM slot is 1 clock cycle)



Traditional design: DMAs in processor nodes

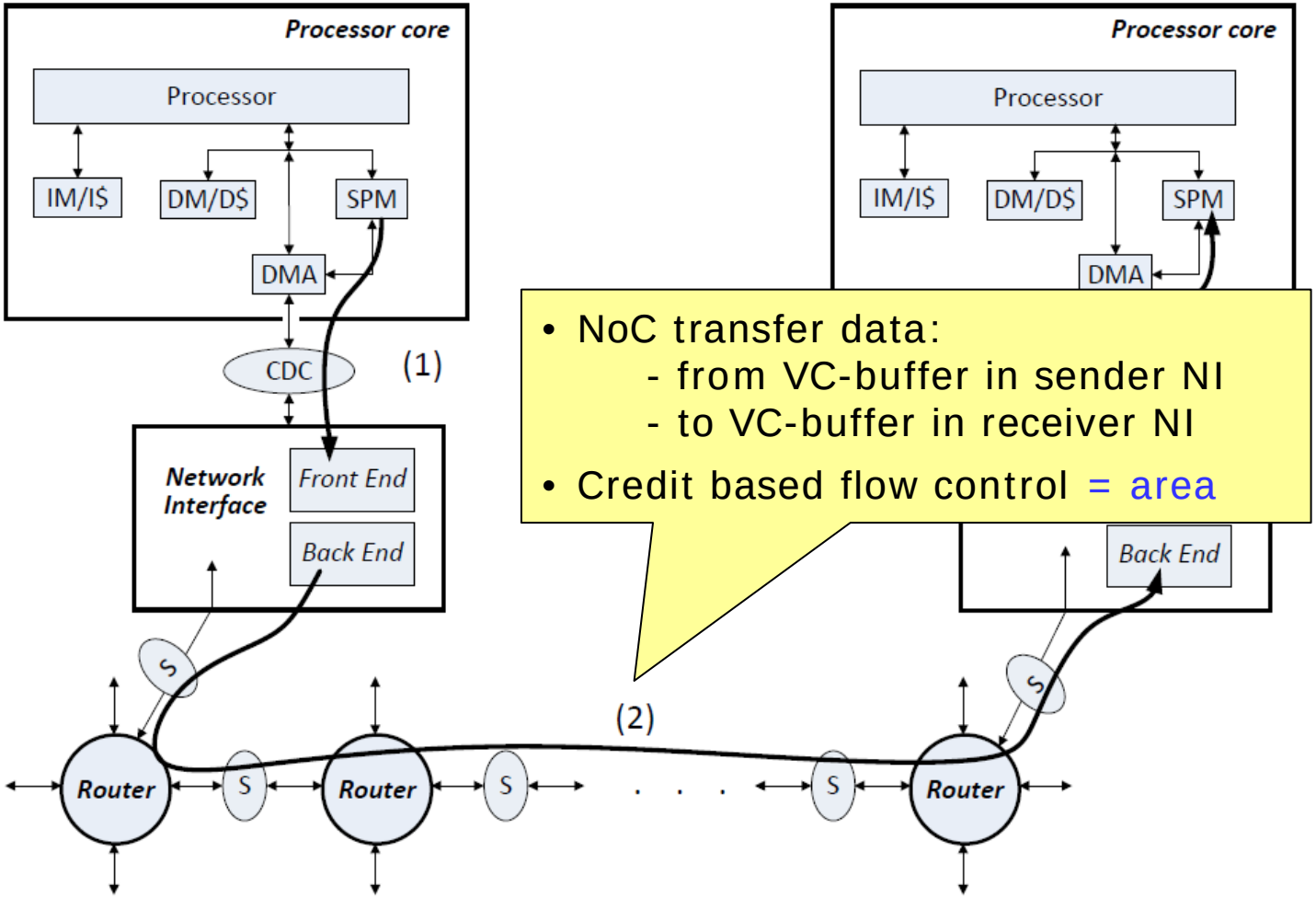


Traditional design: DMAs in processor nodes



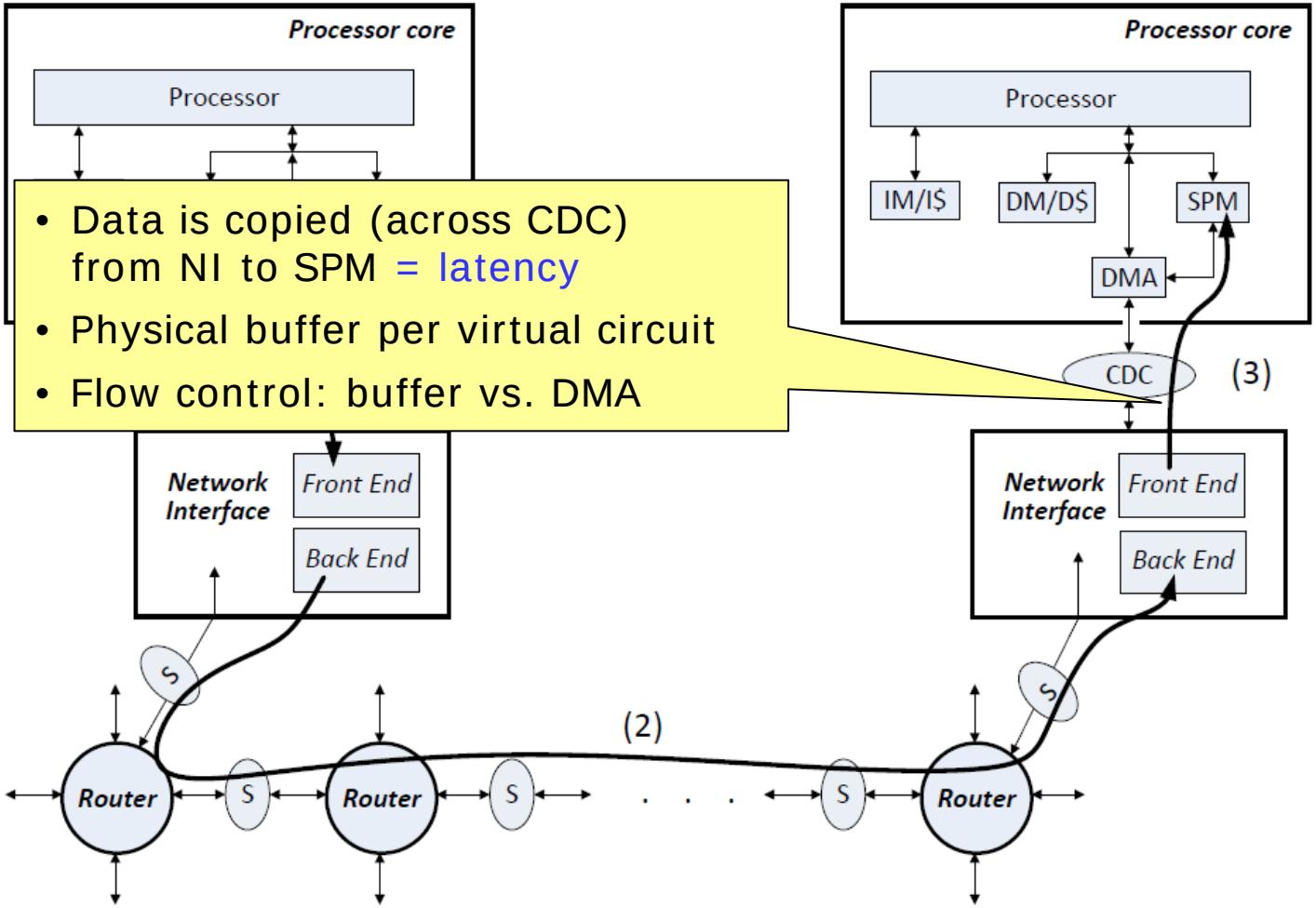
- Data is copied (across CDC) from SPM to VC-buffer in NI = latency
- Physical buffer per virtual circuit = area
- Flow control: VC-buffer vs. DMA

Traditional design: DMAs in processor nodes



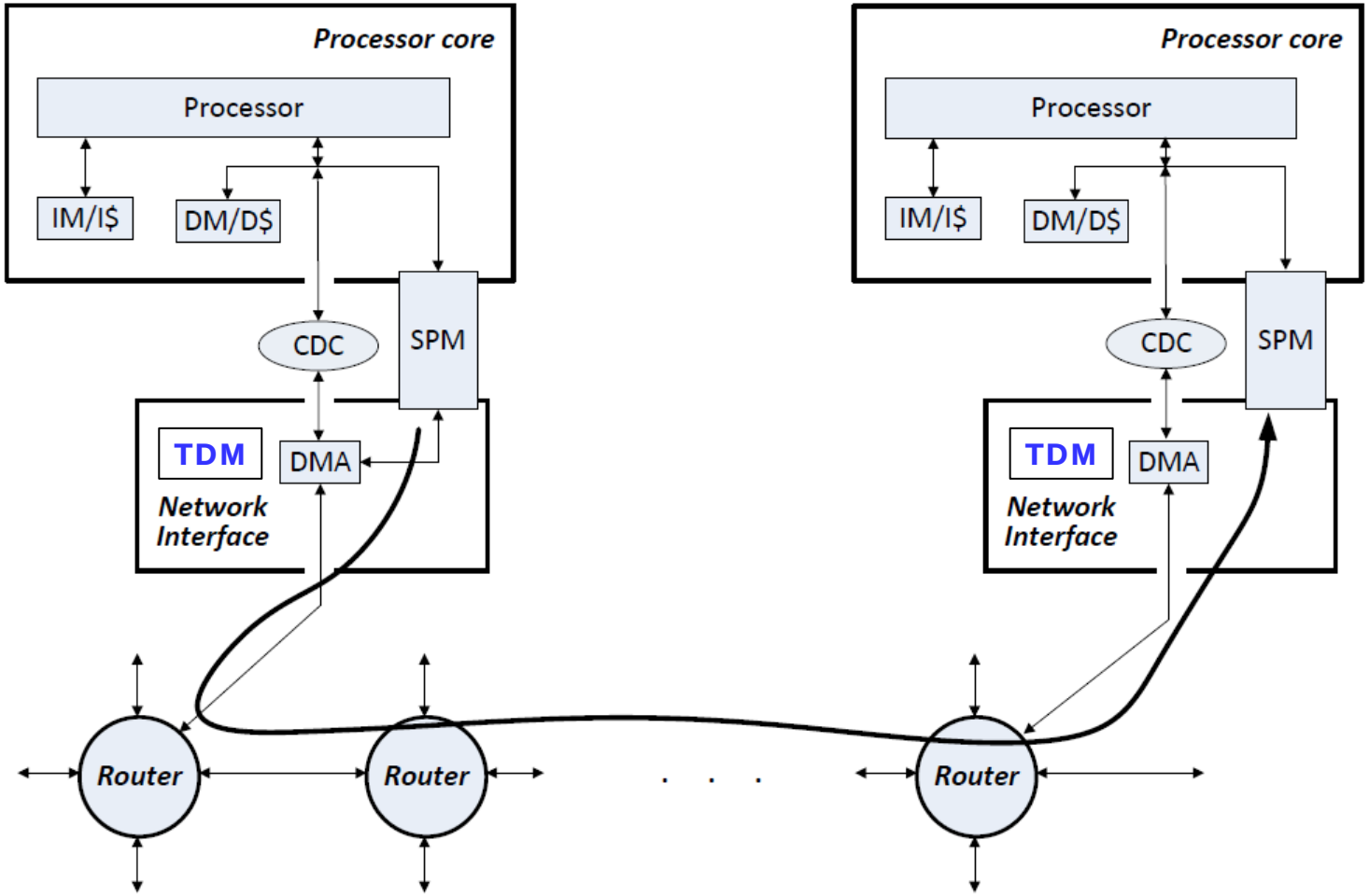
- NoC transfer data:
  - from VC-buffer in sender NI
  - to VC-buffer in receiver NI
- Credit based flow control = area

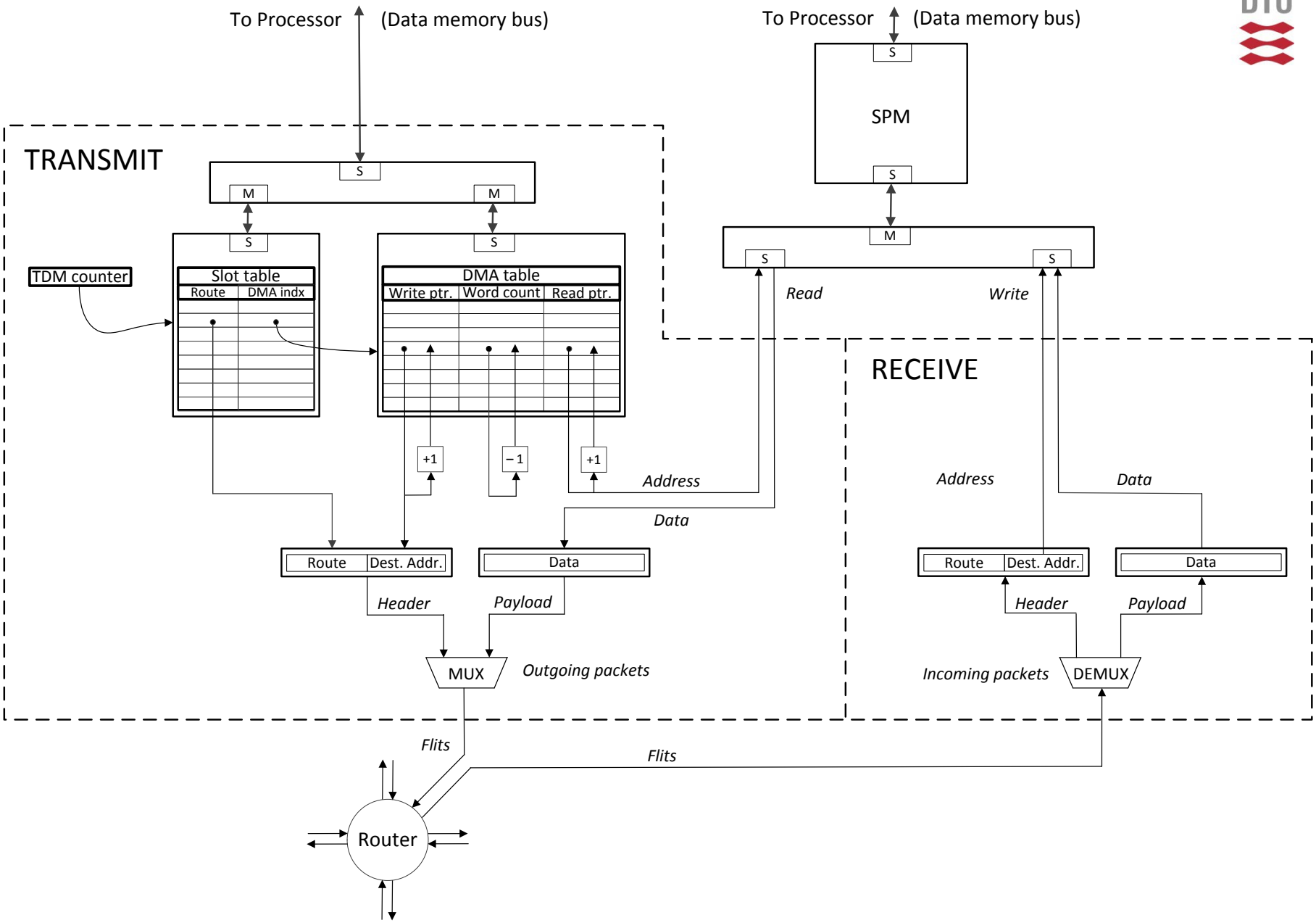
Traditional design: DMAs in processor nodes



- Data is copied (across CDC) from NI to SPM = latency
- Physical buffer per virtual circuit
- Flow control: buffer vs. DMA

New design: DMAs in network interfaces




To Processor
(Data memory bus)
To Processor
(Data memory bus)
TRANSMIT
S
M
M
S
S
TDM counter
Slot table
Route
DMA indx
DMA table
Write ptr.
Word count
Read ptr.
+1
– 1
+1
Address
Data
Route
Dest. Addr.
Data
Header
Payload
MUX
Outgoing packets
S
SPM
S
M
S
S
Read
Write
RECEIVE
Address
Data
Route
Dest. Addr.
Data
Header
Payload
Incoming packets
DEMUX
Flits
Flits
Router

NI synthesized for Altera EP2C70 FPGA

TDM period = 3 clock cycles.

NI design size		NI Logic		Slot and DMA tables		
TDM Period	DMAs	LUTs	FFs	LUTs	FFs	BRAM bits
16	4	326	162	237	374	-
	8	337	162	450	647	-
	16	341	162	116	88	1024
32	4	326	163	286	422	-
	8	339	163	378	579	128
	32	346	163	34	3	2240
64	4	328	164	175	323	192
	8	340	164	378	579	256
	64	351	164	34	3	4544

ASIC Results for a 16-node bi-torus NoC

- 4 x 4 bi-torus (16 NIs and 16 routers)
- All-to-all schedule. TDM-period = 23 slots = 69 clock cycles
- Total of 16 x 15 = 240 virtual circuits
- 65 nm CMOS
- 1.5 x 1.5 mm^2 tile size

	Routers	NIs	Links	FIFOs	total
Cell area					
total (μm^2)	127446	537397	43289	12613	720745
per node (μm^2)	7965	33587	2706	788	45047
relative (%)	17.68	74.56	6.01	1.75	100
Energy					
total (pJ/cyc)	96.40	430.80	363.20	13.41	903.81
per node (pJ/cyc)	6.03	26.93	22.70	0.84	56.49
relative (%)	10.67	47.66	40.19	1.48	100

ASIC Results for a 16-node bi-torus NoC

- 4 x 4 bi-torus (16 NIs and 16 routers)
- All-to-all schedule. TDM-period = 23 slots = 69 clock cycles
- Total of 16 x 15 = 240 virtual circuits
- 65 nm CMOS
- 1.5 x 1.5 mm^2 tile size

How big is your NoC?

	Routers	NIs	Links	FIFOs	total
Cell area					
total (μm^2)	127446	537397	43289	12613	720745
per node (μm^2)	7965	33587	2706	788	45047
relative (%)	17.68	74.56	6.01	1.75	100
Energy					
total (pJ/cyc)	96.40	430.80	363.20	13.41	903.81
per node (pJ/cyc)	6.03	26.93	22.70	0.84	56.49
relative (%)	10.67	47.66	40.19	1.48	100

Results summary

- Relative area:

	Argo	Other TDM NoC
Router	1	1 (synchronous) 2 (mesochronous)
NI	1	2-4

- Other routers and NI's ? / easily 10+

Outline

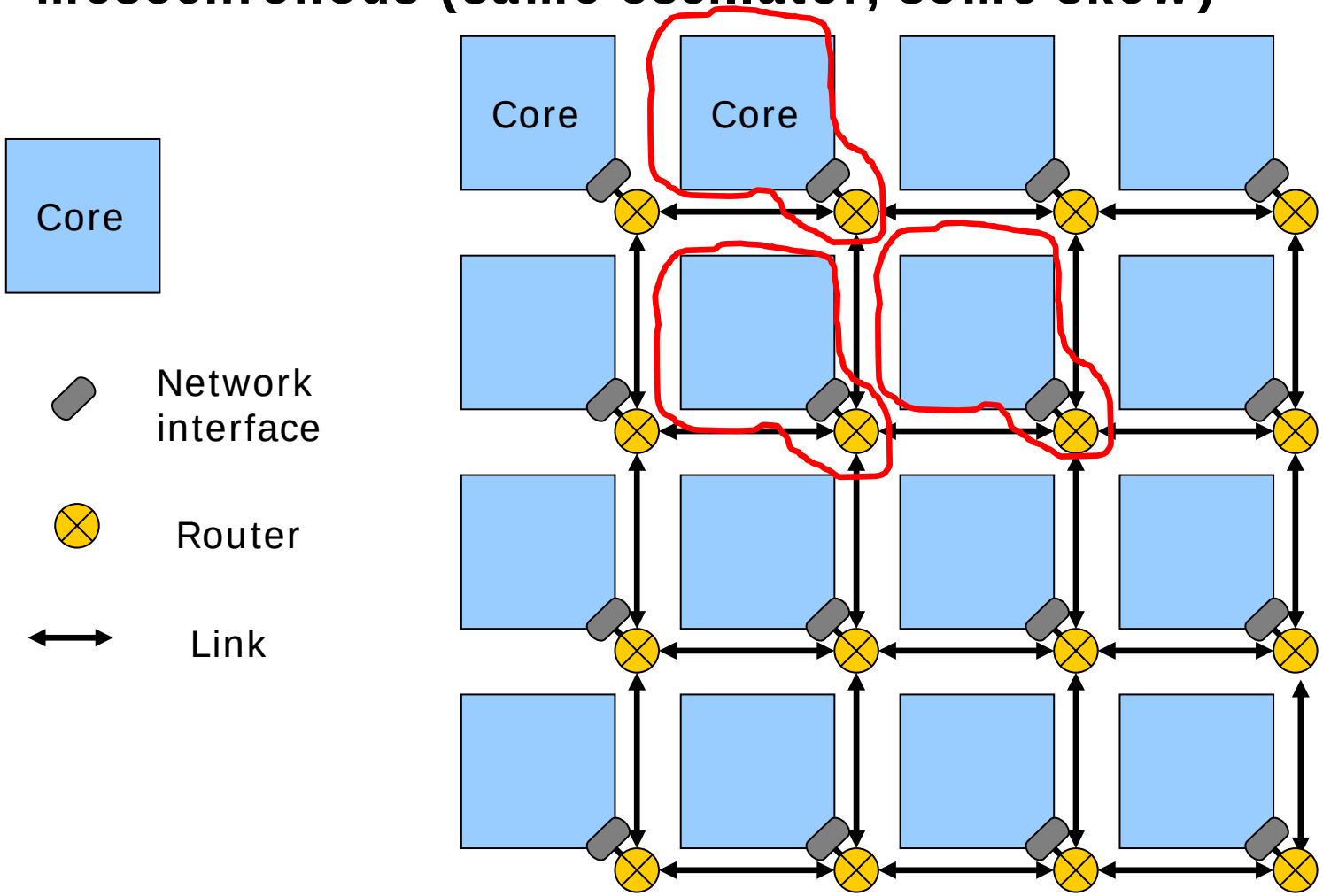
1. Introduction
2. Background
   - The T-CREST multi-core platform
   - Message passing, TDM and static scheduling
3. Architecture and implementation of Argo (globally synchronous)
   - Router
   - NI microarchitecture
   - Results (area)

---

4. Timing organization of Argo
   - *Individually* clocked processor cores
   - *Mesochronous* NIs
   - Network of *asynchronous* routers
5. Analysis of (clock and reset) skew tolerance
6. Generating schedules
7. Conclusion

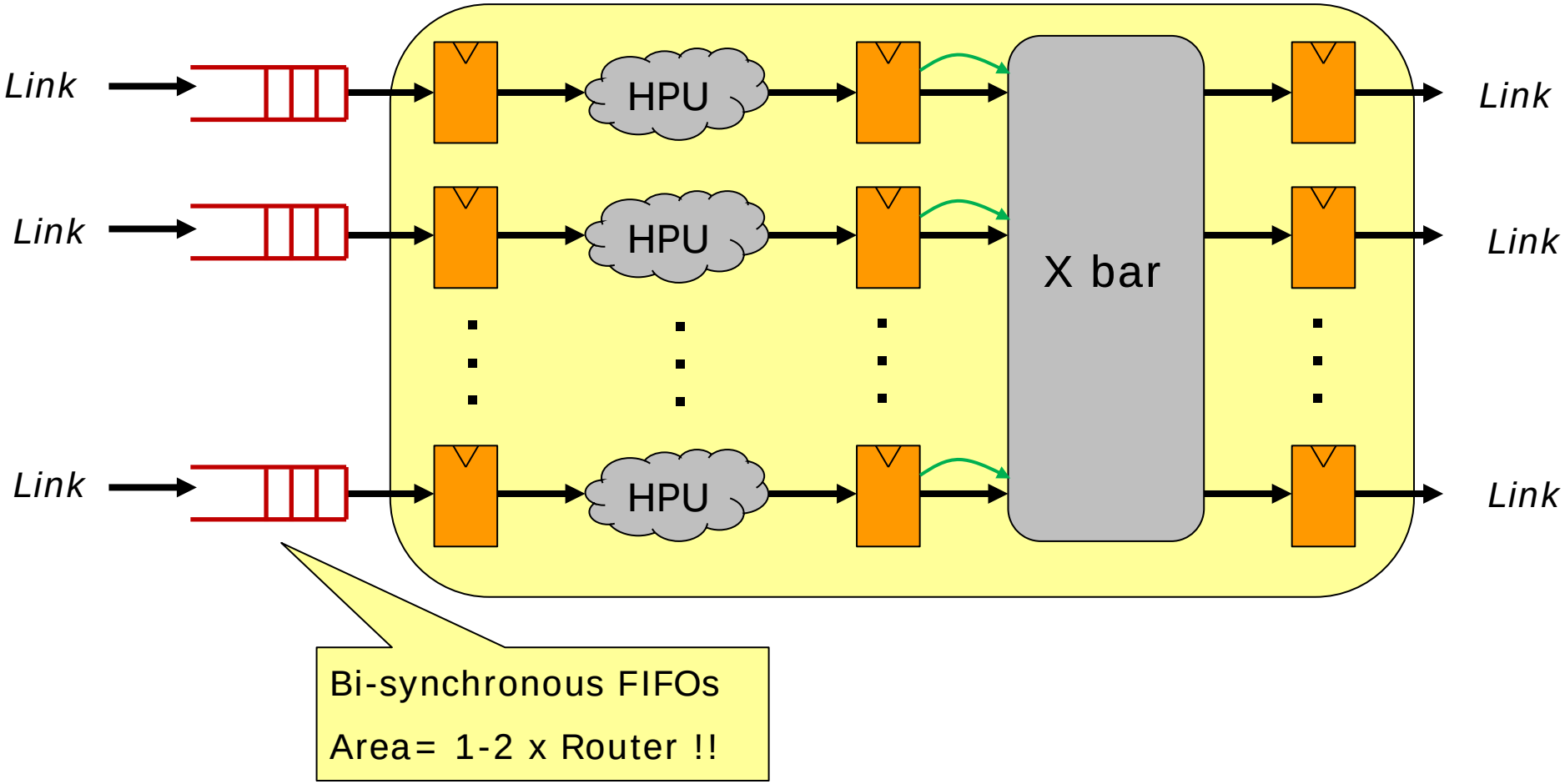
Timing organization?

- mesochronous (same oscillator, some skew)



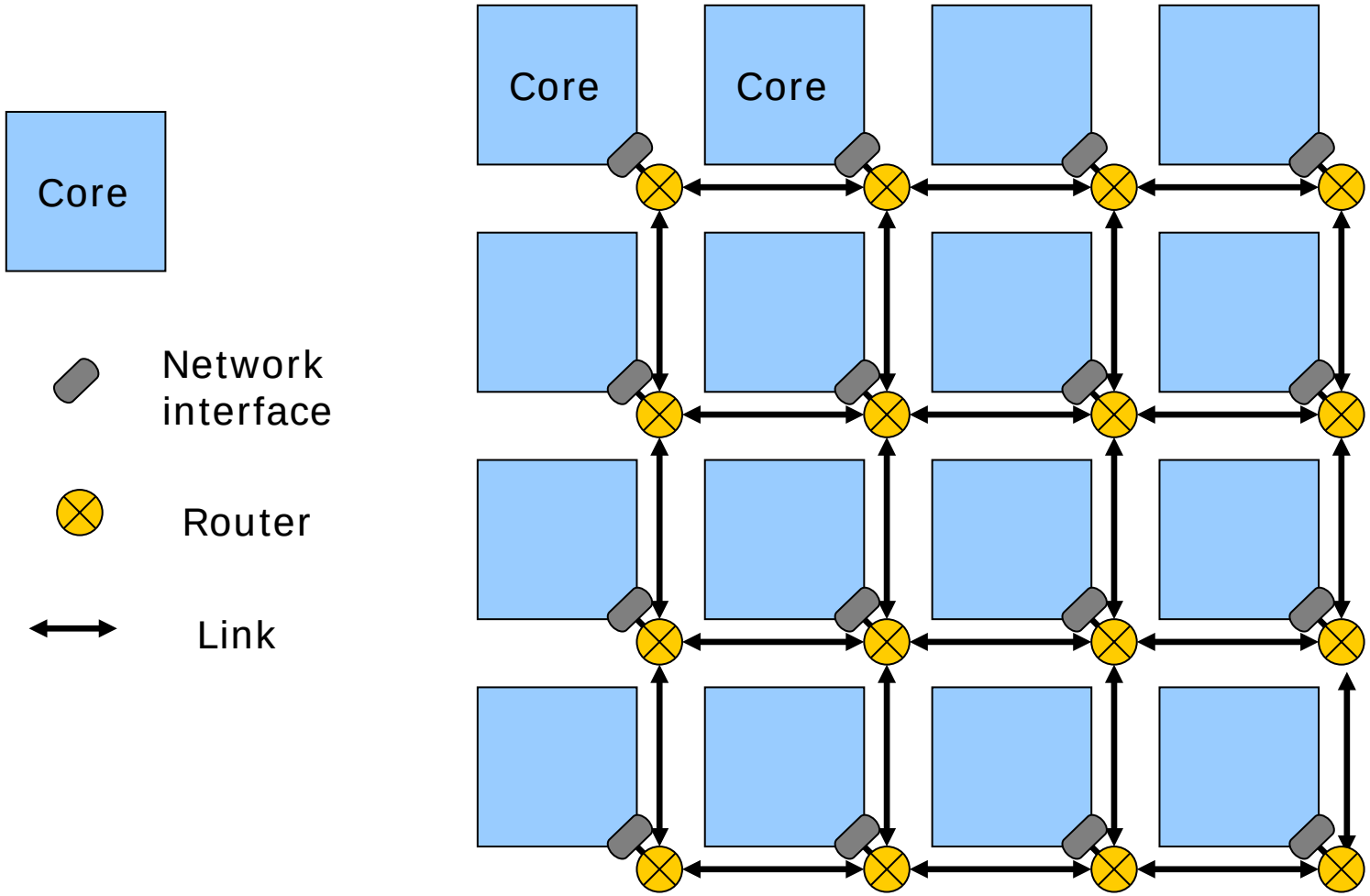
Mesochronous router

- synchronous router + bi-synchronous FIFOs



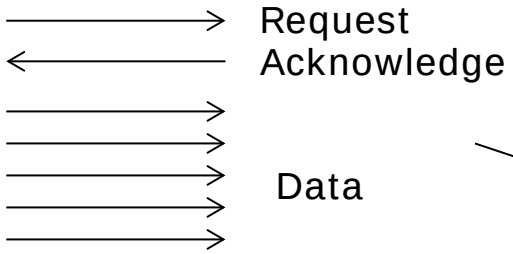
Timing organization?

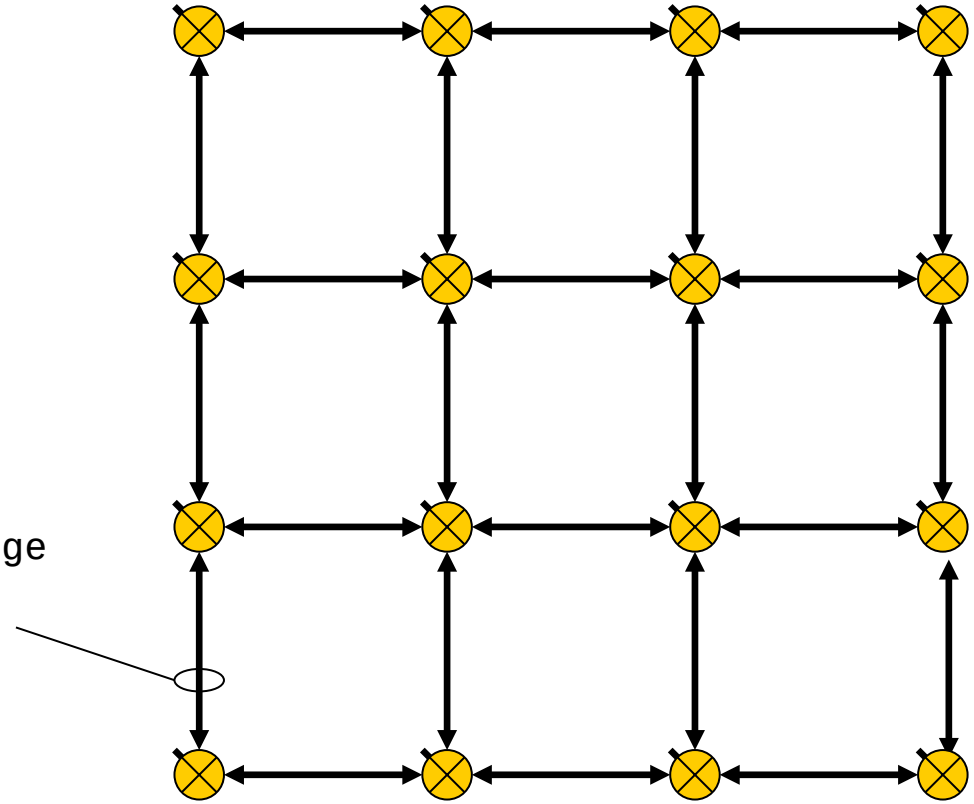
- Globally-Asynchronous Locally-Synchronous



1) Asynchronous routers and links

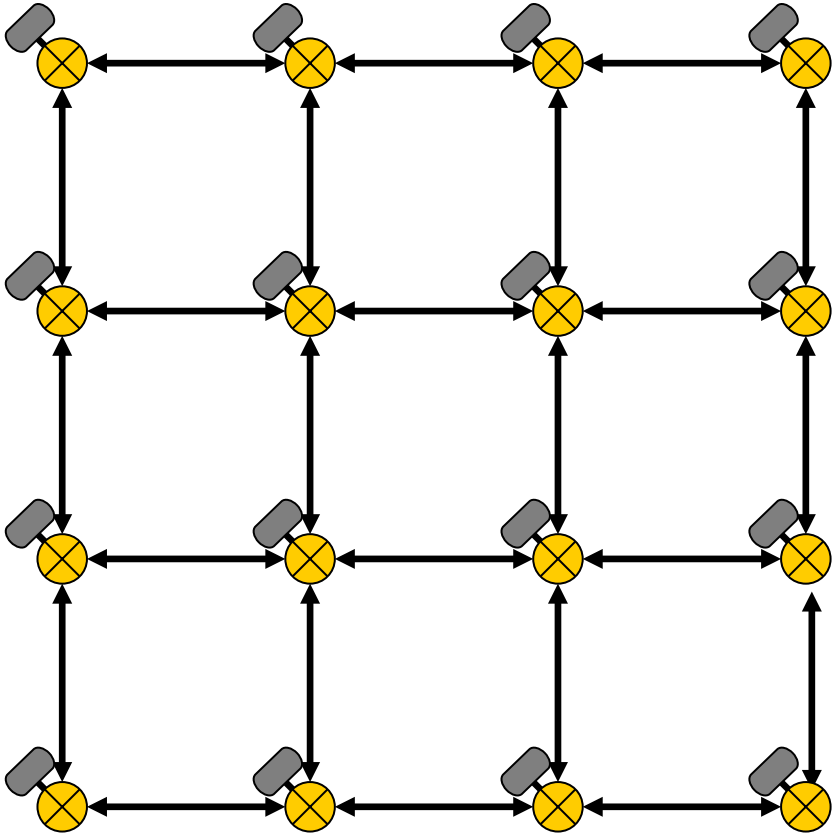
- Link:





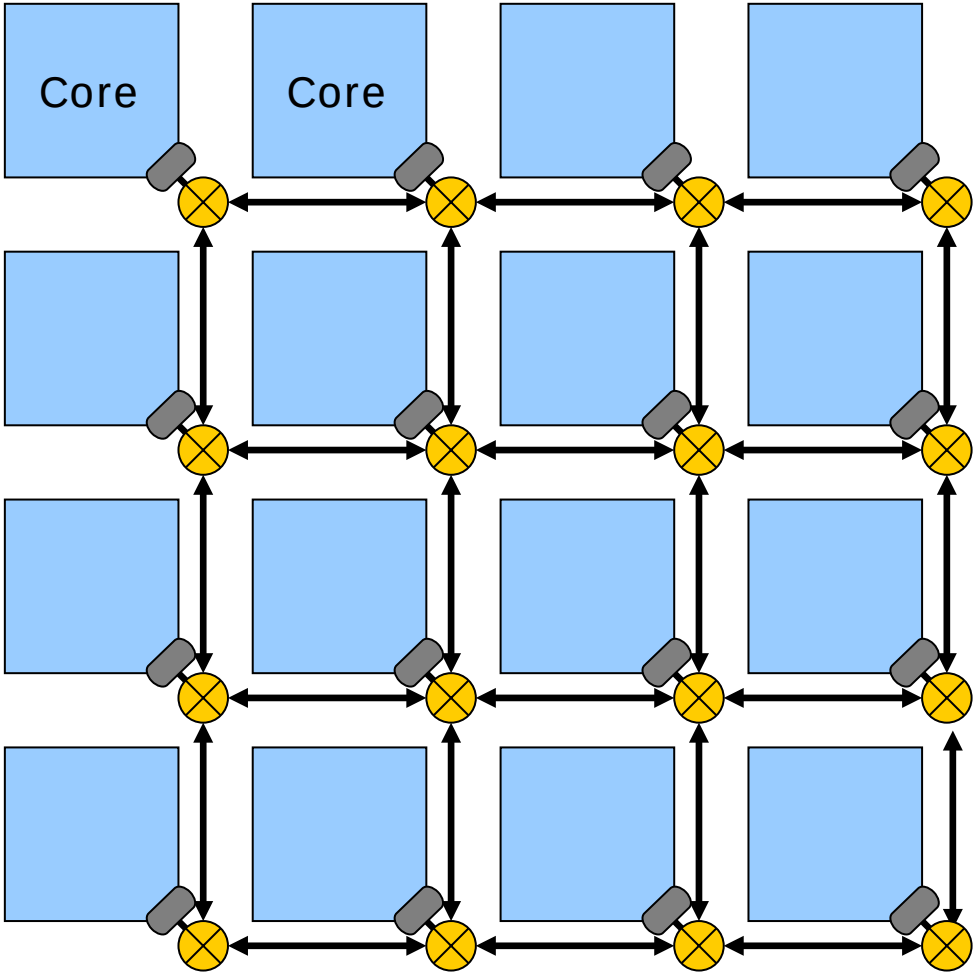
2) Mesochronous Network Interfaces

- Mesochronous =
  - A single oscillator
  - Bounded skew (possibly varying)



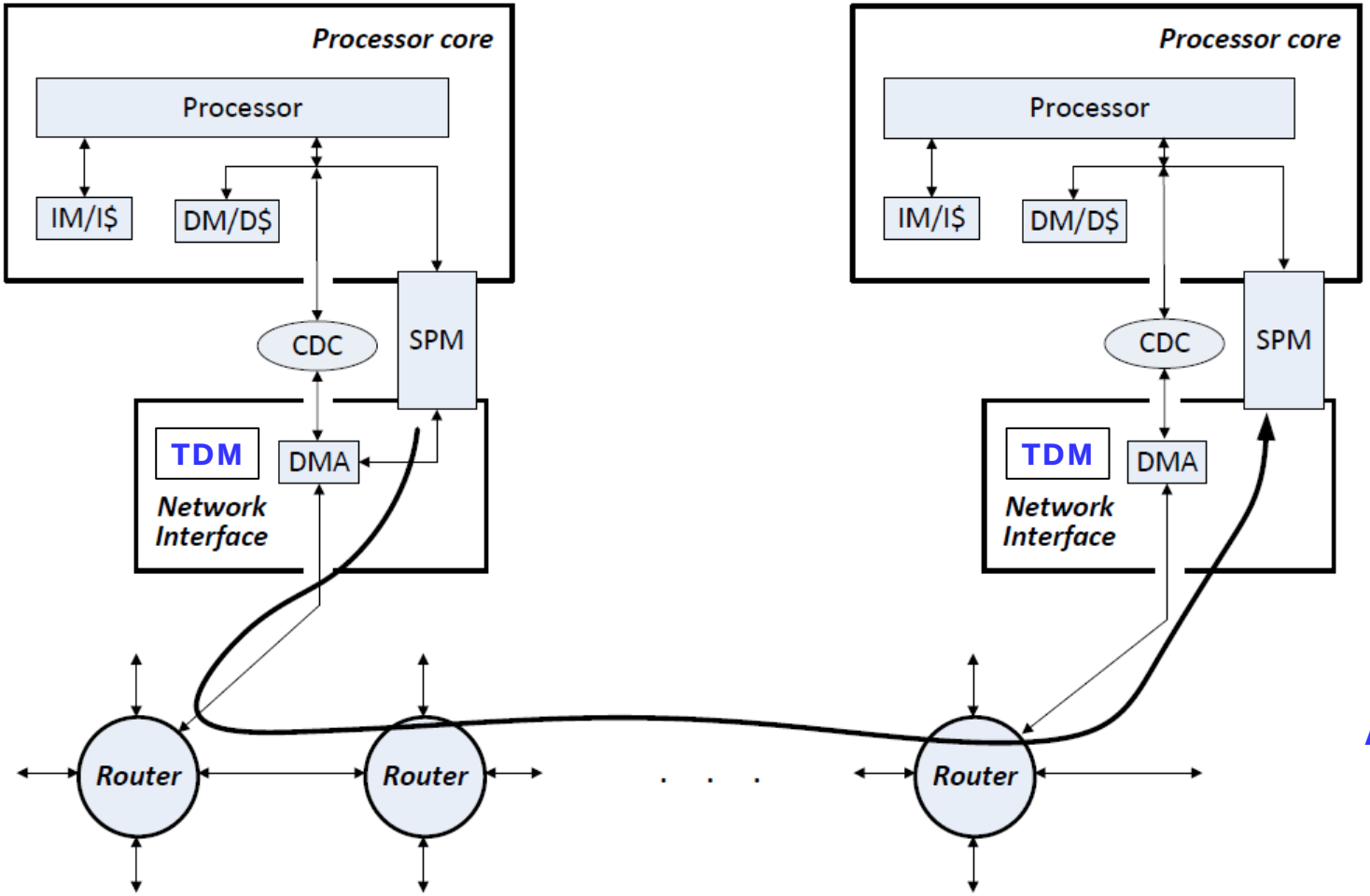
3) Independently clocked IP coores

- Different cores have different “natural” speeds
- Frequency scaling
- Voltage scaling



Clocking strategy

-mesochronous (same oscillator, some skew)

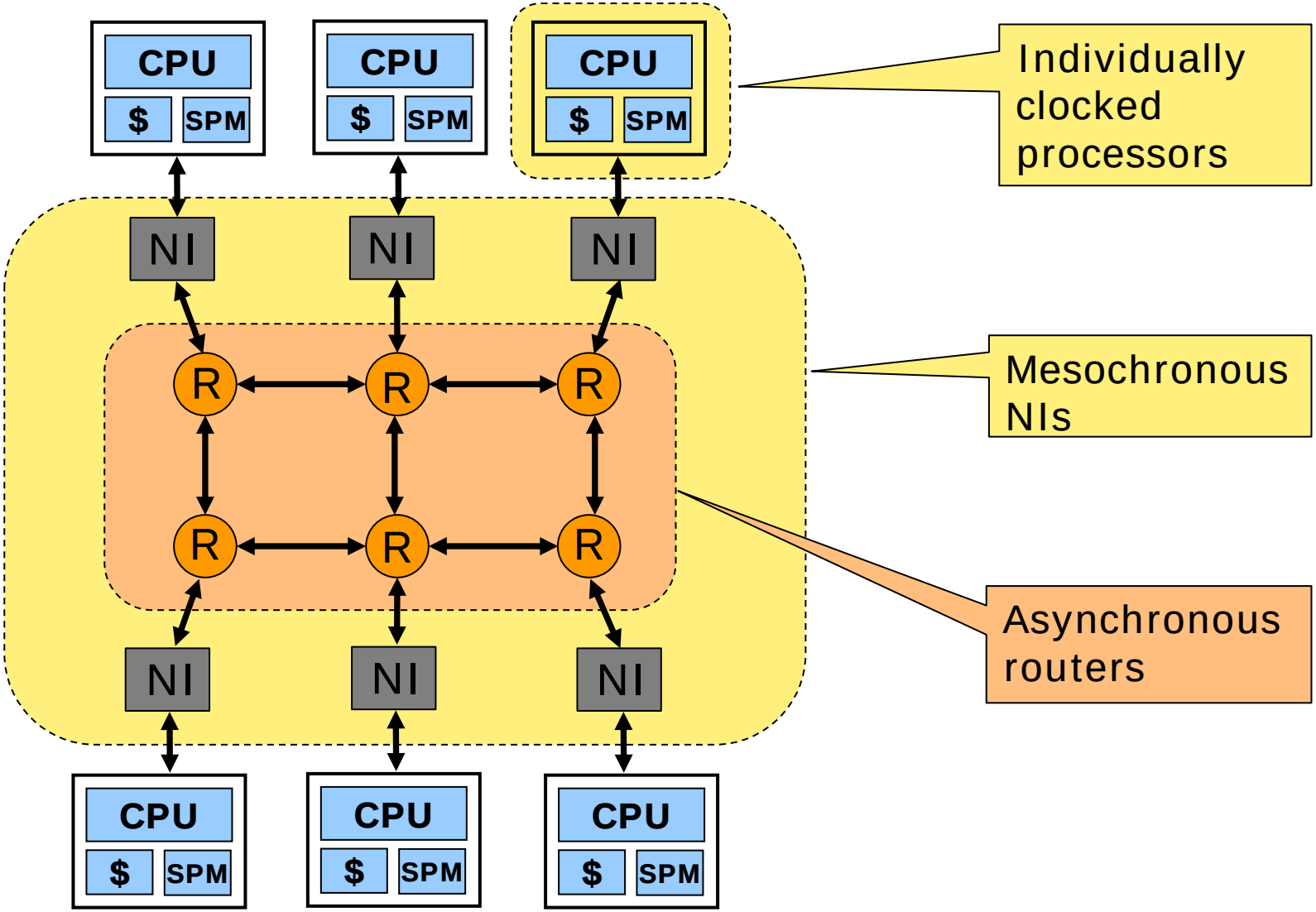


Individually clocked processor cores

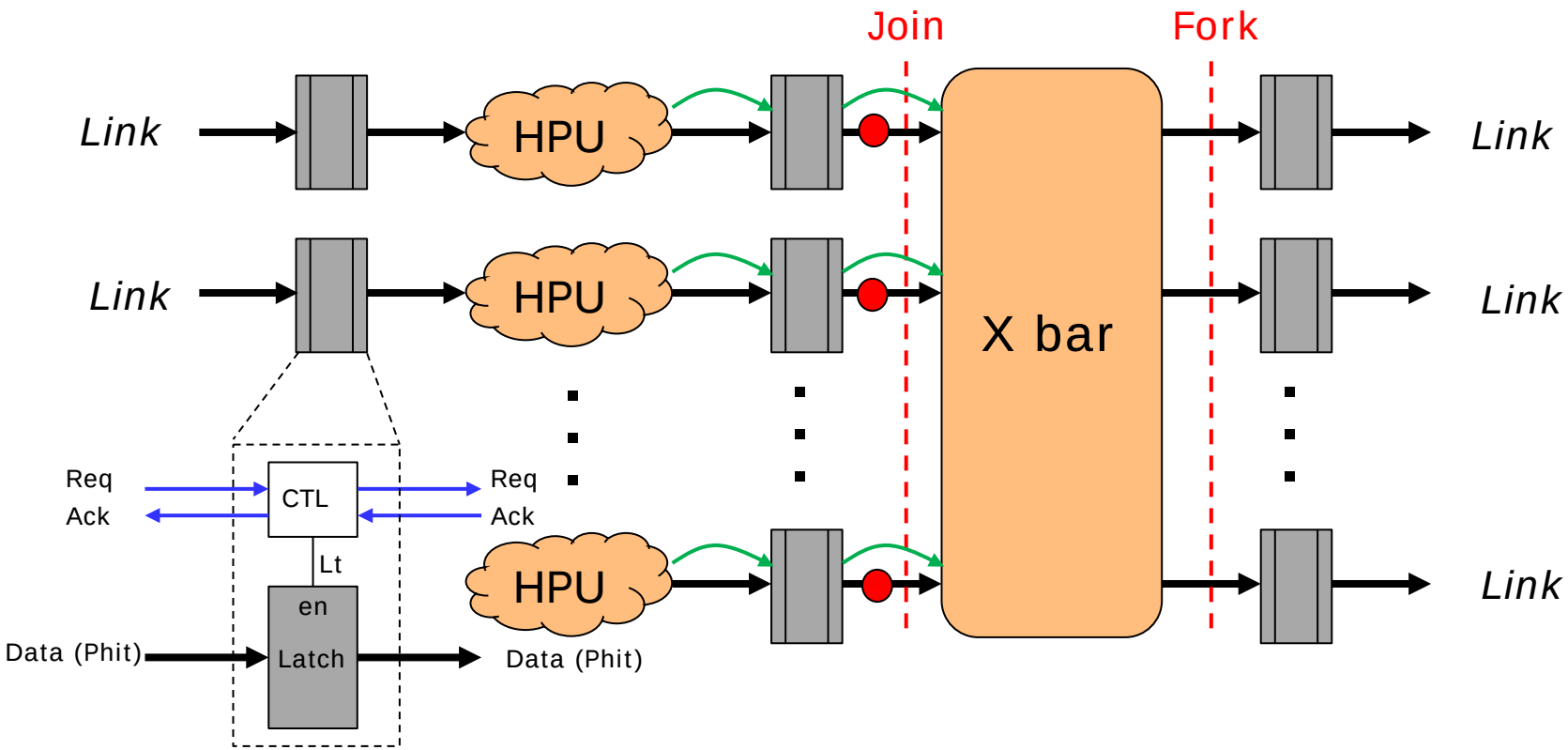
Mesochronous NIs

Asynchronous links and routers

Timing Organization of ARGO



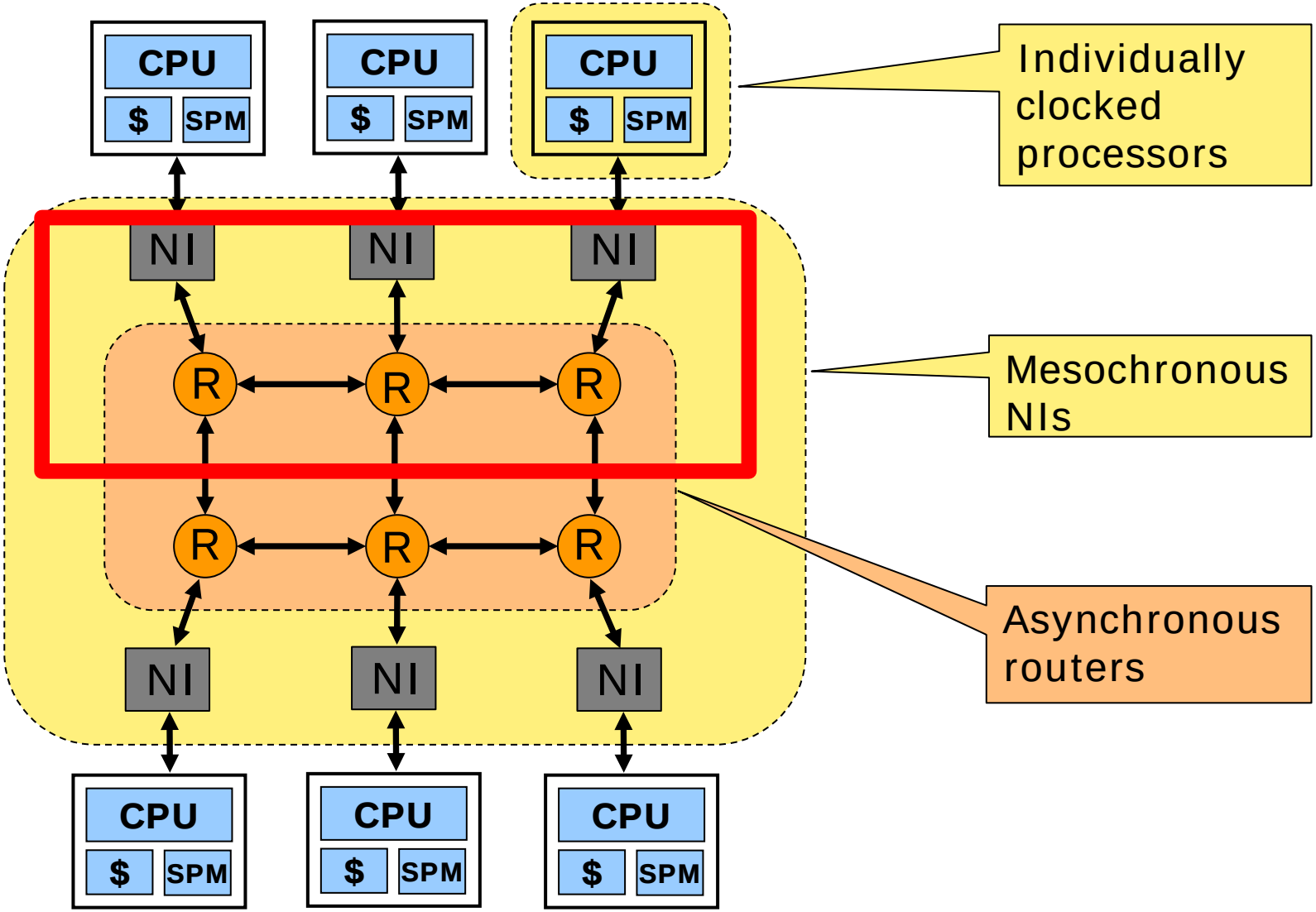
Asynchronous router for source-routed TDM-based NoC



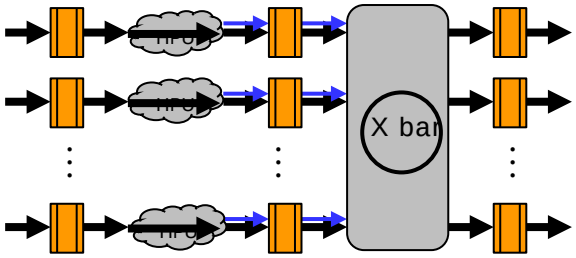
Note: A token is a flit:

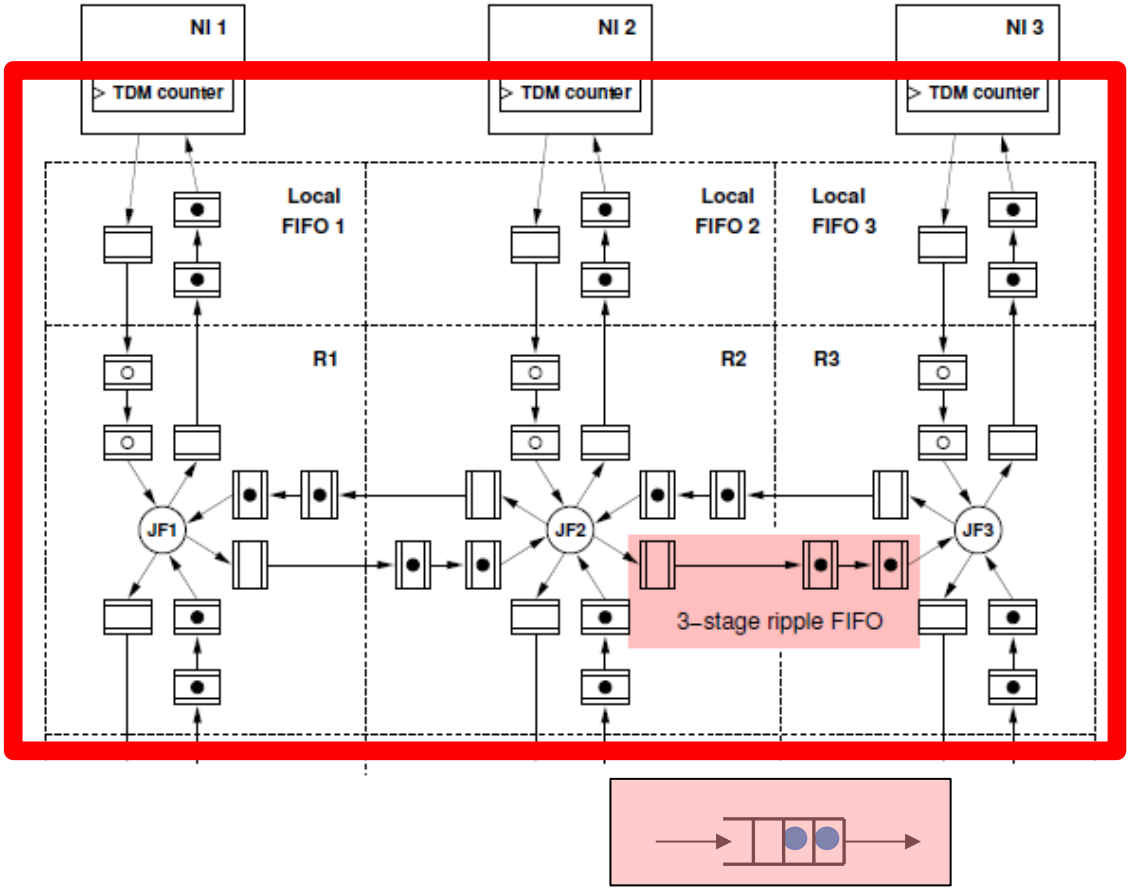
- a word in a packet
- a void

Timing Organization of ARGO

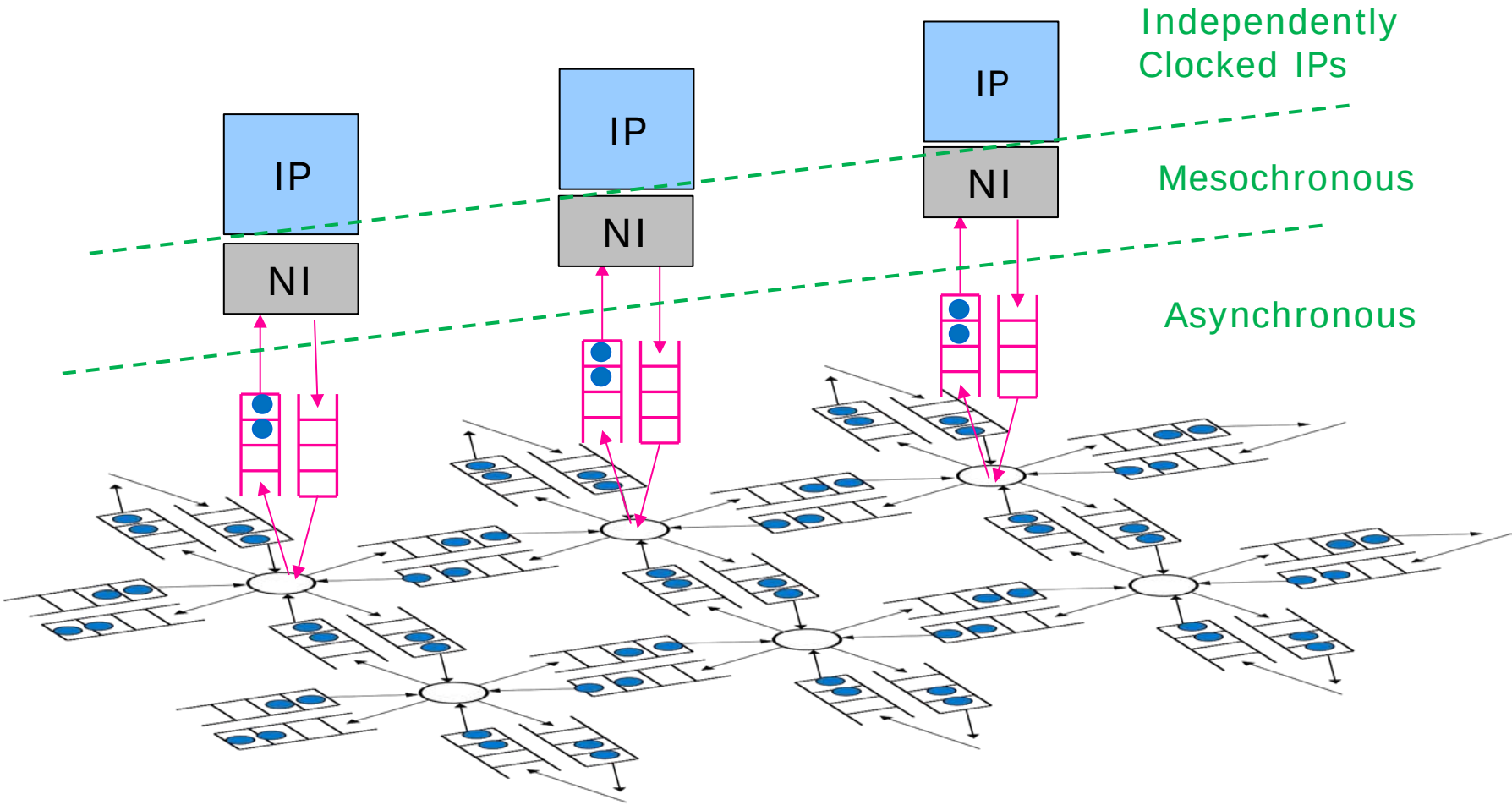


Understanding the NoC

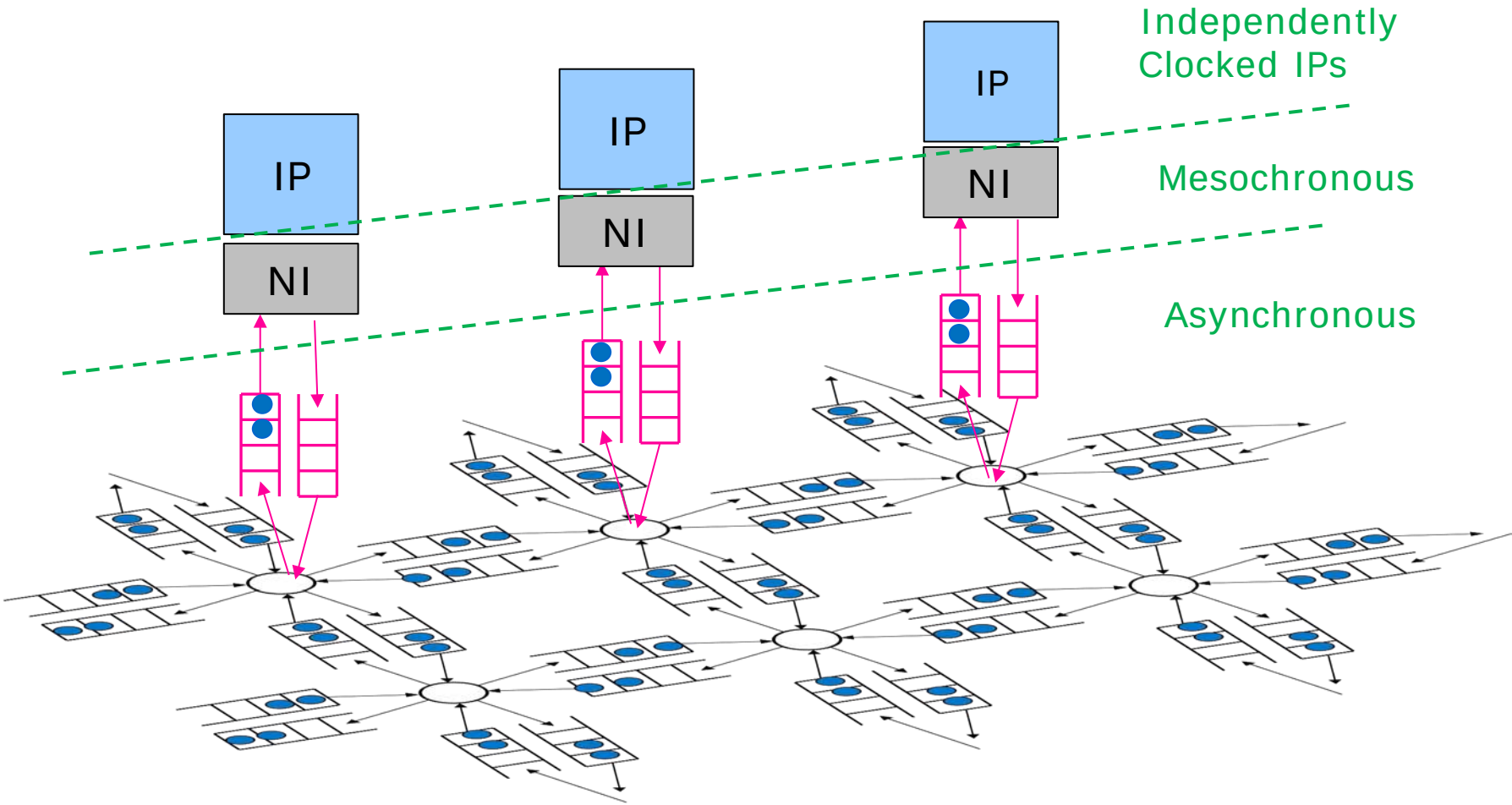




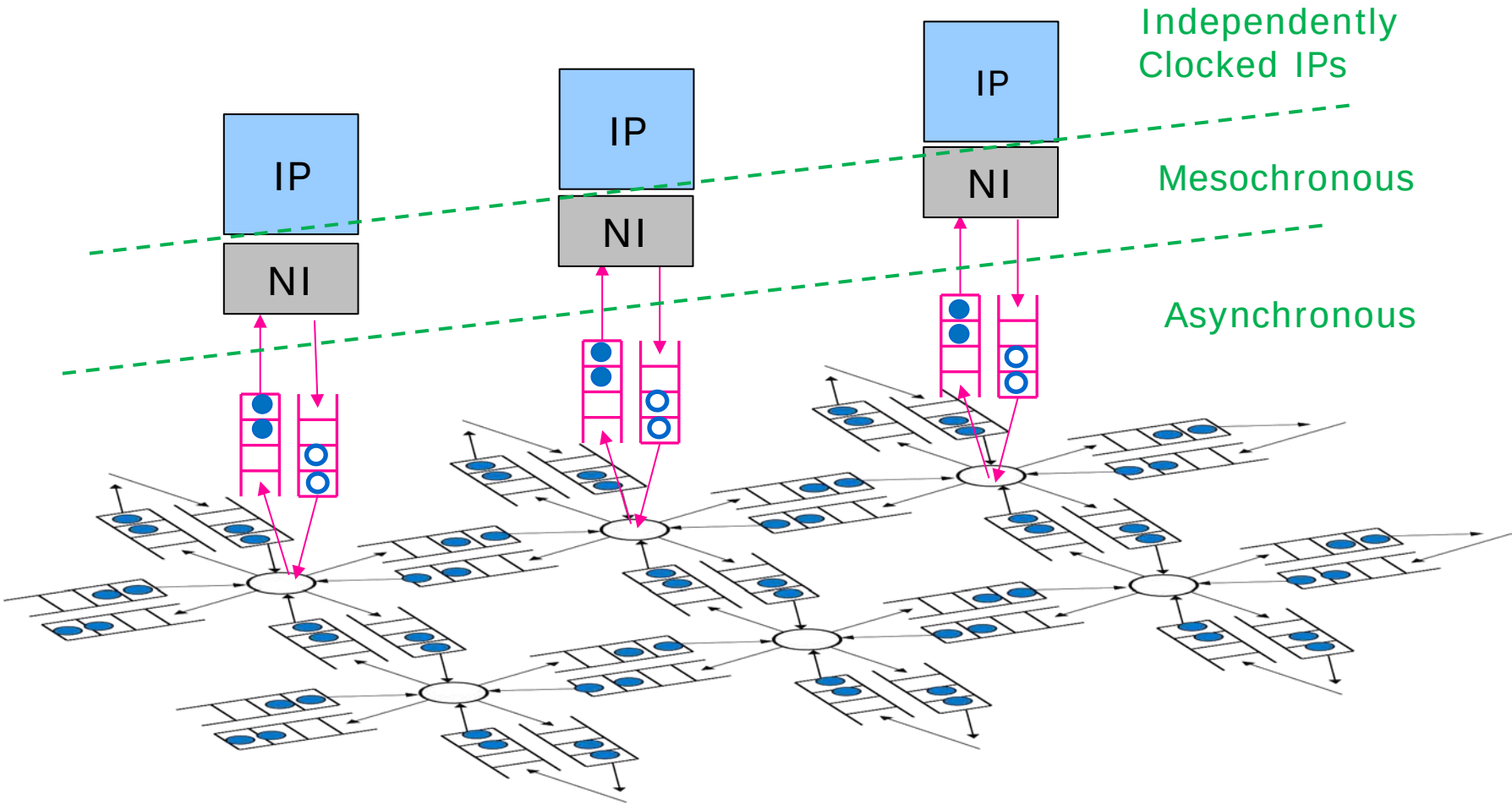
Understanding the NoC



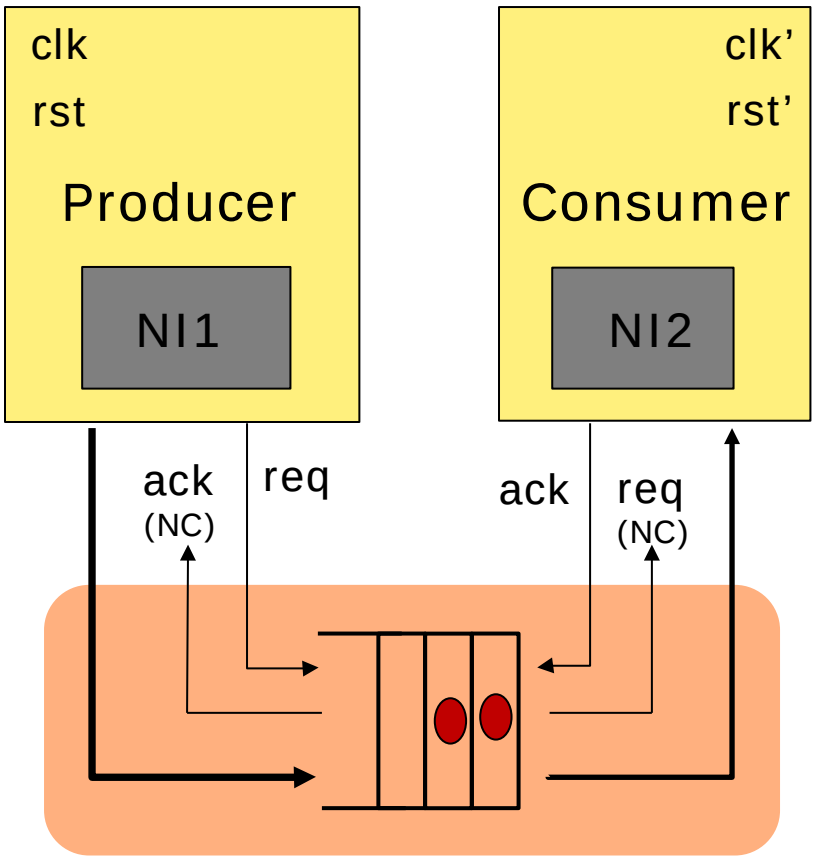
Resetting the NoC



Resetting the NoC



Mesochronous-Asynchronous Interface



- Mesochronous NIs
  - Producer and consumer operate at TDM clk : T_{clk}
  - Skew in reset and clock results in phase shift ($\pm\delta$) among TDM schedules in NIs.
    Possibly more than 1 cycle!

- Timing assumption:
  - Routers are faster than NIs

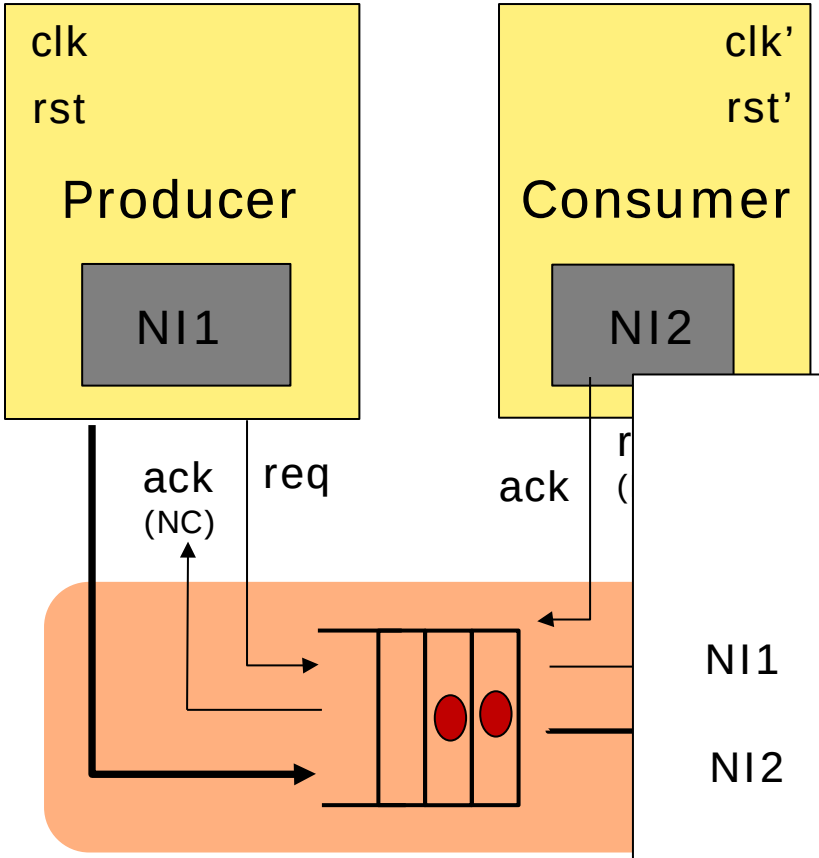
    $$T_{Handshake} < T_{clk}$$

  - Producer ignores ack
    consumer ignores req

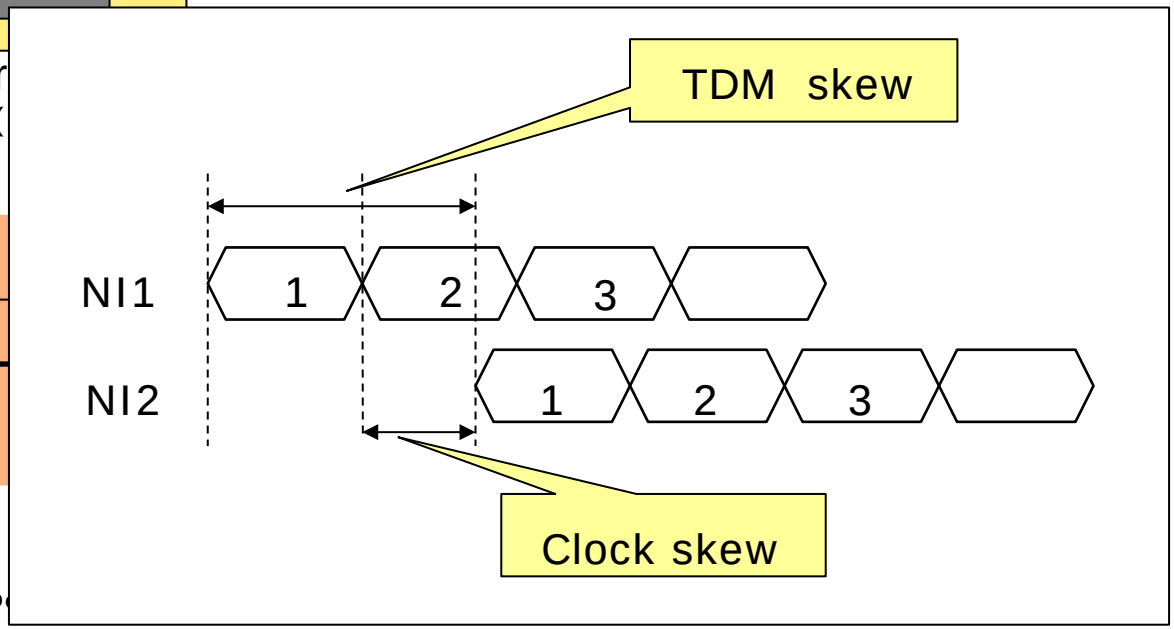
  → no synchronization

  → no metastability

Mesochronous-Asynchronous Interface

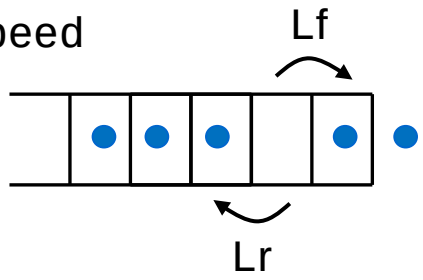


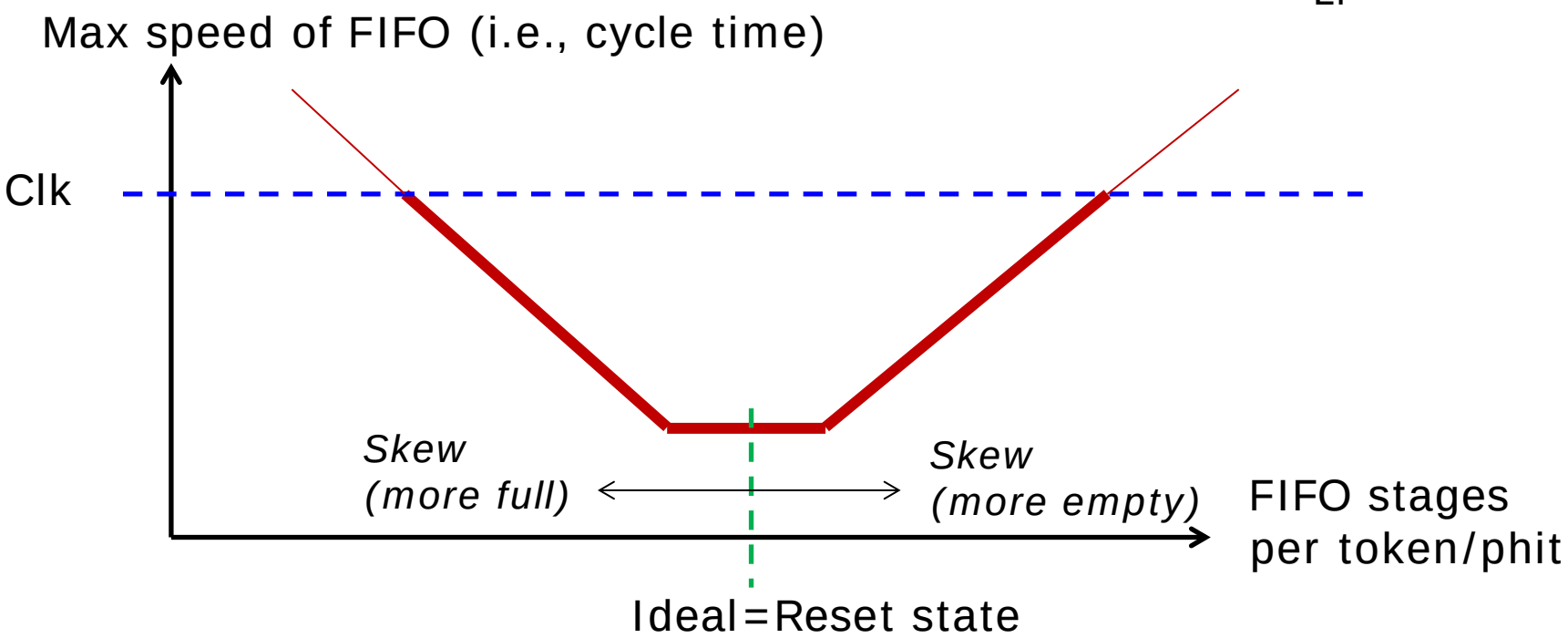
- Mesochronous NIs
  - Producer and consumer operate at TDM clk : T_{clk}
  - Skew in reset and clock results in phase shift ($\pm\delta$) among TDM schedules in NIs.
    Possibly more than 1 cycle!



Tolerating skew – How much?

- Producer and consumer drive FIFO below its max. speed
- Speed of a FIFO depends on how full it is
- Skew tolerance = F(Structure, Lr, Lf, Tclk)





Performance Analysis of Concurrent Systems

- System model
  - Timed marked graph (a subclass of Petri nets)
  - Time Separation of Events (TSE)

- Two classes of algorithms:
  - Steady state average TSE
  - Worst case TSE (possibly an initial phase followed by oscillations)

- We have used the worst-case TSE algorithm of Hulgaard et al.:
  - H. Hulgaard, S. M. Burns, T. Amon, and G. Borriello. An algorithm for exact bounds on the time separation of events in concurrent systems. IEEE Transactions on Computers, 44(11), Nov. 1995.

- Model 1: Detailed model of handshake latch implementation.
- Model 2: Coarser model of handshake latch implementation:
  - Analysis of complete 2 x 2 NoC (same results)
  - Study of possible oscillations of max TSE.

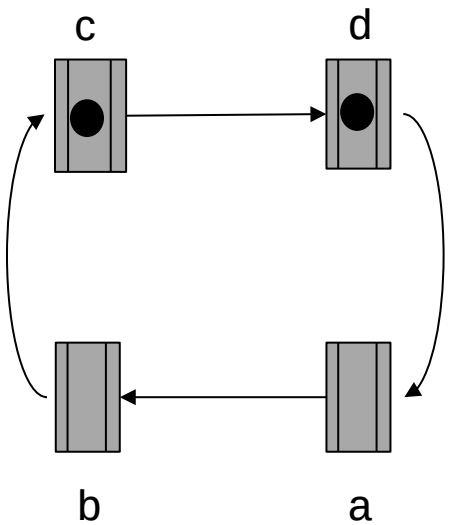
Example of coarse model

- Ring w. 4 handshake-latches and 2 tokens

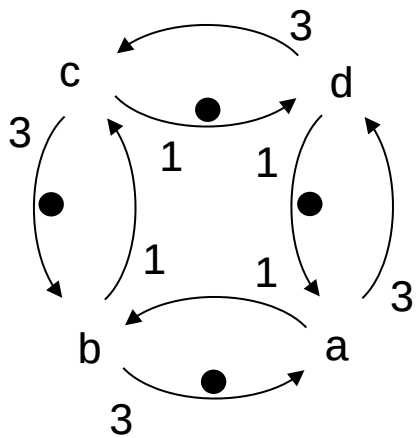
$\longrightarrow$ Forward latency: $Lf = 1$

$\longleftarrow$ Reverse latency: $Lr = 3$

Circuit:



Timed marked graph:



- Time separation between successive tokens written into latch a:

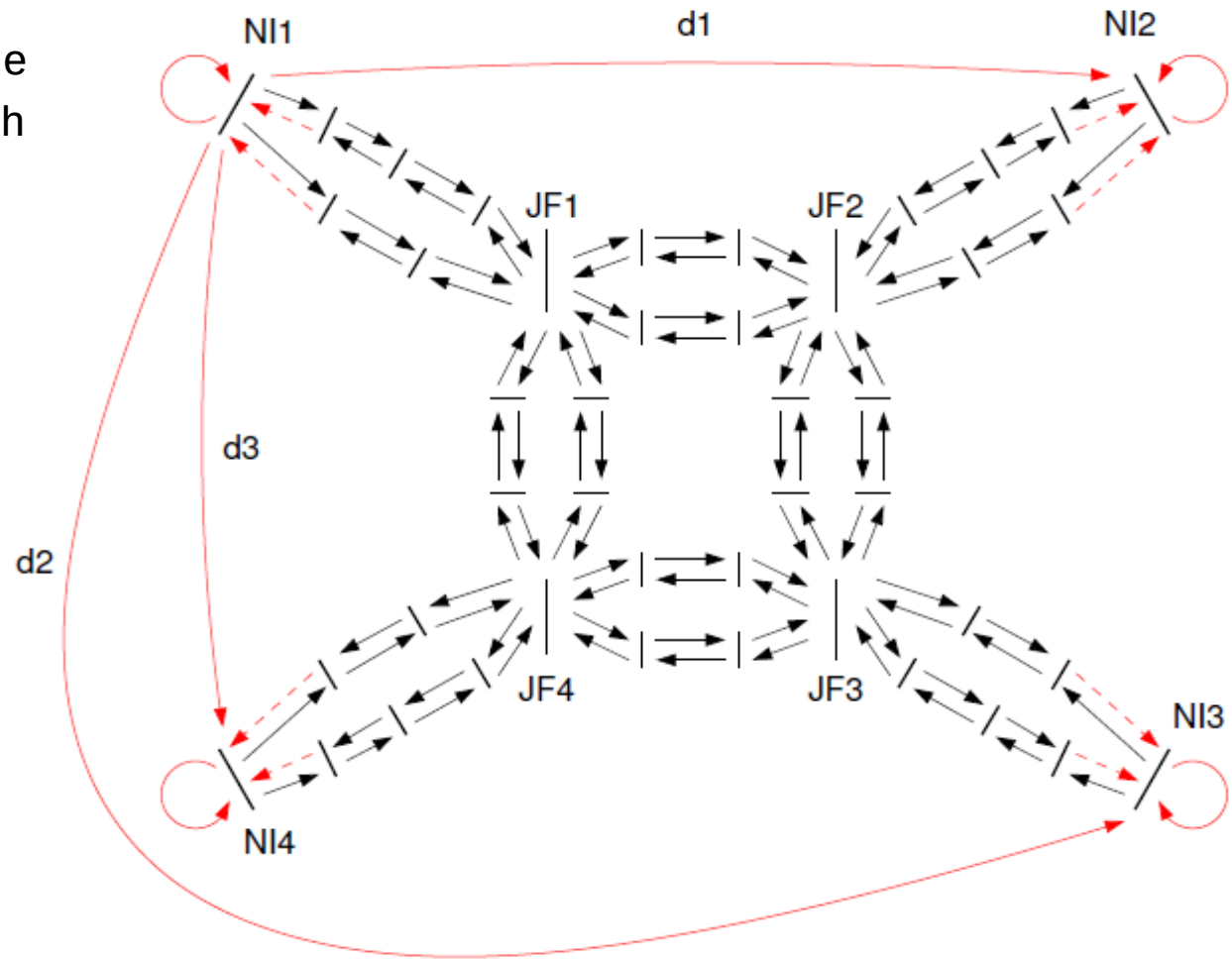
  4, 8, 4, 8, ... average is 6

Coarse-Grained Model of 2x2 NoC

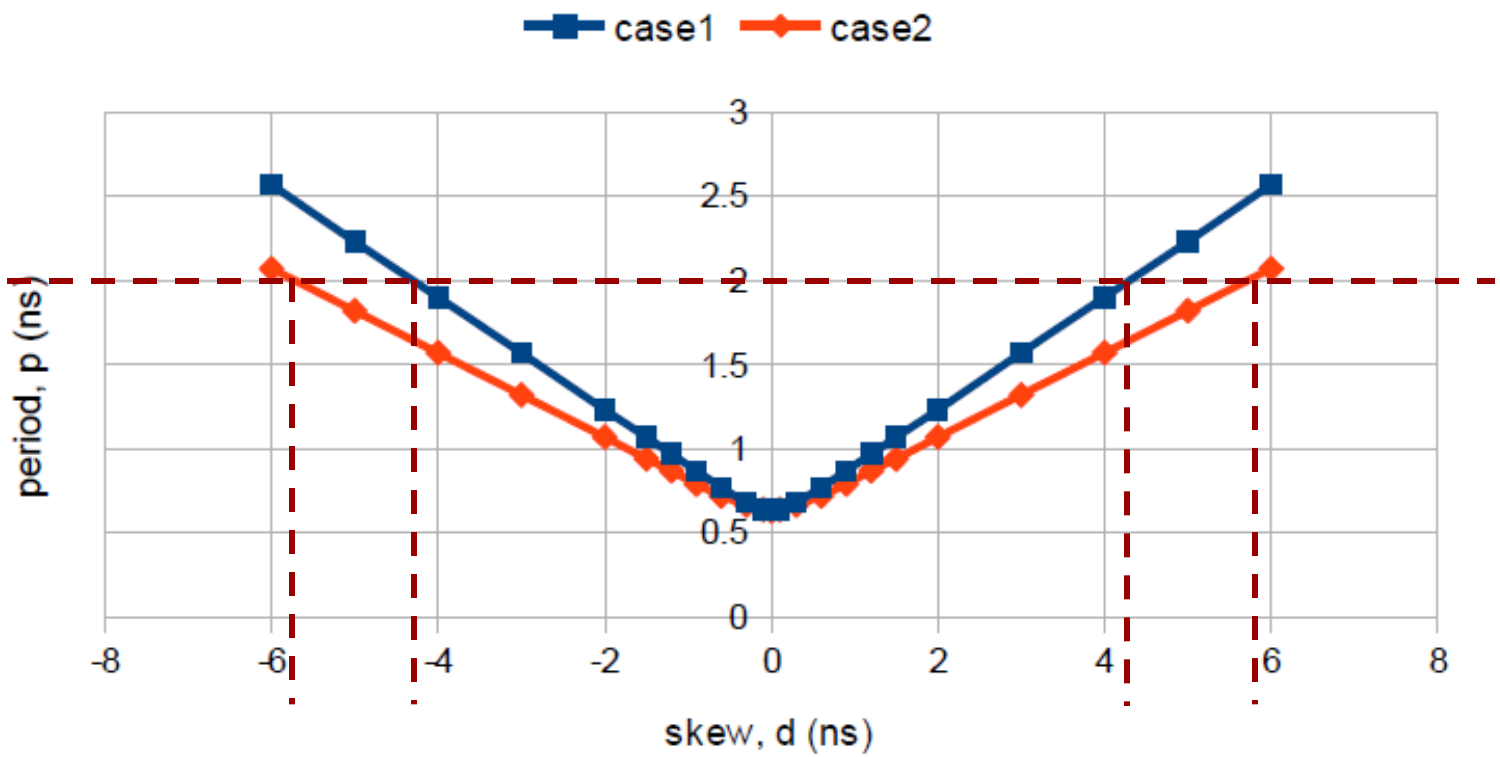
- Pipeline stage = node
- Edges annotated with Lf and Lr

- Case 1:
  Skew among neighbour nodes
  - d1
  - d2=d3=0

- Case 2:
  Skew among diagonal nodes
  - d2
  - d1=d3=0



Skew – Period Results



- Neighboring nodes (case 1) is the worst-case skew tolerance

Generating schedules

- Metaheuristic scheduler
  - Input: Core communication graph w. bandwidth requirements
  - Output: Schedule + (clock)frequency

- Scheduler works on normalized BW requirements
  - Lowest BW requirement assigned 1 slot
  - Highest BW requirement assigned n slots (n can be large)
  - Schedules can be compressed by over-assigning BW to channels with small BW requirements. For a range of benchmarks we found that compressing to 100 slot TDM periods is possible with negligible effects (i.e., need to increase clock frequency)

- NoC is effecticely drained for packets between schedule periods (except for pipeline depth in shortest NI-to-NI path.
  - Perspectives for fast reconfiguration (support for mode changes)

Conclusions

- The Argo NoC combines TDM and GALS
  - Asynchronous routers
  - Mesochronous NIs
  - Independently clocked processor cores

- Significant time-elasticity provided by network of asynchronous routers.

- Small and efficient implementation
  - Avoids run time synchronization, arbitration and buffering
  - End-to-end, SPM-to-SPM transfer of data controlled by TDM schedule.

References

- Network interface:
  - J. Sparsø, E. Kasapaki, and M. Schoeberl, “An area-efficient network interface for a TDM-based network-on-chip,” *in Proc. Design, Automation and Test in Europe (DATE),* 2013, pp. 1044–1047.
  - R. B. Sørensen, L. Pezzarossa, and J. Sparsø, “An area-efficient TDM NoC supporting reconfiguration for mode changes,” in Proceedings of the International Symposium on Networks-on-Chip (NOCS). IEEE, 2016.
- Asynchronous router and timing analysis:
  - E. Kasapaki and J. Sparsø,“Argo: A Time-Elastic Time-Division-Multiplexed NoC using Asynchronous Routers,” *in Proc. IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*. IEEE Computer Society Press, 2014, pp. 45–52.
  - E. Kasapaki and J. Sparsø, “The Argo NoC: Combining TDM and GALS”, *in Proc. European Conference on Circuit Theory and Design (ECCTD),* 2015, pp. 1-4.
- Scheduler
  - R. B. Sørensen, J. Sparsø, M. R. Pedersen, and J. Højgaard, “A Metaheuristic Scheduler for Time Division Multiplexed Networks-on-Chip,” *in Proc. IEEE/IFIP Workshop on Software Technologies for Future Embedded and Ubiquitous Systems (SEUS),* 2014, pp. 309–316.

References (cont.)

- Journal articles with a broader perspective
  - E. Kasapaki, M. Schoeberl, R. B. Sørensen, C. T. Müller, K. Goossens, and J. Sparsø, "Argo: A Real-Time Network-on-Chip Architecture with an Efficient GALS Implementation," *IEEE Transactions on VLSI Systems,* vol. 24, no. 2, pp. 479–492, 2015.
  - M. Schoeberl, S. Abbaspour, B. Akesson, N. Audsley, R. Capasso, J. Garside, K. Goossens, S. Goossens, S. Hansen, R. Heckmann, S. Hepp, B. Huber, A. Jordan, E. Kasapaki, J. Knoop, Y. Li, D. Prokesch, W. Puffitsch, P. Puschner, A. Rocha, C. Silva, J. Sparso, and A. Tocchi. "T-CREST: Time-predictable multi-core architecture for embedded systems," *Journal of Systems Architecture,* 61(9):449-471, 2015.

- Open source:
  - Argo hardware: https://github.com/t-crest/argo
  - Scheduler: https://github.com/t-crest/poseidon

- Starting point for more information about the T-CREST plaform:
  - http://patmos.compute.dtu.dk/

END