Technische Universität Braunschweig

Efficient Latency Guarantees for Mixed-criticality Networks-on-Chip

Sebastian Tobuschat, Rolf Ernst

IDA, TU Braunschweig, Germany

18. April 2017, Pittsburgh, PA, USA **CPSWeek • RTAS 2017**

Current and Future Embedded Systems

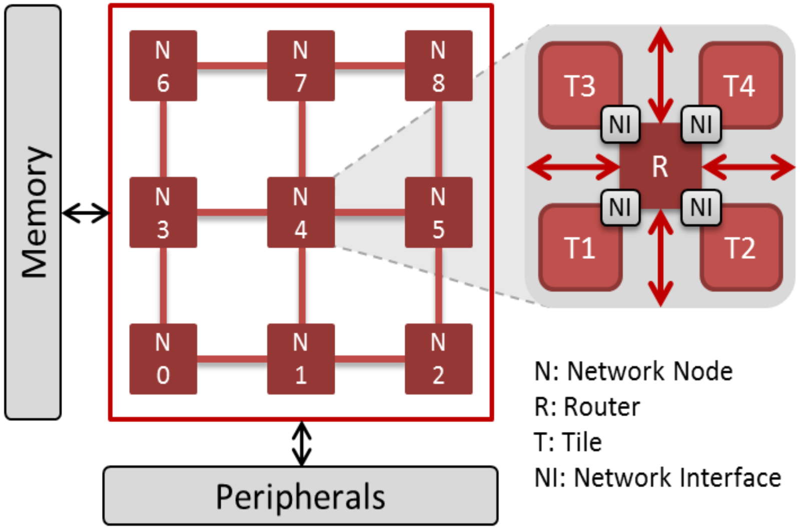
- **multicore architectures** are reaching safety-critical embedded systems
  - e.g. sensor fusion and recognition in highly automated driving
- integrate previously distributed functions in a single chip

  → **mixed-criticality systems**
- standards require isolation in case of shared resources
  - e.g. IEC 61508: "**sufficient independence**"

Networks-on-Chip (NoC)

- offer high-performance, scalability and flexibility
- transmissions share NoC resources
  - e.g. buffers, links

  → provide **isolation**



- consequences
  - highest relevant safety level for shared parts

    → expensive
  - or implement "**sufficient independence**"

    → **Quality of Service mechanisms** (QoS)
  - main Challenge: QoS guarantees + high performance

Providing Quality of Service – Related Work

- static partitioning (e.g. TDMA):
  e.g. [Milberg2004], [Goossens2010], [Psarras2015], [Panades2006], [Hansson2007]
  → typically reduced utilization
- prioritization:
  e.g. [Bolotin2004], [Bjerregaard2005]
  - "criticality as priority"
  → reduced performance for BE
- dual Priority:
  e.g. [Burns2014], [Indrusiak2015]
  - based on behavior of safety-critical sender:
    - send with either high or low priority
  - not accounting for NoC load; only for whole path
  → reduced exploitation of latency slack

Providing Quality of Service – Related Work

- static partitioning (e.g. TDMA):
  e.g. [Milberg2004], [Goossens2010], [Psarras2015], [Panades2006], [Hansson2007]
  → typically reduced utilization
- prioritization:
  e.g. [Bolotin2004
  - “criticality as
  → reduced pe
- dual Priority:
  e.g. [Burns2014]
  - based on be
    - send with either high or low priority
  - not accounting for NoC load; only for whole path
  → reduced exploitation of latency slack

Goal: minimize negative performance impact of QoS mechanisms (on non-critical senders)

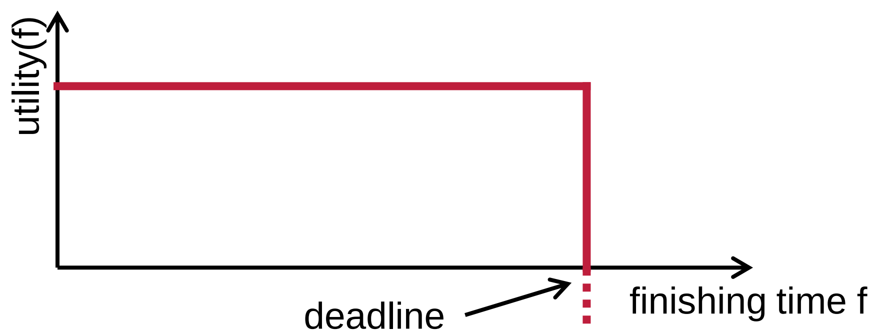
Idea: prioritize BE to exploit (latency) slack of critical applications

Outline

- Motivation
- Providing Quality of Service
- Latency Guarantees
- Experimental Results
- Conclusion

Idea

- latency slack
  - difference between worst-case latency and deadline
- safety critical applications do not benefit from finishing before deadline
- but BE applications benefit from low latency
- baseline approach:
  - two traffic classes: guaranteed latency (GL) and best effort (BE)
  - **prioritize BE** over GL and **limit interference** BE induces to GL to exploit slack of GL



utility function of a firm/hard real-time task (Stankovic1998)

Limit BE Interference

- extend (GL) packet header with **blocking counter (BC)**
  - packet or flit level (tradeoff: performance and overhead)
    - for small or single size packets → packet level sufficient
    - different sizes or more fine granular → flit level

- BC is evaluated and adapted in each router
  - decremented when packet is blocked by higher priority packet (this can be BE or other GL with BC=0)

- if BC of a packet/flit reaches zero:
  - prioritize queue containing it over BE, until no packet/flit with BC=0 is remaining
    - other implementation possible: sorting, forwarding/overtaking

Blocking Counter

- header field allows to freely distributed the allowed blocking on the path, based on actual needs of BE
  - account for local or temporary traffic hot spots

- initial value obtained from analysis
  - optimization problem: find initial BC value that minimizes slack, while all (GL) streams are schedulable

- can account for local behavior of sender and (online) adapt BC on packet level
  - e.g. using sender information (cf. [Burns2014], [Indrusiak2015])
  - e.g. allow mode change, software update, task re-mapping

Outline

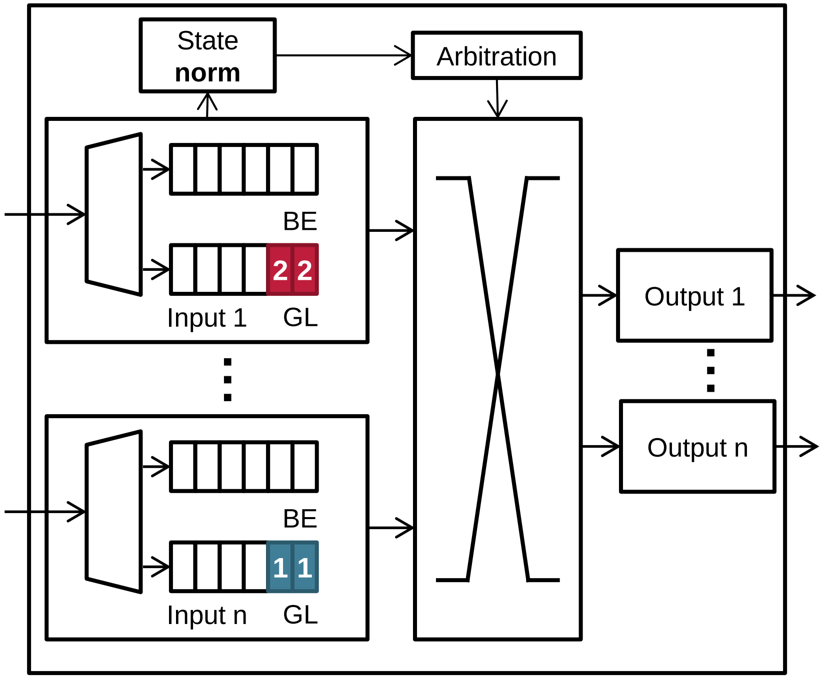
- Motivation
- Providing Quality of Service
- Latency Guarantees
- Experimental Results
- Conclusion

Worst-case Latency

- based on [Rambo2015] – compositional performance analysis
- local router analysis
  - **worst-case multiple activation processing time** for a stream $\boldsymbol{B_i^+}$
    - maximum time resource (router) is busy processing q flits of a stream
    - used to derive **worst-case latency** $\boldsymbol{R_i^+}$ of each hop
  - break down into **sum of** different terms addressing **different blocking factors**
- for each stream
  - analyze routers along its path and **propagate event models** downstream
- formally analyze routers iteratively

Worst-case Multiple Activation Processing Time

$$B_i^+(q, a_i^q) \leq \mathbf{q} * \mathbf{C} + B_i^{out}(B_i^+(q, a_i^q) - C, q) + B_i^{in}(B_i^+(q, a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q, a_i^q) - C)$$



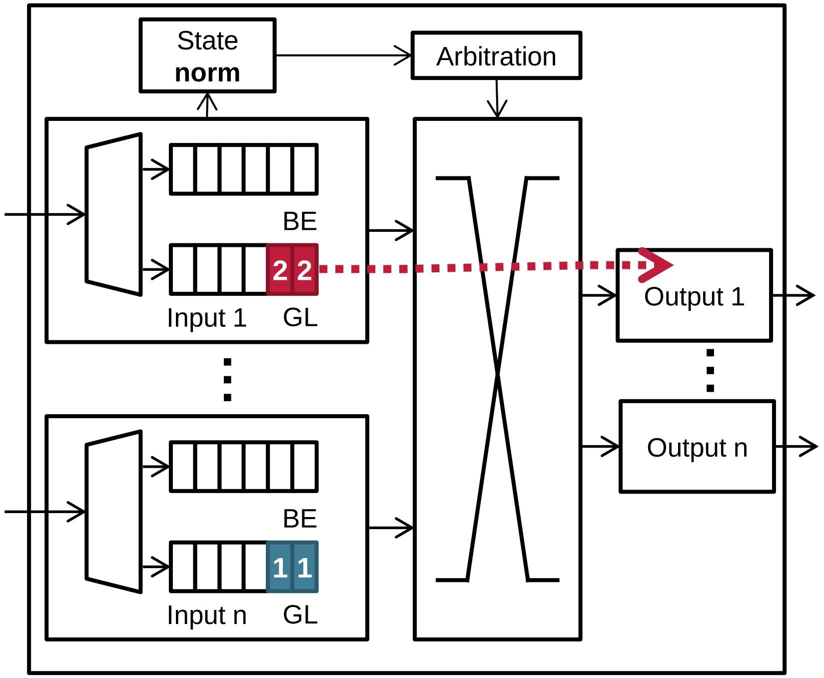
q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q,a_i^q) \leq \mathbf{q * C} + B_i^{out}(B_i^+(q,a_i^q) - C, q) + B_i^{in}(B_i^+(q,a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q,a_i^q) - C)$$

flit transmission

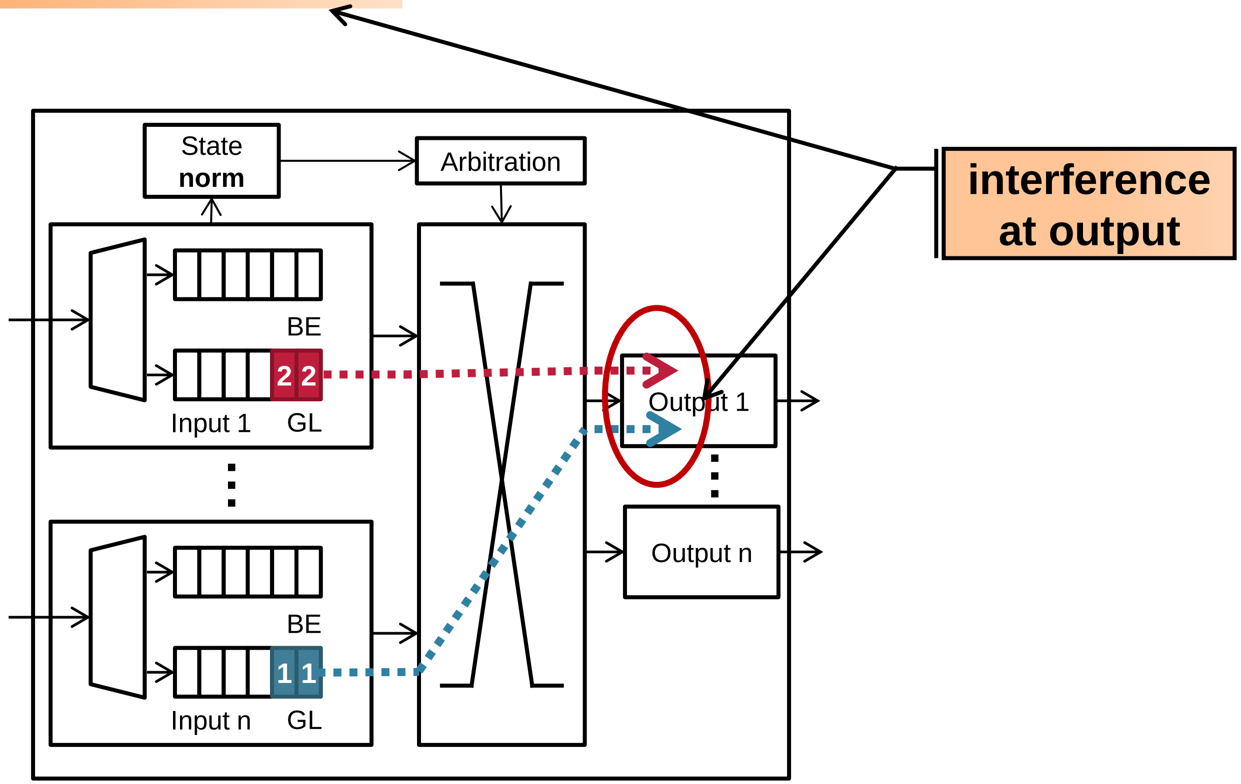


q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q, a_i^q) \leq \mathbf{q * C} + B_i^{out}\left(B_i^+(q, a_i^q) - C, q\right) + B_i^{in}\left(B_i^+(q, a_i^q), q, a_i^q\right) + B_{i,q}^{LP}\left(B_i^+(q, a_i^q) - C\right)$$

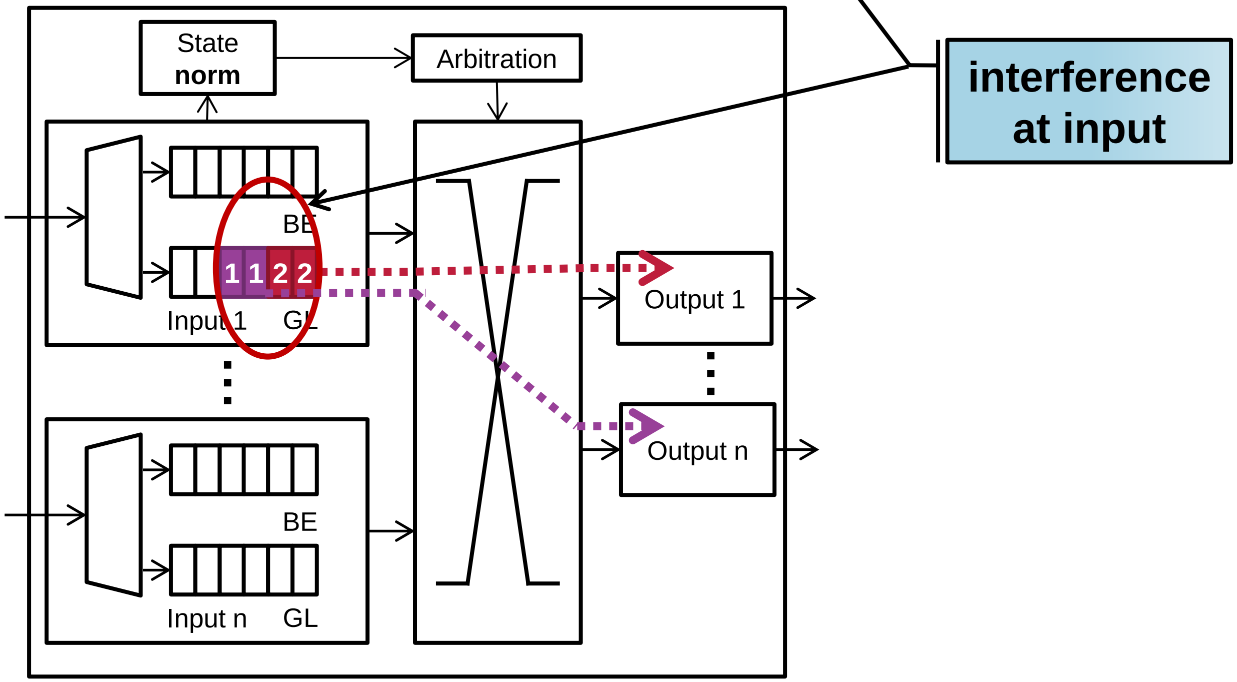


q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q, a_i^q) \leq \mathbf{q * C} + B_i^{out}\left(B_i^+(q, a_i^q) - C, q\right) + B_i^{in}\left(B_i^+(q, a_i^q), q, a_i^q\right) + B_{i,q}^{LP}\left(B_i^+(q, a_i^q) - C\right)$$



q : number of flits

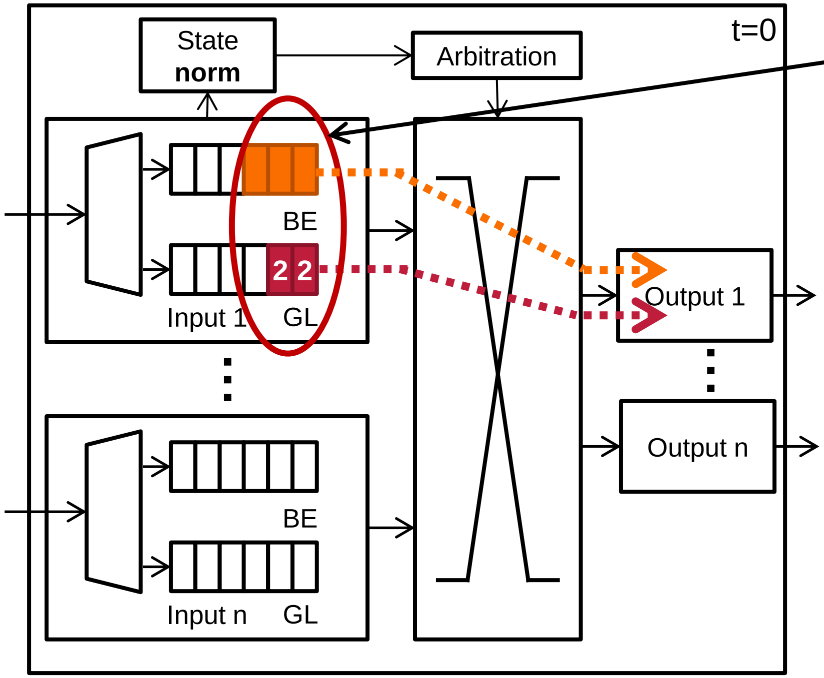
a_i^q : arrival time of event q

C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q, a_i^q) \leq \mathbf{q * C} + B_i^{out}(B_i^+(q, a_i^q) - C, q) + B_i^{in}(B_i^+(q, a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q, a_i^q) - C)$$



new interference through BC

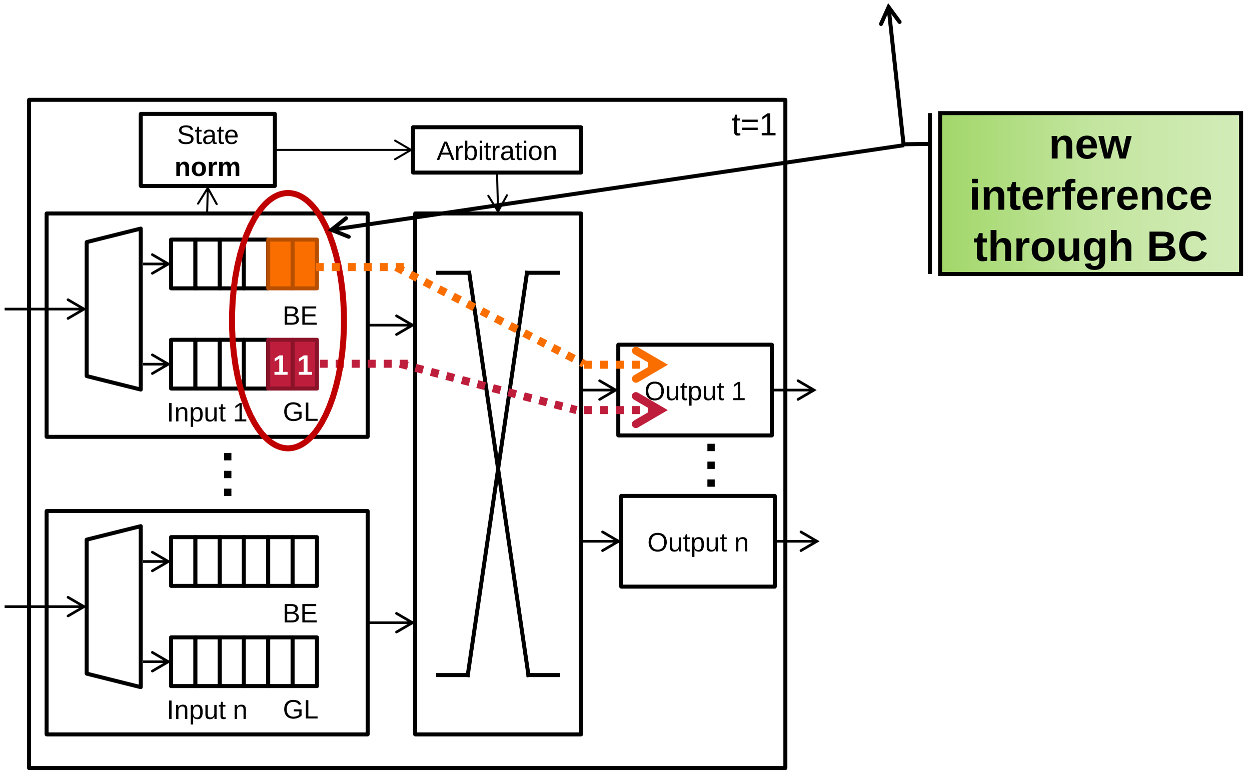
q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q,a_i^q) \leq \mathbf{q * C} + B_i^{out}(B_i^+(q,a_i^q) - C, q) + B_i^{in}(B_i^+(q,a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q,a_i^q) - C)$$

send BE
and
decrement BC



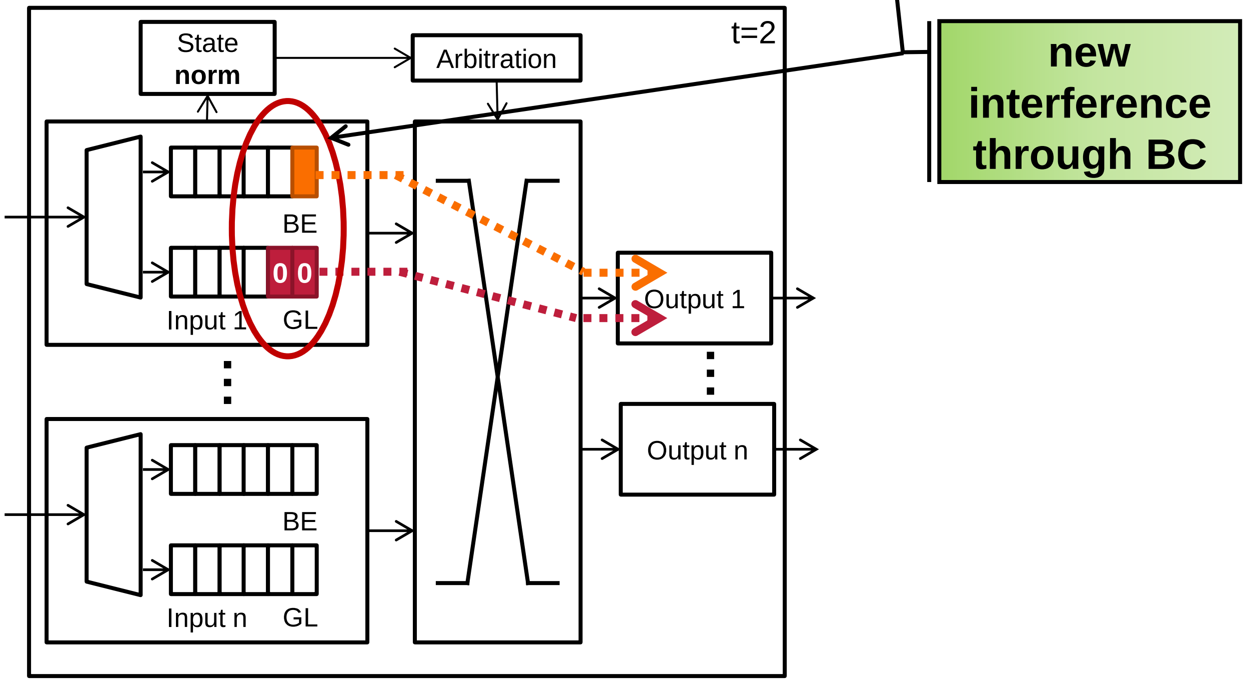
q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q,a_i^q) \leq \mathbf{q * C} + B_i^{out}(B_i^+(q,a_i^q) - C, q) + B_i^{in}(B_i^+(q,a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q,a_i^q) - C)$$

send BE
and
decrement BC



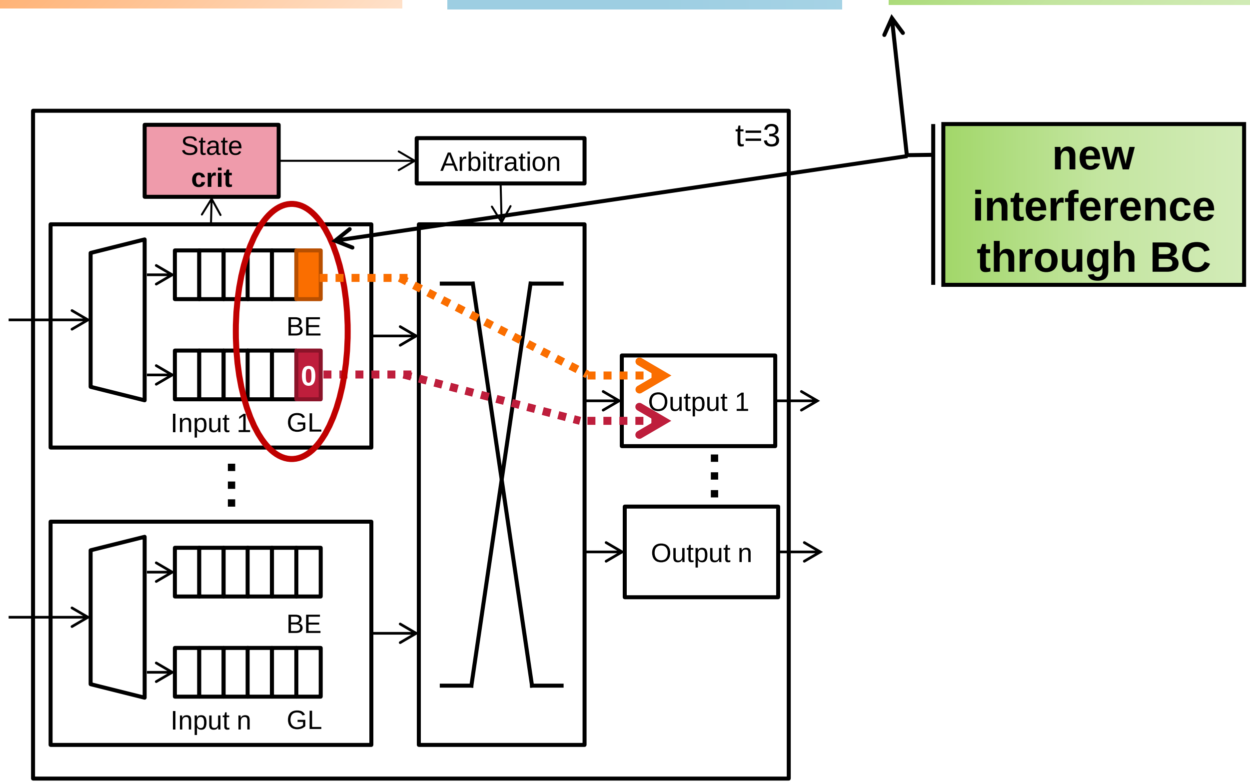
q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Worst-case Multiple Activation Processing Time

$$B_i^+(q, a_i^q) \leq \mathbf{q * C} + B_i^{out}(B_i^+(q, a_i^q) - C, q) + B_i^{in}(B_i^+(q, a_i^q), q, a_i^q) + B_{i,q}^{LP}(B_i^+(q, a_i^q) - C)$$

as BC=0
send GL



q : number of flits
a_i^q : arrival time of event q
C : single flit transmission time

For details and equations look into the paper

Interference through BC

- additional blocking allowed by the blocking counter (BC)
- depends on:
  - higher priority traffic (BE or GL with BC=0)
  - blocking counter
- part of event model propagation

$$\boldsymbol{B^{LP}_{i,q}(\Delta t) \le C * min\left\{\sum_{j\in hp(i)} \eta^{+}_{j}(\Delta t), \chi_i\right\}}$$

number of interfering flits

current value of BC (in flits)

$$\chi_i = \begin{cases} BC^q_i * \hat{n}, \text{if BC counts packets} \\ BC^q_i, \text{otherwise} \end{cases}$$

$\hat{n}$: packet sitze in flits

C : single flit transmission time

$\eta^{+}_{j}(\Delta t)$: maximum number of flits that arrive in Δt

$hp(i)$: set of streams with higher priority than i

For details look into the paper

Outline

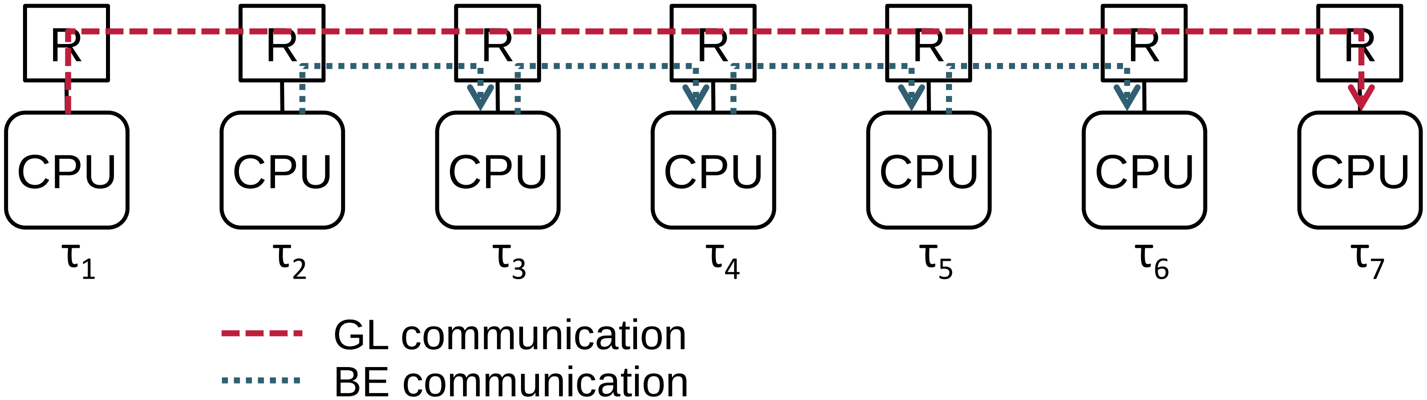
- Motivation
- Providing Quality of Service
- Latency Guarantees
- Experimental Results
- Conclusion

Evaluation

- OMNeT++ framework + HNOCs library
- one VC for GL; 4 VCs for BE
- buffer size: 6 packets
- router with 4 stage pipeline
- packet size: 4 flits
- XY-routing
- BC counting flits
- two sets of experiments:
  - synthetic workload: general properties
  - benchmark based: performance improvement

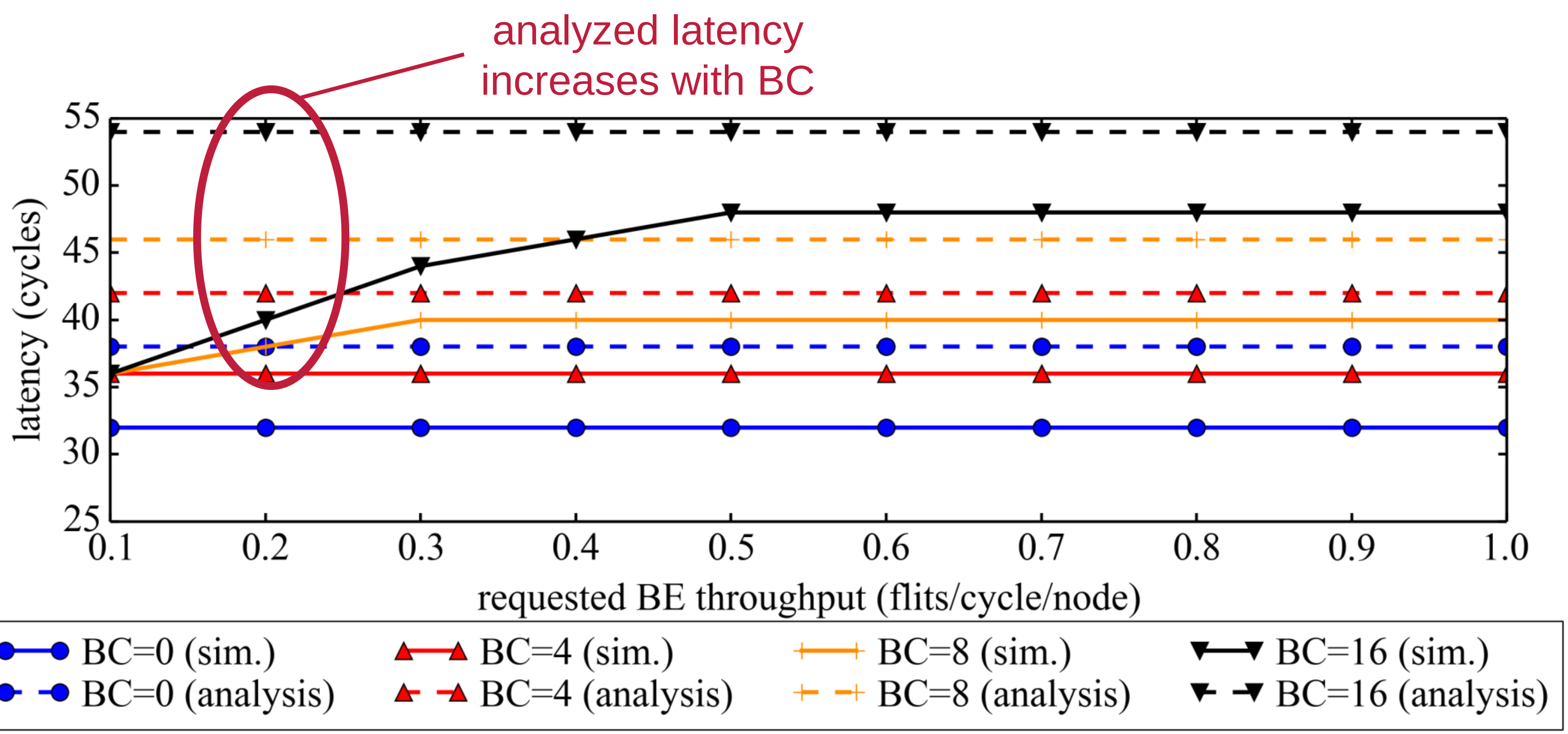
Experiment 1

- synthetic workload, simple line topology
  - periodically injecting packets
  - injection jitter: 25% of period
  - increase load → decrease period
  - one GL stream overlapped by four BE streams
  - different values for BC (note BC=0 → classic prioritization)



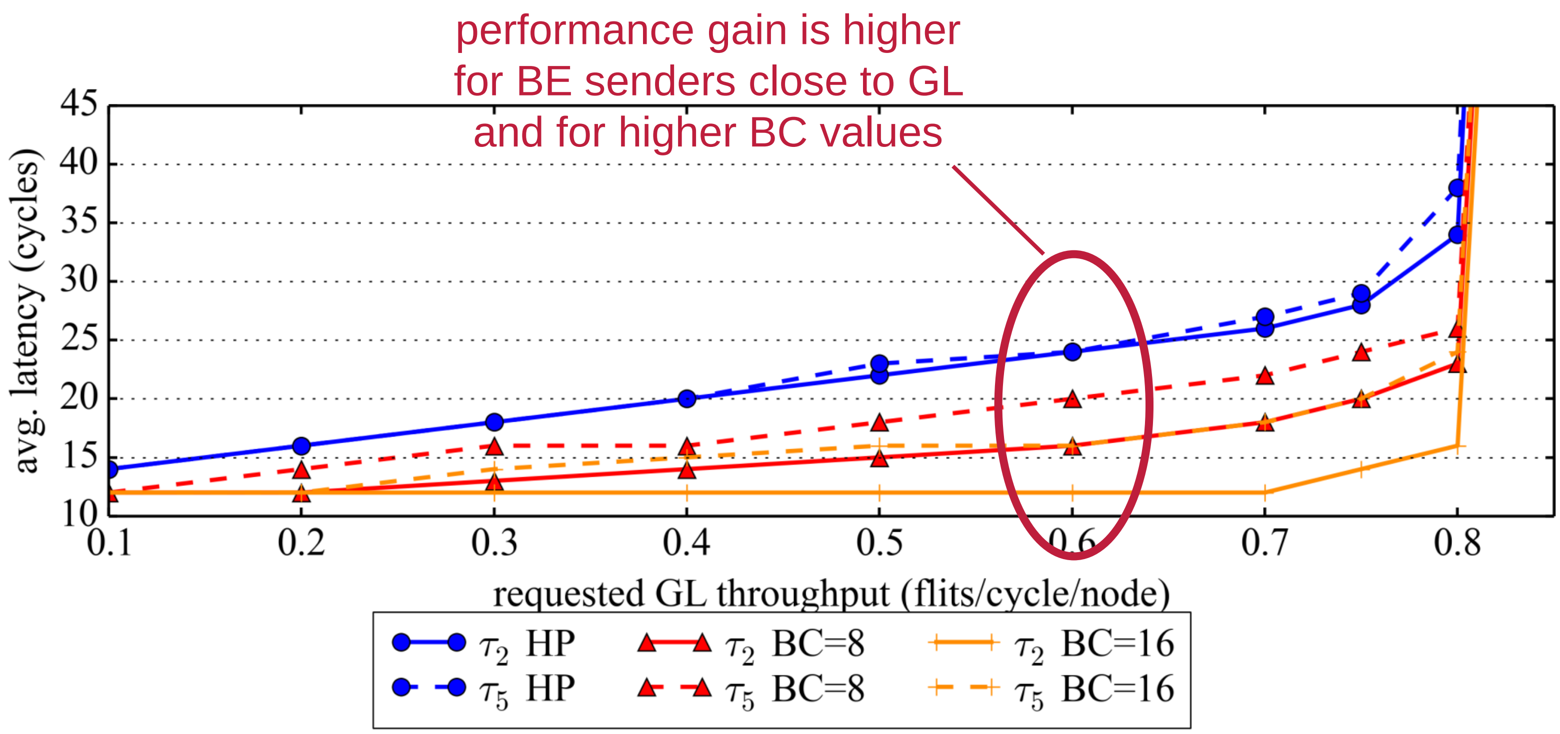
Experiment 1 – GL Latency (τ_1)

- GL load: 0.1 flits/cycle/node *(i.e. 10% link bandwidth)*



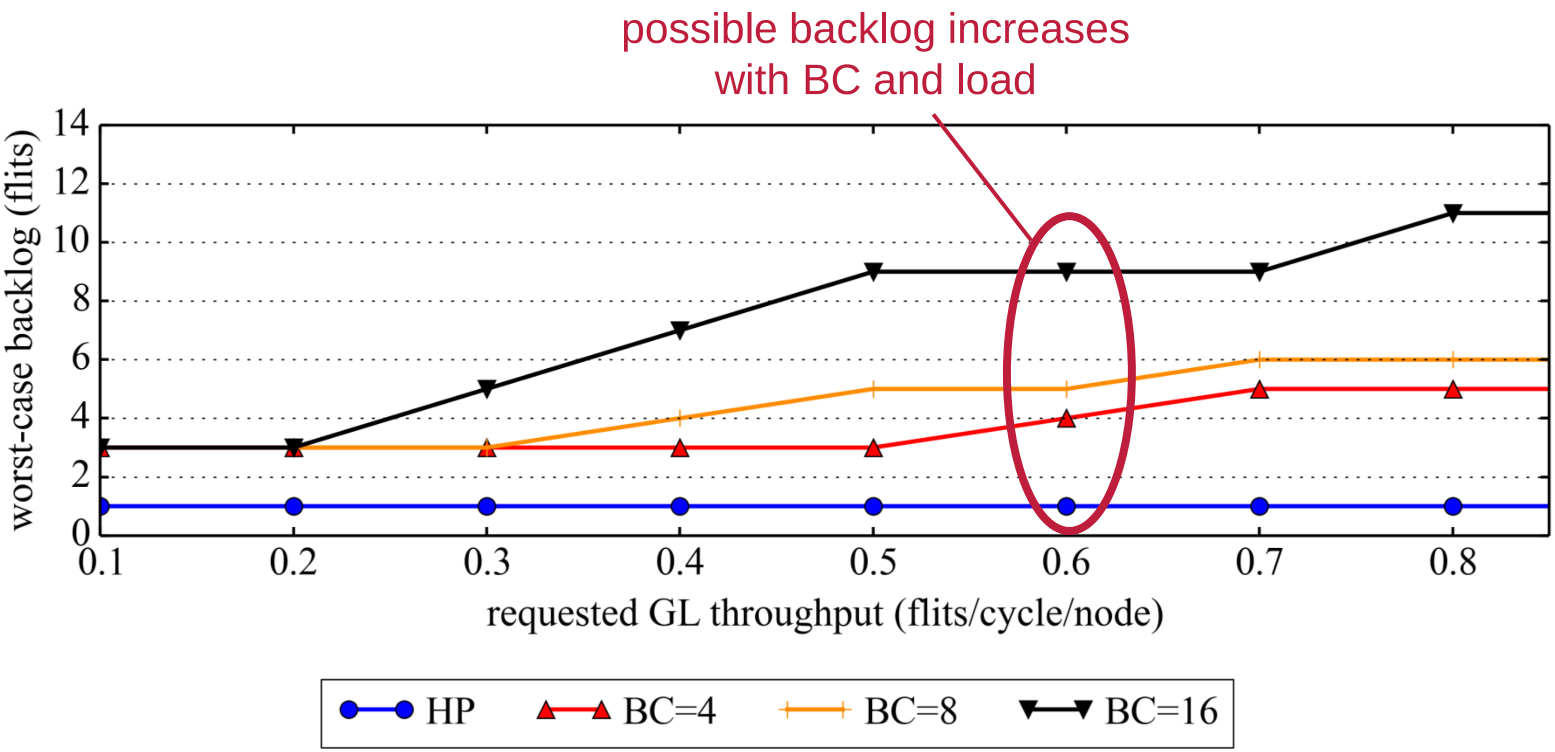
Experiment 1 – BE Latency

- BE load: 0.2 flits/cycle/node *(i.e. 20% link bandwidth)*
- latency of τ_2 (solid) and τ_5 (dashed)



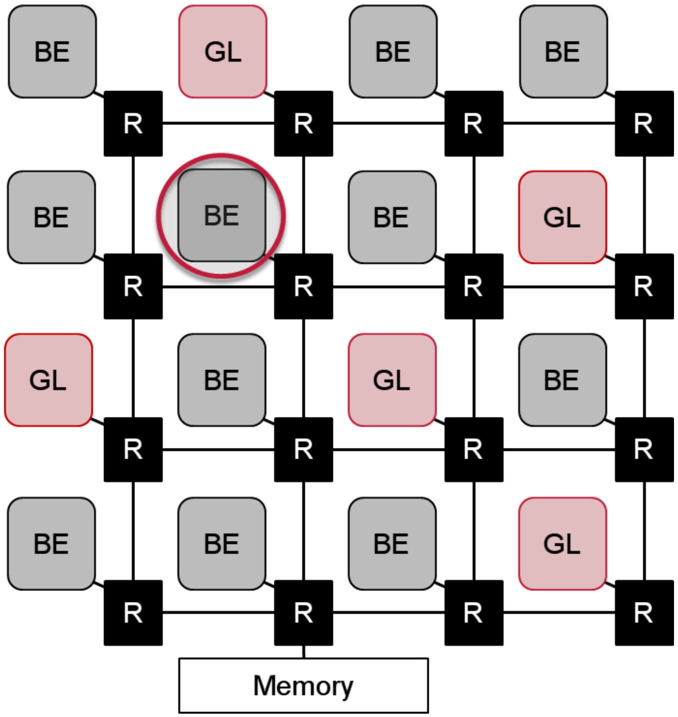
Experiment 1 – GL Backlog (τ_1)

- BE load: 0.2 flits/cycle/node *(i.e. 20% link bandwidth)*



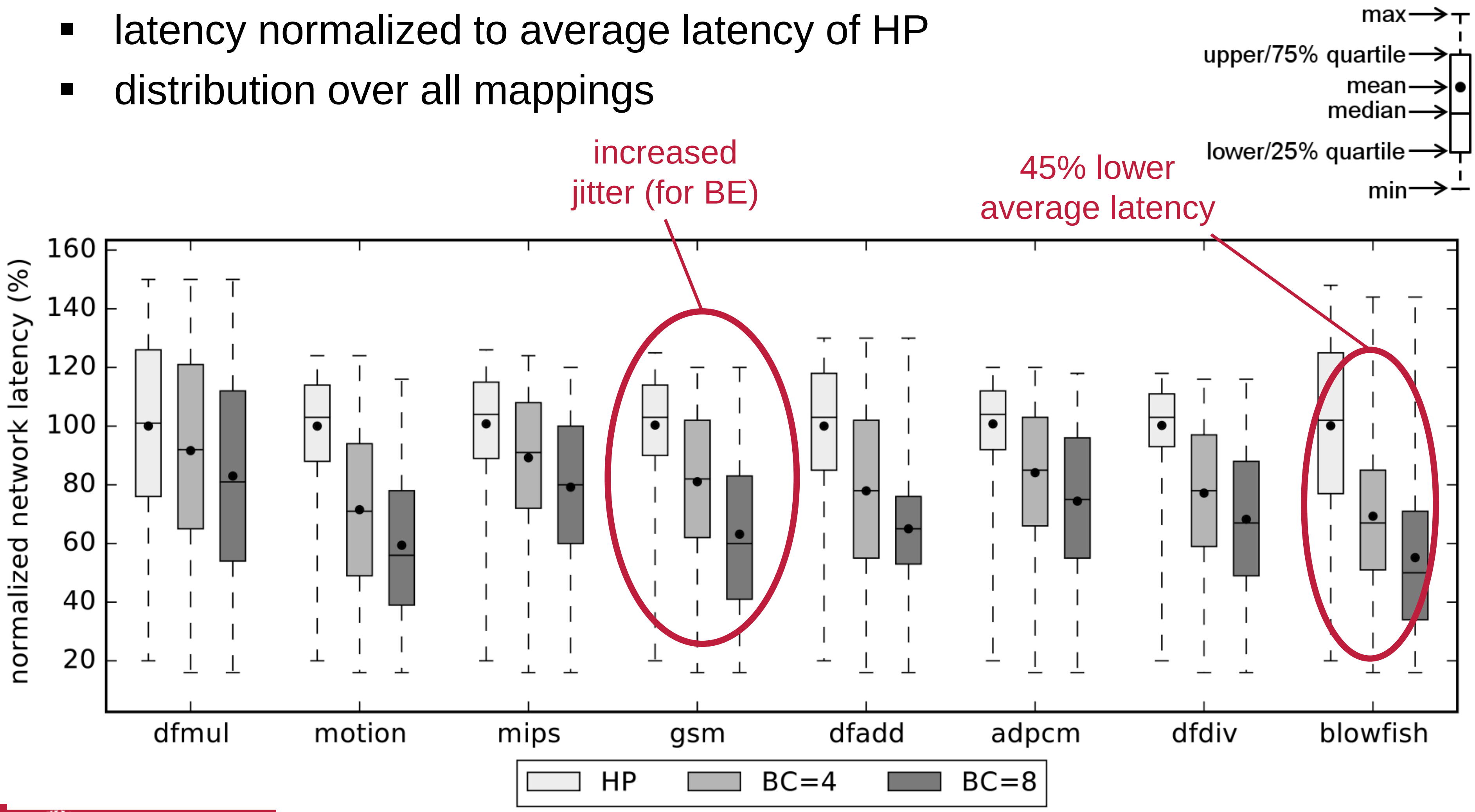
Experiment 2

- benchmark based
  - traces from CHStone
    - extracted using Gem5: ARMv7, 32kB L1
    - accesses to network (e.g. memory access, communication, cache access)
  - random destinations
  - random mappings of interfering load
  - latency for highlighted BE node



Experiment 2 – Random Destinations - BE Latency

- latency normalized to average latency of HP
- distribution over all mappings



Synthesis Results

- 2x2 NoC, 5 VCs, buffer size of 6 packets per VC
- Virtex-6 LX760 FPGA, Xilinx ISE 14.6, standard settings
- 4 approaches
  - baseline: round robin
  - FP: one prioritized VC for GL (RR for requests of the same priority)
  - DP: flag to change priority of GL (i.e. higher or lower than BE)
  - BC: proposed approach (priority change on BC value)

+4.5%

Unit	Baseline	FP	DP	BC
#Registers	9365	9395	9389	9740
#LUTs	12149	12205	12199	12688
Freq. (MHz)	210	210	210	210

+4.0%

Outline

- Motivation
- Providing Quality of Service
- Latency Guarantees
- Experimental Results
- Conclusion

Conclusion

- run-time **configurable, dynamic prioritization of GL** to exploit latency slack of safety-critical applications
  - based on actual blocking through BE
  - prioritize BE over GL when possible

→ **increased performance** for BE

  - up to 45% lower average latency
- increased jitter
- less than 5% hardware overhead (for non optimized solution)
- future work:
  - evaluate different strategies for BC (e.g. limit end-to-end and per router)
  - account for backpressure

Thank you for your attention.
Questions?

References

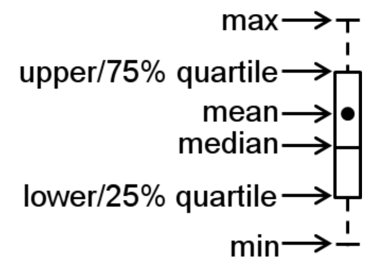
- [**Bjerregaard2005**]T. Bjerregaard and J. Sparsoe, “Scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip,” in Asynchronous Circuits and Systems, 2005. ASYNC 2005. Proceedings. 11th IEEE International Symposium on, pp. 34–43, March 2005.
- [**Bolotin2004**] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, “QNoC: QoS architecture and design process for network on chip,” J. Syst. Archit., vol. 50, pp. 105–128, Feb. 2004.
- [**Burns2014**] A. Burns, J. Harbin, and L. Indrusiak, “A wormhole NoC protocol for mixed criticality systems,” in Real-Time Systems Symposium (RTSS), 2014 IEEE, pp. 184–195, Dec. 2014.
- [**Goossens2010**] K. Goossens and A. Hansson, “The aethereal network on chip after ten years: Goals, evolution, lessons, and future,” in Proceedings of the 47th Design Automation Conference, DAC ’10, (New York, NY, USA), pp. 306–311, ACM, 2010.
- [**Hansson2007**] A. Hansson, M. Coenen, and K. Goossens, “Channel trees: Reducing latency by sharing time slots in time-multiplexed networks on chip,” in CODES+ISSS, pp. 149–154, Sept 2007.
- [**Indrusiak2015**] L. Indrusiak, J. Harbin, and A. Burns, “Average and worst-case latency improvements in mixed-criticality wormhole networks-on-chip,” in Real-Time Systems (ECRTS), 2015 27th Euromicro Conference on, pp. 47–56, July 2015.
- [**Milberg2004**] M. Millberg, E. Nilsson, R. Thid, and A. Jantsch, “Guaranteed bandwidth using looped containers in temporally disjoint networks within the nostrum network on chip,” in Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings, vol. 2, pp. 890–895 Vol.2, Feb 2004.
- [**Panades2006**] I. Miro Panades, A. Greiner, and A. Sheibanyrad, “A low cost network-on-chip with guaranteed service well suited to the gals approach,” in Nano-Networks and Workshops, 2006. NanoNet ’06. 1st International Conference on, pp. 1–5, Sept 2006.
- [**Psarras2015**] A. Psarras, I. Seitanidis, C. Nicopoulos, and G. Dimitrakopoulos, “Phasenoc: Tdm scheduling at the virtual-channel level for efficient network traffic isolation,” in Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE ’15, (San Jose, CA, USA), pp. 1090–1095, EDA Consortium, 2015.
- [**Rambo2015**] E. A. Rambo and R. Ernst, “Worst-case communication time analysis of networks-on-chip with shared virtual channels,” in Proceedings of the 2015 Design, Automation & Test in Europe Conference & Exhibition, DATE ’15, (San Jose, CA, USA), pp. 537–542, EDA Consortium, 2015.

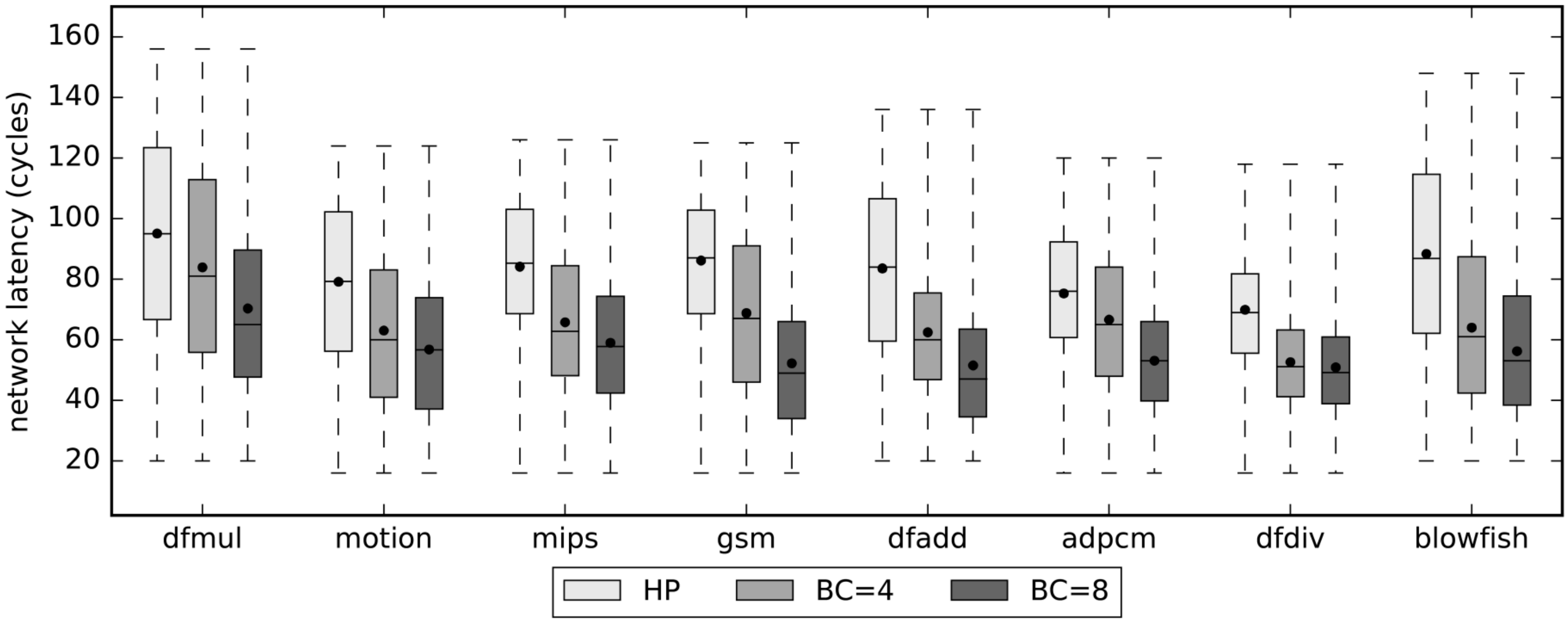
Back Up

Backup Slides

Experiment 2 – Hot Module - BE Latency

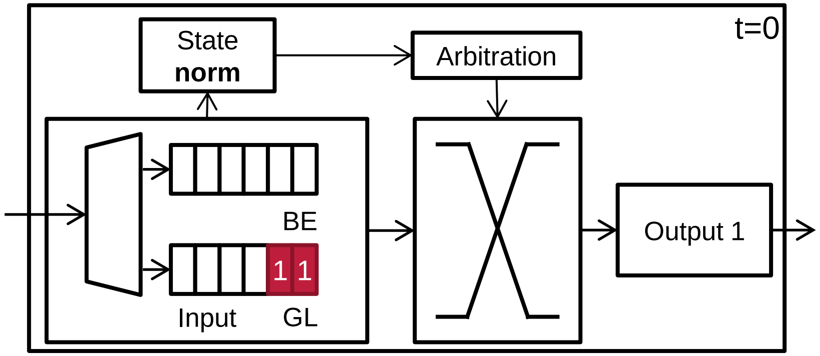
- all nodes sending to memory
- distribution over all mappings





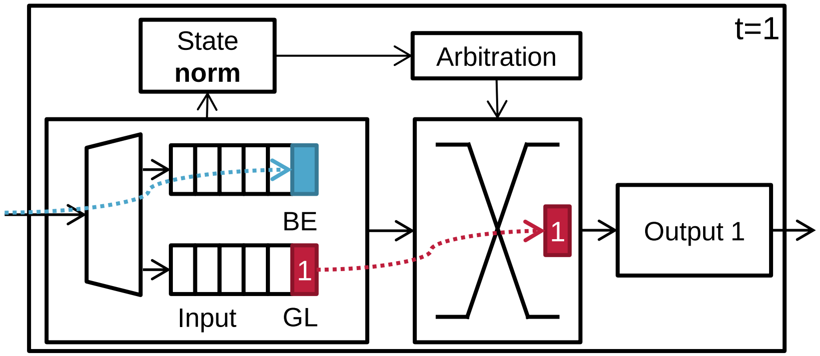
Operational Example (step 0)

- Router in normal state (i.e. BE has high priority)
- Two GL packets (with BC=1) waiting



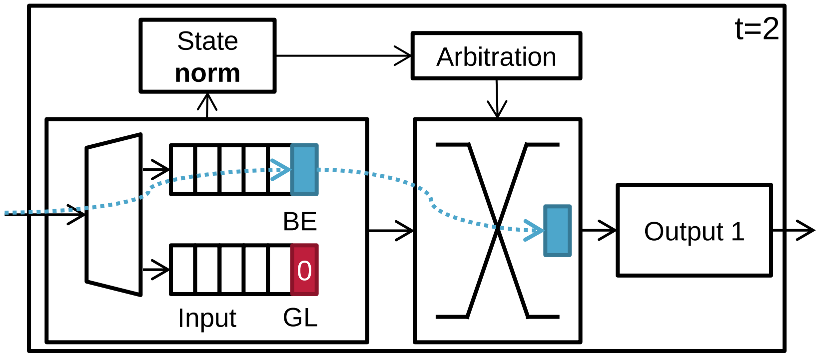
Operational Example (step 1)

- GL packet is sent
- BE packet arrives



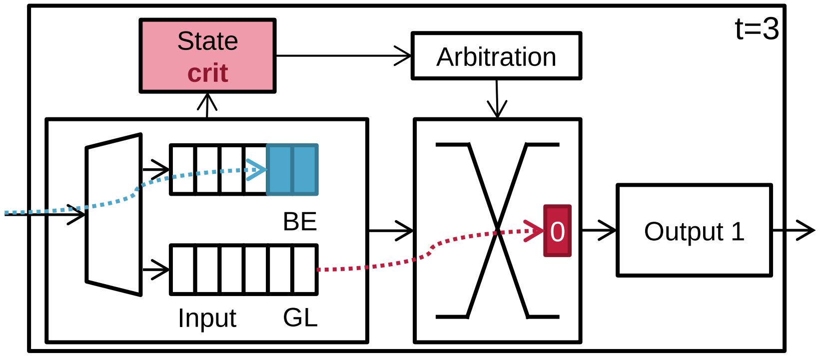
Operational Example (step 2)

- Send BE packet, as GL still allows blocking
  - BC of GL is decremented
- New BE packet arrives



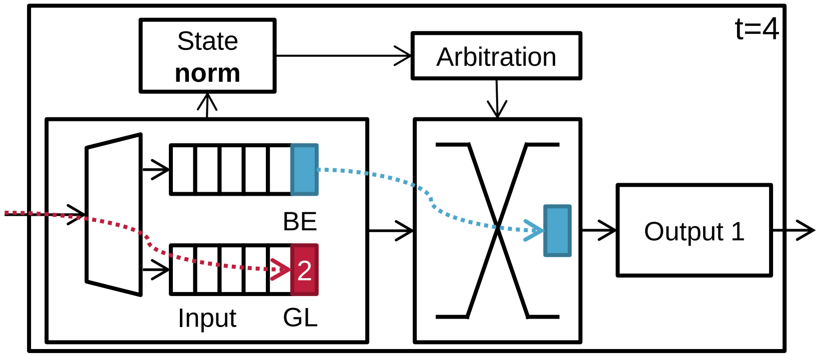
Operational Example (step 3)

- GL achieves higher priority (as BC==0)
  - Send GL packet, BE is blocked
- New BE packet arrives



Operational Example (step 4)

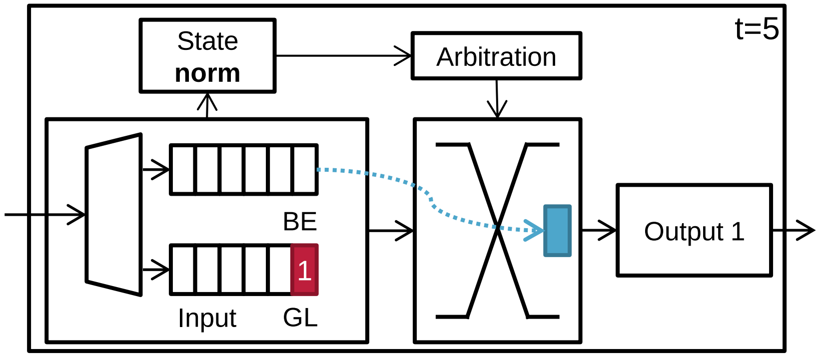
- Send BE packet (as no GL was waiting)
- New GL packet arrives



Operational Example (step 5)

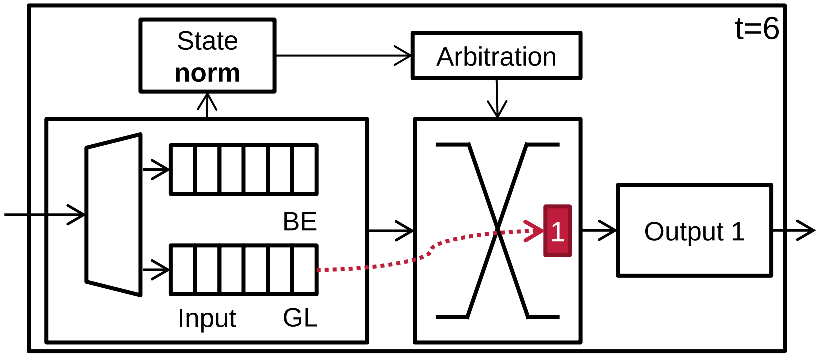
- Send BE packet, as GL still allows blocking
  - BC of GL is decremented

→ BE achieves lower latency



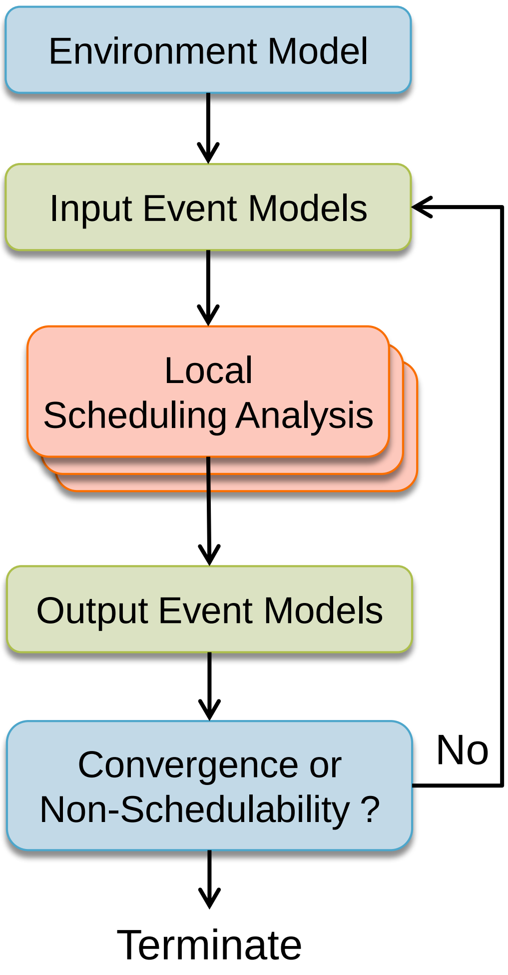
Operational Example (step 6)

- Send GL packet (with BC>0) as no BE is waiting



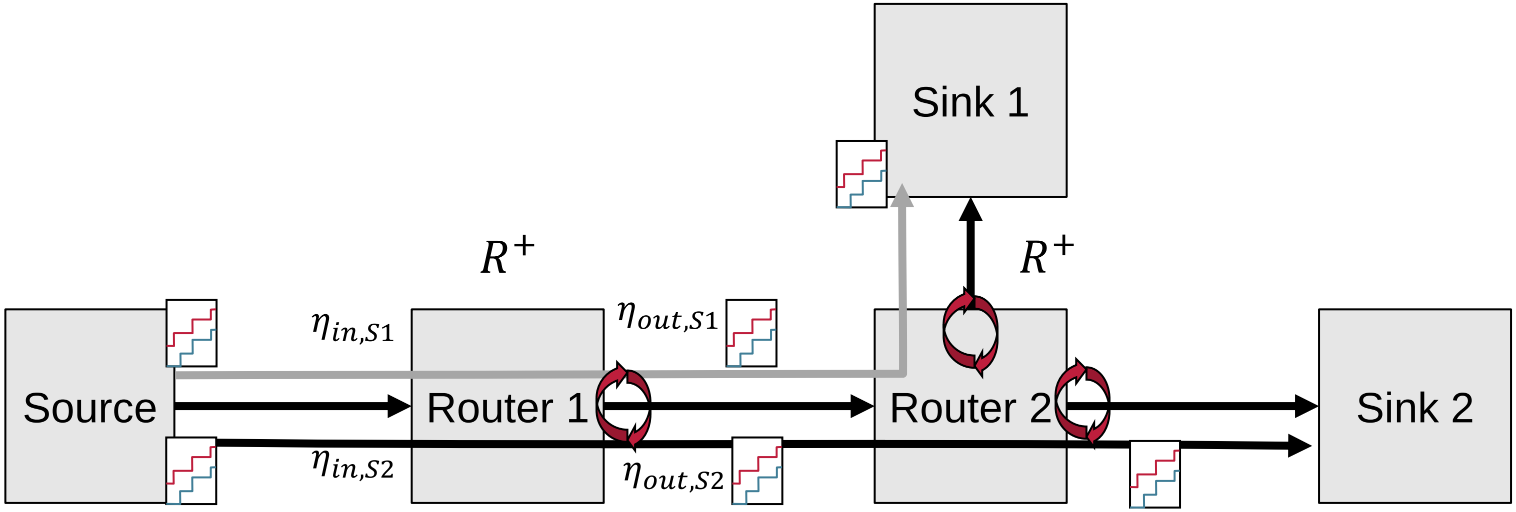
Compositional Performance Analysis for NoCs

- based on analysis from *[Rambo2015]*
- analysis performed iteratively
- **step 1**: **local analysis** (at each router)
  - compute worst-case latency R_i^+ of flits based on critical instant (**busy window**)
  - derive output event models
- **step 2**: global analysis
  - propagate event models downstream
  - go to step 1 if any event model has changed
  - otherwise, terminate



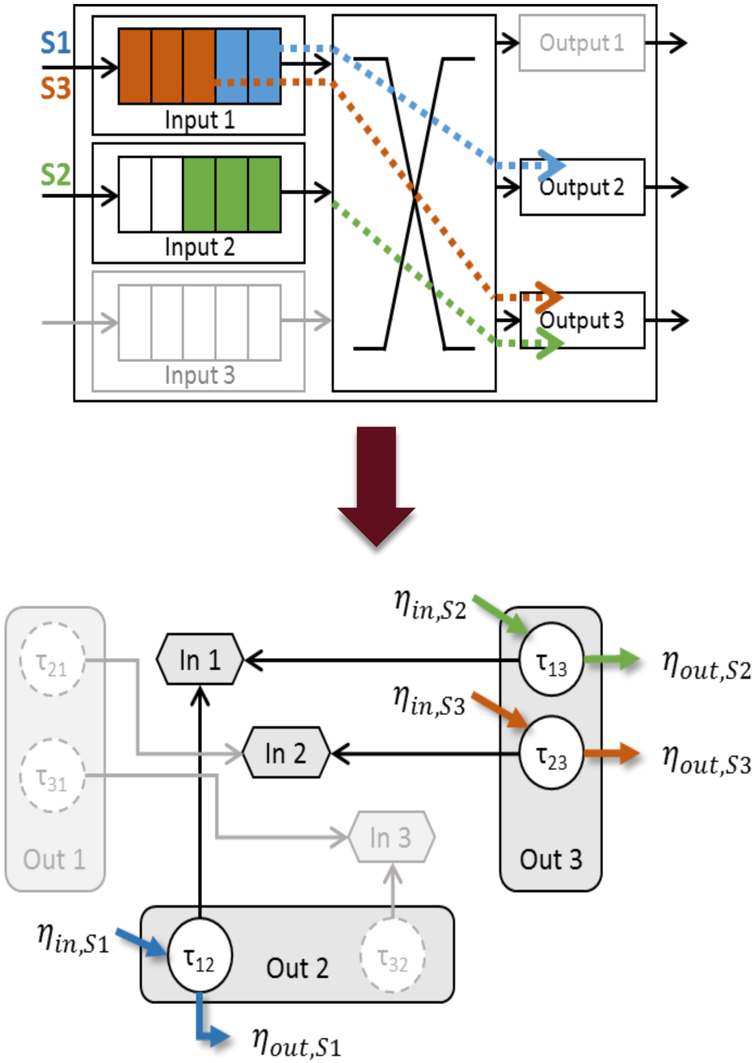
CPA Approach

- worst-case end-to-end latency relies on response times $\boldsymbol{R}^{+}$ from local analyses
- for each stream
  - analyze routers along its path and propagate event models downstream
- formally analyze routers iteratively



Mapping NoC Domain to Processor Resource Model

- output ports → processing resources
- input ports → shared resources with mutually exclusive access
- traffic stream → chain of tasks mapped to resources
- flit transmission → task execution
- flit arrival → task activation
  - input and output event models



Network Latency

- derive single hop latency $\boldsymbol{R}^{+}$ based on
  - multiple activation busy time
  - router's overhead (e.g. time to determine and acquire output port)
- network latency $\boldsymbol{l}^{+}$:
  - sum of single hop latencies on path

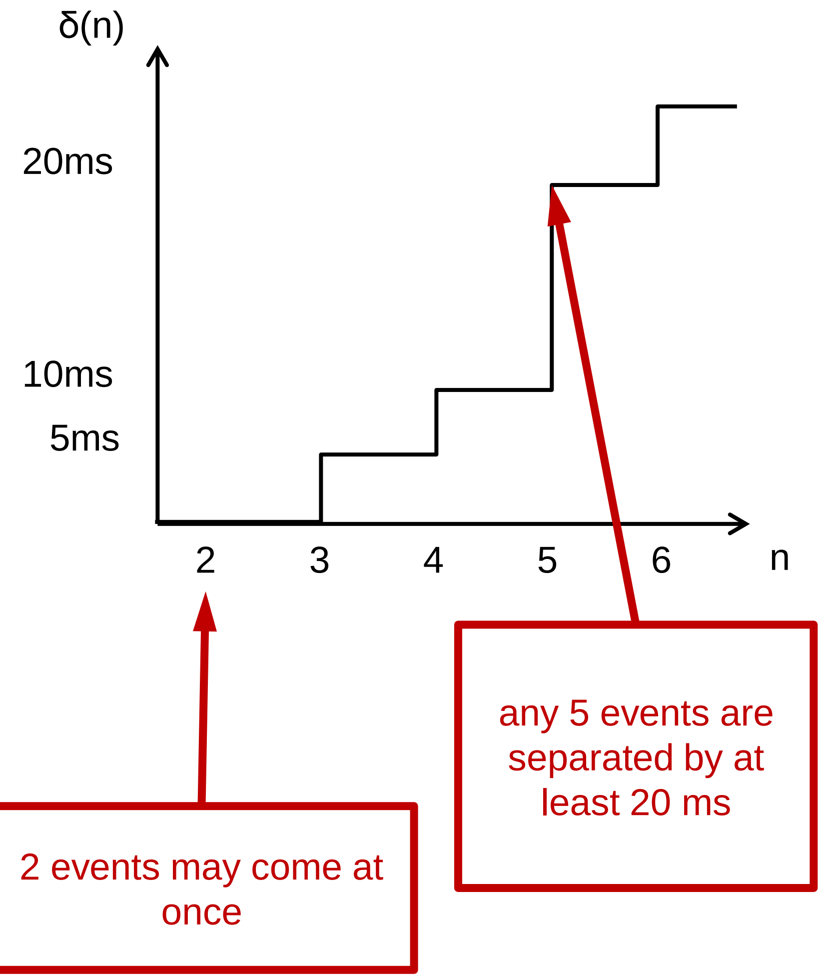
    + injection time (including backpressure at source)

    + de-/packetization overhead

$$l_i^{+}(q) = InjectionTime(q) + PacketizationOverhead + \sum_{j \in Tasks(i)} R_j^{+}$$

Complex Activation Patterns

- variety of activation patterns used in practice
  e.g. periodic + spontaneous, dual cyclic, on change
- timing verification can consider them through use of minimum distance functions
  - i.e. specification of the minimum distance between any n consecutive events
  - derived from specification or rate-limiter



Accounting for BC in Analysis

- extend event model propagation for BC
  - minimum and maximum value for each router on path
- or: test all possible combinations where BC can be consumed on path
  - $\#combinations = \binom{\#hops+BC-1}{BC}$
  - set of possible combinations can be reduced with knowledge on event model propagation
- for each possible combination
  - check for deadline violation