Real-Time Communication Analysis for Networks-on-Chip with Backpressure

Sebastian Tobuschat and Rolf Ernst
Institute of Computer and Network Engineering
Technische Universität Braunschweig, Germany
{tobuschat, ernst}@ida.ing.tu-bs.de

***Abstract*—Networks-on-Chip (NoCs) for safety-critical domains require formal guarantees for the worst-case behavior of all real-time senders. The majority of existing analysis approaches is capable of providing such guarantees only under the assumption that the queues in the routers never overflow, i.e., that no backpressure occurs. This leads to overly pessimistic guarantees or unfulfilled design requirements in many setups using commercially available NoCs where buffer space is limited. Therefore, we propose an alternative analysis methodology providing formal timing guarantees for packet latencies also in a NoC where backpressure occurs. The analysis allows exploiting the behavior of individual traffic streams to determine safe upper bounds on the latency of individual packets. The correctness of the analysis is evaluated experimentally through comparison with simulation results.**

I. Introduction

Networks-on-Chip (NoCs) become increasingly interesting for safety-critical domains due to their performance, power, and size benefits [1]. In such systems, it is necessary to provide formal guarantees that all safety-relevant functions meet their worst-case timing requirements [2]. However, sharing of NoC resources, such as output ports, between concurrently running senders couples their behavior and challenges safety. Therefore, it is crucial to provide methods for the analysis of the worst-case behavior of these systems, which is a known outstanding research problem in NoC design [3].

In this work we focus on approaches based on compositional performance analysis (CPA) [4]. It uses event models to capture and exploit the dynamics of data streams, enabling tight bounds on worst-case latencies. However, existing CPA analysis for NoCs can not handle backpressure [5]. That is, they assume there is always enough free buffer space in the queues of network nodes to accept arriving flits. For this to hold, the approaches calculate the maximum possible backlog of the network and use it as a design input defining the size of the buffers. Alternatively, they allow to iteratively reduce the injected traffic through traffic shaping until the maximum backlog satisfies the available queue size. Unfortunately, in the majority of commercial setups the buffer size can not be changed. Additionally, limiting the injected traffic drastically reduces the platform performance. This is due to the fact, that the shaping bounds provided by the analysis are usually conservative, i.e., adjusted to the worst-case behavior. This raises a need for an CPA analysis capable of handling finite buffer space. To the best of our knowledge, no compositional real-time communication time analysis method has been proposed for best-effort NoCs with a single shared channel and backpressure.

Contribution: In this work, we present a formal worst-case analysis for providing an upper bound on transmission latencies in NoCs with backpressure. We assume a basic NoC without special quality of service mechanisms similar to existing, commercially available architectures [6]–[8]. Nonetheless, our approach can be easily extended to account for architectures with QoS support and virtual channels [5].

The remainder of the paper is structured as follows. In Section II we provide an overview on related work. This is followed by a brief introduction to the CPA approach in Section III. Section IV then provides the new backpressure aware analysis. In Section V we experimentally evaluate our analysis and conclude in Section VI.

II. Related Work

There exist various methodologies for the analysis of networks-on-chip. For example, [9]–[12] provide techniques for timing analysis of priority-based wormhole switching NoCs. In [9] the authors formulate a contention tree that captures interference in the network. Similarly, [10] defines two different delay components: direct interference and indirect interference. Based on these, a worst-case network latency analysis is presented. However, all these schemes do not account for the effects of pipelining and parallel transmission of data. This is tackled in [11], by refining the communication resource and its associated communication task model.

Still, the mentioned approaches assume global and unique priorities with unique virtual channel assignment for each priority. Such implementation policy typically results in high buffer cost and energy overhead. Hence, in many commercially available NoCs the foreseen number of (virtual) channels is usually lower than the number of tasks or priority levels. To address this problem, [12] allows shared priorities in a priority-based NoCs.

In [5], [13]–[15] the authors present worst-case latency analysis approaches for networks without special QoS support and round-robin arbitration. For this, [13], [15] use a recursive calculus to obtain an upper bound on the traversal time of a packet. However, they do not take the individual behavior of streams (e.g. inter-arrival time and periods of packets) into account, resulting in overly pessimistic results. This is addressed in [14], where the authors extend the existing model by integrating the characteristics of the tasks that generate

the packets. Another approach is presented in [5], which uses a compositional performance analysis approach to analyze a NoC with *iSLIP* arbitration and shared virtual channels.

However, the aforementioned approaches assume that network nodes and routers are equipped with sufficient buffer space to prevent backpressure. For this, the network must be adopted to the particular application set, which is hard or even impossible, or the traffic injection-rate must be limited. As the rate limiting must be done according to the worst-case behavior, this can lead to a decrease in system performance as the network can not fully be utilized. To overcome these drawbacks, resent research focused on analysis supporting backpressure [16], [17].

In [16] the authors present an analysis for latency bounds in wormhole networks with finite sized buffers. The approach is based on network calculus [18] and computes global end-to-end service curves for each stream. Based on these performance parameters, as the maximum latency and minimum throughput, can be derived. In [17] the authors present an extension of SLA [11] accounting for backpressure for priority based NoCs. However, the analysis assumes unique priorities for each stream with individual virtual channels for each priority level and focuses on simple periodic activation models. On the contrary, our approach allows arbitrary activation models and sharing of the same (virtual) channel between different applications, as common for commercially available NoC architectures.

III. Compositional Performance Analysis

This section provides an overview on the *compositional performance analysis (CPA)* [4]. The CPA approach uses a similar composition and event models as *Real-Time Calculus (RTC)* [19], but differs in the used local analysis for the links and routers. For this, the *network-on-chip (NoC)* domain is translated to the processor resource model known from real-time scheduling [20].

The CPA model uses a multicore processor to represent the NoC [5]. In the model, *processing resources* represent the output ports of a router and *shared resources* with mutually exclusive access the input ports. The exclusive access models the limitation of an input port to send only one flit at a time to an output port. A traffic stream is modeled as a *chain of tasks* mapped to the resources based on its path in the network. An exemplary mapping of a router with four streams is shown in Fig. 1. In the example, streams 2 and 3, represented by tasks τ_2 and τ_3, share the same input port and thus access the same shared resource InS. Stream 3 additionally shares the same output port with stream 4.

In this model, the arrival of a flit is a task activation at the processing resource and the transmission of a flit at an output port is the execution of that task. Each task τ_i is assigned a best- and worst-case execution time C_i^- and C_i^+ for each task activation. The activations of a task can be triggered by an external source (network interface) or other tasks (routers). These activation events are modeled by minimum and maximum arrival curves $\eta_i^-(\Delta t)$ and $\eta_i^+(\Delta t)$, defining the minimum and maximum number of events for task τ_i that can arrive within any half-open time window $[t, t+\Delta t)$. These functions have

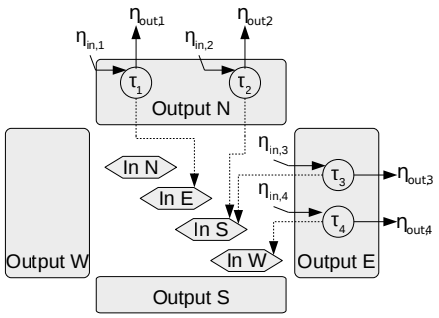

Fig. 1. Four-port router with four traffic streams as a multiprocessor with four processing resources (*Out*) and mutually exclusive shared resources (*In*)

pseudo-inverse counterparts, the so-called distance functions $\delta^+(n)$ and $\delta^-(n)$, which define respectively the maximum and minimum time interval between the first and the last event of any sequence of n consecutive event arrivals. Such an event model covers all possible event arrivals of a task and is not just a specific trace of events.

In CPA, system-level analysis is performed iteratively using individual resource-level analysis steps to obtain the worst-case timing information. For this, CPA performs a local *busy window* analysis for each resource to compute worst-case timings and output event models for each task. The local resource-level analysis uses a *critical instant* scenario that assumes the worst-case arrival of all interfering tasks to obtain the maximum delay for the task under consideration. The output event models from the local analysis are then forwarded as input models for all dependent tasks and resources. With the new input models these tasks are then analyzed again. The local analysis and propagation are iteratively applied until all output event models remain stable [4].

IV. Analysis

In this section we describe analysis of a network-on-chip with backpressure using CPA. To improve readability, we restrict the explanations to a NoC with a single buffer queue per input port (i.e. no virtual channels) shared by multiple streams, round-robin arbitration and the same packet size of n flits for all streams. This is a common setup for existing NoCs [6]–[8]. However, the analysis also directly applies to a system with output buffering, which provides for each input port a buffer queue at the output port, as for example the Kalray MPPA-256 [6]. Additionally, it can be easily extended to handle multiple virtual channels at the input port and different sized packets by adding the new blocking terms and propagated models for backpressure to the analysis in [5]. For the analysis we derive the corresponding worst-case multiple activation processing time of a stream. Based on this, we then derive metrics for a single router and for a complete network (e.g. path latency).

Definition 1. *The worst-case multiple activation processing time* $B_i^+(q, a_i^q)$ *of a stream* i *denotes the maximum time the resource is busy processing* q *flits of stream* i*, given that all but the first flit arrive before their respective predecessor has been transferred and the* q*-th flit arrives at time instant* a_i^q.

To conservatively capture all possible worst-case scenarios, we break down the multiple activation processing time into a sum of different terms addressing different blocking factors. For a router with a single channel per input port, round-robin arbitration on the outputs, and backpressure, the processing time is influenced by:

- **Flit transfer time** C: the time to transfer a flit in a router excluding any kind of blocking. For the sake of simplicity and since it is the usual case in NoCs, we consider that all flits have the same constant transfer time of 1 cycle.
- **Packet size** n: the length of a packet in flits.
- **Buffer size** Q_b: the size of the buffer in the number of flits that can be stored. Without loss of generality, we assume all buffers to have the same size.
- **Output blocking** B_i^{out}: the time streams from other inputs than i use the same output port.
- **FIFO blocking** B_i^{fifo}: the time required to transmit other flits in the FIFO queue preceding the q-th flit of stream i.
- **Backpressure blocking** B_p^{bp}: the blocking resulting from lack of free buffer space at the downstream router using port p.

To upper bound the multiple activation processing time $B_i^+(q,a_i^q)$, i.e., the longest time required to transfer q flits of stream i, we maximize all blocking effects and sum them up:

$$\begin{aligned} B_i^+(q,a_i^q) \leq & q\cdot C + B_i^{out}(B_i^+(q,a_i^q)-C,q) \\ & + B_i^{fifo}(B_i^+(q,a_i^q),q,a_i^q) + B_{P(i)}^{bp}(q), \end{aligned} \tag{1}$$

where $P(i)$ denotes the output port of stream i. The equation forms an integer fixed point problem, which is typical for busy-time based scheduling analysis. It can be resolved iteratively starting with $B_i^+(q,a_i^q) = q\cdot C$.

Next, we derive upper bounds for the individual blocking factors from Eq. 1. For this we first define two auxiliary functions:

Definition 2. *Let $\rho_i^+(\Delta t)$ be the maximum number of flits that can arrive in any time interval Δt at a stream* i *considering whole packets:*

$$\rho_i^+(\Delta t) = \left\lceil \frac{\eta_i^+(\Delta t)}{n} \right\rceil \cdot n \tag{2}$$

Definition 3. *Let Θ^k denote the set of all possible mappings of* k *packets to available output ports (i.e. all but the input port from which these* k *packets are coming). Then $\theta, \theta \in \Theta^k$ defines a specific mapping for* k *packets, such that θ denotes for each of the* k *packets the destined output port. Note, that for an output buffered switch, all packets in a buffer are destined for the same port.*

With these definitions, we can now derive the individual blocking terms.

Lemma 1. *The output blocking B_i^{out} that a stream* i *observes is bounded by:*

$$\begin{aligned} B_i^{out}(\Delta t,q) &= \sum_{j\in Out_i} C\cdot\chi + B_{P(i)}^{bp}(\chi) \\ with \quad \chi &= \min\left\{ \left\lceil \frac{q}{n} \right\rceil \cdot n, \rho_j^+(\Delta t) \right\}, \end{aligned} \tag{3}$$

where Out_i denotes the set of other input ports that are mapped to the same output port as i*. Hence, $j, j \in Out_i$ denotes the cumulative interference input port* j *induces to stream* i*.*

Proof. Due to wormhole switching, once the scheduler grants access to an output port, no other input port can access this port until the port is released, i.e., the packet is fully transmitted. This is captured by ρ_i^+, which considers that after a head flit from j arrives within the time interval Δt, the whole packet will be served before i. Additionally, due to the round-robin arbitration, each head flit belonging to stream i may only be blocked once by each other input port. This is addressed with the *min*-function, where $\lceil \frac{q}{n} \rceil$ is an upper bound on the number of head flits. Each of these head flits can be blocked at most for n flits from each other input port. Moreover, each of the interfering flits then will block stream i for the flit transfer time C and the backpressure blocking these flits experience. □

Lemma 2. *The FIFO blocking B_i^{fifo} that a stream* i *observes is bounded by:*

$$\begin{aligned} B_i^{fifo}(\Delta t,q,a_i^q) &= m\cdot C + \max_{\theta\in\Theta^k}\{A_\theta\} + \max_{\theta\in\Theta^1}\left\{ B_{P(\theta)}^{bp}(m-k\cdot n) \right\} \\ with \quad m &= \min\left\{ Q_b - 1, \sum_{j\in Buf_i} \left\{ \rho_j^+(a_i^q) \right\} \right\} \\ A_\theta &= \sum_{j\in\theta} \left\{ B_j^{out}(\Delta t - C, n) + B_{P(j)}^{bp}(n) \right\}, \end{aligned} \tag{4}$$

where Buf_i denotes the set of all streams sharing the buffer of stream i *(including* i *itself); and* k *denotes the maximum number of whole packets (and hence head flits) of other streams given by $k = \lfloor \frac{m}{n} \rfloor$.*

Proof. The blocking caused by other streams in the same buffer consists of the number of flits that will be transmitted before i and the interference those flits observe. The first term accounts for the transmission of these flits.

Due to backpressure, new flits can only arrive when there is free space. Hence, when the q-th flit of i arrives, there can be at most $Q_b - 1$ flits in the buffer. Additionally, only flits that arrived before the arrival of the q-th flit can be in the buffer, covered by the *min*-term. Thus, m provides the maximum number of flits of other packets that can be before the q-th flit of stream i in the buffer.

The interfering flits can be grouped into flits of whole packets as well as a packet partially transmitted at the front of the queue. The second term accounts for the worst-case blocking whole packets can observe. It considers for the k packets all possible mappings to output ports and takes the maximum blocking. This blocking consists of the output and backpressure blocking each packet will experience. The third term accounts for the blocking of an partial packet. As the header of this packet has already been sent, we only need to account for backpressure blocking. □

Definition 4. *The backpressure blocking B_p^{bp} that a stream* i *observes on its path through output* p *in router* k *is bounded by:*

$$B_p^{bp}(q) = \hat{B}_{p,k+1}^+(q), \tag{5}$$

*where $\hat{B}^+_{p,k+1}$ is the worst-case waiting time at the downstream router (*k+1*) until the* q*-th flit can be received. Note that this accounts for the propagation of the new event model for backpressure in CPA.*

Lemma 3. *The worst-case waiting time $\hat{B}^+_p(q)$ of a port (i.e. buffer)* p *denotes the time until the port is ready to receive the* q*-th flit. For a router it can be bounded by:*

$$\hat{B}^+_p(q) = \begin{cases} q \cdot C + \max_{\theta \in \Theta^k} \{A_\theta\}, & \text{if } b_p > Q_b \\ 0, & \text{otherwise} \end{cases}$$

$$\text{with} \quad A_\theta = \sum_{j \in \theta} \left\{ B^{out}_j(\hat{B}^+_p(q) - C_i, n) + B^{bp}_{P(j)}(n) \right\}, \quad (6)$$

where b_p denotes the worst-case backlog of the port and k *is a bound on the number of packets* q *flits form ($k = \lceil \frac{q}{n} \rceil$). For router ports connected to a resource, the worst-case waiting time can directly be derived from service curve of the resource. This enables to use rate-limited resources that do not allow consuming a flit each cycle.*

Proof. Backpressure (and hence waiting) can only occur, if the worst-case backlog of the port exceeds the buffer size (i.e. $b_p > Q_b$). If backpressure occurs, the port is conservatively assumed to be fully backlogged (i.e. buffer full). Hence, to receive q flits, the port must transmit q flits to any output. For these flits we must account for their transmission time ($q \cdot C$) and the worst-case interference they suffer. For this, the term $\max_{\theta \in \Theta^l} \{A_\theta\}$ obtains the worst-case blocking from each possible mapping of flits to output ports. For this only output and backpressure blocking must be taken into account. □

With lemmas 1 to 3, we have fully derived the inequality in Eq. 1 for the local analysis step. For the iterative approach, we now define derived metrics that are propagated between the routers and can be used for performance and schedulability characterization.

A. Derived Metrics

With the worst-case multiple activation processing time, we now derive the worst-case latency of a single router. The worst-case single hop latency R^+_i of a stream i denotes the maximum time between the arrival time a^q_i of the q-th activation and the processing of q activations $B^+_i(q, a^q_i)$ [21]:

$$R^+_i = \max_{q \in Q_i} \{R_i(q)\} \quad \text{with}$$
$$R_i(q) = \max_{a^q_i \in A^q_i} \left\{ B^+_i(q, a^q_i) - a^q_i \right\} + O_r, \quad (7)$$

where O_r denotes the router's overhead, such as the time required by the router to determine and acquire the output port, and $R_i(q)$ is the worst-case response time of the q-th activation.

This equation considers for each number of q activations within a busy-period all possible arriving times a^q_i of the q-th event. This is necessary, as a later arrival time might increase the interference in the FIFO queues. Additionally, a delayed arrival reduces the response time. However, in [5], [21] the authors proved that the number of possible candidates for a^q_i is finite. The authors showed, that it is sufficient to consider only candidates that coincide with activations of the interfering workload. Hence, we can define the set of candidates as:

$$A^q_i = \bigcup_{j \in I(i)} \left\{ \delta^-_j(k) \mid \delta^-_i(q) \leq \delta^-_j(k) < \delta^-_i \left(\left\lceil \frac{q}{n} \right\rceil \cdot n + 1 \right) \right\}_{k \geq 1} \quad (8)$$

where $I(i)$ defines the set of all interfering streams for stream i using the same queue in the router. Additionally, we need to consider all scenarios where an activation arrives within the busy-period of the previous activation when defining the number of activations q. Thus, we have to find all $q \geq 1$ that are smaller than the maximum number of events q^+_i forming one busy-period [21]:

$$Q_i = \{1, 2, \ldots, q^+_i\} \qquad \text{with}$$
$$q^+_i = \min \left\{ q \in \mathbb{N}^+ \mid \delta^-_i(q+1) \geq \max_{a^q_i \in A^q_i} \left\{ B^+_i(q, a^q_i) \right\} \right\}. \quad (9)$$

Based on the multiple activation processing time, we can also derive the worst-case backlog in each buffer. The worst-case backlog of a port b_p of a router can be defined as:

$$b_p = \sum_{i \in Buf_p} \left\{ \max_{q \in Q_i} \left\{ \max_{a^q_i \in A^q_i} \left\{ \eta^+_i(B^+_i(q, a^q_i)) - q + 1 \right\} \right\} \right\}, \quad (10)$$

where Buf_p denotes the set of streams sharing the buffer p. This equation examines all numbers of activations q which arrive during the processing time of their predecessor. For each, $B^+_i(q, a^q_i)$ yields the completion time of the q-th activation. At this instance of time, at most $\eta^+_i(B^+_i(q, a^q_i))$ activations may have arrived since the start of the busy window, of which $q-1$ have already been processed. Hence, the difference yields the amount of backlogged activations, of which we have to determine the maximum for all $q \in Q_i$ providing the worst-case backlog of a stream. The sum of the worst-case backlogs of all streams sharing a buffer then returns the worst-case buffer backlog. However, due to the backpressure, this backlog only describes an analysis artifact used to estimate whether backpressure occurs. The number of backlogged flits inside a router can never exceed the buffer size.

B. Metrics for the Network

With the analysis of a single router, we can now analyze the whole network using the compositional approach from Section III. For this we iteratively perform a local analysis of the routers and propagate the event models (including backpressure) to neighboring routers. Based on the local analysis, we can define the output events models for each stream that become the input event models of the subsequent routers according to [5] as:

$$\delta^-_{i,out}(q) = \max \left\{ (q-1) \cdot C, \delta^-_{i,in}(q) - (R^+_i - (C + O_r)) \right\} \quad (11)$$

where $(q-1) \cdot C$ denotes the best-case execution time, $\delta^-_{i,in}(q)$ the input event model, and $(R^+_i - (C + O_r))$ denotes the response time jitter (i.e. the difference between worst and best case execution times of a flit).

With the input models of all routers, we can limit the worst-case end-to-end latency $l^+_p(q)$ for transmitting q flits on a

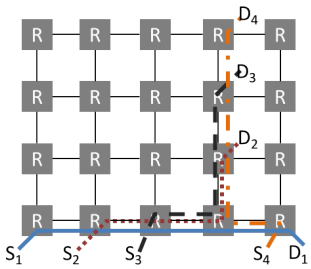

Fig. 2. Setup with four different streams, each sending from source S_x to destination D_x

path p for each stream. It consists of the worst-case response time for each hop on the path p, the time to inject the q flits, and the packetization overhead:

$$l_p^+(q) \leq \max\left\{\delta^-_{First(p)}(q), B_1^{bp}(q+b_i-Q_b)\right\} + O_p + \sum_{j\in Tasks(p)} R_j^+, \quad (12)$$

where $First(p)$ defines the first task of the chain (i.e. network path); $Tasks(p)$ the set of all tasks of path p (i.e. one per hop); O_p the constant de/packetization overhead; R_j^+ the worst-case single hop latency; $\delta^-_{First(p)}(q)$ denotes the time the sender needs to inject q flits (assuming no contention); and $B_1^{bp}(q+b_i-Q_b)$ the overhead induced by backpressure until the q-th flit and all backlogged flits of previous transmissions can be injected to the first router. Basically the equation computes the time interval required by a stream to inject q flits when the sender is fully backlogged and then assumes the last one of these to experience the worst-case blocking on all intermediate routers. Due to the in-order delivery of the network, all previous flits will have arrived at the destination before the last one. And the delay previous flits may observe is included as interference in the worst-case blocking of the last flit.

V. Evaluation

In this section we evaluate and compare our analysis with simulation results. The results were obtained using the OMNeT++ framework [22] and the HNOCS library [23]. We focus on the simple system shown in Fig. 2, which is compact enough to be comprehensively displayed but shows all relevant effects of the analysis. It consists of four streams periodically injecting traffic from source S_x to destination D_x with a packet size of 4 flits. Each router induces a 4-cycle routing overhead (O_r) to the flits and has buffer space for 2 packets. This size is similar to the Kalray MPPA-256 [6] that provides two distinct networks with a buffer size of 8 and 401 flits (i.e. approx. 2.67 and 12.15 packets with a packet size of 3 and 33 flits in the default configuration).

For the first experiment we varied the requested bandwidth of each sender, where all senders request for the same bandwidth. Fig. 3 shows for this, for each of the streams, the analyzed and simulated worst-case flit end-to-end latency. The analysis for all streams for a single configuration took in average 908 ms. As can be seen, the analysis always delivers conservative results. For low bandwidths (up to 10% per sender) the results from analysis are comparatively accurate w.r.t. simulation. For higher loads, the influence of backpressure and head-of-line blocking lead to a higher over-approximation for stream 1 and 2. For streams 3 and 4 the analytic results are much tighter, as these streams compete for a lower number of links, thus experience lower interference. Due to the over-approximation of blocking, the analysis reaches earlier the saturation point. This is the bandwidth at which the backlog at the sender, and hence latency, go to infinity due to the NoC load.

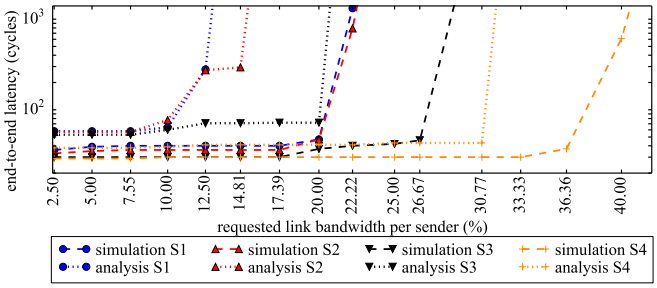

Fig. 3. Flit worst-case latency over requested bandwidth (per sender) with a buffer depth of 8 flits and a packet size of 4 flits

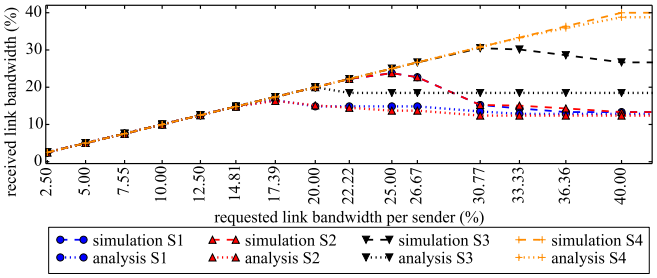

Fig. 4. Received versus requested bandwidth (per sender) for a buffer depth of 8 flits and a packet size of 4 flits

For the same setup, Fig. 4 compares the requested and obtained bandwidth per sender. As long as the analysis provides results for the latency (i.e. before saturation), the analyzed load is similar to the simulated. As soon as the saturation point (for a stream) is reached, the analyzed throughput diverges from the simulation and remains on nearly constant level. Along with the increasing load, the simulated bandwidths start to decrease and converge to the analyzed values, especially visible for streams 1 and 2. This shows, that for systems with a shared channel, the complex blocking scenarios hinder tight latency bounds but nonetheless permit accurate bandwidth estimations.

In Fig. 5 we vary the buffer depth and compare the simulated latency against our analysis and the basic *iSLIP* analysis from [5] that assumes infinite buffers. For this experiment we used a packet size of 4 flits and requested load of 12.5% for each sender. For small buffer sizes, our analysis delivers an high over-approximation of the latency. This is because for small buffers the head-of-line blocking and backpressure occur more likely and propagate faster through the system. For larger buffer sizes, the results of the analysis become tighter, as backpressure is lower or even disappears, diminishing the negative

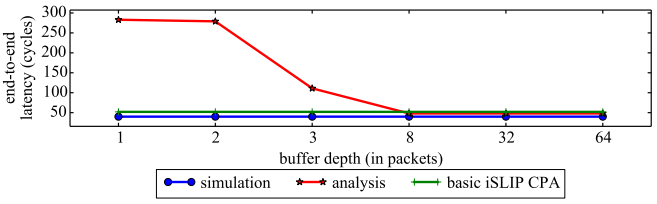

Fig. 5. Flit worst-case latency for stream *S1* for different buffer depths and 12.5% requested bandwidth per sender

effect of blocking propagation. Indeed, for buffer sizes greater than 8 packets, our analysis delivers results as tight as the one from [5]. However, note that the results of [5] are not proved to be safe for this setup, as backpressure can occur. Recall, that approach from [5] is only valid for systems where the backlog is smaller than the buffer size.

The evaluation shows, that our analysis provides safe upper bounds on the worst-case flit latency for a NoC with backpressure. However, it also shows, that backpressure can lead to overly pessimistic guarantees for a system, especially for CPA approaches. This has two reasons. First, the analysis is pessimistic when accounting for the interference. It does not consider correlations or the *pay bursts only once* [18] phenomenon, but applies an over-approximation. This leads to an analytical worst-case that can never happen in the real system. For instance, when sending a flit, the analysis assumes the worst-case backlog to occur during injection (i.e. backlogged sender) followed by the worst-case end-to-end latency for the flit transmission. However, this assumes the worst-case interference to happen twice: when building the backlog and when the flit flows through the NoC.

Second, backpressure constitutes a significant problem in systems with a shared channel. Analyses typically introduce pessimism which will be increased when accounting for backpressure. Hence, for real-time systems, the concurrent access to a shared channel between multiple real-time senders must be avoided (e.g. through a control layer) or the injection rate must be limited. This permits limiting the interference and improve the analysis results as shown in Fig. 3.

VI. Conclusion

In this work, we introduced a real-time communication analysis for best-effort networks-on-chip with finite sized buffers and backpressure. The analysis provides worst-case latency guarantees for individual streams that share a (virtual) channel. In our experimental evaluation we validated the analysis results against simulation. We showed that the CPA framework can be applied for real-world systems where buffer space is limited, such as the Kalray MPPA-256. However, the evaluation also demonstrated that backpressure and blocking propagation can lead to overly pessimistic results, especially for systems with shared buffers. Hence, for future work, we plan to extend the analysis to account for correlations between different blocking terms and the pay bursts only once phenomenon to safely reduce the pessimism.

References

[1] L. Benini and G. D. Micheli, "Networks on chips: a new soc paradigm," *Computer*, vol. 35, pp. 70–78, Jan 2002.

[2] A. Burns and R. Davis, "Mixed criticality systems-a review (7-th ed)," *Department of Computer Science, University of York, Tech. Rep*, January 2016.

[3] R. Marculescu, U. Y. Ogras, L. S. Peh, N. E. Jerger, and Y. Hoskote, "Outstanding research problems in NoC design: System, microarchitecture, and circuit perspectives," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, pp. 3–21, Jan. 2009.

[4] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis - the symta/s approach," *Computers and Digital Techniques, IEE Proceedings* -, vol. 152, pp. 148–166, Mar. 2005.

[5] 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.

[6] M. Harrand and Y. Durand, "Network on chip with quality of service," Dec. 31 2013. US Patent 8,619,622.

[7] Adapteva Inc., *Epiphany Architecture Reference*, Mar 2014.

[8] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J. F. Brown III, and A. Agarwal, "On-chip interconnection architecture of the tile processor," *IEEE Micro*, vol. 27, pp. 15–31, Sept. 2007.

[9] Z. Lu, A. Jantsch, and I. Sander, "Feasibility analysis of messages for on-chip networks using wormhole routing," in *Proceedings of the ASP-DAC 2005. Asia and South Pacific Design Automation Conference, 2005.*, vol. 2, pp. 960–964 Vol. 2, Jan 2005.

[10] Z. Shi and A. Burns, "Real-time communication analysis for on-chip networks with wormhole switching," in *Networks-on-Chip, 2008. NoCS 2008. Second ACM/IEEE International Symposium on*, pp. 161–170, Apr. 2008.

[11] H. Kashif, S. Gholamian, and H. Patel, "SLA: A stage-level latency analysisfor real-time communicationin a pipelined resource model," *IEEE Transactions on Computers*, vol. 64, pp. 1177–1190, April 2015.

[12] Z. Shi and A. Burns, "Improvement of schedulability analysis with a priority share policy in on-chip networks," in *17th International Conference on Real-Time and Network Systems*, pp. 75–84, 2009.

[13] T. Ferrandiz, F. Frances, and C. Fraboul, "A sensitivity analysis of two worst-case delay computation methods for spacewire networks," in *2012 24th Euromicro Conference on Real-Time Systems*, pp. 47–56, July 2012.

[14] D. Dasari, B. Nikoli'c, V. N'elis, and S. M. Petters, "NoC contention analysis using a branch-and-prune algorithm," *ACM Trans. Embed. Comput. Syst.*, vol. 13, pp. 113:1–113:26, Mar. 2014.

[15] D. Rahmati, S. Murali, L. Benini, F. Angiolini, G. D. Micheli, and H. Sarbazi-Azad, "A method for calculating hard qos guarantees for networks-on-chip," in *2009 IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, pp. 579–586, Nov 2009.

[16] Y. Qian, Z. Lu, and W. Dou, "Analysis of worst-case delay bounds for best-effort communication in wormhole networks on chip," in *Networks-on-Chip, 2009. NoCS 2009. 3rd ACM/IEEE International Symposium on*, pp. 44–53, May 2009.

[17] H. Kashif and H. Patel, "Buffer space allocation for real-time priority-aware networks," in *2016 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pp. 1–12, April 2016.

[18] J.-Y. Le Boudec and P. Thiran, *Network Calculus: A Theory of Deterministic Queuing Systems for the Internet*. Berlin, Heidelberg: Springer-Verlag, 2001.

[19] L. Thiele, S. Chakraborty, and M. Naedele, "Real-time calculus for scheduling hard real-time systems," in *Circuits and Systems, 2000. Proceedings. ISCAS 2000 Geneva. The 2000 IEEE International Symposium on*, vol. 4, pp. 101–1044, 2000.

[20] K. W. Tindell, A. Burns, and A. J. Wellings, "An extendible approach for analyzing fixed priority hard real-time tasks," *Real-Time Syst.*, vol. 6, pp. 133–151, Mar. 1994.

[21] J. Diemer, D. Thiele, and R. Ernst, "Formal worst-case timing analysis of Ethernet topologies with strict-priority and AVB switching," in *Industrial Embedded Systems (SIES), 2012 7th IEEE International Symposium on*, pp. 1–10, June 2012.

[22] A. Varga and R. Hornig, "An overview of the omnet++ simulation environment," in *Proceedings of the 1st International Conference on Simulation Tools and Techniques for Communications, Networks and Systems & Workshops*, Simutools '08, (ICST, Brussels, Belgium, Belgium), pp. 60:1–60:10, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2008.

[23] Y. Ben-Itzhak, E. Zahavi, I. Cidon, and A. Kolodny, "Hnocs: Modular open-source simulator for heterogeneous NoCs," in *Embedded Computer Systems (SAMOS), 2012 International Conference on*, pp. 51–57, July 2012.