Design Space Exploration of Many-Core NoCs Based on Queueing-Theoretic Models

Erik Fischer, David Öhmann, Albrecht Fehske, and Gerhard P. Fettweis
Vodafone Chair Mobile Communications Systems
Technische Universität Dresden
Dresden, Germany
Email: {erik.fischer, david.oehmann, albrecht.fehske, fettweis}@ifn.et.tu-dresden.de

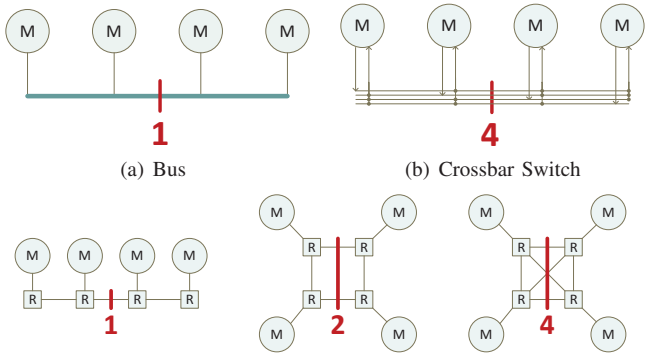
***Abstract*—The design of many-core system-on-chips confronts the developer with a more and more challenging task. Modern embedded applications have a continuously increasing requirement for highly parallelized and flexible heterogeneous processor structures. The interconnection problem becomes a crucial design decision with a growing number of parallel cores. Today, these decisions are usually solely based on the designer's experience. However, this will not be feasible anymore for future many-core systems with thousands of cores on a single chip. Automated guidance and tool support is essential to assist the design of network-on-chips, a common solution for the interconnection of modern system-on-chips. In this paper, we introduce a fast, flexible and accurate analytic model based on queueing theory to analyze the traffic in network-on-chips. The model requires only limited knowledge of the system and is therefore well-suited for the early phase of the design space exploration. It provides a high flexibility in terms of supported topology, routing scheme and traffic pattern, and enables to derive various performance metrics based on the steady-state distribution of the network routers. We evaluate the analytic model against cycle-accurate simulation and demonstrate its application based on some simple design examples, e.g., for buffer dimensioning, localizing bottlenecks, and benchmarking topologies. Several extension of the basic model are proposed to consider finite buffers, dynamic traffic, and to generalize the service time assumptions made for the network routers. This further increases the accuracy of the basic analytic model and expands its application area.**

Keywords-network-on chip; queueing theory; design space exploration; router model; transient behavior

I. Introduction

In recent years, analytic models gain in importance to handle the growing complexity for designing and interconnecting multi-processor system-on-chips (MPSoC). In this paper, we present an extended work of the queueing model for network-on-chip (NoC) based MPSoC introduced in [1].

In embedded computing, todays applications show a common trend towards a continuously increasing computational effort and reliability. This is especially true in the area of multi-media and mobile communication. These requirements can only be fulfilled by massively exploiting parallelism. Emerging technologies like 3D chip stacking [2] promise a significant boost of the number of processor cores per mm^2. The technology allows the vertical stacking of multiple chips, e.g., by using through silicon vias, inductive or capacitive coupling or optical interfaces. It is expected that it will be possible soon to build stacks of hundreds of active layers, i.e., layers with processor elements or memories. By exploiting the third dimension and taking the ongoing technology scaling into account, it is expected that todays MPSoCs soon scale to many-core SoCs with thousands of processors on a single chip [3]. Already in 2015, we may have 1000 or more cores on a chip [4]. This trend becomes already obvious today in the GPU area where existing solutions provide up to 512 parallel cores [5].



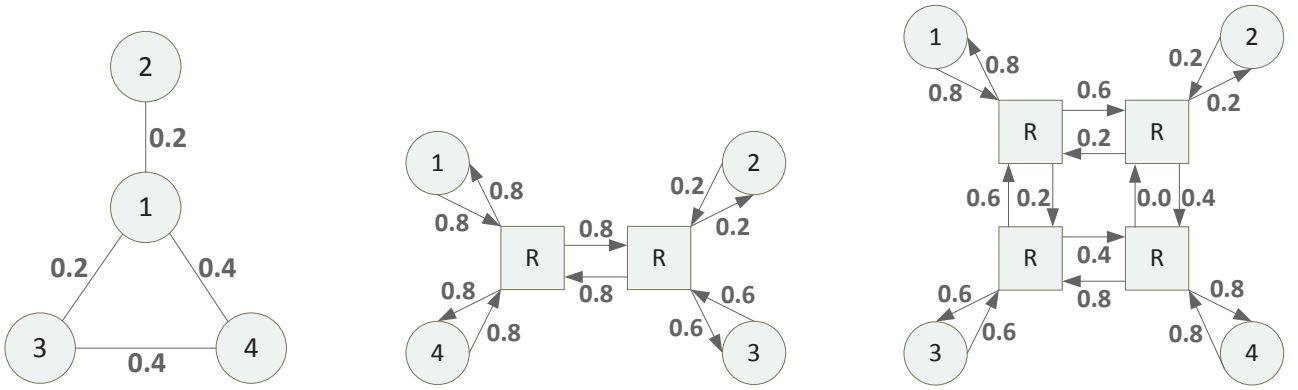
(a) Bus (b) Crossbar Switch

(c) NoC: chain (left), 2D-mesh (middle), fully connected (right)

Figure 1. Different alternatives for the interconnection of an MPSoC with 4 modules (M). (R=router)

Besides the increasing number of cores, we also recognize a trend towards more heterogeneity on-chip. Though heterogeneous multi-processor architectures require a more sophisticated controlling, they enable a better target-oriented computation. I.e., for every computational task a core can be selected which fits best the requirements of the task. Combined with a smart power management concept, this enables building high efficient MPSoCs that offer a high computational performance and low power consumption at the same time. A prototype of a heterogeneous MPSoC has been published in 2008 [6]. The Tomahawk chip consists of six fixed-point vector DSPs and two scalar floating-point DSPs. In addition, an LDPC decoder ASIP, a deblocking filter ASIP and an entropy decoder ASIC is provided. A central control unit (CoreManager) is responsible for the task scheduling and for transferring the data and program code between global and local memories.

If we consider such heterogeneous many-core architectures, the interconnection problem becomes a serious challenge.



(a) Application characterization graph with four modules.

(b) Solution with two modules per router that take module affinity into account.

(c) Solution with one modules per router to increase bisection width.

Figure 2. DSE example with two alternative topologies.

Classical interconnection architectures, such as busses or crossbar switches, cannot offer the necessary flexibility to support the different requirements of the heterogeneous processor cores. Additionally, conventional interconnects offer no good scaling with respect to throughput or area overhead. NoC evolved as a flexible and high-performance solution for the interconnection problem during the last decade [7][8]. NoC is a packet-switched on-chip network where packets are forwarded from a source to a destination via several intermediate router nodes. Each router consists of:

- a small buffer for the intermediate storage of the incoming packets at every input (and/or output),
- a crossbar switch for connecting an input with an output depending on the target address of a packet,
- and a control logic that realizes the routing and arbitration protocol of the NoC.

The routing protocol controls the route that a packet takes from a certain source node to a destination node. The arbitration controls the contention resolution. I.e., it decides which packet is forwarded at first, if two packets arrive simultaneously on different inputs and need to be forwarded to the same output. Routers can be connected in an arbitrary network topology. In addition, one or more processing nodes can be connected to a router. They are called modules. Their functionality is thereby transparent to the NoC, i.e., this could be a processor, memory or an external interface. The smallest transfer unit, to be transmitted over a NoC, is called the flit (flow control digit).

Figure 1 demonstrates the advantage of NoC over conventional solutions for the interconnection. In Figure 1(a), the interconnection of four modules by a bus is depicted. Therein, the red number represents the bisection width of this infrastructure. The bisection width is defined as the minimal number of wires that have to be cut to disassemble the network into two disjoint, equal-sized parts. The minimal cut represents the bottleneck of the network. Therefore, the bisection width can be used as a rough performance indicator for the network throughput. As it can be seen in Figure 1(a), the bus infrastructure is quite limited with respect to achievable throughput, since its bisection width is only 1. This is an issue, if we consider a large number of processors, since the bisection width does not scale with the number of connected modules. In contrast, it can be seen in Figure 1(b) that the crossbar switch offers a very high throughput. The bisection width scales with the number of modules (4 in this case). The drawback of the crossbar switch is that its required chip area grows quadratically with the number of connected modules. In addition, the delay of the switching logic grows linearly with the number of connected modules. Therefore, this interconnection type is not feasible for a large number of modules. NoC is a very flexible solution as depicted in Figure 1(c). Depending on the selected topology (chain, 2D-mesh, or fully connected), the throughput can be adapted according to the application requirements. In this example, the bisection width can be varied between 1 and 4. Moreover, the scalability of the network also depends on the selected topology and is thus adjustable. While the throughput of the chain topology does not scale with the number of connected modules, the fully connected NoC offers a scalability that is equivalent to that of the crossbar switch. The 2D-mesh is an intermediate solution. Finally, NoC allows us to make very flexible design decisions to tradeoff network throughput and latency against chip area, number of wires and achievable clock frequency of the resulting circuit.

However, finding an optimal NoC interconnect for many-core SoCs is a very challenging task, since many different design objectives and constraints have to be considered, like choosing routing and switching methods, selecting topology, application mapping, etc. [9]. This leads to a huge design space. In the following, we discuss a small example to motivate the challenge and the complexity of this process, the so called design space exploration (DSE). Again, we assume that the designed MPSoC shall consist of four modules. The MPSoC is intended for a specific application. The communication requirements between the modules are known in advance, as shown in Figure 2(a). The figure shows the application characterization graph (APCG) [9] for the small example. An edge in the APCG indicates that there is a communication requirement between the two modules. The annotated numbers represent the necessary communication amount. Therein, bidirectional

traffic is assumed with the same communication amount in both direction. From the DSE point of view, the numbers could just be thought of some abstract values that represent the affinity between the modules. More concrete, the numbers could represent the average traffic amount (e.g., flits/cycle). The objective of this DSE example is to find a NoC topology that minimizes the average latency. Additionally, the maximum number of modules per router is constrained to two.

For the given problem, it might be a good heuristic to select two modules with a high affinity, i.e., with a high communication requirement, and assign them to the same router to minimize the path latency. According to Figure 2(a), the highest affinity is between modules 1 and 4, as well as, 3 and 4. Unfortunately, we are only allowed to assign two modules to one router according to our previously defined constraints. Thus, we just select modules 1 and 4. This results in the topology that is depicted in Figure 2(b). As it can be seen, the link between the two routers is quite congested (0.8 flits/cycle). Therefore, it might be a good idea to spend one router per module, instead, to increase the bisection width of the network and relax the congestion on the inter-router links. The resulting topology is shown in Figure 2(c). Though the congestion is reduced on the inter-router link, there is an additional router in the path from module 1 to module 4, which increases the latency of this path. The question is: Which of the two solutions is better and yields the lower average latency? This is hard to answer without making a deeper analysis of the proposed solutions. Thus, the designer might make the wrong decision here so that the resulting MPSoC will not offer the expected performance due to a communication bottleneck.

This small example should motivate why fast and accurate NoC models will be required that give an insight into the system and enable us to reduce the design space already in early design stages. Cycle-accurate simulation based approaches are too slow for this purpose. For the case of many-core SoCs there will be a large number of design alternatives. Moreover, the big number of routers increases the simulation time per run significantly. Simple high-level system models (e.g., only considering the propagation latency and ignoring queueing delays), on the other hand, are able to provide results in very short time. Due to the high abstraction, however, these models loose quite some accuracy. More detailed analytic models provide a good tradeoff between both approaches and are thus well suited for the NoC exploration of a many-core SoC.

Basically, there are two different approaches for analytic NoC models. Models based on the Network Calculus [10] are intended to analyze latency and throughput bounds. Therefore, worst-case assumptions are made for the traffic that the modules inject into the NoC as well as for the service times of the routers to handle a packet. This type of models is well suited to analyze NoCs with respect to quality of service. However, for the purpose of DSE it is necessary to have a more comprehensive view of the system. In general, it is not a good idea to make design decisions only based on the worst-case behavior of the system. A second approach utilizes probabilistic traffic models based on queueing theory [11]. This type of models is much better suited for the purpose of DSE, since it provides more insight into the system, allows to derive distributions (e.g., for the router latency), as well as mean values, and is also suited to yield some information about service guarantees.

A. Overview

In this paper, we propose an analytic NoC model based on queueing theory that provides a high degree of flexibility regarding topology, routing and traffic scheme. In contrast to existing models, it is not restricted to mean value analysis but provides information about the state distribution functions of the routers. It enables us to derive a variety of performance metrics, such as mean latency, buffer usage or overflow probability, and makes the model a very flexible tool for NoC performance analysis. Furthermore, we discuss several extensions of the basic NoC model. They give an even deeper insight into the system and provide more valuable information to the designer or DSE tool to make their design decisions.

- The basic NoC model is limited to the analysis of the system behavior in steady-state. Especially for highly dynamic applications or reconfigurable NoCs, knowledge about the transient behavior of the network is of great interest, cf. [12], [13]. Such an analysis improves the understanding of complex processes and can help to design parameters accurately. Therefore, we present an extension of the basic NoC model that enables us to analyze the system behavior in the time domain and in combination with time-varying rates.
- Making an infinite buffer assumption is a good approach for the basic NoC model. It keeps the computational complexity low and allows to do some analysis on networks in an early DSE design stage where the concrete knowledge of the buffer sizes is still not available. However, the consideration of finite buffers is a valuable extension which makes the model more accurate and allows to model congestion in the network. Moreover, the extension allows to extract new performance metrics, such as blocking probabilities or traffic acceptance rates.
- The assumption of exponentially distributed service times is a good starting point for the development of the basic NoC model. Again, this condition decreases the computational complexity and is a feasible assumption, if the concrete service time distributions are still unknown in an early DSE design stage. Nevertheless, quite accurate approximate solutions are available in literature for estimating the behavior of M/G/1 queueing systems (QS). This allows us to make this generalization without increasing computational complexity.

The remainder of this paper is structured as follows. In Section II, we discuss related work. The necessary mathematical fundamentals of queueing theory are provided in Section III. Section IV introduces the basic analytic NoC model. Starting with the system model and its assumptions (IV-A), the analytic model is discussed on network level (IV-B) and router level (IV-C). We evaluate the accuracy of the proposed approach against cycle-accurate simulation in Section V and provide some DSE application examples. Model extensions are proposed in Section VI for analyzing the transient network

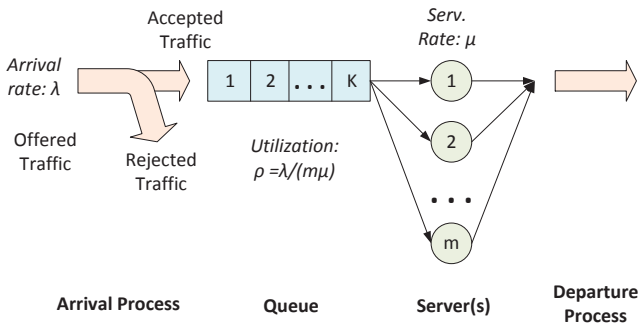


Figure 3. General idea of a queueing system.

behavior (VI-A), considering finite buffers at the router inputs and dealing with arbitrarily distributed service time processes (VI-B). Finally, Section VII concludes the work.

II. State of the art

Much effort has been spent for more than two decades for finding adequate traffic models for the analysis of off-chip and (later) on-chip networks. In 1990, Dally [14] developed analytic tools for investigating latency and throughput in networks, but restricting to k-ary n-cube topologies. Recent approaches focus on the mean value analysis of latency, throughput and energy consumption. Kiasari et al. presented an M/G/1 queueing model for wormhole switched two-dimensional (2D) torus NoC topologies, assuming deterministic routing [15]. This work was extended to G/G/1 queues in [16], which enables the analysis of bursty traffic. Another queueing-theoretic model focusing on buffer allocation was proposed by Hu et al. in [17]. A different approach was published in 2009 in [18], which introduces an empirical model to estimate contention delays for constant service time routers. Thereby, the hybrid router model takes into account Poisson input flows as well as output flows from preceding constant service time routers. Ogras et al. presented a fast and flexible analytic approach in 2010 [19] for the mean value performance analysis of virtual channel first-come first-serve (FCFS) input buffered routers for arbitrary topology and service time distribution. Other recent approaches for modeling on-chip networks [20] focus on the theory of the Network Calculus [10]. This theory provides a powerful tool for an estimation of performance bounds in NoCs, which is essential for giving statements about the realtime capabilities of a network in early design stages. However, for the exploration of the average network behavior, other methods, like stochastic models, are more expedient.

III. Fundamentals of Queueing Theory

For detailed studies of queueing theory in combination with network modeling we refer to [21], [22] and [11]. Subsequently, we give a short overview of the fundamentals. The basic understanding of queueing systems and Markov models is required in Section IV-C to follow the derivation of the router model.

A basic multi-server queueing system (QS) is depicted in Figure 3. A QS models the incoming stream of customers (here: flits) as a stochastic arrival process with a known distribution and mean arrival rate λ. The arrival stream is passed to a queue with a fixed number of K waiting slots. Note that K can also be zero or infinite. If no waiting slot is free upon arrival of the customer, the customer is rejected. Therefore, the accepted traffic that arrives at the queue is in general not equal to the offered traffic at the input of the QS. This is only the case, if an infinite queue is assumed. The customers in the queue are forwarded to m (parallel) servers in a certain order, which is defined by the service discipline of the QS (e.g., FIFO, LIFO). The service (i.e., service time) is again modeled by a stochastic process with a known distribution and mean arrival rate μ. The served customers of the m servers leave the QS as a combined departure process. Its mean rate is equal to that of the accepted traffic. A queueing system can mainly be characterized by four parameters:

- the type of arrival process (A),
- the type of service process (B),
- the number of servers (m),
- and the number of waiting slots (K).

An easy way to describe a QS is provided by the Kendall notation [23]: A/B/m/K. Examples for the arrival and service time process are: exponential/memoryless (M), constant/deterministic (D), Erlang-k-distributed (E_k), general (G). If the number of waiting slots is infinite, K is often omitted. Some examples are: M/M/1, M/D/1, G/G/1/K, M/G/m, etc.

The models presented in this work (Section IV-C) are based on the classical M/M/1 QS, which can be characterized by closed-form expressions. The simplicity of this queueing system arises from the Markov property (also called memoryless property) which is an inherent property of the exponentially distributed arrival and service process. It makes the system independent of past events, i.e., the next arrival/departure time is independent of the last arrival/departure time. As a consequence, the state of an M/M/1 system can fully be characterized by the probability distribution of finding a certain number of customers k in the system which is described by

$$\pi_k = (1 - \rho)\rho^k.$$

Therein, ρ describes the average utilization of the queue, which depends on the relation between arrival rate and service rate and is defined as $\rho = \lambda/\mu$. An extended version of this probability distribution is applied in Section IV-C to derive the steady-state of NoC routers. Based on the distribution various performance indicators can be derived, like the average number of customers:

$$\overline{N} = E[\pi_k] = \sum_{k=0}^{\inf} k\pi_k = \frac{\rho}{1 - \rho}.$$

A fundamental result of the queueing theory, known as Little's law, describes the relation between the average number of customers and the average time spent in the system (T):

$$\overline{N} = \lambda T.$$

This is a general result which is independent of the distribution of the arrival or service process. It enables the analytic

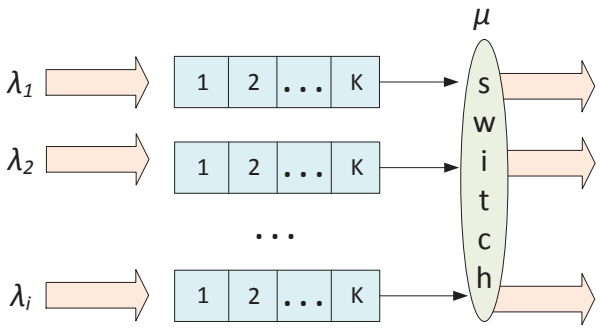


Figure 4. Modeling routers by multiple queueing systems.

computation of mean waiting times (i.e., latencies), which is an important performance measure for NoC, i.e.,

$$T = \frac{1/\mu}{1-\rho}.$$

Furthermore, it is possible to derive the tail probability based on the distribution of the number of customers. It represents the probability that the number of customers in the system exceeds a given capacity K:

$$P[k > K] = \sum_{k=K+1}^{\inf} \pi_k = \rho^{K+1}.$$

This is an interesting performance measure which can be employed for the task of buffer dimensioning (see Section V) and to provide certain service guarantees (e.g., maximum latencies).

The preceding expressions refer to a steady-state where transients have faded away and the system has reached stationarity. Our basic NoC model (Section IV) assumes steady-state which is sufficient for most analyses. However, for some purposes, e.g., analysis of startup behavior or time-varying traffic rates, it might be attractive to analyze transient characteristics as well. Therefore, we extend our basic model in Section VI-A. Unfortunately, analytical expressions for the transient behavior are often cumbersome or even hard to find [11]. In this case, numerical techniques can be applied (see Section VI-A).

Figure 4 depicts how the QS approach can be employed to model a single NoC router. Therein, every input is modeled by a separate QS. This is a natural mapping, since we assume indeed that every input has a separate buffer. The customer arrival stream is mapped to the stream of incoming flits with known mean arrival rates $\lambda_1, ..., \lambda_i$. Every input queue is served by a single server: the switch. Actually, all inputs are served by the same server. As mentioned before, an arbiter resolves contention cases. However, the contention resolution has a significant influence on the mean service rate for every input. For this purpose, a service time model has to be employed that decouples the input queues first under consideration of contention and arbitration policy. In [24], a service time estimation for round-robin arbitration has been proposed.

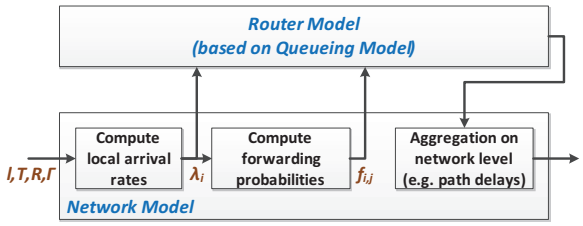


Figure 5. The Hierarchical structure of the proposed analytic model.

IV. Basic NoC Model

Before we can start to introduce the analytic model, we need to define the system model and model assumptions first (Section IV-A). Some common restrictions are made to reduce the model complexity and keep it practicable. Still, we focus on preserve a high flexibility to serve a broad spectrum of applications. The basic NoC model is introduced in two steps, on network level (Section IV-B) and on router level (Section IV-C).

A. System Model

We assume the routers to be arranged in an arbitrary topology. An arbitrary number of cores is allowed to be connected to a single router. Due to the low buffer requirements, wormhole switching is the most favored switching technique for realizing best-effort services in on-chip networks today [9]. Therefore, we restrict our model to this technique. The routing protocol, on the other hand, shall not be restricted. Concerning the arbitration scheme, we restrict to the first-come first-serve method. Extensions to other arbitration schemes, like the popular round-robin, are feasible, as shown in [24]. Routers consist of an arbitrary number of buffered input ports and an arbitrary number of (unbuffered) output ports. In this section, we assume infinite buffer size.

Furthermore, we assume external packet arrivals from modules to possess Poisson characteristic [11], i.e., they have exponentially distributed inter-arrival times with known mean values. This assumption is often made to approximate real network traffic while reducing the model complexity at the same time. The router service times include processing delay for arbitration as well as forwarding delay for the packet and are assumed to be exponentially distributed. Furthermore, knowledge of the mean router service rate and router service latency is required. We assume it w.l.o.g. to be equal for all routers in the network. Finally, we imply a common clock for all routers.

To provide a flexible as well as a fast analytic model we propose to follow a hierarchical approach as depicted in Figure 5. We split the NoC model into an analysis on network level and on router level. By performing the analysis on router level and combining the results on network level, we thus reduce the complexity.

The network model receives multiple inputs that have to be specified by the user. The traffic scenario is described by the *traffic characterization matrix* $\boldsymbol{T}$ and the *external arrival rate vector* $\boldsymbol{l}$. The topology and interconnection are specified via the *connectivity matrix* $\boldsymbol{\Gamma}$. Finally, information about the

TABLE I: Model parameters and notation.

N_M	Number of modules
N_R	Number of router nodes
N_E	Number of edges
$\boldsymbol{T} = [t_{s,d}]$	Traffic characterization matrix (of size $N_M \times N_M$) with elements $t_{s,d}$ that specify the send probability from module s to module d
$\boldsymbol{l} = [l_s]$	External arrival rate vector (of size $N_M \times 1$) with elements l_s representing the arrival rate (packets/cycle) from source module s
$\boldsymbol{\Gamma} = [\gamma_{s,d}]$	Connectivity matrix (of size $(N_M + N_R) \times (N_M + N_R)$) with elements $\gamma_{s,d}$; $\gamma_{s,d} > 0$, if there is a directed connection from s to d; the value $\gamma_{s,d}$ represents the link ID for this connection (sgn($\boldsymbol{\Gamma}$) ≡topology matrix)
$\boldsymbol{R} = [r_{s,d,i}]$	Routing matrix (of size $N_M \times N_M \times N_E$) with elements $r_{s,d,i}$ defines the probability that link i is occupied for routing a packet from source module s to target module t ($\sum_{\forall i} r_{s,d,i} = 1$)
$\bar{x}$	Average router service time

applied routing scheme is provided via the *routing matrix* $\boldsymbol{R}$. An overview of the notation and a more detailed explanation is given in Table I.

Based on this information, the network model is able to compute local parameters for each router node individually, i.e., the inputs for the router model. The local parameters comprise the local arrival rates λ_i that is the accumulated arrival rate over all traffic flows that cross router input i. Furthermore, the forwarding probabilities $f_{i,j}$ are computed. $f_{i,j}$ defines the probability that a packet arriving at a router input i is forwarded to a router output j (please note that the indices i and j correspond to the unique identifier of the link that is connected to router input or output). The computation of the local arrival rates and forwarding probabilities is discussed in more detail in Section IV-B.

The local parameters can now be applied to a queueing model on router level. It is responsible for deriving the compound distribution for the number of packets in the input queues, which represent the router state. Consequently, the knowledge of the compound distribution enables a computation of key performance indicators, such as average buffer usage, overflow probabilities or mean queueing delays. The queueing model on router level is introduced in Section IV-C.

Finally, the performance metrics, computed on router level, have to be combined on network level, e.g., to derive path delays by summing up the queueing delays and the fix router propagation latencies.

B. Network Model

We can derive the vector of local arrival rates $\boldsymbol{\lambda}$, with elements λ_i ($1 \le i \le N_E$), by summing up all traffic flows that cross a specific link (and router input queue, respectively). Therein, N_E is the number of links in the network. The traffic characterization matrix $\boldsymbol{T}$ provides information about a pairwise traffic flow probability between each module s and d. By weighting $\boldsymbol{T}$ with the external arrival rates $\boldsymbol{l}$, we get the traffic intensities (in packets/cycle) for each pair of modules. Finally, we multiply the traffic intensities with the probability that the flow will pass link i (given by routing matrix $\boldsymbol{R}$) and sum up the fractions of the contributing traffic flows:

$$\lambda_i = \sum_{s=1}^{N_M} \sum_{d=1}^{N_M} l_s \cdot t_{s,d} \cdot r_{s,d,i}, 1 \le i \le N_E. \quad (1)$$

The notation is given in Table I. By applying the definition of the Frobenius inner product [25], we can rewrite (1) as matrix equation as follows:

$$\lambda_i = \mathrm{tr}\left((\boldsymbol{L} \cdot \boldsymbol{T})^{\mathrm{T}} \boldsymbol{R_i}\right). \quad (2)$$

In (2), tr(•) represents the trace of the matrix, $\boldsymbol{L}$ is the $N_M \times N_M$ diagonal matrix representation of vector $\boldsymbol{l}$:

$$\boldsymbol{L} := \mathrm{diag}(\boldsymbol{l}),$$

and $\boldsymbol{R_i}$ the corresponding submatrix of $\boldsymbol{R}$ that consists of all elements $r_{s,d,i}$ with $1 \le s, d \le N_M$. We can select the set of local arrival rates Λ^r for a single router node r by exploiting the knowledge of the topology that is contained in the connectivity matrix $\boldsymbol{\Gamma}$. I.e., we collect all λ_i where i is the ID of an input edge of router r:

$$\Lambda^r := \left\{\lambda_{\gamma_{s,r}} \mid \gamma_{s,r} \neq 0; s \in \{1, \ldots, N_M + N_R\}\right\}. \quad (3)$$

We continue to compute the forwarding probability matrix $\boldsymbol{F}$. The matrix element $f_{i,j}$ ($1 \le i, j \le N_E$) can be defined as traffic intensity between router input i and router output j normalized to the total arrival rate at input i, i.e., λ_i:

$$f_{i,j} := \frac{\sum_{s=1}^{N_M} \sum_{d=1}^{N_M} l_s \cdot t_{s,d} \cdot r_{s,d,i} \cdot r_{s,d,j} \cdot \delta_{i,j}}{\lambda_i}, 1 \le i, j \le N_E. \quad (4)$$

We call the term $\delta_{i,j}$ the *link selector matrix*. It ensures that there is only a forwarding probability $f_{i,j} > 0$, if (i, j) represents an input/output link pair of the same router:

$$\delta_{i,j} := \begin{cases} 1, & if\ \exists s, r, d\ with\ \gamma_{s,r} = i\ \wedge \gamma_{r,d} = j \\ 0, & otherwise \end{cases}.$$

Therein, $\gamma_{s,r}$ and $\gamma_{r,d}$ are corresponding elements of the connectivity matrix $\boldsymbol{\Gamma}$. Equation (4) can be rewritten in matrix form:

$$f_{i,j} := \frac{\delta_{i,j}}{\lambda_i} \cdot \mathrm{tr}\left((\boldsymbol{L} \cdot \boldsymbol{T})^{\mathrm{T}} \left(\boldsymbol{R_i} \circ \boldsymbol{R_j}\right)\right), \quad (5)$$

where $\circ$ represents the entry-wise multiplication (i.e., the Hadamard product) of two matrices. Finally, we also restrict the set of forwarding probabilities F^r to a single router node r, similar to the approach in (3), and come to (6):

$$F^r := \left\{f_{\gamma_{s,r},\gamma_{r,d}} \mid \gamma_{s,r}, \gamma_{r,d} \neq 0; s, r \in \{1, \ldots, N_M + N_R\}\right\}. \quad (6)$$

C. Router Model

Based on the assumptions that we made in Section IV-A, an M/M/1 queueing system [11] with exponential interarrival and service times will be appropriate to model the router behavior. However, in reality, the traffic situation within a router looks more complicated, as the example in Figure 6a) shows.

Therein, we find splitting and merging of traffic flows that contend with other input queues for multiple output ports. Furthermore, each input has different probabilities of being

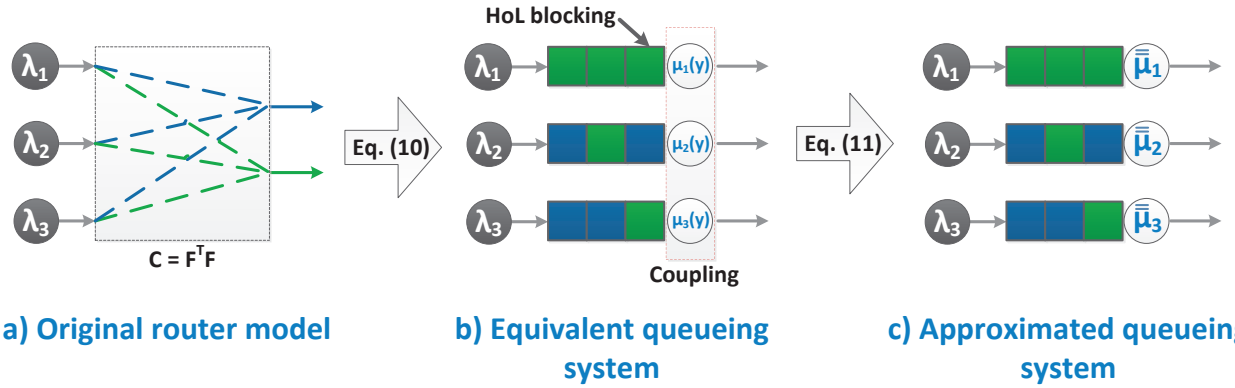


Figure 6. The original router model is transformed to an equivalent queueing model where the service rates of the input queues are mutually coupled. As a second step, an approximation by state aggregation is applied to decouple the queues.

forwarded to a specific output. To be able to represent the router system by a queueing model, we propose using a simplified equivalent system, as depicted in Figure 6b). The idea is to include the contention delays into the service times and thereby receiving port specific service times (or service rates, respectively). In fact, if a packet in front of a (FIFO) queue is blocked due to a contending queue, this is nothing else than a delayed service. Therefore, it is reasonable to consider the contention delay as a service time increase (or service rate decrease). Consequently, we come to a reduced equivalent system that consists of one queue per input, each with an individual server with a service rate $\mu_i(y)$, as shown in Figure 6b). Therein, the service rates depend on the current router state y, i.e., contention situation. The service rates decrease according to the degree of contention in the current router state. We recognize that the service of contending queues is still coupled.

Due to the memoryless property of the exponentially distributed arrival and service processes, the state of the equivalent router system can now solely be defined by the number of flits contained in the input queues. If we represent the state by a vector where each element represents the fill level of a single input queue, we can model the system by means of a multidimensional Markov chain. This is illustrated in Figure 7 for the case of a router with two inputs (please ignore the depicted macro states for now). Therein, the transition rates are defined by the arrival rate λ_i and service rate μ_i for each input independently. Let $\boldsymbol{x}$ be the current state vector of the router. Then, a transition from state $\boldsymbol{x} \rightarrow \boldsymbol{x} + e_i$ (where e_i is the i'th unit vector, i.e., a vector of all zeros except element i, which is equal to one) has an intensity of λ_i. On the other hand, a transition $\boldsymbol{x} \rightarrow \boldsymbol{x} - e_i$ has an intensity of μ_i. The boundaries of the Markov chain are an exception to that rule (first column/row in Figure 7). There, we find a different contention situation. In the case of two inputs, we have no contention caused by the second input anymore. Therefore, the transition rates for $\boldsymbol{x} \rightarrow \boldsymbol{x} - e_i$ change to μ, i.e., the basic router service rate without contention delay.

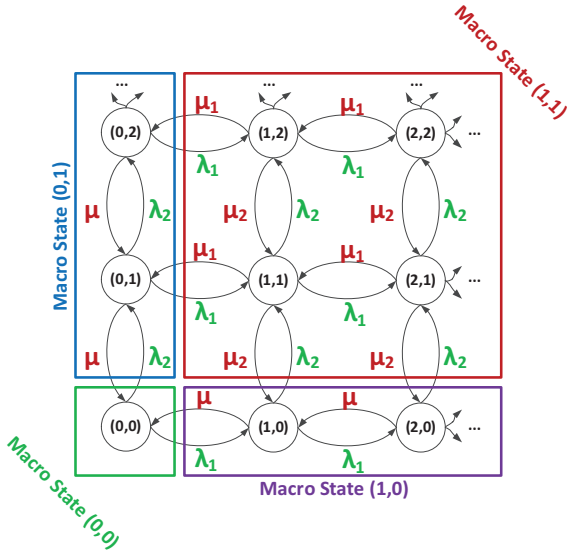


Figure 7. Example of a two-dimensional Markov model for a router with two inputs and the decomposition into reversible sub-chains.

For solving the Markov chain, we still need to know the service rates μ_i that include the contention delay to be able to define the transition rates. For this purpose, we apply an idea that was proposed in [19] to determine the mean waiting time for a similar input buffered router model assuming an FCFS arbiter. We modify this approach to find an estimation for the mean service time, i.e., the waiting time of the flit in front of the queue. Similar to [19], we first compute the pairwise contention probability $c_{i,j}$ for all inputs pairs (i, j) of a router with P inputs based on the forwarding probabilities $\boldsymbol{F}$ that can be derived according to (5):

$$c_{i,j} = \sum_{k=1}^{P} f_{i,k} f_{j,k}, i \neq j, 1 \leq i, j \leq P. \quad (7)$$

From (7), an equivalent matrix equation can be derived

$$\boldsymbol{C} = \boldsymbol{F} \cdot \boldsymbol{F}^{\mathrm{T}}. \quad (8)$$

Note that the main diagonal of the contention probability matrix $\boldsymbol{C}$ in (8) is set to "1" which makes the following computation more convenient. Based on the contention probabilities, we can derive an expression to estimate the mean service times

$\overline{x}_i(\boldsymbol{y})$ under contention:

$$\overline{x}_i(\boldsymbol{y}) := \overline{x} + \overline{x} \sum_{j=1, j\neq i}^{P} c_{i,j} y_j, 1 \leq i \leq P. \quad (9)$$

The first summand $\overline{x}$ of (9) represents the mean router service time for the packet in front of queue i. The second summand considers the contention delay. Therein, the vector $\boldsymbol{y}$ represents the instantaneous fill state of each input queue, i.e., $y_i = 0$, if input queue i is empty and does not contribute to the contention delay and $y_i = 1$ otherwise. We call $\boldsymbol{y}$ the *router macro state* in the following and can directly derive it from the router state $\boldsymbol{x}$:

$$y_i = \begin{cases} 0, & if\ x_i = 0 \\ 1, & if\ x_i > 0 \end{cases},$$

or rather informally: $\boldsymbol{y} = \text{sgn}(\boldsymbol{x})$.

We can still condense (9) somewhat by exploiting the convenient definition of contention probability matrix $\boldsymbol{C}$ and provide a short form matrix equation for the mean service rates $\mu_i(\boldsymbol{y})$ (i.e., the inverse of the mean service times):

$$\mu_i(\boldsymbol{y}) := \left[\frac{1}{\mu}\boldsymbol{C}_i{}^{\mathrm{T}}\boldsymbol{y}\right]^{-1}, 1 \leq i \leq P. \quad (10)$$

With the definition for the mean service rates $\mu_i(\boldsymbol{y})$ in (10) we have now all necessary inputs to solve the Markov chain in order to obtain the steady-state probability distribution. However, in trying to do so, we are confronted with another challenge. If we apply the Kolmogorov criterion for reversibility of Markov chains, we soon realize that it does not hold for some cases in the peripheral region of our Markov chain. Accordingly, the chain is not time reversible; see Figure 7 and examine the following state transitions: $(0,0) \rightarrow (1,0) \rightarrow (1,1) \rightarrow (0,1) \rightarrow (0,0)$, and the corresponding return path. We notice that the product of the transition rates is not equal for both directions, and thus, it does not fulfill the Kolmogorov criterion [26]:

$$\lambda_1 \cdot \lambda_2 \cdot \mu_1 \cdot \mu \neq \lambda_2 \cdot \lambda_1 \cdot \mu_2 \cdot \mu.$$

Consequently, we are not allowed to apply local balance equations to solve the chain. Unfortunately, we are not able to find a closed-form solution for the infinite Markov chain solely based on the global balance equations. Fehske and Fettweis [27] recently encountered exactly the same problem when trying to solve an equivalent Markov chain. They proposed an approximation to find a solution for the stationary distribution. The approach is based on the concept of *aggregation of variables* that is well known by economics for quite some years [28]. The proposed algorithm consists of the following steps.

We start decomposing our Markov chain into reversible sub-chains. This is done by collecting all states $\boldsymbol{x}$ that belong to the same macro state (or aggregate state) $\boldsymbol{y} = \text{sgn}(\boldsymbol{x})$ in a common set $\mathcal{S}(\boldsymbol{y})$:

$$\mathcal{S}(\boldsymbol{y}) := \left\{\boldsymbol{x} \in \mathbb{N}_0^P \mid \text{sgn}(\boldsymbol{x}) = \boldsymbol{y}\right\}.$$

The idea behind the definition is that all states are collected in the macro state where we find a similar contention situation. If we consider a contending queue, it does not matter how many packets it contains, only if it contains at least one packet or not. Consequently, the mean service rates are homogeneous within each macro state. Thus, service rates are decoupled by this aggregation approach, as Figure 6 c) shows. An example for the Markov chain decomposition for the case of two input ports is provided in Figure 7. Therein, we decompose the two-dimensional Markov chain into four macro states. Macro state $(0,0)$ contains all states where both input queues are empty (which is only a single router state $(0,0)$). Macro states $(1,0)$ and $(0,1)$ collect the states where only one of the two queues is empty. Hence, we have no contention within these two macro states. Macro state $(1,1)$ represents all router states where both queues are not empty. In this two-dimensional example, this is the only macro state where contention occurs.

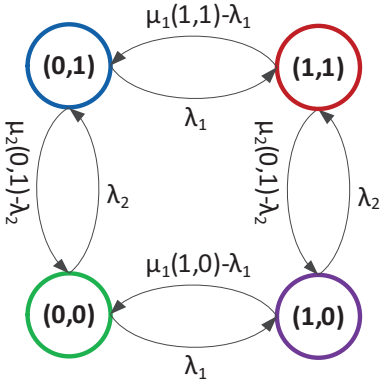


Figure 8. Example: Markov chain on macro state level assuming a router with two inputs.

Since the transition rates are homogeneous within each macro state, the sub-chains are reversible and can be solved. This leads to a product form solution for the stationary probability distribution of the number of customers (i.e., packets) $\widetilde{\pi}$ in an M/M/1 queueing system that is well known from classical queueing theory [11][27]:

$$\widetilde{\pi}(\boldsymbol{x}) = \begin{cases} \prod_{i\in N_1(\boldsymbol{y})} (1-\rho_i(\boldsymbol{y}))\, \rho_i^{x_i-1}(\boldsymbol{y})\sigma(\boldsymbol{y}), & \text{for } \boldsymbol{y} \neq \boldsymbol{0} \\ \sigma(0), & \text{for } \boldsymbol{y} = \boldsymbol{0} \end{cases} \quad (11)$$

with utilization $\rho_i(\boldsymbol{y})$ of input queue i defined as

$$\rho_i(\boldsymbol{y}) := \frac{\lambda_i}{\mu_i(\boldsymbol{y})}.$$

The terms $\sigma(\boldsymbol{y})$ denote the probabilities of finding the system in macro state $\boldsymbol{y}$ (i.e., one of the states in $\mathcal{S}(\boldsymbol{y})$). Note that (11) only yields an estimate for the solution of the stationary probability distribution. This is because we omit the transitions between the macro states at this consideration. Also, note that (11) is conditioned on the probabilities of the corresponding macro state $\sigma(\boldsymbol{y})$ to ensure that $\sum_{\boldsymbol{x}} \widetilde{\pi}(\boldsymbol{x}) = 1$.

So far, we have no knowledge about the macro state probabilities $\sigma(\boldsymbol{y})$. We can compute $\sigma(\boldsymbol{y})$ by solving the (now finite) Markov chain on macro state level. Figure 8 shows a solution for the transition rate $p(\boldsymbol{y}, \boldsymbol{y}')$ from macro state $\boldsymbol{y}$ to macro

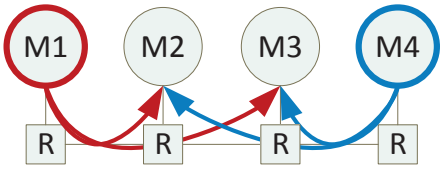


Figure 9. Topology and traffic pattern of first simple test scenario (M=module, R=router).

state $\boldsymbol{y}'$, as provided by [27]:

$$p(\boldsymbol{y},\boldsymbol{y}') = \begin{cases} \lambda_i, & \text{for } \boldsymbol{y}' = \boldsymbol{y} + e_i \\ \mu_i(\boldsymbol{y}) - \lambda_i, & \text{for } \boldsymbol{y}' = \boldsymbol{y} - e_i \\ 0, & \text{else} \end{cases}, \qquad (12)$$

where e_i again represents the unit vector for dimension i. Based on (12), we can now define the transition probability matrix $P = \left[p_{ij}\right]$ with $p_{ij} := p\left(y_i, y_j\right)$. With the definition of

$$p_{ii} := -\sum_{j=1}^{2^P} p_{ij}$$

we normalize the row sum to 0.

Finally, we can follow the usual approach and solve the equation system for the vector of macro state probabilities σ based on the transition probability matrix P:

$$\sigma P = 0,$$

under the side condition $\sum_{\boldsymbol{y}} \sigma(\boldsymbol{y}) = 1$.

Based on (11), we can now compute the approximates for the state probabilities $\widetilde{\pi}(\boldsymbol{x})$. We can derive several key performance indicators, such as the mean number of packets in the queue $\mathbb{E}[x_i]$:

$$\mathbb{E}[x_i] \approx \sum_{\boldsymbol{x}} \widetilde{\pi}(\boldsymbol{x}) x_i = \sum_{\boldsymbol{y}} \frac{\rho_i(\boldsymbol{y})}{1-\rho_i(\boldsymbol{y})} \sigma(\boldsymbol{y}), \qquad (13)$$

or the mean queueing delay W_i for input queue i by applying Little's law [11]:

$$W_i = \frac{\mathbb{E}[x_i]}{\lambda_i}.$$

V. Numerical Evaluation

We show the accuracy of the proposed NoC model by comparing it against cycle-accurate NoC simulation. Due to the similar system model assumptions we decided to compare our approach against the model proposed in [19] as well as the NoC simulation tool that has been used therein [29].

We assumed following common simulation parameters:

- deterministic, dimension-ordered XY-routing,
- flit traffic, i.e., packet size = 1,
- input buffered routers with FCFS arbiter and service rates of $\mu = 0.5$,
- large buffer size (256 flits) to approximate the infinite buffer model, and
- simulation run time of 10^5 cycles with a warm-up period of 10^4 cycles.

We investigate the following two topology/traffic scenarios under different load conditions (defined by number of injected packets/cycle) and compare the average packet transmission latency in the network.

A. Introductory Example

First, we choose a very simple scenario to investigate the model behavior under a clear contention situation. Therefore, we consider a simple chain of four routers, as depicted in Figure 9, where a single module is connected to each router. The modules 1 and 4 are sending their packets to modules 2 and 3 with equal probability. Modules 2 and 3 do not send any packets. Hence, we find at router 2 and 3 a contention situation with the following forwarding probability matrix $\boldsymbol{F}$:

$$\boldsymbol{F} = \begin{pmatrix} 0 & 0 & 0 \\ 0.5 & 0 & 0.5 \\ 1 & 0 & 0 \end{pmatrix}.$$

The result under different load conditions is shown in Figure 10. We find that the latency estimation for our proposed approach (red curve with + marker) follows very well the cycle-accurate simulation results (black curve with point marker) under a low and medium load condition. However, it significantly underestimates the network saturation limit where latency tends to infinity (0.66 packets/cycle in our model compared to 0.8 packets/cycle in the cycle-accurate simulation). The reference mean value model from [19] (blue curve with circle marker) shows a slight overestimation of the latencies under mid load conditions but estimates the network saturation point quite well.

The reason for the poor estimation of the network saturation point of our model is the applied aggregation approach for approximating the solution of a Markov chain. Therein, the stability of the overall solution is determined by the stability of the "worst-case" aggregate, i.e., the aggregate with the highest contention. If the solution for the "worst-case" aggregate tends to infinity the overall solution tends to infinity as well. To avoid this behavior, we propose to determine an average service time

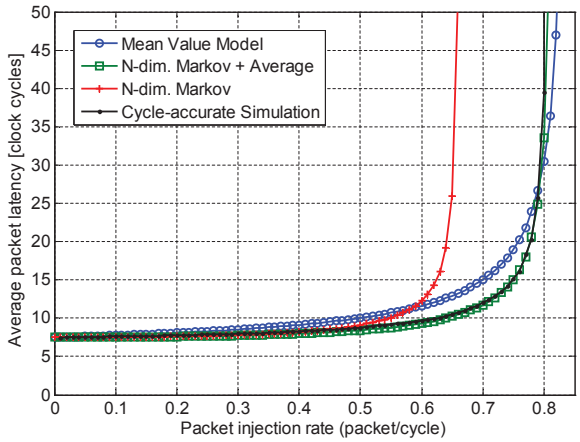


Figure 10. Performance results for 4x1 chain analyzing the average packet latency in comparison to cycle-accurate simulation.

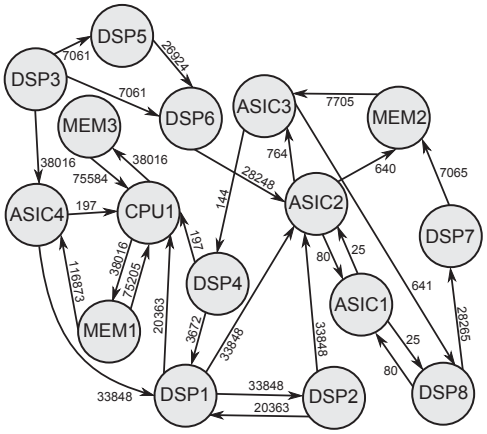


Figure 11. Traffic pattern of 4x4 2D-mesh test scenario with application-specific traffic [30] [19].

$\bar{\bar{x}}_i$ over all macro states for every router input. This is done by computing the expectation of the mean service times $\bar{x}_i(\boldsymbol{y})$ over all macro states based on the known macro state probabilities $\sigma(\boldsymbol{y})$:

$$\bar{\bar{x}}_i = \sum_{\boldsymbol{y}\in\{0,1\}^P} \bar{x}_i(\boldsymbol{y})\sigma(\boldsymbol{y})y_i. \quad (14)$$

Therein, y_i constrains the expectation to those macro states where queue i is not empty. We compute the average waiting time W_i for input queue i based on (14):

$$W_i = \frac{\bar{\bar{x}}_i}{1-\lambda_i\bar{x}_i}.$$

The result of the refined approach is also depicted in Figure 10 (green curve with square marker). It shows a very good match compared to the cycle-accurate simulation. The latencies under low/mid load conditions, as well as the network saturation point, are estimated very accurately by this approach. The average estimation error is less than 3%.

B. Multimedia application scenario

In addition, we choose a 4x4 2D-mesh topology using a more diverse traffic pattern of the generic multimedia application from [30]. The traffic scenario is depicted in Figure 11 while the topology and the core mapping is shown in Figure 12. We target to compare the estimation quality of the average latencies under more complex contention situations. The results are plotted in Figure 13 and confirm the accurate results of the first scenario. Again, the average estimation error is around 3% (9% for the reference model). However, we still notice a slight underestimation of the network saturation limit of about 2.5% for that case. The reference mean value model shows a better accuracy in this region.

Note that the presented results only serve as proof of concept. However, they easily scale to larger networks. The relative accuracy of the latency estimation is expected to stay in the same range under similar contention situations, independent of the NoC size because the analysis of the queueing delay is done on router level and only accumulated on network level.

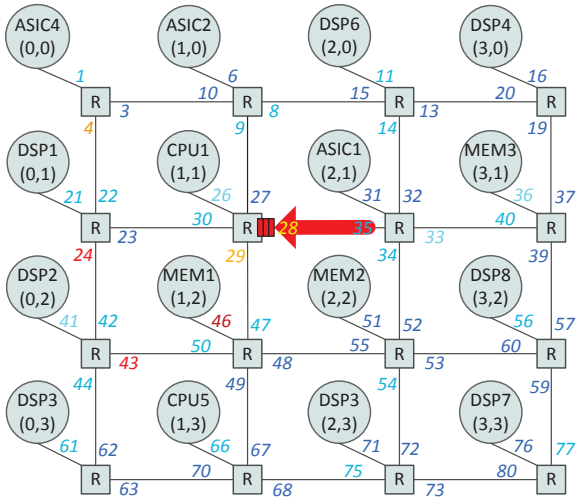


Figure 12. Core mapping based on 4x4 2D-mesh topology (R=router) [19]. The annotated numbers represent the index of the associated input buffer. The color denotes the tail probabilities ($P[x \geq 1]$) at the corresponding input buffers (red=high, yellow=medium, blue=low) which reveals the bottlenecks in the 4x4 2D-mesh.

C. Buffer dimensioning based on tail probability estimation

One advantage of the presented NoC traffic model is its flexibility to derive arbitrary performance metrics based on the distribution of the number of customers from (11). This is demonstrated in the following. An important step of the design space exploration for NoC based MPSoC is the so called buffer dimensioning. Therein, the necessary buffer capacity K is estimated for every router input for a given, topology, application (i.e., traffic pattern) and routing scheme. Memory consumes a lot of chip area and is therefore a crucial factor to reduce chip production cost. Hence, a careful investigation and optimization of the router memories is recommended. We employ the tail probability P_{tail} to invest the necessary buffer amount. This measure indicates the probability that a certain

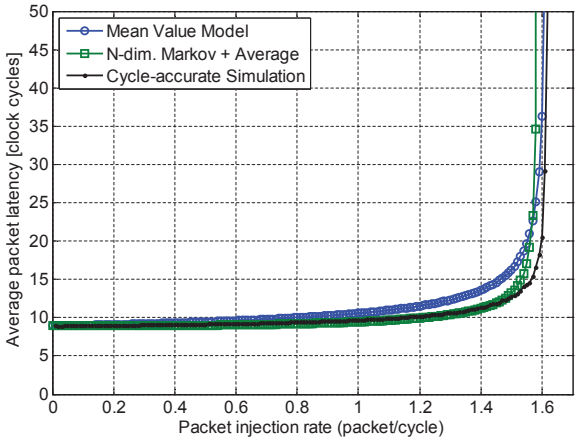


Figure 13. Performance results for 4x4 2D-mesh with generic multimedia application traffic analyzing the average packet latency in comparison to cycle-accurate simulation.

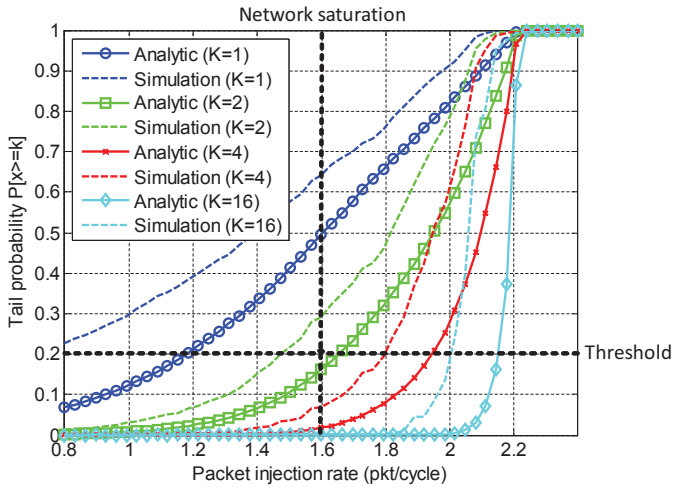


Figure 14. Analysis of the tail probability for a single router input buffer in a 4x4 2D-mesh with generic multimedia application traffic in comparison to cycle-accurate simulation.

buffer fill level is exceeded and is thus well suited for this purpose. It can easily be derived from the distribution of the number of customers $\widetilde{\pi}$ of the presented infinite buffer model.

$$P_{tail} = P[x_i \geq K] = 1 - \sum_{\boldsymbol{x}, x_i < K} \widetilde{\pi}(\boldsymbol{x}) \qquad (15)$$

Equation (15) computes the tail probability for router input queue i. On this basis, we can now examine the buffer requirements focusing on a single router input of the presented 4x4 2D-mesh scenario. The location of the selected buffer (index 28) is illustrated in Figure 12. We investigate the tail probability for different injection rates (in range 0.8 to 2.4 packets/cycle) and different K (1, 2, 4, and 16). The results are again validated by comparison with cycle-accurate NoC simulations, as shown in Figure 14.

The figure yields much information that can help a designer to optimize the buffer according to the expected traffic volume. First, we define a threshold that is used to decide whether it is worth to investigate additional memory or not. We first consider the curves obtained from the analytical model. We take a tail probability of 0.2 as our threshold. This means that for a given K and traffic load, it is recommended to increase K, if the buffer is completely filled in at least 20% of the time. In Figure 14, we can see that under low and medium traffic load the threshold is not exceeded; even for a very small buffer size (K=1). At an injection rate of 1.2 packets/cycle, the curve "Analytic (K=1)" reaches the threshold and thus it is recommended to use a buffer size of 2 for this traffic load. This is sufficient up to an injection rate of 1.7 packets/cycle, where the green curve (Analytic K=2) exceeds the defined threshold. For higher injection rates it is recommended to use a buffer size of K=4. Only within the small region of 2 to 2.2 packets/cycle, it would make sense to use an even bigger buffer (K=16). For higher injection rates, the incoming traffic cannot be served anymore and the buffer is completely filled, independent of the buffer size. From our previous analysis of Figure 13, we know that the overall network saturation is already reached at an injection rate of about 1.6 packets/cycle. Taking this additional information into account, there is no need for over-provisioning the buffer by considering injection rates beyond network saturation. Finally, we can conclude from Figure 14 that a buffer size of only 2 is already sufficient to deal with all sensible traffic loads (up to 1.6 packets/cycle) for the given application scenario. Furthermore, Figure 14 depicts the results from the cycle-accurate simulation as reference. We find that the analytic model is able to represent the general behavior of the simulation quite well. However, we also see that it generally underestimates the tail probability due to the simplified assumptions of the arrival and service time distributions (M/M/1), as well as, the additional inaccuracy caused by the aggregation approach. Nevertheless, the influence on the design decisions is insignificant so that the proposed approach is well suited for the early DSE phases.

Up to now, we focused on the investigation of a single input buffer under different injections rates. If we fix the injection rate to a value close to the network saturation (1.6 packets/cycles), we are able to analyze the buffer requirements under worst-case conditions for the whole NoC at once. The results are presented in Figure 15 and provide the NoC designer an easily comprehensible and intuitive tool for the buffer dimensioning.

The figure illustrates the tail probability as color-coded blocks for all buffers of the network (x-axis) and different buffer sizes K (y-axis). Thereby, a blue block corresponds to a low P_{tail}, i.e., a quite relaxed traffic situation. A red block represents a high P_{tail}, and reveals potential bottlenecks. For these cases, it is suggested to increase the buffer size until reaching the green or blue region to optimize the traffic flow and avoid network congestion. We observe in Figure 15 that a quite low buffer size (K=1 or K=2) is sufficient for most buffers. Only a few buffers require some additional memory. E.g., a buffer size of 4 would be reasonable for input buffers 28 and 29. The bottleneck in this network scenario is clearly buffer 46 which is connected to the output of "MEM1". This is accordant to the traffic scenario in Figure 11, where MEM1 has a very high traffic load at its output. The diversity of the traffic load is caused by the application specific traffic pattern, which is clearly represented in Figure 15. The scenario demonstrates the advantage of a careful network analysis and buffer dimensioning quite well. Individual buffer sizing can save a lot of memory while ensuring a high performance at the same time.

Annotating the network topology with the corresponding color-coded tail probabilities (for K=1) at the input identifiers yields a clear picture concerning the localization of the bottlenecks. This is illustrated in Figure 12. We find the highest congestion around the modules "MEM1", "CPU1", "DSP1", and "DSP2". Relating to the traffic pattern in Figure 11, we can verify that these components are indeed highly interconnected, communication intensive, centric nodes.

As mentioned before, we observe a big diversity of the buffer loads due to the application scenario. Assuming a uniform traffic scenario (i.e., each module communicating with every other module with the same probability), we expect a more

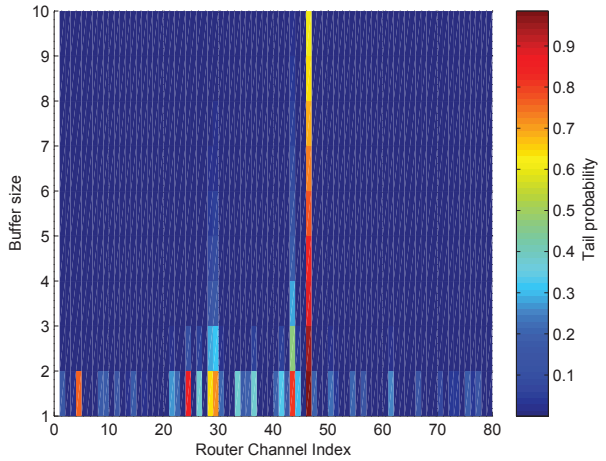


Figure 15. Buffer dimensioning for 4x4 2D-mesh with generic multimedia application traffic at fix injection rate (1.6 packets/cycle).

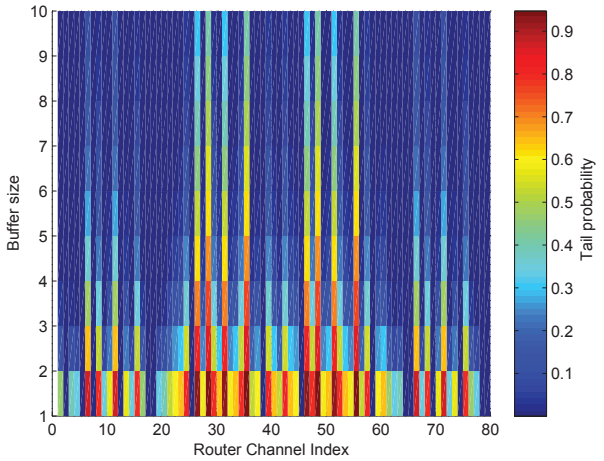


Figure 16. Buffer dimensioning analysis for 4x4 2D-mesh with uniform traffic at 0.31 packets/cycle.

uniform utilization of the buffers in the network. For this purpose, we repeat the buffer dimensioning analysis for the 4x4 2D-mesh applying uniform traffic. Again the injection rate was chosen close to the network saturation (0.31 packets/cycle at a basic flit service rate of $\mu = 0.5$). The results are depicted in Figure 16. We can see that the buffer utilization is now much more uniform than in the case of the application specific traffic. Nevertheless, we still find some bottlenecks in the region of buffer 26 to 35 and 46 to 55. Referring to Figure 12, we find that all these buffers are inputs of center routers. The center bottleneck is a commonly known characteristic of 2D-mesh topologies under uniform traffic.

The result confirms the validity and suitability of the proposed analytic model. It shows that we can already gain much insight into the system behavior in an early design stage with only a limited amount of information concerning system parameters.

D. DSE Scenario

Now, we have all necessary analytic tools at hand to clarify the question put at the end of the small DSE example in Section I. Therein, we proposed two alternative topologies (Figure 2(b) and Figure 2(c)) for a given application specific traffic scenario (Figure 2(a)). The first topology (solution a) employs two modules per router, while the second topology (solution b) spends a single router for every module. Finding a trade-off between latency, throughput and area consumption, we were not able to find a clear answer in Section I concerning which of the two alternatives is better suited. We investigated the small DSE example more closely performing a tail probability analysis using the proposed analytic model. Annotating the color-coded tail probabilities $P[x \geq 1]$ (according to Figure 12), we can visualize the performance bottlenecks for the two topologies, as depicted in Figure 17. It can be seen that the left router in solution a) is a serious bottleneck in the network. Indeed, the arriving traffic even exceeds the service capabilities of the router. Therefore, the average packet latency in the NoC tends to infinity, if we employ the infinite buffer queueing model. We can conclude that solution a) is not suited to fulfill the performance requirement of the given application scenario. Solution b) offers an increased bisection width. This results in a better spatial distribution of the traffic load in the network. Though the link between the routers at module 4 and 3 has still a high load, all routers are able to handle the incoming traffic. The average latency in the network is 8.4 cycles, according to the analytic results. Now, we are able to make a clear decision. Topology a) is not able to achieve the required throughput. Therefore, solution b) would be the better alternative in this case, even though the additional two routers cause increased chip area. The results of this small DSE demonstrate that a little change can sometimes make a big difference. Hence, it is worth to spend some effort to investigate the interconnection more closely.

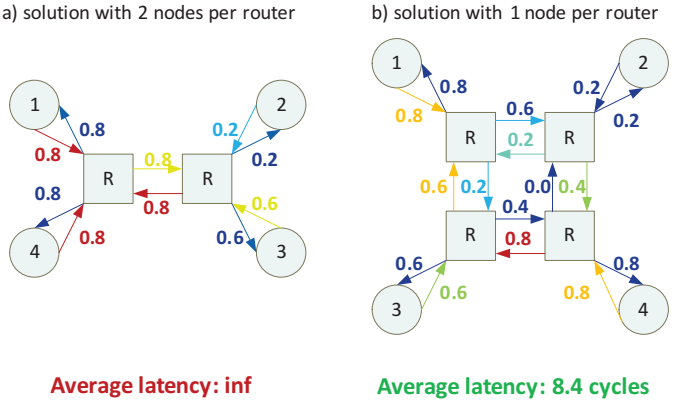


Figure 17. Results for small DSE example from Section I employing analytic model; comparison of solution with one (a) and two (b) modules per router. The color denotes the tail probabilities ($P[x \geq 1]$) at the corresponding input buffers (red=high, yellow=medium, blue=low).

VI. Extensions

The previous section demonstrated the feasibility and accuracy of the basic NoC model. However, several extensions are possible to further increase the accuracy and expand the application area of the proposed model.

A. Transient Behavior of NoC Router

The router and NoC models presented in Section IV describe the steady-state behavior of NoCs with adequate accuracy, but real systems do not necessarily work in steady-state. For instance, after starting a system, it needs time to converge to steady-state. Furthermore, application and communication patterns can change over time causing time-varying traffic rates in NoCs. An example of adaptive application and communication patterns can be found in [12], where a runtime adaptive NoC with dynamic routes and buffer allocations is proposed. This example illustrates that stationarity is not assured and it is of principal interest to consider transient characteristics as well. In the following, we outline model extensions that capture the transient behavior of a router.

In order to enable a transient analysis, we apply a generalization of the aggregation technique published in [31]. The technique in [31] is designed for continuous-time queueing systems, but we adapt it to discrete-time processes. Most variables used so far become time-dependent, e.g., arrival rates are described by $\lambda(t)$ instead of λ.

1) Transient behavior of uncoupled queues: The first step of the modified aggregation approach is to determine the transient behavior of uncoupled, independent queues described by the state probabilities $\pi_k(\boldsymbol{y}, t)$, which has to be done for all macro states $\boldsymbol{y}$ separately. We utilize standard tools for discrete-time markov chains for constructing a transition matrix Q [11]. The transient behavior of the state probabilities is computed iteratively by multiplying the state probabilities of the previous time step by the transition matrix, i.e., $\pi(\boldsymbol{y}, t+1) = \pi(\boldsymbol{y}, t)Q$. In order to enable this computation, we restrict the state space of a single queue to be finite.

2) Transient behavior of coupled queues: In the second part of the aggregation approach, the state probabilities of independent queues are utilized for approximating the transient behavior of coupled queues. Referring to [31], the transient rates among macro states (see (12)) can be generalized to

$$p(\boldsymbol{y}, \boldsymbol{y}', t) = \begin{cases} \lambda_i(t) & \text{for } \boldsymbol{y}' = \boldsymbol{y} + e_i, \\ (1-\lambda_i(t))\mu_i(\boldsymbol{y}, t)\frac{\pi_1(\boldsymbol{y},t)}{(1-\pi_0(\boldsymbol{y},t))} & \text{for } \boldsymbol{y}' = \boldsymbol{y} - e_i, \\ 0 & \text{else.} \end{cases}$$

Next, we summarize the transition rates to a transition matrix $\tilde{Q}(t)$. By iterative multiplication of the previous system state by the transition matrix, we derive the transient behavior of the coupled system, i.e., $\sigma(\boldsymbol{y}, t+1) = \sigma(\boldsymbol{y}, t)\tilde{Q}(t)$. Finally, in analogy to Section IV-C, the resulting macro state probabilities $\sigma(\boldsymbol{y}, t)$ can be used to determine various key performance indicators.

3) Numerical evaluation: For illustration, we consider a single router with three input/output pairs. Flits enter the router at one input and leave it at an output of a different input/output pair. Therefore, flits cannot be routed from the input to the output of the same input/output pair. We assume a maximum finite buffer length of 10 flits per queue. At each input, an arrival process with a fixed mean intensity λ is assumed, and the traffic destinations are uniformly distributed. In different simulation runs, we use diverse mean intensities between 0.6 and 0.8 flits per cycle. If needed, time-varying traffic intensities can also be applied within one simulation run. We utilize the transient model and determine the average buffer load according to (13) over the first 500 cycles after system startup. The results in Figure 18 show that the time the system needs to converge to stationarity clearly depends on the arrival rate. For high traffic scenarios the system needs about 300 cycles, while the same system approaches stationarity for $\lambda = 0.6$ within 50 cycles.

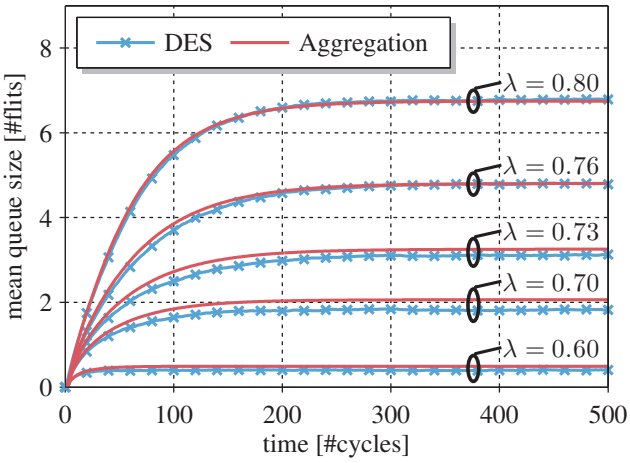


Figure 18. Mean queue length of one input of the router over time and after system startup. The proposed aggregation technique is compared to a discrete event simulation for different arrival rates λ.

Furthermore, we compare the aggregation technique to a cycle-accurate discrete event simulation (DES), where we simulate the system event by event. We perform a Monte Carlo simulation and average over 30000 realizations in order to obtain the average behavior of the system. The comparison of the aggregation technique to the DES reveals that the approximation works quite accurately, especially for low and high traffic scenarios the error is negligibly small. The largest error can be found for $\lambda = 0.7$, where the relative error of the mean queue size is 11%.

There are several applications for the transient model. It can, for instance, be used to predict dynamic behavior of NoC in algorithms that adapt application mappings or traffic patterns dynamically.

B. Blocking in NoC with finite buffers and generalized service

The model approach, proposed in Section IV-C, is based on an infinite buffer queueing model, which offers the advantage of low computational complexity. However, this assumption also entails some drawbacks. First, the spatial distribution of the traffic congestion in the network cannot be considered. Therefore, the prediction of performance measures becomes inaccurate, since every router is analyzed independently of the

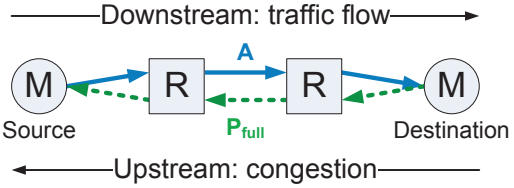


Figure 19. Mutual dependency of accepted traffic (A) and congestion feedback parameter P_{full} in finite buffer queueing networks.

subsequent (i.e., downstream) routers. This does not reflect the real on-chip situation accurately. Finally, buffer dimensioning based on tail probabilities, as demonstrated in Section V, is feasible but with limited accuracy. This is because blocking of injected traffic is not considered by the infinite buffer model.

Therefore, it is reasonable to consider an extension of the basic NoC model for finite buffer constraints. In the scope of the section, we sketch first ideas and challenges, which come with this extension. The integration and numerical evaluation is up for future work. Finite buffer models are well-known in queueing theory. Closed-form solutions are available for the most simple form, the M/M/1/K system [11]. However, first investigations show that the assumption of exponentially distributed service times limits the accuracy gain of the finite buffer model. Hence, it seems reasonable to consider a more general M/G/1/K system for this purpose. Unfortunately, there is no closed-form solution available for the distribution of the number of customers in an M/G/1 queueing system. A quite accurate and computationally efficient two-moment approximation is proposed in [32]. It can easily be extended to M/G/1/K systems following the approach of [33]. A good estimation of the blocking probability in an M/G/1/K system is also proposed by [34].

The biggest challenge in modeling finite buffer queueing networks is the spatial distribution of the traffic congestion. In case that an input buffer is filled, the service at the preceding (i.e., upstream) router stops for the corresponding output port. This means that the traffic congestion is propagated in upstream direction, the opposite direction of the data flow. It is necessary to derive an analytic expression for the probability that the buffer is filled, which is propagated as parameter in upstream direction to model traffic congestion. Note that this probability does not correspond to the blocking probability, since packets are never blocked (i.e., rejected) once they are injected in the NoC.

We realize that the accepted traffic (A) is propagated downstream while the congestion parameter P_{full} is propagated upstream, as depicted in Figure 19. Unfortunately, we find mutual dependencies between these two parameters in the network, if we consider general topologies. We propose an iterative fixed-point algorithm on network level to approximate the steady-state solution of a finite buffer queueing network.

We conclude that though this approach will become more computationally complex, it promises for an increased accuracy and enables analyzing network congestion and blocking of the injected traffic.

VII. Conclusion & Future Work

In this paper, we presented an analytic approach for modeling on-chip networks for many-core SoC based on queueing theory. In contrast to many existing models, the approach is very flexible in terms of supported topology, routing scheme and traffic pattern. The approach overcomes the limitations of the mean value analysis introduced in the existing work. Instead, it provides information about a steady-state distribution of the network routers. This allows to derive various key performance indicators, such as overflow probabilities or average queueing delays, which is very important information for dimensioning network resources, such as buffers, links, etc. We demonstrated the very high accuracy of the approach by comparison to a cycle-accurate simulation. The average estimation error for the mean latencies in a 4x4 2D-mesh is only 3%. The application of the proposed model was shown based on some simple design examples for buffer dimensioning, localizing bottlenecks, and benchmarking topologies. Extensiona of the basic model were proposed to consider finite buffers, dynamic traffic, and to generalize the service time assumptions made for the network routers. This further increases the accuracy of the basic analytic model and expands its application area.

The application and detailed evaluation of the suggested model extensions are left for future work. Furthermore, the consideration of link failures and error-prone routers employing a stochastic error model allows to investigate resiliency mechanisms for NoC. Finally, supporting multiple clock domains (i.e., globally asynchronous locally synchronous systems) and frequency scaling is another open topic in order to explore a many-core NoC more accurately.

Acknowledgment

This work was partly sponsored by the European Social Fund and the Free State of Saxony within the project *Secure Remote Execution* (SREX, nr 100111037).

References

[1] E. Fischer, A. Fehske, and G. Fettweis, “A Flexible Analytic Model for the Design Space Exploration of Many-Core Network-on-Chips Based on Queueing Theory,” in Proceedings of the Fourth International Conference on Advances in System Simulation, ser. SIMUL ’12, 2012, pp. 119–124.

[2] TU Dresden, “ESF Young Investigators Group; 3D Chip Stack Intraconnects - 3DCSI,” last visited on 12/12/2013. [Online]. Available: http://tu-dresden.de/die_tu_dresden/fakultaeten/fakultaet_elektrotechnik_und_informationstechnik/3dcsi

[3] J. Manferdelli, N. Govindaraju, and C. Crall, “Challenges and Opportunities in Many-Core Computing,” Proc. of IEEE, vol. 96, no. 5, May 2008, pp. 808 –815.

[4] S. Borkar, “Thousand Core Chips: A Technology Perspective,” in Proceedings of the 44th Annual Design Automation Conference, ser. DAC ’07, 2007, pp. 746–749.

[5] J. Nickolls and W. Dally, “The GPU Computing Era,” Micro, IEEE, vol. 30, no. 2, March-April 2010, pp. 56 –69.

[6] T. Limberg, M. Winter, M. Bimberg, R. Klemm, E. Matus, M. Tavares, G. Fettweis, H. Ahlendorf, and P. Robelly, “A fully programmable 40 GOPS SDR single chip baseband for LTE/WiMAX terminals,” in Solid-State Circuits Conference, 2008. ESSCIRC 2008. 34th European, 2008, pp. 466–469.

[7] W. Dally and B. Towles, "Route packets, not wires: on-chip interconnection networks," in Design Automation Conference, 2001. Proceedings, 2001, pp. 684–689.

[8] L. Benini and G. De Micheli, "Networks on chips: a new SoC paradigm," Computer, vol. 35, no. 1, Jan 2002, pp. 70 –78.

[9] R. Marculescu, U. Ogras, L.-S. Peh, N. 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, no. 1, Jan. 2009, pp. 3 –21.

[10] J.-Y. Le Boudec and P. Thiran, Network calculus: a theory of deterministic queueing systems for the internet. Berlin, Heidelberg: Springer-Verlag, 2001.

[11] L. Kleinrock, Queueing systems - 1 : Theory. New York: Wiley, 1975.

[12] M. Al Faruque, T. Ebi, and J. Henkel, "AdNoC: Runtime Adaptive Network-on-Chip Architecture," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 20, no. 2, 2012, pp. 257–269.

[13] P. Bogdan and R. Marculescu, "Non-Stationary Traffic Analysis and Its Implications on Multicore Platform Design," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 30, no. 4, 2011, pp. 508–519.

[14] W. Dally, "Performance analysis of k-ary n-cube interconnection networks," IEEE Transactions on Computers, vol. 39, no. 6, Jun 1990, pp. 775 –785.

[15] A. E. Kiasari, D. Rahmati, H. Sarbazi-Azad, and S. Hessabi, "A Markovian Performance Model for Networks-on-Chip," in 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing, 2008. PDP 2008., 2008, pp. 157–164.

[16] A. Kiasari, Z. Lu, and A. Jantsch, "An Analytical Latency Model for Networks-on-Chip," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, vol. 21, no. 1, 2013, pp. 113–123.

[17] J. Hu, U. Ogras, and R. Marculescu, "System-Level Buffer Allocation for Application-Specific Networks-on-Chip Router Design," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 25, no. 12, 2006, pp. 2919–2933.

[18] N. Nikitin and J. Cortadella, "A Performance Analytical Model for Network-on-Chip with Constant Service Time Routers," in Proceedings of the 2009 International Conference on Computer-Aided Design, ser. ICCAD '09, 2009, pp. 571–578.

[19] U. Ogras, P. Bogdan, and R. Marculescu, "An Analytical Approach for Network-on-Chip Performance Analysis," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 29, no. 12, Dec. 2010, pp. 2001 –2013.

[20] M. Bakhouya, S. Suboh, J. Gaber, and T. El-Ghazawi, "Analytical modeling and evaluation of On-Chip Interconnects using Network Calculus," in 3rd ACM/IEEE International Symposium on Networks-on-Chip, ser. NoCS 2009, 2009, pp. 74–79.

[21] A. E. Kiasari, A. Jantsch, and Z. Lu, "Mathematical formalisms for performance evaluation of networks-on-chip," ACM Computing Surveys (CSUR), vol. 45, no. 3, 2013, p. 38.

[22] U. Ogras and R. Marculescu, "Target noc platform," in Modeling, Analysis and Optimization of Network-on-Chip Communication Architectures, ser. Lecture Notes in Electrical Engineering. Springer Netherlands, 2013, vol. 184, pp. 39–47.

[23] M. Bossert and M. Breitbach, Digitale Netze / Funktionsgruppen digitaler Netze und Systembeispiele. Stuttgart ; Leipzig: Teubner, 1999.

[24] E. Fischer and G. P. Fettweis, "An accurate and scalable analytic model for round-robin arbitration in network-on-chip," in Seventh IEEE/ACM International Symposium on Networks on Chip, ser. NoCS '13, 2013, pp. 1–8.

[25] Seber and A. F. George, A Matrix Handbook for Statisticians. John Wiley & Sons, Inc., 2008.

[26] R. Nelson, Probability, stochastic processes, and queueing theory / the mathematics of computer performance modeling. New York ; Heidelberg [u.a.]: Springer, 1995.

[27] A. Fehske and G. Fettweis, "Aggregation of variables in load models for interference-coupled cellular data networks," in IEEE International Conference on Communications (ICC), 2012, 2012, pp. 5102–5107.

[28] H. A. Simon and A. Ando, "Aggregation of Variables in Dynamic Systems," Econometrica, vol. 29, no. 2, Apr 1961, pp. 111–138.

[29] worm_sim, "Cycle-accurate noc simulator," last visited on 12/12/2013. [Online]. Available: https://www.ece.cmu.edu/~sld/software.html

[30] J. Hu and R. Marculescu, "Energy- and performance-aware mapping for regular NoC architectures," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 24, no. 4, 2005, pp. 551–562.

[31] D. Öhmann, A. Fehske, and G. P. Fettweis, "Transient Flow Level Models for Interference-Coupled Cellular Networks," in 51st Annual Allerton Conference on Communication, Control, and Computing, Monticello, 2013.

[32] D. S. Myers and M. K. Vernon, "Estimating queue length distributions for queues with random arrivals," SIGMETRICS Perform. Eval. Rev., vol. 40, no. 3, Jan. 2012, pp. 77–79.

[33] J. Virtamo, "Queueing Theory; M/G/1-queue," 2013, lecture notes of Queuing theory, 38.3143, last visited on 12/12/2013. [Online]. Available: http://www.netlab.tkk.fi/opetus/s383143/kalvot/english.shtml

[34] J. M. Smith, "Properties and performance modelling of finite buffer M/G/1/K networks," Computers & Operations Research, vol. 38, no. 4, 2011, pp. 740 – 754.