Application Driven Traffic Modeling for NoCs

Leonel Tedesco ltedesco@inf.pucrs.br
Aline Mello alinev@inf.pucrs.br
Leonardo Giacomet luigi@inf.pucrs.br
Ney Calazans calazans@inf.pucrs.br
Fernando Moraes moraes@inf.pucrs.br

Pontifícia Universidade Católica do Rio Grande do Sul (FACIN-PUCRS)
Av. Ipiranga, 6681 - 90619-900 - Porto Alegre – BRASIL

ABSTRACT

The network on chip (NoC) design process requires an adequate characterization of the application running on it to optimize communication resources utilization and dimensioning. The traffic modeling process is the most essential step for characterizing complex applications. It is possible to identify three methods to model traffic in NoC literature. The first one assumes sources continually send data at a constant rate to the network and it is the most commonly used. The second method employs probabilistic functions to model the traffic behavior for typical applications, as audio and video streams. The accuracy of this method is better, at the extra cost of modeling complexity and simulation time. The third method employs traffic traces to evaluate network performance. Even with small traces, simulation time can be prohibitive. The advantage is accuracy, superior to the previous models. Even if a given application is correctly modeled, other flows interfere on how the application traffic behaves within the network. Results about the mutual interference of different traffic flows in NoCs are scarce. This work has two main objectives: (*i*) compare NoC performance, in terms of throughput and latency, when different traffic models are used for the same application; (*ii*) evaluate the impact of network *noise traffic* on some specific modeled flow. Preliminary results show how far is the real NoC performance for a given application when an oversimplified model is employed. The conclusion is that NoCs must employ internal mechanisms to ensure QoS, since noise traffic makes modeled traffic to depart from its predicted behavior.

Categories and Subject Descriptors

B.7.1 [Integrated Circuits]: Types and Design Styles – advanced technologies, algorithms implemented in hardware, VLSI (very large scale integration).

General Terms: Design, Experimentation, Measurement, Performance, Theory, Verification.

Keywords: Networks on Chip, Traffic Modeling, Applications, QoS.

1. INTRODUCTION

The design of embedded systems is currently an arduous task, because the system is necessarily composed by heterogeneous components, including: (i) general purpose CPUs; (ii) DSP for compute-intensive data handling workloads; (iii) dedicated IP blocks; (iv) communication infrastructure for data distribution and on-chip communication; (v) analog and R/F components to receive and transmit data to the external world. One method to manage design complexity is to separate computation from communication [1]. This is possible through IP reuse, standard interfaces and system level communication modeling. This prevents system design from scratch, reducing the time-to-market.

As SoCs grow in complexity and size, one of these components, on-chip communication, is becoming increasingly important. Point-to-point, busses and networks on chip (NoCs) [2] are alternatives to implement the communication infrastructure of SoCs. Proposals of NoCs for implementing communication in complex systems-on-a-chip are justified by reusability, scalability and energy efficiency properties displayed by these networks.

Typical NoC designs include steps like application modeling, performance evaluation and NoC synthesis [3][4][5]. The first step consists in describing the applications that will run on the NoC with traffic models, differentiating traffic without temporal restrictions from traffic with quality of service (QoS) requirements. The second step starts with a general NoC template and uses it to evaluate the performance of the NoC in terms of e.g. throughput, latency and jitter. The last step optimizes the NoC description to cope with the required performance. The traffic modeling process is the most essential step for characterizing complex applications. Even if a given application model is correct, other network flows interfere on how the application traffic behaves within the network, since these share resources with the modeled application, modifying the application traffic behavior.

Different traffic modeling methods are used by NoC research groups to characterize applications: constant injection rate [3][6][7], probabilistic functions [7][8] and trace based [4][6]. A simplified traffic model allows fast performance evaluation and wider design space exploration, at the cost of accuracy and/or inappropriate NoC parameters. As model complexity grows, accuracy increases at the cost of performance evaluation time reducing design space exploration possibilities. The first objective of this work is to compare NoC performance, in terms of throughput and latency, when different traffic modeling methods are used for the same application.

Traffic modeling must also consider the interfering traffic generated by others IPs. Results about the mutual interference of different traffic flows in NoCs are scarce [9]. This interfering traffic, modeled as random noise (in terms of interval time, packet size and type of service), enables to identify how the traffic influence modeled flows, and which internal mechanisms are required to ensure QoS. Evaluating the impact of the interfering traffic over the modeled traffic is the second objective of this paper.

SBCCI'06, August 28–September 1, 2006, Minas Gerais, Brazil.

This paper contains five other Sections. Section 2 presents related works in NoC traffic modeling. Section 3 describes traffic characterization, in terms of delivery requirements and service levels. Section 4 presents traffic modeling, with synthetic and traces models. Section 5 applies continuous and On-Off models presented in Section 4 to different experiments, evaluating their quality. Section 6 presents conclusions and directions for future work.

2. RELATED WORK

As Section 1 stated, applications can be modeled with analytical approaches (constant injection rate and probabilistic functions) and/or using real traces. This Section presents related works in NoC literature containing application traffic modeling and performance evaluation data.

Bolotin et al. [3] employ the first method: data injected into the NoC at constant injection rate. They define four service levels, modeling four kinds of traffic occurring in most computational systems: (i) signaling, composed typically by interrupt requests and other control signals; (ii) real time, such as audio and video stream processing; (iii) read/write, when accessing individual words of a memory; and (iv) block transfer, exemplified by file transfer application or large blocks of memory data. They also use two spatial traffic scenarios: uniform (all nodes have the same probability to receive packets) and local (neighbors with higher probability to receive packets). Although the Authors define data rates close to real situations, there is neither variation on data injection rates nor in packet sizes. Thus, the modeled traffic does not accurately correspond to real application behavior.

Santi et al. in [8] employ the second method: probabilistic functions to model network traffic. This work uses three types of injection methods. The first one, Bernoulli, randomly inserts each packet in the network, varying both network insertion time and rate. The second and third methods use Markov On-Off and Pareto On-Off models to generate packet bursts. The Authors also use uniform and local spatial traffic scenarios. Results show that for an expectedly bursty traffic, with an offered load superior to 10%, it is necessary to insert QoS mechanisms in the NoC. Because results present normalized data (such as latency and injection rates), it is not possible to correlate the results to real applications.

Yum et al. in [7] also use probabilistic functions to characterize MPEG-2 applications (VBR traffic class). They select the size of each frame according to a normal distribution, and a fixed interval between each frame. This work also uses CBR and ABR traffic models. CBR generates equal packet sizes, while ABR models best-effort traffic. Experiments considered ABR traffic interfering with CBR and VBR flows. They also conclude that connectionless networks need some explicit mechanism to guarantee QoS. The Authors employ a method to allocate bandwidth for QoS flows, named virtual clock. Using virtual clocks, they obtained a smaller interference of best-effort traffic over QoS flows.

Genko et al. [4] employs emulation to evaluate network performance. Two types of traffic generators are used: stochastic and trace-based. The stochastic traffic generator allows the designer to specify the data injection rate, packet size, target node and the stochastic model (uniform and burst-mode). This generator uses an LFSR to inject packets with a given probability. Traffic traces contain packet descriptors stored in memory. Each descriptor contains packet size, target node and injection timestamp. The emulation method presented in this paper allows evaluating the network performance for a huge number of packets, which is unfeasible by simulation. Nonetheless, experiments use short packets (about 5 flits) and an excessive offered load (45%), which again does not correspond to real applications.

Hu and Marculescu [6] adopted two traffic scenarios to evaluate an adaptive routing algorithm. In the first one, IPs are connected in a 6x6 mesh, generating 5-flit packets, with transmission interval varying according to an exponential distribution. In the second one, nine IPs disposed in a 4x4 mesh are randomly chosen to generate MPEG-2 traffic (bursty data obtained from traces), and the remaining IPs generate traffic at a constant rate (traffic noise). The Authors justify the use of traces to simulate situations near to real applications. Features as self-similarity and long-range dependence may reduce traces sizes, leading to smaller simulation times.

This state of the art review shows heterogeneity in terms of traffic modeling for NoCs. Most Authors employ synthetic workloads to validate some NoC features, e.g. QoS mechanisms or routing algorithms, without concern for which applications will finally run on these systems. Thus, it is important to define more realistic traffic models, which directly take into account the application requirements. With such a realistic traffic models, it will become possible to better optimize network resources (as buffers and wires) without penalizing NoC area and power consumption. This will happen while respecting temporal restrictions required by some modeled application.

3. TRAFFIC CHARACTERIZATION

In terms of delivery requirements, it is possible to classify traffic as either real-time or non-real-time. Real-time traffic implies the respect of strict temporal constraints, such as deadlines to receive data. Real-time may be further divided into streaming, block transfer and hard-real-time applications [10]. Continuous data transmission characterizes streaming applications, as audio and video. A same rate should be observable by both transmitter and receiver. This class is expected to sustain throughput, implying the loss of information if this requirement is not respected. Block transfer is similar to streaming in terms of deadlines, but in this class data is not transmitted continuously as in the streaming class. Typical block transfer applications are cache refill and voice. The critical performance figure in this class is latency, with rigid jitter requirements. Hard-real-time flows have strict latency (as in signaling traffic) and/or jitter requirements, not allowing deadline losses. Due to this feature, the hard-real-time class normally requires connection establishment. Non-real-time application data are transmitted, and eventually stored in buffers, for later consumption. In this case, there is no strict time constraint for data transmission. Because of this feature, non-real-time applications require a small fraction of network bandwidth.

QoS service levels differentiate applications according to their requirements (such as transmission rate, latency and jitter). Services levels as defined for ATM networks [11] are:

- CBR (Continuous Bit Rate) - service level used for connections that require a static amount of bandwidth, which remains available during the connection lifetime.
- VBR (Variable Bit Rate) - service level used by sources that generate data at injection rates that vary in time.
- ABR (Available Bit Rate) - service level projected for traffic with unknown bandwidth, being appropriate for best-effort applications.

4. TRAFFIC MODELING

This Section presents the first contribution of this work, traffic modeling for different applications targeting NoCs. Figure 1 illustrates the main parameters used to model traffic: packet size, packet generation interval, burst size and/or burst generation interval. Without appropriate traffic modeling, designers may overestimate or underestimate the use of network resources, penalizing NoC area and power consumption.

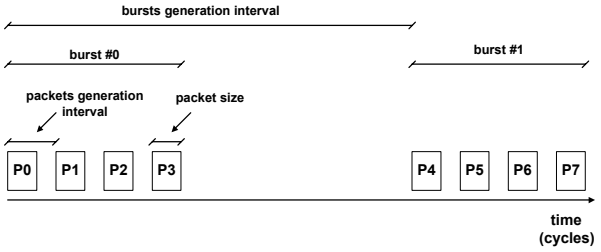


Figure 1 – Definition of network parameters to model traffic.

The analytic approach to traffic modeling uses mathematical functions to describe some chosen traffic parameters. The advantage of using this approach is the possibility to perform simulations with a reduced amount of packets that capture the main properties of the application, assuming that the model is sufficiently accurate to maintain the same statistical properties of the modeled application [12]. However, it is necessary to keep in mind that the exact characterization of real traffic sources is generally unfeasible.

This exact characterization is obtainable from traffic traces. In this case, the designer configures values of traffic parameters based on values collected from a network environment, allowing a precise representation of a known system. The main drawback of this approach is incurring in excessive simulation time. To circumvent the problem trace approaches are often associated to hardware emulation [4][6].

4.1 Constant Injection Rate

Constant injection rate models generate packets at a fixed rate. Applications such as digital non-compacted voice (8-bit samples transmitted at 64 Kbps), audio and non-compacted video are typical examples of constant injection rate traffic. Most research groups suggest to use this model either as a background noise traffic or to characterize applications [3][6][13]. Implementation ease is the advantage of this traffic model, since inputs for the traffic generator are just one value for the packet size and one value for the interval between packets. The main drawback of this model is the fact that real applications normally vary its data injection rates.

4.2 Probabilistic Models

It is possible to divide probabilistic models for traffic generation into two classes: one where the average traffic rate is not known a priori, named here Unknown Rate (UR) models, and another where rates are previously defined, named Known Rate (KR) models. The first class includes On-Off processes, while normal and exponential distributions are instances of the second class. The next paragraphs characterize each class.

The general structure of an On-Off process comprises alternate periods of traffic generation activity and inactivity, as depicted in Figure 2. During activity periods, the traffic source produces fixed-length packets at regular intervals, while during inactive periods there is no packet generation. The size of packet bursts and the duration of the Off period vary.

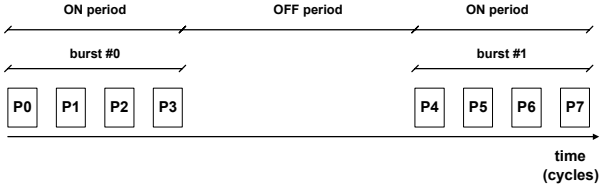


Figure 2 – Definition of parameters described the On-Off traffic model.

According to [14], an On-Off model using Pareto distribution function is useful to characterize applications like MPEG-2 video and internet traffic. Equation 1 describes Pareto On-Off distributions used to generate traffic.

$$t_{ON} = (1-r)^{\frac{-1}{\alpha_{ON}}} \quad \therefore \quad t_{OFF} = (1-r)^{\frac{-1}{\alpha_{OFF}}} \qquad (1)$$

In this Equation, r is a value in the interval [0:1], randomly chosen allowing to dynamically generate the size of the On and Off periods; α is a constant, On-Off formatting parameter, defined by the user to adapt the traffic generation to the application characteristic (e.g. for self-similar traffic, it is recommended [14] to use α_{On}= 1.9 and α_{Off}= 1.25). A UR model alternative to Pareto On-Off is the Markov On-Off. In this model, also known as Markov Modulate Process (MMP), the current state of a traffic source in a Markov Chain specifies data generation at a rate r. The function that describes state changes is an exponential.

Since a random variable defines the duration of On and Off periods at each use of the defining equation without any recourse to previously computed values, it should be clear that using On-Off models does not allow controlling the global average injection rate of the network. However, these models do allow controlling the rate of injection for each packet/burst. This characteristic makes On-Off appropriate to model noise traffic.

KR models, on the other hand, capitalize on discretized versions of continuous probabilistic functions, such as the normal and exponential distributions. Figure 3 illustrates the general process for using these models. Given some probabilistic distributions, e.g. those shown in (a), and given the amount of packets, together with a set of transmission rates, the process mounts a packet transmission table as in (b) indicating the number of packets for each defined rate. The table thus constructed corresponds to the discretization of the associated probabilistic function. Next, a random number generator (c) selects the rate to use when transmitting each packet and produces its generation timestamp. Following to this, the process fills a table (d) with the transmission timestamp and the contents of the packet. Processes (b) to (c) and (c) to (d) are repeated until all packets are inserted in table (d). Table (d) represents the traffic model to use in simulation/emulation.

Equations 2 and 3 present the normal and exponential distribution, respectively. In these equations, μ corresponds to the average injection rate and σ the injection rate standard deviation. Equations 2 and 3 compute the probability p for each rate.

$$p(rate) = \frac{1}{\sqrt{2\pi\sigma}} e^{\frac{-(rate-\mu)^2}{2\sigma^2}} \quad (2) \qquad p(rate) = \frac{1}{\mu} e^{\frac{-rate}{\mu}} \quad (3)$$

Contrary to what occurs on UR models, the global average injection rate for the KR models is directly obtainable from the particular distribution equation together with other parameters. Often, the inputs for defining the traffic model are the type of curve, the global average injection rate and a discrete set of rates.

4.3 Traffic based on real traces

Traces are real descriptions of application behavior contrary to the previously described methods. To obtain traces, it is necessary to attach some measurement equipment to an output port of an instance of the device to model. Traces enable the computation of parameters like transmitted number of bits and/or intervals between transmissions. They also enable storing these parameters in trace files and their conversion later to packet format. This method allows the designer to deal with the exact behavior of a network, being more adequate for emulation, due to the possibly huge amount of packets observed in real life situations.

4.4 Summary of applications and traffic generation methods

Table 1 presents a summary of traffic types and their characteristics. Signaling corresponds to control information that often represents requests originated from processing units to memories and/or to other processing units. This type of traffic normally requires hard-real time delivery bounds, to avoid stalling processors. Sensor network is another example application, where control signals are sent to all nodes periodically, to verify which of these are active.

HTTP web browsing is a network application where users require images and text with average size around 10 Kbytes with real-time constraints, due to the interactivity requirements between servers and clients. Traditionally, web browsing is modeled by Markov On-Off arrivals, with On and Off periods varying according to an exponential distribution. Pareto On-Off arrival processes can model Web applications, due to their self-similar nature [15].

Complex SoC operations normally involve large data blocks that transferred between IP cores. Another difference of this traffic is the real-time tolerance in terms of delivery time bounds, which makes it more tolerant to delay than signaling traffic. Constant injection rate [3] and probabilistic functions [7] can be used to model data block transfers.

Performance requirements of HDTV and MPEG-2 video are two relevant workloads for SoC research groups. These applications comprise large data blocks and must respect strict time intervals. This type of traffic is modeled with probabilistic functions [7], Pareto ON-OFF [14] and trace files [4][6].

5. EXPERIMENTAL SETUP

HERMES [16] is the reference NoC used to evaluate traffic modeling, a packet-switched mesh topology NoC. Its basic elements are a router with centralized control logic and five bi-directional ports. The Local port connects the router to an IP and the others ports connect routers to neighbor routers. Each input port has associated to it a buffer for temporary storage. The switching mode is wormhole. Handshake or credit-based flow control may be used. The flit size is parameterizable, and the maximum number of flits in a packet is fixed at $(2^{(\text{flit width, in bits})-1})$. The first and second flits of a packet contain header information, being respectively the address of the target node, and the number of flits in the payload. The remaining portion of the packet is the data payload.

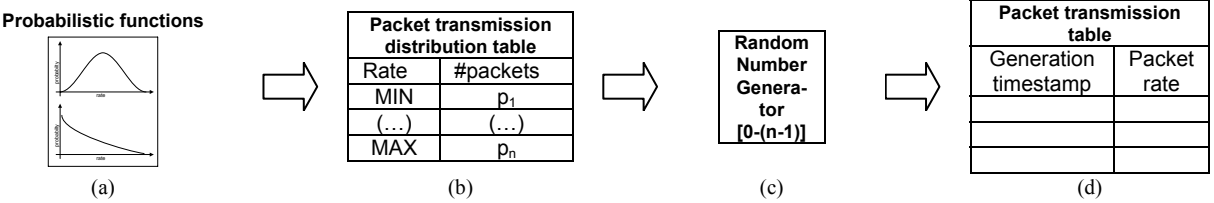


Figure 3 – Traffic generation according to probabilistic functions.

Table 1 – Summary of applications and their requirements.

Feature	Signaling	HTTP object	Data block transferred between IP cores	MPEG-2 HDTV
Applicable Model	- Constant injection rate	- Pareto On-Off - Markov On-Off	- Constant injection rate - Probabilistic functions	- Pareto On-Off - Trace files - Probabilistic functions
Typical Data Size	- ~bytes	- 3 Kbytes (typical) - 20 Kbytes (large)	- ~Kbytes	- ~Kbytes - ~Mbytes
Required Bandwidth	- ~Kbps	- 240 Kbps (typical) - 1.6Mbps (large)	- Kbps - Mbps	- 4– 6 Mbps (MPEG-2) - 20 Mbps (HDTV)
Delivery Requirement	- Hard Real-time RT)	- RT block transfer	- RT block transfer	- RT streaming
QoS Service Level	- CBR	- VBR – ABR	- CBR - VBR	- CBR (non-compacted video) - VBR (compacted video)
Simulation / Emulation	- Both	- Both	- Both	- Emulation

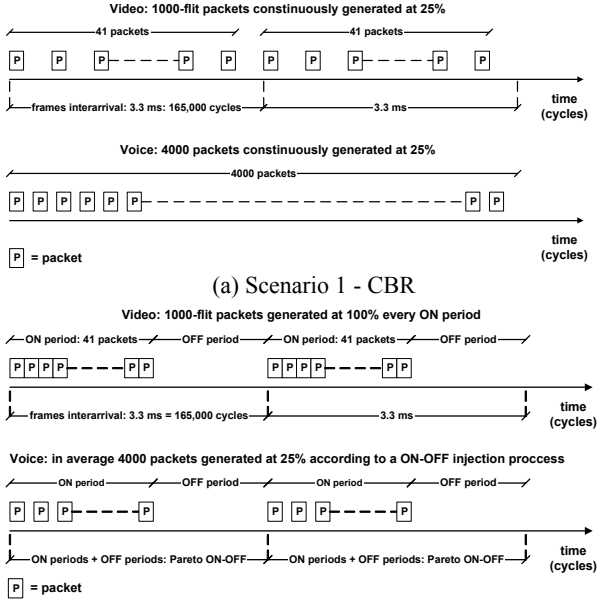
HERMES is described in RTL VHDL. The fixed network design parameters are: 8x8 mesh; credit-based flow control; 16-bit flit size; 8-flit buffers; XY routing algorithm. The NoC frequency is 50 MHz, corresponding to a link rate of 800 Mbps.

The next step is to construct traffic scenarios to verify the behavior of applications with different QoS requirements, as latency (e.g. for voice) and throughput (e.g. for video). In the first traffic scenario, all applications generate packets continuously in time, in what characterizes CBR QoS level. In the second traffic scenario, applications generate On-Off traffic, characterizing a modeling of video streams and block transfers, called here VBR QoS level. Streaming and block transfer are the applications used in these two traffic scenarios.

Streaming applications are modeled with fixed frames size (82,000 bytes). Each frame contains packets with 1000 flits generated every 3.3 ms, corresponding thus to a 200 Mbps rate. Considering that the link rate is 800 Mbps, the relative offered load is 25%. This workload is equivalent to 10 HDTV channels. Each video generator IP transmits 10 frames.

Block transfer applications are modeled with 40-flit packets. This is equivalent to 80 digitalized non-compacted voice channels. Each voice generator transmits 4000 packets. Figure 4 summarizes both scenarios, illustrating how both applications are modeled, using fixed injection rate and ON-OFF. Table 2 summarizes the traffic modeling adopted in the experiments.

Figure 5 illustrates the spatial distribution of packets. Note that almost all network nodes generate voice traffic using a complement distribution, and characterizing *noise* traffic. The goal of generating noise traffic is to disturb the QoS flows. Nodes 24 and 39 generate the two QoS flows having as targets node 60 and node 52 respectively. The adoption of this placement increases the number of flows in the network bisection [13].



(a) Scenario 1 - CBR

(b) Scenario 2 - VBR

Figure 4 – Traffic generation scenarios used in the experimental setup.

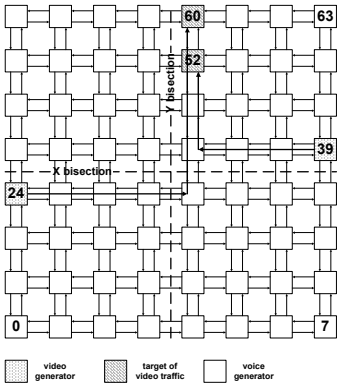


Figure 5 - Spatial distribution of packets used in the experimental setup.

6. PERFORMANCE RESULTS AND DISCUSSION

Latency (for voice and video) and frame inter-arrival time (for video) were the performance figures considered for evaluation purposes. In the case of voice traffic, each packet latency is computed considering the interval between its creation timestamp and the last flit arrival on the target node, as proposed in [13]. Latency values for each video frame are computed considering the interval between the creation timestamp of the first packet of a frame and the last flit arrival of the last packet of the same frame on the target node. Frame inter-arrival time accounts for the interval between each frame arrival on the target node. Table 3 summarizes performance results for voice and video applications, according to the adopted figures.

The adoption of On-Off traffic modeling (VBR) allows latency reduction of voice packets and video frames. This fact occurs due to the interval between bursts, which allows delivery of blocked packets. Such results for different traffic models for the same application shows that correct modeling the application, the NoC can deliver the required performance.

When QoS flows, video streams, are prioritized (Scenario 3), voice packets decrease their performance (average latency increases) and video frames are delivered with almost no standard deviation in latency (jitter). This is correct, since the network with priority mechanisms transmit all prioritized packets first.

Note that the frames inter-arrival time for the three scenarios meet the QoS requirements, since the frames are received with the same injection rate (constant throughput). The adoption of a different traffic model (Scenario 2) and a different NoC implementation (Scenario 3) improves the standard deviation of this performance figure. For video frames, a small inter-arrival time standard deviation is acceptable, since this corresponds to a small lost, not perceptible by the user. However, to meet hard-real time constraints, this performance figure must be zero. In this case, mechanism as circuit-switching can be adopted.

Figure 6 illustrates the latency spreading of voice traffic, for the three scenarios adopted. Voice packets in scenario 1, with CBR traffic, presents an important latency spreading, since these small packets (40-flits) compete with large streaming packets (1000-flits). The application of On-Off model (VBR) to voice traffic, Scenario 2, reduces the latency spreading.

Table 2 – Video and voice traffic modeling.

Application	Delivery requirement	Packets size	Traffic Modeling	
			Scenario 1 - CBR	Scenario 2 - VBR
Video: 10 channels HDTV	Real-time streaming	1000 16-bit flits	25% constant injection rate (200 Mbps)	On-Off On state: 25%
Voice: 80 channels	Real-time block transfer	40 16-bit flits	25% constant injection rate (200 Mbps)	Pareto On-Off On state: 25%

Table 3 – Performance results for voice packets and video frames.

Application	Scenario 1 (CBR)		Scenario 2 (VBR)		Scenario 3 (VBR + priority)	
	Mean	Std deviation	Mean	Std deviation	Mean	Std deviation
Average latency for voice packets (ck cycles)	137,755	117,203	1,775	8,389	181,054	331,520
Average latency for video frames (ck cycles)	162,114	80	88,696	433	82,680	13
Frames inter-arrival for video (ck cycles)	163,998	115	164,997	37	165,000	3

The reason for latency spreading reduction in Figure 6 is the Off-period between bursts, allowing blocked packets to be transmitted. Note that 95% of packets are transmitted with the minimal latency. If this same model is applied to voice traffic, Scenario 3, but higher priority video flows are inject competing with it, the behavior is similar to the first scenario. This is an expected result, since these small packets have low arbitration priority to be transmitted.

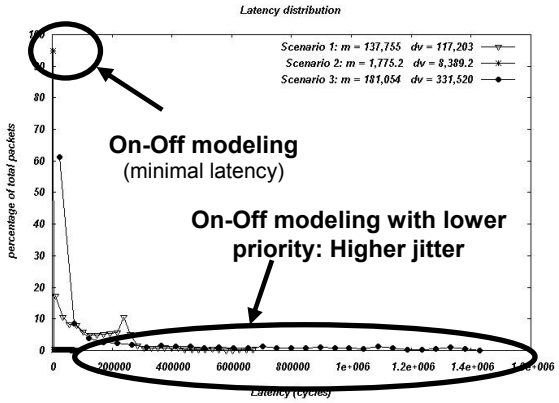


Figure 6 – Latency distribution for voice packets.

7. CONCLUSIONS AND FUTURE WORK

This work presented application driven traffic modeling for NoCs. Applications can be characterized according to their delivery requirements (e.g. real-time streaming and block transfer) and QoS service levels (e.g. CBR and VBR). In terms of modeling, three methods were presented. Constant injection rate is the most commonly used, due to its ease of implementation. This method does not model applications correctly, since most applications have variable injection rates. Probabilistic methods are normally used in simulation for applications with variable rates, with lower number of packets transmitted, but not with the same accuracy level provided by traffic traces. Trace-based traffic models are more suitable for emulation, where a huge number of packets is feasible.

Results regarding the use of constant injection rate and On-Off models indicate that On-Off models imply a better performance. This fact occurs due to the high interval between bursts observed in On-Off modeling, when compared to constant injection rate models. In the latter, packets are continuously transmitted in time. The utilization of On-Off modeling allowed transmitting packets with lower latency and jitter.

Future works include generation of more complex traffic models (e.g. self-similar, Gaussian noise) and experiments with real traffic traces.

8. ACKNOWLEDGMENTS

This research was supported partially by CNPq (Brazilian Research Agency), project 307655/2003-2.

9. REFERENCES

[1] Keutzer, K. et al. “System-Level Design: Orthogonalization of Concerns and Platform-Based Design”. IEEE Transactions on Computer-Aided Design, v.19(12), 2000, pp. 1523-1543.

[2] Benini, L. De Micheli,G. "Networks on chips: a new SoC paradigm". IEEE Comp., v.35(1), 2002, pp. 70-78.

[3] Bolotin E. et al. “QNoC: QoS architecture and design process for network on chip”. JSA, v.50(2-3), 2004, pp. 105-128.

[4] Genko, N. et al. “A Complete Network-On-Chip Emulation Framework”. In: DATE´05, pp. 246-251.

[5] Ost, L. et al. “MAIA - A Framework for Networks on Chip Generation and Verification”. In: ASP-DAC´05, pp. 49-52.

[6] Hu, J.; Marculescu, R. “Dyad – Smart routing for networks on chip”. In: DAC’04, pp. 260-263.

[7] Yum, K. et al. “Investigating QoS Support for Traffic Mixes with the MediaWorm Router”. In: 6th HPCA´00, pp. 97-106.

[8] Santi, S. et al. “On the Impact of Traffic Statistics on Quality of Service for Networks on Chip”. In: ISCAS´05, pp. 2349-2352.

[9] Harmanci, M. et al. “Quantitative Modelling an Comparison of Communication Schemes to Guarantee Quality-of-Service in Networks-on-Chip”. In: ISCAS´05, pp. 1782-1785.

[10] Kwok, T. “ATM – The New Paradigm for Internet, Intranet & Residential Broadband Services & Applications”. Prentice Hall PTR, 1998, 352p.

[11] Giroux, N.; Ganti, S. “Quality of Service in ATM Networks”. Prentice Hall PTR, 1999, 252p.

[12] Varatkar, G.; Marculescu,, R. “On-Chip Traffic Modeling and Synthesis for MPEG-2 Video Applications”. IEEE Transactions on VLSI Systems, v.12(1), 2004, pp. 108-119.

[13] Tedesco, L. et al. “Traffic Generation and Performance Evaluation for Mesh-Based NoCs”. In: SBCCI’05, pp. 184-189.

[14] Pande, P. et al. “Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures”. IEEE Transactions on Computers, v.54(8), 2005, pp. 1025-1040.

[15] Leland, W. et al. “On the Self-Similar Nature of Ethernet Traffic”. IEEE/ACM Transactions on Networking, v.2, 1994, pp. 1-15.

[16] Moraes, F. et al. “Hermes: an Infrastructure for Low Area Overhead Packet-switching Networks on Chip”. Integration, the VLSI Journal, v.38(1), 2004, pp. 69-93.