Analysis and Synthesis of Cycle-Accurate On-Chip Traffic with Long-Range Dependence

Antoine Scherrer, Antoine Fraboulet, Tanguy Risset

Laboratoire de l'Informatique du Parallélisme

École Normale Supérieure de Lyon
Unité Mixte de Recherche CNRS-INRIA-ENS LYON-UCBL n° 5668

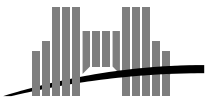
Analysis and Synthesis of Cycle-Accurate On-Chip Traffic with Long-Range-Dependence

Antoine Scherrer
Antoine Fraboulet
Tanguy Risset

November 2005

Research Report N° 2005-53

École Normale Supérieure de Lyon
46 Allée d'Italie, 69364 Lyon Cedex 07, France
Téléphone : +33(0)4.72.72.80.37
Télécopieur : +33(0)4.72.72.80.80
Adresse électronique : `lip@ens-lyon.fr`



Analysis and Synthesis of Cycle-Accurate On-Chip Traffic with Long-Range-Dependence

Antoine Scherrer
Antoine Fraboulet
Tanguy Risset

November 2005

Abstract

In this report we propose a stochastic traffic analysis and synthesis methodology adapted to on-chip network traffic. This work confirms and extends recent works (Marculescu et al. [13]) by providing more precise (cycle-accurate) simulations. Our framework based on System-C simulations is able to model precisely the latency of each request-acknowledge transaction or aggregated throughput taking into account its bursty behavior. Our experiments show that on-chip traffic is non-stationary, with long-range-dependence property and that simulation platforms of systems-on-chip will now need to use advanced statistical models for traffic simulation.

Keywords: on-chip traffic, stochastic analysis, long-range-dependance, process synthesis

Résumé

Ce rapport de recherche présente une méthodologie d'analyse et de synthèse de trafic adapté au trafic d'information transitant sur une puce. Ce travail confirme et étends les travaux de Marculescu et al. [13] en utilisant des simulations plus précises (cycle-près). Notre environnement de simulation est basée sur SystemC et nous permet de modéliser avec des processus stochastique le trafic, en tenant compte des caractéristiques du premier (distribution de probabilité) et du second (fonction de covariance) ordres statistique. Les expérimentations montre la présence de longue mémoire dans le trafic au niveau cycle, et ainsi que l'utilisation de modèle stochastique avancés est nécessaire pour faire une modélisation pertinente.

Mots-clés: trafic sur puce, processus stochastiques, longue mémoire, synthèse de processus

1 Introduction

Systems on chip (SoC) are now commonly used in embedded systems for multimedia and telecommunication applications. The computing power required by emerging applications running on mobile terminals, such as video on mobile phone for instance, has pushed the development of more complex SoC infrastructure so called multi-processor SoC (MPSoC) typically composed of a number of master components (processors or DMA for hardware accelerators) connected to a network on chip (NoC) or a hierarchy of busses.

The advent of on-chip network communications has significantly increased the design complexity of such systems with some hard problems related to parallelism: non-determinism, memory and cache coherency, efficient workload distribution and network contention. Solving these problems during the short time available for design requires fundamental improvements in design methodologies. The most important shift is the setting of a refinement design methodology allowing designers to explore design space at various levels of precision. These levels, called *transaction*, *bus-accurate*, *synthesizable*, etc. allow the designer to check quickly that performances related to various metrics are achieved before writing the complete description of the system.

During design space exploration, simulation time is a major problem. There are two run-time behaviors very difficult to model at a high level: cache behavior and network contention. Precise simulation of these two behaviors can only be done with a low-level description of the components. This means hours (sometimes days) of simulation for a single execution or, as it is usually preferred, the use of extremely expensive hardware emulators.

Reducing simulation time can be achieved by a clever analysis of the behavior of the system during execution. We are interested in the simulation of on-chip network behavior and performance evaluation. This is usually done by replacing each component by traffic generators.

Using traffic generator in a simulation platform involves the following steps: *i*) collecting simulation traces by observing the behavior at the interface of each master component, *ii*) building traffic models as close as possible to these traces, *iii*) writing traffic generator for each master component based on these models, and *iv*) inserting them in the simulation platform in place of original components. Most recent traffic generation methods use stochastic models. Statistical analysis and synthesis of on-chip traffic are difficult because this traffic usually presents complex statistical behavior. The precision of the simulation and the possibility of integrating the whole process in an automatic (or at least semi-automatic) framework are important parameters for evaluating the usefulness of a traffic generation environment.

In this paper we use stochastic traffic generators which are generated from trace analysis and used in cycle-accurate simulations and performance evaluation of network-on-chip. Our main goal here is to validate the statistical properties of the traffic generators with regards to the statistical properties of original traces they were generated from. Especially, we want the *bursty* behavior of traffic to be taken into account. There are two contributions in this paper:

• We present a complete methodology for stochastic on-chip traffic analysis and synthesis at the cycle-accurate level.

• We perform cycle-accurate simulations in SystemC of a complete SoC including a NoC running JPEG2000 and MP3 applications. We extend the result of [13] to cycle-accurate behavior: i) on-chip traffic is non-stationary and must be split into phases and ii) on-chip traffic flows contain long range dependent behavior that must be taken into account when synthesizing traffic.

The paper is organized as follows. In Section 2 we present the different techniques of traffic generation. Section 3 presents the flow that we propose for analyzing and synthesizing on-chip traffic. Section 4 presents our experimental results that highlight the points mentioned above.

2 On-chip Traffic Generators: Related Work

Traffic generators can be separated into two main categories: the *deterministic approach*, in which traffic is produced using a finite state machine (FSM) configured by the IP designer or using a previous simulation trace, and the *stochastic approach*, in which the traffic is produced by

parameterized non-deterministic process. In this section we shortly review these two methods with their pros and cons for on-chip traffic modelling.

2.1 Deterministic traffic generation

A deterministic traffic generator (TG) [5, 8, 7] is derived from real simulation traces or written from scratch by IP designers. Such a TG can generate accurate transactions in time, burst size and idle time that match the behavior of an IP. The advantages of such traffic generators are their precision and the speedup factor that they can achieve compared to the complete IP simulation. One limitation of this approach is that the length of the simulation is limited by the length of input traces used, furthermore, the main drawback of the deterministic TG is that it cannot handle behaviors that are dependent on input data sets.

2.2 Stochastic traffic generation

The major part of NoC performance evaluation is currently done using random sources [14, 12, 6, 10]. These works mainly focus on the evaluation of the NoC in its early stage of development, and on its performance under random traffic. However none of these works propose a *fitting procedure* to determine the adequate statistical parameters that should be used to simulate traffic. This is what we present in Section 3).

A stochastic modelling considers the traffic as realizations of stochastic processes, of which we will consider the following two main characteristics :

• **First order statistics** represent how the values of a process are distributed. They are fully described by the probability distribution function (PDF), which corresponds basically to the frequency of apparition of each possible value taken by the process.

• **Second order statistics** represent how values of the process are correlated at all possible lag. The covariance function $\gamma(l) = \mathbb{E}(X(t)X(t+l)) - \mathbb{E}(X(t))\mathbb{E}(X(t+l))$ of a process $\{X(t)\}_{t\in\mathbb{N}}$ holds such an information, this is how random variables distant of l samples influence each other ($\mathbb{E}$ is the expectation).

We present now the most important stochastic processes used to model these series. We classify them into the *classical models* (no long-range-dependence) and the *long-range-dependent models*. All of them are stationary processes models, that is to say their statistical properties (mean and covariance function) do not change in time.

2.2.1 Classical Models

• IID (Independent Identically Distributed) processes are the most common way of obtaining random number series. Each random variable of the process is independent of the others. As a consequence, bursty behavior cannot be taken into account and hence, IID processes cannot be used as an accurate representation of traffic.

• ARMA (Auto Regressive Moving Average) processes correspond to the filtering of IID processes, introducing some correlations between random variables. The covariance function decays exponentially and is therefore only able to reproduce small-scales correlations. Bursty behavior can hence be taken into account at small scales but not at large scales. For large scale bursty behavior, long-range-dependent processes have been introduced [13].

2.2.2 Long-range-dependant models

Long-range-dependence (LRD) [3] is a very significant characteristic because it has been shown to have very important impact on network performance [13, 9]. Not taking LRD into account leads to a dramatic under-estimation of needed buffer sizes in the network. LRD behavior has been widely found on Internet traffic [4] and demonstrated for on-chip multimedia traffic at the coarse-grain level [13]. The important difference with the work presented in [13] is that our simulation occurs at a cycle-accurate level leading to important results concerning some problems mentioned above (cache behavior, network contention).

LRD processes are characterized by a slowly decaying covariance function that is no more summable. Data are correlated over a non-limited range of time lags and this property results in a scale invariance phenomenon. No characteristic time scale can be identified in the process, they are all equivalent for describing its statistics: the part resembles the whole and vice e versa. This is why LRD is also called Self-Similarity[1]. LRD implies in the scope of traffic analysis a *bursty* behavior over a range of time scales. Long-range-dependence is fully described by the Hurst parameter H that controls the decaying velocity of the covariance function, and therefore how much the process is self-similar. H varies from 0.5, which corresponds to the IID case (no LRD), to 1. If H is near or above 1, LRD is not sufficient to model the behavior, it can be the signature of the non-stationarity of the trace.

The LRD processes that we consider are the following:

• **T**he FGN (Fractional Gaussian Noise) is a common stationary LRD process. Its covariance function exhibits a power-law decay, whose exponent is directly related to the Hurst parameter. Originally it is a Gaussian process, the generation of non-Gaussian version of this process is discussed in Section 3.2.

• FARIMA (Fractionally Integrated ARMA) processes combine both the short-range correlations of an ARMA process and the long-range-dependence introduced by the fractional integration. Both behaviors have distinct parameters, basically an LRD parameter H is added to the ARMA parameters. This is a versatile model adaptable to many situations.

2.3 On-chip traffic formalism

The traffic produced by a component is a sequence of transaction. The k^{th} transaction is a 4-uple $(A(k), C(k), S(k), D(k))$ meaning in this order, target address, command (read or write), size of transaction and delay (number of cycles between two successive requests). This is illustrated in Figure 1. From this transaction sequence, we define the aggregated throughput $W_\Delta(k)$, which corresponds to the amount of bytes transfered in the time interval $[k\Delta \quad (k+1)\Delta]$. Δ is called the window size.

We will model the series $S(k)$, $A(k)$, $C(k)$, $D(k)$ and $W_\Delta(k)$ by means of stochastic processes. Hence in the following, $\{S(k)\}_{k\in\mathbb{N}}$ will represent the stochastic process of the successive transaction sizes.

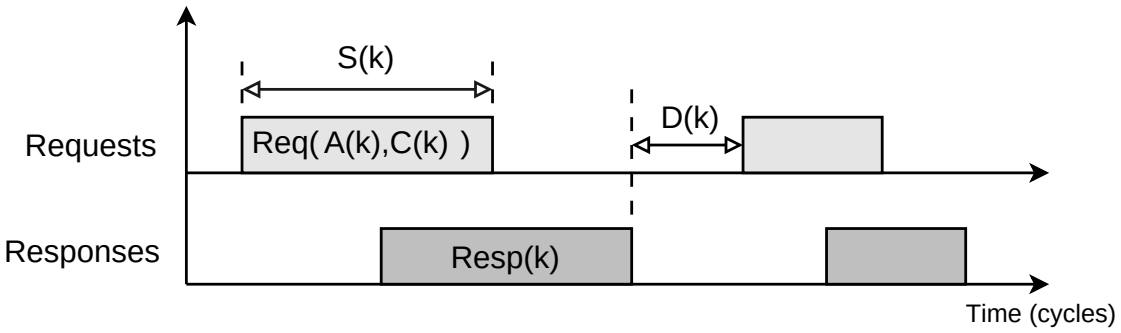


Figure 1: Traffic modelling formalism

3 Traffic Analysis and Synthesis

Generated traffic has to match the real execution to a certain extent. In the macro-network community (Internet traffic analysis for instance), it is now widely admitted the first order statistics

[1] Actually, LRD strictly corresponds to asymptotic second order self-similarity.

are not precise enough to model the behavior of the traffic. Long-range-dependence behavior must be taken into account when analyzing or synthesizing traffic. Our proposal is that traffic generators should be automatically synthesized and built from the analysis of execution traces to find out which statistical laws can be used.

We present in this section our analysis and synthesis flow for building multi-model traffic generators that can be used to replace an IP in cycle-accurate NoC performance evaluation.

3.1 Analysis Flow

Figure 2 shows the steps of our framework. We start from an initial cycle-accurate simulation of a system including one or several masters. During the simulation we record the signal variations at each interface of the master components in a VCD (Value Change Dump) trace file.

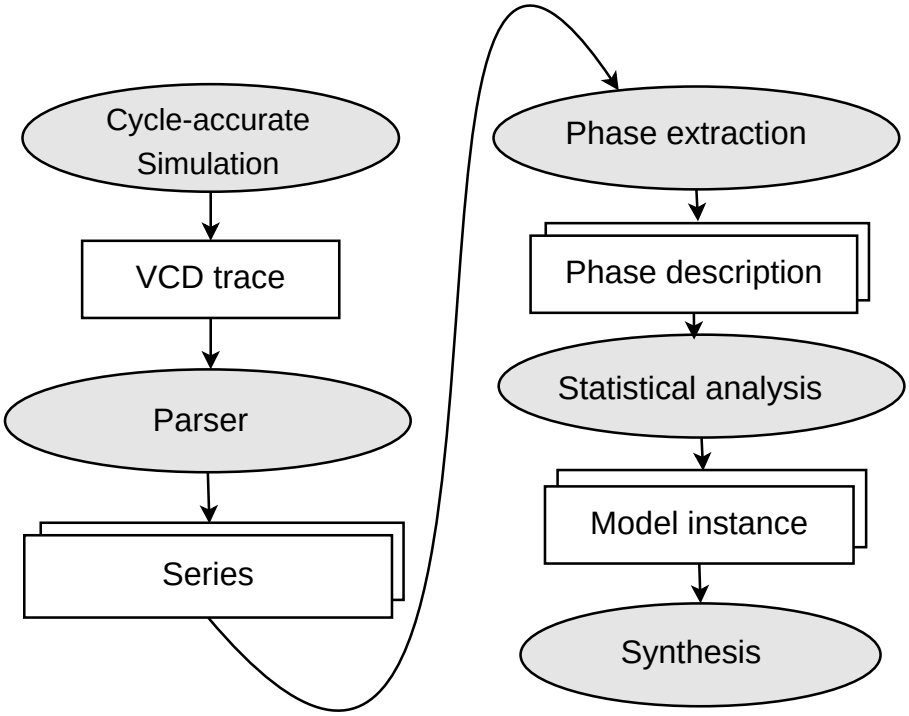


Figure 2: Traffic analysis and synthesis flow

A dedicated parser extracts from this trace all series introduced in 2.3.

• **Phase extraction:** As mentioned above, these series are usually non stationary, that is why we need to cut them into reasonably stationary part and apply statistical analysis to each of them. This splitting is currently done manually but we are studying solutions to automate this part using the hypothesis that our traces are piecewise stationary.

• **Statistical analysis** is then performed on each extracted phase by a semi-automatic fitting procedure that adjusts the first and second statistical orders [11].

The probability distribution function (PDF) (first statistical order) can be either fitted to some classical distributions (Gaussian, Exponential, Gamma, Log-Normal, ...) or kept as they are (the model is then the probability of apparition of each value of the process). The fit is done using Maximum Likelihood Expectation and a χ^2 goodness-of-fit test is used to compare and evaluate all different solutions.

The covariance function (second statistical order) can be fitted to an ARMA (short range correlations only), Fractional Gaussian noise (long-range-dependence only) or a FARIMA (both short and long-range correlations). IID processes do not need a covariance fit. We use a wavelet-based

estimation of the Hurst parameter [2] widely adopted in the network traffic analysis domain. This type of fit is new, it was never proposed for cycle-accurate NoC simulation. For FARIMA fitting, we first have to remove long-range-dependence in the Fourier domain, and then run a standard ARMA estimation procedure [11].

This fitting procedure (including the phase extraction) can be applied to the series presented in Section 2.3, with possibly different models. The choice of what model to fit is currently done manually, but we are studying some ways of automating it. The output for this procedure is a phase description file which contains each chosen stochastic processes and fitted parameters for each phase, and for each transaction process.

Currently, we use this complete fitting procedure either for the delay and size processes or for the aggregated throughput one which combines both. In this paper we will only show results and comments about the aggregated throughput.

3.2 Traffic synthesis

We have implemented a generic cycle-accurate traffic generator working as a multi-phase random transaction generator. Transactions (the 4-uple $(A(k),\ C(k),\ S(k),\ D(k))$ introduced in Section 2.3) are randomly generated according to the phase description file and a sequencer is in charge of switching between phases.

Note that the TG issues a request on the network only if possible (if the network is ready). At the k^{th} response reception, the TG waits $D(k)$ cycles before attempting to issue the next request, hence taking into account the network latency. A realistic traffic is therefore produced on the network. We point out that our TG is flexible enough to be extended to other communication schemes.

The random number generators we use are independent of the traffic generator itself. For IID and short range dependent processes, implementation is straightforward, as well as for the Gaussian LRD case [3]. The synthesis of non-Gaussian long-range-dependent processes is however not a simple mathematical issue. Basically we have first to generate a Gaussian LRD process and then to take a function of this process in order to get the right probability distribution function. There exits two approaches. The one presented in [11] guarantees precise behavior but suffers some restrictions in the shape of the probability distribution function and the one used in [13] only guarantees asymptotic validity of the covariance function, but has less restrictions and is faster. We use the first approach when the probability distribution function meets the restrictions, otherwise we switch to the second one.

4 Experimentation

We now present some experiments showing that both the multi-phase behavior of application's traffic and the long-range-dependence property during some of the phases.

4.1 Experimental Platform

We use an open source, SystemC-based, cycle-accurate and bit-accurate simulation environment: SocLib [1]. It also contains cycle-accurate models for real network on chip that can be used for network parameters estimations once the traffic generator have been synthesized.

The components of the platform are: a MIPS R3000 processor (with its associated data and instruction cache), two on-chip memories, and a component used for displaying output (referred to as TTY). All these components are connected via VCI ports to a simple network. The internal architecture of this network is not precisely simulated, only the latency and bandwidth can be parameterized.

The application running on the MIPS, in addition to bootstrapping information, is composed of the C program cross-compiled with GCC to a MIPS target.

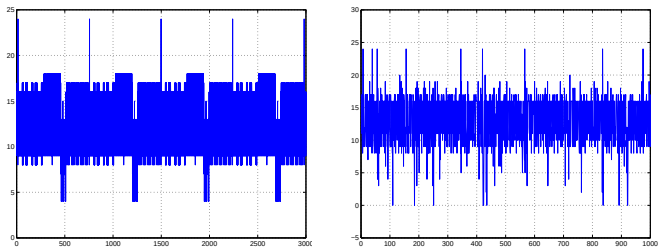
We used two embedded programs: an implementation of the JPEG2000 image decompression standard processing a 256x256 image, and a streaming MP3 audio decoder, processing a 4KB stream. Table 1 shows the simulation parameters.

	jpeg2000	mp3
Simulated cycles (Millions)	91	24,3
Icache geometry (Lines $*$ Words)	32x8	32x8
Dcache geometry (Lines $*$ Words)	32x8	32x8
# Transactions (Millions)	4,3	1,2
# Data transferred (Mbytes)	84,4	24

Table 1: Simulation parameters

4.2 Results about phase decomposition

We have extracted the aggregated throughput $\{W_\Delta\}_{k\in\mathbb{N}}$, with the window size Δ fixed to 128 cycles. In this data, we manually identified different behaviors corresponding to different phases of the algorithm. For the JPEG2000 code, we have clearly identified the so-called "Tier1" arithmetic decoder (T1) and the Inverse discrete wavelet transform (IDWT). The figure 3 shows respectively a zoom of the aggregated throughput for IDWT (a) and T1 (b) parts of the algorithm. One can see that the T1 part is clearly a reasonable candidate for stochastic modelling.



(a) (b)

Figure 3: (a) shows the IDWT part, and (b) the T1 part of the aggregated throughput in 128 cycles windows

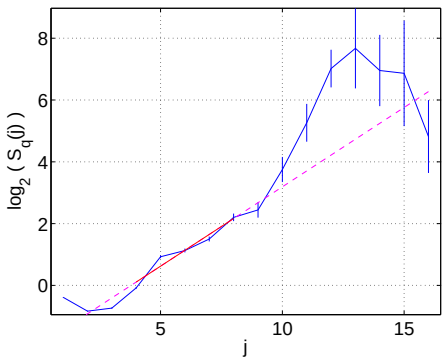
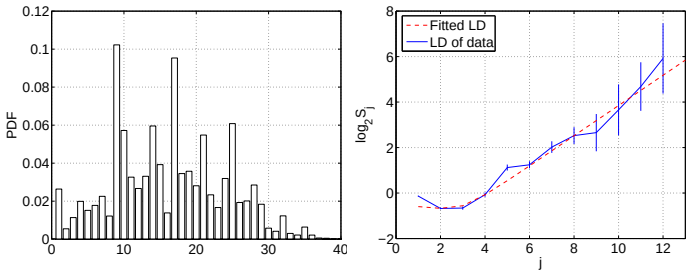


Figure 4: Log-diagram (LD) of the complete T1 phase of jpeg2000

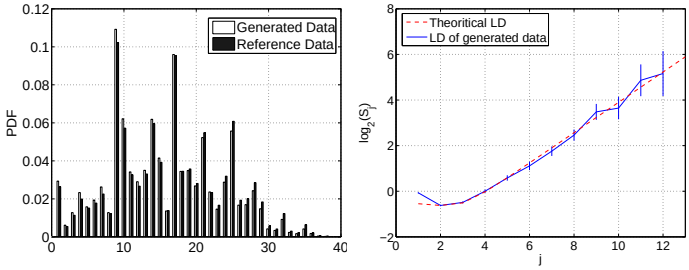
We estimated the covariance function over the complete T1 phase and Figure 4 shows representation of it as a so-called *log-diagram*. This can be viewed as a spectral log-log representation of the covariance, as a function of time scales. In such a diagram, long-range-dependence results in a straight line behavior over a range of scales. The slope of the linear regression α is directly related to the Hurst parameter ($H = (\alpha + 1)/2$), and especially if the slope is 0 (horizontal line), $H = 0.5$ and there is no LRD.

Over scales $2^4 - 2^{11}$, a regular long-range-dependent behavior is observed, which will be discussed in the next section. At higher scale range, we observed a line which would lead to $H > 1$, being interpreted as non-stationarity in the complete trace. This shows clearly that the trace should not be analyzed completely, but rather cut into reasonably stationary part for the results to make sense. So we manually identified stationary parts and next section will show statistical results on one of the extracted piece.

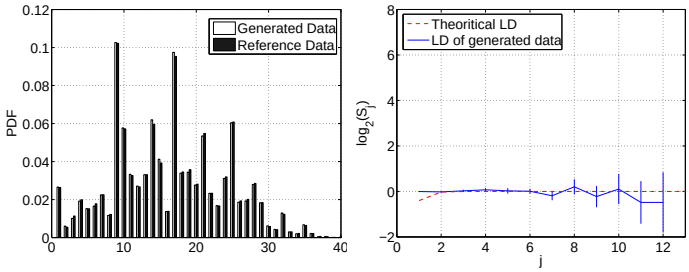
On the MP3 code, similar results were found not presented here. We can identify each frame decoding, and inside each of these frames, we have been able to identify a randomly-varying part corresponding to the heart of the algorithm, and a very regular write back to memory part.



(a) Original trace - PDF (b) Original trace - Covariance



(c) Synth. trace (LRD) - PDF (d) Synth. trace (LRD) - Cov.



(e) Synth. trace (IID) - PDF (f) Synth. trace (IID) - Cov.

Figure 5: Comparison of original and synthetic (generated by TG) aggregated throughput trace for a phase in the T1 part of the jpeg2000 application. Left column is the first order statistics, right column is the Log-diagram of the covariance function.

4.3 Results about stochastic modelling

Figure 5 shows results for a phase extracted in the T1 part of JPEG2000 application on the SoC platform mentioned above. Original traffic and synthetic traffic (obtained from synthesized TG) are analyzed. As a summary of the discussion below, these results show clearly that our TG precisely follow first and second order statistic properties of traces analyzed while this is not the case for TG based on classical models.

• **First order statistics** are presented in Figure 5-(a), which shows the probability distribution function. One can see that the distribution does not look like any classical distribution, so we decided to keep it as it is, that is the parameters are the probabilities of apparition of each value.

• **Second order statistics** are depicted in the Figure 5-(b). The long-range-dependence is clearly attested in the range of scale 2^4-2^{11} (linear behavior), which means over a range of about 250000 cycles. The trace has been fitted to a FARIMA process, so that the short range correlation (slight decrease of the curve at small scale) is also fitted. We estimated $H = 0.85$, which is a classically observed value in computer networks traffic analysis.

• **Synthetic trace generation:** We generated a realization of the fitted process (FARIMA and custom probability distribution function) as explained in Section 3.2. Figure 5-(c,d,e,f) present both the PDF (left figures) and the covariance (right figures) of a realization with the same size as the input trace with LRD (c,d) and without LRD (e,f). We can see that first and second statistical orders characteristics are in agreement with the one of the original trace in the LRD case, so that statistical properties of first and second order have been captured. In the case of IID process, we clearly see that only first order statistics are reproduced, leading in major difference in the resulting traffic shape.

5 Conclusion and Future Work

In this paper, we have presented an on-chip traffic analysis and synthesis flow that can use several traffic models for NoC traffic simulation. In particular, we introduced the use of long-range-dependence stochastic models that can be used to generate bursty traffic in cycle-accurate simulations. This LRD traffic generator bridges the gap between classical random generators, for which each transaction is independent, and deterministic traffic generators.

Simulation phase splitting and TG generation have been illustrated using the SocLib cycle-accurate simulation environment. Results show that first and second order statistical properties are precisely simulated by our traffic generator, hence taking into account the *bursty* behavior of traffic.

Future work is currently going on for automating the splitting procedure and validating this flow for NoC architecture prototyping, including the optimization of buffer sizes in NoC routers.

References

[1] Soclib simulation environment. On-line, available at http://soclib.lip6.fr/, 2005.

[2] P. Abry and D. Veitch. Wavelet analysis of long-range dependent traffic. *IEEE Trans. on Info. Theory*, 44(1):2–15, January 1998.

[3] P. Doukhan, G. Openheim, and M. Taqqu. *Theory and Applications of Long Range Dependance*. Birkhäuser, 2002.

[4] A. Feldmann, A. Gilbert, W. Willinger, and T. Kurtz. The changing nature of network traffic: Scaling phenomena. *ACM/SIGCOMM Computer Communication Review*, 2(28), April 1998.

[5] N. Genko, D. Atienza, G. De Micheli, J. M. Mendias, R. Hermida, and F. Catthoor. A complete network-on-chip emulation framework. In *DATE 05*, pages 246–251, 2005.

[6] Kanishka Lahiri, Sujit Dey, and Anand Raghunathan. Evaluation of the traffic-performance characteristics of system-on-chip communication architectures. In *VLSID '01*, page 29. IEEE Computer Society, 2001.

[7] Mirko Loghi, Federico Angiolini, Davide Bertozzi, Luca Benini, and Roberto Zafalon. Analyzing on-chip communication in a mpsoc environment. In *DATE 04*, page 20752, 2004.

[8] Shankar Mahadevan, Federico Angiolini, Michael Storgaard, Rasmus Grøndahl Olsen, Jens Sparsø, and Jan Madsen. A network traffic generator model for fast network-on-chip simulation. In *DATE 05*, pages 780–785, 2005.

[9] Kihong Park and Walter Willinger, editors. *Self-Similar Network Traffic and Performance Evaluation*. John Wiley & Sons, 2000.

[10] Santiago Gonzalez Pestana, Edwin Rijpkema, Andrei R¿dulescu, Kees Goossens, and Om Prakash Gangwal. Cost-performance trade-offs in networks on chip: A simulation-based approach. In *DATE 04*, page 20764, 2004.

[11] Antoine Scherrer, Nicolas Larrieu, Philippe Owezarski, Pierre Borgnat, and Patrice Abry. Non gaussian and long memory statistical characterisations for internet traffic with anomalies. Technical report, LIP - ENS Lyon, 2005.

[12] R. Thid, M. Millberg, and A. Jantsch. Evaluating NoC communication backbones with simulation. In *21th IEEE Norchip Conference*, Riga, November 2003.

[13] Girish Varatkar and Radu Marculescu. On-chip traffic modeling and synthesis for mpeg-2 video applications. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 12(1):108–119, 2004.

[14] Daniel Wiklund, Sumant Sathe, and Dake Liu. Network on chip simulations for benchmarking. In *IWSOC*, pages 269–274, 2004.