Journal of Systems Architecture 61 (2015) 197–209

Contents lists available at ScienceDirect

Journal of Systems Architecture

journal homepage: www.elsevier.com/locate/sysarc

A control-based methodology for power-performance optimization in NoCs exploiting DVFS ☆

Davide Zoni *, Federico Terraneo, William Fornaciari

Politecnico di Milano-DEIB, Via Ponzio 34/5, 20133 Milano, Italy

ARTICLE INFO

Article history:
Received 25 September 2014
Received in revised form 2 February 2015
Accepted 10 April 2015
Available online 18 April 2015

Keywords:
Network-on-Chip
Multiprocessors
DVFS
DFS
Power-performance tradeoff

ABSTRACT

Networks-on-Chip (NoCs) are considered a viable solution to fully exploit the computational power of multi- and many-cores, but their non negligible power consumption requires ad hoc power-performance design methodologies. In this perspective, several proposals exploited the possibility to dynamically tune voltage and frequency for the interconnect, taking steps from traditional CPU-based power management solutions. However, the impact of the actuators, i.e. the limited range of frequencies for a PLL (Phase Locked Loop) or the time to increase voltage and frequency for a Dynamic Voltage and Frequency Scaling (DVFS) modules, are often not carefully accounted for, thus overestimating the benefits. This paper presents a control-based methodology for the NoC power-performance optimization exploiting the Dynamic Frequency Scaling (DFS). Both timing and power overheads of the actuators are considered, thanks to an ad hoc simulation framework. Moreover the proposed methodology eventually allows for user and/or OS interactions to change between different high level power-performance modes, i.e. to trigger performance oriented or power saving system behaviors. Experimental validation considered a 16-core architecture comparing our proposal with different settings of threshold-based policies. We achieved a speedup up to 3 for the timing and a reduction up to 33.17% of the power $*$ time product against the best threshold-based policy. Moreover, our best control-based scheme provides an averaged power-performance product improvement of 16.50% and 34.79% against the best and the second considered threshold-based policy setting.

1. Introduction

The many-core revolution allows to exploit the available silicon area by providing highly parallel architectures to run multiple applications. At the same time the bus-based interconnects show their lacks to support the traffic generated by parallel applications. In this perspective on-chip networks are emerging as one of the promising solutions to face the communication issues in such architectures. However, the growing of communication requirements due to the increasing number of cores integrated on a chip [6], motivates the need to account and optimize the NoC power consumption. Results in [35,29] point out a contribution of the NoC from 10% up to 30% of the whole chip power. Such impact is depending on the actual multi-core configuration, i.e. message passing or hardware coherence support, complexity of the CPUs and of the NoC. Nevertheless, the power budget is expected to remain very limited due to packaging constraints, thus the challenge becomes to design high-bandwidth low latency and power-efficient NoCs [28]. In this scenario, Dynamic Voltage and Frequency Scaling (DVFS) and Dynamic Frequency Scaling (DFS) schemes for NoC routers represent two viable solutions to dynamically adapt the performance to the application needs. However, most of the literature proposals on this topic rely on heuristics to dynamically change frequencies, thus without guaranteeing any properties with respect to the whole system. Moreover, the proposed methodologies consider ideal actuators without accounting for their transient behavior or their overhead, thus overestimating the actual benefits.

This paper proposes a complete power-performance control-based scheme for NoC routers, which is able to adapt to

* Corresponding author. Tel.: +39 0223993504.
E-mail addresses: davide.zoni@polimi.it (D. Zoni), federico.terraneo@polimi.it (F. Terraneo), william.fornaciari@polimi.it (W. Fornaciari).

http://dx.doi.org/10.1016/j.sysarc.2015.04.004
1383-7621/

the actual network load, while ensuring the Lyapunov stability. The interested reader can find a complete discussion related to stability in [7].

Our proposal has been validated against threshold-based policies as the ones proposed in [24]. Moreover, we provide a framework to develop power-performance policies exploiting Dynamic Frequency Scaling actuators, carefully modeled to consider timing and energy overheads, that automatically set the router frequencies according to the actual load.

The rest of this section is organized in three parts. First of all a motivational example is provided in Section 1.1 focusing on the needs of the run-time power-performance policies. The novel contribution of the paper is discussed in Section 1.2, while the structure of the whole paper is outlined in Section 1.3.

1.1. Motivational example

The power-performance trade-off represents one of the most celebrated scenarios where two different opposed metrics must be tuned to impose the desired behavior to a system. The power-performance Pareto graphs have been also analyzed for on-chip interconnects due to the non-negligible power envelope of the NoC. However, few of the presented methodologies focus on valuable and guaranteed properties for the solution, i.e. stability, non-divergence and optimality. Heuristic approaches are the common ones and they can be very efficient in term of overheads, implementability and obtained results. This section tries to motivate the need for power-performance methodologies capable also to provide properties, such as stability, in order to produce really usable solutions. Moreover, we also show how the proposed framework can outperform the threshold-based policies from the power-performance viewpoint.

The rest of this section uses a contention metric (buffer utilization) per router to decide whether to increase the frequency of the router itself. The intuition behind such an approach comes from the fact that if a router is getting congested, it is due to the increase in traffic that uses a path through this router. We use the router frequency as the actuator to increase or decrease performance. Being both contention and frequency the figures of merit constituting the core of the proposal, they will be discussed in detail in Section 3 where the full methodology is also presented.

Fig. 1 shows the contention level as well as the dynamic power for router five considering a 16-core 2D-mesh where the NoC runs at 100 MHz (blue lines) and 1 GHz (red lines). In both scenarios the cores run at 1 GHz, executing the *qsort* benchmark taken from the *MiBench* suite [16]. It is obvious that the contention is lowered when the NoC operates at the higher frequency, a benefit paid in terms of time and power. The time required to complete the benchmark are roughly 7.4 ms and 12.5 ms for the high and low frequency NoCs, respectively. As anticipated, a router in the high frequency NoC requires five times more dynamic power than the low frequency one. Hence, it is crucial the availability of effective methodologies to trade-off power and performance. Moreover, since the contention is variable and highly depending on traffic, a methodology with run-time tuning is preferred to optimize the power consumption with acceptable impacts on performance. In this context, frequency scaling represents one of the most used actuators to achieve such a goal [24].

After providing the motivation for the adoption of run-time power-performance optimization methodologies for NoC-based multi-cores, the rest of the section addresses the need to guarantee some useful properties for the proposed solution. Starting from the same 16-core 2D-mesh architecture, using contention and frequency as proxy for performance and power, we focus on the physical limitations of the real actuators, i.e. it is not possible to change the frequency to arbitrarily high or low values. In light of this and without loss of generality, we focus on a 100 MHz–1 GHz frequency range for the actuator. Moreover, we consider a policy where the frequency (f_t) varies according to a function depending on the contention (C_t) of the router at time t as follow: $f_t = k * C_t$. The parameter k can be selected at design time to tune the sensitivity of the frequency actuator to the level of contention. In other words, given a level of contention, a higher k will result in a higher frequency. Fig. 2 shows the imposed frequency and contention of router five in the NoC considering k equal 0.04 and 5, respectively. In particular the proportional policy using $k = 0.04$ imposes frequencies that are always between the higher and lower boundaries, i.e. 1 GHz and 100 MHz, without saturation. Moreover, the imposed adaptation strategy of the frequency is reasonable, since the frequency increases when higher contention is observed on the router. For example between 4 and 8 ms the buffer filling level reaches 40 flits, thus the frequency is dynamically increased by the policy up to roughly 650 MHz. It is worth noticing that such policy takes around 7.8 ms to run the benchmarks, that is close to the 7.4 ms value obtained by keeping the router at the high frequency. On the other hand, the policy using $k = 5$ determines a completely

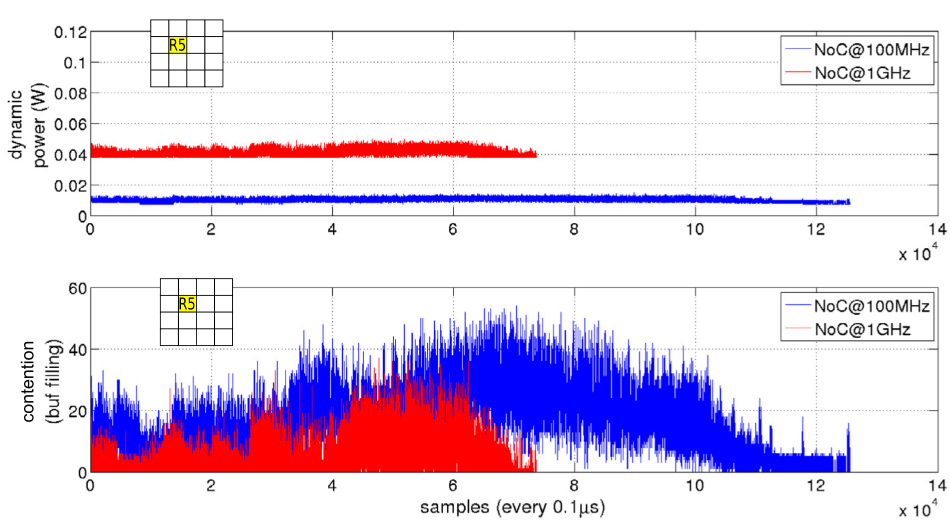


Fig. 1. Contention and dynamic power consumption for router five in a 16-core NoC multi-core, when the NoC runs at 100 MHz and 1 GHz respectively. The core runs always at 1 GHz. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

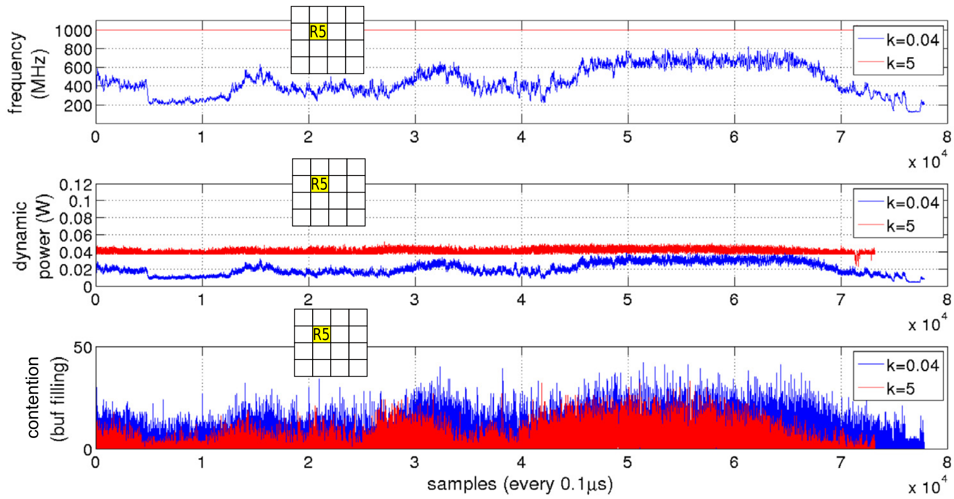


Fig. 2. Contention and frequency for router five in a 16-core NoC multi-core, where the NoC run uses a policy defined as $f_t = k * C_t$ with $k = 0.04$ blue line and $k = 5$ red line. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

different behavior imposing higher frequencies that most of the time saturate on the upper bound, i.e. 1 GHz. In this perspective even if the simulation is fast, the behavior of the policy is almost equal to the behavior of the high frequency NoC. This means that all the logic in the policy can be cut away, since it is equivalent to always imposing the higher frequency. The same considerations can be done for threshold based policies where the selection of the threshold levels must be accurately designed.

1.2. Novel contributions

The paper presents a control-based methodology and policies for power-performance optimization in NoC-based architectures. Each policy can run on its own but the approach allows to switch between different optimization policies while ensuring Lyapunov stability at the same time. Our approach is complementary to the optimal application mapping strategy targeting NoC-based multi-cores, a topic which is out of the scope of this paper (a good survey can be found in [32]). In a nutshell, the work encompasses three different contributions:

- *Run-time control-based power-performance NoC framework* – we propose a flexible control-based set of policies to improve the NoC power efficiency by setting the router frequency based on its contention. Such a policy outperforms threshold-based ones taken from [24]. Moreover, the scheme can be possibly tuned at high-level: the OS can choose if the policy to run should be more power-oriented or performance-oriented while guaranteeing non saturation of the frequency control. Note that the methodology does not require any OS intervention while such possibility represents an extension of this work that enhance its flexibility.
- *Accurate analytical formulation for both the NoC traffic and control policies* – The whole methodology focuses on both a controller design and an analytical model that binds the impact of the router frequency on its contention, i.e. the filling level of router buffers, that we use as a proxy for performance [24]. While the model of the process takes steps from [40] it has been greatly improved as described in Section 3.1.
- *Simulation framework with PLL and GALS support* – Starting from the work in [34] an enhanced simulation framework has been proposed improving the Phased-Locked Loop (PLL) timing model and providing a worst case PLL power consumption extracted from SPICE-level simulations. Moreover, an handshake resynchronizer is modeled and inserted into the simulation flow, thus allowing to partition the NoC in multiple voltage and frequency islands. This simulation framework is used to support the model identification and policy validation stages of the presented methodology. Moreover, it supports the assessment of any NoC-based policy using DFS actuators.

1.3. Paper structure

The rest of the paper is organized in four sections. First, a comprehensive state of the art review is provided in Section 2 with particular focus on the simulation frameworks to study the power-performance trade-off for NoC-based architectures. Section 3 presents the mathematical formulation of the proposed approach as well as the simulation flow. In particular, we discuss the dynamic models of the router behavior as well as the control-theoretical analysis to model the control policies. Then, we detail the main improvements to the simulation framework. The results are discussed in Section 4 focusing on three different aspects. First, we address the accuracy of the analytical model for the process against the real data generated using the cycle accurate simulator. Then, we discuss the single control-based policies against run-time heuristics and static policies applied to the whole system. Last, we present results focusing on the run-time framework allowing to switch between multiple control-based policies while ensuring stability. The ongoing activities and conclusions are drawn in Section 5.

2. Related works

Two different aspects related to DVFS mechanisms for Networks-on-Chip are discussed in this section. A review of the simulation frameworks supporting power-performance analysis of the on-chip networks together with an overview of the different power-performance design methodologies is presented.

2.1. Simulation frameworks

In literature appeared several simulation frameworks enabling power-performance exploration. Only few of them focus specifically on power-performance trade-off analysis in multi-core scenarios considering NoC routers exploiting DVFS/DFS. Another typical lack is not accounting for the PLL dynamics. Table 1 provides a classification of the proposed simulation frameworks mainly focused on the DVFS/DFS support, GALS design capabilities and availability of PLL models. The *SESC* simulator [31] provides cycle-accurate simulation of bus-based multi-core processors, based on the MIPS architecture. However, it does not support Network-on-Chip architectures as well as neither DVFS nor asynchronous NoC design.

The *Polaris* framework [33] permits power and area design space exploration for Network-on-Chip architectures without providing detailed power estimations for both processors and memory hierarchy. Moreover, it does not implement an heterogeneous NoC model to simulate run-time frequency changes.

Lis et al. [20] is meant to simulate large-scale architectures, and exploits parallel simulation on physical hardware with particular emphasis on the on-chip network. Although the framework enables power-performance trade-off analysis, it is missing a complete asynchronous on-chip network model, hence it is not possible to explore different GALS configurations for the interconnect, as well as the simulation of Dynamic Frequency Scaling driven by high level policies.

Bartolini et al. [2] proposes a framework to evaluate different thermal policies based on control theory. Even if they support the DVFS, the framework lacks of an accurate evaluation of the PLL model as well as the implementation of the DVFS for the on-chip network.

The *HANDS* [39] framework sits on GEM5 and allows to simulate multi-core architectures collecting power-performance, thermal and reliability estimates at the same time. However, it does not provide a complete asynchronous NoC model, thus it is not possible to test different DVFS schemes to trade-off power vs. performance.

The work in [9] presents *Sniper*, a framework based on *Graphite* [22] which can simulate multi-cores underpinned by an on-chip network interconnect. Moreover, the simulator supports per core DVFS. However such support is not present for the NoC model and there is no possibility to consider power-performance trade-off for NoC-based multi-core since the Sniper integration with the McPAT power model [19] is not completed yet.

The simulation flow used in this paper introduces some new features in modeling the PLL and the baseline resynchronization scheme, namely the handshake. Zoni et al. [41] extends such models including an accurate voltage regulator model and a FIFO-based resynchronizer thus providing a complete DVFS actuator.

2.2. NoC-based power-performance optimizations

NoCs are becoming a de facto solution for multi-core interconnect. The resource constrained nature of an on-chip network imposes HW architects to optimize their designs considering two directions: increasing performance and reducing power.

Many proposals rely on microarchitectural modifications to improve performance and reduce power. In this perspective, the use of virtual channels and dynamic traffic distribution [15] allows to reduce contention, improve throughput and provide fault-tolerance. The proposed approach has two main differences with respect to these works. First, we do not change the router pipeline, i.e. we do not modify its critical path, instead we change the routers frequencies dynamically to manage contention, thus improving latency. Moreover, our solution is adaptive to the actual NoC load allowing to trade-off power minimization and performance maximization.

Moscibroda and Mutlu [25] proposes a bufferless routing approach that leads to lower power consumption without significantly sacrificing the NoC performance. However, such bufferless schemes deliver reasonable performance only when the injection rate into the network is low. In this scenario, our methodology adapts to the load in the network allowing more flexibility in the power-performance optimization.

Mishra et al. [24] discussed a fine-grained frequency tuning scheme for NoC routers to optimally manage the power-performance trade-off. In particular the methodology exploits signaling between routers to collect critical information to steer frequency. Moreover, the work allows a run-time VC reconfiguration to aggressively save power. However, the proposed solution does not model the relations between routers' frequencies and real performance and power measures. To this extent the proposed solution cannot be easily improved, since it represents a fixed heuristic. Moreover, the proposed methodology can better manage contention providing more efficient solutions.

The work in [3] leverages the traffic unbalancing within a specific NoC topology to exploit the classical technique of DVFS to minimize the power consumption, coupled with ad hoc routing

Table 1
State-of-the-art of multi-core simulators: features, advantages and drawbacks with focus on GALS and DFS support for NoC.

Framework	Cycle-accurate CPU + NoC	NoC support	Power support	GALS support	DVFS/DFS projection	PLL/divider exploration	Objectives
Renau et al. (`SESC`) [31]	✓	✗	✗	✗	✗	✗	Multi-core simulation, parallel applications
Soteriou et al. (`Polaris`) [33]	✗	✓	✓	✗	✗	✗	Network-on-Chip design-space exploration
Hsieh et al. (`SST`) [17]	✗	✓	✓	✗	✗	✗	Microarchitecture, power and thermal
Lis et al. (`HORNET`) [20]	✗	✓	✓	✓	✗	✗	Many-core processors, mainly NoC interconnect
Bartolini et al. [2]	✗	✓	✓	✗	✓	✗	Run-time control policies evaluation for multicores
Zoni et al. (`HANDS` [10,39])	✓	✓	✓	✗	✗	✗	Power, performance and reliability analysis for multicore architectures
Miller et al. (`Graphite` [22])	✗	✓	✗	✗	✗	✗	Distributed parallel simulator for multicore architectures
Carlson et al. (`Sniper` [9])	✓	✓	✓	✗	✓	✗	Distributed parallel simulator for multicore architectures
Subodh Prabhu (`Ocin tsim` [30])	✗	✓	✗	✗	✓	✗	Test different DVFS schemes for NoC
Toolchain in this paper	✓	✓	✓	✓	✓	✓	CPU + NoC simulation NoC PLL/divider DFS support and GALS design

algorithms. A power minimization linear programming model has been proposed to find a routing that minimizes the power consumption while satisfying the traffic demands and meeting the link capacity constraint. The solution relies on a mathematical formulation that must be solved at design time considering a static (even in average) behavior for the system as a whole. In contrast, our paper presents a model of the router frequency and performance, that allows to design run-time optimization policies.

Different proposals focus on the queueing theory to model on-chip networks. Nikitin and Cortadella [26] presented analytical model that focuses on QoS assurance. However, it assumes that the NoC has an underlying synchronous behavior with constant service time routers, thus it is not suitable for optimization using the well known DVFS-based actuators. Moreover, it assumes infinite buffers, and taking into account the finite nature of NoC buffers would greatly complicate the model. An analytical approach exploiting the queueing theory to model NoCs accounting for finite buffers has been presented in [27]. The solution follows the classic queue theory, while the router serving time model has been derived from real data. However, the methodology relies on exponential distribution for flit arrival times which cannot in general be guaranteed in NoCs [26], and does not account for run-time frequency variations.

The work in [11] proposes an heuristic approach focused on DVFS actuators to mitigate power consumption on the real Intel SCC multi-core. Even if this solution has been tested on a real multi-core, it does not provide an accurate model of the relation between frequency and performance thus it does not allow to exploit the solution for further improvements.

3. Proposed methodology

This section presents the NoC frequency control scheme in three steps. First, it is expressed the relation between dynamic frequency and contention for a single router of the NoC; the identified analytical model is detailed in Section 3.1. In the following Section 3.2, the attention is shifted to the controller design and the related closed loop system. Finally, Section 3.3 summarizes the main improvements to the employed simulation flow to support the identification, the controller design as well as the experimental validation. It is worth noticing from now on, that we addressed the power-performance optimization using the DFS actuator instead of DVFS for several reasons, but in any case the proposed methodology can be safely applied even to such systems that support DVFS instead of DFS for each router. In particular, Kim et al. [18] studied on-chip DVFS module integration highlighting two different issues that limit such strategy, namely area overhead and delay. While CPU area overhead due to the integration of multiple DVFS modules, i.e. one per core, is a critical metric to be considered, such overhead is exacerbated when compared to the traditional area footprint of a NoC router. On the other hand, the on-chip PLL area footprint is orders of magnitude smaller than the DVFS at the same technology node [12], thus allowing to implement a per router DFS actuator. Furthermore, the PLL overhead is limited to few microseconds, while the Voltage Regulator (VR) timing overhead is higher and in traditional DVFS schemes it is additive to the PLL one, thus imposing an excessive delay to the reaction of the system to answer the quick and wide load variations in the NoC.

3.1. Model identification

The controller design starts from an analytical model of the process under control, i.e. the impact the frequency of the router has on its load. This section discusses the relation between frequency and contention for an on-chip network router, providing the inputs, the outputs and the internal law which governs the real process. The proposed model takes steps from [40], where the *flow balance equation* described below represents the key difference between the two proposals allowing for better predictions at lower sampling frequencies: the model can be evaluated at lower frequencies without affecting its performance. The next section exploits the model of the process to design a family of controllers.

The real process is the NoC router and we want to describe how a frequency change impacts the contention level. The final equation was devised starting from a set of physical considerations, complemented by the identification of a single parameter. As a result, the model is of the gray-box type [21], with its structure and some of its parameters dictated by the underlying process nature, and the remaining parameters identified based on experimental data.

The model of the process relies on the concept of *control volume*. Given a NoC router, its control volume is the virtual envelope that contains all its direct neighbors, i.e. all the other routers, cores, caches and memory controllers that have a network link connecting to the router, as depicted in Fig. 3.

Given a pair $i, t \in \{Routers, Time\}, f_{i,t}$ represents the frequency of the router i at time t. The frequency is a controllable input, i.e. the knob for the controller. Moreover, the number of flits entering ($InFlits_{i,t}$) and exiting ($OutFlits_{i,t}$) the router's control volume are not controllable inputs, as they depend on the NoC activity, and are therefore considered disturbances. However, the input and

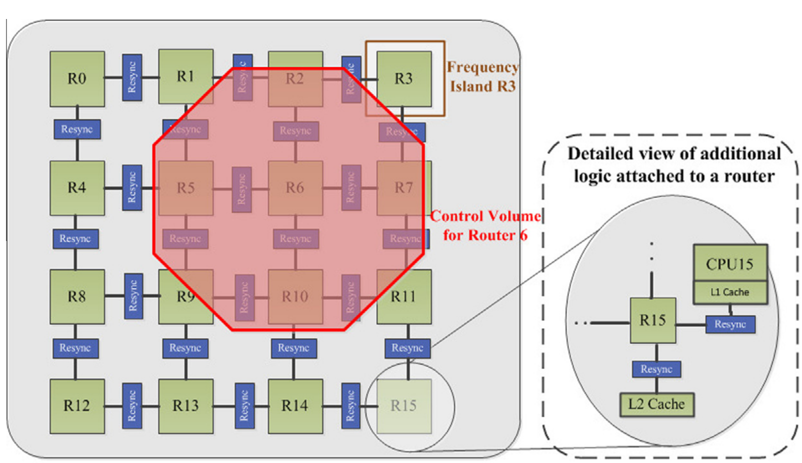


Fig. 3. Control volume for Router 6, in the case when each router in the NoC has its own frequency island. The control volume contains all one hop neighbors of the considered router. It includes L1 and L2 links to count injected flits from cores and cache memory.

output flits crossing the control volume are used for the modeling phase only, since the controller does not account for them directly as - since it operates in closed loop - it can compensate for disturbances without the need to measure them. This aspect greatly simplifies the implementation of the proposed scheme in a real NoC multi-core chip, since sensors for these two inputs are not needed.

The contention $C_{i,t}$ for router i at time t represents the state of a router in our formulation, defined as the sum of the number of flits in the input buffers of the control volume whose next hop is the considered router. The contention is also the output of the model, i.e. the quantity that has to be measured at runtime to close the control loop.

It is worth noticing the need to measure contention at runtime requires communication between routers to exchange the buffer status, resulting in a decentralized control scheme. However, the control loop operates at a significantly lower frequency than the NoC itself, i.e. 10 MHz that is 10 to 100 times slower considering frequencies for the NoC between 100 MHz and 1 GHz. Thus, the requirements for communications are relaxed.

Eq. (1) describes the proposed model for the real process with particular focus on the *flow balance equation* nature of the process:

$$C_{i,t} = C_{i,t-1} + InFlits_{i,t} - OutFlits_{i,t} - \alpha * f_{i,t-1} \tag{1}$$

where the contention of router i at time t is defined as the value at the previous sample point in time, plus the number of flits that entered the control volume, minus the ones that exited it, and minus the router's frequency multiplied by a parameter α. It must be noted that the complete model includes saturations not shown here to ease its understanding. Their purpose is simply to account for the fact that the contention value can never become negative, nor increase above the sum of all buffer sizes in the control volume.

The model contains an integrator, i.e. the contention at time t is equal to the contention at time $(t-1)$, which can easily explained considering that if no flits enter or leave the control volume, and the frequency of the router is zero, the contention remains constant.

The frequency impact on contention however depends on several implementation details, i.e. the internal router structure and the arbiter policies. Moreover, experimental results have shown that the impact the frequency has on the contention is also influenced by the actual traffic patterns. For this reason, the α parameter was introduced to gather such considerations.

This last step needed to complete the model is therefore the identification of the α parameter based on experimental data. For this purpose, it can be noticed that the proposed dynamic model belongs to the family of deterministic autoregressive models with exogenous input (ARX) [5,21]. This allows the use of standard identification techniques to determine α. The resulting ARX model is the one depicted in the right part of Fig. 4.

3.2. Controller design

Starting from the model of the real process developed in Section 3.1, this part discusses the design of a control scheme aiming at router contention reduction, while being conservative in increasing its frequency, since it directly impacts on its dynamic power consumption.

The proposed control scheme is the one shown in Fig. 4. The main idea behind this scheme is to measure contention at run time, and close a control loop around it to control the router frequency. To minimize the control scheme's complexity, which simplifies its hardware implementation, a proportional control scheme was selected which, as the results will show, provides good performance.

Starting from the left side of the figure, the controller's set point is chosen as the constant zero, as ideally the desire is to have no contention. The set point is then subtracted by the (filtered) measured contention, obtaining an error value that is greater than zero if contention is higher than the set point. Note that the signs at the summation node are reversed with respect to a standard proportional control scheme due to the process having a negative gain, i.e. an increase in frequency causes a decrease in contention. The error value is then multiplied by k, the proportional gain. This parameter specifies how aggressively the controller will increase the router frequency in response to an increase in contention. The resulting value is however a frequency value that is zero with no contention. As it is undesirable to completely halt the clock of a router, the computed frequency value is summed to a baseline frequency value, f_{min}. The last computation is a saturation at a value f_{max}, used to set a limit on the maximum frequency that the controller will select. The so computed frequency value is then applied to the PLL to actually change the router's frequency. It should be noted that the PLL response to a change in its frequency set point is not immediate, and its dynamics are significantly slower than the ones of the process to be controlled. Therefore, it is not possible to neglect the PLL dynamics during the design of the controller.

As the PLL is the slowest component in the control loop, it also determines the rate at which the controller is to be clocked. The PLL model used in this paper will be explained in greater detail in the next section, but for now it is important to note that it has a settling time to a step change in frequency of around 2us. To be able to correctly model its dynamics, the control scheme needs to operate 10 to 20 times faster, leading to a 10 MHz controller operation frequency. This value is used to quantize the continuous time PLL model, which is then used to compute the loop transfer function.

The last component to be discussed is the contention filter. Its purpose is to smooth the contention measure, which from the performed experiments was found to contain undesired high frequency components, caused by the fast traffic variations experienced by the NoC routers. The filter cutoff frequency can be tuned using the p parameter. Experimental results have led to the selection of a value of 0.99.

Once a model for all the components composing the control loop are available, it is possible to compute the loop transfer function, and determine the range of k values that guarantee the closed loop stability. A way to obtain this range is through root locus analysis. The root locus technique is a graphical method to evaluate the stability of a system subject to a multiplicative gain uncertainty [37].

Table 2 reports α and the maximum k values for different benchmarks extracted from a 2D-mesh 16-core for router 5 using 2 VCs per virtual network. These values complete the control loop model presented in Fig. 4. The maximum k values are reported on the root locus stability analysis while the minimum allowed k value is zero for each benchmark, which is equivalent to opening the control loop and always selecting the lowest frequency. Starting from the data it is clear how the product of the two numbers, i.e. α and k_{max} provide always the same results, without

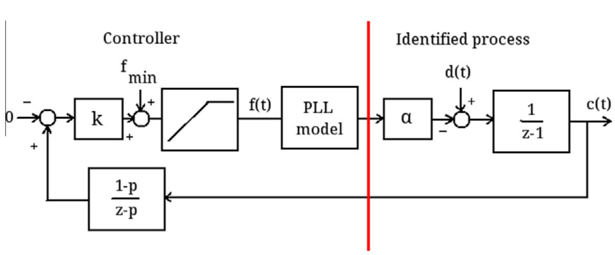


Fig. 4. Block diagram of the proposed NoC frequency control scheme.

Table 2
α and maximum k values for different benchmarks, considering a 2D-mesh 16-core architectures with 2 virtual channels (VCs) per virtual network (VNET). The number refers to router 5.

Name	α	k_{max}	$\alpha * k_{max}$
basicmath	0.847	0.229	0.194
bitcount	1.261	0.152	0.192
crc	1.709	0.113	0.193
dijkstra	0.462	0.415	0.192
fft	2.130	0.091	0.193
susan	2.575	0.075	0.193
qsort	2.497	0.075	0.192
strsearch	1.433	0.135	0.193
ALL	1.880	0.102	0.192

considering rounding. As each benchmark has a different identified model, i.e. different α, to keep it stable in the control loop a different k_{max} is allowed. The selection of the k value when implementing the control loop can however be performed statically, without having to profile individual applications, by being conservative in the k selection.

Although each application results in a slightly different α, we identified a common α value, reported in the *ALL* line of Table 2. This value was obtained by performing the identification procedure using the concatenation of all the benchmarks, thereby taking into account a significant variety of NoC usage patterns. Subsequently, the control law was tuned using that value, i.e. not tailoring k for each specific application, without resulting in unstable behaviors. Moreover, results in Section 4 (see Fig. 10) demonstrates how $k = 0.01$ provides almost the best power-performance product for all the considered benchmarks, while ensuring theoretical stability.

3.3. Simulation flow

To support the model identification process, as well as to test the performance of the proposed control scheme, a NoC simulator is required, capable to change the frequency of individual routers at runtime, and to perform the required measurements.

We start from the GEM5 simulator [4], that is an event-driven simulation framework for multi-core architectures integrating an accurate NoC model [1]. The baseline simulator has been modified introducing four different main logical blocks.

First, support for DFS at the NoC router level was implemented. This required an extension to the event management system of the simulator to support the possibility to move already scheduled events between different simulation times. In fact, changing the frequency of a component entails moving already scheduled events to the time they will need to be served considering the frequency change and storing the new frequency value; in such a way subsequently generated events are scheduled at the appropriate time. Furthermore, we reorganized the router pipeline model in the Garnet network allowing to move events without introducing semantic misbehavior. As described in [8] the stages in a pipeline model for an event driven simulator must be executed in backward order, thus from the last pipeline stage. The DFS implementation allows both to change the frequency of individual NoC routers, as well as to group routers in frequency islands operating at the same frequency.

Second, a model of a new component, the resynchronizer, was added to GEM5 to connect network links crossing frequency domains. The main purpose of this component is to allow reliable data transfers between components operating at different frequencies and/or with a different clock phase, overcoming synchronization and metastability issues. The considered resynchronizer is based on a 2-way handshaking protocol that only adds two single bit lines, (request and acknowledge) to a network link. Its model is the one depicted in Fig. 6, where the *busy* signal is in logical AND with the signal of available credits. Such resulting signal is an input to the switch allocator stage to enable or disable the allocation on the specific output port. In order to simulate an asynchronous multi-core an adequate number of resynchronizer components must be inserted on each frequency domain border, as shown in Fig. 5 and Fig. 3, where the resynchronizers are the blue components.

Third, the actuator modules have been introduced, used by the different policies to change the frequency of NoC routers. Two possible actuators have been implemented: (i) a simple frequency divider, which allows to clock a NoC router at a submultiple of a given base frequency, allowing fast frequency changes (with a delay of just one clock cycle to prevent clock glitches) over a small set of frequencies; (ii) a PLL model which, conversely, allows to accurately select the router frequency, but with a non-negligible settling time.

In particular, both timing and power overheads for the PLL have been examined. Fig. 7 shows the logic blocks of the implemented 45 nm SPICE PLL, starting from a reference charge pump PLL design proposed in [13]. Even if the we lack a compact power model for the PLL, the SPICE-based implementation highlights a power consumption up to 2mW, which is aligned to the power consumption of other proposed PLL models for the same application field, i.e. [14].

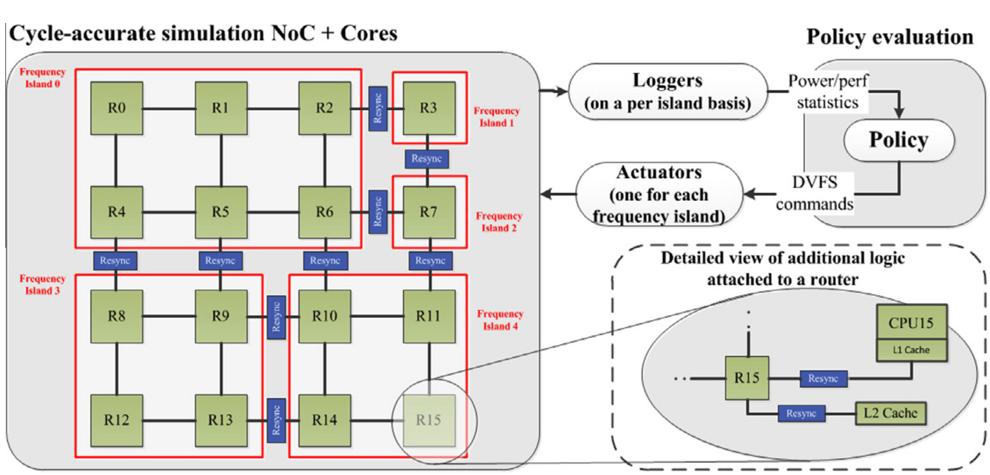


Fig. 5. Logic view of the proposed simulation toolchain.

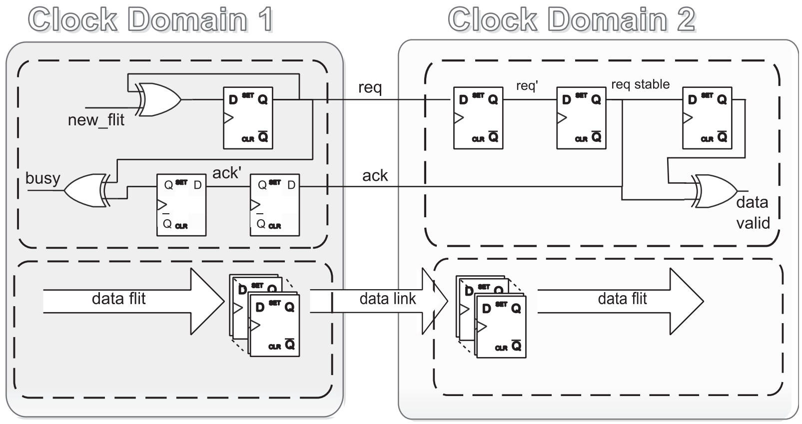


Fig. 6. The implemented edge sensitive resynchronization scheme.

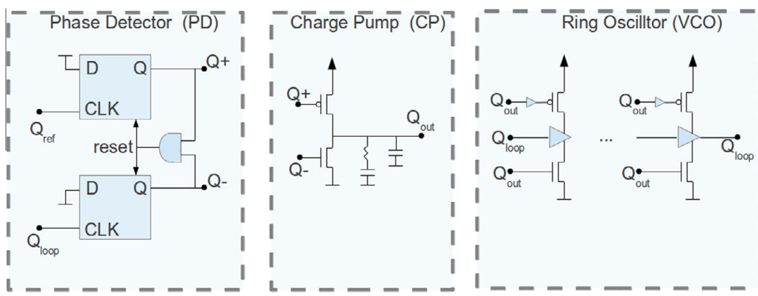


Fig. 7. Logic scheme for the implemented 45 nm SPICE PLL scheme.

From the timing viewpoint, the PLL step response has been modeled using the two pole transfer function of Eq. (2) whose parameters are configurable to approximate the response of a given PLL.

$$G(s) = \frac{1}{1 + 2\frac{\xi}{\omega}s + \frac{s^2}{\omega^2}} \tag{2}$$

A step change in the frequency set point is then simulated by performing multiple individual frequency changes to the frequency island controlled by the PLL to track the two-pole step response. This is performed by turning the transfer function into a state space model, which in this case is a system of two differential equations, that is then integrated using the backward Euler method. The chosen integration step is variable, and is selected as the current PLL period, resulting in continuously changing the clock period on a cycle-by-cycle basis, allowing an accurate simulation of the frequency changes. An example of the PLL model operation under multiple frequency changes is shown in Fig. 8.

The policy module shown as the top right side of Fig. 5, directly interfaces with the simulation module, reading measured data and performing actuation. During the identification phase, a policy is used to perform frequency changes in order to stress the relation between frequency and contention, while the logger submodule is used to save the contention traces to perform identification. During the assessment of the control scheme, a policy is used implementing the controller logic, which allows to simulate how the controller behaves when confronted to realistic traffic patterns produced by a cycle-accurate system-level simulator.

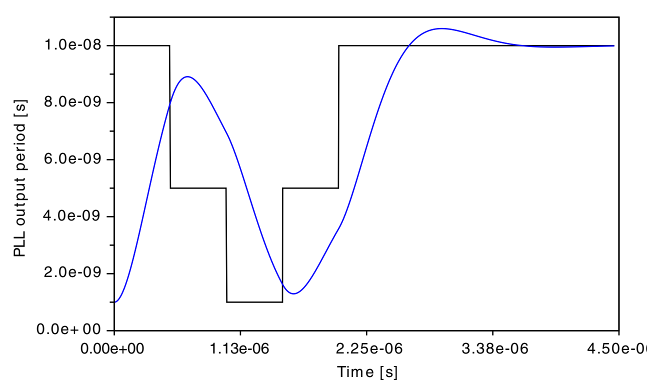


Fig. 8. Simulation of the PLL dynamics with multiple frequency changes.

4. Results

This section details the results for the proposed methodology in four different steps. First, the experimental setup is summarized in Section 4.1, while Section 4.2 compares the baseline control policies against threshold policies. Moreover, we compared our methodology against fixed frequency policies to bound the evaluation. We use contention as performance proxy in Section 4.2 to assess the validity of the proposed model that relies on this metric. Section 4.3 discusses the control policies applied to the whole system, when simulated time and dynamic power are used as performance and power metrics, respectively. Finally, in Section 4.4 we propose a control-based switched system which is able to switch between different control policies at run-time ensuring the stability property.

4.1. Experimental setup

Albeit the run-time power-performance methodology proposed in Section 3 is general, we assess its validity on a representative reference 16-core 2D-mesh tiled architecture, whose parameters are reported in Table 3. Each tile is composed of an Alpha 21264 core running at 1 GHz, with private L1 cache and a shared L2 cache composed of 16 physically distributed banks, one connected to each router. We simulated different scenarios considering several applications taken from the *MiBench* suite using the simulation toolchain detailed in Section 3.3, equipped with the PLL and the DFS actuator models. For each simulation we collected detailed traces for all the routers to computes dynamic power consumption, latency and contention. Average measures have been used to compare different policies while single accurate timing diagram are adopted to show the evolution of the different metrics over the time. We assume a 2-pole PLL model as detailed in Section 3.3 with a $\omega = 4 \cdot 10^6 \xi = 0.6$. Moreover, and without any loss of generality, we bound the upper and lower PLL frequencies between 1 GHz and 100 MHz, providing a realistic and reasonably flexible environment. The Tilera iMesh NoC is similarly clocked up to 1 GHz [36].

Moreover, we assess the proposed control-based methodology against two different classes of policies, namely *static policies*, *threshold policies*. Table 4 reports the details related to the policies we evaluate in the rest of the paper. In particular we rely on two static policies *Upper* and *Lower*, which set the router frequency to 1 GHz and 100 MHz, respectively. These policies represent the baseline for all the other evaluations, since are the best power saving and the most performance oriented policies we can cast. it is worth noticing they implements the resynchronizers as well thus they are actually DFS/DVFS capable even if their frequency is kep fixed. We employ two threshold-based policies, *Th1* and *Th2*, that allow to switch between three different states (three different frequencies) using two different threshold levels. Such policies, used as representative threshold-based policies, have been selected after extensive experiments considering a wide set of parametrized threshold policies. Finally, we evaluate four control-based solutions starting from the design provided in Section 3. In particular each control-based policy has a different k value.

Table 3
Experimental setup: processor and router micro-architectures and technology parameters.

Processor core	1GHz, out-of-order Alpha core
Int-ALU	4 integer ALU functional units
Int-Mult/Div	4 integer multiply/divide functional units
FP-Mult/Div	4 floating-point multiply/divide functional units
L1 cache	64 kB 2-way set assoc. split I/D, 2 cycles latency
L2 cache	2 MB per bank, 8-way associative
Coherence Prot.	MESI
Router	3-stage wormhole virtual channeled switched with 64bit link width 4 fl/VC 2/4 virtual channels (VCs) for each virtual network 3 virtual networks (Garnet network [1])
Topology	2D-mesh, based on Tilera iMesh network [36] for link width and NoC frequency (@1 GHz)
Technology	45 nm at 1.0V
PLL model	$\omega = 4 \cdot 10^6$, $\xi = 0.6$, See Eq. (2) worst case power consumption 2 mW, in accordance to our SPICE developed model and to [12]

Table 4
Evaluated policies organized by classes.

Class	NameID	Details
Static	Upper	Fixed router freq @ 1 GHz
	Lower	Fixed router freq @ 100 MHz
Threshold	Th1	3 switching freq 250–500–800 MHz threshold values buffer fill level 10–20 flits
	Th2	3 switching freq 100–500–1000 MHz threshold values buffer fill level 10–20 flits
Control	$k = 0.01$	The scheme presented in
	$k = 0.04$	Section 3.2, where the NameID
	$k = 0.075$	field in the table provides
	$k = 0.15$	the actual k value

4.2. The baseline control policies

The proposed framework allows accurate run-time estimation of both power and performance metrics considering Dynamic Frequency Scaling modules; in such a way different power-performance trade-off and policies can be explored. The goal of the section is twofold. First, such experiments show how the control policies behave better or at least are more flexible than the threshold based or baseline static policies. Second, we focus on contention as a proxy for performance, since the model is developed starting from this metric that is available at microarchitectural level. Thus, we show how the methodology can be used to explore the design-space using DFS actuators and a proxy for performance. Results pointing out the latency will be provided in the next section, assessing the contention as a good proxy for the performance in this scenario. We report data considering the 16-core architecture presented in Section 4.1 using different benchmarks from *MiBench*. In particular, Fig. 9 reports the results when the previously described policies are implemented in a single router of the NoC, leaving the frequency of the other routers fixed at 1 GHz, to better show the effect of the policies. The selected router is router five, which is in the middle of the mesh, thus has to face higher traffic since we employ the XY routing [23]. The figure reports the three policy classes (see Table 4) for each benchmark at the bottom x axis, while the benchmark names are shown on the top x axis. The left y axis reports the dynamic power for the considered router, while the right y axis shows the contention and power times contention values. It is worth noticing that contention and the power $*$ contention product are normalized to one on a per-benchmark basis to better shape the figure. The figure reports three different measures, namely the *dynamic power* (blue line), the contention (red line) and the *power contention product* (light blue bars).

The control policy with three parameters, i.e. $k = 0.01,\ k = 0.04$ and $k = 0.15$, is evaluated against the *Upper* and the *Lower* static policies as well as the two threshold based policies *Th1* and *Th2*. An important aspect is the selection of an actual value of k. Even if the root locus technique provides a stability region for all the applications when $k < 0.075$, the specific nature of the problem on hand imposes two additional constraints that cannot be directly evaluated with such a technique. Both, the number of flits stored in the buffers and the frequency of the router have the 0 lower bound. Thus, we analyzed a single k above the stability boundary, i.e. $k = 0.15$, proving experimentally that the controller is more responsive while the system is still stable. Besides, we still experimentally observed that higher values of k do not produce any benefit for the system. In summary, the root locus provides a valuable identification of the stability region which, in any case, need to be experimentally refined according to the physical constraints of the problem.

The reported results lead to three main considerations. First, the two static policies actually are the upper and lower bound for the analysis, since the *Lower* has the higher contention level for each benchmark, while the collected dynamic power is always the lowest. The same can be observed for the *Upper* policy that provides the lowest contention with the highest dynamic power consumption. The second observation is on the control-based policy that can explore the power-performance space. In particular, for each benchmark the contention reported for the control-based policy always decreases when increasing k, since we apply a stronger

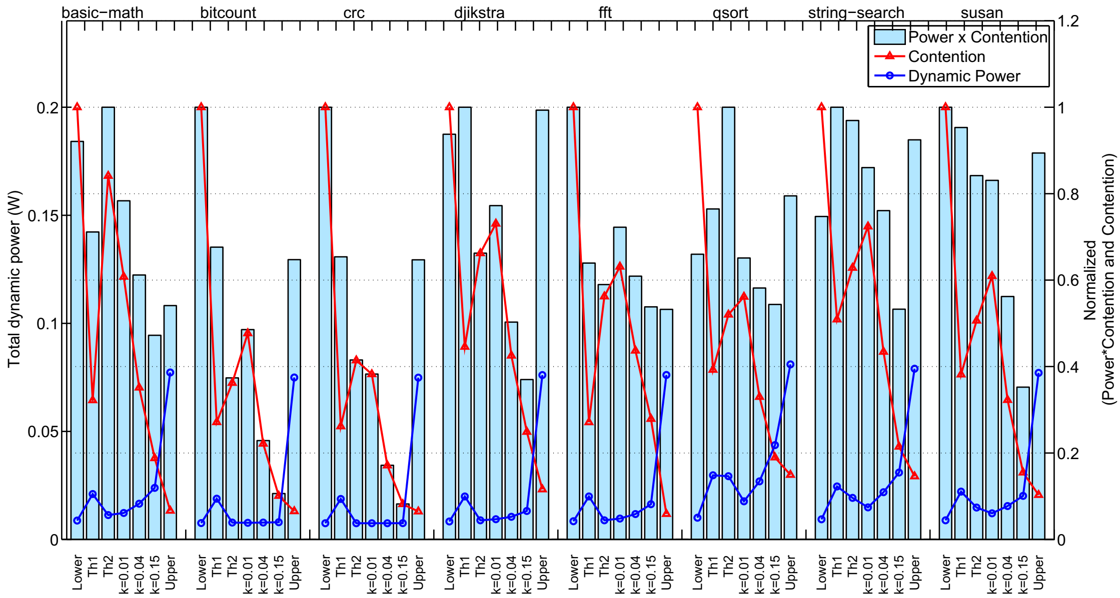


Fig. 9. Router 5 average power-contention trade-off considering different policies. Experiments run on a 16-core NoC multi-core, and the frequency of router 5 only is changed depending on the policy between 100 MHz and 1 GHz. Note that in a 4×4 2D-mesh as the one considered for these results, Router 5 is positioned at the second row and second column of the NoC topology.

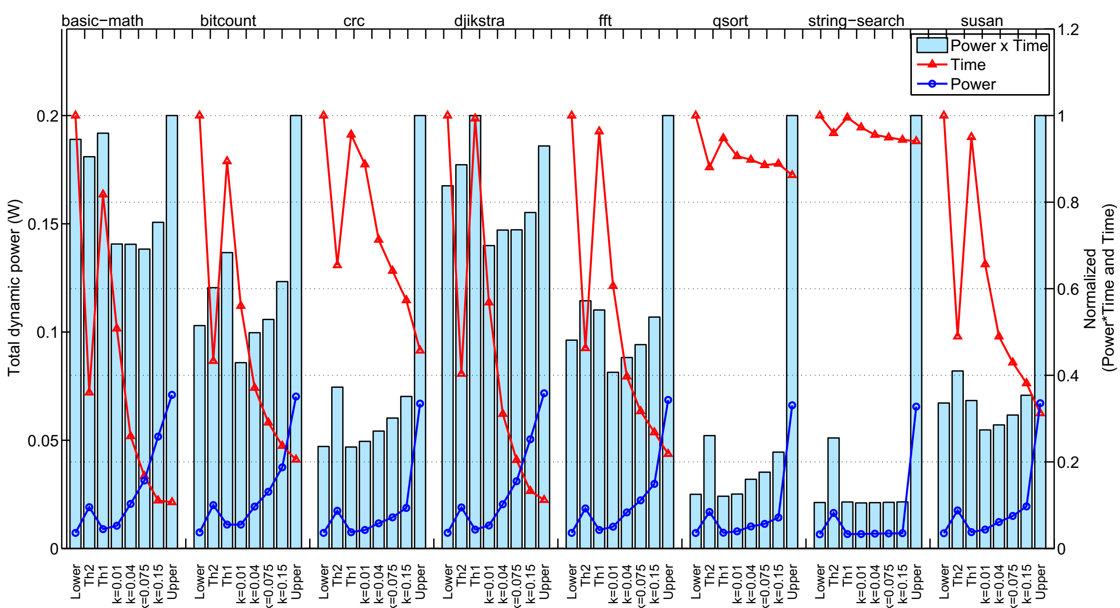


Fig. 10. Evaluation of dynamic power, total execution time and their product when all the routers are running the policy.

actuation, i.e. higher frequencies, in face of a contention increase. On the other side also the blue lines for the control-based policies always increases with the k, even though in some cases the increase is so small that is not noticeable in the graph, since a more strong reaction to the contention increase means higher dynamic power consumption. This aspect is evident on the *string-search* benchmark in Fig. 9, which has 72% of the contention of the static lower frequency policy using $k = 0.01$, than decreases to 43% with $k = 0.04$ and reaches 21% using a $k = 0.15$. The average power consumption is respectively 15, 22 and 31 mW.

The last point aims at the comparison between our methodology and the other policies. In particular, for each benchmark and for each metric we can find at least one control-based policy that behaves better than the threshold based. This means we have more flexibility using our solution to get better results. Moreover, such flexibility depends on a single k parameter easy to understand and manage, while threshold based policies require to set up both threshold levels and operating frequencies.

Note that all the control-based policies considers both timing and worst case power overheads, while the threshold-based ones are supports to change the frequency immediately with no power overhead. Even in this case we can find out, for each benchmark, at least one control-based policy that behaves better than the threshold-based ones.

4.3. Applying the proposed scheme to the whole system

This section discusses the power-performance trade-off on the whole system when all the routers are equipped with a DFS module running one of the presented policies. In particular, Fig. 10 reports data on time to completion as the performance metric instead of contention, while contention represent again the low level metric used by the policy to assess the trade-off. To this extent we can assess the contention metric as a good performance metric, since it impacts directly the simulation time. The graphs for the control policies have the same shape and behavior of the graph in Fig. 9. However, the right side y axis reports the execution time as well as the product between power and execution time, both normalized to one.

The proposed control-based policy is evaluated with different parameters, i.e. $k = 0.01$, $k = 0.04$, $k = 0.075$ and $k = 0.15$, against the *Upper* and the *Lower* static policies as well as the two threshold based policies *Th1* and *Th2*. Fig. 10 shows how the use of the same policy on the entire system does not modify the policy behavior. The control policies can provide better trade-off than the static threshold-based ones, consistently reaching the lowest power-time product for almost all the benchmarks. Last, the proposed control policies can be combined together providing an additional level of flexibility to the final system as shown in the next section.

4.4. The run-time control framework

While Section 4.2 shows how the control-based policies can outperform the threshold-based methodologies, it is often a desirable feature to be able to dynamically select a more performance oriented or power saving policy based on application or user requirements. Many operating systems, for example, incorporate a feature that switches a laptop computer in a higher performance mode when the power cord is attached. To expose such a feature, the operating system needs access to low level actuators that allow to tune the system performance. Moreover, it is quite common to allow a user intervention to set the desired level of power-performance trade-off thus changing the system behavior at run-time. In this perspective threshold-based policies are not flexible enough to provide this feature; although the thresholds can be reset to different values, such modification requires a deep understanding of the low level microarchitecture, thus a wrong tuning of the thresholds can steer to a saturated behavior as discussed in Section 1.1.

This section discusses the possibility to switch between different control policies at run-time, allowing user intervention, and to better fit the required power-performance goal. The possibility to switch between different policies does not invalidate the stability of the system and the interaction require a single action by the higher levels, i.e. OS or user, when the control policy is changed without any further interaction. Moreover, the switch between control policies is different from a threshold policy, since inside a control policy the frequency can be adapted to the actual load while the switch between the control policies allows to change the way the frequency is adapted to cope with the NoC load.

This section demonstrates how the possibility to switch between different controllers can increase the flexibility of the system against different application behaviors. Considering a switched control system, i.e. different controllers can be changed at run-time on the same system, the stability property can be preserved if all the controllers in the switch set are stable and a minimum time between two switches is guaranteed, i.e. a *dwell time*. In this perspective all of the controllers we employ guarantee for an asymptotically stable closed-loop system, thus there exist a minimum finite time interval between two switches that ensures the stability of the switched system [38].

Fig. 11 shows the capability of switching the k parameter. It refers to the *qsort* benchmark with all the router running the control based policy. The figure shows the imposed frequency, the dynamic power and contention level on router five only due to lack of space; the values for the other routers are similar. In particular the blue line reports the data when all routers are running the $k = 0.04$ policy, while the red line shows the same data while routers are running a switched policy that uses $k = 0.04$ from the beginning of the simulation until 3 ms, then switches to $k = 0.075$. The figure points out three aspects. First, a change in the policy at 3 ms does not produce a divergent or saturated behavior in the system, as the frequency is always in between 100 MHz and 1 GHz. Second, the $k = 0.075$ provides a stronger reaction to contention increase, since from 3 ms to the end of simulation the frequency for $k = 0.075$ is higher than the frequency imposed using $k = 0.04$. Moreover, the switched policy allows the benchmark to complete sooner, although with a higher power consumption. To this extent we assess the possibility to change the control policy with a simple command from the higher levels to

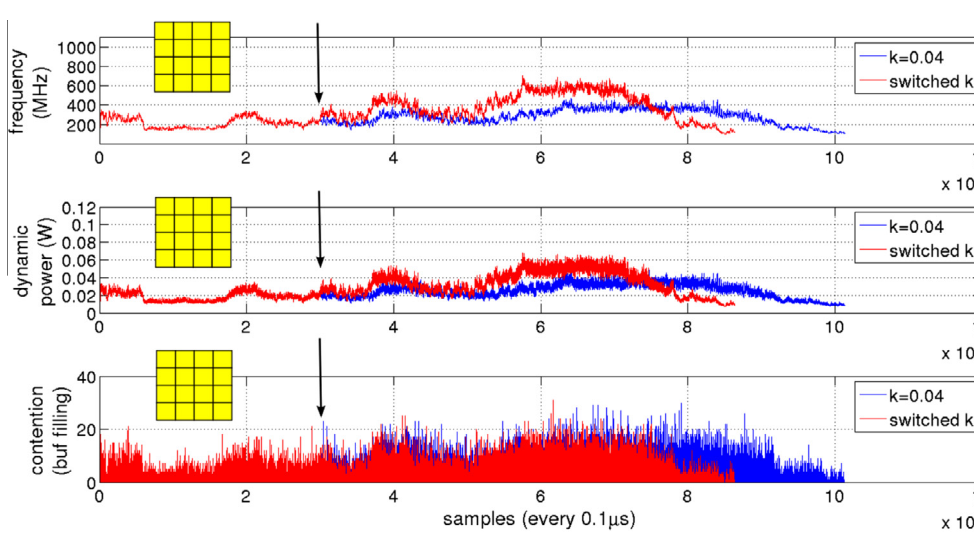


Fig. 11. Evaluation of dynamic power and total execution time on all routers running *qsort*, when the policy is changed from $k = 0.04$ to k=.075 after 3 ms of simulation.

move the system towards a more conservative or more performance oriented behavior. However, the complete development of an optimal switching policy is out the scope of this work.

5. Conclusions

This paper presented a complete methodology to explore and optimize the trade-off between the power and the performance in NoC architectures exploiting the Dynamic Frequency Scaling (DFS). The choice to employ DFS instead of DVFS is supported by many reasons mainly focused on the impossibility to easily integrate too many DVFS actuators, one per each core and one per each NoC router, on the same chip. Moreover, the voltage regulator could impose an excessive delay to the reactivity of the system in the case of NoC load variations, that are usually fast and of relevant magnitude. However, our methodology can be easily adapted to encompass also DVFS actuators.

The investigation we carried out is organized in three steps. First, a model of the frequency-contention for a single router is developed as a dynamic system. Then, a set of controllers have been implemented to cope with stability issues and dynamic frequency adaption according to the NoC load. Last, a new simulation framework accounting for GALS NoC and PLL modules is developed to assess the methodology. Experimental validation compared a family on control policies against static and threshold based policies [24] providing better results both in term of power saving and simulated time. In particular, the best control policy provides a speedup on the simulated time up to 3.05 against the best threshold policy (see *crc* in Fig. 10). Moreover, the control policies can save up to 33.17% on the power-time product against the best threshold policy (see Fig. 10, *basicmath* benchmark comparing *Th1* and $k = 0.01$ policies). All in all, the $k = 0.01$ control-based scheme shows an averaged power-performance product improvement of 16.50% against the best considered threshold-based policy. Finally we explored the possibility to change the behavior of the control policy at run-time, by tuning the k value, to further increase the flexibility of the proposed approach. Furthermore the exploration highlights the possibility to interact with higher layers, i.e. the operating systems power management module.

References

[1] N. Agarwal, T. Krishna, L.-S. Peh, N. Jha, Garnet: a detailed on-chip network model inside a full-system simulator, in: ISPASS, 2009.

[2] A. Bartolini, M. Cacciari, A. Tilli, L. Benini, M. Gries, A virtual platform environment for exploring power, thermal and reliability management control strategies in high-performance multicores, in: GLSVLSI'10, ACM, New York, NY, USA, 2010, pp. 311–316.

[3] A. Bianco, P. Giaccone, N. Li, Exploiting dynamic voltage and frequency scaling in networks on chip, in: High Performance Switching and Routing (HPSR), 2012, IEEE, 2012, pp. 229–234.

[4] N. Binkert, B. Beckmann, G. Black, S.K. Reinhardt, A. Saidi, A. Basu, J. Hestness, D.R. Hower, T. Krishna, S. Sardashti, R. Sen, K. Sewell, M. Shoaib, N. Vaish, M.D. Hill, D.A. Wood, The gem5 simulator, SIGARCH Comput. Archit. News 39 (2) (2011) 1–7.

[5] S. Bittanti, Model identification and adaptive systems, in: Pitagora Editrice, Bologna, 2004, 304p. (in italian).

[6] S. Borkar, Networks for multi-core chips: a contrarian view, in: Special Session at ISLPED, 2007.

[7] W.L. Brogan, Modern Control Theory, third ed., Prentice-Hall Inc, Upper Saddle River, NJ, USA, 1991, ISBN 0-13-589763-7.

[8] D. Brooks, V. Tiwari, M. Martonosi, Wattch: a framework for architectural-level power analysis and optimizations, in: Proceedings of the 27th Annual International Symposium on Computer Architecture, ISCA '00, ACM, New York, NY, USA, 2000, pp. 83–94.

[9] T. Carlson, W. Heirman, L. Eeckhout, Sniper: exploring the level of abstraction for scalable and accurate parallel multi-core simulation. in: 2011 International Conference for High Performance Computing, Networking, Storage and Analysis (SC), 2011, pp. 1–12.

[10] S. Corbetta, D. Zoni, W. Fornaciari, A temperature and reliability oriented simulation framework for multi-core architectures, in: IEEE Computer Society Annual Symposium on VLSI (ISVLSI2012), University of Massachusetts, Amherst, USA, 2012.

[11] R. David, P. Bogdan, R. Marculescu, U. Ogras, Dynamic power management of voltage-frequency island partitioned networks-on-chip using intel's single-chip cloud computer, in: 2011 Fifth IEEE/ACM International Symposium on Networks on Chip (NoCS), 2011, pp. 257–258.

[12] W. Deng, A. Musa, T. Siriburanon, M. Miyahara, K. Okada, A. Matsuzawa, A dual-loop injection-locked pll with all-digital background calibration system for on-chip clock generation, in: Design Automation Conference (ASP-DAC), 2014 19th Asia and South Pacific, 2014, pp. 21–22.

[13] D.E. Duarte. Clock Network and Phase-Locked Loop Power Estimation and Experimentation (PhD thesis), Pennsylvania State University, 2002.

[14] A. Elshazly, R. Inti, M. Talegaonkar, P. Hanumolu, A 1.5ghz 1.35mw 112dbc/hz in-band noise digital phase-locked loop with 50fs/mv supply-noise sensitivity, in: VLSI Circuits (VLSIC), 2012, pp. 188–189.

[15] P. Gratz, B. Grot, S. Keckler, Regional congestion awareness for load balance in networks-on-chip, in: IEEE 14th International Symposium on High Performance Computer Architecture, 2008. HPCA 2008, 2008, pp. 203–214.

[16] M.R. Guthaus, J.S. Ringenberg, D. Ernst, T.M. Austin, T. Mudge, R.B. Brown, Mibench: a free, commercially representative embedded benchmark suite, in: Workshop of the Workload Characterization. WWC-4, IEEE, Washington, DC, USA, 2001, pp. 3–14.

[17] M.-Y. Hsieh, A. Rodrigues, R. Riesen, K. Thompson, W. Song, A framework for architecture-level power, area, and thermal simulation and its application to network-on-chip design exploration, SIGMETRICS Perform. Eval. Rev. 38 (2011) 63–68.

[18] W. Kim, M. Gupta, G.-Y. Wei, D. Brooks, System level analysis of fast, per-core dvfs using on-chip switching regulators, in: IEEE 14th International Symposium on High Performance Computer Architecture, 2008. HPCA 2008, 2008, pp. 123–134.

[19] S. Li, J.H. Ahn, R. Strong, J. Brockman, D. Tullsen, N. Jouppi, Mcpat: an integrated power, area, and timing modeling framework for multicore and manycore architectures, in: 42nd Annual IEEE/ACM International Symposium on Microarchitecture, 2009.

[20] M. Lis, P. Ren, M.H. Cho, K.S. Shim, C. Fletcher, O. Khan, S. Devadas, Scalable, accurate multicore simulation in the 1000-core era, in: IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS), 2011, pp. 175–185.

[21] L. Ljung, System Identification: Theory for the User, second ed., Prentice Hall, 1999.

[22] J. Miller, H. Kasture, G. Kurian, C. Gruenwald, N. Beckmann, C. Celio, J. Eastep, A. Agarwal, Graphite: a distributed parallel simulator for multicores, in: High Performance Computer Architecture (HPCA) 2010, IEEE, 2010, pp. 1–12.

[23] A.K. Mishra, N. Vijaykrishnan, C.R. Das, A case for heterogeneous on-chip interconnects for cmps, in: Proceedings of the 38th Annual International Symposium on Computer Architecture, ISCA '11, ACM, New York, NY, USA, 2011, pp. 389–400.

[24] A.K. Mishra, A. Yanamandra, R. Das, S. Eachempati, R. Iyer, N. Vijaykrishnan, C.R. Das, Raft: a router architecture with frequency tuning for on-chip networks, J. Parallel Distrib. Comput. 71 (5) (2011) 625–640.

[25] T. Moscibroda, O. Mutlu, A case for bufferless routing in on-chip networks, in: Proceedings of the 36th Annual International Symposium on Computer Architecture, ISCA '09, ACM, New York, NY, USA, 2009, pp. 196–207.

[26] N. Nikitin, J. Cortadella, A performance analytical model for network-on-chip with constant service time routers, in: IEEE/ACM International Conference on Computer-Aided Design – Digest of Technical Papers, 2009. ICCAD 2009, 2009, pp. 571–578.

[27] U. Ogras, P. Bogdan, R. Marculescu, An analytical approach for network-on-chip performance analysis, IEEE Trans. CAD-ICS 29 (12) (2010) 2001–2013.

[28] J. Owens, W. Dally, R. Ho, D.N. Jayasimha, S. Keckler, L.-S. Peh, Research challenges for on-chip interconnection networks, Micro, IEEE 27 (5) (2007) 96–108.

[29] R. Parikh, R. Das, V. Bertacco, Power-aware nocs through routing and topology reconfiguration, in: Design Automation Conference (DAC), 2014 51st ACM/EDAC/IEEE, 2014, pp. 1–6.

[30] S. Prabhu, B. Grot, P.V. Gratz, J. Hu, 1 ocin tsim- dvfs aware simulator for nocs.

[31] J. Renau, B. Fraguela, J. Tuck, W. Liu, M. Prvulovic, L. Ceze, S. Sarangi, P. Sack, K. Strauss, P. Montesinos, SESC simulator, 2005. <http://sesc.sourceforge.net>.

[32] P.K. Sahu, S. Chattopadhyay, A survey on application mapping strategies for network-on-chip design, J. Syst. Arch., 1383-7621 59 (1) (2013) 60–76. <http://www.sciencedirect.com/science/article/pii/S1383762112000902>.

[33] V. Soteriou, N. Eisley, H. Wang, B. Li, L.-S. Peh, Polaris: a system-level roadmap for on-chip interconnection networks, in: International Conference on Computer Design, 2006. ICCD 2006, 2006, pp. 134–141.

[34] F. Terraneo, D. Zoni, W. Fornaciari, A cycle accurate simulation framework for asynchronous noc design, in: International Symposium on System on Chip (SoC), 2013, 2013, pp. 1–8.

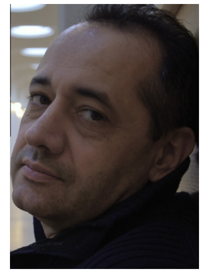
[35] E. Totoni, B. Behzad, S. Ghike, J. Torrellas, Comparing the power and performance of intel's scc to state-of-the-art cpus and gpus, in: Proceedings of the 2012 IEEE International Symposium on Performance Analysis of Systems & Software, ISPASS '12, IEEE Computer Society, Washington, DC, USA, 2012, ISBN 978-1-4673-1143-4, pp. 78–87, http://dx.doi.org/10.1109/ISPASS.2012.6189208.

[36] D. Wentzlaff, P. Griffin, H. Hoffmann, L. Bao, B. Edwards, C. Ramey, M. Mattina, C.-C. Miao, J.F. Brown III, A. Agarwal, On-chip interconnection architecture of the tile processor, IEEE Micro 27 (5) (Sept. 2007) 15–31.

[37] X.-L. Wu, L. Xia, Z.-Z. Wang, Note on the root locus method, J. Shanghai Univ. (English Edition), 1007-6417 13 (2009) 105–107, http://dx.doi.org/10.1007/s11741-009-0203-1.
[38] G. Zhai, B. Hu, K. Yasuda, A. Michel, Qualitative analysis of discrete-time switched systems, in: American Control Conference, 2002, vol. 3, 2002, pp. 1880–1885.
[39] D. Zoni, S. Corbetta, W. Fornaciari, Hands: heterogeneous architectures and networks-on-chip design and simulation, in: Proceedings of the 2012 ACM/IEEE International Symposium on Low Power Electronics and Design, ISLPED '12, ACM, New York, NY, USA, 2012, pp. 261–266.
[40] D. Zoni, F. Terraneo, W. Fornaciari, An analytical, dynamic, power-performance router model for run-time noc optimizations, in: SoCC, 2013, pp. 290–295.
[41] D. Zoni, F. Terraneo, W. Fornaciari, A dvfs cycle accurate simulation framework with asynchronous noc design for power-performance optimizations, J. Signal Process. Syst., 1939-8018 (2015) 1–15, http://dx.doi.org/10.1007/s11265-015-0989-1.

Davide Zoni received the Master Degree in Computer Engineering from Politecnico di Milano in 2010 and the Ph.D. degree in Information Technology from Politecnico di Milano in 2014. He currently holds a Post-Doc position at the Dipartimento di Elettronica Informazione e Bioingegneria (DEIB), Politecnico di Milano, Italy. His main research interests include networks-on-chip, computer architecture as well as the application of control theory methodologies for power, performance and reliability optimizations in multi-cores. He received a best paper award in 2012 and two HiPEAC Collaboration Grants in 2013 and 2014. In FP7 he participates to the EU-funded HARPA project.

Federico Terraneo received his Ph.D. degree in Information Technology from Politecnico di Milano in 2015. Currently he holds a Post-Doc position at Politecnico di Milano – Dipartimento di Elettronica, Informazione e Bioingegneria (DEIB). His research interests include embedded systems and the application of principles of control theory to the design of software systems.
Since 2008 he is the main developer and maintainer of the Miosix embedded operating system.



William Fornaciari MS, PhD, is Associate Professor at POLIMI. He published six books and around 200 papers, collecting 5 best paper awards, one certification of appreciation from IEEE and holds 3 international patents on low power design. Since 1993 he is member of program committees and chair of international conferences in the field of computer architectures, EDA and system-level design. Since 1997 he has been involved in 14 EU-funded international projects. In FP7 he has been WP leader for the IP COMPLEX projects and Project Technical Manager of 2PARMA (ranked as success story by the EU) and he also participates to the Artemis SMECY project on smart multicore embedded systems.
Currently he is WP leader of the CONTREX project on mixed criticalities and Project Coordinator the HARPA project on embedded and HPC technologies to ensure dependable performance.
He cooperated for around 20 years with the Technology Transfer Center of POLIMI, gaining significant experience in cooperating with international companies for the development of leading edge products: industrial exploitation of research ideas is one of his main attitudes and teaching task, and in 2013 he created a startup company focusing on embedded systems.
His main research interests cover multi-many core architectures, NoC, low power design, software power estimation, run time resource management, wireless sensor networks, thermal management, and EDA-based design methodologies. He is co-author of the first italian book on embedded systems and he acted as project reviewer for EC funded projects and invited speaker during EU consultation/information workshops.
William Fornaciari is IEEE Senior Member.