Tiny NoC: A 3D Mesh Topology with Router Channel Optimization for Area and Latency Minimization

César Marcon, Ramon Fernandes, Rodrigo Cataldo, Fernando Grando, Thais Webber, Ana Benso, Letícia B. Poehls
PUCRS - Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil – 90619-900
{cesar.marcon, ana.benso}@pucrs.br, {ramon.fernandes, rodrigo.cataldo, fernando.grando, thais.webber}@acad.pucrs.br

Abstract—This paper presents Tiny NoC, which is a scalable and efficient 3D mesh architecture developed to minimize latency and NoC area. First, we show a theoretical analysis of latency and area occupancy to demonstrate Tiny NoC efficiency when compared to a basic mesh NoC. Then, we select a set of synthetic and mapping independent traffic with several injection rates to analyze the advantages and weaknesses of Tiny NoC. The experimental results highlight that Tiny NoC always reduces area occupancy and for several cases it provides latency minimization.

Keywords - 3D mesh NoC, optimization, latency, area.

I. INTRODUCTION

Rapid advances in manufacturing technology of integrated circuits (ICs) and the growth in the number of transistors per area allowed implementing complete system functionality into a single IC, which is called System-on-Chip (SoC). Furthermore, the SoC is called Multiprocessor SoC (MPSoC) when components are Processing Elements (PEs) containing a processor, a possible memory hierarchy and a local communication infrastructure that enables their joint operation. Nevertheless, there is a need to implement efficient communication infrastructures to meet the performance requirement of several applications such as Network-on-Chip (NoC), which performs PEs communications [1].

NoC provides scalability and communication parallelism, enabling the PEs communication by packets exchange [2]. NoC may be based on two-dimensional (2D) architecture, where communication infrastructure and PEs are arranged in the same plane [3]. However, 2D architecture has limitations pertaining latency and power consumption that are even more striking as the number of PEs and the distance between routers increase [4]. Thus, the natural evolution in NoC architectures is to add another dimension in the manufacturing process of ICs, resulting in three-dimensional architecture - 3D NoC [5].

3D NoC allows the reduction of the communication distance and the number of hops needed for packet traffic, reducing latency and increasing throughput, and hence reducing the execution time of the application [6].

The contribution of this paper is to present a new exploration of 3D mesh NoCs, increasing the number of PEs on a basic 3D structure based on Through Silicon Via (TSV) in order to propose a significant reduction on NoC area and latency. We introduce a new addressing model based on a straightforward deterministic routing XYZ-based to accomplish these goals.

This paper is organized as follows. Section II discusses some related works. Section III presents the Tiny NoC architecture emphasizing the proposed optimizations. Section IV presents a theoretical comparison between Tiny NoC and a basic 3D mesh NoC. Section V shows experimental results of NoC latency and area, and Section VI presents the conclusion and future work.

II. RELATED WORK

This section presents some relevant characteristics of similar works that explore architectural optimization aspects on 2D/3D NoCs, comparing them with Tiny NoC approach.

Jeang et al. [9] present a mesh pyramidal 3D NoC, where each layer is a mesh and trees connect adjacent layers. They optimize interlayer links enabling efficient broadcasting and high speed communications. Experiments comparing the total transfer time in the mesh-tree to a basic mesh topology show that the extra area used is rewarded by the rise of performance.

Wang and Thota [10] developed Multiprocessor-on-a-Programmable-Chip (MPoPC) - a communication architecture for multiprocessors that is resource-efficient -. The PEs of MPoPCs follow a mesh topology, whereas routers are organized in torus topology. By sharing a router among PEs and a PE among routers, they increased the routing flexibility and reduce communication hops. They implemented the proposed communication design in various sizes on a variety of FPGAs. Results show good performance with significant reduced resource requirements for all configurations.

Chen et al. [11] proposed a 3D NoC based on De Bruijn graph, whose architecture is called DB_DB. It is a topology with small diameter, simple routing and high reliability. It uses TSV and enables reduction on the energy consumption and average latency. They use a message simulator to compare performance results of DB_DB with two other mesh architectures. Simulator applies Poisson process to perform packet injection and uniform and hotspot traffics. Results point out that DB_DB can achieve smaller network latency than mesh architectures but increases the energy consumption.

Camacho and Flich [12] developed the Homogeneous-Parallel-Concentrated-Mesh (HPC-Mesh), which consists of 4 disjoint homogeneous concentrated 2D mesh NoCs. HPC-Mesh provides connectivity to all these 4 NoCs by using an injection algorithm. They optimize the communication by dynamically adjusting topology according to traffic demands. HPC-Mesh has fault-tolerance and energy saving as goals. Results show significant reduction on power consumption without impairing throughput for some application traffics. In addition experiments on 3D topology displayed low and practical resource overhead.

Camacho et al. [13] developed 2D Nearest neighboR-Mesh (NR-Mesh), which allows several alternative paths for most

source-destination pairs. Nodes are directly connected to four neighbor routers enabling to inject messages through different routers, thereby reducing latency. Results show that NR-Mesh reduces execution time and power consumption when compared to traditional 2D mesh. The power management is able to achieve significant power savings when allowing links and routers to switch off and when combining adaptive routing.

Daneshtalab et al. [14] propose two novel clustering stacked topologies for 3D NoC (i.e. CMIT - Clustered Mesh Interlayer Topology; CIT- Concentrated Interlayer Topology) to reduce the area overhead of TSVs and power dissipation on each layer with minimal loss on performance. A 4×4×4 3D stacked architecture is compared to conventional structures under uniform and non-uniform traffic. In the latter case, CIT reduces the average NoC latency. On the former, CIT and CMIT reduce the power consumption whereas reducing the number of TSVs.

Seifi and Eshghi [15] proposed the Clustered NoC (C-NoC) as improvement of 2D basic mesh NoC (H-NoC) by grouping 4 PEs into each router aiming to reduce NoC size and the average messages latency. C-NoC presented less average packet delay than H-NoC in corner-to-corner communications. Since C-NoC has fewer channels than H-NoC when all PEs send packets randomly in both NoCs, the average delay in C-NoC augments with contention. In random traffic, when NoC size rises, the performance of C-NoC is comparable to H-NoC. In addition, optimized buffers assure that when both NoCs have the same number of PEs, C-NoC size is 75% smaller than H-NoC.

All works presented in this section are summarized on Table I. They explore architectural optimization of 3D or 2D NoCs, focusing mainly on the minimization of latency and energy consumption. Similarly to the works [14][15], this paper explore clusters on the border routers to simplify routing algorithm implementation and to minimize area consumption.

TABLE I - RELATED WORK SUMMARY.

Work	NoC name	Topology	Optimization goal and type/feature
[9]	Mesh-tree	3D mesh pyramidal	Optimizes interlayer links enabling broadcasting and high speed communications
[10]	MPoPCs	2D mesh, torus	Increases the links quantity to improve routing flexibility and to reduce communication hops
[11]	DB_DB	De Bruijn 3D NoC	Small diameter, simple routing to minimizes average latency and energy consumption
[12]	HPC-Mesh	2D/3D mesh	Dynamic topology with 4 disjoint NoCs performing fault-tolerance and saving energy
[13]	NR-Mesh	2D mesh	PEs connected to neighbor routers to reduce latency and routers switch off to save energy
[14]	CMIT, CIT	clustered 3D mesh	Clusters of routers or PEs that reduces the TSV area overhead and power dissipation
[15]	C-NoC	2D mesh	Cluster of PEs to minimizes NoC area and size, and application delay time
This	Tiny NoC	3D mesh	Optimization of routers links usage implying to mix of a single PEs and a cluster of PEs that minimizes NoC area and latency

III. TINY 3D MESH NOC ARCHITECTURE

Tiny 3D mesh NoC is based on a straightforward 3D direct mesh NoC architecture called Lasio [7]. Tiny NoC was designed to minimize area occupation and latency. The main underlying idea is to keep the NoC topology and layout as simple and regular as possible, enabling to achieve high scalable network that fulfill more PEs communication with less area cost than a basic direct mesh topology [16]. The Tiny NoC characteristics are described in the subsequent sections.

A. Router Architecture

Figure 1 shows the components of Tiny router architecture that includes 7 input/output ports, a 7×7 crossbar switch responsible for ports connections, a control logic that performs routing and arbitration as well as data and control links.

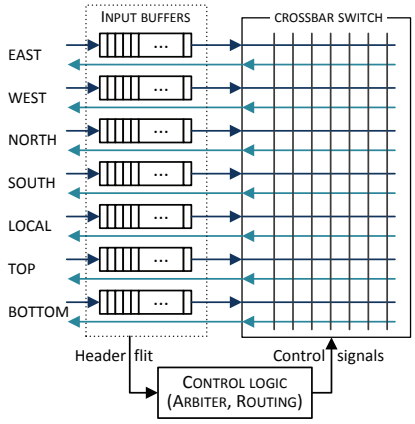

Figure 1 - Main components of Tiny NoC router architecture.

All ports support bidirectional communication and allow temporary data storage through configurable size FIFO buffers. North, South, East and West ports are employed for interconnecting neighboring routers or PEs within each layer; and Local port is used for PEs interconnection only. Communication with other routers or PEs across chip layers is carried out through Top and Bottom ports by TSV technology.

When a router receives the first flit (header - containing the target router address), the arbiter uses Round Robin algorithm for granting equal access to the crossbar switch and output ports. If the routing algorithm is unable to establish a connection to an output port (i.e. due to NoC congestion), the input port requires to the arbiter a new routing request. Even though this approach could cause packet contention at certain times (and therefore increasing overall network latency), it ensures that all incoming packets are processed preventing starvation phenomenon.

B. Router Interface

Figure 2 shows a bidirectional link enabling full-duplex communication, using a credit based protocol. Input ports are composed of 4 signals: (i) *clockRx* for data synchronization; (ii) *rx* for data availability signaling; (iii) *dataIn*, a flit-bit size bus for receiving data; (iv) *creditOut* for indicating buffer availability. Output ports employ the counterpart control/data signals: (i) *clockTx*; (ii) *tx*; (iii) *dataIn*; and (iv) *creditIn*.

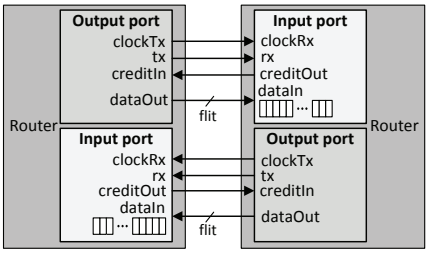

Figure 2 - Control and data signals of a bidirectional port of Tiny NoC routers.

C. Tiny Topology

Tiny NoC enables to connect multiple PEs per router using some ports of routers placed on the NoC surface. It implies different PEs placement according to each router physical location. Figure 3 shows a 3×3×3 Tiny NoC example, where it can be seen that using Tiny NoC approach instead of a basic mesh NoC, the number of PEs may jump from 27 to 81.

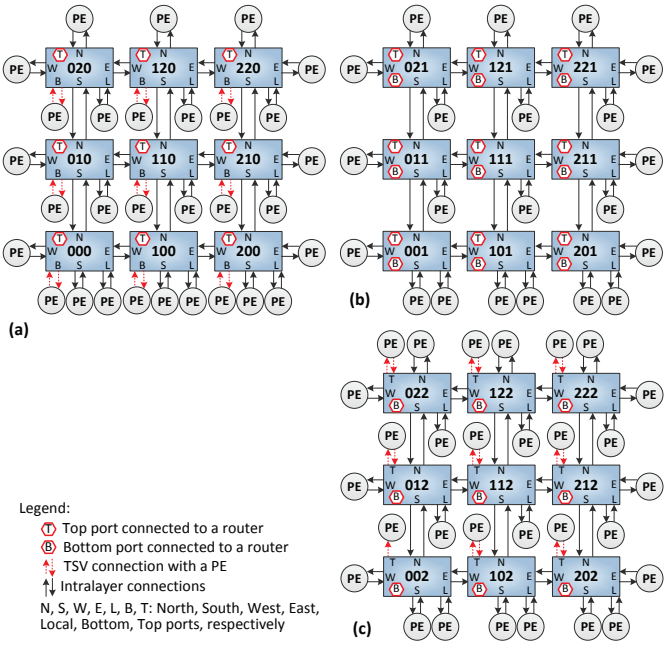

Figure 3 - 3×3×3 Tiny NoC with layers: (a) bottom; (b) middle and (c) top.

D. Address Composition

Tiny NoC uses an addressing scheme similar to the basic 3D mesh NoC [7], named here as BMesh NoC, whose addressing is based on router's XYZ coordinates. Since BMesh NoC has a single PE connected at each router through its Local port, identifying destination addresses depends solely on those coordinates information. Once a packet arrives at a router, if the packet destination address matches with the router address, the router delivers the packet to the PE through its Local port. But, this procedure is insufficient for identifying to which port a given router should deliver packets in Tiny NoC architecture.

Addressing at Tiny NoC requires to identify to which port a router should deliver incoming packets, as well as the router XYZ coordinates. Since each router is composed of seven ports, Tiny's address aggregates 3-bit code representing routers ports, as follows: (000, East); (001, West); (010, North); (011, South); (100, Local); (101, Bottom); and (110, Top).

Figure 4 (a) illustrates that Tiny NoC packet is composed of *Header* and *Data* fields, where *Header* is split into: (i) *destination address* field; and (ii) *size* that contains the quantity of flits comprising the *Data* field.

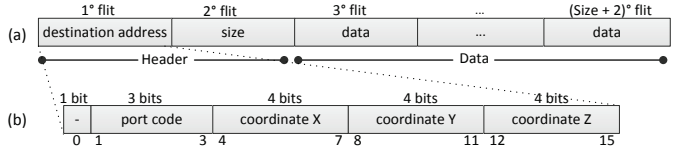

Figure 4 - Fields composition of a Tiny NoC packet.

Figure 4 (b) shows an example of 16-bit flit, where the *destination address* is composed of 3-bit port code field with X, Y and Z destination coordinate fields.

Except by the *port code* field, the *destination address* of Tiny NoC is identical to BMesh NoC, implying that Tiny NoC is compatible with the BMesh NoC routing logic. Consequently, internal routers of Tiny NoC may have exactly the same architecture of internal routers of BMesh NoC. This could represent interesting architectural explorations, such as heterogeneous Tiny NoC (i.e. composed of BMesh and Tiny NoC routers) that may reduce yet more area.

IV. Theoretical Comparison of Tiny NoC and a Basic Mesh NoC

This section shows theoretical comparisons between Tiny NoC and BMesh NoC in terms of PE's support implying area consumption as in terms of average number of hops of all communications, implying latency.

A. Area Consumption Evalution

BMesh NoC topology has a single PE connected to each router. Thus the quantity of PEs is directly proportional to X, Y and Z NoC dimensions, as stated on Eq. 1. On the other hand, Tiny NoC supports to connect PEs on the unused links of the border routers, which increases the quantity of PEs proportionally to the NoC surface, as defined by Eq. 2.

$$nPE_{BMesh} = X \times Y \times Z \quad (1)$$

$$nPE_{Tiny} = 2 \times ((X \times Y) + (X \times Z) + (Y \times Z)) + X \times Y \times Z \quad (2)$$

When considering a cubic NoC, with all dimensions equal to **N**, Eq. 1 and Eq. 2 are rewrite in Eq. 3 and Eq. 4, respectively.

$$nPE_{BMesh} = N^3 \quad (3)$$

$$nPE_{Tiny} = 6 \times N^2 + N^3 \quad (4)$$

For both topologies, Figure 5 shows the curves relating the quantity of PEs according to **N** variation.

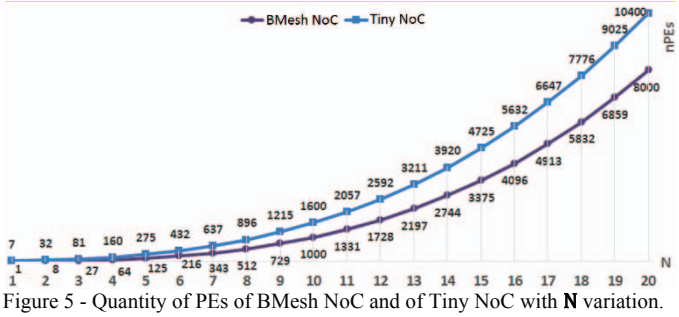

Figure 5 - Quantity of PEs of BMesh NoC and of Tiny NoC with **N** variation.

Figure 6 illustrates the percentage variation of the quantity of PEs gain obtained with Tiny NoC over BMesh NoC according to **N** variation. The curves show that is highly significant the Tiny NoC approach for small NoCs (e.g. **N** = 6 implies 100% of gains, since Tiny NoC enables to connect 432 PEs, whereas BMesh NoC enables only to connect 216 PEs). On the other hand, this advantage is linearly reduced with NoC size increasing (e.g. for **N** = 20 the advantage of Tiny NoC approach versus BMesh NoC is 2400 PEs, but it means only 30% of gain).

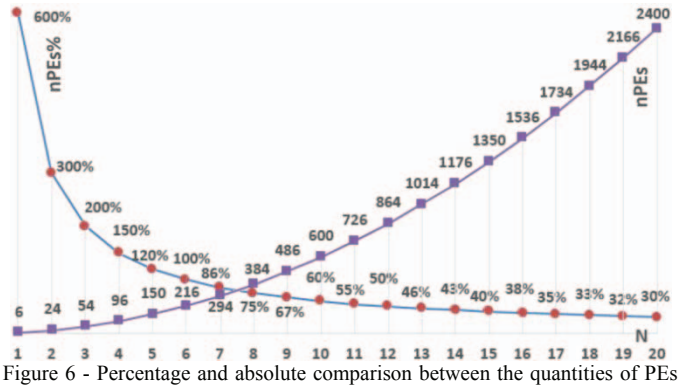

Figure 6 - Percentage and absolute comparison between the quantities of PEs supported by Tiny NoC versus BMesh NoC.

B. Physical Aspects Comparison

Tiny NoC was designed to be scalable as BMesh NoC is, in this sense it is regular in terms of logical topology and tiles geometry, and consequently links size. On the other hand, the optimizations inserted on Tiny NoC imply some differences between border and internal tiles, which are discussed here.

Figure 7 exemplifies possible tiles placement, and links size of a 2×2×2 Tiny NoC. The figure shows four planes of tiles: (i) levels Z=0 and Z=3 - that are the bottom and top planes, respectively, both containing only four PEs; and (ii) levels Z=1 and Z=2 - that are 2 intermediate planes, each one containing 12 PEs. Thus, this example has 8 routers connecting 32 PEs.

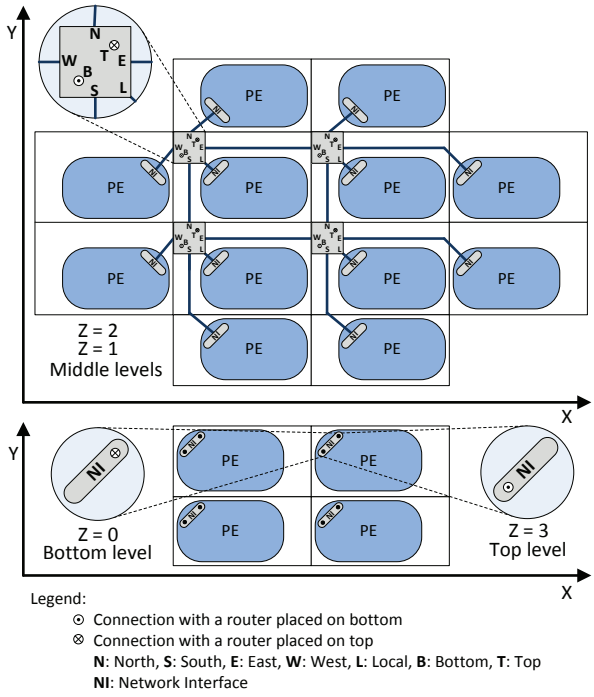

Figure 7 - Example of 2×2×2 Tiny NoC with regular tiles, supporting 32 PEs.

As can be observed, Tiny NoC brings some kind of irregularity with respect to the links size and area available to PEs placement. As Section V presents next, the routers of BMesh NoC and Tiny NoC spent almost the same area; and since borders tiles don't have routers, they encompass more area to place larger PEs. Additionally, corner routers may be also larger, if we envision a rectangular SoC geometry.

Tiny router implementation is almost similar to the BMesh router producing the same critical path delay. Since links size of both NoCs are nearly the same, and the load impedance is equivalent, Tiny NoC and BMesh NoC may operate at the same clock frequency, allowing fair comparison on latency.

C. Latency Evalution

The communication latency depends on the: (i) topology, (ii) NoC size, (iii) application traffic and (iv) PE/task mapping. These two last latency dependences are influenced by some static and dynamic application features (e.g. traffic known only at execution time) making difficult to achieve theoretical latency comparison between the NoC topologies. Thus, in order to avoid application dependences, we chose as a metric for latency comparison the hops average (**hopsAvg**) of all possible communications that each topology enables to perform.

The total number of hops of all communication paths of BMesh NoC (**nH_{BM}**) is computed by Eq. 5, which performs six nested sums. The first three sums represent the source PE address (i.e. referenced by x,y,z) whereas the last three sums represent the target address (i.e. referenced by i,j,k), and the Eq. 5 is not defined when (x,y,z) = (i,j,k). For instance, there are two hops from the source PE placed on (0,0,0) to the target PE placed on (0,0,1), i.e. the same X,Y-coordinates, but in a neighbor Y plane, since a packet passes through two routers.

$$nH_{BM} = \sum_{x=0}^{X-1}\sum_{y=0}^{Y-1}\sum_{z=0}^{Z-1}\sum_{i=0}^{X-1}\sum_{j=0}^{Y-1}\sum_{k=0}^{Z-1}(|x-i|+|y-j|+|z-k|+1) \quad | (x,y,z) \neq (i,j,k) \quad (5)$$

The total number of hops of all communication paths of Tiny (**nH_{Tiny}**) is computed by Eq. 6, which is similar to Eq. 5, but includes the **oE(x,y,z,i,j,k)** optimization (i.e. Eq. 7) that captures the hops optimization caused by PEs placement inside the same routers. Eq. 7 is composed of **qProc(a,b,c)** and **equal(x,y,z,i,j,k)** equations (i.e. Eqs. 8 and 10) that compute the quantity of PEs linked to each border router and the effect of a communication among PEs connected in ports of the same router.

$$nH_{Tiny} = \sum_{x=0}^{X-1}\sum_{y=0}^{Y-1}\sum_{z=0}^{Z-1}\sum_{i=0}^{X-1}\sum_{j=0}^{Y-1}\sum_{k=0}^{Z-1}(|x-i|+|y-j|+|z-k|+1)\times oE(x,y,z,i,j,k) \quad (6)$$

$$oE(x,y,z,i,j,k) = qProc(x,y,z)\times(qProc(i,j,k) - equal(x,y,z,i,j,k)) \quad (7)$$

$$qProc(a,b,c) = border(a,X) + border(b,Y) + border(c,Z) + 1 \quad (8)$$

$$border(n,l) = \begin{cases} 1, & n = 0 \vee n = l-1 \\ 0, & else \end{cases} \quad (9)$$

Eq. 8 is composed of **border(n,l)** equation (Eq. 9) that returns one (1) when a PE address corresponds to a router placed in a border of the NoC in a given dimension. For instance, the address (0,0,0), which is a corner router enables to connect four PEs, since it is in the inferior border of X, Y, Z dimensions.

$$equal(x,y,z,i,j,k) = \begin{cases} 1, & (x,y,z) = (i,j,k) \\ 0, & else \end{cases} \quad (10)$$

Eq. 10 is used on Eq. 7 to inform communications inside the same router, which implies only one hop of distance because this parcel reduces the total quantity of hops. Eq. 11 computes the quantity of communication channels for both NoCs, which is all communications among all PEs.

$$nComm = nPE \times (nPE - 1) \quad (11)$$

Eq. 12 computes **hopsAvg** for both NoC topologies. Where **nHops** is $\mathbf{nH_{BM}}$ or $\mathbf{nH_{Tiny}}$ depending on the NoC topology, i.e. BMesh NoC or Tiny NoC, respectively.

$$hopsAvg = \frac{nHops}{nComm} \quad (12)$$

Eq. 13 is the number of clock cycles spent in a transmission of a flit that passes through η routers (t_r), η-1 inter-router links (t_{lr}) and 2 local links (t_{lp}) - i.e. between a PE and a router. Remark that Eq. 13 may be applied for both NoCs, even if a communication is performed between two PEs placed in the same router.

$$nCk = \eta \times t_r + (\eta - 1) \times t_{lr} + 2 \times t_{lp} \quad (13)$$

Tiny NoC and BMesh NoC use credit based control flow that usually transmit a flit in 1-clock cycle (i.e. $t_{lr} = t_{lp} = 1$-clock cycle). Moreover, the routing algorithm of both NoCs takes in average four clock cycles (i.e. $t_r = 4$ clock cycles). Thus, considering η = **hopsAvg**, Eq. 14 enables to estimate the number of clock cycles used in the average of all communications.

$$nCk = 5 \times hopsAvg + 1 \quad (14)$$

Using Eq. 14, the Figure 8 illustrates Tiny NoC and BMesh NoC comparison with respect to the average of clock cycles according to some NoC sizes, which are represented by the quantity of PEs. Remark that some latencies comparison are not achieved according to the same quantity of PEs, due to the fact that this experiment takes into account totally populated NoCs, and the same quantity of PEs does not match for some NoC sizes. In such cases, we search for a nearest quantity of PEs (e.g. 270 PEs for BMesh NoC and 275 for Tiny NoC).

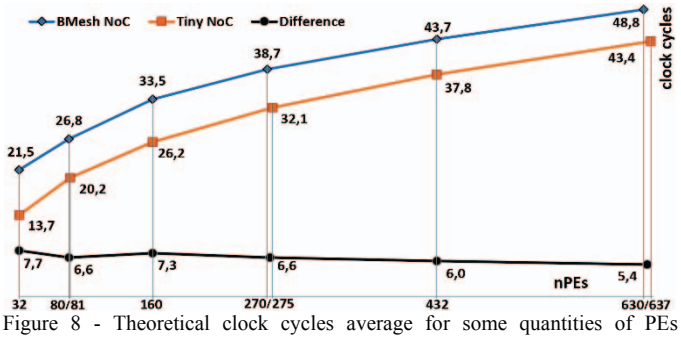

Figure 8 - Theoretical clock cycles average for some quantities of PEs connected to BMesh NoC or Tiny NoC.

Figure 8 depicts that, without contention, not considering task/PE mapping and ignoring traffic injection, according to the theoretical model described, Tiny always enables to reduce latency when compared to BMesh. The latency reduction tends to be less significant with the NoC size increase, which is reasonable since the proportional quantity of border routers is reduced with NoC increase if compared to the internal routers quantity. For the same NoC type, with the same tasks/PEs placement, the packets contention is directly dependent on the number of PEs sending packets at the same time as well as on the transmission rate of these simultaneous packets.

BMesh NoC has more communication paths and more total storage area when compared with Tiny NoC. However, Tiny NoC has shorter communication paths in average. These features make BMesh more appropriate to minimize packets contention; since it has much more buffers spread on the NoC to minimize the latencies of packets. But, Tiny NoC provides more efficient communication for traffics with low contention.

In addition, the switching of ports is implemented with a crossbar circuit enabling full parallel communications. Then, PEs of Tiny NoC connected directly to the same router may communicate without delaying other communications. This feature highlights the importance of mapping approaches to be further analyzed when applying Tiny NoC architecture.

V. Experimental Results

This section compares Tiny NoC area and latency with BMesh NoC implementations.

A. Area Results

In order to evaluate area consumption, the routers of BMesh NoC and Tiny NoC were synthesized targeting the 65nm STMicroelectronics CMOS technology, assuming 16-bit size flit and 8-bit depth buffer. The syntheses were performed with Cadence RTL Compiler and employed a general purpose standard cell library provided by the foundry. As a result, the routers of BMesh NoC and Tiny NoC occupied 0.0784 mm^2 and 0.0807 mm^2, respectively, showing less than 3% of area overhead on the implementation of Tiny NoC router. Also, since the quantity of routers is reduced on Tiny NoC, Figure 9 shows the area gain in percentage for some set of PEs.

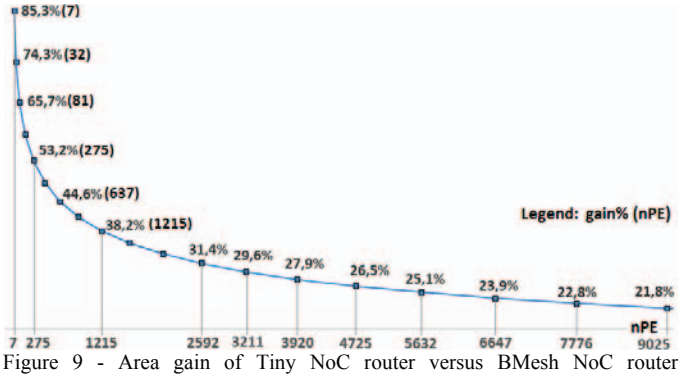

Figure 9 - Area gain of Tiny NoC router versus BMesh NoC router implementations w.r.t the increase of nPEs.

B. Latency Results

Packet latency is highly dependent of PE/task mapping. Hence, we choose to calculate latency with *all-to-all* traffic, whose symmetry enables to evaluate independent task mappings. In this traffic pattern, all PEs deterministically send the same quantity of data to all other PEs, except to itself. Firstly, all PEs simultaneously send a packet to the first PE address, and then all PEs send packets to the second PE address, and so on. This traffic may cover many traffic patterns and blocking situations since large quantity of packets travels at the same time on the NoC. Our experiments explore two variations of this traffic that impacts directly on the packets contention: (i) the quantity of PEs that are simultaneously injecting packets; and (ii) the injection rate of these packets.

Figure 10 shows a graphic covering the gains on average latency when using Tiny NoC as communication architecture of

some sizes of MPSoCs instead of using BMesh NoC. We choose 16-bit size flit and 8-bit depth buffer for both NoC simulations, and we compare three basic NoC sizes: (i) nPE = 32, which refers to 32 PEs achieved with 2×2×2 Tiny NoC and 4×4×2 BMesh NoC; (ii) nPE = 80/81, which refers to 80 PEs achieved with 5×4×4 BMesh NoC and 81 PEs reached with 3×3×3 Tiny NoC; and (iii) nPE = 160, which refers to 160 PEs achieved with 4×4×4 Tiny NoC and 8×5×4 BMesh. Note that we do not use the same quantity of PEs in nPE = 80/81, because for all NoCs we apply the total capacity of PEs, and the topological differences between Tiny NoC and BMesh NoC prohibit to achieve exactly the same quantity of PEs.

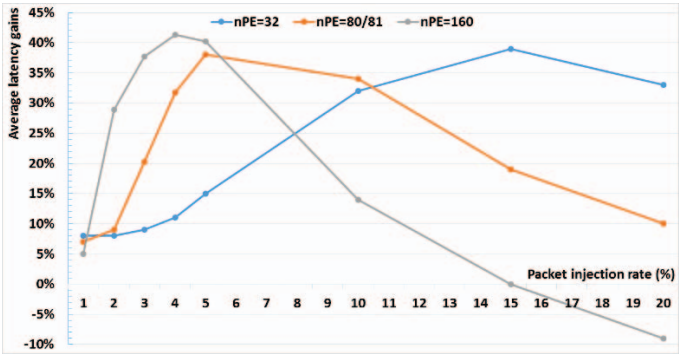

Figure 10 - Gains percentage of average packet latency when comparing Tiny NoC with BMesh NoC. The graphic takes into account some sizes of MPSoCs (i.e. quantities of PEs) and several injection rates.

Figure 10 shows that for very low injection rates, which imply few packet concurrences, the results achieved are near the ones computed in the theoretical section (refer to Section IV.C). When packet injection rate increases the minimization of packet latency becomes evident for Tiny NoC. For instance, with 4% of injection rate Tiny NoC reduces more than 40% on the average packet latency. The latency gains versus packet injection rate vary according to the NoC size. While the experiment nPE = 32 shows maximum gain on 15% of injection rate, the experiment nPE = 160 reaches the maximum at 4%. For all experiments, when the packet injection rate remains increasing, the gains are minimized and tend to be negative (i.e. Tiny NoC becomes inefficient). This problem is due to the fact that increasing the quantity of packets also increases the concurrence for NoC resources (i.e. links and buffers), which are reduced in Tiny NoC implementation if compared with BMesh NoC. However, this problem may be minimized with real applications, if the designer performs a task/PE mapping step before the system operation.

VI. Conclusion and Future Work

Tiny NoC provides a simplified XYZ based routing algorithm, which is deadlock free and highly scalable.

Comparing Tiny NoC with a basic mesh (BMesh) NoC, we conclude that: (i) due to the routers quantity reduction, Tiny NoC always enables to minimize the total NoC area, even with small increase of internal router area needed to implement the routing algorithm modifications; (ii) Tiny NoC reduces the average of packets latency, when has few communications and/or low packet injection rate. On the other hand, when increasing the packet injection rate and there are huge concurrent communications, BMesh NoC provides less packet latency on average due to larger quantity of buffered communication paths. Additionally, we emphasize that, even not exploring energy evaluation in this work, Tiny NoC is a promising structure to ensure energy consumption minimization, since communication occurs on less links and routers with lower latency on average.

As a future work we intend to evaluate mapping techniques that better explore the localities advantages provided by Tiny NoC (i.e. place in the same router high communicating PEs).

Acknowledgment

This work is partially funded by FAPERGS under grants PqG 12/1777-4 and Docfix SPI n.2843-25.51/12-3.

References

[1] A. Ahmed et al. **Architecture and Design of Efficient Network-on-Chip (3D NoC) for Custom Multicore SoC.** *Conf. on Broadband, Wireless Computing, Communication and Applications (BWCCA)*, pp.67-73, 2010.

[2] T. Bjerregaard; S. Mahadevan. **A Survey of Research and Practices of Network-on-Chip.** *ACM Computing Surveys*, v.38, pp.1-51, Mar. 2006.

[3] L. Feng et al. **Design and Performance Evaluation of a 2D-Mesh Network on Chip Prototype Using FPGA**. *IEEE Asia Pacific Conference on Circuits and Systems (APCCAS)*, pp.1264-1267, 2008.

[4] P. Pande et al. **Performance evaluation and design trade-offs for network-on-chip interconnect architectures.** *IEEE Transactions on Computers*, v.54, n.8, pp.1025-1040, Aug. 2005.

[5] A. Sheibanyrad; F. Pétrot; A. Jantsch. **3D Integration for NoC-based SoC Architectures.** Series Integrated Circuits and Systems. *Springer*, 2011, 278p.

[6] B. Feero; P. Pande. **Networks-On-Chip in a Three-Dimensional Environment: A Performance Evaluation.** *IEEE Transactions on Computers*, v.58, n.1, pp.32-45, Jan. 2009.

[7] Y. Ghidini et al. **Buffer depth and traffic influence on 3D NoCs performance**. *IEEE International Symposium on Rapid System Prototyping (RSP)*, pp.9-15, 2012.

[8] A. Ahmed; A. Abdallah. **Low-overhead Routing Algorithm for 3D Network-on-Chip**. *Third International Conference on Networking and Computing (ICNC)*, pp.23-32, 2012.

[9] Y.-L. Jeang et al. **Mesh-Tree Architecture for Network-on-Chip Design.** International Conference on *Innovative Computing, Information and Control (ICICIC)*, pp.1-4.2007.

[10] X. Wang; S. Thota. **Design and Implementation of a Resource-Efficient Communication Architecture for Multiprocessors on FPGAs**. *International Conference on Reconfigurable Computing and FPGAs (ReConFig)*, pp.25-30, 2008.

[11] Y. Chen et al. **De Bruijn Graph based 3D Network on Chip Architecture Design.** *International Conference on Communications, Circuits and Systems (ICCCAS)*, pp.986-990, 2009.

[12] J. Camacho; J. Flich. **HPC-Mesh: An Homogeneous Parallel Concentrated Mesh for Fault-Tolerance and Energy Savings.** *ACM/IEEE Symposium on* Architectures for Networking and Communications Systems *(ANCS)*, pp.69-80, 2011.

[13] J. Camacho et al. **A power-efficient network on-chip topology.** *International Workshop on Interconnection Network Architecture: On-Chip, Multi-Chip (INA-OCMC)*, pp.23-26, 2011.

[14] M. Daneshtalab et al. **Cluster-based topologies for 3D stacked architectures.** *ACM International Conference on Computing Frontiers (CF)*, Art.14, 3p, 2011.

[15] M. R. Seifi, M. Eshghi. **Clustered NOC, a suitable design for group communications in Network on Chip**. *Computers & Electrical Engineering*, v.38, n.1, pp.82-95, Jan. 2012.

[16] S. Murali; G. De Micheli. **SUNMAP: a tool for automatic topology selection and generation for NoCs**. *Design Automation Conference (DAC)*, pp.914-919, 2004.