Capturing Topology-Level Implications of Link Synthesis Techniques for Nanoscale Networks-on-Chip

Daniele Ludovici, Georgi N. Gaydadjiev
Computer Engineering Lab., TUDelft
2628 CD Delft, The Netherlands
{d.ludovici,g.n.gaydadjiev}@tudelft.nl

Davide Bertozzi
ENDIF, University of Ferrara
44100 Ferrara, Italy
dbertozzi@ing.unife.it

Luca Benini
DEIS, University of Bologna
40136 Bologna, Italy
luca.benini@unibo.it

ABSTRACT

In the context of nanoscale networks-on-chip (NoCs), each link implementation solution is not just a specific synthesis optimization technique with local performance and power implications, but gives rise to a well-differentiated point in the architecture design space. This in an effect of the tight interaction existing between architecture and physical design layers in nanoscale technologies. This work assesses several NoC link inference techniques (buffering options, link pipelining) by means of commercial backend synthesis tools, taking the system-level perspective. In fact, performance speed-ups and power overhead are not evaluated for the links in isolation but for the network topology as a whole, thus showing their sensitivity to the link inference strategy. k-ary n-mesh topologies are considered for the sake of analysis, in that they provide a range of topologies with increasing total wirelength.

Categories and Subject Descriptors

B.m [**Hardware**]: MISCELLANEOUS

General Terms

Design, Performance

Keywords

Link design techniques, Network-on-Chip

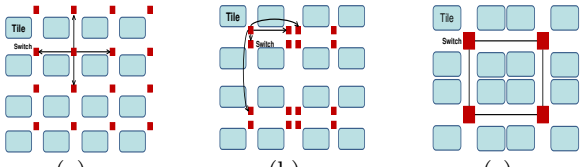
1. INTRODUCTION

Previous studies of wire scaling effects based on ITRS roadmaps return a grim view. [6] extends performance projections of wires out to the 13nm technology node and sees both local and global wires degrading relative to gates over nine generations, by one and three orders of magnitude respectively. This trend is not just impacting circuit design, but is having heavy architecture-level implications as well. In this direction, network-on-chip (NoC) architectures rely on aggressive path segmentation and regular connectivity patterns, at least for general-purpose tile-based NoCs. The ultimate objective is to come up with a modular architecture removing global wire delays from the critical path. This design technique has been successful since the early days of NoC design since the critical path is often confined into the arbitration logic of NoC switches [7]. However, the unrelenting pace of technology scaling to the nanoscale regime is bringing interconnect-related issues to the forefront even for NoC design. For a 65nm technology, [2] points out the significant gap between post-synthesis and post-place&route performance reports affecting NoC modules when logic synthesis and placement are carried out as two clearly separated stages. Inaccurate wire load models are at the root of this gap, therefore placement-aware logic synthesis tools are advocated. Moreover, although global wires are intrinsically segmented, the maximum interswitch link length still plays a key role in topology design. [4] and [5] proved that a bufferless implementation of interswitch links already suffices to move critical path back to them. This trend has to be taken into close account especially when evaluating topologies with complex connectivity patterns. Fortunately, designers can today rely on a number of well-known techniques to engineer delay-optimized wires, thus making system performance less sensitive to the raw numbers of wire performance. As an example, a traditional design technique for long links consists of inserting equally spaced CMOS repeaters to deal with resistive loss along the wire. This makes delay of repeated wires almost linear with length rather than quadratic. However, with the increase in number and density of the wires with each new technology, interconnect area and power might be severely impacted. Power overheads in the order of tens of Watts might be expected [8]. Another way of tackling timing violations on long links is by pipelining it. By providing one or more extra clock periods to traverse long distances, pipeline stages along links solve the link infeasibility issue at a much lower cost than deploying whole NoC switches in place of them. However, the major drawback is that flow control must be extended to account for the fact that feedback signals now return after multiple clock cycles instead of in the same clock period. This can be handled by deeper buffers at link endpoints or by pipelining the link with flow control-aware elements. In all cases, a significant power overhead might be incurred.The effectiveness of a link performance boosting technique cannot be assessed on the link in isolation, but has to be captured at system (topology) level. In fact, link performance and power have a number of high-level implications, especially overall network speed. Unfortunately, most previous work on physical link design investigates cost-benefit trade-offs only for the link in isolation, while ignoring the sensitivity of topology metrics to a specific link design technique. This paper aims to go a step further and at demonstrating topology level implications of high-impact interconnect optimization techniques such as link buffering and pipelining. Baseline topologies with bufferless links will be used for the sake of comparison. The paper pursues a twofold objective. On one hand, it assesses whether for a given topology an investment on a faster link pays off in terms of total energy. On the other hand, it investigates whether different link synthesis techniques can put well-known performance-power trade-offs between various kinds of topologies in discussion.

GLSVLSI'09, May 10–12, 2009, Boston, Massachusetts, USA.

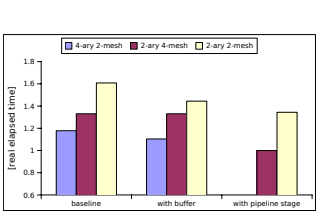
2. TOPOLOGIES UNDER TEST

In contrast to previous work, we capture the sensitivity of quality metrics of NoC topologies with a regular connectivity pattern to the specific synthesis technique for their links. This way, guidelines are provided to NoC designers concerning the opportunity to invest more area and power to speed up the links of their topology. Some non-trivial implications will be demonstrated. The link variants analyzed in this work and the topologies making use of them have been laid out by means of a backend synthesis flow leveraging industrial tools. Please refer to [12] for details. Link design techniques impact topology quality metrics in two ways. On one hand, it is the longest link in the topology that determines its maximum achievable speed. Therefore, cutting down on the delay of that link is beneficial for the whole topology. On the other hand, the cost for performance boosting of the critical link becomes relevant for the topology only when there are more such critical links and they account for a significant fraction of the total topology wirelength. We found representative design points for our link sensitivity analysis to be available in the k-ary n-mesh family of regular NoC topologies. In this paper we focus on 16 tile systems, hence consider a 4-ary 2-mesh (referred to as 2D-mesh from now on) with one tile per switch and a 2-ary 2-mesh with 4 tiles per switch. The latter solution is denoted as the *concentrated topology*. Both the 2D mesh and its concentrated counterpart feature 2 dimensions and homogeneous interswitch wire lengths. However, such wire length will not be the same in the two topologies due to floorplan constraints reflecting a well-known trade-off for these topologies: spread-around topologies on one hand (large number of low-radix switches) as opposed to more concentrated ones with a small number of high-radix switches *placed further apart* to optimize connectivity with a large number of tiles. In fact, these latter have to be placed around the switches they have to be connected to, thus separating the switches in space. In these two topologies, all the links have to be performance-optimized in order to speed-up the entire topology, hence the associated cost will be more relevant in relative terms with respect to the baseline unoptimized topologies. A different design point is represented by hypercubes. The n-hypercube topology is a particular case of a mesh where k is always 2. For 16 tile systems, a 2-ary 4-mesh (4-hypercube) can be obtained from the 2D mesh by increasing the number of dimensions and by reducing the number of switches in each dimension. Interestingly, the maximum switch radix stays the same, only the 4-hypercube has all the switches with the same radix while the 2D mesh has the central switches with a higher radix than the peripheral ones. Of course, the 4-hypercube has larger bisection bandwidth and lower network latency. Unfortunately, this comes at the cost of links with uneven length. In fact, in a mesh with more than two dimensions the links used to connect the dimensions greater than two (often denoted as *express links*) are longer, and this holds for 50% of the 4-hypercube interswitch links. In theory, the length of a link of the dimension t is generally assumed to be $k^{(d-2)/2}$, where d is equal to t if t is an even number and d is equal to $t+1$ if it is an odd number. Unfortunately, placement and layout constraints put this picture in discussion, thus making it very difficult to predict the impact that the cost of a specific link performance boosting technique might have on the cost metrics of the entire topology. As an example,

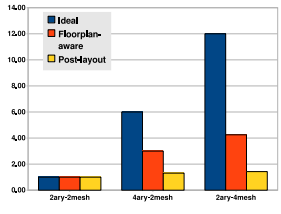


(a) (b) (c)

Figure 1: Floorplan directives for (a) the 2D mesh, (b) the 4-hypercube and (c) the 2-ary 2-mesh.



(a) Real Elapsed Time



(b) Total wire length

Figure 2: Performance and wiring results

Topology	4-ary 2-mesh	2-ary 4-mesh	2-ary 2-mesh
Post-Synthesis (WC Switch))	1 ns	1 ns	1.15 ns
Post-P&R (WC Switch)	1.12 ns	1.12 ns	1.4 ns
Post-P&R repeater-less (Topology)	1.27 ns	1.56 ns	1.67 ns
Post-P&R with buffers (Topology)	1.19 ns	1.56 ns	1.5 ns
Post-P&R with pipeline stages (Topology)	–	1.19	1.42

Table 1: Timing results.

consider the floorplanning directives given for topology synthesis in this paper. They are reported in Fig.1(a), Fig.1(b) and Fig.1(c). The asymmetric tile size plays in favor of the 4-hypercube wiring, since the length of the horizontal and of the vertical express links turns out to be comparable to that of horizontal wires in the 2D mesh. This latter also features horizontal and vertical links of unequal length, indicating that the layout regularity often assumed in high-level considerations does not materialize in practice. Our guiding principle for floorplan definition consists of *shortening the longest links in each topology* and of coming up with a scalable floorplanning style. For the 2-ary 2-mesh, we placed the computation tiles around the switch they are attached to, which is an ideal scenario for pipelining interswitch links. In all cases, network interfaces were placed close to their tile but also to the connected switch, so to move the critical path away from these critical links. As a result of topology synthesis and place-and-route, Fig.2(b) shows the total wiring length for the three topologies (*Post-Layout* curve), normalized to the least wire-hungry topology. It is compared with the results of traditional pencil-and-paper floorplanning considerations. Curve *Ideal* computes wire length based on the ideal formula given above, which only considers the number of hops crossed by a wire. Curve *Floorplan-aware* updates the previous formula with the knowledge of the asymmetric tile size and of switch placement. The ideal analysis largely overestimates the amount of wiring needed for the 4-hypercube. Floorplan awareness allows to account for specific floorplanning techniques that optimize wiring of a given topology, and therefore leads to more conservative estimations of the wiring overhead. However, this is still far away from real-life, where the post-layout report of total wire length gives only a 10% overhead of the 4-hypercube wiring with respect to 2D mesh one and a 43% with respect to the 2-ary 2-mesh. This is because switch-to-switch and switch-to-network interface wiring only accounts for a relatively small percentage of total wiring, ranging from 7% for the 2-ary 2-mesh to 26% for the 4-hypercube. This explains the relatively small total wire length difference between the different topologies. This scenario plays in favor of engineering performance-optimized interswitch links with a possibly minor impact on topology cost metrics.

3. LINK DESIGN TECHNIQUES

Three fundamental link inference techniques were studied in this work and an associated topology was laid out for each of them. When moving from one technique to another, the objective was to speed up a topology by speeding up its links. Therefore, our primary design objective was high-performance. In our first round of topology implementations we forced the inference of unrepeated links. In essence, we prevented the backend tools to instantiate buffers or inverters along the link. Therefore, the only degree of freedom for

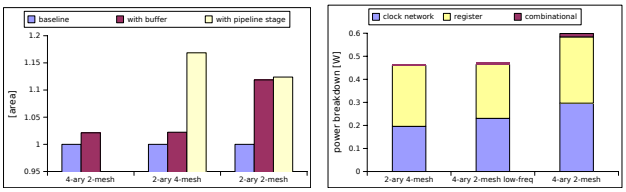
such tools was to prevent timing violations on the links by inferring a driver of suitable driving strength in the switch output ports. The first performance boosting strategy was to allow repeater insertion along the link. While an increase of instantiated buffers can be expected, the lower driving strength of switch output gates partially offset the added cost. In any case, a significant cut down on link delay is expected. Finally, we applied link pipelining to break long timing paths across interconnects. When implementing such as technique, flow control issues need to be considered. In our architecture, a backward propagating *stall* signal has to be retimed as well for each link. As a result, our pipeline stages are not simple registers but true retiming and flow control stages, consisting of 2 slot buffers and a control logic. This prevents from oversizing input buffers of downstream switches to avoid data loss.

4. EXPERIMENTAL RESULTS

For all cases, we had to impose a target performance. Since we are heading for high-performance, our goal was to materialize, after topology place-and-route, the maximum speed achievable by the slowest NoC module (characterized in isolation with post-layout timing analysis). In our architecture, the critical module turns out to be always the switch with the highest radix in the topology. The resulting speed upper bound, which ignores the effect of interswitch links, is reported in the first two rows in Table 1. For the slowest switch of each topology in isolation, the gap between post-synthesis and post-P&R speed ranges between 12 to 21%. The 2-ary 2-mesh exhibits a higher post-synthesis value due to the larger max. switch radix and a more severe post-P&R degradation due to the larger switch area compared with non-concentrated topologies. When we consider timing closure for the entire topologies, then the degradation associated with interswitch link routing and with their synthesis techniques becomes apparent. Let us consider repeaterless links first (Table 1, third row). While 2D mesh and 4-hypercube had the same performance upper-bound (since they have switches with the same maximum radix), the more challenging routing of the 4-hypercube gave rise to a 39% degradation of the critical path, while routing of the 2D mesh turn out to be less critical. The 2-ary 2-mesh has some links longer than 4mm, therefore interswitch routing has an even more relevant impact. This also hides the effect of the higher switch radix, which is not the ultimate responsible for the slower operating frequency. For all topologies, the critical path goes through the network links. Although post-layout effects of interswitch wires paint a dismal picture of topologies using long wires (because of the use of more dimensions or of the switch separation in the layout), designers can optimize wires to overcome these large delays. Activating repeater-insertion during topology synthesis enables the speedups illustrated by the fourth row of Table 1, denoting the best critical delay at which timing closure was achieved. Results are quite heterogeneous. The 2D mesh further benefits from repeaters and achieves a 6% speedup of its critical path, which results 1.19ns. This value is quite close to the performance upper-bound (1.12ns), thus indicating that while links are still critical in this topology, their length is such that their degradation of topology performance can be made irrelevant. The opposite holds for the 4-hypercube, where repeater insertion did not surprisingly provide any performance improvement. The reason for this lies in the fact that horizontal routing channels (see Fig.1(b)) were sized conservatively small, approximately 2.5x the switch side. For this reason, switch-to-switch links sometimes end up finding another switch on their way. This is a placement constraint for the buffers which prevents their insertion with ideal spacing. The result is that link performance does not improve, indicating that buffer insertion should not be taken for granted in NoC design, but should be carefully engineered to materialize its expected advantages. In practice, widening the routing channel ideally to 4x the switch side would solve the problem but would also lead to a large floorplan area overhead. Finally, no such constraints exist for the 2-ary 2-mesh (its connectivity pattern is trivial), which improves its baseline performance by 10%. Further critical path improvements were expected from link pipelining. When applying this technique, our objective was to materialize the same critical delay of the 2D mesh (with buffers) even for the 4-hypercube and the 2-ary 2-mesh. Interestingly, we noticed that link pipelining was effective even for the 4-hypercube, regardless of its under-sized routing channels. In fact, the target critical delay of 1.19ns was achieved. This result clearly indicates the lower sensitivity of link performance to non-ideal pipeline stage placement compared with non-ideal repeater spacing. Hence, link pipelining proves more robust for area-optimized floorplans with more challenging placements. Pipelining was effective also for the 2-ary 2-mesh, however the upper bound for its performance is the post-layout critical delay of its high-radix switches (1.4ns), far worse than the 2D mesh delay of 1.19ns.

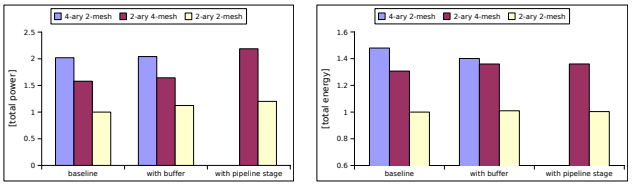
In order to assess the implementation cost for link performance boosting techniques, we illustrate area reports in Fig.3(a). Results are grouped by topology and normalized to the baseline implementation of each topology. Buffer insertion is quite cheap for the 2D mesh and the 4-hypercube, while generates a significant area overhead for the 2-ary 2-mesh. This is due to the backend tools, that have dealt with the performance maximization of long links by dramatically increasing the number of buffering gates. As explained later on, this implies also a relevant power cost. However, when pipeline stages are inferred, the additional area overhead of the 2-ary 2-mesh is marginal. This due to the fact that by inserting pipeline stages the tool was able to remove an equal amount of buffering area, so that the two contributions offset each other. This does not hold for the 4-hypercube, which raises its area by 14% when pipelining is used. Not only the area overhead of the pipeline stages is incurred, but many buffers are kept in the links since the frequency boost is significant when moving from buffering to pipelining. The trend of leakage power fully reflects that of area overhead and is therefore omitted for lack of space.

Although there is a price to pay to boost link performance in terms of area and power, the speedup in job completion can be exploited to cut down on total energy of the on-chip network. However, this energy saving materializes only if the gain in execution time outweighs the power overhead. This section sheds light on this aspect. Moreover, it is also investigated whether the different link synthesis techniques can change the relative energy ratios between the different topologies. Experiments were carried out with a parallel synthetic benchmark consisting of one producer task, 14 worker tasks and 1 consumer task. Every task is mapped on a different computation tile. The producer task reads in data units from the I/O interface of the chip and distributes it to the worker tasks. Output data from each worker tile is then collected by a consumer tile, which writes them back to the I/O interface. During computation, all-to-all accesses are generated to account for a cooperative computation process. Transactions are generated in compliance with the OCP protocol (burst accesses) by programmable OCP traffic generators [10]. We assume loose synchronization between the cores. Traffic generation rates were set to avoid system bottlenecks (like the I/O) which would stress other design issues other than network bandwidth and link performance. Real elapsed time results for



(a) Area (b) Clock tree power

Figure 3: Normalized area and CT power impact



(a) Total Power (b) Total Energy

Figure 4: Link synthesis techniques results

the baseline repeater-less topologies along with their respective buffered and pipelined variants are reported in Figure 2(a). For each variant, the maximum achievable post-P&R speed is applied (see Table 1). Although the 4-hypercube reports a lower clock cycle count to complete the benchmark, its execution time is always higher in the baseline topologies and in the repeated ones since its running speed is much lower than the 2D mesh. A 16-tile system is too small for an hypercube to take full advantage of its better properties (bisection bandwidth, diameter) and thus to offset the speed degradation associated with its relatively longer links. The situation is even worse for the 2-ary 2-mesh, which is worse than the other two topologies in the baseline and buffered variants both in execution cycles (because of poor bisection bandwidth) and in running speed (overly long links). However, when the 4-hypercube can operate at the same speed of the 2D mesh thanks to link pipelining, than it becomes the most performance-efficient solution. The interesting thing is that link pipelining not necessarily adverses total performance. In this experiment, it enables a significant topology speedup and provides additional buffering to the topology itself, which is effective to handle a bandwidth-sensitive traffic pattern. The high switch radix of the 2-ary 2-mesh causes a saturation effect to the performance gain that this topology can achieve by means of link pipelining. The power cost incurred to boost link and hence topology performance is illustrated in Figure 4(a). Power is affected by the operating frequency of each topology. If we compare topologies with boosted links with their baseline variants we observe that buffers do not cost a lot in terms of power. This holds for the 2D mesh and for the 4-hypercube, which is in line with the hardly relevant share of interswitch links over total wire length (as explained in Section 2). Power overhead of the 2-ary 2-mesh is an exception to this, due to the large power cost of driving a long link effectively. When it comes to link pipelining, the power cost abruptly increases, especially for the 4-hypercube, while the overhead for the 2-ary 2-mesh increases more smoothly. Fortunately, this is also the scenario where the 4-hypercube gains more in terms of performance. When we compare topologies with each other, we observe that the topology of choice when low-power is the primary design goal is the 2-ary 2-mesh. The lower power of the 4-hypercube with respect to the 2D mesh mostly derives from its lower speed, although this is not the only explanation. This is confirmed also by the marginal power overhead of the 4-hypercube when it can run at the same speed of the 2D mesh (see buffered 2D mesh vs pipelined 4-hypercube in Figure 4(a)). This is counterintuitive, since the hypercube has many more buffering resources than the 2D mesh. The answer has be sought in the power breakdown. This is reported in Figure 3(b) for the baseline variants of the two topologies. The 2D mesh was re-synthesized and analyzed both at full speed (rightmost column) and at the same speed of the 4-hypercube for a fair comparison. In all cases, it is evident that the clock tree has a relatively lower impact than register power in the 4-hypercube, while in the 2D mesh it weighs more. As a consequence, when the same speed is inferred, the two topologies have surprisingly the same power. While register power obviously increases for the 4-hypercube, the clock tree is cheaper, and this explains the result. The reason lies in the fact that while the 2D mesh has heterogeneous switches, the 4-hypercube has all switches with exactly the same radix. Hence, the clock tree is inherently more balanced and thus easier to synthesize while meeting skew constraints. By combining the performance results of Figure 2(a) with the power reports of Figure 4(a)), we get the energy results of Fig.4(b). Overall, buffering the links of a 2D mesh is always an energy-efficient strategy. On the contrary, 4-hypercube and 2-ary 2-mesh show minor energy variations when moving from one scenario to the other. This indicates that performance improvements have been achieved at the cost of a proportional power overhead.

5. DISCUSSION AND CONCLUSIONS

The critical path of NoC architectures is moving once again to the interswitch links, thus making the synthesis technique of these links highly critical for overall topology speed. This paper explores how repeater-less, repeated and pipelined links affect the critical path and the cost of a family of k-ary n-mesh topologies. When it comes to the achievable speedup, we noticed that repeater insertion is quite sensitive to floorplanning constraints, so that a non-ideal repeater spacing might result in no performance boost. Widening routing channels is the obvious but costly workaround for this. In contrast, pipeline stage positioning is more flexible and the expected performance gain is always materialized in practice. Moreover, we noticed that buffers do not cost much when applied to relatively short links (around 2.5 mm in 65nm technology), while on longer links the backend tool attempts a performance optimization at a dramatic power (buffering) cost. In all other cases, the steep increase in area arises only when pipeline stages are inferred. Given this, we found it always convenient to infer repeated interswitch wires in a 2D mesh, since the performance speedup effect is prevailing. As regards the 4-hypercube, it really needs large scale systems to take advantage of its bisection bandwidth and low diameter. However, already for 16 tile systems, when it can be operated at the same speed of the 2D mesh (by means of link pipelining) it provides much better performance with equal if not lower energy. For the 2-ary 2-mesh, the high switch radix poses a saturation limit to the speedups that link optimization techniques can provide. It is anyway the technique of choice when low-power is the primary design goal. In any case, whenever the topology has already been selected, in general boosting link performance pays off in terms of total topology performance while only minor or even no energy overhead is incurred. When however the connectivity pattern becomes intricate, the validity of these considerations becomes very sensitive to layout constraints.

6. REFERENCES

[1] U.Y. Ogras, R. Marculescu, H.G. Lee, C. Puru, D. Marculescu, M. Kaufman, N. Peter, "Challenges and Promising Results in NoC Prototyping Using FPGAs". IEEE Micro Special Issue on Interconnects for Multi-Core Chips, 27(5):86-95, 2007.

[2] A. Pullini, F. Angiolini, S. Murali, D. Atienza, G. De Micheli, L. Benini, "Bringing NoCs to 65 nm". IEEE Micro Special Issue on Interconnects for Multi-Core Chips, 27(5):75–78, 2007.

[3] L. Zhong, N.K. Jha, "Interconnect-aware low-power high-level synthesis". IEEE Transaction on Computer-Aided Design of Integrated Circuits and Systems, 24(3):336–351, 2005.

[4] F. Gilabert, S. Medardoni, D. Bertozzi, L. Benini, M. E. Gómez, P. López, J. Duato; "Exploring High-Dimensional Topologies for NoC Design Through an Integrated Analysis and Synthesis Framework", Int. Network-on-Chip Symp., pp.107-116, 2008.

[5] D. Ludovici, F. Gilabert, S. Medardoni, C. Gómez, M. E. Gómez, P. López, D. Bertozzi, G. N. Gaydadjiev; "Assessing Fat-Tree topologies for Regular Network-on-Chip Design under Nanoscale Technology Constraints", Proc. of Design, Automation and Test in Europe (DATE), to appear, 2009.

[6] R. Ho, K.W. Mai, M.A Horowitz, "Managing wire scaling: a circuit perspective". Proc. of the IEEE 2003 International Interconnect Technology Conference, pp.177–179, 2003.

[7] S. Medardoni, D. Bertozzi, L. Benini, E. Macii, "Control and datapath decoupling in the design of a NoC switch: area, power and performance implications". Proc. of International Symposium on System-on-Chip, pp.1–4, 2007.

[8] D. Sylvester and K. Keutzer, "Getting to the bottom of deep sub-micron II: A global paradigm". Proc. of IEEE International Symposium on Physical Design, pp.193–200, 1999.

[9] I. Hatirnaz, S. Badel, N. Pazos, Y. Leblebici, S. Murali, D. Atienza, G. De Micheli. "Early wire characterization for predictable network-on-chip global interconnects", Proc of SLIP'07, pp.57–64, 2007.

[10] S. Mahadevan, F. Angiolini, M. Storoaard, R.G. Olsen, J. Sparso, J. Madsen, "Network tra¡c generator model for fast network-on-chip simulation". Proc. of Design, Automation and Test in Europe (DATE), pp.780–785, 2005.

[11] S. Stergiou, F. Angiolini, S. Carta, L. Ra↑o, D. Bertozzi, G. De Micheli, "XPipes Lite: a Synthesis Oriented Design Library for Networks on Chips". Proc. of DATE, pp.1188–1193, 2005.

[12] F. Gilabert, D. Ludovici, S. Medardoni, D. Bertozzi, L. Benini, G. N. Gaydadjiev, "Designing Regular Network-on-Chip Topologies under Technology, Architecture and Software Constraints". Proc. of IEEE MuCoCos, in press, Fukuoka, Japan, 2009.